Под редакцией
С. В. Симоновича

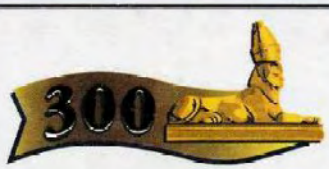
ПИТЕР

ИНФОРМАТИКА

БАЗОВЫЙ КУРС

2-е издание

УЧЕБНИК ДЛЯ ВУЗОВ



для студентов технических специальностей и преподавателей высших учебных заведений

фундаментальный курс, охватывающий все основные разделы информатики

Книга представлена отдельными главами

УЧЕБНИК ДЛЯ ВУЗОВ

Под редакцией
С. В. Симоновича

ИНФОРМАТИКА

БАЗОВЫЙ КУРС

2-е издание

Рекомендовано Министерством образования Российской Федерации
в качестве учебного пособия для студентов
высших технических учебных заведений

300.piter.com

Издательская программа

300 лучших учебников для высшей школы
в честь 300-летия Санкт-Петербурга

осуществляется при поддержке Министерства образования РФ

Книга представлена отдельными главами

ПИТЕР®

Москва · Санкт-Петербург · Нижний Новгород · Воронеж
Новосибирск · Ростов-на-Дону · Екатеринбург · Самара
Киев · Харьков · Минск · Новосибирск
2005

ББК 32.973.233я7
УДК 681.3(075)
С37

Рецензенты:

Кафедра САПР Московского государственного технического университета им. Н. Э. Баумана
Калин С. В., генеральный директор ЗАО «Открытые технологии '98»

С37 **Информатика. Базовый курс. 2-е издание** / Под ред. С. В. Симоновича. — СПб.: Питер, 2005. — 640 с.: ил.

ISBN 5-94723-752-0

В учебнике рассмотрены основные категории аппаратных и программных средств вычислительной техники. Указаны базовые принципы построения архитектур вычислительных систем. Обеспечено методическое обоснование процессов взаимодействия информации, данных и методов. Приведены эффективные приемы работы с распространенными программными продуктами. Рассмотрены основные средства, приемы и методы программирования.

Книга предназначена для студентов технических вузов, изучающих информационные технологии в рамках дисциплины «Информатика», для преподавательского состава, для слушателей военных учебных заведений, учреждений системы повышения квалификации и для лиц, изучающих средства вычислительной техники самостоятельно.

Рекомендовано Министерством образования Российской Федерации в качестве учебного пособия для студентов высших технических учебных заведений.

ББК 32.973.233я7
УДК 681.3(075)

Книга представлена отдельными главами

Информация, содержащаяся в данной книге, получена из источников, рассматриваемых издательством как надежные. Тем не менее, имея в виду возможные человеческие или технические ошибки, издательство не может гарантировать абсолютную точность и полноту приводимых сведений и не несет ответственности за возможные ошибки, связанные с использованием книги.

ISBN 5-94723-752-0

© С. В. Симонович, Г. А. Евсеев, В. И. Мураховский, С. И. Бобровский, 2003
© ЗАО Издательский дом «Питер», 2005

Содержание

Введение

Коренное отличие информатики от других технических дисциплин, изучаемых в высшей школе, состоит в том, что ее предмет изучения меняется ускоренными темпами. Сегодня количество компьютеров в мире превышает 500 миллионов единиц, при этом каждая вычислительная система по-своему уникальна. Найти две системы с одинаковыми аппаратными и программными конфигурациями весьма сложно, и потому для эффективной эксплуатации вычислительной техники от специалистов требуется достаточно широкий уровень знаний и практических навыков.

Вместе с тем, в количественном отношении темп численного роста вычислительных систем заметно превышает темп подготовки специалистов, способных эффективно работать с ними. При этом в среднем один раз в полтора года удваиваются основные технические параметры аппаратных средств, один раз в два-три года меняются поколения программного обеспечения и один раз в пять-семь лет меняется база стандартов, интерфейсов и протоколов.

Таким образом, кардинальным отличием информатики от других технических дисциплин является тот факт, что ее предметная область изменяется чрезвычайно динамично. Все, кто причастен к преподаванию информатики в высшей школе, хорошо знают, как часто приходится менять содержание учебных планов, рабочих программ, учебно-методической документации. Далеко не всегда удается обеспечить соответствие материально-технической базы учебного процесса текущему состоянию предметной области. И даже своевременное реагирование на научно-технические достижения не всегда позволяет обеспечить уровень знаний и навыков выпускника, адекватный потребностям сферы материального производства и коммерческого рынка, — настолько динамичны процессы в области информационных технологий.

Ныне информатика сталкивается с парадоксальным фактом. Ее основная задача состоит в преодолении общечеловеческого кризисного явления, называемого «информационным бумом», путем внедрения средств и методов, автоматизирующих операции с данными. Однако даже в собственной предметной области информатика

испытывает такой информационный бум, какого не знает ни одна другая область человеческой деятельности. Например, мировой ассортимент изданий, имеющих прямое отношение к информатике (не считая периодических и электронных), составляет порядка десяти тысяч томов в год и полностью обновляется не реже, чем раз в два года.

Анализируя вышеуказанные особенности информатики, авторы данного пособия приходят к выводу, что для преподавания информатики в сложившихся условиях необходимо расширенное взаимодействие между учебными программами общетехнических и специальных дисциплин и учебной программой курса информатики. Основные принципы, вытекающие из такого подхода, включают непрерывность и системность образования, а также раннюю профессиональную ориентацию.

Непрерывность образования. Практические приемы работы со средствами вычислительной техники закрепляются не только при изучении информатики, но и в течение всего периода обучения. Они используются при проведении учебных занятий по самым разным дисциплинам.

Системность образования. В едином методическом подходе, основанном на системе *задача — средство — методы — приемы*, происходит перекрестное взаимодействие изучаемых дисциплин. Конкретная дисциплина поставляет комплекс *задача — методы*, а информатика обеспечивает комплекс *средства — приемы*.

Ранняя профессиональная ориентация. В системе высшего технического образования действует многоуровневая иерархическая система, основанная на том, что знания студента по общетехническим дисциплинам, как правило, реализуются в практические навыки опосредованно, то есть через дисциплины специального цикла, базирующиеся на общетехнических. Информатика — одна из немногих общетехнических дисциплин, развивающая такие практические навыки, которые востребуются напрямую и немедленно, сразу после включения молодого специалиста в профессиональную деятельность.

Свою работу над книгой авторы подчинили реализации указанных принципов. Этому подчинены структура и содержание пособия. В целом книга состоит из двадцати глав, содержащих достаточно полные сведения о современном состоянии аппаратных и программных средств вычислительной техники.

Главы 1, 2, 8, 15 являются теоретическими и обеспечивают единую методическую базу как для изучения информатики, так и для взаимодействия различных учебных дисциплин на платформе информатики.

Главы 9–14, 16, 18 представляют единую технологическую базу для взаимодействия информатики и других предметных дисциплин. Средства, рассмотренные здесь, могут быть использованы при подготовке домашних заданий, контрольных, курсовых и дипломных работ, при обработке результатов экспериментов, сборе исходной информации для самостоятельных исследований, при выполнении графических работ, математическом моделировании физических и технических процессов и при математическом обосновании разрабатываемых проектов.

Главы 3–7, 10, 12, 13, 16, 17, 19, 20 служат тем же задачам, но являются дополнительным средством ранней профессиональной ориентации. Сведения и навыки, полученные в ходе их изучения, могут быть востребованы немедленно после включения выпускника в практическую деятельность. Эти разделы позволяют обеспечить общую уверенность студента в востребованности его знаний по окончании учебного заведения, независимо от обстоятельств и особенностей конкретного трудоустройства.

Главы, имеющие теоретическое и методообразующее содержание, завершаются списком контрольных вопросов, которые могут обсуждаться на лекционных и семинарских занятиях. Главы, имеющие практическое содержание, завершаются упражнениями и исследовательскими работами. Предполагается, что практические упражнения носят инструктивно-методический характер и выполняются под руководством преподавателя (лаборанта), а исследовательские работы имеют творческий характер и комплексное содержание. Они предназначены для самостоятельной работы и предполагают подготовку итогового отчета. Различие между этими видами занятий отражено в балансе отводимого на них времени.

Исходя из структуры и содержания книги, авторы рассчитывают на то, что она будет полезна следующим категориям читателей:

- студентам технических специальностей вузов, изучающим информатику как самостоятельную дисциплину;
- преподавательскому составу, осуществляющему теоретическую и практическую подготовку студентов по дисциплине «Информатика»;
- преподавателям иных дисциплин, использующим персональные компьютеры в качестве технического средства обучения и (или) средства подготовки учебно-методических материалов (бумажных и электронных) по своей предметной области;
- лицам, самостоятельно изучающим или осваивающим аппаратные и программные средства вычислительной техники.

1.1. Информация в материальном мире

Сигналы и данные

Мы живем в материальном мире. Все, что нас окружает и с чем мы сталкиваемся ежедневно, относится либо к *физическим телам*, либо к *физическим полям*. Из курса физики мы знаем, что состояния абсолютного покоя не существует и физические объекты находятся в состоянии непрерывного движении и изменения, которое сопровождается обменом энергией и ее переходом из одной формы в другую.

Все виды *энергообмена* сопровождаются появлением сигналов, то есть все сигналы имеют в своей основе материальную энергетическую природу. При взаимодействии сигналов с физическими телами в последних возникают определенные изменения свойств — это явление называется *регистрацией сигналов*. Такие изменения можно наблюдать, измерять или фиксировать иными способами — при этом возникают и регистрируются новые сигналы, то есть образуются данные.

Данные — это зарегистрированные сигналы.

Данные и методы

Обратим внимание на то, что данные несут в себе *информацию* о событиях, произошедших в материальном мире, поскольку они являются регистрацией сигналов, возникших в результате этих событий. Однако данные не тождественны информации. Наблюдая излучения далеких звезд, человек получает определенный поток данных, но станут ли эти данные информацией, зависит еще от очень многих обстоятельств. Рассмотрим ряд примеров.

Наблюдая за состязаниями бегунов, мы с помощью механического секундомера регистрируем начальное и конечное положение стрелки прибора. В итоге мы замеряем величину ее перемещения за время забега — это регистрация данных. Однако информацию о времени преодоления дистанции мы пока не получаем. Для того чтобы данные о перемещении стрелки дали информацию о времени забега, необ-

ходимо наличие *метода* пересчета одной физической величины в другую. Надо знать цену деления шкалы секундомера (или знать метод ее определения) и надо также знать, как умножается цена деления прибора на величину перемещения, то есть надо еще обладать математическим методом умножения.

Если вместо механического секундомера используется электронный, суть дела не меняется. Вместо регистрации перемещения стрелки происходит регистрация количества тактов колебаний, произошедших в электронной системе за время измерения. Даже если секундомер непосредственно отображает время в секундах и нам не нужен метод пересчета, то метод преобразования данных все равно присутствует — он реализован специальными электронными компонентами и работает автоматически, без нашего участия.

Прослушивая передачу радиостанции на незнакомом языке, мы получаем данные, но не получаем информацию в связи с тем, что не владеем методом преобразования данных в известные нам понятия. Если эти данные записать на лист бумаги или на магнитную ленту, изменится форма их представления, произойдет новая регистрация и, соответственно, образуются новые данные. Такое преобразование можно использовать, чтобы все-таки извлечь информацию из данных путем подбора метода, адекватного их новой форме. Для обработки данных, записанных на листе бумаги, адекватным может быть метод перевода со словарем, а для обработки данных, записанных на магнитной ленте, можно пригласить переводчика, обладающего своими методами перевода, основанными на знаниях, полученных в результате обучения или предшествующего опыта.

Если в нашем примере заменить радиопередачу телевизионной трансляцией, ведущейся на незнакомом языке, то мы увидим, что наряду с данными мы все-таки получаем определенную (хотя и не полную) информацию. Это связано с тем, что люди, не имеющие дефектов зрения, априорно владеют адекватным методом восприятия данных, передаваемых электромагнитным сигналом в полосе частот видимого спектра с интенсивностью, превышающей порог чувствительности глаза. В таких случаях говорят, что *метод известен по контексту*, то есть данные, составляющие информацию, имеют свойства, однозначно определяющие адекватный метод получения этой информации. (Для сравнения скажем, что слепому «телезрителю» контекстный метод неизвестен и он оказывается в положении радиослушателя, пример с которым был рассмотрен выше.)

Понятие об информации

Несмотря на то что с понятием информации мы сталкиваемся ежедневно, строгого и общепризнанного ее определения до сих пор не существует, поэтому вместо определения обычно используют *понятие об информации*. Понятия, в отличие от определений, не даются однозначно, а вводятся на примерах, причем каждая научная дисциплина делает это по-своему, выделяя в качестве основных компонентов те, которые наилучшим образом соответствуют ее предмету и задачам. При этом типична ситуация, когда понятие об информации, введенное в рамках одной научной дисциплины, может опровергаться конкретными примерами и фактами, полученными в рамках другой науки. Например, представление об информации как о совокупности дан-

ных, повышающих уровень знаний об объективной реальности окружающего мира, характерное для естественных наук, может быть опровергнуто в рамках социальных наук. Нередки также случаи, когда исходные компоненты, составляющие понятие информации, подменяют свойствами информационных объектов, например, когда понятие информации вводят как совокупность данных, которые «могут быть усвоены и преобразованы в знания».

Для информатики как технической науки понятие информации не может основываться на таких антропоцентрических понятиях, как *знание*, и не может опираться только на объективность фактов и свидетельств. Средства вычислительной техники обладают способностью обрабатывать информацию автоматически, без участия человека, и ни о каком знании или незнании здесь речь идти не может. Эти средства могут работать с искусственной, абстрактной и даже с ложной информацией, не имеющей объективного отражения ни в природе, ни в обществе.

В этой работе мы даем новое определение информации, основанное на ранее продемонстрированном факте взаимодействия данных и методов в момент ее образования.

Информация — это продукт взаимодействия данных и адекватных им методов.

Поскольку в такой форме определение информации дается впервые, читатель приглашается для его всесторонней проверки в рамках других известных ему научных дисциплин, а мы рассмотрим пример, в свое время использованный Норбертом Винером для того, чтобы показать, как информация отдельных членов популяции становится информацией общества.

> *Допустим, я нахожусь в лесах вдвоем со смышленым дикарем, который не может говорить на моем языке и на языке которого я тоже не могу говорить. Даже без какого-либо условного языка знаков, известного нам обоим, я могу многое узнать от него. Мне нужно лишь быть особо внимательным в те моменты, когда он обнаруживает признаки волнения или интереса. Тогда я должен посмотреть вокруг, особенно в направлении его взгляда, и запомнить все, что я увижу и услышу. Не пройдет много времени, как я открою, какие предметы представляются важными для него, — не потому, что он сообщил мне о них словами, но потому, что я сам их заметил. Иначе говоря, сигнал, лишенный внутреннего содержания, может приобрести для моего спутника смысл по тому, что наблюдает он в данный момент, и может приобрести для меня смысл по тому, что наблюдаю я в данный момент. Способность дикаря замечать моменты моего особенно активного внимания сама по себе образует язык, возможности которого столь же разнообразны, как и диапазон впечатлений, доступных нам обоим.*
>
> *Н. Винер. Кибернетика*

Анализируя этот пример, мы видим, что здесь речь идет о данных и методах. Прежде всего, здесь автор прямо говорит о целой группе методов, связанных с наблюдением и анализом, и даже приводит вариант конкретного алгоритма, адекватного рамкам его гипотетического эксперимента (*посмотреть, запомнить, открыть...*). Автор неоднократно подчеркивает требование адекватности метода (дикарь должен быть *смышленым*, а наблюдатель должен быть *особо внимательным*), без которого информация может и не образоваться.

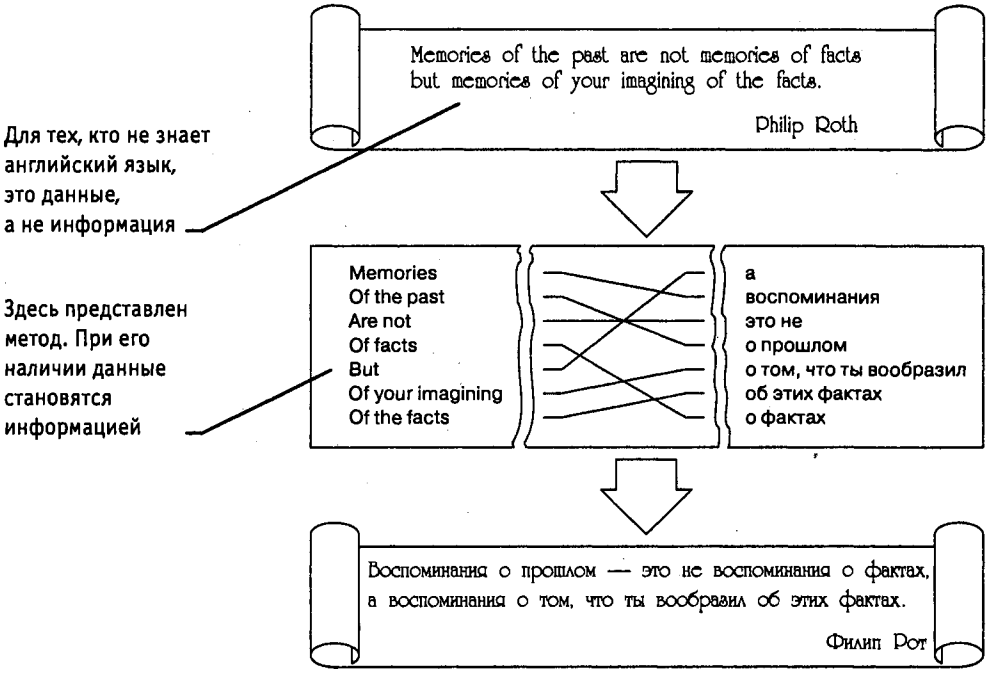


Рис. 1.1. Связь между данными и информацией

Диалектическое единство данных и методов в информационном процессе

Рассмотрим данное выше определение информации и обратим внимание на следующие обстоятельства.

1. *Динамический характер информации.* Информация не является статичным объектом — она динамически меняется и существует только в момент взаимодействия данных и методов. Все прочее время она пребывает в состоянии данных. Таким образом, информация существует только в момент протекания *информационного процесса*. Все остальное время она содержится в виде данных.
2. *Требование адекватности методов.* Одни и те же данные могут в момент потребления поставлять разную информацию в зависимости от степени адекватности взаимодействующих с ними методов. Например, для человека, не владеющего китайским языком, письмо, полученное из Пекина, дает только ту информацию, которую можно получить методом наблюдения (количество страниц, цвет и сорт бумаги, наличие незнакомых символов и т. п.). Все это информация, но это не вся информация, заключенная в письме. Использование более адекватных методов даст иную информацию.
3. *Диалектический характер взаимодействия данных и методов.* Обратим внимание на то, что данные являются *объективными*, поскольку это результат регистрации объективно существовавших сигналов, вызванных изменениями в материальных

телах или полях. В то же время, методы являются *субъективными.* В основе искусственных методов лежат алгоритмы (упорядоченные последовательности команд), составленные и подготовленные людьми (субъектами). В основе естественных методов лежат биологические свойства субъектов информационного процесса. Таким образом, *информация возникает и существует в момент диалектического взаимодействия объективных данных и субъективных методов.*

Такой дуализм известен своими проявлениями во многих науках. Так, например, в основе важнейшего вопроса философии о первичности материалистического и идеалистического подходов к теории познания лежит не что иное, как двойственный характер информационного процесса. В обоснованиях обоих подходов нетрудно обнаружить упор либо на объективность данных, либо на субъективность методов. Подход к информации как к объекту особой природы, возникающему в результате диалектического взаимодействия объективных данных с субъективными методами, позволяет во многих случаях снять противоречия, возникающие в философских обоснованиях ряда научных теорий и гипотез.

Свойства информации

Итак, информация является динамическим объектом, образующимся в момент взаимодействия объективных данных и субъективных методов. Как и всякий объект, она обладает свойствами (объекты различимы по своим свойствам). Характерной особенностью информации, отличающей ее от других объектов природы и общества, является отмеченный выше дуализм: на свойства информации влияют как свойства данных, составляющих ее содержательную часть, так и свойства методов, взаимодействующих с данными в ходе информационного процесса. По окончании процесса свойства информации переносятся на свойства новых данных, то есть свойства методов могут переходить на свойства данных.

Можно привести немало разнообразных свойств информации. Каждая научная дисциплина рассматривает те свойства, которые ей наиболее важны. С точки зрения информатики наиболее важными представляются следующие свойства: объективность, полнота, достоверность, адекватность, доступность и актуальность информации.

Объективность и субъективность информации. Понятие объективности информации является относительным. Это понятно, если учесть, что методы являются субъективными. Более объективной принято считать ту информацию, в которую методы вносят меньший субъективный элемент. Так, например, принято считать, что в результате наблюдения фотоснимка природного объекта или явления образуется более объективная информация, чем в результате наблюдения рисунка того же объекта, выполненного человеком. В ходе информационного процесса степень объективности информации всегда понижается. Это свойство учитывают, например, в правовых дисциплинах, где по-разному обрабатываются показания лиц, непосредственно наблюдавших события или получивших информацию косвенным путем (посредством умозаключений или со слов третьих лиц). В не меньшей степени объективность информации учитывают в исторических дисциплинах. Одни и те же события, зафиксированные в исторических документах разных стран и народов, выглядят совершенно по-разному. У историков имеются свои методы для тестирова-

ния объективности исторических данных и создания новых, более достоверных данных путем сопоставления, фильтрации и селекции исходных данных. Обратим внимание на то, что здесь речь идет не о повышении объективности данных, а о повышении их достоверности (это совсем другое свойство).

Полнота информации. Полнота информации во многом характеризует *качество информации* и определяет *достаточность* данных для принятия решений или для создания новых данных на основе имеющихся. Чем полнее данные, тем шире диапазон методов, которые можно использовать, тем проще подобрать метод, вносящий минимум погрешностей в ход информационного процесса.

Достоверность информации. Данные возникают в момент регистрации сигналов, но не все сигналы являются «полезными» — всегда присутствует какой-то уровень посторонних сигналов, в результате чего полезные данные сопровождаются определенным уровнем «информационного шума». Если полезный сигнал зарегистрирован более четко, чем посторонние сигналы, достоверность информации может быть более высокой. При увеличении уровня шумов достоверность информации снижается. В этом случае для передачи того же количества информации требуется использовать либо больше данных, либо более сложные методы.

Адекватность информации — это степень соответствия реальному объективному состоянию дела. Неадекватная информация может образовываться при создании новой информации на основе неполных или недостоверных данных. Однако и полные, и достоверные данные могут приводить к созданию неадекватной информации в случае применения к ним неадекватных методов.

Доступность информации — мера возможности получить ту или иную информацию. На степень доступности информации влияют одновременно как доступность данных, так и доступность адекватных методов для их интерпретации. Отсутствие доступа к данным или отсутствие адекватных методов обработки данных приводят к одинаковому результату: информация оказывается недоступной. Отсутствие адекватных методов для работы с данными во многих случаях приводит к применению неадекватных методов, в результате чего образуется неполная, неадекватная или недостоверная информация.

Актуальность информации — это степень соответствия информации текущему моменту времени. Нередко с актуальностью, как и с полнотой, связывают коммерческую ценность информации. Поскольку информационные процессы растянуты во времени, то достоверная и адекватная, но устаревшая информация может приводить к ошибочным решениям. Необходимость поиска (или разработки) адекватного метода для работы с данными может приводить к такой задержке в получении информации, что она становится неактуальной и ненужной. На этом, в частности, основаны многие современные системы шифрования данных с *открытым ключом*. Лица, не владеющие ключом (методом) для чтения данных, могут заняться поиском ключа, поскольку алгоритм его работы доступен, но продолжительность этого поиска столь велика, что за время работы информация теряет актуальность и, соответственно, связанную с ней практическую ценность.

1.2. Данные

Носители данных

Данные — диалектическая составная часть информации. Они представляют собой зарегистрированные сигналы. При этом физический метод регистрации может быть любым: механическое перемещение физических тел, изменение их формы или параметров качества поверхности, изменение электрических, магнитных, оптических характеристик, химического состава и (или) характера химических связей, изменение состояния электронной системы и многое другое. В соответствии с методом регистрации данные могут храниться и транспортироваться на носителях различных видов.

Самым распространенным носителем данных, хотя и не самым экономичным, по-видимому, является бумага. На бумаге данные регистрируются путем изменения оптических характеристик ее поверхности. Изменение оптических свойств (изменение коэффициента отражения поверхности в определенном диапазоне длин волн) используется также в устройствах, осуществляющих запись лазерным лучом на пластмассовых носителях с отражающим покрытием (*CD-ROM*). В качестве носителей, использующих изменение магнитных свойств, можно назвать магнитные ленты и диски. Регистрация данных путем изменения химического состава поверхностных веществ носителя широко используется в фотографии. На биохимическом уровне происходит накопление и передача данных в живой природе.

Носители данных интересуют нас не сами по себе, а постольку, поскольку свойства информации весьма тесно связаны со свойствами ее носителей. Любой носитель можно характеризовать параметром *разрешающей способности* (количеством данных, записанных в принятой для носителя единице измерения) и *динамическим диапазоном* (логарифмическим отношением интенсивности амплитуд максимального и минимального регистрируемого сигналов). От этих свойств носителя нередко зависят такие свойства информации, как полнота, доступность и достоверность. Например, мы можем рассчитывать на то, что в базе данных, размещаемой на компакт-диске, проще обеспечить полноту информации, чем в аналогичной по назначению базе данных, размещенной на гибком магнитном диске, поскольку в первом случае плотность записи данных на единице длины дорожки намного выше. Для обычного потребителя доступность информации в книге заметно выше, чем той же информации на компакт-диске, поскольку не все потребители обладают необходимым оборудованием. И наконец, известно, что визуальный эффект от просмотра слайда в проекторе намного больше, чем от просмотра аналогичной иллюстрации, напечатанной на бумаге, поскольку диапазон яркостных сигналов в проходящем свете на два-три порядка больше, чем в отраженном.

Задача преобразования данных с целью смены носителя относится к одной из важнейших задач информатики. В структуре стоимости вычислительных систем устройства для ввода и вывода данных, работающие с носителями информации, составляют до половины стоимости аппаратных средств.

Операции с данными

В ходе информационного процесса данные преобразуются из одного вида в другой с помощью методов. Обработка данных включает в себя множество различных

операций. По мере развития научно-технического прогресса и общего усложнения связей в человеческом обществе трудозатраты на обработку данных неуклонно возрастают. Прежде всего это связано с постоянным усложнением условий управления производством и обществом. Второй фактор, также вызывающий общее увеличение объемов обрабатываемых данных, тоже связан с научно-техническим прогрессом, а именно с быстрыми темпами появления и внедрения новых носителей данных, средств их хранения и доставки.

В структуре возможных операций с данными можно выделить следующие основные:

- *сбор данных* — накопление информации с целью обеспечения достаточной полноты для принятия решений;
- *формализация данных* — приведение данных, поступающих из разных источников, к одинаковой форме, чтобы сделать их сопоставимыми между собой, то есть повысить их уровень доступности;
- *фильтрация данных* — отсеивание «лишних» данных, в которых нет необходимости для принятия решений; при этом должен уменьшаться уровень «шума», а достоверность и адекватность данных должны возрастать;
- *сортировка данных* — упорядочение данных по заданному признаку с целью удобства использования; повышает доступность информации;
- *архивация данных* — организация хранения данных в удобной и легкодоступной форме; служит для снижения экономических затрат по хранению данных и повышает общую надежность информационного процесса в целом;
- *защита данных* — комплекс мер, направленных на предотвращение утраты, воспроизведения и модификации данных;
- *транспортировка данных* — прием и передача (доставка и поставка) данных между удаленными участниками информационного процесса; при этом источник данных в информатике принято называть *сервером*, а потребителя — *клиентом*;
- *преобразование данных* — перевод данных из одной формы в другую или из одной структуры в другую. Преобразование данных часто связано с изменением типа носителя: например книги можно хранить в обычной бумажной форме, но можно использовать для этого и электронную форму, и микрофотопленку. Необходимость в многократном преобразовании данных возникает также при их транспортировке, особенно если она осуществляется средствами, не предназначенными для транспортировки данного вида данных. В качестве примера можно упомянуть, что для транспортировки цифровых потоков данных по каналам телефонных сетей (которые изначально были ориентированы только на передачу аналоговых сигналов в узком диапазоне частот) необходимо преобразование цифровых данных в некое подобие звуковых сигналов, чем и занимаются специальные устройства — *телефонные модемы*.

Приведенный здесь список типовых операций с данными далеко не полон. Миллионы людей во всем мире занимаются созданием, обработкой, преобразованием и транспортировкой данных, и на каждом рабочем месте выполняются свои специфи-

ческие операции, необходимые для управления социальными, экономическими, промышленными, научными и культурными процессами. Полный список возможных операций составить невозможно, да и не нужно. Сейчас нам важен другой вывод: *работа с информацией может иметь огромную трудоемкость и ее надо автоматизировать.*

Кодирование данных двоичным кодом

Для автоматизации работы с данными, относящимися к различным типам, очень важно унифицировать их форму представления — для этого обычно используется прием *кодирования*, то есть выражение данных одного типа через данные другого типа. Естественные человеческие *языки* — это не что иное, как системы кодирования понятий для выражения мыслей посредством речи. К языкам близко примыкают *азбуки* (системы кодирования компонентов языка с помощью графических символов). История знает интересные, хотя и безуспешные попытки создания «универсальных» языков и азбук. По-видимому, безуспешность попыток их внедрения связана с тем, что национальные и социальные образования естественным образом понимают, что изменение системы кодирования общественных данных непременно приводит к изменению общественных методов (то есть норм права и морали), а это может быть связано с социальными потрясениями.

Та же проблема универсального средства кодирования достаточно успешно реализуется в отдельных отраслях техники, науки и культуры. В качестве примеров можно привести систему записи математических выражений, телеграфную азбуку, морскую флажковую азбуку, систему Брайля для слепых и многое другое.

Своя система существует и в вычислительной технике — она называется *двоичным кодированием* и основана на представлении данных последовательностью всего двух знаков: 0 и 1. Эти знаки называются *двоичными цифрами*, по английски — *binary digit* или, сокращенно, *bit* (*бит*).

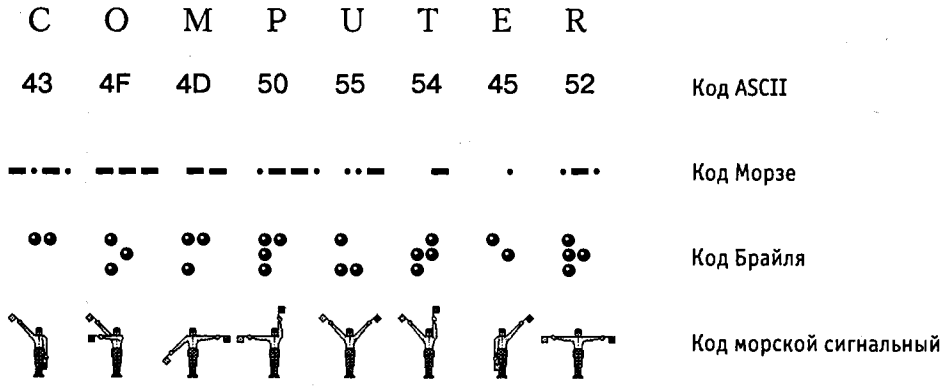


Рис. 1.2. Примеры различных систем кодирования

Одним битом могут быть выражены два понятия: 0 или 1 (*да* или *нет*, *черное* или *белое*, *истина* или *ложь* и т. п.). Если количество битов увеличить до двух, то уже можно выразить четыре различных понятия:

00 01 10 11

Тремя битами можно закодировать восемь различных значений:

000 001 010 011 100 101 110 111

Увеличивая на единицу количество разрядов в системе двоичного кодирования, мы увеличиваем в два раза количество значений, которое может быть выражено в данной системе, то есть общая формула имеет вид:

$$N = 2^m, \quad \text{где:}$$

N — количество независимых кодируемых значений;

m — разрядность двоичного кодирования, принятая в данной системе.

Кодирование целых и действительных чисел

Целые числа кодируются двоичным кодом достаточно просто — достаточно взять целое число и делить его пополам до тех пор, пока в остатке не образуется ноль или единица. Совокупность остатков от каждого деления, записанная справа налево вместе с последним остатком, и образует двоичный аналог десятичного числа.

$$19 : 2 = 9 + 1$$
$$9 : 2 = 4 + 1$$
$$4 : 2 = 2 + 0$$
$$2 : 2 = 1$$

Таким образом, $19_{10} = 1011_2$.

Для кодирования целых чисел от 0 до 255 достаточно иметь 8 разрядов двоичного кода (8 бит). Шестнадцать бит позволяют закодировать целые числа от 0 до 65535, а 24 бита — уже более 16,5 миллионов разных значений.

Для кодирования действительных чисел используют 80-разрядное кодирование. При этом число предварительно преобразуется в *нормализованную форму*:

$$3{,}1415926 = 0{,}31415926 \cdot 10^1$$
$$300\,000 = 0{,}3 \cdot 10^6$$
$$123\,456\,789 = 0{,}123456789 \cdot 10^{10}$$

Первая часть числа называется *мантиссой*, а вторая — *характеристикой*. Большую часть из 80 бит отводят для хранения мантиссы (вместе со знаком) и некоторое фиксированное количество разрядов отводят для хранения характеристики (тоже со знаком).

Кодирование текстовых данных

Если каждому символу алфавита сопоставить определенное целое число (например, порядковый номер), то с помощью двоичного кода можно кодировать и текстовую

информацию. Восьми двоичных разрядов достаточно для кодирования 256 различных символов. Этого хватит, чтобы выразить различными комбинациями восьми битов все символы английского и русского языков, как строчные, так и прописные, а также знаки препинания, символы основных арифметических действий и некоторые общепринятые специальные символы, например символ «§».

Технически это выглядит очень просто, однако всегда существовали достаточно веские организационные сложности. В первые годы развития вычислительной техники они были связаны с отсутствием необходимых стандартов, а в настоящее время вызваны, наоборот, изобилием одновременно действующих и противоречивых стандартов. Для того чтобы весь мир одинаково кодировал текстовые данные, нужны единые таблицы кодирования, а это пока невозможно из-за противоречий между символами национальных алфавитов, а также противоречий корпоративного характера.

Для английского языка, захватившего де-факто нишу международного средства общения, противоречия уже сняты. Институт стандартизации США (*ANSI — American National Standard Institute*) ввел в действие систему кодирования *ASCII* (*American Standard Code for Information Interchange — стандартный код информационного обмена США*). В системе *ASCII* закреплены две таблицы кодирования — *базовая* и *расширенная*. Базовая таблица закрепляет значения кодов от 0 до 127, а расширенная относится к символам с номерами от 128 до 255.

Первые 32 кода базовой таблицы, начиная с нулевого, отданы производителям аппаратных средств (в первую очередь производителям компьютеров и печатающих устройств). В этой области размещаются так называемые *управляющие коды*, которым не соответствуют никакие символы языков, и, соответственно, эти коды не выводятся ни на экран, ни на устройства печати, но ими можно управлять тем, как производится вывод прочих данных.

Начиная с кода 32 по код 127 размещены коды символов английского алфавита, знаков препинания, цифр, арифметических действий и некоторых вспомогательных символов. Базовая таблица кодировки *ASCII* приведена в таблице 1.1.

Таблица 1.1. Базовая таблица кодировки ASCII

32	пробел	48	0	64	@	80	P	96	`	112	p
33	!	49	1	65	A	81	Q	97	a	113	q
34	"	50	2	66	B	82	R	98	b	114	r
35	#	51	3	67	C	83	S	99	c	115	s
36	$	52	4	68	D	84	T	100	d	116	t
37	%	53	5	69	E	85	U	101	e	117	u
38	&	54	6	70	F	86	V	102	f	118	v
39	'	55	7	71	G	87	W	103	g	119	w
40	(	56	8	72	H	88	X	104	h	120	x
41	)	57	9	73	I	89	Y	105	i	121	y
42	*	58	:	74	J	90	Z	106	j	122	z
43	+	59	;	75	K	91	[	107	k	123	{
44	,	60	<	76	L	92	\	108	l	124	|
45	-	61	=	77	M	93	]	109	m	125	}
46	.	62	>	78	N	94	^	110	n	126	~
47	/	63	?	79	O	95	_	111	o	127	

Аналогичные системы кодирования текстовых данных были разработаны и в других странах. Так, например, в СССР в этой области действовала система кодирования КОИ-7 (*код обмена информацией, семизначный*). Однако поддержка производителей оборудования и программ вывела американский код *ASCII* на уровень международного стандарта, и национальным системам кодирования пришлось «отступить» во вторую, расширенную часть системы кодирования, определяющую значения кодов со 128 по 255. Отсутствие единого стандарта в этой области привело к множественности одновременно действующих кодировок. Только в России можно указать три действующих стандарта кодировки и еще два устаревших.

Так, например, кодировка символов русского языка, известная как кодировка *Windows-1251*, была введена «извне» — компанией *Microsoft*, но, учитывая широкое распространение операционных систем и других продуктов этой компании в России, она глубоко закрепилась и нашла широкое распространение (таблица 1.2). Эта кодировка используется на большинстве локальных компьютеров, работающих на платформе *Windows*.

Таблица 1.2. Кодировка Windows 1251

128 Ђ	144 ђ	160	176 °	192 А	208 Р	224 а	240 р
129 Ѓ	145 ‘	161 Ў	177 ±	193 Б	209 С	225 б	241 с
130 ‚	146 ’	162 ў	178 І	194 В	210 Т	226 в	242 т
131 ѓ	147 “	163 Ј	179 і	195 Г	211 У	227 г	243 у
132 „	148 ”	164 ¤	180 ґ	196 Д	212 Ф	228 д	244 ф
133 …	149 •	165 Ґ	181 µ	197 Е	213 Х	229 е	245 х
134 †	150 –	166 ¦	182 ¶	198 Ж	214 Ц	230 ж	246 ц
135 ‡	151 —	167 §	183 ·	199 З	215 Ч	231 з	247 ч
136	152	168 Ё	184 ё	200 И	216 Ш	232 и	248 ш
137 ‰	153 ™	169 ©	185 №	201 Й	217 Щ	233 й	249 щ
138 Љ	154 љ	170 Є	186 є	202 К	218 Ъ	234 к	250 ъ
139 ‹	155 ›	171 «	187 »	203 Л	219 Ы	235 л	251 ы
140 Њ	156 њ	172 ¬	188 ј	204 М	220 Ь	236 м	252 ь
141 Ќ	157 ќ	173 -	189 Ѕ	205 Н	221 Э	237 н	253 э
142 Ћ	158 ћ	174 ®	190 ѕ	206 О	222 Ю	238 о	254 ю
143 Џ	159 џ	175 Ї	191 ї	207 П	223 Я	239 п	255 я

Другая распространенная кодировка носит название КОИ-8 (*код обмена информацией, восьмизначный*) — ее происхождение относится ко временам действия Совета Экономической Взаимопомощи государств Восточной Европы (таблица 1.3). Сегодня кодировка КОИ-8 имеет широкое распространение в компьютерных сетях на территории России и в российском секторе Интернета.

Международный стандарт, в котором предусмотрена кодировка символов русского алфавита, носит название кодировки *ISO* (*International Standard Organization — Международный институт стандартизации*). На практике данная кодировка используется редко (таблица 1.4).

На компьютерах, работающих в операционных системах *MS-DOS*, могут действовать еще две кодировки (кодировка *ГОСТ* и кодировка *ГОСТ-альтернативная*). Первая из них считалась устаревшей даже в первые годы появления персональной вычислительной техники, но вторая используется и по сей день (см. таблицу 1.5).

Таблица 1.3. Кодировка КОИ-8

128	144 ▒	160 ─	176 ├	192 ю	208 п	224 Ю	240 П
129 │	145 ▒	161 Ё	177 ├	193 а	209 я	225 А	241 Я
130 ┌	146 ▒	162 ┌	178 ┤	194 б	210 р	226 Б	242 Р
131 ┐	147 ┌	163 ё	179 Ё	195 ц	211 с	227 Ц	243 С
132 └	148 ■	164 ┌	180 ┤	196 д	212 т	228 Д	244 Т
133 ┘	149 •	165 ┌	181 ┤	197 е	213 у	229 Е	245 У
134 ├	150 √	166 ┐	182 ┬	198 ф	214 ж	230 Ф	246 Ж
135 ┤	151 ≈	167 ┐	183 ┬	199 г	215 в	231 Г	247 В
136 ┬	152 ≤	168 ┐	184 ┬	200 х	216 ь	232 Х	248 Ь
137 ┴	153 ≥	169 └	185 ┴	201 и	217 ы	233 И	249 Ы
138 ┼	154	170 └	186 ┴	202 й	218 з	234 Й	250 З
139 ■	155 ⌠	171 └	187 ┴	203 к	219 ш	235 К	251 Ш
140 ▄	156 °	172 ┘	188 ┼	204 л	220 э	236 Л	252 Э
141 ▌	157 ²	173 ┘	189 ┼	205 м	221 щ	237 М	253 Щ
142 ▐	158 ·	174 ┘	190 ┼	206 н	222 ч	238 Н	254 Ч
143 ▀	159 ÷	175 ├	191 ё	207 о	223 ъ	239 О	255 Ъ

Таблица 1.4. Кодировка ISO

	160	176 А	192 Р	208 а	224 р	240 №
	161 Ё	177 Б	193 С	209 б	225 с	241 ё
	162 Ђ	178 В	194 Т	210 в	226 т	242 ђ
	163 Ѓ	179 Г	195 У	211 г	227 у	243 ѓ
	164 Є	180 Д	196 Ф	212 д	228 ф	244 є
	165 S	181 Е	197 Х	213 е	229 х	245 s
	166 I	182 Ж	198 Ц	214 ж	230 ц	246 і
В *ISO* не определены	167 Ї	183 З	199 Ч	215 з	231 ч	247 ї
	168 J	184 И	200 Ш	216 и	232 ш	248 j
	169 Љ	185 Й	201 Щ	217 й	233 щ	249 љ
	170 Њ	186 К	202 Ъ	218 к	234 ъ	250 њ
	171 Ћ	187 Л	203 Ы	219 л	235 ы	251 ћ
	172 Ќ	188 М	204 Ь	220 м	236 ь	252 ќ
	173 -	189 Н	205 Э	221 н	237 э	253 §
	174 Ў	190 О	206 Ю	222 о	238 ю	254 ў
	175 Џ	191 П	207 Я	223 п	239 я	255 џ

Таблица 1.5. ГОСТ-альтернативная кодировка

128 А	144 Р	160 а	176 ▒	192 └	208 ┴	224 р	240 Ё
129 Б	145 С	161 б	177 ▒	193 Г	209 ┬	225 с	241 ё
130 В	146 Т	162 в	178 ▒	194 ┬	210 ┬	226 т	242 Є
131 Г	147 У	163 г	179 │	195 ├	211 └	227 у	243 є
132 Д	148 Ф	164 д	180 ┤	196 ─	212 └	228 ф	244 Ї
133 Е	149 Х	165 е	181 ┤	197 ┼	213 ┌	229 х	245 ї
134 Ж	150 Ц	166 ж	182 ┤	198 ├	214 ┌	230 ц	246 Ў
135 З	151 Ч	167 з	183 ┐	199 ├	215 ┼	231 ч	247 ў
136 И	152 Ш	168 и	184 ┐	200 └	216 ┼	232 ш	248 °
137 Й	153 Щ	169 й	185 ┤	201 ┌	217 ┘	233 щ	249 •
138 К	154 Ъ	170 к	186 │	202 ┴	218 ┌	234 ъ	250 ·
139 Л	155 Ы	171 л	187 ┐	203 ┬	219 ▌	235 ы	251 √
140 М	156 Ь	172 м	188 ┘	204 ├	220 ▄	236 ь	252 №
141 Н	157 Э	173 н	189 ┘	205 ─	221 ▐	237 э	253 ¤
142 О	158 Ю	174 о	190 ┘	206 ┼	222 ▐	238 ю	254 ■
143 П	159 Я	175 п	191 ┐	207 ┴	223 ▀	239 я	255

В связи с изобилием систем кодирования текстовых данных, действующих в России, возникает задача межсистемного преобразования данных — это одна из распространенных задач информатики.

Универсальная система кодирования текстовых данных

Если проанализировать организационные трудности, связанные с созданием единой системы кодирования текстовых данных, то можно прийти к выводу, что они вызваны ограниченным набором кодов (256). В то же время очевидно, что если, например, кодировать символы не восьмиразрядными двоичными числами, а числами с большим количеством разрядов, то и диапазон возможных значений кодов станет намного больше. Такая система, основанная на 16-разрядном кодировании символов, получила название *универсальной — UNICODE*. Шестнадцать разрядов позволяют обеспечить уникальные коды для 65 536 различных символов — этого поля достаточно для размещения в одной таблице символов большинства языков планеты.

Несмотря на тривиальную очевидность такого подхода, простой механический переход на данную систему долгое время сдерживался из-за недостаточных ресурсов средств вычислительной техники (в системе кодирования *UNICODE* все текстовые документы автоматически становятся вдвое длиннее). Во второй половине 90-х годов технические средства достигли необходимого уровня обеспеченности ресурсами, и сегодня мы наблюдаем постепенный перевод документов и программных средств на универсальную систему кодирования. Для индивидуальных пользователей это еще больше добавило забот по согласованию документов, выполненных в разных системах кодирования, с программными средствами, но это надо понимать как трудности переходного периода.

Кодирование графических данных

Если рассмотреть с помощью увеличительного стекла черно-белое графическое изображение, напечатанное в газете или книге, то можно увидеть, что оно состоит из мельчайших точек, образующих характерный узор, называемый *растром* (рис. 1.3).

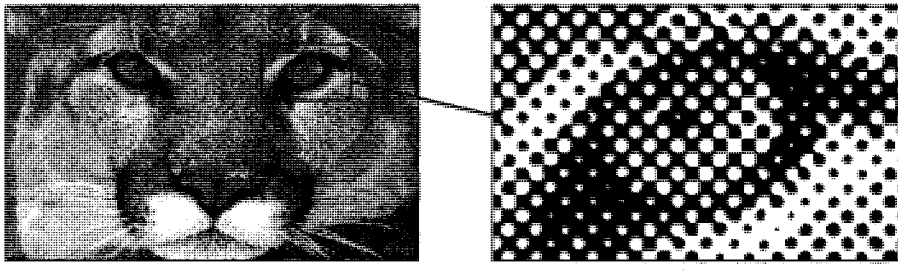


Рис. 1.3. Растр — это метод кодирования графической информации, издавна принятый в полиграфии

Поскольку линейные координаты и индивидуальные свойства каждой точки (яркость) можно выразить с помощью целых чисел, то можно сказать, что растровое кодирование позволяет использовать двоичный код для представления графических данных. Общепринятым на сегодняшний день считается представление черно-белых

иллюстраций в виде комбинации точек с 256 градациями серого цвета, и, таким образом, для кодирования яркости любой точки обычно достаточно восьмиразрядного двоичного числа.

Для кодирования цветных графических изображений применяется *принцип декомпозиции* произвольного цвета на основные составляющие. В качестве таких составляющих используют три основные цвета: красный (*Red, R*), зеленый (*Green, G*) и синий (*Blue, B*). На практике считается (хотя теоретически это не совсем так), что любой цвет, видимый человеческим глазом, можно получить путем механического смешения этих трех основных цветов. Такая система кодирования называется системой *RGB* по первым буквам названий основных цветов.

Если для кодирования яркости каждой из основных составляющих использовать по 256 значений (восемь двоичных разрядов), как это принято для полутоновых черно-белых изображений, то на кодирование цвета одной точки надо затратить 24 разряда. При этом система кодирования обеспечивает однозначное определение 16,5 млн различных цветов, что на самом деле близко к чувствительности человеческого глаза. Режим представления цветной графики с использованием 24 двоичных разрядов называется *полноцветным* (*True Color*).

Каждому из основных цветов можно поставить в соответствие дополнительный цвет, то есть цвет, дополняющий основной цвет до белого. Нетрудно заметить, что для любого из основных цветов дополнительным будет цвет, образованный суммой пары остальных основных цветов. Соответственно, дополнительными цветами являются: голубой (*Cyan, C*), пурпурный (*Magenta, M*) и желтый (*Yellow, Y*). Принцип декомпозиции произвольного цвета на составляющие компоненты можно применять не только для основных цветов, но и для дополнительных, то есть любой цвет можно представить в виде суммы голубой, пурпурной и желтой составляющей. Такой метод кодирования цвета принят в полиграфии, но в полиграфии используется еще и четвертая краска — черная (*Black, K*). Поэтому данная система кодирования обозначается четырьмя буквами *CMYK* (черный цвет обозначается буквой *K*, потому, что буква *B* уже занята синим цветом), и для представления цветной графики в этой системе надо иметь 32 двоичных разряда. Такой режим тоже называется *полноцветным* (*True Color*).

Если уменьшить количество двоичных разрядов, используемых для кодирования цвета каждой точки, то можно сократить объем данных, но при этом диапазон кодируемых цветов заметно сокращается. Кодирование цветной графики 16-разрядными двоичными числами называется режимом *High Color*.

При кодировании информации о цвете с помощью восьми бит данных можно передать только 256 цветовых оттенков. Такой метод кодирования цвета называется *индексным*. Смысл названия в том, что, поскольку 256 значений совершенно недостаточно, чтобы передать весь диапазон цветов, доступный человеческому глазу, код каждой точки растра выражает не цвет сам по себе, а только его номер (*индекс*) в некоей справочной таблице, называемой *палитрой*. Разумеется, эта палитра должна прикладываться к графическим данным — без нее нельзя воспользоваться методами воспроизведения информации на экране или бумаге (то есть, воспользоваться, конечно,

можно, но из-за неполноты данных полученная информация не будет адекватной: листва на деревьях может оказаться красной, а небо — зеленым).

Кодирование звуковой информации

Приемы и методы работы со звуковой информацией пришли в вычислительную технику наиболее поздно. К тому же, в отличие от числовых, текстовых и графических данных, у звукозаписей не было столь же длительной и проверенной истории кодирования. В итоге методы кодирования звуковой информации двоичным кодом далеки от стандартизации. Множество отдельных компаний разработали свои корпоративные стандарты, но если говорить обобщенно, то можно выделить два основных направления.

Метод FM (*Frequency Modulation*) основан на том, что теоретически любой сложный звук можно разложить на последовательность простейших гармонических сигналов разных частот, каждый из которых представляет собой правильную синусоиду, а следовательно, может быть описан числовыми параметрами, то есть кодом. В природе звуковые сигналы имеют непрерывный спектр, то есть являются аналоговыми. Их разложение в гармонические ряды и представление в виде дискретных цифровых сигналов выполняют специальные устройства — *аналогово-цифровые преобразователи* (*АЦП*). Обратное преобразование для воспроизведения звука, закодированного числовым кодом, выполняют *цифро-аналоговые преобразователи* (*ЦАП*). При таких преобразованиях неизбежны потери информации, связанные с методом кодирования, поэтому качество звукозаписи обычно получается не вполне удовлетворительным и соответствует качеству звучания простейших электромузыкальных инструментов с окрасом, характерным для электронной музыки. В то же время, данный метод кодирования обеспечивает весьма компактный код, и потому он нашел применение еще в те годы, когда ресурсы средств вычислительной техники были явно недостаточны.

Метод таблично-волнового (*Wave-Table*) синтеза лучше соответствует современному уровню развития техники. Если говорить упрощенно, то можно сказать, что где-то в заранее подготовленных таблицах хранятся образцы звуков для множества различных музыкальных инструментов (хотя не только для них). В технике такие образцы называют *сэмплами*. Числовые коды выражают тип инструмента, номер его модели, высоту тона, продолжительность и интенсивность звука, динамику его изменения, некоторые параметры среды, в которой происходит звучание, а также прочие параметры, характеризующие особенности звука. Поскольку в качестве образцов используются «реальные» звуки, то качество звука, полученного в результате синтеза, получается очень высоким и приближается к качеству звучания реальных музыкальных инструментов.

Основные структуры данных

Работа с большими наборами данных автоматизируется проще, когда данные *упорядочены*, то есть образуют заданную структуру. Существует три основных типа структур данных: *линейная*, *иерархическая* и *табличная*. Их можно рассмотреть на примере обычной книги.

Если разобрать книгу на отдельные листы и перемешать их, книга потеряет свое назначение. Она по-прежнему будет представлять набор данных, но подобрать адекватный метод для получения из нее информации весьма непросто. (Еще хуже дело будет обстоять, если из книги вырезать каждую букву отдельно, — в этом случае вряд ли вообще найдется адекватный метод для ее прочтения.)

Если же собрать все листы книги в правильной последовательности, мы получим простейшую структуру данных — *линейную*. Такую книгу уже можно читать, хотя для поиска нужных данных ее придется прочитать подряд, начиная с самого начала, что не всегда удобно.

Для быстрого поиска данных существует *иерархическая структура*. Так, например, книги разбивают на части, разделы, главы, параграфы и т. п. Элементы структуры более низкого уровня входят в элементы структуры более высокого уровня: разделы состоят из глав, главы из параграфов и т. д.

Для больших массивов поиск данных в иерархической структуре намного проще, чем в линейной, однако и здесь необходима *навигация*, связанная с необходимостью просмотра. На практике задачу упрощают тем, что в большинстве книг есть вспомогательная перекрестная *таблица*, связывающая элементы иерархической структуры с элементами линейной структуры, то есть связывающая разделы, главы и параграфы с номерами страниц. В книгах с простой иерархической структурой, рассчитанных на последовательное чтение, эту таблицу принято называть *оглавлением*, а в книгах со сложной структурой, допускающей выборочное чтение, ее называют *содержанием*.

Линейные структуры (списки данных, векторы данных)

Линейные структуры — это хорошо знакомые нам списки. *Список* — это простейшая структура данных, отличающаяся тем, что каждый элемент данных однозначно определяется своим номером в массиве. Проставляя номера на отдельных страницах рассыпанной книги, мы создаем структуру списка. Обычный журнал посещаемости занятий, например, имеет структуру списка, поскольку все студенты группы зарегистрированы в нем под своими *уникальными* номерами. Мы называем номера *уникальными* потому, что в одной группе не могут быть зарегистрированы два студента с одним и тем же номером.

При создании любой структуры данных надо решить два вопроса: как разделять элементы данных между собой и как разыскивать нужные элементы. В журнале посещаемости, например, это решается так: каждый новый элемент списка заносится с новой строки, то есть разделителем является конец строки. Тогда нужный элемент можно разыскать по номеру строки.

N п/п	Фамилия, Имя, Отчество
1	Аистов Александр Алексеевич
2	Бобров Борис Борисович
3	Воробьева Валентина Владиславовна
...	.....
27	Сорокин Сергей Семенович

Разделителем может быть и какой-нибудь специальный символ. Нам хорошо известны разделители между словами — это пробелы. В русском и во многих европейских языках общепринятым разделителем предложений является точка. В рассмотренном нами классном журнале в качестве разделителя можно использовать любой символ, который не встречается в самих данных, например символ «*». Тогда список выглядел бы так:

Аистов Александр Алексеевич * Бобров Борис Борисович * Воробьева Валентина Владиславовна * ... * Сорокин Сергей Семенович

В этом случае для розыска элемента с номером n надо просмотреть список начиная с самого начала и пересчитать встретившиеся разделители. Когда будет отсчитано $n-1$ разделителей, начнется нужный элемент. Он закончится, когда будет встречен следующий разделитель.

Еще проще можно действовать, если все элементы списка имеют равную длину. В этом случае разделители в списке вообще не нужны. Для розыска элемента с номером n надо просмотреть список с самого начала и отсчитать $a(n-1)$ символ, где a — длина одного элемента. Со следующего символа начнется нужный элемент. Его длина тоже равна a, поэтому его конец определить нетрудно. Такие упрощенные списки, состоящие из элементов равной длины, называют *векторами данных*. Работать с ними особенно удобно.

Таким образом, *линейные структуры данных (списки) — это упорядоченные структуры, в которых адрес элемента однозначно определяется его номером.*

Табличные структуры (таблицы данных, матрицы данных)

С таблицами данных мы тоже хорошо знакомы, достаточно вспомнить всем известную таблицу умножения. Табличные структуры отличаются от списочных тем, что элементы данных определяются *адресом ячейки*, который состоит не из одного параметра, как в списках, а из нескольких. Для таблицы умножения, например, адрес ячейки определяется номерами строки и столбца. Нужная ячейка находится на их пересечении, а элемент выбирается из ячейки.

При хранении табличных данных количество разделителей должно быть больше, чем для данных, имеющих структуру списка. Например, когда таблицы печатают в книгах, строки и столбцы разделяют графическими элементами — линиями вертикальной и горизонтальной разметки (рис. 1.4).

Если нужно сохранить таблицу в виде длинной символьной строки, используют один символ-разделитель между элементами, принадлежащими одной строке, и другой разделитель для отделения строк, например так:

Меркурий*0,39*0,056*0#Венера*0,67*0,88*0#Земля*1,0*1,0*1#Марс*1,51*0,1*2#...

Для розыска элемента, имеющего адрес ячейки (m, n), надо просмотреть набор данных с самого начала и пересчитать внешние разделители. Когда будет отсчитан $m-1$ разделитель, надо пересчитывать внутренние разделители. После того как будет найден $n-1$ разделитель, начнется нужный элемент. Он закончится, когда будет встречен любой очередной разделитель.

Планета	Расстояние до Солнца, а.е.	Относительная масса	Количество спутников
Меркурий	0,39	0,056	0
Венера	0,67	0,88	0
Земля	1,0	1,0	1
Марс	1,51	0,1	2
Юпитер	5,2	318	16

Рис. 1.4. В двумерных таблицах, которые печатают в книгах, применяется два типа разделителей — вертикальные и горизонтальные

Еще проще можно действовать, если все элементы таблицы имеют равную длину. Такие таблицы называют *матрицами*. В данном случае разделители не нужны, поскольку все элементы имеют равную длину и количество их известно. Для розыска элемента с адресом (m, n) в матрице, имеющей M строк и N столбцов, надо просмотреть ее с самого начала и отсчитать $a\,[N(m-1)+(n-1)]$ символ, где a — длина одного элемента. Со следующего символа начнется нужный элемент. Его длина тоже равна a, поэтому его конец определить нетрудно.

Таким образом, табличные структуры данных (*матрицы*) — это упорядоченные структуры, в которых *адрес элемента определяется номером строки и номером столбца, на пересечении которых находится ячейка, содержащая искомый элемент.*

Многомерные таблицы. Выше мы рассмотрели пример таблицы, имеющей два измерения (строка и столбец), но в жизни нередко приходится иметь дело с таблицами, у которых количество измерений больше. Вот пример таблицы, с помощью которой может быть организован учет учащихся.

Номер факультета:	3
Номер курса (на факультете):	2
Номер специальности (на курсе):	2
Номер группы в потоке одной специальности:	1
Номер учащегося в группе:	19

Размерность такой таблицы равна пяти, и для однозначного отыскания данных об учащемся в подобной структуре надо знать все пять параметров (координат).

Иерархические структуры данных

Нерегулярные данные, которые трудно представить в виде списка или таблицы, часто представляют в виде *иерархических структур*. С подобными структурами мы очень хорошо знакомы по обыденной жизни. Иерархическую структуру имеет система почтовых адресов. Подобные структуры также широко применяют в научных систематизациях и всевозможных классификациях (рис. 1.5).

В иерархической структуре адрес каждого элемента определяется путем доступа (маршрутом), ведущим от вершины структуры к данному элементу. Вот, например, как выглядит путь доступа к команде, запускающей программу Калькулятор (стандартная программа компьютеров, работающих в операционной системе *Windows 98*):

Пуск ▸ Программы ▸ Стандартные ▸ Калькулятор.

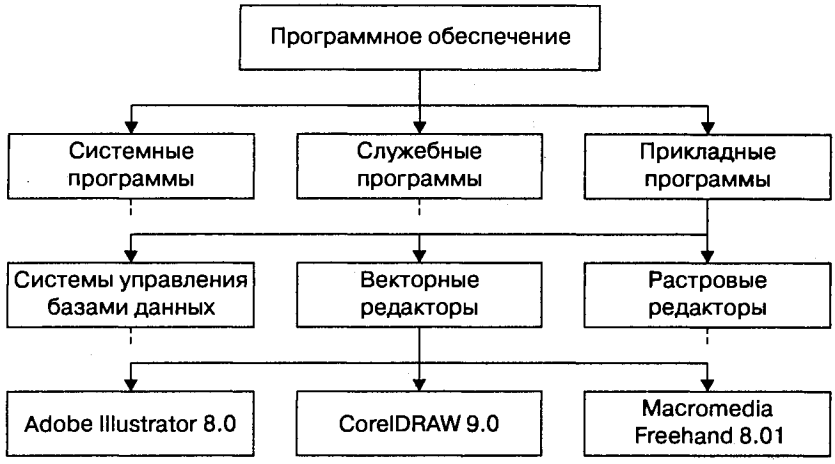


Рис. 1.5. Пример иерархической структуры данных

Дихотомия данных. Основным недостатком иерархических структур данных является увеличенный размер пути доступа. Очень часто бывает так, что длина маршрута оказывается больше, чем длина самих данных, к которым он ведет. Поэтому в информатике применяют методы для регуляризации иерархических структур с тем, чтобы сделать путь доступа компактным. Один из методов получил название *дихотомии*. Его суть понятна из примера, представленного на рис. 1.6.

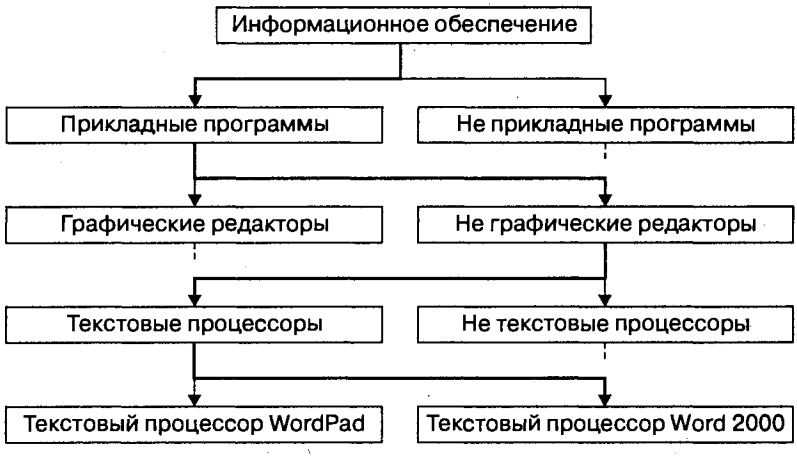


Рис. 1.6. Пример, поясняющий принцип действия метода дихотомии

В иерархической структуре, построенной методом дихотомии, путь доступа к любому элементу можно представить как путь через рациональный лабиринт с поворотами налево (0) или направо (1) и, таким образом, выразить путь доступа в виде компактной двоичной записи. В нашем примере путь доступа к текстовому процессору Word 2000 выразится следующим двоичным числом: 1010.

Упорядочение структур данных

Списочные и табличные структуры являются простыми. Ими легко пользоваться, поскольку адрес каждого элемента задается числом (для списка), двумя числами (для двумерной таблицы) или несколькими числами для многомерной таблицы. Они также легко упорядочиваются. Основным методом упорядочения является *сортировка*. Данные можно сортировать по любому избранному критерию, например: по алфавиту, по возрастанию порядкового номера или по возрастанию какого-либо параметра.

Несмотря на многочисленные удобства, у простых структур данных есть и недостаток — их трудно обновлять. Если, например, перевести студента из одной группы в другую, изменения надо вносить сразу в два журнала посещаемости; при этом в обоих журналах будет нарушена списочная структура. Если переведенного студента вписать в конец списка группы, нарушится упорядочение по алфавиту, а если его вписать в соответствии с алфавитом, то изменятся порядковые номера всех студентов, которые следуют за ним.

Таким образом, *при добавлении произвольного элемента в упорядоченную структуру списка может происходить изменение адресных данных у других элементов*. В журналах успеваемости это пережить нетрудно, но в системах, выполняющих автоматическую обработку данных, нужны специальные методы для решения этой проблемы.

Иерархические структуры данных по форме сложнее, чем линейные и табличные, но они не создают проблем с обновлением данных. Их легко развивать путем создания новых уровней. Даже если в учебном заведении будет создан новый факультет, это никак не отразится на пути доступа к сведениям об учащихся прочих факультетов.

Недостатком иерархических структур является относительная трудоемкость записи адреса элемента данных и сложность упорядочения. Часто методы упорядочения в таких структурах основывают на предварительной *индексации*, которая заключается в том, что каждому элементу данных присваивается свой уникальный индекс, который можно использовать при поиске, сортировке и т. п. Ранее рассмотренный принцип дихотомии на самом деле является одним из методов индексации данных в иерархических структурах. После такой индексации данные легко разыскиваются по двоичному коду связанного с ними индекса.

Адресные данные. Если данные хранятся не как попало, а в организованной структуре (причем любой), то каждый элемент данных приобретает новое свойство (параметр), который можно назвать *адресом*. Конечно, работать с упорядоченными данными удобнее, но за это приходится платить их размножением, поскольку адреса элементов данных — это тоже данные и их тоже надо хранить и обрабатывать.

1.3. Файлы и файловая структура

Единицы представления данных

Существует множество систем представления данных. С одной из них, принятой в информатике и вычислительной технике, двоичным кодом, мы познакомились выше. Наименьшей единицей такого представления является бит (*двоичный разряд*).

Совокупность двоичных разрядов, выражающих числовые или иные данные, образует некий битовый рисунок. Практика показывает, что с битовым представлением удобнее работать, если этот рисунок имеет регулярную форму. В настоящее время в качестве таких форм используются группы из восьми битов, которые называются *байтами*.

Десятичное число	Двоичное число	Байт
1	1	0000 0001
2	10	0000 0010
...	...	...
255	11111111	1111 1111

Понятие о байте как группе взаимосвязанных битов появилось вместе с первыми образцами электронной вычислительной техники. Долгое время оно было *машинно-зависимым*, то есть для разных вычислительных машин длина байта была разной. Только в конце 60-х годов понятие байта стало универсальным и *машиннонезависимым*.

Выше мы видели, что во многих случаях целесообразно использовать не восьми-разрядное кодирование, а 16-разрядное, 24-разрядное, 32-разрядное и более. Группа из 16 взаимосвязанных бит (двух взаимосвязанных байтов) в информатике называется *словом*. Соответственно, группы из четырех взаимосвязанных байтов (32 разряда) называются *удвоенным словом*, а группы из восьми байтов (64 разряда) — *учетверенным словом*. Пока, на сегодняшний день, такой системы обозначения достаточно.

Единицы измерения данных

Существует много различных систем и единиц измерения данных. Каждая научная дисциплина и каждая область человеческой деятельности может использовать свои, наиболее удобные или традиционно устоявшиеся единицы. В информатике для измерения данных используют тот факт, что разные типы данных имеют универсальное двоичное представление и потому вводят свои единицы данных, основанные на нем.

Наименьшей единицей измерения является байт. Поскольку одним байтом, как правило, кодируется один символ текстовой информации, то для текстовых документов размер в байтах соответствует лексическому объему в символах (пока исключение представляет рассмотренная выше универсальная кодировка *UNICODE*).

Более крупная единица измерения — килобайт (Кбайт). Условно можно считать, что 1 Кбайт примерно равен 1000 байт. Условность связана с тем, что для вычислительной техники, работающей с двоичными числами, более удобно представление чисел в виде степени двойки и потому на самом деле 1 Кбайт равен 2^{10} байт (1024 байт). Однако всюду, где это не принципиально, с инженерной погрешностью (до 3 %) «забывают» о «лишних» байтах.

В килобайтах измеряют сравнительно небольшие объемы данных. Условно можно считать, что одна страница неформатированного машинописного текста составляет около 2 Кбайт.

Более крупные единицы измерения данных образуются добавлением префиксов *мега-*, *гига- тера-*; в более крупных единицах пока нет практической надобности.

1 Мбайт = 1024 Кбайт = 10^{20} байт

1 Гбайт = 1024 Мбайт = 10^{30} байт

1 Тбайт = 1024 Гбайт = 10^{40} байт

Особо обратим внимание на то, что при переходе к более крупным единицам «инженерная» погрешность, связанная с округлением, накапливается и становится недопустимой, поэтому на старших единицах измерения округление производится реже.

Единицы хранения данных

При хранении данных решаются две проблемы: как сохранить данные в наиболее компактном виде и как обеспечить к ним удобный и быстрый доступ (если доступ не обеспечен, то это не хранение). Для обеспечения доступа необходимо, чтобы данные имели упорядоченную структуру, а при этом, как мы уже знаем, образуется «паразитная нагрузка» в виде адресных данных. Без них нельзя получить доступ к нужным элементам данных, входящих в структуру.

Поскольку адресные данные тоже имеют размер и тоже подлежат хранению, хранить данные в виде мелких единиц, таких как байты, неудобно. Их неудобно хранить и в более крупных единицах (килобайтах, мегабайтах и т. п.), поскольку неполное заполнение одной единицы хранения приводит к неэффективности хранения.

В качестве единицы хранения данных принят объект переменной длины, называемый *файлом*. *Файл — это последовательность произвольного числа байтов, обладающая уникальным собственным именем.* Обычно в отдельном файле хранят данные, относящиеся к одному типу. В этом случае тип данных определяет *тип файла*.

Проще всего представить себе файл в виде безразмерного канцелярского досье, в которое можно по желанию добавлять содержимое или извлекать его оттуда. Поскольку в определении файла нет ограничений на размер, можно представить себе файл, имеющий 0 байтов (*пустой файл*), и файл, имеющий любое число байтов.

В определении файла особое внимание уделяется имени. Оно фактически несет в себе адресные данные, без которых данные, хранящиеся в файле, не станут информацией из-за отсутствия метода доступа к ним. Кроме функций, связанных с адресацией, имя файла может хранить и сведения о типе данных, заключенных в нем. Для автоматических средств работы с данными это важно, поскольку по имени файла они могут автоматически определить адекватный метод извлечения информации из файла.

Понятие о файловой структуре

Требование уникальности имени файла очевидно — без этого невозможно гарантировать однозначность доступа к данным. В средствах вычислительной техники требование уникальности имени обеспечивается автоматически — создать файл с именем, тождественным уже имеющемуся, не может ни пользователь, ни автоматика.

Хранение файлов организуется в иерархической структуре, которая в данном случае называется *файловой структурой*. В качестве вершины структуры служит имя носителя, на котором сохраняются файлы. Далее файлы группируются в *каталоги* (*папки*), внутри которых могут быть созданы *вложенные каталоги* (*папки*). *Путь доступа к файлу* начинается с имени устройства и включает все имена каталогов (папок), через которые проходит. В качестве разделителя используется символ «\» (обратная косая черта).

Уникальность имени файла обеспечивается тем, что *полным именем файла считается собственное имя файла вместе с путем доступа к нему*. Понятно, что в этом случае на одном носителе не может быть двух файлов с тождественными полными именами.

Пример записи полного имени файла:

<имя носителя>\<имя каталога-1>\...\<имя каталога-N>\<собственное имя файла>

Вот пример записи двух файлов, имеющих одинаковое собственное имя и размещенных на одном носителе, но отличающихся путем доступа, то есть полным именем. Для наглядности имена каталогов (папок) напечатаны прописными буквами.

C:\АВТОМАТИЧЕСКИЕ АППАРАТЫ\ВЕНЕРА\АТМОСФЕРА\Результаты исследований

C:\РАДИОЛОКАЦИЯ\ВЕНЕРА\РЕЛЬЕФ\Результаты исследований

О том, как на практике реализуются файловые структуры, мы узнаем несколько позже, когда познакомимся со средствами вычислительной техники и с понятием *файловой системы*.

1.4. Информатика

Предмет и задачи информатики

Информатика — это техническая наука, систематизирующая приемы создания, хранения, воспроизведения, обработки и передачи данных средствами вычислительной техники, а также принципы функционирования этих средств и методы управления ими.

Из этого определения видно, что информатика очень близка к технологии, поэтому ее предмет нередко называют *информационной технологией*.

Предмет информатики составляют следующие понятия:

- аппаратное обеспечение средств вычислительной техники;
- программное обеспечение средств вычислительной техники;
- средства взаимодействия аппаратного и программного обеспечения;
- средства взаимодействия человека с аппаратными и программными средствами.

Как видно из этого списка, в информатике особое внимание уделяется вопросам *взаимодействия*. Для этого даже есть специальное понятие — *интерфейс*. Методы и средства взаимодействия человека с аппаратными и программными средствами называют *пользовательским интерфейсом*. Соответственно, существуют *аппаратные интерфейсы*, *программные интерфейсы* и *аппаратно-программные интерфейсы*.

Основной задачей информатики является систематизация приемов и методов работы с аппаратными и программными средствами вычислительной техники. *Цель* систематизации состоит в выделении, внедрении и развитии передовых, наиболее эффективных технологий, в автоматизации этапов работы с данными, а также в методическом обеспечении новых технологических исследований.

Информатика — практическая наука. Ее достижения должны проходить подтверждение практикой и приниматься в тех случаях, когда они соответствуют критерию повышения эффективности. В составе основной задачи информатики сегодня можно выделить следующие направления для практических приложений:

- архитектура вычислительных систем (приемы и методы построения систем, предназначенных для автоматической обработки данных);
- интерфейсы вычислительных систем (приемы и методы управления аппаратным и программным обеспечением);
- программирование (приемы, методы и средства разработки компьютерных программ);
- преобразование данных (приемы и методы преобразования структур данных);
- защита информации (обобщение приемов, разработка методов и средств защиты данных);
- автоматизация (функционирование программно-аппаратных средств без участия человека);
- стандартизация (обеспечение совместимости между аппаратными и программными средствами, а также между форматами представления данных, относящихся к различным типам вычислительных систем).

На всех этапах технического обеспечения информационных процессов для информатики ключевым понятием является *эффективность*. Для аппаратных средств под эффективностью понимают отношение производительности оборудования к его стоимости (с учетом стоимости эксплуатации и обслуживания). Для программного обеспечения под эффективностью понимают производительность лиц, работающих с ними (пользователей). В программировании под эффективностью понимают объем программного кода, создаваемого программистами в единицу времени.

В информатике все жестко ориентировано на эффективность. Вопрос, *как сделать ту или иную операцию*, для информатики является важным, но не основным. Основным же является вопрос, *как сделать данную операцию эффективно*.

Истоки и предпосылки информатики

Слово *информатика* происходит от французского слова *Informatique*, образованного в результате объединения терминов *Informacion* (*информация*) и *Automatique* (*автоматика*), что выражает ее суть как науки об автоматической обработке информации. Кроме Франции термин *информатика* используется в ряде стран Восточной Европы. В то же время, в большинстве стран Западной Европы и США используется другой термин — *Computer Science* (*наука о средствах вычислительной техники*).

В качестве источников информатики обычно называют две науки — *документалистику* и *кибернетику*. Документалистика сформировалась в конце XIX века в связи с бурным развитием производственных отношений. Ее расцвет пришелся на 20–30-е годы XX века, а основным предметом стало изучение рациональных средств и методов повышения эффективности документооборота.

Основы близкой к информатике технической науки *кибернетики* были заложены трудами по математической логике американского математика Норберта Винера, опубликованными в 1948 году, а само название происходит от греческого слова (*kybernetic os* — искусный в управлении).

Впервые термин *кибернетика* ввел французский физик Андре Мари Ампер в первой половине XIX веке. Он занимался разработкой единой системы классификации всех наук и обозначил этим термином гипотетическую науку об управлении, которой в то время не существовало, но которая, по его мнению, должна была существовать.

Сегодня предметом кибернетики являются принципы построения и функционирования систем автоматического управления, а основными задачами — методы моделирования процесса принятия решений техническими средствами, связь между психологией человека и математической логикой, связь между информационным процессом отдельного индивидуума и информационными процессами в обществе, разработка принципов и методов искусственного интеллекта. На практике кибернетика во многих случаях опирается на те же программные и аппаратные средства вычислительной техники, что и информатика, а информатика, в свою очередь, заимствует у кибернетики математическую и логическую базу для развития этих средств.

Подведение итогов

Все процессы в природе сопровождаются *сигналами*. Зарегистрированные сигналы образуют *данные*. Данные преобразуются, транспортируются и потребляются с помощью *методов*. При взаимодействии данных и адекватных им методов образуется *информация*. Информация — это динамический объект, образующийся в ходе *информационного процесса*. Он отражает диалектическую связь между объективными данными и субъективными методами. Свойства информации зависят как от свойств данных, так и от свойств методов.

Данные различаются *типами*, что связано с различиями в физической природе сигналов, при регистрации которых образовались данные. В качестве средства хранения и транспортировки данных используются *носители данных*. Для удобства операций с данными их структурируют. Наиболее широко используются следующие структуры: *линейная*, *табличная* и *иерархическая* — они различаются методом адресации к данным. При сохранении данных образуются данные нового типа — *адресные данные*.

Вопросами систематизации приемов и методов создания, хранения, воспроизведения, обработки и передачи данных средствами вычислительной техники занимается техническая наука — *информатика*. С целью унификации приемов и методов работы с данными в вычислительной технике применяется универсальная система кодирования данных, называемая *двоичным кодом*. Элементарной единицей представления

данных в двоичном коде является *двоичный разряд* (*бит*). Другой, более крупной единицей представления данных является *байт*.

Основной единицей хранения данных является *файл*. Файл представляет собой последовательность байтов, имеющую собственное имя. Совокупность файлов образует файловую структуру, которая, как правило, относится к иерархическому типу. *Полный адрес* файла в файловой структуре является уникальным и включает в себя собственное имя файла и путь доступа к нему.

Вопросы для самоконтроля

1. Как вы можете объяснить бытовой термин «переизбыток информации»? Что имеется в виду: излишняя полнота данных; излишняя сложность методов; неадекватность поступающих данных и методов, имеющихся в наличии?
2. Как вы понимаете термин «средство массовой информации»? Что это? Средство массовой поставки данных? Средство, обеспечивающее массовое распространение методов? Средство, обеспечивающее процесс информирования путем поставки данных гражданам, обладающим адекватными методами их потребления?
3. Как вы полагаете, являются ли данные товаром? Могут ли методы быть товаром?
4. На примере коммерческих структур, обеспечивающих коммуникационные услуги, покажите, как взаимодействуют между собой маркетинг данных и маркетинг методов? Можете ли вы привести примеры лизинга данных и методов?
5. Как вы понимаете диалектическое единство данных и методов? Можете ли вы привести примеры аналогичного единства двух понятий из других научных дисциплин: естественных, социальных, технических?
6. Как вы понимаете динамический характер информации? Что происходит с ней по окончании информационного процесса?
7. Можем ли мы утверждать, что данные, полученные в результате информационного процесса, адекватны исходным? Почему? От каких свойств исходных данных и методов зависит адекватность результирующих данных?
8. Что такое *вектор данных*? Является ли список номеров телефонов населенного пункта вектором данных? Является ли вектором данных текстовый документ, закодированный двоичным кодом, если он не содержит элементов оформления?
9. Является ли цифровой код цветного фотоснимка вектором данных? Если нет, то чего ему не хватает?
10. Как вы понимаете следующие термины: *аппаратно-программный интерфейс, программный интерфейс, аппаратный интерфейс?* Как бы вы назвали специальность людей, разрабатывающих аппаратные интерфейсы? Как называется специальность людей, разрабатывающих программные интерфейсы?
11. На основе личных наблюдений сделайте вывод о том, какими средствами может пользоваться преподаватель для обеспечения интерфейса с аудиторией. Можете ли вы рассмотреть отдельно методические и технические средства, имеющиеся в его распоряжении? Может ли преподаватель рассматривать *вашу* тетрадь и авторучку как *свое* средство обеспечения интерфейса? Если да, то в какой мере?

2.1. История развития средств вычислительной техники

Вычислительная система, компьютер

Изыскание средств и методов механизации и автоматизации работ — одна из основных задач технических дисциплин. Автоматизация работ с данными имеет свои особенности и отличия от автоматизации других типов работ. Для этого класса задач используют особые виды устройств, большинство из которых являются электронными приборами. Совокупность устройств, предназначенных для автоматической или автоматизированной обработки данных, называют *вычислительной техникой*. Конкретный набор взаимодействующих между собой устройств и программ, предназначенный для обслуживания одного рабочего участка, называют *вычислительной системой*. Центральным устройством большинства вычислительных систем является *компьютер*.

Компьютер — это электронной прибор, предназначенный для автоматизации создания, хранения, обработки и транспортировки данных.

Принцип действия компьютера

В определении компьютера как прибора мы указали определяющий признак — *электронный*. Однако автоматические вычисления не всегда производились электронными устройствами. Известны и механические устройства, способные выполнять расчеты автоматически.

Анализируя раннюю историю вычислительной техники, некоторые зарубежные исследователи нередко в качестве древнего предшественника компьютера называют механическое счетное устройство *абак*. Подход «от абака» свидетельствует о глубоком методическом заблуждении, поскольку абак не обладает свойством автоматического выполнения вычислений, а для компьютера оно определяющее.

☑ Абак — наиболее раннее счетное механическое устройство, первоначально представлявшее собой глиняную пластину с желобами, в которых раскладывались камни, представляющие числа. Появление абака относят к четвертому тысячелетию до н. э. Местом появления считается Азия. В средние века в Европе абак сменился разграфленными таблицами. Вычисления с их помощью называли *счетом на линиях*, а в России в XVI–XVII веках появилось намного более передовое изобретение, применяемое и поныне, — *русские счеты*.

В то же время, нам хорошо знаком другой прибор, способный автоматически выполнять вычисления, — это часы. Независимо от принципа действия, все виды часов (песочные, водяные, механические, электрические, электронные и др.) обладают способностью генерировать через равные промежутки времени перемещения или сигналы и регистрировать возникающие при этом изменения, то есть выполнять автоматическое суммирование сигналов или перемещений. Этот принцип прослеживается даже в солнечных часах, содержащих только устройство регистрации (роль генератора выполняет система Земля — Солнце).

☑ Механические часы — прибор, состоящий из устройства, автоматически выполняющего перемещения через равные заданные интервалы времени и устройства регистрации этих перемещений. Место появления первых механических часов неизвестно. Наиболее ранние образцы относятся к XIV веку и принадлежат монастырям (*башенные часы*).

В основе любого современного компьютера, как и в электронных часах, лежит *тактовый генератор*, вырабатывающий через равные интервалы времени электрические сигналы, которые используются для приведения в действие всех устройств компьютерной системы. Управление компьютером фактически сводится к управлению распределением сигналов между устройствами. Такое управление может производиться автоматически (в этом случае говорят о *программном управлении*) или вручную с помощью внешних органов управления — кнопок, переключателей, перемычек и т. п. (в ранних моделях). В современных компьютерах внешнее управление в значительной степени автоматизировано с помощью специальных аппаратно-логических интерфейсов, к которым подключаются устройства управления и ввода данных (клавиатура, мышь, джойстик и другие). В отличие от программного управления такое управление называют *интерактивным*.

Механические первоисточники

Первое в мире автоматическое устройство для выполнения операции сложения было создано на базе механических часов. В 1623 году его разработал Вильгельм Шикард, профессор кафедры восточных языков в университете Тюбингена (Германия). В наши дни рабочая модель устройства была воспроизведена по чертежам и подтвердила свою работоспособность. Сам изобретатель в письмах называл машину «суммирующими часами».

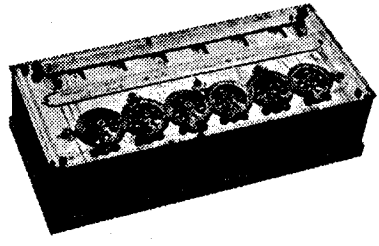


Рис. 2.1. Суммирующая машина Паскаля

В 1642 году французский механик Блез Паскаль (1623–1662) разработал более компактное суммирующее устройство (рис. 2.1), которое стало первым в мире механическим калькулятором, выпус-

кавшимся серийно (главным образом для нужд парижских ростовщиков и менял). В 1673 году немецкий математик и философ Г. В. Лейбниц (1646–1717) создал механический калькулятор, который мог выполнять операции умножения и деления путем многократного повторения операций сложения и вычитания.

На протяжении XVIII века, известного как эпоха Просвещения, появились новые, более совершенные модели, но принцип механического управления вычислительными операциями оставался тем же. Идея программирования вычислительных операций пришла из той же часовой промышленности. Старинные монастырские башенные часы были настроены так, чтобы в заданное время включать механизм, связанный с системой колоколов. Такое программирование было *жестким* — одна и та же операция выполнялась в одно и то же время.

Идея гибкого программирования механических устройств с помощью перфорированной бумажной ленты впервые была реализована в 1804 году в ткацком станке Жаккарда, после чего оставался только один шаг до программного управления вычислительными операциями.

Этот шаг был сделан выдающимся английским математиком и изобретателем Чарльзом Бэббиджем (1792–1871) в его Аналитической машине, которая, к сожалению, так и не была до конца построена изобретателем при жизни, но была воспроизведена в наши дни по его чертежам, так что сегодня мы вправе говорить об Аналитической машине, как о реально существующем устройстве. Особенностью Аналитической машины стало то, что здесь впервые был реализован *принцип разделения информации на команды и данные*. Аналитическая машина содержала два крупных узла — «склад» и «мельницу». Данные вводились в механическую память «склада» путем установки блоков шестерен, а потом обрабатывались в «мельнице» с использованием команд, которые вводились с перфорированных карт (как в ткацком станке Жаккарда).

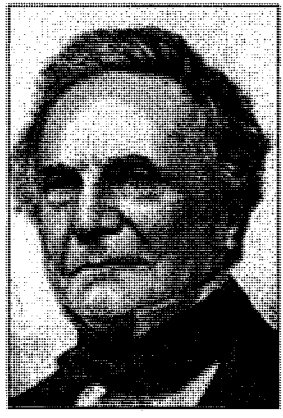


Рис. 2.2. Чарльз Бэббидж

Исследователи творчества Чарльза Бэббиджа непременно отмечают особую роль в разработке проекта Аналитической машины графини Огасты Ады Лавлейс (1815–1852), дочери известного поэта лорда Байрона. Именно ей принадлежала идея использования перфорированных карт для программирования вычислительных операций (1843). В частности, в одном из писем она писала: «Аналитическая машина точно так же плетет алгебраические узоры, как ткацкий станок воспроизводит цветы и листья». Леди Аду можно с полным основанием назвать самым первым в мире программистом. Сегодня ее именем назван один из известных языков программирования.

Идея Чарльза Бэббиджа о раздельном рассмотрении *команд* и *данных* оказалась необычайно плодотворной. В XX в. она была развита в принципах Джона фон Неймана (1941 г.), и сегодня в вычислительной технике принцип раздельного рассмотрения *программ* и *данных* имеет очень важное значение. Он учитывается и при разработке архитектур современных компьютеров, и при разработке компьютерных программ.

Математические первоисточники

Если мы задумаемся над тем, с какими объектами работали первые механические предшественники современного электронного компьютера, то должны признать, что числа представлялись либо в виде линейных перемещений цепных и реечных механизмов, либо в виде угловых перемещений зубчатых и рычажных механизмов. И в том и в другом случае это были перемещения, что не могло не сказываться на габаритах устройств и на скорости их работы. Только переход от регистрации перемещений к регистрации сигналов позволил значительно снизить габариты и повысить быстродействие. Однако на пути к этому достижению потребовалось ввести еще несколько важных принципов и понятий.

Двоичная система Лейбница. В механических устройствах зубчатые колеса могут иметь достаточно много фиксированных и, главное, *различимых* между собой положений. Количество таких положений, по крайней мере, равно числу зубьев шестерни. В электрических и электронных устройствах речь идет не о регистрации *положений* элементов конструкции, а о регистрации *состояний* элементов устройства. Таких устойчивых и *различимых* состояний всего два: включен — выключен; открыт — закрыт; заряжен — разряжен и т. п. Поэтому традиционная десятичная система, использованная в механических калькуляторах, неудобна для электронных вычислительных устройств.

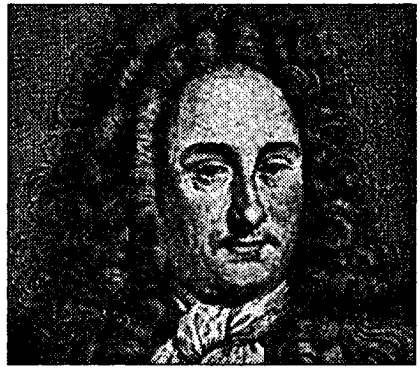


Рис. 2.3. Готфрид Вильгельм Лейбниц

Возможность представления любых чисел (да и не только чисел) двоичными цифрами впервые была предложена Готфридом Вильгельмом Лейбницем в 1666 году Он пришел к двоичной системе счисления, занимаясь исследованиями философской концепции единства и борьбы противоположностей. Попытка представить мироздание в виде непрерывного взаимодействия двух начал («черного» и «белого», мужского и женского, добра и зла) и применить к его изучению методы «чистой» математики подтолкнули Лейбница к изучению свойств двоичного представления данных с помощью нулей и единиц. Надо сказать, что Лейбницу уже тогда приходила в голову мысль о возможности использования двоичной системы в вычислительном устройстве, но, поскольку для механических устройств в этом не было никакой необходимости, он не стал использовать в своем калькуляторе (1673 году) принципы двоичной системы.

Математическая логика Джорджа Буля. Говоря о творчестве Джорджа Буля, исследователи истории вычислительной техники непременно подчеркивают, что этот выдающийся английский ученый первой половины XIX века был самоучкой. Возможно, именно благодаря отсутствию «классического» (в понимании того времени) образования Джордж Буль внес в логику как в науку революционные изменения.

Занимаясь исследованием законов мышления, он применил в логике систему формальных обозначений и правил, близкую к математической. Впоследствии эту сис-

тему назвали *логической алгеброй* или *булевой алгеброй*. Правила этой системы применимы к самым разнообразным объектам и их группам (*множествам*, по терминологии автора). Основное назначение системы, по замыслу Дж. Буля, состояло в том, чтобы кодировать логические высказывания и сводить структуры логических умозаключений к простым выражениям, близким по форме к математическим формулам. Результатом формального расчета логического выражения является одно из двух логических значений: *истина* или *ложь*.

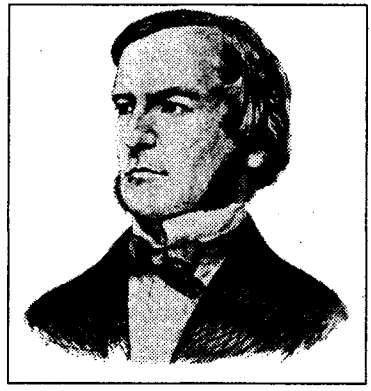


Рис. 2.4. Джордж Буль

Значение логической алгебры долгое время игнорировалось, поскольку ее приемы и методы не содержали практической пользы для науки и техники того времени. Однако, когда появилась принципиальная возможность создания средств вычислительной техники на электронной базе, операции, введенные Булем, оказались весьма полезны. Они изначально ориентированы на работу только с двумя сущностями: *истина* и *ложь*. Нетрудно понять, как они пригодились для работы с двоичным кодом, который в современных компьютерах тоже представляется всего двумя сигналами: *ноль* и *единица*.

Не вся система Джорджа Буля (как и не все предложенные им логические операции) были использованы при создании электронных вычислительных машин, но четыре основные операции: И (*пересечение*), ИЛИ (*объединение*), НЕ (*обращение*) и ИСКЛЮЧАЮЩЕЕ ИЛИ – лежат в основе работы всех видов процессоров современных компьютеров.

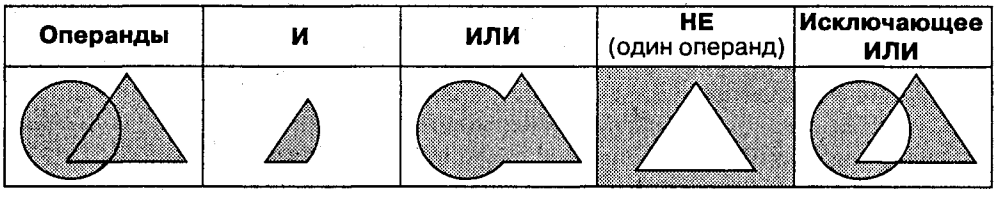


Рис. 2.5. Основные операции логической алгебры

2.2. Методы классификации компьютеров

Существует достаточно много систем классификации компьютеров. Мы рассмотрим лишь некоторые из них, сосредоточившись на тех, о которых наиболее часто упоминают в доступной технической литературе и средствах массовой информации.

Классификация по назначению

Классификация по назначению — один из наиболее ранних методов классификации. Он связан с тем, как компьютер применяется. По этому принципу различают *большие ЭВМ* (*электронно-вычислительные машины*), *мини-ЭВМ*, *микро-ЭВМ* и

персональные компьютеры, которые, в свою очередь, подразделяют на *массовые, деловые, портативные, развлекательные* и *рабочие станции*.

Большие ЭВМ. Это самые мощные компьютеры. Их применяют для обслуживания очень крупных организаций и даже целых отраслей народного хозяйства. За рубежом компьютеры этого класса называют *мэйнфреймами* (*mainframe*). В России за ними закрепился термин *большие ЭВМ*. Штат обслуживания большой ЭВМ достигает многих десятков человек. На базе таких суперкомпьютеров создают *вычислительные центры*, включающие в себя несколько отделов или групп.

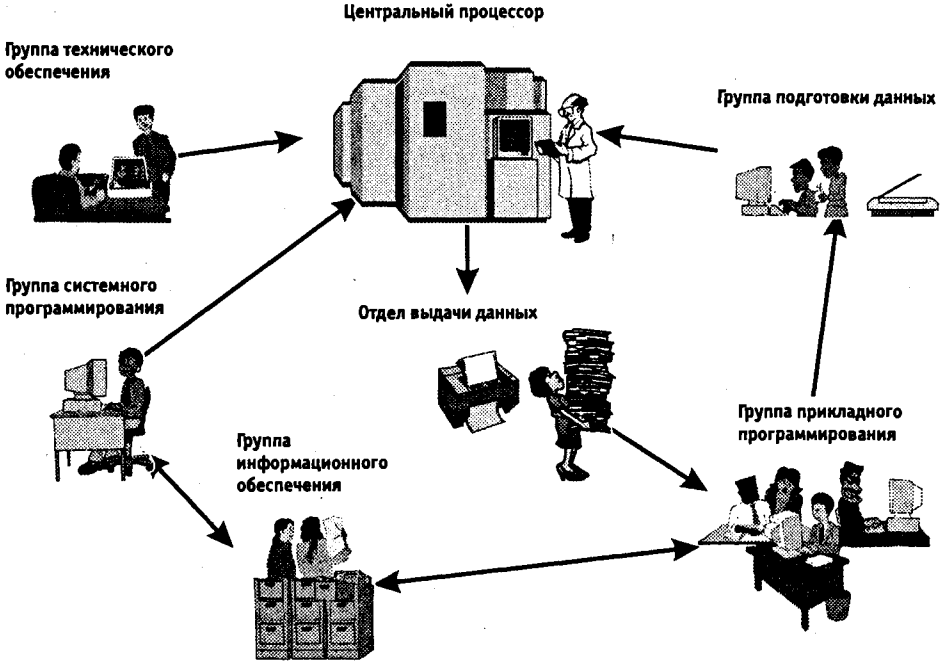


Рис. 2.6. Структура современного вычислительного центра на базе большой ЭВМ

Центральный процессор — основной блок ЭВМ, в котором непосредственно и происходит обработка данных и вычисление результатов. Обычно центральный процессор представляет собой несколько стоек аппаратуры и размещается в отдельном помещении, в котором соблюдаются повышенные требования по температуре, влажности, защищенности от электромагнитных помех, пыли и дыма.

Группа системного программирования занимается разработкой, отладкой и внедрением программного обеспечения, необходимого для функционирования самой вычислительной системы. Работников этой группы называют *системными программистами*. Они должны хорошо знать техническое устройство всех компонентов ЭВМ, поскольку их программы предназначены в первую очередь для управления физическими устройствами. Системные программы обеспечивают взаимодействие программ более высокого уровня с оборудованием, то есть группа системного программирования обеспечивает *программно-аппаратный интерфейс* вычислительной системы.

Группа прикладного программирования занимается созданием программ для выполнения конкретных операций с данными. Работников этой группы называют *прикладными программистами*. В отличие от системных программистов им не надо знать техническое устройство компонентов ЭВМ, поскольку их программы работают не с устройствами, а с программами, подготовленными системными программистами. С другой стороны, с их программами работают пользователи, то есть конкретные исполнители работ. Поэтому можно говорить о том, что группа прикладного программирования обеспечивает *пользовательский интерфейс* вычислительной системы.

Группа подготовки данных занимается подготовкой данных, с которыми будут работать программы, созданные прикладными программистами. Во многих случаях сотрудники этой группы сами вводят данные с помощью клавиатуры, но они могут выполнять и преобразование готовых данных из одного вида в другой. Например, они могут получать иллюстрации, нарисованные художниками на бумаге, и преобразовывать их в электронный вид с помощью специальных устройств, называемых *сканерами*.

Группа технического обеспечения занимается техническим обслуживанием всей вычислительной системы, ремонтом и наладкой устройств, а также подключением новых устройств, необходимых для работы прочих подразделений.

Группа информационного обеспечения обеспечивает технической информацией все прочие подразделения вычислительного центра по их заказу. Эта же группа создает и хранит архивы ранее разработанных программ и накопленных данных. Такие архивы называют *библиотеками программ* или *банками данных*.

Отдел выдачи данных получает данные от центрального процессора и преобразует их в форму, удобную для заказчика. Здесь информация распечатывается на печатающих устройствах (*принтерах*) или отображается на экранах дисплеев.

Большие ЭВМ отличаются высокой стоимостью оборудования и обслуживания, поэтому работа таких суперкомпьютеров организована по непрерывному циклу. Наиболее трудоемкие и продолжительные вычисления планируют на ночные часы, когда количество обслуживающего персонала минимально. В дневное время ЭВМ исполняет менее трудоемкие, но более многочисленные задачи. При этом для повышения эффективности компьютер работает одновременно с несколькими задачами и, соответственно, с несколькими пользователями. Он поочередно переключается с одной задачи на другую и делает это настолько быстро и часто, что у каждого пользователя создается впечатление, будто компьютер работает только с ним. Такое распределение ресурсов вычислительной системы носит название *принципа разделения времени*.

Мини-ЭВМ

От больших ЭВМ компьютеры этой группы отличаются уменьшенными размерами и, соответственно, меньшей производительностью и стоимостью. Такие компьютеры используются крупными предприятиями, научными учреждениями и некоторыми высшими учебными заведениями, сочетающими учебную деятельность с научной.

Мини-ЭВМ часто применяют для управления производственными процессами. Например, в механическом цехе компьютер может поддерживать ритмичность

подачи заготовок, узлов и комплектующих на рабочие места; управлять гибкими автоматизированными линиями и промышленными роботами; собирать информацию с инструментальных постов технического контроля и сигнализировать о необходимости замены изношенных инструментов и приспособлений; готовить данные для станков с числовым программным управлением; а также своевременно информировать цеховые и заводские службы о необходимости выполнения мероприятий по переналадке оборудования.

Тот же компьютер может сочетать управление производством с другими задачами. Например, он может помогать экономистам в осуществлении контроля над себестоимостью продукции, нормировщикам в оптимизации времени технологических операций, конструкторам в автоматизации проектирования станочных приспособлений, бухгалтерии в осуществлении учета первичных документов и подготовки регулярных отчетов для налоговых органов. Для организации работы с мини-ЭВМ тоже требуется специальный вычислительный центр, хотя и не такой многочисленный, как для больших ЭВМ.

Микро-ЭВМ

Компьютеры данного класса доступны многим предприятиям. Организации, использующие микро-ЭВМ, обычно не создают вычислительные центры. Для обслуживания такого компьютера им достаточно небольшой вычислительной лаборатории в составе нескольких человек. В число сотрудников вычислительной лаборатории обязательно входят программисты, хотя напрямую разработкой программ они не занимаются. Необходимые системные программы обычно покупают вместе с микро-ЭВМ, а разработку нужных прикладных программ заказывают более крупным вычислительным центрам или специализированным организациям.

Программисты вычислительной лаборатории занимаются внедрением приобретенного или заказанного программного обеспечения, выполняют его доводку и настройку, согласовывают его работу с другими программами и устройствами компьютера. Хотя программисты этой категории и не разрабатывают системные и прикладные программы, они могут вносить в них изменения, создавать или изменять отдельные фрагменты. Это требует высокой квалификации и универсальных знаний. Программисты, обслуживающие микро-ЭВМ, часто сочетают в себе качества системных и прикладных программистов одновременно.

Несмотря на относительно невысокую производительность по сравнению с большими ЭВМ, микро-ЭВМ находят применение и в крупных вычислительных центрах. Там им поручают вспомогательные операции, для которых нет смысла использовать дорогие суперкомпьютеры. К таким задачам, например, относится предварительная подготовка данных.

Персональные компьютеры (ПК)

Эта категория компьютеров получила особо бурное развитие в течение последних двадцати лет. Из названия видно, что такой компьютер предназначен для обслуживания одного рабочего места. Как правило, с персональным компьютером работает один человек. Несмотря на свои небольшие размеры и относительно невысо-

кую стоимость, современные персональные компьютеры обладают немалой производительностью. Многие современные персональные модели превосходят большие ЭВМ 70-х годов, мини-ЭВМ 80-х годов и микро-ЭВМ первой половины 90-х годов. Персональный компьютер (*Personal Computer, PC*) вполне способен удовлетворить большинство потребностей малых предприятий и отдельных лиц.

Особенно широкую популярность персональные компьютеры получили после 1995 года в связи с бурным развитием Интернета. Персонального компьютера вполне достаточно для использования всемирной сети в качестве источника научной, справочной, учебной, культурной и развлекательной информации. Персональные компьютеры являются также удобным средством автоматизации учебного процесса по любым дисциплинам, средством организации дистанционного (заочного) обучения и средством организации досуга. Они вносят большой вклад не только в производственные, но и в социальные отношения. Их нередко используют для организации надомной трудовой деятельности, что особенно важно в условиях ограниченной трудозанятости.

До последнего времени модели персональных компьютеров условно рассматривали в двух категориях: *бытовые ПК* и *профессиональные ПК*. Бытовые модели, как правило, имели меньшую производительность, но в них были приняты особые меры для работы с цветной графикой и звуком, чего не требовалось для профессиональных моделей. В связи с достигнутым в последние годы резким удешевлением средств вычислительной техники границы между профессиональными и бытовыми моделями в значительной степени стерлись, и сегодня в качестве бытовых нередко используют высокопроизводительные профессиональные модели, а профессиональные модели, в свою очередь, комплектуют устройствами для воспроизведения мультимедийной информации, что ранее было характерно для бытовых устройств.

Под термином *мультимедиа* подразумевается сочетание нескольких видов данных в одном документе (текстовые, графические, музыкальные и видеоданные) или совокупность устройств для воспроизведения этого комплекса данных.

С 1999 по 2002 год в области персональных компьютеров действовали международные сертификационные стандарты — *спецификации PC99–PC2002*. Они регламентировали принципы классификации персональных компьютеров и оговаривали минимальные и рекомендуемые требования к каждой из категорий. Стандарты устанавливали следующие категории персональных компьютеров:

- *Consumer PC* (массовый ПК);
- *Office PC* (деловой ПК);
- *Mobile PC* (портативный ПК);
- *Workstation PC* (рабочая станция);
- *Entertainmemt PC* (развлекательный ПК).

Каждая категория имела свои особенности: для *портативных ПК* обязательным было наличие средств компьютерной связи, в категории *рабочих станций* предъявлялись повышенные требования к устройствам хранения данных, а в категории *развлекательных ПК* — к средствам воспроизведения графики и звука.

Одна из целей такой стандартизации состояла и в том, чтобы наметить пути дальнейшего развития и совершенствования персональных компьютеров. Однако развитие аппаратных средств персонального компьютера привело к постепенному размытию границ между разными категориями, а планы развития часто не оправдывались. Поэтому обновление этих стандартов было прекращено, хотя при приобретении компьютера для конкретных задач классификацию, введенную этими стандартами, все еще полезно держать в голове.

Другие виды классификации компьютеров

Классификация по уровню специализации. По уровню специализации компьютеры делят на *универсальные* и *специализированные*. На базе универсальных компьютеров можно собирать вычислительные системы произвольного состава (состав компьютерной системы называется *конфигурацией*). Так, например, один и тот же персональный компьютер можно использовать для работы с текстами, музыкой, графикой, фото- и видеоматериалами.

Специализированные компьютеры предназначены для решения конкретного круга задач. К таким компьютерам относятся, например, бортовые компьютеры автомобилей, судов, самолетов, космических аппаратов. Бортовые компьютеры управляют средствами ориентации и навигации, осуществляют контроль состояния бортовых систем, выполняют некоторые функции автоматического управления и связи, а также большинство функций по оптимизации параметров работы систем объекта (например, оптимизацию расхода топлива в зависимости от конкретных условий движения объекта). Специализированные мини-ЭВМ, ориентированные на работу с графикой, называют *графическими станциями*. Их используют при подготовке кино- и видеофильмов, а также рекламной продукции. Специализированные компьютеры, объединяющие компьютеры предприятия в одну сеть, называют *файловыми серверами*. Компьютеры, обеспечивающие передачу информации между различными участниками всемирной компьютерной сети, называют *сетевыми серверами*.

Во многих случаях с задачами специализированных компьютерных систем могут справляться и обычные универсальные компьютеры, но считается, что использование специализированных систем все-таки эффективнее. Критерием оценки эффективности выступает отношение производительности оборудования к величине его стоимости.

Классификация по типоразмерам. Персональные компьютеры можно классифицировать по типоразмерам. Так, различают *настольные* (*desktop*), *портативные* (*notebook*) и *карманные* (*palmtop*) модели.

Настольные модели распространены наиболее широко. Они являются принадлежностью рабочего места. Эти модели отличаются простотой изменения конфигурации за счет несложного подключения дополнительных внешних приборов или установки дополнительных внутренних компонентов. Достаточные размеры корпуса в настольном исполнении позволяют выполнять большинство подобных работ без привлечения специалистов, а это позволяет настраивать компьютерную систему оптимально для решения именно тех задач, для которых она была приобретена.

Портативные модели удобны для транспортировки. Их используют бизнесмены, коммерсанты, руководители предприятий и организаций, проводящие много времени в командировках и переездах. С портативным компьютером можно работать при отсутствии рабочего места. Особая привлекательность портативных компьютеров связана с тем, что их можно использовать в качестве средства связи. Подключив такой компьютер к телефонной сети, можно из любой географической точки установить обмен данными между ним и центральным компьютером своей организации. Так производят обмен данными, передачу приказов и распоряжений, получение коммерческих данных, докладов и отчетов. Для эксплуатации на рабочем месте портативные компьютеры не очень удобны, но их можно подключать к настольным компьютерам, используемым стационарно.

Карманные модели выполняют функции «интеллектуальных записных книжек». Они позволяют хранить оперативные данные и получать к ним быстрый доступ. Некоторые карманные модели имеют жестко встроенное программное обеспечение, что облегчает непосредственную работу, но снижает гибкость в выборе прикладных программ.

Мобильные вычислительные устройства сочетают в себе функции карманных моделей компьютеров и средств мобильной связи (сотовых радиотелефонов). Их отличительная особенность — возможность мобильной работы с Интернетом, а в ближайшем будущем и возможность приема телевизионных передач. Дополнительно *МВУ* комплектуют средствами связи по инфракрасному лучу, благодаря которым эти карманные устройства могут обмениваться данными с настольными ПК и друг с другом.

Классификация по совместимости. В мире существует множество различных видов и типов компьютеров. Они выпускаются разными производителями, собираются из разных деталей, работают с разными программами. При этом очень важным вопросом становится *совместимость* различных компьютеров между собой. От совместимости зависит взаимозаменяемость узлов и приборов, предназначенных для разных компьютеров, возможность переноса программ с одного компьютера на другой и возможность совместной работы разных типов компьютеров с одними и теми же данными.

Аппаратная совместимость. По аппаратной совместимости различают так называемые *аппаратные платформы.* В области персональных компьютеров сегодня наиболее широко распространены две аппаратные платформы — *IBM PC* и *Apple Macintosh.* Кроме них существуют и другие платформы, распространенность которых ограничивается отдельными регионами или отдельными отраслями. Принадлежность компьютеров к одной аппаратной платформе повышает совместимость между ними, а принадлежность к разным платформам — понижает.

Кроме аппаратной совместимости существуют и другие виды совместимости: *совместимость на уровне операционной системы, программная совместимость, совместимость на уровне данных.*

Классификация по типу используемого процессора. *Процессор* — основной компонент любого компьютера. В электронно-вычислительных машинах это специальный

блок, а в персональных компьютерах — специальная микросхема, которая выполняет все вычисления в компьютере. Даже если компьютеры принадлежат одной аппаратной платформе, они могут различаться по типу используемого процессора. Основные типы процессоров для платформы *IBM PC* мы рассмотрим в соответствующем разделе, а здесь укажем на то, что тип используемого процессора в значительной (хотя и не в полной) мере характеризует технические свойства компьютера.

2.3. Состав вычислительной системы

Состав вычислительной системы называется *конфигурацией*. Аппаратные и программные средства вычислительной техники принято рассматривать отдельно. Соответственно, отдельно рассматривают *аппаратную конфигурацию* вычислительных систем и их *программную конфигурацию*. Такой принцип разделения имеет для информатики особое значение, поскольку очень часто решение одних и тех же задач может обеспечиваться как аппаратными, так и программными средствами. Критериями выбора аппаратного или программного решения являются производительность и эффективность.

☑ Обычно принято считать, что аппаратные решения в среднем оказываются дороже, зато реализация программных решений требует более высокой квалификации персонала.

Аппаратное обеспечение

К *аппаратному обеспечению* вычислительных систем относятся устройства и приборы, образующие аппаратную конфигурацию. Современные компьютеры и вычислительные комплексы имеют блочно-модульную конструкцию — аппаратную конфигурацию, необходимую для исполнения конкретных видов работ, можно собирать из готовых узлов и блоков.

По способу расположения устройств относительно *центрального процессорного устройства* (*ЦПУ — Central Processing Unit, CPU*) различают *внутренние* и *внешние* устройства. Внешними, как правило, являются большинство устройств ввода-вывода данных (их также называют *периферийными* устройствами) и некоторые устройства, предназначенные для длительного хранения данных.

Согласование между отдельными узлами и блоками выполняют с помощью переходных аппаратно-логических устройств, называемых *аппаратными интерфейсами*. Стандарты на аппаратные интерфейсы в вычислительной технике называют *протоколами*. Таким образом, *протокол — это совокупность технических условий, которые должны быть обеспечены разработчиками устройств для успешного согласования их работы с другими устройствами.*

Многочисленные интерфейсы, присутствующие в архитектуре любой вычислительной системы, можно условно разделить на две большие группы: *последовательные* и *параллельные*. Через последовательный интерфейс данные передаются последовательно, бит за битом, а через параллельный — одновременно группами битов. Количество битов, участвующих в одной посылке, определяется разрядностью интерфейса, например, восьмиразрядные параллельные интерфейсы передают один байт (8 бит) за один цикл.

Параллельные интерфейсы обычно имеют более сложное устройство, чем последовательные, но обеспечивают более высокую производительность. Их применяют там, где важна скорость передачи данных: для подключения печатающих устройств, устройств ввода графической информации, устройств записи данных на внешний носитель и т. п. Производительность параллельных интерфейсов измеряют *байтами в секунду* (байт/с; Кбайт/с; Мбайт/с).

Устройство последовательных интерфейсов проще; как правило, для них не надо синхронизировать работу передающего и принимающего устройства (поэтому их часто называют *асинхронными интерфейсами*). Первоначально пропускная способность последовательных интерфейсов была меньше, а коэффициент полезного действия — ниже. Из-за отсутствия синхронизации посылок полезные данные предваряют и завершают посылками служебных данных, то есть на один байт полезных данных могут приходиться 1–3 служебных бита (состав и структуру посылки определяет конкретный протокол).

Поскольку обмен данными через последовательные устройства производится не байтами, а битами, их производительность измеряют битами в секунду (бит/с, Кбит/с, Мбит/с). Несмотря на кажущуюся простоту перевода единиц измерения скорости последовательной передачи в единицы измерения скорости параллельной передачи данных путем механического деления на 8, такой пересчет не выполняют, поскольку он не корректен из-за наличия служебных данных. В крайнем случае, с поправкой на служебные данные, иногда скорость последовательных устройств выражают в *знаках в секунду* или, что то же самое, в *символах в секунду* (с/с), но эта величина имеет не технический, а справочный, потребительский характер.

Первоначально последовательные интерфейсы применяли для подключения «медленных» устройств (простейших устройств печати низкого качества, устройств ввода и вывода знаковой и сигнальной информации, контрольных датчиков, малопроизводительных устройств связи и т. п.), а также в тех случаях, когда отсутствуют существенные ограничения по продолжительности обмена данными.

Однако с развитием техники появились новые, высокоскоростные последовательные интерфейсы, не уступающие параллельным, а нередко и превосходящие их по пропускной способности. Сегодня последовательные интерфейсы применяют для подключения к компьютеру любых типов устройств.

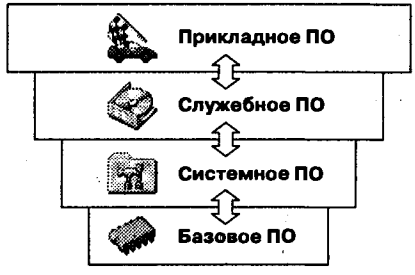
Программное обеспечение

Программы — это упорядоченные последовательности команд. Конечная цель любой компьютерной программы — управление аппаратными средствами. Даже если на первый взгляд программа никак не взаимодействует с оборудованием, не требует никакого ввода данных с устройств ввода и не осуществляет вывод данных на устройства вывода, все равно ее работа основана на управлении аппаратными устройствами компьютера.

Программное и аппаратное обеспечение в компьютере работают в неразрывной связи и в непрерывном взаимодействии. Несмотря на то что мы рассматриваем эти две категории отдельно, нельзя забывать, что между ними существует диалектическая связь и раздельное их рассмотрение является по меньшей мере условным.

Состав программного обеспечения вычислительной системы называют *программной конфигурацией*. Между программами, как и между физическими узлами и блоками существует взаимосвязь — многие программы работают, опираясь на другие программы более низкого уровня, то есть мы можем говорить о межпрограммном *интерфейсе*. Возможность существования такого интерфейса тоже основана на существовании технических условий и протоколов взаимодействия, а на практике он обеспечивается распределением программного обеспечения на несколько взаимодействующих между собой уровней.

Уровни программного обеспечения представляют собой пирамидальную конструкцию. Каждый следующий уровень опирается на программное обеспечение предшествующих уровней. Такое членение удобно для всех этапов работы с вычислительной системой, начиная с установки программ до практической эксплуатации и технического обслуживания. Обратите внимание на то, что каждый вышележащий уровень повышает функциональность всей системы. Так, например, вычислительная система с программным обеспечением базового уровня не способна выполнять большинство функций, но позволяет установить системное программное обеспечение.



Базовый уровень. Самый низкий уровень программного обеспечения представляет *базовое программное обеспечение*. Оно отвечает за взаимодействие с *базовыми аппаратными средствами*. Как правило, базовые программные средства непосредственно входят в состав базового оборудования и хранятся в специальных микросхемах, называемых *постоянными запоминающими устройствами* (*ПЗУ — Read Only Memory, ROM*). Программы и данные записываются («прошиваются») в микросхемы ПЗУ на этапе производства и не могут быть изменены в процессе эксплуатации.

В тех случаях, когда изменение базовых программных средств во время эксплуатации является технически целесообразным, вместо микросхем ПЗУ применяют *перепрограммируемые постоянные запоминающие устройства* (*ППЗУ — Erasable and Programmable Read Only Memory, EPROM*). В этом случае изменение содержания ПЗУ можно выполнять как непосредственно в составе вычислительной системы (такая технология называется *флэш-технологией*), так и вне нее, на специальных устройствах, называемых *программаторами*.

Системный уровень. Системный уровень — переходный. Программы, работающие на этом уровне, обеспечивают взаимодействие прочих программ компьютерной системы с программами базового уровня и непосредственно с аппаратным обеспечением, то есть выполняют «посреднические» функции.

От программного обеспечения этого уровня во многом зависят эксплуатационные показатели всей вычислительной системы в целом. Так, например, при подключении к вычислительной системе нового оборудования на системном уровне должна быть установлена программа, обеспечивающая для других программ взаимосвязь

с этим оборудованием. Конкретные программы, отвечающие за взаимодействие с конкретными устройствами, называются *драйверами устройств* — они входят в состав программного обеспечения системного уровня.

Другой класс программ системного уровня отвечает за взаимодействие с пользователем. Именно благодаря им он получает возможность вводить данные в вычислительную систему, управлять ее работой и получать результат в удобной для себя форме. Эти программные средства называют *средствами обеспечения пользовательского интерфейса*. От них напрямую зависит удобство работы с компьютером и производительность труда на рабочем месте.

Совокупность программного обеспечения системного уровня образует *ядро операционной системы компьютера*. Полное понятие операционной системы мы рассмотрим несколько позже, а здесь только отметим, что если компьютер оснащен программным обеспечением системного уровня, то он уже подготовлен к установке программ более высоких уровней, к взаимодействию программных средств с оборудованием и, самое главное, к взаимодействию с пользователем. То есть *наличие ядра операционной системы — непременное условие для возможности практической работы человека с вычислительной системой.*

Служебный уровень. Программное обеспечение этого уровня взаимодействует как с программами базового уровня, так и с программами системного уровня. Основное назначение служебных программ (их также называют *утилитами*) состоит в автоматизации работ по проверке, наладке и настройке компьютерной системы. Во многих случаях они используются для расширения или улучшения функций системных программ. Некоторые служебные программы (как правило, это программы обслуживания) изначально включают в состав операционной системы, но большинство служебных программ являются для операционной системы внешними и служат для расширения ее функций.

В разработке и эксплуатации служебных программ существует два альтернативных направления: *интеграция с операционной системой* и *автономное функционирование*. В первом случае служебные программы могут изменять потребительские свойства системных программ, делая их более удобными для практической работы. Во втором случае они слабо связаны с системным программным обеспечением, но предоставляют пользователю больше возможностей для персональной настройки их взаимодействия с аппаратным и программным обеспечением.

Прикладной уровень. Программное обеспечение прикладного уровня представляет собой комплекс прикладных программ, с помощью которых на данном рабочем месте выполняются конкретные задания. Спектр этих заданий необычайно широк: от производственных до творческих и развлекательно-обучающих. Огромный функциональный диапазон возможных приложений средств вычислительной техники обусловлен наличием прикладных программ для разных видов деятельности.

Поскольку между прикладным программным обеспечением и системным существует непосредственная взаимосвязь (первое опирается на второе), то можно утверждать, что универсальность вычислительной системы, доступность прикладного программного обеспечения и широта функциональных возможностей компьютера

напрямую зависят от типа используемой операционной системы, от того, какие системные средства содержит ее ядро, как она обеспечивает взаимодействие триединого комплекса *человек — программы — оборудование*.

Классификация прикладных программных средств

Текстовые редакторы. Основные функции этого класса прикладных программ заключаются во вводе и редактировании текстовых данных. Дополнительные функции состоят в автоматизации процессов ввода и редактирования. Для операций ввода, вывода и сохранения данных текстовые редакторы вызывают и используют системное программное обеспечение. Впрочем, это характерно и для всех прочих видов прикладных программ, и в дальнейшем мы не будем специально указывать на этот факт. С этого класса прикладных программ обычно начинают знакомство с программным обеспечением и на нем отрабатывают первичные навыки взаимодействия с компьютерной системой.

Текстовые процессоры. Основное отличие текстовых процессоров от текстовых редакторов в том, что они позволяют не только вводить и редактировать текст, но и *форматировать* его, то есть оформлять. Соответственно, к основным средствам текстовых процессоров относятся средства обеспечения взаимодействия текста, графики, таблиц и других объектов, составляющих итоговый документ, а к дополнительным — средства автоматизации процесса форматирования.

Современный стиль работы с документами подразумевает два альтернативных подхода — работу с бумажными документами и работу с электронными документами (по безбумажной технологии). Поэтому, говоря о форматировании документов средствами текстовых процессоров, надо иметь в виду два принципиально разных направления — форматирование документов, предназначенных для печати, и форматирование электронных документов, предназначенных для отображения на экране. Приемы и методы в этих случаях существенно различаются. Соответственно, различаются и текстовые процессоры, хотя многие из них успешно сочетают оба подхода.

Графические редакторы. Это обширный класс программ, предназначенных для создания и (или) обработки графических изображений. В данном классе различают следующие категории: *растровые редакторы*, *векторные редакторы* и программные средства для создания и обработки трехмерной графики (*3D-редакторы*).

Растровые редакторы применяют в тех случаях, когда графический объект представлен в виде комбинации точек, образующих растр и обладающих свойствами яркости и цвета. Такой подход эффективен в тех случаях, когда графическое изображение имеет много полутонов и информация о цвете элементов, составляющих объект, важнее, чем информация об их форме. Это характерно для фотографических и полиграфических изображений. Растровые редакторы широко применяются для обработки изображений, их ретуши, создания фотоэффектов и художественных композиций (коллажей).

Возможности создания новых изображений средствами растровых редакторов ограниченны и не всегда удобны. В большинстве случаев художники предпочитают пользоваться традиционными инструментами, после чего вводить рисунок в ком-

пьютер с помощью специальных аппаратных средств (*сканеров*) и завершать работу с помощью растрового редактора путем применения спецэффектов.

Векторные редакторы отличаются от растровых способом представления данных об изображении. Элементарным объектом векторного изображения является не точка, а линия. Такой подход характерен для чертежно-графических работ, в которых форма линий имеет большее значение, чем информация о цвете отдельных точек, составляющих ее. В векторных редакторах каждая линия рассматривается как математическая кривая третьего порядка и, соответственно, представляется не комбинацией точек, а математической формулой (в компьютере хранятся числовые коэффициенты этой формулы). Такое представление намного компактнее, чем растровое, соответственно данные занимают много меньше места, однако построение любого объекта выполняется не простым отображением точек на экране, а сопровождается непрерывным пересчетом параметров кривой в координаты экранного или печатного изображения. Соответственно, работа с векторной графикой требует более производительных вычислительных систем.

Из элементарных объектов (линий) создаются простейшие геометрические объекты (примитивы) из которых, в свою очередь, составляются законченные композиции. Художественная иллюстрация, выполненная средствами векторной графики, может содержать десятки тысяч простейших объектов, взаимодействующих друг с другом.

Векторные редакторы удобны для создания изображений, но практически не используются для обработки готовых рисунков. Они нашли широкое применение в рекламном бизнесе, их применяют для оформления обложек полиграфических изданий и всюду, где стиль художественной работы близок к чертежному.

Редакторы трехмерной графики используют для создания трехмерных композиций. Они имеют две характерные особенности. Во-первых, они позволяют гибко управлять взаимодействием свойств поверхности изображаемых объектов со свойствами источников освещения и, во-вторых, позволяют создавать трехмерную анимацию. Поэтому редакторы трехмерной графики нередко называют также *3D-аниматорами*.

Системы управления базами данных. Базами данных называют огромные массивы данных, организованных в табличные структуры. Основными функциями систем управления базами данных являются:

- создание пустой (незаполненной) структуры базы данных;
- предоставление средств ее заполнения или импорта данных из таблиц другой базы;
- обеспечение возможности доступа к данным, а также предоставление средств поиска и фильтрации.

Многие системы управления базами данных дополнительно предоставляют возможности проведения простейшего анализа данных и их обработки. В результате возможно создание новых таблиц баз данных на основе имеющихся. В связи с широким распространением сетевых технологий к современным системам управления базами данных предъявляется также требование возможности работы с удаленными и распределенными ресурсами, находящимися на серверах всемирной компьютерной сети.

Электронные таблицы. Электронные таблицы предоставляют комплексные средства для хранения различных типов данных и их обработки. В некоторой степени они аналогичны системам управления базами данных, но основной акцент смещен не на хранение массивов данных и обеспечение к ним доступа, а на преобразование данных, причем в соответствии с их внутренним содержанием.

В отличие от баз данных, которые обычно содержат широкий спектр типов данных (от числовых и текстовых до мультимедийных), для электронных таблиц характерна повышенная сосредоточенность на числовых данных. Зато электронные таблицы предоставляют более широкий спектр методов для работы с данными числового типа.

Основное свойство электронных таблиц состоит в том, что при изменении содержания любых ячеек таблицы может происходить автоматическое изменение содержания во всех прочих ячейках, связанных с измененными соотношением, заданным математическими или логическими выражениями (формулами). Простота и удобство работы с электронными таблицами снискали им широкое применение в сфере бухгалтерского учета, в качестве универсальных инструментов анализа финансовых, сырьевых и товарных рынков, доступных средств обработки результатов технических испытаний, то есть всюду, где необходимо автоматизировать регулярно повторяющиеся вычисления достаточно больших объемов числовых данных.

Системы автоматизированного проектирования (CAD-системы). Предназначены для автоматизации проектно-конструкторских работ. Применяются в машиностроении, приборостроении, архитектуре. Кроме чертежно-графических работ эти системы позволяют проводить простейшие расчеты (например, расчеты прочности деталей) и выбор готовых конструктивных элементов из обширных баз данных.

Отличительная особенность *CAD*-систем состоит в автоматическом обеспечении на всех этапах проектирования технических условий, норм и правил, что освобождает конструктора (или архитектора) от работ нетворческого характера. Например, в машиностроении *CAD*-системы способны на базе сборочного чертежа изделия автоматически выполнить рабочие чертежи деталей, подготовить необходимую технологическую документацию с указанием последовательности переходов механической обработки, назначить необходимые инструменты, станочные и контрольные приспособления, а также подготовить управляющие программы для станков с числовым программным управлением (ЧПУ), промышленных роботов и гибких автоматизированных линий. Сегодня системы автоматизированного проектирования являются необходимым компонентом, без которого теряется эффективность реализации гибких производственных систем (ГПС) и автоматизированных систем управления технологическими процессами (АСУТП).

Настольные издательские системы. Назначение программ этого класса состоит в автоматизации процесса верстки полиграфических изданий. Этот класс программного обеспечения занимает промежуточное положение между текстовыми процессорами и системами автоматизированного проектирования.

Теоретически текстовые процессоры предоставляют средства для внедрения в текстовый документ объектов другой природы, например объектов векторной и растровой графики, а также позволяют управлять взаимодействием между параметрами

текста и параметрами внедренных объектов. Однако на практике для изготовления полиграфической продукции эти средства либо функционально недостаточны с точки зрения требований полиграфии, либо недостаточно удобны для производительной работы.

От текстовых процессоров настольные издательские системы отличаются расширенными средствами управления взаимодействием текста с параметрами страницы и с графическими объектами. С другой стороны, они отличаются пониженными функциональными возможностями по автоматизации ввода и редактирования текста. Типичный прием использования настольных издательских систем состоит в том, что их применяют к документам, прошедшим предварительную обработку в текстовых процессорах и графических редакторах.

Экспертные системы. Предназначены для анализа данных, содержащихся в *базах знаний*, и выдачи рекомендаций по запросу пользователя. Такие системы применяют в тех случаях, когда исходные данные хорошо формализуются, но для принятия решения требуются обширные специальные знания. Характерными областями использования экспертных систем являются юриспруденция, медицина, фармакология, химия. По совокупности признаков заболевания медицинские экспертные системы помогают установить диагноз и назначить лекарства, дозировку и программу лечебного курса. По совокупности признаков события юридические экспертные системы могут дать правовую оценку и предложить порядок действий как для стороны обвинения, так и для стороны защиты.

Характерной особенностью экспертных систем является их способность к *саморазвитию*. Исходные данные хранятся в базе знаний в виде *фактов*, между которыми с помощью специалистов-экспертов устанавливается определенная система *отношений*. Если на этапе тестирования экспертной системы устанавливается, что она дает некорректные рекомендации и заключения по конкретным вопросам или не может дать их вообще, это означает либо отсутствие важных фактов в ее базе, либо нарушения в логической системе отношений. И том и в другом случае экспертная система сама может сгенерировать достаточный набор запросов к эксперту и автоматически повысить свое качество.

С использованием экспертных систем связана особая область научно-технической деятельности, называемая *инженерией знаний*. Инженеры знаний — это специалисты особой квалификации, выступающие в качестве промежуточного звена между разработчиками экспертной системы (программистами) и ведущими специалистами в конкретных областях науки и техники (экспертами).

Web-редакторы. Это особый класс редакторов, объединяющих в себе свойства текстовых и графических редакторов. Они предназначены для создания и редактирования так называемых *Web-документов* (*Web-страниц Интернета*). *Web*-документы — это электронные документы, при подготовке которых следует учитывать ряд особенностей, связанных с приемом/передачей информации в Интернете.

Теоретически для создания *Web*-документов можно использовать обычные текстовые редакторы и процессоры, а также некоторые из графических редакторов векторной графики, но *Web*-редакторы обладают рядом полезных функций, повы-

шающих производительность труда *Web*-дизайнеров. Программы этого класса можно также эффективно использовать для подготовки электронных документов и мультимедийных изданий.

Браузеры (обозреватели, средства просмотра Web). К этой категории относятся программные средства, предназначенные для просмотра электронных документов, выполненных в формате *HTML* (документы этого формата используются в качестве *Web*-документов). Современные браузеры воспроизводят не только текст и графику. Они могут воспроизводить музыку, человеческую речь, обеспечивать прослушивание радиопередач в Интернете, просмотр видеоконференций, работу со службами электронной почты, с системой телеконференций (групп новостей) и многое другое.

Интегрированные системы делопроизводства. Представляют собой программные средства автоматизации рабочего места руководителя. К основным функциям подобных систем относятся функции создания, редактирования и форматирования простейших документов, централизация функций электронной почты, факсимильной и телефонной связи, диспетчеризация и мониторинг документооборота предприятия, координация деятельности подразделений, оптимизация административно-хозяйственной деятельности и поставка по запросу оперативной и справочной информации.

Бухгалтерские системы. Это специализированные системы, сочетающие в себе функции текстовых и табличных редакторов, электронных таблиц и систем управления базами данных. Предназначены для автоматизации подготовки первичных бухгалтерских документов предприятия и их учета, для ведения счетов плана бухгалтерского учета, а также для автоматической подготовки регулярных отчетов по итогам производственной, хозяйственной и финансовой деятельности в форме, принятой для предоставления в налоговые органы, внебюджетные фонды и органы статистического учета. Несмотря на то что теоретически все функции, характерные для бухгалтерских систем, можно исполнять и другими вышеперечисленными программными средствами, использование бухгалтерских систем удобно благодаря интеграции разных средств в одной системе.

При решении о внедрении на предприятии автоматизированной системы бухгалтерского учета необходимо учитывать необходимость наличия в ней средств адаптации при изменении нормативно-правовой базы. В связи с тем, что в данной области нормативно-правовая база в России отличается крайней нестабильностью и подвержена частым изменениям, возможность гибкой перенастройки системы является обязательной функцией, хотя это требует от пользователей системы повышенной квалификации.

Финансовые аналитические системы. Программы этого класса используются в банковских и биржевых структурах. Они позволяют контролировать и прогнозировать ситуацию на финансовых, товарных и сырьевых рынках, производить анализ текущих событий, готовить сводки и отчеты.

Геоинформационные системы (ГИС). Предназначены для автоматизации картографических и геодезических работ на основе информации, полученной топографическими или аэрокосмическими методами.

Системы видеомонтажа. Предназначены для цифровой обработки видеоматериалов, их монтажа, создания видеоэффектов, устранения дефектов, наложения звука, титров и субтитров.

Отдельные категории прикладных программных средств, обладающие своими развитыми внутренними системами классификации, представляют *обучающие*, *развивающие*, *справочные* и *развлекательные* системы и программы. Характерной особенностью этих классов программного обеспечения являются повышенные требования к мультимедийной составляющей (использование музыкальных композиций, средств графической анимации и видеоматериалов)

Классификация служебных программных средств

Диспетчеры файлов (файловые менеджеры). С помощью программ данного класса выполняется большинство операций, связанных с обслуживанием файловой структуры: копирование, перемещение и переименование файлов, создание каталогов (папок), удаление файлов и каталогов, поиск файлов и навигация в файловой структуре. Базовые программные средства, предназначенные для этой цели, обычно входят в состав программ системного уровня и устанавливаются вместе с операционной системой. Однако для повышения удобства работы с компьютером большинство пользователей устанавливают дополнительные служебные программы.

Средства сжатия данных (архиваторы). Предназначены для создания архивов. Архивирование данных упрощает их хранение за счет того, что большие группы файлов и каталогов сводятся в один архивный файл. При этом повышается и эффективность использования носителя за счет того, что архивные файлы обычно имеют повышенную плотность записи информации. Архиваторы часто используют для создания резервных копий ценных данных.

Средства просмотра и воспроизведения. Обычно для работы с файлами данных необходимо загрузить их в «родительскую» прикладную систему, с помощью которой они были созданы. Это дает возможность просматривать документы и вносить в них изменения. Но в тех случаях, когда требуется только просмотр без редактирования, удобно использовать более простые и более универсальные средства, позволяющие просматривать документы разных типов.


В тех случаях, когда речь идет о звукозаписи или видеозаписи, вместо термина *просмотр* применяют термин *воспроизведение* документов.

Средства диагностики. Предназначены для автоматизации процессов диагностики программного и аппаратного обеспечения. Они выполняют необходимые проверки и выдают собранную информацию в удобном и наглядном виде. Их используют не только для устранения неполадок, но и для оптимизации работы компьютерной системы.

Средства контроля (мониторинга). Программные средства контроля иногда называют *мониторами*. Они позволяют следить за процессами, происходящими в компьютерной системе. При этом возможны два подхода: наблюдение в реальном режиме времени или контроль с записью результатов в специальном протокольном файле. Первый подход обычно используют при изыскании путей для оптимизации работы

вычислительной системы и повышения ее эффективности. Второй подход используют в тех случаях, когда мониторинг выполняется автоматически и (или) дистанционно. В последнем случае результаты мониторинга можно передать удаленной службе технической поддержки для установления причин конфликтов в работе программного и аппаратного обеспечения.

Средства мониторинга, работающие в режиме реального времени, особенно полезны для практического изучения приемов работы с компьютером, поскольку позволяют наглядно отображать те процессы, которые обычно скрыты от глаз пользователя.

Мониторы установки. Программы этой категории предназначены для контроля над установкой программного обеспечения. Необходимость в данном программном обеспечении связана с тем, что между различными категориями программного обеспечения могут устанавливаться связи. Вертикальные связи (между уровнями) являются необходимым условием функционирования всех компьютеров. Горизонтальные связи (внутри уровней) характерны для компьютеров, работающих с операционными системами, поддерживающими принцип совместного использования одних и тех же ресурсов разными программными средствами. И в тех и в других случаях при установке или удалении программного обеспечения могут происходить нарушения работоспособности прочих программ.

Мониторы установки следят за состоянием и изменением окружающей программной среды, отслеживают и протоколируют образование новых связей и позволяют восстанавливать связи, утраченные в результате удаления ранее установленных программ.

Простейшие средства управления установкой и удалением программ обычно входят в состав операционной системы и размещаются на системном уровне программного обеспечения, однако они редко бывают достаточны. Поэтому в вычислительных системах, требующих повышенной надежности, используют дополнительные служебные программы.

Средства коммуникации (коммуникационные программы). С появлением электронной связи и компьютерных сетей программы этого класса приобрели очень большое значение. Они позволяют устанавливать соединения с удаленными компьютерами, обслуживают передачу сообщений электронной почты, работу с телеконференциями (группами новостей), обеспечивают пересылку факсимильных сообщений и выполняют множество других операций в компьютерных сетях.

Средства обеспечения компьютерной безопасности. К этой весьма широкой категории относятся средства пассивной и активной защиты данных от повреждения, а также средства защиты от несанкционированного доступа, просмотра и изменения данных.

В качестве *средств пассивной защиты* используют служебные программы, предназначенные для резервного копирования. Нередко они обладают и базовыми свойствами диспетчеров архивов (архиваторов). В качестве *средств активной защиты* применяют *антивирусное программное обеспечение*. Для защиты данных от несанкционированного доступа, их просмотра и изменения служат специальные системы, основанные на криптографии.

Понятие об информационном и математическом обеспечении вычислительных систем

Наряду с аппаратным и программным обеспечением средств вычислительной техники в некоторых случаях целесообразно рассматривать *информационное обеспечение*, под которым понимают совокупность программ и предварительно подготовленных данных, необходимых для работы данных программ.

Рассмотрим, например, систему автоматической проверки орфографии в редактируемом тексте. Ее работа заключается в том, что лексические единицы исходного текста сравниваются с заранее заготовленным эталонным массивом данных (словарем). В данном случае для успешной работы системы необходимо иметь кроме аппаратного и программного обеспечения специальные наборы словарей, подключаемые извне. Это пример *информационного обеспечения* вычислительной техники.

В специализированных компьютерных системах (бортовых компьютерах автомобилей, судов, ракет, самолетов, космических летательных аппаратов и т. п.) совокупность программного и информационного обеспечения называют *математическим обеспечением*. Как правило, оно «жестко» записывается в микросхемы ПЗУ и может быть изменено только путем замены ПЗУ или его перепрограммирования на специальном оборудовании.

Подведение итогов

Вычислительная техника прошла те же исторические этапы эволюции, которые прошли и все прочие технические устройства: от ручных приспособлений к механическим устройствам и далее к гибким автоматическим системам. Современный компьютер — это прибор. Его принцип действия — электронный, а назначение — автоматизация операций с данными. Гибкость автоматизации основана на том, что операции с данными выполняются по заранее заготовленным и легко сменяемым программам. Универсальность компьютеров основана на том, что любые типы данных представляются в нем с помощью универсального двоичного кодирования.

Работа компьютерной системы протекает в непрерывном взаимодействии аппаратных и программных средств. Физически аппаратные средства согласуются друг с другом с помощью механических и электрических разъемов и контактов. Логически они согласуются друг с другом с помощью программ, называемых *драйверами устройств*.

Работа компьютерных программ имеет многоуровневый характер. Программы низшего (базового) уровня занимаются только взаимодействием с базовыми аппаратными средствами и согласованием их работы. Ключевая роль программ базового уровня проявляется в момент первичного запуска компьютера.

Программы *системного уровня* опираются на программы базового уровня и обеспечивают взаимодействие пользователя с оборудованием, взаимодействие дополнительного оборудования с базовым, а также предоставляют возможность для установки и работы программ более высоких уровней.

Программы *служебного уровня* выполняют обслуживание компьютерной системы, обеспечивают ее контроль и настройку. В своей работе они опираются на программы базового и системного уровней.

Программы *прикладного* уровня используются человеком для исполнения практических задач с помощью компьютера. Эти программы опираются на программы нижележащих уровней.

Совокупность программ, установленных на компьютере, называется его *программной конфигурацией*. Совокупность оборудования, подключенного к компьютеру, называется его *аппаратной конфигурацией*. Несмотря на то что по своей архитектуре и функциональному назначению разные компьютеры могут быть весьма близки друг другу, найти два компьютера, имеющих одинаковые аппаратные и программные конфигурации, практически невозможно. На каждом рабочем месте программно-аппаратная конфигурация создается такой, чтобы наиболее эффективно решать конкретные практические задачи, характерные для данного рабочего места.

Вопросы для самоконтроля

1. В чем вы видите диалектический характер связи между *программным обеспечением* и *аппаратным*?
2. Назовите четыре основных уровня программного обеспечения. Каков порядок их взаимодействия?
3. К какому классу относятся программные средства, встроенные в видеомагнитофон, программируемую стиральную машину, СВЧ-плиту?
4. В чем преимущества и недостатки выполнения офисных работ (например, копировально-множительных) аппаратными и программными средствами?
5. Какие категории программного обеспечения могут быть использованы в работе малого предприятия и для каких целей?
6. Какие виды работ, характерные для крупного промышленного предприятия (например, машиностроительного завода), могут быть автоматизированы с помощью компьютеров? Какие категории программных средств для этого необходимы?
7. Назовите основные категории программного обеспечения, относящиеся к классу графических редакторов. В чем состоит принципиальная разница между этими категориями?
8. Что общего и в чем различие между понятиями *программное обеспечение* и *информационное обеспечение* средств вычислительной техники?

3.1. Базовая аппаратная конфигурация персонального компьютера

Персональный компьютер — универсальная техническая система. Его *конфигурацию* (состав оборудования) можно гибко изменять по мере необходимости. Тем не менее, существует понятие *базовой конфигурации*, которую считают типовой. В таком комплекте компьютер обычно поставляется. Понятие базовой конфигурации может меняться. В настоящее время в базовой конфигурации рассматривают четыре устройства (рис. 3.1):

- системный блок;
- монитор;
- клавиатура;
- мышь.

Системный блок

Системный блок представляет собой основной узел, внутри которого установлены наиболее важные компоненты. Устройства, находящиеся внутри системного блока, называют *внутренними*, а устройства, подключаемые к нему снаружи, — *внешними*. Внешние дополнительные устройства, предназначенные для ввода, вывода и длительного хранения данных, также называют *периферийными*.

По внешнему виду системные блоки различаются формой корпуса. Корпуса персональных компьютеров выпускают в горизонтальном (*desktop*) и вертикальном (*tower*) исполнении. Корпуса, имеющие вертикальное исполнение, различают по габаритам: *полноразмерный* (*big tower*), *среднеразмерный* (*midi tower*) и *малоразмерный* (*mini tower*). Среди корпусов, имеющих горизонтальное исполнение, выделяют *плоские* и *особо плоские* (*slim*).

Кроме формы, для корпуса важен параметр, называемый *форм-фактором*. От него зависят требования к размещаемым устройствам. Прежним стандартом корпуса

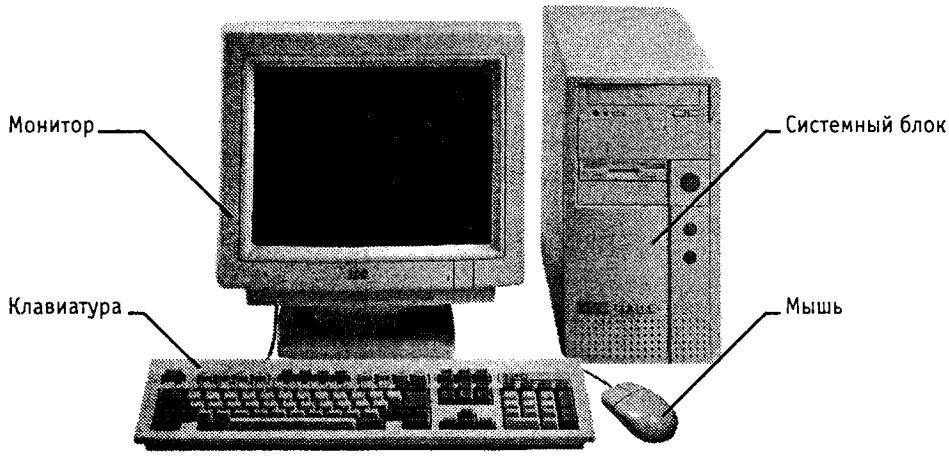


Рис. 3.1. Базовая конфигурация компьютерной системы

персональных компьютеров был форм-фактор *AT*, в настоящее время в основном используются корпуса форм-фактора *ATX*. Форм-фактор корпуса должен быть обязательно согласован с форм-фактором главной (системной) платы компьютера, так называемой *материнской платы* (см. ниже).

Корпуса персональных компьютеров поставляются вместе с блоком питания и, таким образом, мощность блока питания также является одним из параметров корпуса. Для массовых моделей достаточной является мощность блока питания 250–300 Вт.

Монитор

Монитор — устройство визуального представления данных. Это не единственно возможное, но главное устройство вывода. Его основными потребительскими параметрами являются: тип, размер и шаг маски экрана, максимальная частота регенерации изображения, класс защиты.

Сейчас наиболее распространены мониторы двух основных типов на основе электронно-лучевой трубки (ЭЛТ) и плоские жидкокристаллические (ЖК). ЭЛТ-мониторы обеспечивают лучшее качество изображения, но в пользу жидкокристаллических мониторов говорит их компактность, небольшой вес, идеально плоская поверхность экрана.

Размер монитора измеряется между противоположными углами видимой части экрана по диагонали. Единица измерения — дюймы. Стандартные размеры: 14"; 15"; 17"; 19"; 20"; 21". В настоящее время наиболее универсальными являются мониторы размером 15 (ЖК) и 17 дюймов (ЭЛТ), а для операций с графикой желательны мониторы размером 19–21 дюйм (ЭЛТ).

Изображение на экране ЭЛТ-монитора получается в результате облучения люминофорного покрытия остронаправленным пучком электронов, разогнанных в вакуумной колбе. Для получения цветного изображения люминофорное покрытие имеет точки или полоски трех типов, светящиеся красным, зеленым и синим цветом.

Чтобы на экране все три луча сходились строго в одну точку и изображение было четким, перед люминофором ставят маску — панель с регулярно расположенными отверстиями или щелями. Часть мониторов оснащена маской из вертикальных проволочек, что усиливает яркость и насыщенность изображения. Чем меньше шаг между отверстиями или щелями (*шаг маски*), тем четче и точнее полученное изображение. Шаг маски измеряют в долях миллиметра. В настоящее время наиболее распространены мониторы с шагом маски 0,24–0,26 мм. Устаревшие мониторы могут иметь шаг до 0,43 мм, что негативно сказывается на органах зрения при работе с компьютером. Модели повышенной стоимости могут иметь значение менее 0,24 мм.

На экране жидкокристаллического монитора изображение образуется в результате прохождения белого света лампы подсветки через ячейки, прозрачность которых зависит от приложенного напряжения. Элементарная триада состоит из трех ячеек зеленого, красного и синего цвета и соответствует одному пикселу экрана. Размер монитора по диагонали и разрешение экрана однозначно определяет размер такой триады и, тем самым, зернистость изображения.

Частота регенерации (*обновления*) изображения показывает, сколько раз в течение секунды монитор может полностью сменить изображение (поэтому ее также называют *частотой кадров*). Этот параметр зависит не только от монитора, но и от свойств и настроек *видеоадаптера* (см. ниже), хотя предельные возможности определяет все-таки монитор.

Частоту регенерации изображения измеряют в герцах (Гц). Чем она выше, тем четче и устойчивее изображение, тем меньше утомление глаз, тем больше времени можно работать с компьютером непрерывно. При частоте регенерации порядка 60 Гц мелкое мерцание изображения может быть заметно невооруженным глазом. Сегодня такое значение считается недопустимым. Для ЭЛТ-мониторов минимальным считают значение 75 Гц, нормативным — 85 Гц и комфортным — 100 Гц и более. У жидкокристаллических мониторов изображение более инерционно, так что мерцание подавляется автоматически. Для них частота обновления в 75 Гц уже считается комфортной.

Класс защиты монитора определяется стандартом, которому соответствует монитор с точки зрения требований техники безопасности. В настоящее время общепризнанными считаются следующие международные стандарты: *MPR-II*, *TCO-92*, *TCO-95*, *TCO-99* (приведены в хронологическом порядке). Стандарт *MPR-II* ограничил уровни электромагнитного излучения пределами, безопасными для человека. В стандарте *TCO-92* эти нормы были сохранены, а в стандартах *TCO-95* и *TCO-99* — ужесточены. Эргономические и экологические нормы впервые появились в стандарте *TCO-95*, а стандарт *TCO-99* установил самые жесткие нормы по параметрам, определяющим качество изображения (яркость, контрастность, мерцание, антибликовые свойства покрытия).

Большинством параметров изображения, полученного на экране монитора, можно управлять программно. Программные средства, предназначенные для этой цели, обычно входят в системный комплект программного обеспечения — мы рассмотрим их при изучении операционной системы компьютера.

Клавиатура

Клавиатура — клавишное устройство управления персональным компьютером. Служит для ввода *алфавитно-цифровых* (*знаковых*) данных, а также команд управления. Комбинация монитора и клавиатуры обеспечивает простейший *интерфейс пользователя*. С помощью клавиатуры управляют компьютерной системой, а с помощью монитора получают от нее отклик.

Принцип действия. Клавиатура относится к стандартным средствам персонального компьютера. Ее основные функции не нуждаются в поддержке специальными системными программами (драйверами). Необходимое программное обеспечение для начала работы с компьютером уже имеется в микросхеме ПЗУ в составе базовой системы ввода-вывода (*BIOS*), и потому компьютер реагирует на нажатия клавиш сразу после включения.

Принцип действия клавиатуры заключается в следующем.

1. При нажатии на клавишу (или комбинацию клавиш) специальная микросхема, встроенная в клавиатуру, генерирует и выдает так называемый *скан-код*.
2. Скан-код поступает в микросхему, выполняющую функции *порта* клавиатуры. (Порты — специальные аппаратно-логические устройства, отвечающие за связь процессора с другими устройствами.) Порт клавиатуры — это довольно простое устройство, интегрированное в одну из микросхем материнской платы.
3. Порт клавиатуры выдает процессору прерывание с фиксированным номером. Для клавиатуры номер прерывания — 9 (*Interrupt 9, Int 9*).
4. Получив прерывание, процессор откладывает текущую работу и по номеру прерывания обращается в специальную область оперативной памяти, в которой находится так называемый *вектор прерываний*. Вектор прерываний — это список адресных данных с фиксированной длиной записи. Каждая запись содержит адрес программы, которая должна обслужить прерывание с номером, совпадающим с номером записи.
5. Определив адрес начала программы, обрабатывающей возникшее прерывание, процессор переходит к ее исполнению. Простейшая программа обработки клавиатурного прерывания «зашита» в микросхему ПЗУ, но программисты могут «подставить» вместо нее свою программу, если изменят данные в векторе прерываний.
6. Программа-обработчик прерывания направляет процессор к порту клавиатуры, где он находит скан-код, загружает его в свои регистры, потом под управлением обработчика определяет, какой код символа соответствует данному скан-коду.
7. Далее обработчик прерываний отправляет полученный код символа в небольшую область памяти, известную как *буфер клавиатуры*, и прекращает свою работу, известив об этом процессор.
8. Процессор прекращает обработку прерывания и возвращается к отложенной задаче.
9. Введенный символ хранится в буфере клавиатуры до тех пор, пока его не заберет оттуда та программа, для которой он предназначался, например текстовый

Рис. 3.2. Группы клавиш стандартной клавиатуры

редактор или текстовый процессор. Если символы поступают в буфер чаще, чем забираются оттуда, возможен эффект переполнения буфера. В этом случае ввод новых символов на некоторое время прекращается. На практике в этот момент при нажатии на клавишу мы слышим предупреждающий звуковой сигнал и не наблюдаем ввода данных.

Состав клавиатуры. Стандартная клавиатура имеет более 100 клавиш, функционально распределенных по нескольким группам (см. рис. 3.2).

Группа *алфавитно-цифровых клавиш* предназначена для ввода знаковой информации и команд, набираемых по буквам. Каждая клавиша может работать в нескольких режимах (*регистрах*) и, соответственно, может использоваться для ввода нескольких символов. Переключение между *нижним регистром* (для ввода строчных символов) и *верхним регистром* (для ввода прописных символов) выполняют удержанием клавиши SHIFT (нефиксированное переключение). При необходимости жестко переключить регистр используют клавишу CAPS LOCK (фиксированное переключение). Если клавиатура используется для ввода данных, абзац закрывают нажатием клавиши ENTER. При этом автоматически начинается ввод текста с новой строки. Если клавиатуру используют для ввода команд, клавишей ENTER завершают ввод команды и начинают ее исполнение.

Для разных языков существуют различные схемы закрепления символов национальных алфавитов за конкретными алфавитно-цифровыми клавишами. Такие схемы называются *раскладками клавиатуры*. Переключения между различными раскладками выполняются программным образом — это одна из функций операционной системы. Соответственно, способ переключения зависит от того, в какой операционной системе работает компьютер. Например, в системе *Windows XP* для этой цели могут использоваться следующие комбинации: левая клавиша ALT+SHIFT

или CTRL+SHIFT. При работе с другой операционной системой способ переключения можно установить по справочной системе той программы, которая выполняет переключение.

Общепринятые раскладки клавиатуры имеют свои корни в раскладках клавиатур пишущих машинок. Для персональных компьютеров *IBM PC* типовыми считаются раскладки QWERTY (английская) и ЙЦУКЕН (русская). Раскладки принято именовать по символам, закрепленным за первыми клавишами верхней строки алфавитной группы.

Группа функциональных клавиш включает двенадцать клавиш (от F1 до F12), размещенных в верхней части клавиатуры. Функции, закрепленные за данными клавишами, зависят от свойств конкретной работающей в данный момент программы, а в некоторых случаях и от свойств операционной системы. Общепринятым для большинства программ является соглашение о том, что клавиша F1 вызывает справочную систему, в которой можно найти справку о действии прочих клавиш.

Служебные клавиши располагаются рядом с клавишами алфавитно-цифровой группы. В связи с тем, что ими приходится пользоваться особенно часто, они имеют увеличенный размер. К ним относятся рассмотренные выше клавиши SHIFT и ENTER, регистровые клавиши ALT и CTRL (их используют в комбинации с другими клавишами для формирования команд), клавиша TAB (для ввода позиций табуляции при наборе текста), клавиша ESC (от английского слова *Escape*) для отказа от исполнения начатой операции и клавиша BACKSPACE для удаления только что введенных знаков (она находится над клавишей ENTER и часто маркируется стрелкой, направленной влево).

Служебные клавиши PRINT SCREEN, SCROLL LOCK и PAUSE/BREAK размещаются справа от группы функциональных клавиш и выполняют специфические функции, зависящие от действующей операционной системы. Общепринятыми являются следующие действия:

- PRINT SCREEN — печать текущего состояния экрана на принтере (для *MS-DOS*) или сохранение его в специальной области оперативной памяти, называемой *буфером обмена* (для *Windows*).
- SCROLL LOCK — переключение режима работы в некоторых (как правило, устаревших) программах.
- PAUSE/BREAK — приостановка/прерывание текущего процесса (для *MS-DOS*).

Две группы *клавиш управления курсором* расположены справа от алфавитно-цифровой панели. *Курсором* называется экранный элемент, указывающий место ввода знаковой информации. Курсор используется при работе с программами, выполняющими ввод данных и команд с клавиатуры. Клавиши управления курсором позволяют управлять позицией ввода.

Четыре клавиши со стрелками выполняют смещение курсора в направлении, указанном стрелкой (их обычно называют просто *курсорными клавишами*). Действие прочих клавиш описано ниже.

PAGE UP/PAGE DOWN — перевод курсора на одну страницу вверх или вниз. Понятие «страница» обычно относится к фрагменту документа, видимому на экране. В гра-

фических операционных системах (например, *Windows*) этими клавишами выполняют «прокрутку» содержимого в текущем окне. Действие этих клавиш во многих программах может быть модифицировано с помощью служебных регистровых клавиш, в первую очередь SHIFT и CTRL. Конкретный результат модификации зависит от конкретной программы и/или операционной системы.

Клавиши HOME и END переводят курсор в начало или конец текущей строки соответственно. Их действие также модифицируется регистровыми клавишами.

Традиционное назначение клавиши INSERT состоит в переключении режима ввода данных (переключение между режимами *вставки* и *замены*). Если текстовый курсор находится внутри существующего текста, то в режиме вставки происходит ввод новых знаков без замены существующих символов (текст как бы раздвигается). В режиме замены новые знаки заменяют текст, имевшийся ранее в позиции ввода.

В современных программах действие клавиши INSERT может быть иным. Конкретную информацию следует получить в справочной системе программы. Возможно, что действие этой клавиши является настраиваемым, — это также зависит от свойств конкретной программы.

Клавиша DELETE предназначена для удаления знаков, находящихся справа от текущего положения курсора. При этом положение позиции ввода остается неизменным.


Сравните действие клавиши DELETE с действием служебной клавиши BACKSPACE. Последняя служит для удаления знаков, но при ее использовании позиция ввода смещается влево, и, соответственно, удаляются символы, находящиеся не справа, а слева от курсора.

Группа клавиш дополнительной панели дублирует действие цифровых и некоторых знаковых клавиш основной панели. Во многих случаях для использования этой группы клавиш следует предварительно включать клавишу-переключатель NUM LOCK (о состоянии переключателей NUM LOCK, CAPS LOCK и SCROLL LOCK можно судить по светодиодным индикаторам, обычно расположенным в правом верхнем углу клавиатуры).

Появление дополнительной панели клавиатуры относится к началу 80-х годов. В то время клавиатуры были относительно дорогостоящими устройствами. Первоначальное назначение дополнительной панели состояло в снижении износа основной панели при проведении расчетно-кассовых вычислений, а также при управлении компьютерными играми (при выключенном переключателе NUM LOCK клавиши дополнительной панели могут использоваться в качестве клавиш управления курсором).

В наши дни клавиатуры относят к малоценным быстроизнашивающимся устройствам и приспособлениям, и существенной необходимости оберегать их от износа нет. Тем не менее за дополнительной клавиатурой сохраняется важная функция ввода символов, для которых известен расширенный код *ASCII* (см. выше), но неизвестно закрепление за клавишей клавиатуры. Так, например, известно, что символ «§» (параграф) имеет код 0167, а символ «°» (угловой градус) имеет код 0176, но соответствующих им клавиш на клавиатуре нет. В таких случаях для их ввода используют дополнительную панель.

Порядок ввода символов по известному *ALT*-коду.

1. Убедиться в том, что включен переключатель NUM LOCK.
2. Нажать и удержать клавишу ALT.
3. Не отпуская клавиши ALT, набрать последовательно на дополнительной панели *ALT*-код вводимого символа, например: 0 1 6 7.
4. Отпустить клавишу ALT. Символ, имеющий код 0167, появится на экране в позиции ввода.

Узнать ALT-коды некоторых символов позволяет программа Таблица символов (см. раздел 7.3).

Настройка клавиатуры. Клавиатуры персональных компьютеров обладают *свойством повтора знаков*, которое используется для автоматизации процесса ввода. Оно состоит в том, что при длительном удержании клавиши начинается автоматический ввод связанного с ней кода. При этом настраиваемыми параметрами являются:

- интервал времени после нажатия, по истечении которого начнется автоматический повтор кода;
- темп повтора (количество знаков в секунду).

Средства настройки клавиатуры относятся к системным и обычно входят в состав операционной системы. Кроме параметров режима повтора, настройке подлежат также используемые раскладки и органы управления, используемые для переключения раскладок. Со средствами настройки клавиатуры мы познакомимся при изучении функций операционной системы.

Мышь

Мышь — устройство управления манипуляторного типа. Представляет собой плоскую коробочку с двумя-тремя кнопками. Перемещение мыши по плоской поверхности синхронизировано с перемещением графического объекта (*указателя мыши*) на экране монитора.

Принцип действия. В отличие от рассмотренной ранее клавиатуры мышь не является стандартным органом управления, и персональный компьютер не имеет для нее выделенного порта. Для мыши нет и постоянного выделенного прерывания, а базовые средства ввода и вывода (*BIOS*) компьютера, размещенные в постоянном запоминающем устройстве (ПЗУ), не содержат программных средств для обработки прерываний мыши.

В связи с этим в первый момент после включения компьютера мышь не работает. Она нуждается в поддержке специальной системной программы — *драйвера мыши*. Драйвер устанавливается либо при первом подключении мыши, либо при установке операционной системы компьютера. Хотя мышь и не имеет выделенного порта на материнской плате, для работы с ней используют один из стандартных портов, средства для работы с которыми имеются в составе *BIOS*. Драйвер мыши предназначен для интерпретации сигналов, поступающих через порт. Кроме того, он обеспечивает механизм передачи информации о положении и состоянии мыши операционной системе и работающим программам.

Компьютером управляют перемещением мыши по плоскости и кратковременными нажатиями правой и левой кнопок. (Эти нажатия называются *щелчками*.) В отличие от клавиатуры мышь не может напрямую использоваться для ввода знаковой информации — ее принцип управления является *событийным*. Перемещения мыши и щелчки ее кнопок являются *событиями* с точки зрения ее программы-драйвера. Анализируя эти события, драйвер устанавливает, когда произошло событие и в каком месте экрана в этот момент находился указатель. Эти данные передаются в прикладную программу, с которой работает пользователь в данный момент. По ним программа может определить команду, которую имел в виду пользователь, и приступить к ее исполнению.

Комбинация монитора и мыши обеспечивает наиболее современный тип интерфейса пользователя, который называется *графическим*. Пользователь наблюдает на экране графические *объекты* и *элементы управления*. С помощью мыши он изменяет *свойства объектов* и приводит в действие *элементы управления* компьютерной системой, а с помощью монитора получает от нее отклик в графическом виде.

Стандартная мышь имеет только две кнопки, хотя существуют нестандартные мыши с тремя кнопками. Сегодня наиболее распространены мыши, в которых роль третьей кнопки играет вращающееся колесико-регулятор. Функции дополнительных органов управления определяются тем программным обеспечением, которое поставляется вместе с устройством.

К числу регулируемых параметров мыши относятся: *чувствительность* (выражает величину перемещения указателя на экране при заданном линейном перемещении мыши), функции левой и правой кнопок, а также *чувствительность к двойному нажатию* (максимальный интервал времени, при котором два щелчка кнопкой мыши расцениваются как один двойной щелчок). Программные средства, предназначенные для этих регулировок, обычно входят в системный комплект программного обеспечения — мы рассмотрим их при изучении операционной системы.

3.2. Внутренние устройства системного блока

Материнская плата

Материнская плата — основная плата персонального компьютера. На ней размещаются:

- *процессор* — основная микросхема, выполняющая большинство математических и логических операций;
- *микропроцессорный комплект* (*чипсет*) — набор микросхем, управляющих работой внутренних устройств компьютера и определяющих основные функциональные возможности материнской платы;
- *шины* — наборы проводников, по которым происходит обмен сигналами между внутренними устройствами компьютера;
- *оперативная память* (*оперативное запоминающее устройство, ОЗУ*) — набор микросхем, предназначенных для временного хранения данных, когда компьютер включен;

- *ПЗУ* (*постоянное запоминающее устройство*) — микросхема, предназначенная для длительного хранения данных, в том числе и когда компьютер выключен;
- разъемы для подключения дополнительных устройств (*слоты*).

Устройства, входящие в состав материнской платы, рассматриваются отдельно в разделе 3.3.

Жесткий диск

Жесткий диск — основное устройство для долговременного хранения больших объемов данных и программ. На самом деле это не один диск, а группа соосных дисков, имеющих магнитное покрытие и вращающихся с высокой скоростью. Таким образом, этот «диск» имеет не две поверхности, как должно быть у обычного плоского диска, а $2n$ поверхностей, где n — число отдельных дисков в группе.

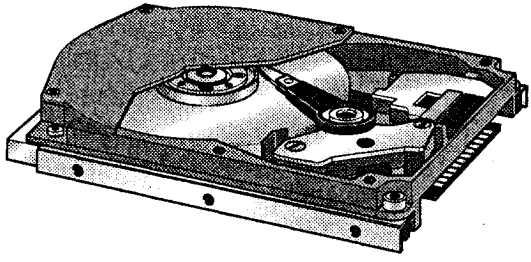


Рис. 3.3. Жесткий диск

Над каждой поверхностью располагается головка, предназначенная для чтения-записи данных. При высоких скоростях вращения дисков (90–250 об/с) в зазоре между головкой и поверхностью образуется аэродинамическая подушка, и головка парит над магнитной поверхностью на высоте, составляющей несколько тысячных долей миллиметра. При изменении силы тока, протекающего через головку, происходит изменение напряженности динамического магнитного поля в зазоре, что вызывает изменения в стационарном магнитном поле ферромагнитных частиц, образующих покрытие диска. Так осуществляется запись данных на магнитный диск.

Операция считывания происходит в обратном порядке. Намагниченные частицы покрытия, проносящиеся на высокой скорости вблизи головки, наводят в ней ЭДС самоиндукции. Электромагнитные сигналы, возникающие при этом, усиливаются и передаются на обработку.

Управление работой жесткого диска выполняет специальное аппаратно-логическое устройство — *контроллер жесткого диска*. В прошлом оно представляло собой отдельную *дочернюю плату*, которую подключали к одному из свободных слотов материнской платы. В настоящее время функции контроллеров дисков частично интегрированы в сам жесткий диск, а частично выполняются микросхемами, входящими в микропроцессорный комплект (чипсет), хотя некоторые виды высокопроизводительных контроллеров жестких дисков по-прежнему могут поставляться на отдельной плате.

К основным параметрам жестких дисков относятся *емкость* и *производительность*. Емкость дисков зависит от технологии их изготовления. В настоящее время большинство производителей жестких дисков используют изобретенную компанией *IBM* технологию с использованием *гигантского магниторезистивного эффекта*

(*GMR — Giant Magnetic Resistance*). В настоящее время на пластину может приходиться 40 и более Гбайт, но развитие продолжается.

С другой стороны, производительность жестких дисков меньше зависит от технологии их изготовления. Сегодня все жесткие диски имеют очень высокий показатель скорости внутренней передачи данных (до 30–60 Мбайт/с), и потому их производительность в первую очередь зависит от характеристик интерфейса, с помощью которого они связаны с материнской платой. В зависимости от типа интерфейса разброс значений может быть очень большим: от нескольких Мбайт/с до 13–16 Мбайт/с для интерфейсов типа *EIDE*; до 80 Мбайт/с для интерфейсов типа *SCSI* и от 50 Мбайт/с и более для наиболее современных интерфейсов типа *IEEE 1394* и *Serial ATA*.

Кроме скорости передачи данных с производительностью диска напрямую связан параметр *среднего времени доступа*. Он определяет интервал времени, необходимый для поиска нужных данных, и зависит от скорости вращения диска. Для дисков, вращающихся с частотой 5400 об/мин, среднее время доступа составляет 9–10 мкс, для дисков с частотой 7200 об/мин — 7–8 мкс. Изделия более высокого уровня обеспечивают среднее время доступа к данным 4–6 мкс.

Дисковод гибких дисков

Информация на жестком диске может храниться годами, однако иногда требуется ее перенос с одного компьютера на другой. Несмотря на свое название, жесткий диск является весьма хрупким прибором, чувствительным к перегрузкам, ударам и толчкам. Теоретически, переносить информацию с одного рабочего места на другое путем переноса жесткого диска возможно, и в некоторых случаях так и поступают, но все-таки этот прием считается нетехнологичным, поскольку требует особой аккуратности и определенной квалификации.

Для оперативного переноса небольших объемов информации используют так называемые *гибкие магнитные диски* (дискеты), которые вставляют в специальный накопитель — *дисковод*. Приемное отверстие накопителя находится на лицевой панели системного блока. Правильное направление подачи гибкого диска отмечено стрелкой на его пластиковом кожухе.

Основными параметрами гибких дисков являются: технологический размер (измеряется в дюймах), плотность записи (измеряется в кратных единицах) и полная емкость.

Первый компьютер *IBM PC* (родоначальник платформы) был выпущен в 1981 году. К нему можно было подключить внешний накопитель, использующий односторонние гибкие диски диаметром 5,25 дюйма. Емкость диска составляла 160 Кбайт. В следующем году появились аналогичные двусторонние диски емкостью 320 Кбайт. Начиная с 1984 года выпускались гибкие диски 5,25 дюйма высокой плотности (1,2 Мбайт). В наши дни диски размером 5,25 дюйма не используются, так что производство и применение соответствующих дисководов практически прекратилось с середины 90-х годов.

Гибкие диски размером 3,5 дюйма выпускают с 1980 года. Односторонний диск *обычной плотности* имел емкость 180 Кбайт, двусторонний — 360 Кбайт, а *двусто-*

ронний двойной плотности — 720 Кбайт. Ныне стандартными считают диски размером 3,5 дюйма *высокой плотности*. Они имеют емкость 1440 Кбайт (1,4 Мбайт) и маркируются буквами *HD* (*high density — высокая плотность*).

С нижней стороны гибкий диск имеет центральную втулку, которая захватывается шпинделем дисковода и приводится во вращение. Магнитная поверхность прикрыта сдвигающейся шторкой для защиты от влаги, грязи и пыли. Если на гибком диске записаны ценные данные, его можно защитить от стирания и перезаписи, сдвинув защитную задвижку так, чтобы образовалось открытое отверстие. Для разрешения записи задвижку перемещают в обратную сторону и перекрывают отверстие. В некоторых случаях для безусловной защиты информации на диске задвижку выламывают физически, но и в этом случае разрешить запись на диск можно, если, например, заклеить образовавшееся отверстие тонкой полоской липкой ленты.

Гибкие диски считаются малонадежными носителями информации. Пыль, грязь, влага, температурные перепады и внешние электромагнитные поля очень часто становятся причиной частичной или полной утраты данных, хранившихся на гибком диске. Поэтому использовать гибкие диски в качестве основного средства хранения информации недопустимо. Их используют только для транспортировки информации или в качестве дополнительного (резервного) средства хранения.

При передаче данных на гибком носителе следует придерживаться следующих правил этикета.

1. Все данные передаются в двух экземплярах.
2. Данные не удаляются с жесткого диска до тех пор, пока потребитель не подтвердил их благополучное получение, например по телефону.

При использовании гибких носителей в качестве резервного средства хранения данных следует придерживаться следующих рекомендаций.

1. Если эти данные неизменяемые, следует создать одну копию на гибком носителе, но не удалять данные с жесткого диска. Если данные с жесткого диска следует удалить, количество копий, закладываемых на хранение, должно быть не менее двух.
2. Если резервируемые данные подлежат периодическому изменению, то с жесткого диска их не удаляют, а количество резервных копий на гибких дисках должно быть не менее двух. Для этих копий устраивают периодическую ротацию с заданной периодичностью. Например, в конце первой рабочей недели копируют данные с жесткого диска на первый резервный комплект, а в конце второй недели — на второй резервный комплект, после чего еженедельно производят ротацию резервных комплектов.

При получении данных на гибком диске следует придерживаться следующих рекомендаций.

1. До начала работы с данными диск следует проверить антивирусными программными средствами. Среди вредоносных программ есть такие, которые поражают не только файлы программ и данных, но и носители информации. Даже «чистый» гибкий диск может содержать так называемые «загрузочные вирусы».

2. С данными, поставленными на гибком диске, работать не рекомендуется. Это не только непроизводительно, но и небезопасно (для данных). Прежде всего следует скопировать полученные данные на жесткий диск компьютера, после чего работать только с жестким диском.
3. Даже если работа с полученными данными в ближайшее время не предполагается, все равно их следует скопировать на жесткий диск немедленно после получения, так как во время хранения гибкого диска данные могут быть утрачены.
4. Правила делового этикета требуют немедленно после копирования данных с гибкого диска на жесткий оповестить лицо, предоставившее гибкий диск, о том, что прием данных состоялся. Это позволит ему сознательно распорядиться своими резервными копиями.

В новейших компьютерах происходит постепенный отказ и от этого типа носителей, которые вытесняются записывающими дисководами *CD-RW*.

Дисковод компакт-дисков CD-ROM

В период 1994–1995 годов в базовую конфигурацию персональных компьютеров перестали включать дисководы гибких дисков диаметром 5,25 дюйма, но вместо них стандартной стала считаться установка дисковода *CD-ROM*, имеющего такие же внешние размеры.

Аббревиатура *CD-ROM* (*Compact Disc Read-Only Memory*) переводится на русский язык как *постоянное запоминающее устройство на основе компакт-диска*. Принцип действия этого устройства состоит в считывании числовых данных с помощью лазерного луча, отражающегося от поверхности диска (рис. 3.4). Цифровая запись на компакт-диске отличается от записи на магнитных дисках очень высокой плотностью, и стандартный компакт-диск может хранить примерно 650 Мбайт данных.

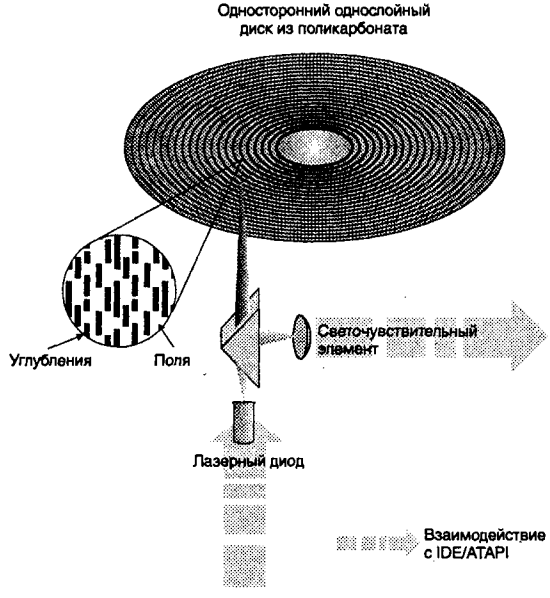


Рис. 3.4. Принцип действия дисковода CD-ROM

Большие объемы данных характерны для *мультимедийной информации* (графика, музыка, видео), поэтому дисководы *CD-ROM* относят к аппаратным средствам мультимедиа. Программные продукты, распространяемые на компакт-дисках, называют *мультимедийными изданиями*. Сегодня мультимедийные издания завоевывают все более прочное место среди других традиционных видов изданий. Так, например, существуют книги, альбомы, энциклопедии и даже периодические издания (электронные журналы), выпускаемые на *CD-ROM*.

Основным недостатком стандартных дисководов *CD-ROM* является невозможность записи данных, но параллельно с ними сегодня существуют и устройства записи компакт-дисков — дисководы *CD-RW*. Для записи используются специальные заготовки. Некоторые из них допускают только однократную запись (после записи диск превращается в обычный компакт-диск *CD-ROM*, доступный только для чтения), другие позволяют стереть ранее записанную информацию и выполнить запись заново.

Основным параметром дисководов *CD-ROM* является скорость чтения данных. Она измеряется в кратных долях. За единицу измерения принята скорость чтения музыкальных компакт-дисков, составляющая в пересчете на данные 150 Кбайт/с. Таким образом, дисковод с удвоенной скоростью чтения обеспечивает производительность 300 Кбайт/с, с учетверенной скоростью — 600 Кбайт/с и т. д. В настоящее время наибольшее распространение имеют устройства чтения *CD-ROM* с производительностью 48×–56×. Для заготовок, рассчитанных на однократную запись, скорость записи в соответствующих устройствах не уступает скорости чтения. Для заготовок многократной записи скорость записи может составлять 12×–24×.

Видеокарта (видеоадаптер)

Совместно с монитором *видеокарта* образует *видеоподсистему* персонального компьютера. Видеокарта не всегда была компонентом ПК. На заре развития персональной вычислительной техники в общей области оперативной памяти существовала небольшая выделенная *экранная область памяти*, в которую процессор заносил данные об изображении. Специальный *контроллер экрана* считывал данные о яркости отдельных точек экрана из ячеек памяти этой области и в соответствии с ними управлял разверткой горизонтального луча электронной пушки монитора.

С переходом от черно-белых мониторов к цветным и с увеличением *разрешения экрана* (количества точек по вертикали и горизонтали) области видеопамяти стало недостаточно для хранения графических данных, а процессор перестал справляться с построением и обновлением изображения. Тогда и произошло выделение всех операций, связанных с управлением экраном, в отдельный блок, получивший название *видеоадаптер*. Физически видеоадаптер выполнен в виде отдельной *дочерней платы*, которая вставляется в один из слотов материнской платы и называется *видеокартой*. Видеоадаптер взял на себя функции *видеоконтроллера*, *видеопроцессора* и *видеопамяти*.

За время существования персональных компьютеров сменилось несколько стандартов видеоадаптеров: *MDA* (*монохромный*); *CGA* (4 *цвета*); *EGA* (16 *цветов*); *VGA* (256 *цветов*). В настоящее время применяются видеоадаптеры *SVGA*, обеспечивающие по выбору воспроизведение до 16,7 миллионов цветов с возможностью произвольного выбора разрешения экрана из стандартного ряда значений (640×480, 800×600, 1024×768, 1152×864; 1280×1024 точек и далее).

Разрешение экрана является одним из важнейших параметров видеоподсистемы. Чем оно выше, тем больше информации можно отобразить на экране, но тем меньше размер каждой отдельной точки и, соответственно, тем меньше видимый размер элементов изображения. Использование завышенного разрешения на мониторе

малого размера приводит к тому, что элементы изображения становятся неразборчивыми и работа с документами и программами вызывает утомление органов зрения. Использование заниженного разрешения приводит к тому, что элементы изображения становятся крупными, но на экране их располагается очень мало. Если программа имеет сложную систему управления и большое число экранных элементов, они не полностью помещаются на экране. Это приводит к снижению производительности труда и неэффективной работе.

Таким образом, для каждого размера монитора существует свое оптимальное разрешение экрана, которое должен обеспечивать видеоадаптер (табл. 3.1). При качественном мониторе, хорошем зрении и ограниченном времени работы за компьютером разрешение можно увеличить на одну ступень.

Таблица 3.1. Разрешение экрана монитора

Размер монитора	Оптимальное разрешение экрана	Примечание
14 дюймов ЭЛТ	640×480	Не поддерживается в Windows XP
15 дюймов ЭЛТ	800×600	Минимальное разрешение для современных программ
15 дюймов ЖК или 17 дюймов ЭЛТ	1024×768	Типичное разрешение для современных программ
17 дюймов ЖКили 19 дюймов ЭЛТ	1280×1024	

Большинство современных прикладных и развлекательных программ рассчитано на работу с разрешением экрана 800×600 и более. Именно поэтому сегодня минимально приемлемый размер монитора составляет 15 дюймов. Для работы с документами, подготовленными для печати на стандартных листах бумаги формата А4, необходимо экранное разрешение не менее 1024×768 и, соответственно, размер монитора в 17 дюймов.

Для работы в Интернете параметр разрешения зависит от способа оформления *Web*-страниц. Современные *Web*-страницы рассчитаны на работу с разрешением экрана 1024×768, хотя многие приемлемо выглядят и при разрешении 800×600.

Для большинства прикладных программ оптимальным также является разрешение 1024×768 и более, хотя в случае необходимости программы, как правило, допускают настройку своих панелей управления, делающую возможной работу в разрешении 800×600. Надо понимать, что при этом снижается производительность труда.

Таким образом, в настоящее время для работы с документами и службами Интернета наиболее приемлем размер ЭЛТ-монитора в 17 дюймов. Почти такое же изображение обеспечивает ЖК-монитор размером в 15 дюймов. Размеры экранов более 17 дюймов и разрешения выше, чем 1024×768, применяют при работе с компьютерной графикой, системами автоматизированного проектирования и системами компьютерной верстки изданий.

Цветовое разрешение (*глубина цвета*) определяет количество различных оттенков, которые может принимать отдельная точка экрана. Максимально возможное цветовое разрешение зависит от свойств видеоадаптера и, в первую очередь, от количе-

ства установленной на нем видеопамяти. Кроме того, оно зависит и от установленного разрешения экрана. При высоком разрешении экрана на каждую точку изображения приходится отводить меньше места в видеопамяти, так что информация о цветах вынужденно оказывается более ограниченной.

В зависимости от заданного экранного разрешения и глубины цвета размер буфера кадра видеопамяти можно определить по следующей формуле:

$$P = \frac{(m \cdot n) \cdot b}{8},$$ где:

P — необходимый объем памяти видеоадаптера;

m — горизонтальное разрешение экрана (точек);

n — вертикальное разрешение экрана (точек);

b — разрядность кодирования цвета (бит).

Минимальное требование по глубине цвета на сегодняшний день — 256 цветов, хотя большинство программ требуют не менее 65 тыс. цветов (режим *High Color*). Наиболее комфортная работа достигается при глубине цвета 16,7 млн. цветов (режим *True Color*).

Работа в полноцветном режиме *True Color* с высоким экранным разрешением требует значительных размеров видеопамяти. Современные видеоадаптеры способны также выполнять функции обработки изображения, снижая нагрузку на центральный процессор ценой дополнительных затрат видеопамяти. Объем видеопамяти, установленной на видеоадаптер, сегодня определяется не размером буфера кадра, а необходимостью выполнения подобных дополнительных операций, и обычно составляет 32–128 Мбайт.

Видеоускорение — одно из свойств видеоадаптера, которое заключается в том, что часть операций по построению изображений может происходить без выполнения математических вычислений в основном процессоре компьютера, а чисто аппаратным путем — преобразованием данных в микросхемах *видеоускорителя*. Видеоускорители обычно входят в состав видеоадаптера (в таких случаях говорят о том, что видеокарта обладает функциями аппаратного ускорения). Несколько лет назад существовали и видеоускорители, которые поставлялись в виде отдельной платы, устанавливаемой на материнской плате и подключаемой к видеоадаптеру.

Различают два типа видеоускорителей — ускорители плоской (*2D*) и трехмерной (*3D*) графики. Первые наиболее эффективны для работы с прикладными программами, использующими стандартный интерфейс (обычно офисного применения), и оптимизированы для операционной системы *Windows*, а вторые ориентированы на работу мультимедийных развлекательных программ, в первую очередь компьютерных игр, и профессиональных программ обработки трехмерной графики. Обычно в этих случаях используют разные математические принципы автоматизации графических операций. Все современные видеокарты обладают функциями и двумерного, и трехмерного ускорения.

Звуковая карта

Звуковая карта явилась одним из наиболее поздних усовершенствований персонального компьютера. Она устанавливается в один из разъемов материнской платы в виде дочерней карты и выполняет вычислительные операции, связанные с обработкой звука, речи, музыки. Звук воспроизводится через внешние звуковые колонки, подключаемые к выходу звуковой карты. Специальный разъем позволяет отправить звуковой сигнал на внешний усилитель. Имеется также разъем для подключения микрофона, что позволяет записывать речь или музыку и сохранять их на жестком диске для последующей обработки и использования.

Основным параметром звуковой карты является *разрядность*, определяющая количество битов, используемых при преобразовании сигналов из аналоговой в цифровую форму и наоборот. Чем выше разрядность, тем меньше погрешность, связанная с оцифровкой, тем выше качество звучания. Минимальным требованием сегодняшнего дня являются 16 разрядов, а наибольшее распространение имеют 32-разрядные и 64-разрядные устройства.

В области воспроизведения звука наиболее сложно обстоит дело со стандартизацией. В отсутствие единых централизованных стандартов, стандартом де-факто стали устройства, совместимые с устройством *Sound Blaster*, торговая марка на которое принадлежит компании *Creative Labs*.

В последнее время обработка звука рассматривается как относительно простая операция, которую, в связи с возросшей мощностью процессора, можно возложить и на него. В отсутствие повышенных требований к качеству звука можно использовать *интегрированные звуковые системы*, в которых функции обработки звука выполняются центральным процессором и микросхемами материнской платы. В этом случае колонки или иное устройство воспроизведения звука подключается к гнездам, установленным непосредственно на материнской плате.

3.3. Системы, расположенные на материнской плате

Оперативная память

Оперативная память (*RAM — Random Access Memory*) — это массив кристаллических ячеек, способных хранить данные. Существует много различных типов оперативной памяти, но с точки зрения физического принципа действия различают *динамическую память* (*DRAM*) и *статическую память* (*SRAM*).

Ячейки динамической памяти (*DRAM*) можно представить в виде микроконденсаторов, способных накапливать заряд на своих обкладках. Это наиболее распространенный и экономически доступный тип памяти. Недостатки этого типа связаны, во-первых, с тем, что как при заряде, так и при разряде конденсаторов неизбежны переходные процессы, то есть запись данных происходит сравнительно медленно. Второй важный недостаток связан с тем, что заряды ячеек имеют свойство рассеиваться в пространстве, причем весьма быстро. Если оперативную память постоянно не «подзаряжать», утрата данных происходит через несколько сотых долей секунды. Для борьбы с этим явлением в компьютере происходит постоянная *регенерация*

(*освежение*, *подзарядка*) ячеек оперативной памяти. Регенерация осуществляется несколько десятков раз в секунду и вызывает непроизводительный расход ресурсов вычислительной системы.

Ячейки статической памяти (*SRAM*) можно представить как электронные микроэлементы — *триггеры*, состоящие из нескольких транзисторов. В триггере хранится не заряд, а состояние (*включен/выключен*), поэтому этот тип памяти обеспечивает более высокое быстродействие, хотя технологически он сложнее и, соответственно, дороже.

Микросхемы динамической памяти используют в качестве основной оперативной памяти компьютера. Микросхемы статической памяти используют в качестве вспомогательной памяти (так называемой *кэш-памяти*), предназначенной для оптимизации работы процессора.

Каждая ячейка памяти имеет свой адрес, который выражается числом. В большинстве современных процессоров предельный размер адреса обычно составляет 32 разряда, а это означает, что всего независимых адресов может быть 2^{32}. Одна адресуемая ячейка содержит восемь двоичных ячеек, в которых можно сохранить 8 бит, то есть один байт данных.

Таким образом, в современных компьютерах возможна *непосредственная адресация* к полю памяти размером 2^{32} байт = 4 Гбайт. Однако это отнюдь не означает, что именно столько оперативной памяти непременно должно быть в компьютере. Предельный размер поля оперативной памяти, установленной в компьютере, определяется микропроцессорным комплектом (*чипсетом*) материнской платы и обычно не может превосходить нескольких Гбайт. Минимальный объем памяти определяется требованиями операционной системы и для современных компьютеров составляет 128 Мбайт.

Представление о том, сколько оперативной памяти *должно быть* в типовом компьютере, непрерывно меняется. В середине 80-х годов поле памяти размером 1 Мбайт казалось огромным, в начале 90-х годов достаточным считался объем 4 Мбайт, к середине 90-х годов он увеличился до 8 Мбайт, а затем и до 16 Мбайт. Сегодня типичным считается размер оперативной памяти в 256 Мбайт, но тенденция к росту сохраняется.

Оперативная память в компьютере размещается на стандартных панельках, называемых *модулями*. Модули оперативной памяти вставляют в соответствующие разъемы на материнской плате. Если к разъемам есть удобный доступ, то операцию можно выполнять своими руками. Если удобного доступа нет, может потребоваться неполная разборка узлов системного блока, и в таких случаях операцию поручают специалистам.

В современных компьютерах обычно применяют три типа модулей оперативной памяти. Модули памяти *SDRAM* (*DIMM*-модули) сегодня уже считаются устаревшими и используются в компьютерах прошлых поколений. Наиболее распространены модули типа *DDR SDRAM* (*DDR DIMM*), обеспечивающие более быстрый доступ к памяти. Модули типа *RDRAM* (*RIMM*-модули) применяются на некоторых компьютерах с процессором *Pentium* 4, но стоят заметно дороже и поэтому менее распространены.

Основными характеристиками модулей оперативной памяти являются объем памяти и скорость передачи данных. Сегодня наиболее распространены модули объемом 128–512 Мбайт. Скорость передачи данных определяет максимальную пропускную способность памяти (в Мбайт/с или Гбайт/с) в оптимальном режиме доступа. При этом учитывается время доступа к памяти, ширина шины и дополнительные возможности, такие как передача нескольких сигналов за один такт работы. Одинаковые по объему модули могут иметь разные скоростные характеристики.

Иногда в качестве определяющей характеристики памяти используют *время доступа*. Оно измеряется в миллиардных долях секунды (*наносекундах, нс*). Для современных модулей памяти это значение может составлять 5 нс, а для особо быстрой памяти, используемой в основном в видеокартах, — снижаться до 2–3 нс.

Процессор

Процессор — основная микросхема компьютера, в которой и производятся все вычисления. Конструктивно процессор состоит из ячеек, похожих на ячейки оперативной памяти, но в этих ячейках данные могут не только храниться, но и изменяться. Внутренние ячейки процессора называют *регистрами*. Важно также отметить, что данные, попавшие в некоторые регистры, рассматриваются не как данные, а как команды, управляющие обработкой данных в других регистрах. Среди регистров процессора есть и такие, которые в зависимости от своего содержания способны модифицировать исполнение команд. Таким образом, управляя засылкой данных в разные регистры процессора, можно управлять обработкой данных. На этом и основано исполнение программ.

С остальными устройствами компьютера, и в первую очередь с оперативной памятью, процессор связан несколькими группами проводников, называемых *шинами*. Основных шин три: *шина данных, адресная шина* и *командная шина*.

Адресная шина. У процессоров семейства *Pentium* (а именно они наиболее распространены в персональных компьютерах) адресная шина 32-разрядная, то есть состоит из 32 параллельных проводников. В зависимости от того, есть напряжение на какой-то из линий или нет, говорят, что на этой линии выставлена единица или ноль. Комбинация из 32 нулей и единиц образует 32-разрядный адрес, указывающий на одну из ячеек оперативной памяти. К ней и подключается процессор для копирования данных из ячейки в один из своих регистров.

Шина данных. По этой шине происходит копирование данных из оперативной памяти в регистры процессора и обратно. В современных персональных компьютерах шина данных, как правило, 64-разрядная, то есть состоит из 64 линий, по которым за один раз на обработку поступают сразу 8 байтов.

Шина команд. Для того чтобы процессор мог обрабатывать данные, ему нужны команды. Он должен знать, что следует сделать с теми байтами, которые хранятся в его регистрах. Эти команды поступают в процессор тоже из оперативной памяти, но не из тех областей, где хранятся массивы данных, а оттуда, где хранятся программы. Команды тоже представлены в виде байтов. Самые простые команды укладываются в один байт, однако есть и такие, для которых нужно два, три и более

байтов. В большинстве современных процессоров шина команд 32-разрядная, хотя существуют 64-разрядные процессоры и даже 128-разрядные.

Система команд процессора. В процессе работы процессор обслуживает данные, находящиеся в его регистрах, в поле оперативной памяти, а также данные, находящиеся во внешних портах процессора. Часть данных он интерпретирует непосредственно как данные, часть данных — как адресные данные, а часть — как команды. Совокупность всех возможных команд, которые может выполнить процессор над данными, образует так называемую *систему команд процессора*. Процессоры, относящиеся к одному семейству, имеют одинаковые или близкие системы команд. Процессоры, относящиеся к разным семействам, различаются по системе команд и невзаимозаменяемы.

Процессоры с расширенной и сокращенной системой команд. Чем шире набор системных команд процессора, тем сложнее его архитектура, тем длиннее формальная запись команды (в байтах), тем выше средняя продолжительность исполнения одной команды, измеренная в тактах работы процессора. Так, например, система команд процессоров семейства *Pentium* в настоящее время насчитывает более тысячи различных команд. Такие процессоры называют *процессорами с расширенной системой команд — CISC-процессорами (CISC — Complex Instruction Set Computing)*.

В противоположность *CISC*-процессорам в середине 80-х годов появились процессоры архитектуры *RISC* с сокращенной системой команд (*RISC — Reduced Instruction Set Computing*). При такой архитектуре количество команд в системе намного меньше и каждая из них выполняется намного быстрее. Таким образом, программы, состоящие из простейших команд, выполняются этими процессорами много быстрее. Оборотная сторона сокращенного набора команд состоит в том, что сложные операции приходится эмулировать далеко не эффективной последовательностью простейших команд сокращенного набора.

В результате конкуренции между двумя подходами к архитектуре процессоров сложилось следующее распределение их сфер применения:

- *CISC*-процессоры используют в универсальных вычислительных системах;
- *RISC*-процессоры используют в специализированных вычислительных системах или устройствах, ориентированных на выполнение единообразных операций.

Персональные компьютеры платформы *IBM PC* ориентированы на использование *CISC*-процессоров.

Совместимость процессоров. Если два процессора имеют одинаковую систему команд, то они полностью совместимы на программном уровне. Это означает, что программа, написанная для одного процессора, может исполняться и другим процессором. Процессоры, имеющие разные системы команд, как правило, несовместимы или ограниченно совместимы на программном уровне.

Группы процессоров, имеющих ограниченную совместимость, рассматривают как *семейства процессоров*. Так, например, все процессоры *Intel Pentium* относятся к так называемому семейству *x86*. Родоначальником этого семейства был 16-разрядный процессор *Intel 8086*, на базе которого собиралась первая модель компьютера

IBM PC. Впоследствии выпускались процессоры *Intel* 80286, *Intel* 80386, *Intel 80486*, несколько моделей *Intel Pentium*; несколько моделей *Intel Pentium MMX*, модели *Intel Pentium Pro*, *Intel Pentium II*, *Intel Celeron*, *Intel Xeon*, *Intel Pentium III*, *Intel Pentium* 4 и другие. Все эти модели, и не только они, а также многие модели процессоров компании *AMD* и некоторых других производителей относятся к семейству *x86* и обладают совместимостью по принципу «сверху вниз».

Принцип совместимости «сверху вниз» — это пример неполной совместимости, когда каждый новый процессор «понимает» все команды своих предшественников, но не наоборот. Это естественно, поскольку двадцать лет назад разработчики процессоров не могли предусмотреть систему команд, нужную для современных программ. Благодаря такой совместимости на современном компьютере можно выполнять любые программы, созданные в последние десятилетия для любого из предшествующих компьютеров, принадлежащего той же аппаратной платформе.

Основные параметры процессоров. Основными параметрами процессоров являются: *рабочее напряжение*, *разрядность*, *рабочая тактовая частота*, *коэффициент внутреннего умножения тактовой частоты* и размер *кэш-памяти*.

Рабочее напряжение процессора обеспечивает материнская плата, поэтому разным маркам процессоров соответствуют разные материнские платы (их надо выбирать совместно). По мере развития процессорной техники происходит постепенное понижение рабочего напряжения. Ранние модели процессоров *x86* имели рабочее напряжение 5 В. С переходом к процессорам *Intel Pentium* оно было понижено до 3,3 В, а в настоящее время оно составляет менее 2 В. Понижение рабочего напряжения позволяет уменьшить расстояния между структурными элементами в кристалле процессора до десятитысячных долей миллиметра, не опасаясь электрического пробоя. Пропорционально квадрату напряжения уменьшается и тепловыделение в процессоре, а это позволяет увеличивать его производительность без угрозы перегрева.

Разрядность процессора показывает, сколько бит данных он может принять и обработать в своих регистрах за один раз (*за один такт*). Первые процессоры *x86* были 16-разрядными. Начиная с процессора 80386 они имеют 32-разрядную архитектуру. Современные процессоры семейства *Intel Pentium* остаются 32-разрядными, хотя и работают с 64-разрядной шиной данных (разрядность процессора определяется не разрядностью шины данных, а разрядностью командной шины). В ближайшем будущем предполагается проникновение 64-разрядных процессоров на персональные компьютеры..

В основе работы процессора лежит тот же тактовый принцип, что и в обычных часах. Исполнение каждой команды занимает определенное количество тактов. В настенных часах такты колебаний задает маятник; в ручных механических часах их задает пружинный маятник; в электронных часах для этого есть колебательный контур, задающий такты строго определенной частоты. В персональном компьютере тактовые импульсы задает одна из микросхем, входящая в микропроцессорный комплект (чипсет), расположенный на материнской плате. Чем выше частота тактов, поступающих на процессор, тем больше команд он может исполнить в единицу времени, тем выше его производительность. Первые процессоры *x86* могли

работать с частотой не выше 4,77 МГц, а сегодня *рабочие частоты* некоторых процессоров уже превосходят 3 миллиарда тактов в секунду (3 ГГц).

Тактовые сигналы процессор получает от материнской платы, которая, в отличие от процессора, представляет собой не кристалл кремния, а большой набор проводников и микросхем. По чисто физическим причинам материнская плата не может работать со столь высокими частотами, как процессор. Сегодня базовая частота материнской платы составляет 100–200 МГц. Для получения более высоких частот в процессоре происходит *внутреннее умножение частоты*. Коэффициент внутреннего умножения в современных процессорах может достигать 10–20 и выше.

Обмен данными внутри процессора происходит в несколько раз быстрее, чем обмен с другими устройствами, например с оперативной памятью. Для того чтобы уменьшить количество обращений к оперативной памяти, внутри процессора создают буферную область — так называемую *кэш-память*. Это как бы «сверхоперативная память». Когда процессору нужны данные, он сначала обращается в кэш-память, и только если там нужных данных нет, происходит его обращение в оперативную память. Принимая блок данных из оперативной памяти, процессор заносит его одновременно и в кэш-память. «Удачные» обращения в кэш-память называют *попаданиями в кэш*. Процент попаданий тем выше, чем больше размер кэш-памяти, поэтому высокопроизводительные процессоры комплектуют повышенным объемом кэш-памяти.

Нередко кэш-память распределяют по нескольким уровням. Кэш первого уровня выполняется в том же кристалле, что и сам процессор, и имеет объем порядка десятков Кбайт. Кэш второго уровня находится либо в кристалле процессора, либо в том же узле, что и процессор, хотя и исполняется на отдельном кристалле. Кэш-память первого и второго уровня работает на частоте, согласованной с частотой ядра процессора.

Кэш-память третьего уровня выполняют на быстродействующих микросхемах типа *SRAM* и размещают на материнской плате вблизи процессора. Ее объемы могут достигать нескольких Мбайт, но работает она на частоте материнской платы.

Микросхема ПЗУ и система BIOS

В момент включения компьютера в его оперативной памяти нет ничего — ни данных, ни программ, поскольку оперативная память не может ничего хранить без подзарядки ячеек более сотых долей секунды, но процессору нужны команды, в том числе и в первый момент после включения. Поэтому сразу после включения на адресной шине процессора выставляется стартовый адрес. Это происходит аппаратно, без участия программ (всегда одинаково). Процессор обращается по выставленному адресу за своей первой командой и далее начинает работать по программам.

Этот исходный адрес не может указывать на оперативную память, в которой пока ничего нет. Он указывает на другой тип памяти — *постоянное запоминающее устройство (ПЗУ)*. Микросхема ПЗУ способна длительное время хранить информацию, даже когда компьютер выключен. Программы, находящиеся в ПЗУ, называют «зашитыми» — их записывают туда на этапе изготовления микросхемы.

Комплект программ, находящихся в ПЗУ, образует *базовую систему ввода-вывода* (*BIOS — Basic Input Output System*). Основное назначение программ этого пакета состоит в том, чтобы проверить состав и работоспособность компьютерной системы и обеспечить взаимодействие с клавиатурой, монитором, жестким диском и дисководом гибких дисков. Программы, входящие в *BIOS*, позволяют нам наблюдать на экране диагностические сообщения, сопровождающие запуск компьютера, а также вмешиваться в ход запуска с помощью клавиатуры.

Энергонезависимая память CMOS

Выше мы отметили, что работа таких стандартных устройств, как клавиатура, может обслуживаться программами, входящими в *BIOS*, но такими средствами нельзя обеспечить работу со всеми возможными устройствами. Так, например, изготовители *BIOS* абсолютно ничего не знают о параметрах наших жестких и гибких дисков, им не известны ни состав, ни свойства произвольной вычислительной системы. Для того чтобы начать работу с другим оборудованием, программы, входящие в состав *BIOS*, должны знать, где можно найти нужные параметры. По очевидным причинам их нельзя хранить ни в оперативной памяти, ни в постоянном запоминающем устройстве.

Специально для этого на материнской плате есть микросхема «энергонезависимой памяти», по технологии изготовления называемая *CMOS*. От оперативной памяти она отличается тем, что ее содержимое не стирается во время выключения компьютера, а от ПЗУ она отличается тем, что данные в нее можно заносить и изменять самостоятельно, в соответствии с тем, какое оборудование входит в состав системы. Эта микросхема постоянно подпитывается от небольшой аккумуляторной батарейки, расположенной на материнской плате. Заряда этой батарейки хватает на то, чтобы микросхема не теряла данные, даже если компьютер не будут включать месяцами.

В микросхеме *CMOS* хранятся данные о гибких и жестких дисках, о процессоре, о некоторых других устройствах материнской платы. Тот факт, что компьютер четко отслеживает время и календарь (даже и в выключенном состоянии), тоже связан с тем, что показания системных часов постоянно хранятся (и изменяются) в *CMOS*.

Таким образом, программы, записанные в *BIOS*, считывают данные о составе оборудования компьютера из микросхемы *CMOS*, после чего они могут выполнить обращение к жесткому диску, а в случае необходимости и к гибкому, и передать управление тем программам, которые там записаны.

Шинные интерфейсы материнской платы

Связь между всеми собственными и подключаемыми устройствами материнской платы выполняют ее шины и логические устройства, размещенные в микросхемах микропроцессорного комплекта (чипсета). От архитектуры этих элементов во многом зависит производительность компьютера.

ISA. Историческим достижением компьютеров платформы *IBM PC* стало внедрение почти двадцать лет назад архитектуры, получившей статус *промышленного стандарта ISA* (*Industry Standard Architecture*). Она не только позволила связать все устройства системного блока между собой, но и обеспечила простое подключе-

ние новых устройств через стандартные разъемы (слоты). Пропускная способность шины, выполненной по такой архитектуре, составляет до 5,5 Мбайт/с, но, несмотря на низкую пропускную способность, эта шина еще может использоваться в некоторых компьютерах для подключения сравнительно «медленных» внешних устройств, например звуковых карт и модемов.

EISA. Расширением стандарта *ISA* стал стандарт *EISA* (*Extended ISA*), отличающийся увеличенным разъемом и увеличенной производительностью (до 32 Мбайт/с). Как и *ISA*, в настоящее время данный стандарт считается устаревшим. После 2000 года выпуск материнских плат с разъемами *ISA/EISA* и устройств, подключаемых к ним, практически прекращен.

VLB. Название интерфейса переводится как *локальная шина стандарта VESA* (*VESA Local Bus*). Понятие «локальной шины» впервые появилось в конце 80-х годов. Оно связано тем, что при внедрении процессоров третьего и четвертого поколений (*Intel 80386* и *Intel 80486*) частоты основной шины (в качестве основной использовалась шина *ISA/EISA*) стало недостаточно для обмена между процессором и оперативной памятью. Локальная шина, имеющая повышенную частоту, связала между собой процессор и память в обход основной шины. Впоследствии в эту шину «врезали» интерфейс для подключения видеоадаптера, который тоже требует повышенной пропускной способности, — так появился стандарт *VLB*, который позволил поднять тактовую частоту локальной шины до 50 МГц и обеспечил пиковую пропускную способность до 130 Мбайт/с.

Основным недостатком интерфейса *VLB* стало то, что предельная частота локальной шины и, соответственно, ее пропускная способность зависят от числа устройств, подключенных к шине. Так, например, при частоте 50 МГц к шине может быть подключено только одно устройство (видеокарта). Для сравнения скажем, что при частоте 40 МГц возможно подключение двух, а при частоте 33 МГц — трех устройств. Активное использование шины *VLB* продолжалось очень недолго, она была вскоре вытеснена шиной *PCI*.

PCI. Интерфейс *PCI* (*Peripheral Component Interconnect — стандарт подключения внешних компонентов*) был введен в персональных компьютерах во времена процессора 80486 и первых версий *Pentium*. По своей сути это тоже интерфейс локальной шины, связывающей процессор с оперативной памятью, в которую врезаны разъемы для подключения внешних устройств. Для связи с основной шиной компьютера (*ISA/EISA*) используются специальные интерфейсные преобразователи — *мосты PCI* (*PCI Bridge*). В современных компьютерах функции моста *PCI* выполняют микросхемы микропроцессорного комплекта (чипсета).

Данный интерфейс поддерживает частоту шины 33 МГц и обеспечивает пропускную способность 132 Мбайт/с. Последние версии интерфейса поддерживают частоту до 66 МГц и обеспечивают производительность 264 Мбайт/с для 32-разрядных данных и 528 Мбайт/с для 64-разрядных данных.

Важным нововведением, реализованным этим стандартом, стала поддержка так называемого режима *plug-and-play*, впоследствии оформившегося в промышленный стандарт на *самоустанавливающиеся устройства*. Его суть состоит в том, что

после физического подключения внешнего устройства к разъему шины *PCI* происходит обмен данными между устройством и материнской платой, в результате которого устройство автоматически получает номер используемого прерывания, адрес порта подключения и номер канала прямого доступа к памяти.

Конфликты между устройствами за обладание одними и теми же ресурсами (номерами прерываний, адресами портов и каналами прямого доступа к памяти) вызывают массу проблем у пользователей при установке устройств, подключаемых к шине *ISA*. С появлением интерфейса *PCI* и с оформлением стандарта *plug-and-play* появилась возможность выполнять установку новых устройств с помощью автоматических программных средств — эти функции во многом были возложены на операционную систему.

FSB. Шина *PCI*, появившаяся в компьютерах на базе процессоров *Intel Pentium* как локальная шина, предназначенная для связи процессора с оперативной памятью, недолго оставалась в этом качестве. Сегодня она используется только как шина для подключения внешних устройств, а для связи процессора и памяти, начиная с процессора *Intel Pentium Pro*, используется специальная шина, получившая название *Front Side Bus* (*FSB*). Эта шина работает на частоте 100–200 МГц. Частота шины *FSB* является одним из основных потребительских параметров — именно он и указывается в спецификации материнской платы. Современные типы памяти (*DDR SDRAM*, *RDRAM*) способны передавать несколько сигналов за один такт шины *FSB*, что повышает скорость обмена данными с оперативной памятью.

AGP. Видеоадаптер — устройство, требующее особенно высокой скорости передачи данных. Как при внедрении локальной шины *VLB*, так и при внедрении локальной шины *PCI* видеоадаптер всегда был первым устройством, «врезаемым» в новую шину. Когда параметры шины *PCI* перестали соответствовать требованиям видеоадаптеров, для них была разработана отдельная шина, получившая название *AGP* (*Advanced Graphic Port — усовершенствованный графический порт*). Частота этой шины соответствует частоте шины *PCI* (33 МГц или 66 МГц), но она имеет много более высокую пропускную способность за счет передачи нескольких сигналов за один такт. Число сигналов, передаваемых за один такт, указывается в виде множителя, например *AGP* 4× (в этом режиме скорость передачи достигает 1066 Мбайт/с). Последняя версия шины *AGP* имеет кратность 8×.

PCMCIA (*Personal Computer Memory Card International Association* — стандарт международной ассоциации производителей плат памяти для персональных компьютеров). Этот стандарт определяет интерфейс подключения плоских карт памяти небольших размеров и используется в портативных персональных компьютерах.

USB (*Universal Serial Bus — универсальная последовательная магистраль*). Это одно из последних нововведений в архитектурах материнских плат. Этот стандарт определяет способ взаимодействия компьютера с периферийным оборудованием. Он позволяет подключать до 256 различных устройств, имеющих последовательный интерфейс. Устройства могут включаться цепочками (каждое следующее устройство подключается к предыдущему). Производительность шины *USB* относительно невелика, но вполне достаточна для таких устройств, как клавиатура, мышь, модем,

джойстик, принтер и т. п. Удобство шины состоит в том, что она практически исключает конфликты между различным оборудованием, позволяет подключать и отключать устройства в «горячем режиме» (не выключая компьютер) и позволяет объединять несколько компьютеров в простейшую локальную сеть без применения специального оборудования и программного обеспечения.

Функции микропроцессорного комплекта (чипсета)

Параметры микропроцессорного комплекта (чипсета) в наибольшей степени определяют свойства и функции материнской платы. В настоящее время большинство чипсетов материнских плат выпускаются на базе двух микросхем, исторически получивших название «северный мост» и «южный мост».

«Северный мост» обычно управляет взаимосвязью процессора, оперативной памяти и порта *AGP*.

«Южный мост» называют также *функциональным контроллером*. Он выполняет функции контроллера жестких и гибких дисков, функции контроллера шины *PCI*, моста *ISA — PCI*, контроллера клавиатуры, мыши, шины *USB* и т. п.

У предыдущих поколений материнских плат связь между северным и южном мостом обеспечивала шина *PCI*, контроллер которой располагался в северном мосте. У современных материнских плат мосты соединены новой шиной повышенной производительности, а контроллер шины *PCI* находится в южном мосте вместе с контроллерами всех прочих устройств.

3.4. Периферийные устройства персонального компьютера

Периферийные устройства персонального компьютера подключаются к его интерфейсам и предназначены для выполнения вспомогательных операций. Благодаря им компьютерная система приобретает гибкость и универсальность.

По назначению периферийные устройства можно подразделить на:

- устройства ввода данных;
- устройства вывода данных;
- устройства хранения данных;
- устройства обмена данными.

Устройства ввода знаковых данных

Специальные клавиатуры. Клавиатура является основным устройством ввода данных. Специальные клавиатуры предназначены для повышения эффективности процесса ввода данных. Это достигается путем изменения формы клавиатуры, раскладки ее клавиш или метода подключения к системному блоку.

Клавиатуры, имеющие специальную форму, рассчитанную с учетом требований эргономики, называют *эргономичными клавиатурами*. Их целесообразно применять на рабочих местах, предназначенных для ввода большого количества знаковой информации. Эргономичные клавиатуры не только повышают производительность наборщика и снижают общее утомление в течение рабочего дня, но и снижают веро-

ятность и степень развития ряда заболеваний, например туннельного синдрома кистей рук и остеохондроза верхних отделов позвоночника.

Раскладка клавиш стандартных клавиатур далека от оптимальной. Она сохранилась со времен ранних образцов механических пишущих машин. В настоящее время существует техническая возможность изготовления клавиатур с оптимизированной раскладкой и существуют образцы таких устройств (в частности, к ним относится *клавиатура Дворака*). Однако практическое внедрение клавиатур с нестандартной раскладкой находится под вопросом в связи с тем, что работе с ними надо учиться специально. На практике подобными клавиатурами оснащают только специализированные рабочие места.

По методу подключения к системному блоку различают *проводные* и *беспроводные* клавиатуры. Передача информации в беспроводных системах осуществляется инфракрасным лучом. Обычный радиус действия таких клавиатур составляет несколько метров. Источником сигнала является клавиатура.

Устройства командного управления

Специальные манипуляторы. Кроме обычной мыши существуют и другие типы манипуляторов, например: трекболы, пенмаусы, инфракрасные мыши.

Трекбол в отличие от мыши устанавливается стационарно, и его шарик приводится в движение ладонью руки. Преимущество трекбола состоит в том, что он не нуждается в гладкой рабочей поверхности, поэтому трекболы нашли широкое применение в портативных персональных компьютерах.

В последнее время, однако, в портативных компьютерах вместо трекболов используются *тачпады* — сенсорные пластины, реагирующие на движение пальца пользователя по поверхности. Удар пальцем по поверхности тачпада воспринимается как нажатие кнопки. Недостатком тачпадов является невысокая точность.

Пенмаус представляет собой аналог шариковой авторучки, на конце которой вместо пишущего узла установлен узел, регистрирующий величину перемещения.

Инфракрасная мышь отличается от обычной наличием устройства беспроводной связи с системным блоком.

Для компьютерных игр и в некоторых специализированных имитаторах применяют также манипуляторы рычажно-нажимного типа (*джойстики*) и аналогичные им *джойпады*, *геймпады* и *штурвально-педальные* устройства. Устройства этого типа подключаются к специальному порту, имеющемуся на звуковой карте, или к порту *USB*.

Устройства ввода графических данных

Для ввода графической информации используют *сканеры*, *графические планшеты* (*дигитайзеры*) и *цифровые фотокамеры*. Интересно отметить, что с помощью сканеров можно вводить и знаковую информацию. В этом случае исходный материал вводится в графическом виде, после чего обрабатывается специальными программными средствами (*программами распознавания образов*).

Планшетные сканеры. Планшетные сканеры предназначены для ввода графической информации с прозрачного или непрозрачного листового материала. Прин-

цип действия этих устройств состоит в том, что луч света, отраженный от поверхности материала (или прошедший сквозь прозрачный материал), фиксируется специальными элементами, называемыми *приборами с зарядовой связью* (ПЗС). Обычно элементы ПЗС конструктивно оформляют в виде линейки, располагаемой по ширине исходного материала. Перемещение линейки относительно листа бумаги выполняется механическим протягиванием линейки при неподвижной установке листа или протягиванием листа при неподвижной установке линейки.

Основными потребительскими параметрами планшетных сканеров являются:

- разрешающая способность;
- производительность;
- динамический диапазон;
- максимальный размер сканируемого материала.

Разрешающая способность планшетного сканера зависит от плотности размещения приборов ПЗС на линейке, а также от точности механического позиционирования линейки при сканировании. Типичный показатель для офисного применения: 600–1200 *dpi* (*dpi — dots per inch, количество точек на дюйм*). Для профессионального применения характерны показатели 1200–3000 *dpi*.

Производительность сканера определяется продолжительностью сканирования листа бумаги стандартного формата и зависит как от совершенства механической части устройства, так и от типа интерфейса, использованного для сопряжения с компьютером.

Динамический диапазон определяется логарифмом отношения яркости наиболее светлых участков изображения к яркости наиболее темных участков. Типовой показатель для сканеров офисного применения составляет 1,8–2,0, а для сканеров профессионального применения — от 2,5 (для непрозрачных материалов) до 3,5 (для прозрачных материалов).

Ручные сканеры. Принцип действия ручных сканеров в основном соответствует планшетным. Разница заключается в том, что протягивание линейки ПЗС в данном случае выполняется вручную. Равномерность и точность сканирования при этом обеспечиваются неудовлетворительно, и разрешающая способность ручного сканера составляет 150–300 *dpi*.

Барабанные сканеры. В сканерах этого типа исходный материал закрепляется на цилиндрической поверхности барабана, вращающегося с высокой скоростью. Устройства этого типа обеспечивают наивысшее разрешение (2400–5000 *dpi*) благодаря применению не ПЗС, а фотоэлектронных умножителей. Их используют для сканирования исходных изображений, имеющих высокое качество, но недостаточные линейные размеры (фотонегативов, слайдов и т. п.)

Сканеры форм. Предназначены для ввода данных со стандартных форм, заполненных механически или «от руки». Необходимость в этом возникает при проведении переписей населения, обработке результатов выборов и анализе анкетных данных. От сканеров форм не требуется высокой точности сканирования, но быстродействие играет повышенную роль и является основным потребительским параметром.

Штрих-сканеры. Эта разновидность ручных сканеров предназначена для ввода данных, закодированных в виде штрих-кода. Такие устройства имеют применение в розничной торговой сети.

Графические планшеты (дигитайзеры) предназначены для ввода художественной графической информации. Существует несколько различных принципов действия графических планшетов, но в основе всех их лежит фиксация перемещения специального пера относительно планшета. Устройства удобны для художников и иллюстраторов, поскольку позволяют им создавать экранные изображения привычными приемами, наработанными для традиционных инструментов (карандаш, перо, кисть).

Цифровые фотокамеры. Как и сканеры, эти устройства воспринимают графические данные с помощью приборов с зарядовой связью, объединенных в прямоугольную матрицу. Основным параметром цифровых фотоаппаратов является разрешающая способность, которая напрямую связана с количеством ячеек ПЗС в матрице. Наилучшие потребительские модели в настоящее время имеют 2–4 млн. ячеек ПЗС и, соответственно, обеспечивают разрешение изображения до 1600×1200 точек и выше. У профессиональных моделей эти параметры еще выше.

Устройства вывода данных

В качестве устройств вывода данных, дополнительных к монитору, используют печатающие устройства (принтеры), позволяющие получать копии документов на бумаге или прозрачном носителе. По принципу действия различают матричные, лазерные, светодиодные и струйные принтеры.

Матричные принтеры. Это простейшие печатающие устройства. Данные выводятся на бумагу в виде оттиска, образующегося при ударе цилиндрических стержней («иголок») через красящую ленту. Качество печати матричных принтеров напрямую зависит от количества иголок в печатающей головке. Наибольшее распространение имеют *9-игольчатые* и *24-игольчатые* матричные принтеры. Последние позволяют получать оттиски документов, практически не уступающие по качеству документам, исполненным на пишущей машинке. В настоящее время матричные принтеры считаются устаревшими и практически не выпускаются.

Производительность работы матричных принтеров оценивают по количеству печатаемых знаков в секунду (*cps — characters per second*). Обычными режимами работы матричных принтеров являются: *draft* — режим черновой печати, *normal* — режим обычной печати и режим *NLQ* (*Near Letter Quality*), который обеспечивает качество печати, близкое к качеству пишущей машинки.

Лазерные принтеры обеспечивают высокое качество печати, не уступающее, а во многих случаях и превосходящее полиграфическое. Они отличаются также высокой скоростью печати, которая измеряется в страницах в минуту (*ppm — page per minute*). Как и в матричных принтерах, итоговое изображение формируется из отдельных точек.

Принцип действия лазерных принтеров следующий:

- в соответствии с поступающими данными лазерная головка испускает световые импульсы, которые отражаются от зеркала и попадают на поверхность светочувствительного барабана;

- горизонтальная развертка изображения выполняется вращением зеркала;
- участки поверхности светочувствительного барабана, получившие световой импульс, приобретают статический заряд;
- барабан при вращении проходит через контейнер, наполненный красящим составом (тонером), и тонер закрепляется на участках, имеющих статический заряд;
- при дальнейшем вращении барабана происходит контакт его поверхности с бумажным листом, в результате чего происходит перенос тонера на бумагу;
- лист бумаги с нанесенным на него тонером протягивается через нагревательный элемент, в результате чего частицы тонера спекаются и закрепляются на бумаге.

К основным параметрам лазерных принтеров относятся:

- разрешающая способность, *dpi* (*dots per inch — точек на дюйм*);
- производительность (страниц в минуту);
- формат используемой бумаги;
- объем собственной оперативной памяти.

При выборе лазерного принтера необходимо также учитывать параметр стоимости оттиска, то есть стоимость расходных материалов для получения одного печатного листа стандартного формата А4. К расходным материалам относятся тонер и барабан, который после печати определенного количества оттисков утрачивает свои свойства. В качестве единицы измерения используют *цент на страницу* (имеются в виду центы США). В настоящее время теоретический предел по этому показателю составляет порядка 1,0–1,5. На практике лазерные принтеры массового применения обеспечивают значения от 2,0 до 6,0.

Основное преимущество лазерных принтеров заключается в возможности получения высококачественных отпечатков. Уже модели среднего класса обеспечивают разрешение печати до 600 *dpi*, а профессиональные модели — до 1800 *dpi* и выше.

Светодиодные принтеры. Принцип действия светодиодных принтеров похож на принцип действия лазерных принтеров. Разница заключается в том, что источником света является не лазерная головка, а линейка светодиодов. Поскольку эта линейка расположена по всей ширине печатаемой страницы, отпадает необходимость в механизме формирования горизонтальной развертки и вся конструкция получается проще, надежнее и дешевле. Типичная величина разрешения печати для светодиодных принтеров составляет порядка 600 *dpi*.

Струйные принтеры. В струйных печатающих устройствах изображение на бумаге формируется из пятен, образующихся при попадании капель красителя на бумагу. Выброс микрокапель красителя происходит под давлением, которое развивается в печатающей головке за счет парообразования. В некоторых моделях капля выбрасывается щелчком в результате пьезоэлектрического эффекта — этот метод позволяет обеспечить более стабильную форму капли, близкую к сферической.

Качество печати изображения во многом зависит от формы капли и ее размера, а также от характера впитывания жидкого красителя поверхностью бумаги. В этих условиях особую роль играют вязкостные свойства красителя и свойства бумаги.

К положительным свойствам струйных печатающих устройств следует отнести относительно небольшое количество движущихся механических частей и, соответственно, простоту и надежность механической части устройства и его относительно низкую стоимость. Основным недостатком, по сравнению с лазерными принтерами, является нестабильность получаемого разрешения, что ограничивает возможность их применения в черно-белой полутоновой печати.

Сегодня струйные принтеры нашли очень широкое применение в цветной печати. Благодаря простоте конструкции они намного превосходят цветные лазерные принтеры по показателю качество/цена. При разрешении выше 600 *dpi* они позволяют получать цветные оттиски, превосходящие по качеству цветные отпечатки, получаемые фотохимическими методами.

При выборе струйного принтера следует обязательно иметь виду параметр стоимости печати одного оттиска. Хотя цена струйных печатающих устройств заметно ниже, чем лазерных, стоимость печати одного оттиска на них может быть в несколько раз выше.

Устройства хранения данных

Необходимость во внешних устройствах хранения данных возникает в двух случаях:

- когда на вычислительной системе обрабатывается больше данных, чем можно разместить на базовом жестком диске;
- когда данные имеют повышенную ценность и необходимо выполнять регулярное резервное копирование на внешнее устройство (копирование данных в пределах того же жесткого диска не является резервным и только создает иллюзию безопасности).

В настоящее время для внешнего хранения данных используют несколько типов устройств, использующих магнитные или магнитооптические носители.

Стримеры. Стри́меры — это накопители на магнитной ленте. Их отличает сравнительно низкая цена. К недостаткам стримеров относят малую производительность (она связана прежде всего с тем, что магнитная лента — это устройство последовательного доступа) и недостаточную надежность (кроме электромагнитных наводок, ленты стримеров испытывают повышенные механические нагрузки и могут физически выходить из строя).

Емкость магнитных кассет (картриджей) для стримеров достигает нескольких десятков гигабайт. Дальнейшее повышение емкости за счет повышения плотности записи снижает надежность хранения, а повышение емкости за счет увеличения длины ленты сдерживается низким временем доступа к данным.

Накопители на съемных магнитных дисках. К этой категории относится несколько разных типов устройств, ни одно из которых так и не стало общепринятым стандартом. Например, *ZIP*-накопители выпускаются компанией *Iomega*, специализирующейся на создании внешних устройств для хранения данных. Устройство работает с дисковыми носителями, по размеру незначительно превышающими стандартные гибкие диски и имеющими емкость 100/250/750 Мбайт. Основным недостатком *ZIP*-нако-

пителей является отсутствие их совместимости со стандартными гибкими дисками 3,5 дюйма. Такой совместимостью обладают устройства *HiFD* компании *Sony*. Они позволяют использовать как специальные носители емкостью 200 Мбайт, так и обычные гибкие диски. Распространение этих устройств сдерживается высокой ценой.

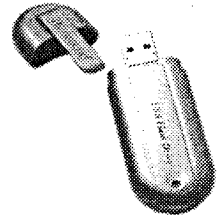
Накопители *JAZ*, как и *ZIP*-накопители, выпускаются компанией *Iomega*. По своим характеристикам *JAZ*-носитель приближается к жестким дискам, но в отличие от них является сменным. В зависимости от модели накопителя на одном диске можно разместить 1 или 2 Гбайт данных.

Магнитооптические устройства. Эти устройства получили широкое распространение в компьютерных системах высокого уровня благодаря своей универсальности. С их помощью решаются задачи резервного копирования, обмена данными и их накопления. Однако достаточно высокая стоимость приводов и носителей не позволяет отнести их к устройствам массового спроса.

В этом секторе параллельно развиваются 5,25- и 3,5-дюймовые накопители, носители для которых отличаются в основном форм-фактором и емкостью. Последнее поколение носителей формата 5,25" достигает емкости 5,2 Гбайт. Емкость носителей 3,5" несколько ниже, от 640 Мбайт до 2,3 Гбайт.

В перспективе ожидается появление накопителей заметно большего объема (до нескольких десятков Гбайт).

Флэш-диски. Это современное устройство хранения данных на основе энергонезависимой флэш-памяти. Устройство имеет минимальные размеры и допускает «горячее» подключение в разъем *USB*, после чего распознается как жесткий диск, причем не требует установки драйвера. Объем флэш-дисков может составлять от 32 Мбайт до 1 Гбайт, их распространение сдерживает относительно высокая цена.



Устройства обмена данными

Модем. Устройство, предназначенное для обмена информацией между удаленными компьютерами по каналам связи, принято называть модемом (МОдулятор + ДЕМодулятор). При этом под каналом связи понимают физические линии (проводные, оптоволоконные, кабельные, радиочастотные), способ их использования (коммутируемые и выделенные) и способ передачи данных (цифровые или аналоговые сигналы). В зависимости от типа канала связи устройства приема-передачи подразделяют на радиомодемы, кабельные модемы и прочие. Наиболее широкое применение нашли модемы, ориентированные на подключение к коммутируемым телефонным каналам связи.

Цифровые данные, поступающие в модем из компьютера, преобразуются в нем путем модуляции (по амплитуде, частоте, фазе) в соответствии с избранным стандартом (протоколом) и направляются в телефонную линию. Модем-приемник, понимающий данный протокол, осуществляет обратное преобразование (демодуляцию) и пересылает восстановленные цифровые данные в свой компьютер. Таким образом обеспечивается удаленная связь между компьютерами и обмен данными между ними.

К основным потребительским параметрам модемов относятся:

- производительность (бит/с);
- поддерживаемые протоколы связи и коррекции ошибок;
- шинный интерфейс, если модем внутренний (*ISA* или *PCI*).

От производительности модема зависит объем данных, передаваемых в единицу времени. От поддерживаемых протоколов зависит эффективность взаимодействия данного модема с сопредельными модемами (вероятность того, что они вступят во взаимодействие друг с другом при оптимальных настройках). От шинного интерфейса в настоящее время пока зависит только простота установки и настройки модема (в дальнейшем при общем совершенствовании каналов связи шинный интерфейс начнет оказывать влияние и на производительность).

Практическое занятие

Упражнение 3.1. Подключение оборудования к системному блоку

Работа выполняется под руководством преподавателя (инструктора).

1. Убедитесь в том, что компьютерная система обесточена.
2. Разверните системный блок задней стенкой к себе.
3. По форме разъема клавиатуры установите форм-фактор материнской платы (разъем формата *PS/2* — форм-фактор *ATX*, разъем формата *DIN5* — *AT*).
4. Установите местоположение следующих разъемов:
   - питания системного блока;
   - питания монитора;
   - сигнального кабеля монитора;
   - клавиатуры;
   - последовательных портов (два разъема);
   - параллельного порта.
5. Убедитесь в том, что все разъемы, выведенные на заднюю стенку системного блока, невзаимозаменяемы, то есть каждое базовое устройство подключается одним-единственным способом.
6. При наличии звуковой карты рассмотрите ее разъемы. Установите местоположение следующих разъемов:
   - подключения головных телефонов;
   - подключения микрофона;
   - вывода сигнала на внешний усилитель;
   - подключения внешних электромузыкальных инструментов и средств управления компьютерными играми (джойстик, джойпад, геймпад и т. п.).

7. Изучите способ подключения мыши. Мышь может подключаться к разъему последовательного порта или к специальному порту *PS/2*, имеющему разъем круглой формы. Последний способ является более современным и удобным. В этом случае мышь имеет собственный выделенный порт, что исключает возможность ее конфликта с другими устройствами, подключаемыми к последовательным портам. Последние модели могут подключаться к клавиатуре через разъем интерфейса *USB*.

Упражнение 3.2. Изучение компонентов системного блока

Работа выполняется под руководством преподавателя (инструктора).

1. Убедитесь в том, что компьютерная система обесточена.
2. Установите местоположение блока питания.
3. Установите местоположение материнской платы.
4. Установите характер подключения материнской платы к блоку питания. Для материнских плат в форм-факторе *AT* подключение питания выполняется двумя разъемами. Обратите внимание на расположение проводников черного цвета — оно важно для правильной стыковки разъемов.
5. Установите местоположение жесткого диска. Установите местоположение его разъема питания. Проследите направление шлейфа проводников, связывающего жесткий диск с материнской платой. Обратите внимание на местоположение проводника, окрашенного в красный цвет (он должен быть расположен рядом с разъемом питания).
6. Установите местоположения дисководов гибких дисков и дисковода *CD-ROM*. Проследите направление их шлейфов проводников и обратите внимание на положение проводника, окрашенного в красный цвет, относительно разъема питания.
7. Установите местоположение звуковой карты и платы видеоадаптера.
8. При наличии прочих дополнительных устройств задайте инструктору вопросы об их назначении.

Упражнение 3.3. Изучение компонентов материнской платы

Работа выполняется под руководством преподавателя (инструктора).

1. Убедитесь в том, что компьютерная система обесточена.
2. Установите местоположение процессора и изучите организацию его системы охлаждения. По маркировке определите тип процессора и фирму-изготовителя.
3. Установите местоположение разъемов для установки модулей оперативной памяти. Выясните их количество и тип используемых модулей.

4. Установите местоположение слотов для установки плат расширения. Выясните их количество и тип (*ISA, VLB, PCI, AGP*). Зафиксируйте их различия по форме и цвету:

Разъем шины	Цвет	Размер
ISA	черный	длинный
PCI	белый	средний
AGP	коричневый	короткий

5. Установите местоположение микросхемы ПЗУ. По наклейке на ней определите производителя системы *BIOS* данного компьютера.
6. Установите местоположение микросхем системного комплекта (чипсета). По маркировке определите тип комплекта и фирму-изготовитель.
7. Заполните отчетные таблицы:

	Изготовитель	Модель
Процессор		
Чипсет		
Система BIOS		— — — — — — — —

Разъемы модулей оперативной памяти		Слоты для установки плат расширения	
Тип	Количество	Тип	Количество
		AGP	
		PCI	
		...	

Упражнение 3.4. Исследование порядка запуска компьютера

Работа выполняется под руководством преподавателя (инструктора).

1. Если монитор вычислительной системы имеет питание, отдельное от системного блока, включите монитор.
2. Включите компьютерную систему выключателем системного блока.
3. При подаче питания на процессор происходит его обращение к микросхеме ПЗУ и запуск программы, инициализирующей работу компьютера. В этот момент на экране монитора наблюдается сообщение о версии *BIOS*.
4. Для наблюдения сообщений, поступающих от компьютера в процессе запуска, используйте клавишу Pause/Break. Она приостанавливает загрузку и дает возможность внимательно прочесть сообщение. Для продолжения запуска используйте клавишу ENTER.
5. Процедура инициализации запускает процедуру *POST*, выполняющую самотестирование базовых устройств (*POST — Power-On Self-Test*). В этот момент

на экране наблюдается сообщение Memory Test: и указание объема проверенной памяти компьютера.

6. При отсутствии дефектов в оперативной памяти или в клавиатуре происходит обращение к микросхеме *CMOS*, в которой записаны данные, определяющие состав компьютерной системы и ее настройки. На экране монитора эти данные отображаются в таблице System Configuration. Приостановив запуск с помощью клавиши PAUSE/BREAK, изучите таблицу и установите:
   - сколько жестких дисков имеет компьютерная система и каков их объем;
   - имеются ли дисководы гибких дисков и каковы параметры используемых гибких дисков;
   - сколько последовательных и параллельных портов имеется в наличии;
   - к какому типу относятся микросхемы, размещенные в банках памяти.

   Продолжите запуск клавишей ENTER.
7. Установив параметры жесткого диска, компьютерная система обращается в его системную область, находит там операционную систему и начинает ее загрузку. Далее работа с компьютером выполняется под управлением операционной системы.
8. Дождавшись окончания запуска операционной системы, выясните у инструктора (преподавателя) порядок завершения работы с компьютером. Приведите компьютер в исходное состояние.

Упражнение 3.5. Настройка компьютерной системы средствами программы SETUP

Работа выполняется под руководством преподавателя (инструктора).

Программа SETUP входит в состав базовой системы ввода-вывода и предназначена для первичной настройки аппаратной конфигурации вычислительной системы. Основная задача настройки — обеспечить возможность автоматического определения состава системы средствами *BIOS*. Дополнительная задача — оптимизировать настройки и повысить эффективность всей системы в целом.

В большинстве случаев программа SETUP вызывается нажатием клавиши DELETE сразу после включения питания. В отдельных случаях может использоваться иная клавиша или комбинация клавиш — необходимая информация выдается на экран монитора при запуске компьютера.

Неквалифицированное изменение настроек микросхемы CMOS может привести к выходу компьютерной системы из строя. В связи с этим примите следующие меры:
- не вносите никаких изменений в настройки без указания инструктора (преподавателя);
- записью на отдельном листе бумаги четко фиксируйте все параметры до их изменения и после;
- по окончании работы закройте программу SETUP без сохранения внесенных изменений. Перед закрытием программы обратитесь к инструктору (преподавателю) для контроля.

1. Если монитор вычислительной системы имеет питание, отдельное от системного блока, включите монитор.

2. Включите компьютерную систему.
3. При появлении информации на экране нажмите клавишу DELETE — произойдет запуск программы SETUP и откроется меню, представленное на рис. 3.5.
4. С помощью клавиш управления курсором выберите пункт меню Standard CMOS Features (Стандартные настройки микросхемы CMOS).

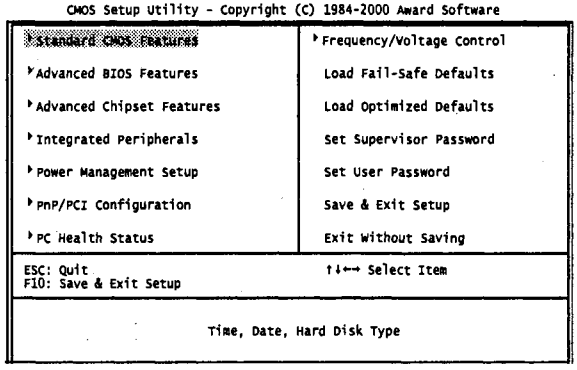


Рис. 3.5. Титульный экран программы SETUP

5. В открывшемся окне проверьте установку системных часов и системного календаря. Выбор настраиваемого параметра выполняется клавишами управления курсором, а изменение параметра — клавишами PAGE UP/PAGE DOWN.
6. Вернитесь в предыдущее меню с помощью клавиши ESC.
7. Выберите пункт Advanced BIOS Features (Настройки параметров BIOS). Нажмите клавишу ENTER.
8. В открывшемся окне проверьте, с какого диска начинается запуск компьютера. Последовательность запуска задается в пункте BOOT SEQUENCE. С помощью клавиш PAGE UP и PAGE DOWN просмотрите все возможные для данного компьютера варианты запуска. Особое внимание обратите на вариант запуска, начинающегося с жесткого диска C: (он используется при штатной работе), и на вариант запуска, начинающегося с гибкого диска A:, — он используется при восстановлении работоспособности компьютера, если загрузка с жесткого диска по каким-то причинам невозможна.
9. Обратите внимание на пункт Typematic Rate Setting — если он включен (Enabled), то путем настройки *BIOS* можно управлять настройкой функции автоматического повтора символов для клавиатуры (см. раздел 3.1). В этом случае интервал времени до начала повтора определяется установкой параметра Typematic Rate Delay (Задержка перед повтором), измеряемого в миллисекундах, а частота повтора определяется установкой параметра Typematic Rate (Частота повтора), измеряемого в знаках в секунду.
10. Вернитесь в предыдущее меню нажатием клавиши ESC.
11. Завершите работу с программой *SETUP* без сохранения результатов изменения. Для этого нажмите клавишу ESC и при получении запроса подтвердите выход без сохранения изменений нажатием клавиши Y (*Yes — Да*).

Операционная система представляет собой комплекс системных и служебных программных средств. С одной стороны, она опирается на базовое программное обеспечение компьютера, входящее в его систему *BIOS* (*базовая система ввода-вывода*); с другой стороны, она сама является опорой для программного обеспечения более высоких уровней — прикладных и большинства служебных приложений. *Приложениями операционной системы* принято называть программы, предназначенные для работы под управлением данной системы.

Основная функция всех операционных систем — посредническая. Она заключается в обеспечении нескольких видов интерфейса:

- интерфейса между пользователем и программно-аппаратными средствами компьютера (*интерфейс пользователя*);
- интерфейса между программным и аппаратным обеспечением (*аппаратно-программный интерфейс*);
- интерфейса между разными видами программного обеспечения (*программный интерфейс*).

Даже для одной аппаратной платформы, например такой, как *IBM PC*, существует несколько операционных систем. Различия между ними рассматривают в двух категориях: внутренние и внешние. Внутренние различия характеризуются методами реализации основных функций. Внешние различия определяются наличием и доступностью приложений данной системы, необходимых для удовлетворения технических требований, предъявляемых к конкретному рабочему месту.

4.1. Обеспечение интерфейса пользователя

Режимы работы с компьютером

Все операционные системы способны обеспечивать как *пакетный*, так и *диалоговый режим* работы с пользователем. В пакетном режиме операционная система автоматически исполняет заданную последовательность команд. Суть диалогового режима

состоит в том, что операционная система находится в ожидании команды пользователя и, получив ее, приступает к исполнению, а исполнив, возвращает отклик и ждет очередной команды. Диалоговый режим работы основан на использовании *прерываний процессора* и *прерываний BIOS* (которые, в свою очередь, также основаны на использовании прерываний процессора). Опираясь на эти *аппаратные прерывания*, операционная система создает свой комплекс *системных прерываний*. Способность операционной системы прервать текущую работу и отреагировать на события, вызванные пользователем с помощью управляющих устройств, воспринимается нами как диалоговый режим работы.

Виды интерфейсов пользователя

Интерфейс командной строки. По реализации интерфейса пользователя различают *неграфические* и *графические операционные системы*. Неграфические операционные системы реализуют *интерфейс командной строки*. Основным устройством управления в данном случае является клавиатура. Управляющие команды вводят в поле командной строки, где их можно и редактировать. Исполнение команды начинается после ее утверждения, например нажатием клавиши ENTER. Для компьютеров платформы *IBM PC* интерфейс командной строки обеспечивается семейством операционных систем под общим названием *MS-DOS* (версии от *MS-DOS 1.0* до *MS-DOS 6.2*).

Графический интерфейс. Графические операционные системы реализуют более сложный тип интерфейса, в котором в качестве органа управления кроме клавиатуры может использоваться мышь или адекватное устройство позиционирования. Работа с графической операционной системой основана на взаимодействии активных и пассивных экранных элементов управления.

Активные и пассивные элементы управления. В качестве активного элемента управления выступает *указатель мыши* — графический объект, перемещение которого на экране синхронизировано с перемещением мыши.

В качестве пассивных элементов управления выступают графические *элементы управления приложений* (экранные кнопки, значки, переключатели, флажки, раскрывающиеся списки, строки меню и многие другие).

Характер взаимодействия между активными и пассивными элементами управления выбирает сам пользователь. В его распоряжении приемы наведения указателя мыши на элемент управления, щелчки кнопками мыши и другие средства.

4.2. Обеспечение автоматического запуска

Все операционные системы обеспечивают свой автоматический запуск. Для дисковых операционных систем в специальной (*системной*) области диска создается запись программного кода. Обращение к этому коду выполняют программы, находящиеся в базовой системе ввода-вывода (*BIOS*). Завершая свою работу, они дают команду на загрузку и исполнение содержимого системной области диска.

Недисковые операционные системы характерны для специализированных вычислительных систем, в частности для компьютеризированных устройств автомати-

ческого управления. Математическое обеспечение, содержащееся в микросхемах ПЗУ таких компьютеров, можно условно рассматривать как аналог операционной системы. Автоматический запуск такой системы осуществляется аппаратно. При подаче питания процессор обращается к фиксированному физическому адресу ПЗУ (его можно изменять аппаратно с использованием логических микросхем), с которого начинается запись программы инициализации операционной системы.

4.3. Организация файловой системы

Все современные дисковые операционные системы обеспечивают создание файловой системы, предназначенной для хранения данных на дисках и обеспечения доступа к ним. Принцип организации файловой системы — табличный. Поверхность жесткого диска рассматривается как трехмерная матрица, измерениями которой являются номера *поверхности*, *цилиндра* и *сектора*. Под цилиндром понимается совокупность всех дорожек, принадлежащих разным поверхностям и находящихся на равном удалении от оси вращения. Данные о том, в каком месте диска записан тот или иной файл, хранятся в системной области диска. Формат служебных данных определяется конкретной файловой системой. Нарушение целостности служебных сведений приводит к невозможности воспользоваться данными, записанными на диске. Поэтому к системной области предъявляются особые требования по надежности. Целостность, непротиворечивость и надежность этих данных регулярно контролируется средствами операционной системы.

Наименьшей физической единицей хранения данных является сектор. Размер сектора равен 512 байт. Теоретически возможна самостоятельная адресация для каждого сектора. Но для дисков большого объема такой подход неэффективен, а для некоторых файловых систем — и просто невозможен. В связи с этим группы секторов объединяются в кластеры. *Кластер является наименьшей единицей адресации при обращении к данным.* Размер кластера, в отличие от размера сектора, строго не фиксирован. Обычно он зависит от емкости диска.

Операционные системы *MS-DOS*, *OS/2*, *Windows 95* и другие используют файловую систему на основе таблиц размещения файлов (*FAT*-таблицы), состоящих из 16-разрядных полей. Такая файловая система называется *FAT16*. Она позволяет разместить в *FAT*-таблицах не более 65 536 записей (2^{16}) о местоположении единиц хранения данных. Для дисков объемом от 1 до 2 Гбайт длина кластера составляет 32 Кбайт (64 сектора). Это не вполне рациональный расход рабочего пространства, поскольку любой файл (даже очень маленький) полностью оккупирует весь кластер, которому соответствует только одна адресная запись в таблице размещения файлов. Даже если файл достаточно велик и располагается в нескольких кластерах, все равно в его конце образуется некий остаток, нерационально расходующий целый кластер.

Для жестких дисков, объем которых приближается к 2 Гбайт, потери, связанные с неэффективностью этой файловой системы, весьма значительны и могут составлять от 25% до 40% полной емкости диска, в зависимости от среднего размера хранящихся файлов. С дисками же размером более 2 Гбайт файловая система *FAT16* вообще работать не может.

Начиная с *Windows 98* операционные системы семейства *Windows* (*Windows 98*, *Windows Me*, *Windows 2000*, *Windows XP*) поддерживают более совершенную версию файловой системы на основе *FAT*-таблиц — *FAT32* с 32-разрядными полями в таблице размещения файлов. Для дисков размером до 8 Гбайт эта система обеспечивает размер кластера 4 Кбайт (8 секторов).

Операционные системы *Windows NT* и *Windows XP* способны поддерживать совершенно другую файловую систему — *NTFS*. В ней хранение файлов организовано иначе — служебная информация хранится в Главной таблице файлов (*MFT*). В системе *NTFS* размер кластера не зависит от размера диска, и, потенциально, для очень больших дисков эта система должна работать эффективнее, чем *FAT32*. Однако с учетом типичных характеристик современных компьютеров можно говорить о том, что в настоящее время эффективность *FAT32* и *NTFS* примерно одинакова.

4.4. Обслуживание файловой структуры

Несмотря на то что данные о местоположении файлов хранятся в табличной структуре, пользователю они представляются в виде иерархической структуры — людям так удобнее, а все необходимые преобразования берет на себя операционная система. К функции обслуживания файловой структуры относятся следующие операции, происходящие под управлением операционной системы:

- создание файлов и присвоение им имен;
- создание каталогов (папок) и присвоение им имен;
- переименование файлов и каталогов (папок);
- копирование и перемещение файлов между дисками компьютера и между каталогами (папками) одного диска;
- удаление файлов и каталогов (папок);
- навигация по файловой структуре с целью доступа к заданному файлу, каталогу (папке);
- управление атрибутами файлов.

Создание и именование файлов

Файл — это именованная последовательность байтов произвольной длины. Поскольку из этого определения вытекает, что файл может иметь нулевую длину, то фактически создание файла состоит в присвоении ему имени и регистрации его в файловой системе — это одна из функций операционной системы. Даже когда мы создаем файл, работая в какой-то прикладной программе, в общем случае для этой операции привлекаются средства операционной системы.

По способам именования файлов различают «короткое» и «длинное» имя. До появления операционной системы *Windows 95* общепринятым способом именования файлов на компьютерах *IBM PC* было *соглашение 8.3*. Согласно этому соглашению, принятому в *MS-DOS*, имя файла состоит из двух частей: собственно *имени* и *расширения имени*. На имя файла отводится 8 символов, а на его расширение — 3 символа. Имя от расширения отделяется точкой. Как имя, так и расширение могут включать только алфавитно-цифровые символы латинского алфавита.

Соглашение 8.3 не является стандартом, и потому в ряде случаев отклонения от правильной формы записи допускаются как операционной системой, так и ее приложениями. Так, например, в большинстве случаев система «не возражает» против использования некоторых специальных символов (восклицательный знак, символ подчеркивания, дефис, тильда и т. п.), а некоторые версии *MS-DOS* даже допускают использование в именах файлов символов русского и других алфавитов. Сегодня имена файлов, записанные в соответствии с *соглашением 8.3*, считаются «короткими».

Основным недостатком «коротких» имен является их низкая содержательность. Далеко не всегда удается выразить несколькими символами характеристику файла, поэтому с появлением операционной системы *Windows 95* было введено понятие «длинного» имени. Такое имя может содержать до 256 символов. Этого вполне достаточно для создания содержательных имен файлов. «Длинное» имя может содержать любые символы, кроме девяти специальных: \ / : * ? « < > |. В имени разрешается использовать пробелы и несколько точек. Расширением имени считаются все символы, идущие после последней точки, их может быть и больше трех.

Введение длинных имен потребовало внесения изменений в организацию файловых систем на основе *FAT*. Появился термин *VFAT*, обозначающий файловую систему на основе *FAT* с поддержкой длинных имен. Файловая система *NTFS* поддерживает длинные имена с самого начала.

Наряду с «длинным» именем операционные системы семейства *Windows* создают также и короткое имя файла — оно необходимо для возможности работы с данным файлом на рабочих местах с устаревшими операционными системами.

Особенности использования длинных имен. Использование «длинных» имен файлов в операционных системах семейства *Windows* имеет ряд особенностей.

1. Если «длинное» имя файла включает пробелы, то в служебных операциях его надо заключать в кавычки. Рекомендуется не использовать пробелы, а заменять их символами подчеркивания.
2. В корневой папке диска (на верхнем уровне иерархической файловой структуры) нежелательно хранить файлы с длинными именами. В файловых системах на основе *FAT* количество единиц хранения в этой папке ограничено. Чем длиннее имена, тем меньше файлов можно разместить в корневой папке.
3. Кроме ограничения на длину имени файла (256 символов) существует гораздо более жесткое ограничение на длину *полного имени файла* (в него входит путь доступа к файлу, начиная от вершины иерархической структуры). Полное имя не может быть длиннее 260 символов.
4. В длинных именах файлов разрешается использовать символы любых алфавитов, в том числе и русского, но если документ готовится для передачи, с заказчиком (потребителем документа) необходимо согласовать возможность воспроизведения файлов с такими именами на его оборудовании.
5. Прописные и строчные буквы в именах не различаются операционной системой. Для нее имена Письмо.txt и письмо.txt соответствуют одному и тому же файлу. Однако отображаются символы разных регистров операционной системой

исправно. Если для наглядности желательно использовать прописные буквы, это можно делать.

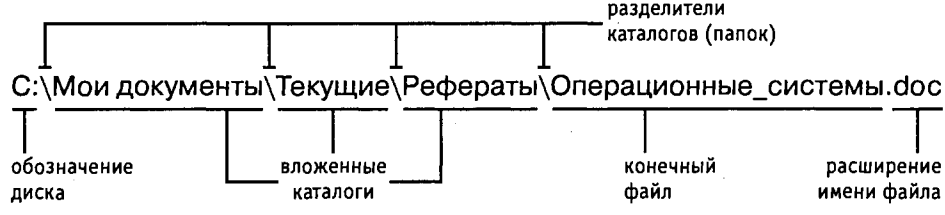
6. Программисты давно научились использовать расширение имени файла для передачи операционной системе, исполняющей программе или пользователю информации о том, к какому типу относятся данные, содержащиеся в файле, и о формате, в котором они записаны. В ранних операционных системах этот факт использовался мало. По существу, операционные системы *MS-DOS* анализировали только расширения .BAT (пакетные файлы с командами *MS-DOS*), .EXE, .COM (исполнимые файлы программ) и .SYS (системные файлы конфигурации). В современных операционных системах любое расширение имени файла может нести информацию для операционной системы. Операционные системы семейства *Windows* имеют средства для регистрации свойств типов файлов по расширению их имени, поэтому во многих случаях выбор расширения имени файла не является частным делом пользователя. Приложения этих систем предлагают выбрать только основную часть имени и указать тип файла, а соответствующее расширение имени приписывают автоматически.

Создание каталогов (папок)

Каталоги (*папки*) — важные элементы иерархической структуры, необходимые для обеспечения удобного доступа к файлам, если файлов на носителе слишком много. Файлы объединяются в каталоги по любому общему признаку, заданному их создателем (по типу, по принадлежности, по назначению, по времени создания и т. п.). Каталоги низких уровней вкладываются в каталоги более высоких уровней и являются для них *вложенными*. Верхним уровнем вложенности иерархической структуры является *корневой каталог* диска.

Все современные операционные системы позволяют создавать каталоги. Правила присвоения имени каталогу ничем не отличаются от правил присвоения имени файлу, хотя негласно для каталогов не принято задавать расширения имен.

Мы знаем, что в иерархических структурах данных адрес объекта задается *маршрутом* (*путем доступа*), ведущим от вершины структуры к объекту. При записи пути доступа к файлу, проходящего через систему вложенных каталогов, все промежуточные каталоги разделяются между собой определенным символом. Во многих операционных системах в качестве такого символа используется «\» (обратная косая черта), например:



Каталоги и папки. До появления операционной системы *Windows 95* при описании иерархической файловой структуры использовался введенный выше термин *каталог*. С появлением этой системы был введен новый термин — *папка*. В том, что

касается обслуживания файловой структуры носителя данных, эти термины равнозначны: каждому каталогу файлов на диске соответствует одноименная папка операционной системы. Основное отличие понятий *папка* и *каталог* проявляется не в организации хранения файлов, а в организации хранения объектов иной природы. Так, например, в операционных системах семейства *Windows* существуют специальные папки, представляющие собой удобные логические структуры, которым не соответствует ни один каталог диска.

Копирование и перемещение файлов

В неграфических операционных системах операции копирования и перемещения файлов выполняются вводом прямой команды в поле командной строки. При этом указывается имя команды, путь доступа к каталогу-источнику и путь доступа к каталогу-приемнику.

В графических операционных системах существуют приемы работы с устройством позиционирования, позволяющие выполнять эти команды наглядными методами.

Удаление файлов и каталогов (папок)

Средства удаления данных не менее важны для операционной системы, чем средства их создания, поскольку ни один носитель данных не обладает бесконечной емкостью. Существует как минимум три режима удаления данных: *удаление*, *уничтожение* и *стирание*, хотя операционные системы обеспечивают только два первых режима (режим надежного стирания данных можно обеспечить лишь специальными программными средствами).

Удаление файлов является временным. В операционных системах семейства *Windows* оно организовано с помощью специальной папки, которая называется Корзина. При удалении файлов и папок они перемещаются в Корзину. Эта операция происходит на уровне файловой структуры операционной системы (изменяется только путь доступа к файлам). На уровне файловой системы жесткого диска ничего не происходит — файлы остаются в тех же секторах, где и были записаны.

Уничтожение файлов происходит при их удалении в операционной системе *MS-DOS* или при очистке Корзины в операционных системах семейства *Windows*. В этом случае файл полностью удаляется из файловой структуры операционной системы, но на уровне файловой системы диска с ним происходят лишь незначительные изменения. В таблице размещения файлов он помечается как удаленный, хотя физически остается там же, где и был. Это сделано для минимизации времени операции. При этом открывается возможность записи новых файлов в кластеры, помеченные как «свободные».

Для справки укажем, что операция *стирания файлов*, выполняемая специальными служебными программами, состоит именно в том, чтобы заполнить якобы свободные кластеры, оставшиеся после уничтоженного файла, случайными данными. Поскольку даже после перезаписи данных их еще можно восстановить специальными аппаратными средствами (путем анализа остаточного магнитного гистерезиса), для надежного стирания файлов требуется провести не менее пяти актов

случайной перезаписи в одни и те же сектора. Эта операция весьма продолжительна, и, поскольку массовому потребителю она не нужна, ее не включают в стандартные функции операционных систем.

Навигация по файловой структуре

Навигация по файловой структуре является одной из наиболее используемых функций операционной системы. Удобство этой операции часто воспринимают как удобство работы с операционной системой. В операционных системах, имеющих интерфейс командной строки, навигацию осуществляют путем ввода команд перехода с диска на диск или из каталога в каталог. В связи с крайним неудобством такой навигации широкое применение нашли специальные служебные программы, называемые *файловыми оболочками*.

Как и операционные системы, файловые оболочки бывают неграфическими и графическими. Наиболее известная неграфическая файловая оболочка для *MS-DOS* — диспетчер файлов *Norton Commander*. Роль графической файловой оболочки для *MS-DOS* в свое время исполняли программы *Windows 1.0* и *Windows 2.0*, которые постепенно развились до понятия *операционной среды* (в версиях *Windows 3.x*) и далее до самостоятельной операционной системы (начиная с *Windows 95*).

С приемами навигации в современных графических операционных системах мы познакомимся при их изучении.

Управление атрибутами файлов

Кроме имени и расширения имени файла операционная система хранит для каждого файла дату его создания (изменения) и несколько флаговых величин, называемых *атрибутами файла*. Атрибуты — это дополнительные параметры, определяющие свойства файлов. Операционная система позволяет их контролировать и изменять; состояние атрибутов учитывается при проведении автоматических операций с файлами.

Основных атрибутов четыре:

- Только для чтения (Read only);
- Скрытый (Hidden);
- Системный (System);
- Архивный (Archive).

Атрибут Только для чтения ограничивает возможности работы с файлом. Его установка означает, что файл не предназначен для внесения изменений.

Атрибут Скрытый сигнализирует операционной системе о том, что данный файл не следует отображать на экране при проведении файловых операций. Это мера защиты против случайного (умышленного или неумышленного) повреждения файла.

Атрибутом Системный помечаются файлы, обладающие важными функциями для работы самой операционной системы. Его отличительная особенность в том, что средствами операционной системы его изменить нельзя. Как правило, большинство файлов, имеющих установленный атрибут Системный, имеют также и установленный атрибут Скрытый.

Атрибут Архивный в прошлом использовался для работы программ резервного копирования. Предполагалось, что любая программа, изменяющая файл, должна автоматически устанавливать этот атрибут, а средство резервного копирования должно его сбрасывать. Таким образом, очередному резервному копированию подлежали только те файлы, у которых этот атрибут был установлен. Современные программы резервного копирования используют другие средства для установления факта изменения файла, и данный атрибут во внимание не принимается, а его изменение вручную средствами операционной системы не имеет практического значения.

4.5. Управление установкой, исполнением и удалением приложений

Понятие многозадачности

Работа с приложениями составляет наиболее важную часть работы операционной системы. Это очевидно, если вспомнить, что основная функция операционной системы состоит в обеспечении интерфейса приложений с аппаратными и программными средствами вычислительной системы, а также с пользователем. С точки зрения управления исполнением приложений различают *однозадачные* и *многозадачные* операционные системы.

Однозадачные операционные системы (например, *MS-DOS*) передают все ресурсы вычислительной системы одному исполняемому приложению и не допускают ни параллельного выполнения другого приложения (*полная многозадачность*), ни его приостановки и запуска другого приложения (*вытесняющая многозадачность*). В то же время, параллельно с однозадачными операционными системами возможна работа специальных программ, называемых *резидентными*. Такие программы не опираются на операционную систему, а непосредственно работают с процессором, используя его систему прерываний.

Большинство современных графических операционных систем — *многозадачные*. Они управляют распределением ресурсов вычислительной системы между задачами и обеспечивают:

- возможность одновременной или поочередной работы нескольких приложений;
- возможность обмена данными между приложениями;
- возможность совместного использования программных, аппаратных, сетевых и прочих ресурсов вычислительной системы несколькими приложениями.

Вопросы надежности

От того, как операционная система управляет работой приложений, во многом зависит надежность всей вычислительной системы. Операционная система должна предоставлять возможность прерывания работы приложений по желанию пользователя и снятия сбойной задачи без ущерба для работы других приложений. При этом требование надежности операционной системы может входить в противоречие с требованием ее универсальности.

У операционных систем семейства *Windows* последних поколений долгое время наблюдались две линии развития. В линию универсальных операционных систем входили *Windows 95*, *Windows 98* и *Windows Me*. Эти системы могут испытывать общесистемные сбои из-за работы с приложениями, недостаточно четко соблюдающими спецификацию операционной системы. Операционные системы *Windows NT* и *Windows 2000* обладают повышенной устойчивостью и не выходят из строя при сбое приложений. Однако они менее универсальны, и, соответственно, парк доступных приложений для них ограничен.

Попытка объединить достоинства обеих линий сделана в операционной системе *Windows XP*. Эта система сегодня активно распространяется по массовым многоцелевым вычислительным системам, но постепенно проникает и на специализированные рабочие места, где требуется повышенная надежность при ограничении круга используемых программ.

Установка приложений

Для правильной работы приложений на компьютере они должны пройти операцию, называемую *установкой*. Необходимость в установке связана с тем, что разработчики программного обеспечения не могут заранее предвидеть особенности аппаратной и программной конфигурации вычислительной системы, на которой предстоит работать их программам. Таким образом, *дистрибутивный комплект* (*установочный пакет*) программного обеспечения, как правило, представляет собой не законченный программный продукт, а полуфабрикат, из которого в процессе установки на компьютере формируется полноценное рабочее приложение. При этом осуществляется привязка приложения к существующей аппаратно-программной среде и его настройка на работу именно в этой среде.

Устаревшие операционные системы (например, *MS-DOS*) не имеют средств для управления установкой приложений. Единственное средство, которое они предоставляют, — возможность запуска устанавливающей программы, прилагаемой к дистрибутивному комплекту. Такая установка отличается крайней простотой, но и невысокой надежностью, поскольку правильность привязки приложения к окружающей программно-аппаратной среде зависит от того, насколько разработчик устанавливающей программы сумел заранее предусмотреть возможные варианты конфигурации вычислительной системы конкретного пользователя.

Современные графические операционные системы берут на себя управление установкой приложений. Они управляют распределением ресурсов вычислительной системы между приложениями, обеспечивают доступ устанавливаемых приложений к драйверам устройств вычислительной системы, формируют общие ресурсы, которые могут использоваться разными приложениями, выполняют регистрацию установленных приложений и выделенных им ресурсов.

Удаление приложений

Процесс удаления приложений, как и процесс установки, имеет свои особенности и может происходить под управлением вычислительной системы. В таких операционных системах, где каждое приложение самообеспечено собственными ресурсами

(например, в *MS-DOS*), его удаление не требует специального вмешательства операционной системы. Для этого достаточно удалить каталог, в котором размещается приложение, со всем его содержимым.

В операционных системах, реализующих принцип совместного использования ресурсов (например, в системах семейства *Windows*), процесс удаления приложений имеет особенности. Нельзя допустить, чтобы при удалении одного приложения были удалены ресурсы, на которые опираются другие приложения, даже если эти ресурсы были когда-то установлены вместе с удаляемым приложением. В связи с этим удаление приложений происходит под строгим контролем операционной системы. Полнота удаления и надежность последующего функционирования операционной системы и оставшихся приложений во многом зависят от корректности установки и регистрации приложений в реестре операционной системы.

4.6. Взаимодействие с аппаратным обеспечением

Средства аппаратного обеспечения вычислительной техники отличаются гигантским многообразием. Существуют сотни различных моделей видеоадаптеров, звуковых карт, мониторов, принтеров, сканеров и прочего оборудования. Ни один разработчик программного обеспечения не в состоянии предусмотреть все варианты взаимодействия своей программы, например, с печатающим устройством.

Гибкость аппаратных и программных конфигураций вычислительных систем поддерживается за счет того, что каждый разработчик оборудования прикладывает к нему специальные программные средства управления — драйверы. Драйверы имеют *точки входа* для взаимодействия с прикладными программами, а диспетчеризация обращений прикладных программ к драйверам устройств — это одна из функций операционной системы. Строго говоря, выпуская устройство, например модем, его разработчик прикладывает к нему несколько драйверов, предназначенных для основных операционных систем, как-то: *MS-DOS*, *Windows XP*, *Linux* и т. п.

В операционных системах *MS-DOS* драйверы устройств загружаются как *резидентные программы*, напрямую работающие с процессором и другими устройствами материнской платы. Здесь участие операционной системы сводится лишь к тому, чтобы предоставить пользователю возможность загрузки драйвера — далее он сам перехватывает прерывания, используемые для обращения к устройству, и управляет его взаимодействием с вызывающей программой. Загрузка драйверов устройств может быть ручной или автоматической. При ручной загрузке после первоначальной загрузки компьютера пользователь сам выдает команды на загрузку драйверов. В автоматическом режиме команды на загрузку и настройку драйверов включаются в состав файлов, автоматически читаемых при загрузке компьютера. В *MS-DOS* такие файлы называются *файлами конфигурации*; их всего два — это файлы autoexec.bat и config.sys. В них прежде всего включают команды загрузки драйвера мыши, дисковода *CD-ROM*, звуковой карты, расширенной памяти (оперативная память, лежащая за пределами 1 Мбайт, рассматривается в *MS-DOS* как дополнительное устройство и требует специального драйвера), а также прочих устройств.

В операционных системах семейства *Windows* операционная система берет на себя все функции по установке драйверов устройств и передаче им управления от приложений. Во многих случаях операционная система даже не нуждается в драйверах, полученных от разработчика устройства, а использует драйверы из собственной базы данных.

Наиболее современные операционные системы позволяют управлять не только установкой и регистрацией программных драйверов устройств, но и процессом аппаратно-логического подключения. Каждое подключенное устройство может использовать до трех аппаратных ресурсов устройств материнской платы: *адресов внешних портов процессора*, *прерываний процессора* и *каналов прямого доступа к памяти*.

При некоторых способах подключения устройства к материнской плате (например, через шину *PCI*) есть техническая возможность организовать между ним и материнской платой обратную связь. Это позволяет операционной системе анализировать требования устройств о выделении им ресурсов и гибко реагировать на них, исключая захват одних и тех же ресурсов разными устройствами. Такой принцип динамического распределения ресурсов операционной системой получил название *plug-and-play*, а устройства, удовлетворяющие этому принципу, называются *самоустанавливающимися*.

Устройства, подключаемые по устаревшим шинам, не являются самоустанавливающимися. В этом случае операционная система не может выделять им ресурсы динамически, но, тем не менее, при распределении ресурсов для самоустанавливающихся устройств она учитывает ресурсы, захваченные ими.

4.7. Обслуживание компьютера

Предоставление основных средств обслуживания компьютера — одна из функций операционной системы. Обычно она решается внешним образом — включением в базовый состав операционной системы первоочередных служебных приложений.

Средства проверки дисков

Надежность работы дисков (особенно жесткого диска) определяет не только надежность работы компьютера в целом, но и безопасность хранения данных, ценность которых может намного превышать стоимость самого компьютера. Поэтому наличие средств для проверки дисков является обязательным требованием к любой операционной системе.

Средства проверки принято рассматривать в двух категориях: средства логической проверки, то есть проверки целостности файловой структуры, и средства физической диагностики поверхности. Логические ошибки, как правило, устраняются средствами самой операционной системы, а физические дефекты поверхности только локализуются — операционная система принимает во внимание факт повреждения магнитного слоя в определенных секторах и исключает их из активной работы.

Возможность возникновения логических ошибок зависит от типа файловой системы. Например, схема организации работы в системе *NTFS* вообще исключает

возникновение внутренних несоответствий в логической структуре, если не принимать во внимание возможность физического сбоя в процессе записи.

В системе на основе *FAT* логические ошибки файловой структуры имеют два характерных проявления: это *потерянные кластеры* или *общие кластеры*. Потерянные кластеры образуются в результате неправильного (или аварийного) завершения работы с компьютером. Так, например, ни в одной операционной системе нельзя выключать компьютер, если на нем запущены приложения, осуществляющие обмен информацией с дисками. Кроме того, в операционных системах *Windows* также нельзя выключать компьютер, если не исполнена специальная процедура завершения работы с операционной системой. Механизм образования потерянных кластеров выглядит так:

- во время работы с файлом приложение манипулирует с кластерами, занимая или освобождая их, и регистрирует сведения об этом в *FAT*-таблице, но не записывает полные сведения о файле в каталог;
- если при завершении работы с приложением происходит сохранение результатов деятельности, оно вносит окончательные изменения в *FAT*-таблицы и регистрирует данные, записанные в кластерах, как файл в каталоге;
- если при завершении работы с приложением файл уничтожается, информация не фиксируется в каталоге, а использованные кластеры освобождаются;
- если компьютер выключается до завершения работы с приложением, кластеры остаются помеченными как «занятые», но ссылки на них в каталоге не создается, так что согласно данным *FAT*-таблицы этим кластерам не соответствует ни один файл.

Ошибка, связанная с *потерянными кластерами*, легко парируется средствами операционной системы. При этом можно либо полностью освободить данные кластеры, либо превратить их в полноценные файлы, которые можно просмотреть в поисках ценной информации, утраченной во время сбоя.

Ошибка, проявляющаяся как *общие кластеры*, характеризуется тем, что, согласно данным *FAT*-таблиц, два или более файлов претендуют на то, что их данные находятся в одном и том же месте диска. При нормальной работе такой ситуации быть не может, и это свидетельствует об ошибке в *FAT*-таблицах. Причиной появления общих кластеров может стать самопроизвольное изменение данных в *FAT*-таблицах или некорректное восстановление ранее удаленных данных с помощью внесистемных средств. Некорректность может быть обусловлена нарушением порядка операций восстановления данных или неадекватностью средств восстановления данных (например, использованием средств *MS-DOS* для восстановления файлов, записанных средствами *Windows*).

Ошибка, связанная с общими кластерами, парируется повторной записью обоих конфликтующих файлов. Один из них обязательно испорчен и подлежит последующему удалению, но велика вероятность того, что испорчены оба файла.

Дополнительно к вышеуказанным логическим ошибкам операционные системы семейства *Windows* определяют логические ошибки, связанные с некорректной

записью даты создания файла и с представлением «короткого» имени файла для заданного «длинного» имени.

В операционной системе *Windows XP* проверка дисков, содержащих системную или служебную информацию рассматривается как потенциально опасная операция, способная поставить дальнейшую работу компьютера под угрозу. В этом случае проверка не выполняется немедленно, а назначается на время очередной перезагрузки системы. Такая же проверка системных дисков обычно производится и в случае аварийного отключения или аварийной перезагрузки компьютера.

Средства «сжатия» дисков

Некоторые операционные системы предоставляют служебные средства для программного «сжатия» дисков путем записи данных на диск в уплотненном виде посредством специального драйвера (резидентного для *MS-DOS* или работающего в фоновом режиме для *Windows*). Механизм работы этих средств будет рассмотрен в главе 14.

Средства управления виртуальной памятью

Ранние операционные системы ограничивали возможность использования приложений по объему необходимой для их работы оперативной памяти. Так, например, без специальных драйверов (*менеджеров оперативной памяти*) операционные системы *MS-DOS* ограничивали предельный размер исполняемых программ величиной около 640 Кбайт.

Современные операционные системы не только обеспечивают непосредственный доступ ко всему полю оперативной памяти, установленной в компьютере, но и позволяют ее расширить за счет создания так называемой *виртуальной памяти* на жестком диске. Виртуальная память реализуется в виде так называемого *файла подкачки*. В случае недостаточности оперативной памяти для работы приложения часть ее временно опорожняется с сохранением образа на жестком диске. В процессе работы приложений происходит многократный обмен между основной установленной оперативной памятью и файлом подкачки. Поскольку электронные операции в оперативной памяти происходят намного быстрее, чем механические операции взаимодействия с диском, увеличение размера оперативной памяти компьютера всегда благоприятно сказывается на ускорении операций и повышении производительности всей вычислительной системы.

Операционная система не только берет на себя весь необходимый обмен данными между ОЗУ и диском, но и позволяет в определенной степени управлять размером файла подкачки вручную.

Средства кэширования дисков

Поскольку, как уже было отмечено, взаимодействие процессора с дисками компьютера происходит намного медленнее операций обмена с оперативной памятью, операционная система принимает специальные меры по сохранению части прочитанных с диска данных в оперативной памяти. В случае, если по ходу работы процессору вновь потребуется обратиться к ранее считанным данным или про-

граммному коду, он может найти их в специальной области ОЗУ, называемой *дисковым кэшем*. В ранних операционных системах функции кэширования диска возлагались на специальное внешнее программное средство, подключаемое через файлы конфигурации. В современных операционных системах эту функцию включают в ядро системы и она работает автоматически, без участия пользователя, хотя определенная возможность настройки размера кэша за ним сохраняется.

Средства резервного копирования данных

Если на компьютере выполняется практическая работа, объем ценных (а зачастую и уникальных) данных нарастает с каждым днем. Ценность данных, размещенных на компьютере, принято измерять совокупностью затрат, которые может понести владелец в случае их утраты. Важным средством защиты данных является регулярное резервное копирование на внешний носитель. В связи с особой важностью этой задачи операционные системы обычно содержат базовые средства для выполнения резервного копирования.

4.8. Прочие функции операционных систем

Кроме основных (базовых) функций операционные системы могут предоставлять различные дополнительные функции. Конкретный выбор операционной системы определяется совокупностью предоставляемых функций и конкретными требованиями к рабочему месту.

Прочие функции операционных систем могут включать следующие:

- возможность поддерживать функционирование локальной компьютерной сети без специального программного обеспечения;
- обеспечение доступа к основным службам Интернета средствами, интегрированными в состав операционной системы;
- возможность создания системными средствами сервера Интернета, его обслуживание и управление, в том числе дистанционное посредством удаленного соединения;
- наличие средств защиты данных от несанкционированного доступа, просмотра и внесения изменений;
- возможность оформления рабочей среды операционной системы, в том числе и средствами, относящимися к категории мультимедиа;
- возможность обеспечения комфортной поочередной работы различных пользователей на одном персональном компьютере с сохранением персональных настроек рабочей среды каждого из них и ограничением доступа к конфиденциальной информации;
- возможность автоматического исполнения операций по обслуживанию компьютера и операционной системы в соответствии с заданным расписанием или под управлением удаленного сервера;
- возможность работы с компьютером для лиц, имеющих физические недостатки, связанные с органами зрения, слуха и другими.

Кроме всего вышеперечисленного, современные операционные системы могут включать минимальный набор прикладного программного обеспечения, которое можно использовать для исполнения простейших практических задач:

- чтение, редактирование и печать текстовых документов;
- создание и редактирование простейших рисунков;
- выполнение арифметических и математических расчетов;
- ведение дневников и служебных блокнотов;
- создание, передача и прием сообщений электронной почты;
- создание и редактирование факсимильных сообщений;
- воспроизведение и редактирование звукозаписи;
- воспроизведение видеозаписи;
- разработка и воспроизведение комплексных электронных документов, включающих текст, графику, звукозапись и видеозапись.

Этим возможности операционных систем не исчерпываются. По мере развития аппаратных средств вычислительной техники и средств связи функции операционных систем непрерывно расширяются, а средства их исполнения совершенствуются.

Подведение итогов

Основные достоинства персональной вычислительной техники проявляются в *диалоговом режиме* работы с пользователем. Диалоговый режим отличается от *пакетного* тем, что в ходе работы процессор регулярно приостанавливает выполнение текущих задач и обращается к другим устройствам и к программам, проверяя их состояние. Если пользователь использует какое-либо средство управления или извне поступает управляющий сигнал, процессор устанавливает этот факт и реагирует на него переходом на исполнение другой программы. Несмотря на то что в любой момент времени процессор работает по жестко заданным программам, динамичное переключение между ними создает впечатление гибкого управления работой компьютера.

Организацией работы процессора в таком режиме ведает относительно небольшая группа системных программ. Она образует *ядро операционной системы*. Дополнительно к ядру операционная система обладает средствами для:

- управления пользовательским интерфейсом компьютера;
- управления аппаратно-программными интерфейсами компьютера;
- обслуживания файловой системы;
- управления распределением оперативной памяти между процессами;
- установки программ и управления их работой;
- обеспечения надежности и устойчивости работы оборудования и программ.

Чем шире функциональные возможности операционной системы, тем большие требования она предъявляет к техническим ресурсам компьютерной системы, но тем

проще работа с компьютером с точки зрения пользователя. Вопрос ресурсной обеспеченности компьютера, универсальности операционной системы, ее надежности, обеспеченности прикладными программами и драйверами устройств, а также простоты и удобства ее использования, — это сложный вопрос баланса, который может по-разному решаться на каждом рабочем месте в зависимости от конкретных задач.

Программы, которые работают под управлением операционных систем, называются их *приложениями*. В графических операционных системах принцип управления приложениями состоит во взаимодействии активных и пассивных элементов управления. Активный элемент управления — *указатель мыши* (его предоставляет операционная система). Пассивные элементы управления — *графические кнопки, поля, флажки, переключатели, меню, списки* и прочие. Их предоставляют конкретные приложения. В момент взаимодействия активного и пассивного элементов управления пользователь выдает управляющие сигналы с помощью органов управления графического манипулятора.

В неграфических операционных системах управление приложениями ограниченно и осуществляется путем ручного ввода текстовых команд в поле командной строки. Органом управления в данном случае является клавиатура.

Вопросы для самоконтроля

1. Что такое операционная система?
2. Перечислите основные функции операционной системы.
3. Расскажите о видах интерфейса пользователя, применяемых в разных операционных системах.
4. Опишите организацию хранения файлов на дисках компьютера.
5. Перечислите функции операционной системы по обслуживанию файловой структуры.
6. Объясните правила, по которым формируются короткое имя файла и длинное имя файла.
7. В чем заключается операция установки приложения?
8. В чем опасность операции удаления приложения?

Глава 5. Основы работы с операционной системой Windows XP

В предыдущей главе мы рассмотрели функции ряда операционных систем и требования к ним. Надо сказать, что многие из этих требований являются противоречивыми. Например, соотношение требований безотказности и совместимости с приложениями иных систем — это вопрос баланса. Соотношение требований безопасности и простоты обеспечения сетевых функций — это тоже вопрос баланса. На каждом конкретном рабочем месте эти вопросы решаются индивидуально.

В этом смысле сегодня особое место занимает операционная система *Windows XP*. Она обладает наибольшей универсальностью, имеет самое широкое распространение и, соответственно, получает особую поддержку со стороны производителей аппаратного и программного обеспечения. Для компьютера, работающего в этой системе, наиболее просто подобрать прикладные программы и драйверы устройств.

Почти все, что здесь сказано об операционной системе *Windows XP*, можно отнести и к другим операционным системам семейства *Windows*. В том, что касается приемов и методов работы, они в значительной степени совпадают.

5.1. Основные объекты и приемы управления Windows

Windows XP является графической операционной системой для компьютеров платформы *IBM PC*. Ее основные средства управления — графический манипулятор (мышь или иной аналогичный) и клавиатура. Система предназначена для управления автономным компьютером, но также содержит все необходимое для создания небольшой локальной компьютерной сети (*одноранговой сети*) и имеет средства для интеграции компьютера во всемирную сеть (*Интернет*).

Рабочий стол Windows XP

Стартовый экран *Windows XP* представляет собой системный объект, называемый *Рабочим столом*. Практически, экран *Windows XP* является Рабочим столом. Однако существуют видеоадаптеры, позволяющие создать Рабочий стол, размер которого больше, чем видимый размер экрана. Кроме того, *Windows XP* имеет штатные сред-

ства, позволяющие разместить Рабочий стол на нескольких экранах, если к компьютеру подключено несколько мониторов.

Рабочий стол — это *графическая среда*, на которой отображаются *объекты Windows* и *элементы управления Windows*. Все, с чем мы имеем дело, работая с компьютером в данной системе, можно отнести либо к *объектам*, либо к *элементам управления*. В исходном состоянии на Рабочем столе можно наблюдать несколько экранных значков и Панель задач (рис. 5.1). Значки — это графическое представление *объектов Windows*, а Панель задач — один из основных *элементов управления*.

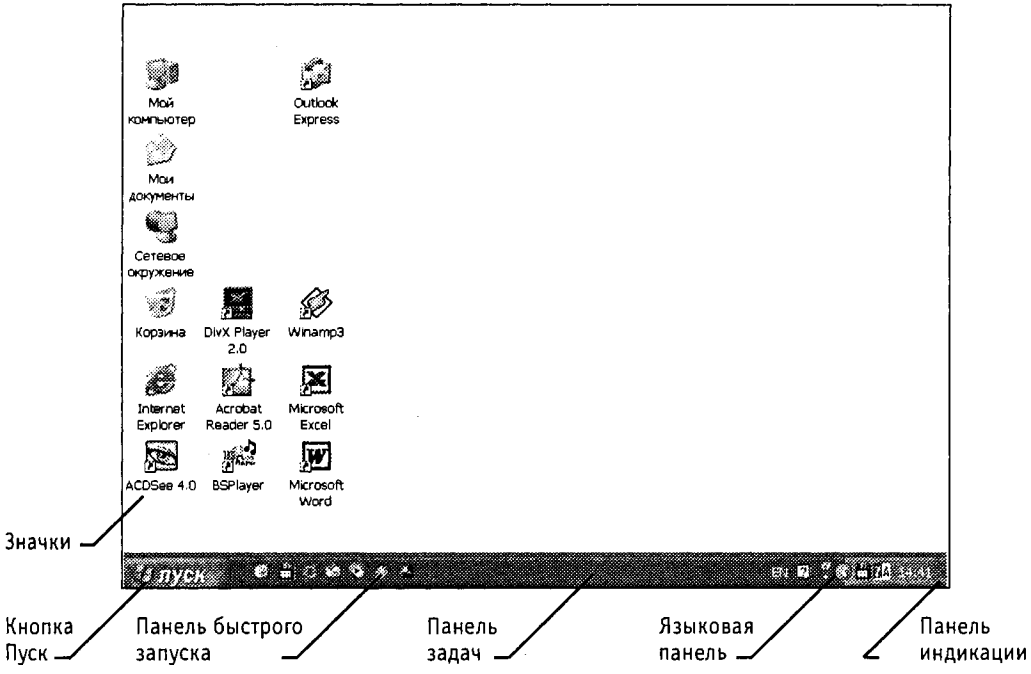


Рис. 5.1. Рабочий стол Windows XP

Управление Windows XP

В *Windows XP* большую часть команд можно выполнять с помощью мыши. С мышью связан активный элемент управления — *указатель мыши*. При перемещении мыши по плоской поверхности указатель перемещается по Рабочему столу, и его можно *позиционировать* на значках объектов или на пассивных элементах управления приложений.

Основными приемами управления с помощью мыши являются:

- *щелчок* — быстрое нажатие и отпускание левой кнопки мыши;
- *двойной щелчок* — два щелчка, выполненные с малым интервалом времени между ними;
- *щелчок правой кнопкой* — то же, что и *щелчок*, но с использованием правой кнопки;

- *перетаскивание* (*drag-and-drop*) — выполняется путем перемещения мыши при нажатой левой кнопке (обычно сопровождается перемещением экранного объекта, на котором установлен указатель);
- *протягивание мыши* (*click-and-drag*) — выполняется, как и *перетаскивание*, но при этом происходит не перемещение экранного объекта, а изменение его формы;
- *специальное перетаскивание* — выполняется, как и *перетаскивание*, но при нажатой правой кнопке мыши, а не левой;
- *зависание* — наведение указателя мыши на значок объекта или на элемент управления и задержка его на некоторое время (при этом обычно на экране появляется *всплывающая подсказка*, кратко характеризующая свойства объекта).

Значки и ярлыки объектов

Создание ярлыков объектов — это одна из функций приема специального перетаскивания, но нам надо пояснить, что же такое *ярлык*. Рассмотрим это понятие на примере Корзины.

Корзина — специальный объект *Windows*, выполняющий функции *контейнера*. Она служит для временного хранения удаляемых объектов. Если какой-то документ или программа стали не нужны, их можно удалить, но при этом они не удаляются безвозвратно, а откладываются в Корзину, из которой их впоследствии можно восстановить.

Откройте окно Мой Компьютер и попробуйте перетащить в него значок Корзины. Это не получится, поскольку Корзина — *реквизитный значок* Рабочего стола. Невозможность перетаскивания отображается специальным указателем мыши.

Теперь попробуйте перетащить значок Корзины в окно Мои документы. Обратите внимание на то, что возле указателя мыши появляется небольшая стрелочка, которая показывает, что при отпускании кнопки мыши будет создан *ярлык* — копия значка Корзина со стрелкой в левом нижнем углу. Ярлыком можно пользоваться точно так же, как обычно пользуются значками.

Значок является *графическим представлением объекта*. То, что мы делаем со значком, мы на самом деле делаем с объектом. Например, удаление значка приводит к удалению объекта; копирование значка приводит к копированию объекта и т. д. Ярлык же является только *указателем* на объект. Удаление ярлыка приводит к удалению указателя, но не объекта; копирование ярлыка приводит к копированию указателя, но не объекта.

Для пользователя приемы работы с ярлыками ничем не отличается от приемов работы со значками. Точно так же можно запускать программы двойным щелчком на их ярлыках, так же можно и открывать документы. Зато ярлыки позволяют экономить место на жестком диске.

Если объект (например, файл с текстовым документом) имеет большой размер, то его многократное копирование в различные окна папок привело бы фактически к появлению новых объектов (копий файла). При этом многократно увеличился бы

расход рабочего пространства на жестком диске, а у пользователя появились бы сложнейшие заботы по синхронизации содержимого этих копий (при редактировании одной копии ее изменения без специальных мер никак не отразятся на содержимом других копий).

С другой стороны, ярлык является лишь указателем, он занимает ничтожно мало места, и его размножение позволяет обеспечить удобный доступ к связанному с ним объекту из разных мест операционной системы. При этом расход рабочего пространства на жестком диске ничтожен, и нет проблем с синхронизацией данных. Из какой бы папки ни открывался документ щелчком на его ярлыке, редактированию всегда подвергается только один связанный с ним объект.

5.2. Файлы и папки Windows

Способ хранения файлов на дисках компьютера называется *файловой системой*. Иерархическая структура, в виде которой операционная система отображает файлы и папки диска, называется *файловой структурой*. Как все дисковые операционные системы, *Windows XP* предоставляет средства для управления этой структурой.

Просмотр папок Windows

Откройте окно Мой компьютер и найдите в нем значок жесткого диска C:. Щелкните на нем дважды, и на экране откроется новое окно, в котором представлены значки объектов, присутствующих на жестком диске. Обратите внимание на значки, представляющие папки, и значки, представляющие файлы. Двойной щелчок на значке любой папки открывает ее окно и позволяет ознакомиться с содержимым. Так можно погружаться вглубь структуры папок до последнего уровня вложения. В соответствующем окне будут представлены только значки файлов.

Окно папки

Окно папки — это *контейнер*, содержимое которого графически отображает содержимое папки. Любую папку *Windows* можно открыть в своем окне. Количество одновременно открытых окон может быть достаточно большим — это зависит от параметров конкретного компьютера. Окна — одни из самых важных объектов *Windows*. Абсолютно все операции, которые мы делаем, работая с компьютером, происходят либо на Рабочем столе, либо в каком-либо окне.

Окна папок — не единственный тип окон в *Windows*. По наличию однородных элементов управления и оформления можно выделить и другие типы окон: *диалоговые окна*, *окна справочной системы* и *рабочие окна приложений*, а внутри окон многих приложений могут существовать отдельные окна документов (если приложение позволяет работать с несколькими документами одновременно).

Если подходить к терминологии с академической строгостью, то за каждым открытым окном скрывается некое работающее приложение (принято говорить *процесс*) и все окна можно было бы назвать окнами приложений (*окнами процессов*), но в учебных целях их лучше все-таки рассматривать порознь.

Структура окна

На рис. 5.2 представлено окно папки \Windows. Такая папка обычно имеется на всех компьютерах, работающих в любой операционной системе семейства *Windows*. Окно папки содержит следующие обязательные элементы.

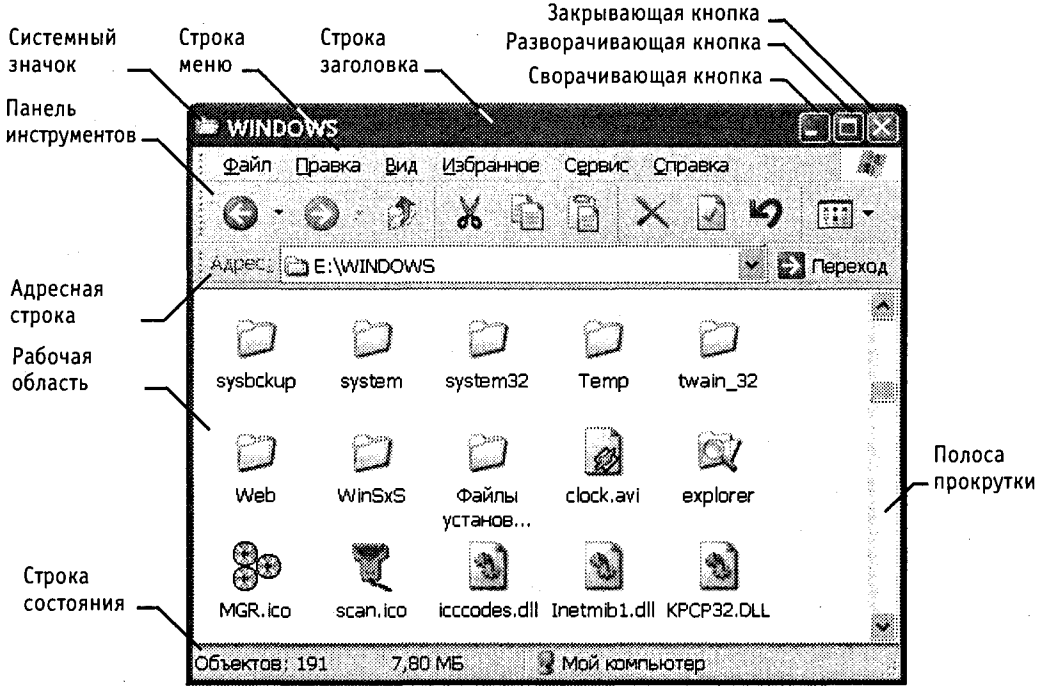


Рис. 5.2. Окно папки Windows

Строка заголовка — в ней написано название папки. За эту строку выполняется перетаскивание папки на Рабочем столе с помощью мыши.

Системный значок. Находится в левом верхнем углу любого окна папки. При щелчке на этом значке открывается меню, называемое *служебным*. Команды, представленные в данном меню, позволяют управлять размером и расположением окна на Рабочем столе — они могут быть полезны, если мышь не работает.

Кнопки управления размером. Эти кнопки дублируют основные команды служебного меню. В операционной системе *Windows XP* исключительно много дублирования. Большинство операций можно выполнить многими различными способами. Каждый пользуется теми приемами, которые ему удобны. Кнопок управления размером три: *закрывающая, сворачивающая, разворачивающая.*

Щелчок на закрывающей кнопке закрывает окно полностью (и прекращает процесс). Щелчок на сворачивающей кнопке приводит к тому, что окно сворачивается до размера кнопки, которая находится на Панели задач (при этом процесс, связанный

с окном, не прекращается). В любой момент окно можно восстановить щелчком на кнопке Панели задач.

Щелчок на разворачивающей кнопке разворачивает окно на полный экран. При этом работать с ним удобно, но доступ к прочим окнам затрудняется. В развернутом окне разворачивающая кнопка сменяется восстанавливающей, с помощью которой можно восстановить исходный размер окна.

Строка меню. Для окон папок строка меню имеет стандартный вид. При щелчке на каждом из пунктов этого меню открывается «ниспадающее» меню, пункты которого позволяют проводить операции с содержимым окна или с окном в целом.

Использование команд, доступных через строку меню, в большинстве случаев не самый эффективный прием работы в *Windows* (есть и более удобные элементы и средства управления), но зато строка меню гарантированно предоставляет *доступ ко всем командам*, которые можно выполнить в данном окне. Это удобно, если неизвестно, где находится нужный элемент управления. Поэтому при изучении работы с новым приложением в первое время принято пользоваться командами строки меню и лишь потом переходить к использованию других средств управления, постепенно повышая эффективность работы.

Панель инструментов. Содержит командные кнопки для выполнения наиболее часто встречающихся операций. В работе удобнее, чем строка меню, но ограничена по количеству команд. В окнах современных приложений панель инструментов часто бывает настраиваемой. Пользователь сам может разместить на ней те командные кнопки, которыми он пользуется чаще всего.

Адресная строка. В ней указан путь доступа к текущей папке, что удобно для ориентации в файловой структуре. Адресная строка позволяет выполнить быстрый переход к другим разделам файловой структуры с помощью раскрывающей кнопки на правом краю строки.

Рабочая область. В ней отображаются значки объектов, хранящихся в папке, причем способом отображения можно управлять (см. ниже). В окнах приложений в рабочей области размещаются окна документов и рабочие панели.

Полосы прокрутки. Если количество объектов слишком велико (или размер окна слишком мал), по правому и нижнему краям рабочей области могут отображаться полосы прокрутки, с помощью которых можно «прокручивать» содержимое папки в рабочей области.

Полоса прокрутки имеет движок и две концевые кнопки. Прокрутку выполняют тремя способами:

- щелчком на одной из концевых кнопок;
- перетаскиванием движка;
- щелчком на полосе прокрутке выше или ниже движка.

Строка состояния. Здесь выводится дополнительная, часто немаловажная информация. Так, например, если среди объектов, представленных в окне, есть скрытые или системные, они могут не отображаться при просмотре, но в строке состояния об их наличии имеется специальная запись.

5.3. Операции с файловой структурой

К основным операциям с файловой структурой относятся:

- навигация по файловой структуре;
- запуск программ и открытие документов;
- создание папок;
- копирование файлов и папок;
- перемещение файлов и папок;
- удаление файлов и папок;
- переименование файлов и папок;
- создание ярлыков.

Система окон Мой компьютер

Все операции с файлами и папками в *Windows XP* можно выполнять несколькими различными способами. Каждый выбирает себе те приемы, которые ему кажутся наиболее удобными. Обычно с приобретением опыта работы на компьютере совокупность используемых приемов меняется.

Простейшие приемы работы с файловой структурой предоставляет иерархическая система окон папок, берущая свое начало от известной нам папки \Мой компьютер. Диски, представленные в окне этой папки, можно открыть, а потом разыскать на них любые нужные папки и файлы. Копирование и перемещение файлов и папок из одной папки в другую можно выполнять путем перетаскивания их значков из окна одной папки в окно другой. Для удаления объектов можно использовать перетаскивание на значок Корзины, а можно пользоваться контекстным меню, которое открывается при щелчке правой кнопкой мыши на объекте. Для создания в папке ярлыка документа или программы можно использовать специальное перетаскивание или команду Создать ▸ Ярлык из контекстного меню.

При таком подходе к операциям с файловой структурой следует иметь в виду несколько замечаний.

1. В *Windows XP* на экране обычно присутствует только одно окно папки. Если в окне папки открыть вложенную папку, то ее окно замещает предыдущее. Это неудобно, если надо выполнять операции перетаскивания между окнами. Чтобы каждая папка открывалась в собственном окне, надо включить следующий переключатель: Пуск ▸ Настройка ▸ Панель управления ▸ Свойства папки ▸ Общие ▸ Открывать каждую папку в отдельном окне.
2. При перетаскивании значков объектов между папками, принадлежащими одному диску, автоматически выполняется *перемещение* объектов. Если нужно выполнить копирование, используют специальное перетаскивание.
3. При перетаскивании значков объектов между папками, принадлежащими разным дискам, автоматически выполняется *копирование* объектов. Если нужно выполнить перемещение, используют специальное перетаскивание.

Программа Проводник

Работа с файловой системой в окнах папок не вполне удобна, но для этой цели есть и более мощное средство — программа Проводник.

Проводник — служебная программа, относящаяся к категории *диспетчеров файлов*. Она предназначена для навигации по файловой структуре компьютера и ее обслуживания. Проводник очень глубоко интегрирован в операционную систему *Windows*. По сути, мы работаем с ним даже тогда, когда его не видим. Если по щелчку правой кнопкой мыши на каком-либо объекте мы получаем контекстное меню, это результат невидимой работы Проводника. Если при перетаскивании объектов из одного окна в другое происходит их копирование или перемещение, это тоже результат заочной деятельности Проводника. Однако с ним можно работать и «очно». Программа запускается командой Пуск ▸ Программы ▸ Стандартные ▸ Проводник.

Окно программы Проводник представлено на рис. 5.3. Как видно из рисунка, по элементам управления это окно очень похоже на окна папок. Основное отличие в том, что окно Проводника имеет не одну рабочую область, а две: левую панель, называемую *панелью папок*, и правую панель, называемую *панелью содержимого*.

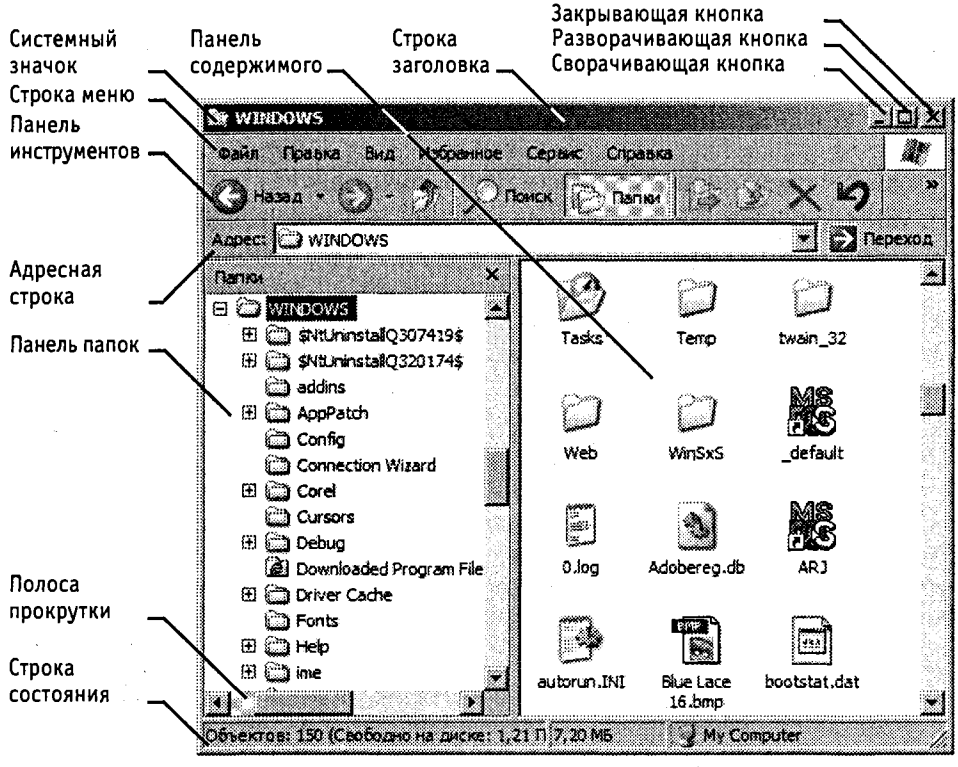


Рис. 5.3. Окно программы Проводник

Навигация по файловой структуре. Цель навигации состоит в обеспечении доступа к нужной папке и ее содержимому. Мы специально не говорим о том, что цель

навигации — это *поиск* нужных файлов и папок, поскольку для этой операции есть специальные средства.

Навигацию по файловой структуре выполняют на левой панели Проводника, на которой показана структура папок. Папки могут быть *развернуты* или *свернуты*, а также *раскрыты* или *закрыты*. Если папка имеет вложенные папки, то на левой панели рядом с папкой отображается *узел*, отмеченный знаком «+». Щелчок на узле разворачивает папку, при этом значок узла меняется на «–». Таким же образом папки и сворачиваются.

Для того чтобы раскрыть папку, надо щелкнуть на ее значке. Содержимое раскрытой папки отображается на правой панели. Одна из папок на левой панели раскрыта всегда. Закрыть папку щелчком на ее значке невозможно — она закроется автоматически при раскрытии любой другой папки.

Запуск программ и открытие документов. Эта операция выполняется двойным щелчком на значке программы или документа на правой панели Проводника. Если нужный объект на правой панели не показан, надо выполнить навигацию на левой панели и найти папку, в которой он находится.

Создание папок. Чтобы создать новую папку, сначала следует на левой панели Проводника раскрыть папку, внутри которой она будет создана. После этого надо перейти на правую панель, щелкнуть правой кнопкой мыши на свободном от значков месте и выбрать в контекстном меню пункт Создать ▸ Папку. На правой панели появится значок папки с названием Новая папка. Название выделено, и в таком состоянии его можно редактировать. После того как папка будет создана, она войдет в состав файловой структуры, отображаемой на левой панели.

Копирование и перемещение файлов и папок. Папку, из которой происходит копирование, называют *источником*. Папку, в которую происходит копирование, называют *приемником*. Копирование выполняют методом перетаскивания значка объекта с правой панели Проводника на левую.

Первая задача — найти и раскрыть папку-источник, чтобы на правой панели был виден копируемый объект. Вторая задача — найти на левой панели папку-приемник, но раскрывать ее не надо. Далее объект перетаскивают с правой панели на левую и помещают на значок папки-приемника. Эта операция требует аккуратности, поскольку попасть одним значком точно на другой не всегда просто. Для контроля точности попадания надо следить за названием папки-приемника. В тот момент, когда наведение выполнено правильно, подпись под значком меняет цвет, и кнопку мыши можно отпускать.

Если и папка-источник, и папка-приемник принадлежат одному диску, то при перетаскивании выполняется перемещение, а если разным — то копирование. В тех случаях, когда нужно обратное действие, выполняют специальное перетаскивание при нажатой правой кнопке мыши.

Удаление файлов и папок. Работа начинается с навигации. На левой панели открывают папку, содержащую удаляемый объект, а на правой панели выделяют нужный объект (или группу объектов).

Удаление можно выполнять несколькими способами. Классический способ — с помощью команды Файл ▸ Удалить из строки меню (если ни один объект не выделен, эта команда не активируется). Более удобный способ — использовать командную кнопку на панели инструментов. Еще более удобно воспользоваться контекстным меню. Щелкните правой кнопкой мыши на удаляемом объекте и выберите в контекстном меню пункт Удалить. Однако самый удобный способ удаления выделенного объекта состоит в использовании клавиши DELETE клавиатуры.

Использование манипуляторов, таких, как мышь, — это важное достоинство графических операционных систем. Однако профессионалами давно отмечено, что наивысшая производительность труда и минимальное утомление при работе достигаются при максимальном использовании клавиатуры. Для команд, представленных в строке меню, часто приводятся клавиатурные комбинации, которыми эти команды можно выполнить. Обращайте на них внимание, запоминайте их и старайтесь постепенно переходить к их использованию. Это один из приемов закрепления навыков профессиональной работы с компьютером.

Создание ярлыков объектов. Ярлыки объектов можно создавать двумя способами: методом специального перетаскивания (вручную) или с помощью специальной программы-мастера (автоматически). С приемом специального перетаскивания мы уже знакомы. Объект выбирается на правой панели Проводника и перетаскивается при нажатой правой кнопке мыши на значок нужной папки на левой панели. В момент отпускания кнопки на экране появляется меню, в котором надо выбрать пункт Создать ярлык.

Второй способ (с использованием мастера) менее нагляден, но во многих случаях более удобен. *Мастерами* в системе *Windows* называют специальные программы, работающие в режиме диалога с пользователем. Диалог строится по принципу «запрос — ответ». Если на все запросы от программы даны корректные ответы, программа автоматически выполнит черновую работу.

1. Для того чтобы запустить Мастер создания ярлыка, надо щелкнуть правой кнопкой мыши в окне той папки, в которой создается ярлык объекта.
2. В открывшемся контекстном меню следует выбрать пункт Создать ▸ Ярлык — произойдет запуск мастера.
3. В диалоговом окне мастера имеется командная строка, в поле которой следует ввести путь доступа к объекту, для которого создается ярлык, например \Windows\System32\Calc.exe — путь доступа к стандартной программе Калькулятор. Разумеется, пользователь не может помнить пути доступа ко всем нужным объектам, поэтому ввод адреса автоматизирован. Для этого служит командная кнопка Обзор.
4. При щелчке на кнопке Обзор открывается диалоговое окно Обзор папок. Это стандартное средство для установления пути доступа к объекту.

   Нужную папку и файл разыскивают примерно так же, как на левой панели программы Проводник. Выбирают диск, на котором расположен искомый файл (в нашем случае это диск C:), затем разворачивают все вышележащие папки. Список файлов отображается в этом окне ниже имени соответствующей папки.

Разыскав нужный объект, его выделяют и щелкают на кнопке OK. Путь доступа к объекту автоматически заносится в командную строку мастера создания ярлыка.

5. Переход к очередному диалоговому окну мастера выполняют щелчком на командной кнопке Далее.
6. В очередном окне мастера вводят название ярлыка, например: Калькулятор. Если это последнее окно мастера, то кнопка Далее сменяется кнопкой Готово. Щелчок на этой кнопке приводит к выполнению заданной операции.

Программа Калькулятор является системной, и ее значок операционной системе хорошо известен. Поэтому Мастер создания ярлыка не задает ни одного вопроса по выбору значка и использует для ярлыка стандартный значок Калькулятора. Если создается ярлык для объекта, неизвестного системе, то мастер продолжает свою работу и предлагает выбрать какой-либо значок из коллекции значков, имеющихся в составе системы.

Приемы повышения эффективности в работе с файловой структурой

Приемы, которые здесь описаны, являются общесистемными. Они относятся не только к Проводнику, но и ко всем окнам папок и большинству окон приложений.

Использование буфера обмена для работы с объектами. Система *Windows* создает и обслуживает на компьютере невидимую для пользователя область памяти, называемую *буфером обмена*. Этой областью можно и нужно уметь пользоваться. В любой момент времени в ней можно хранить только один объект.

Принцип работы с буфером обмена очень прост:

1. Открываем папку-источник. Выделяем щелчком нужный объект.
2. *Копируем* или *забираем* объект в буфер. В первом случае объект остается в папке-источнике и может быть размножен. Во втором случае он удаляется из папки-источника, но может некоторое время храниться в буфере. Последняя операция называется также *вырезанием* объекта.
3. Открываем папку-приемник и помещаем в нее объект из буфера обмена.

Три указанные операции (Копировать, Вырезать и Вставить) можно выполнять разными способами. Классический прием состоит в использовании пункта Правка в строке меню, но более удобно пользоваться командными кнопками панели инструментов:

 — Копировать;

 — Вырезать;

 — Вставить.

Самый же эффективный способ работы с буфером обмена состоит в использовании комбинаций клавиш клавиатуры:

CTRL+C — копировать в буфер;

CTRL+X — вырезать в буфер;

CTRL+V — вставить из буфера.

Эти приемы работают во всех приложениях *Windows*, и их стоит запомнить. Через буфер обмена можно переносить фрагменты текстов из одного документа в другой, можно переносить иллюстрации, звукозаписи, видеофрагменты, файлы, папки и вообще любые объекты. Буфер обмена — мощное средство для работы с приложениями и документами в *Windows*.

В буфере обмена всегда может находиться только один объект. При попытке поместить туда другой объект, предыдущий объект перестает существовать. Поэтому буфер обмена не используют для длительного хранения чего-либо. Поместив объект в буфер, немедленно выполняют вставку из буфера в нужное место.

В общем случае буфер обмена невидим для пользователя, и обычно необходимость просмотра его содержимого не возникает. Однако, если она все-таки возникнет, можно воспользоваться специальной служебной программой Папка обмена, которая входит в состав операционной системы и запускается командой Пуск ▸ Программы ▸ Стандартные ▸ Служебные ▸ Буфер обмена. Если на каком-то конкретном компьютере этой программы нет, это означает, что при установке операционной системы ее *компонент* не был установлен. Его можно доустановить.

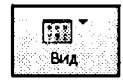
Групповое выделение объектов. Для многих операций (удаление, копирование, перемещение и т. п.) требуется выделить не один объект, а несколько. До сих пор мы использовали для выделения щелчок мыши, но он позволяет выделить только один объект. Для группового выделения при щелчке надо держать нажатой клавишу SHIFT или CTRL.

Если при щелчке держать нажатой клавишу CTRL, то выделение нового объекта не снимает выделение с объектов, выделенных ранее. Так можно выделить любую произвольную группу. Выделение при нажатой клавише CTRL действует как переключатель, то есть повторный щелчок на выделенном объекте снимает выделение.

Если выделяемые объекты расположены подряд, то можно воспользоваться клавишей SHIFT. В этом случае при нажатой клавише щелкают на первом выделяемом объекте группы и на последнем. Все промежуточные объекты выделяются автоматически. Для того чтобы использовать этот прием группового выделения, иногда бывает полезно предварительно упорядочить (отсортировать) объекты, представленные в окне.

Представление объектов. В системе *Windows* можно управлять тем, как представляются объекты в окнах папок или на правой панели программы Проводник. Существует четыре типа представления объектов:

- Плитка;
- Значки;
- Список;
- Таблица.



Выбор метода представления выполняют либо с помощью команд строки меню (пункт Вид), либо с помощью командной кнопки Вид на панели инструментов. Командная кнопка Вид действует как переключатель, автоматически изменяющий способ представления объектов в окне. Если же надо самостоятельно выбрать способ представления, то рядом с этой кнопкой есть раскрывающая кнопка, щелчок на которой раскрывает список возможных режимов.

Режим Плитка применяют в тех случаях, когда в папке находится небольшое количество уникальных объектов (например, программных файлов), каждый из которых важен. В этом режиме отображается не только имя и значок файла, но и некоторые другие его характеристики, зависящие от типа файла.

Режим Значки применяют, когда количество объектов в папке велико и в предыдущем режиме в окне помещается слишком мало значков.

Режим Список применяют в тех случаях, когда в окне присутствуют однотипные объекты, имеющие одинаковые значки. В этом случае содержание объекта характеризует не форма значка, а подпись под ним.

Режим Таблица применяют в тех случаях, когда важны дополнительные свойства объектов, такие как размер, дата создания и т. п. Этот режим интересен также тем, что предоставляет особые возможности по упорядочению объектов в окне.

Упорядочение объектов. Под упорядочением понимают прежде всего сортировку. В системе *Windows XP* существует четыре метода сортировки: Имя, Тип, Размер и Изменен. Метод упорядочения выбирают с помощью команды строки меню Вид ▸ Упорядочить значки.

Если используется метод сортировки Имя, объекты в окне располагаются в алфавитном порядке в соответствии с именами связанных с ними файлов. Когда при упорядочении во внимание принимается Тип, объекты располагаются тоже в алфавитном порядке, но в соответствии с расширениями имен связанных с ними файлов. Вариант Размер применяют перед проведением служебных операций. Например, перед очисткой жесткого диска с целью высвобождения рабочего пространства, удобно знать, какие объекты наиболее ресурсоемки.

Пункт Изменен используют при поиске файлов, изменявшихся в последние дни, или, наоборот, при поиске файлов, не изменявшихся очень долго. Есть вероятность, что документы, не востребованные в течение длительного периода, могут оказаться малонужными и их стоит отправить в архив.

Все методы сортировки работают в восходящем порядке. Файлы сортируются по именам от A до Z или от А до Я; по размерам — от 0 до 9; по датам — от ранних до более поздних. Но если объекты в окне отображаются в виде таблицы, то возможно проведение сортировки в нисходящем порядке. Особенность режима таблицы состоит в том, что каждый столбец имеет заголовок. Этот заголовок обладает свойствами командной кнопки. При первом щелчке на заголовке столбца происходит сортировка объектов по данному столбцу в восходящем порядке, при повторном щелчке — в нисходящем порядке.

5.4. Использование Главного меню

Структура Главного меню

Главное меню — один из основных системных элементов управления *Windows XP*. Оно отличается тем, что независимо от того, насколько Рабочий стол перегружен окнами запущенных процессов, доступ к Главному меню удобен всегда — оно открывается щелчком на кнопке Пуск. С помощью Главного меню можно запустить все программы, установленные под управлением операционной системы или зарегистрированные в ней, открыть последние документы, с которыми выполнялась работа, получить доступ ко всем средствам настройки операционной системы, а также доступ к поисковой и справочной системам *Windows XP*.

Главное меню — необходимый элемент управления для завершения работы с операционной системой. В нем имеется пункт Выключить компьютер, использование которого необходимо для корректного завершения работы с системой перед выключением питания.

В структуру Главного меню входят два раздела — *обязательный* и *произвольный*. Произвольный раздел расположен выше разделительной черты. Пункты этого раздела пользователь может создавать по собственному желанию. Иногда эти пункты образуются автоматически при установке некоторых приложений. Структура обязательного раздела Главного меню представлена в таблице 5.1.

5.5. Установка и удаление приложений Windows

В операционной системе *Windows XP* есть несколько способов установки приложений, но основным является метод, основанный на использовании значка Установка и удаление программ в папке Панель управления (Пуск ▸ Настройка ▸ Панель управления). Во всех случаях рекомендуется использовать именно это средство, поскольку прочие методы установки не гарантируют правильной регистрации приложений в реестре операционной системы.

Перед началом установки нового приложения следует закрыть все работающие программы и все открытые документы. В некоторых случаях необходимо закрывать и ряд фоновых процессов (их наличие может отображаться в виде значков панели индикации на правом краю Панели задач).

Особенности спецификации Windows

Приступая к установке приложений, необходимо знать особенности операционной системы, связанные с *совместным использованием ресурсов*, и помнить, что процедура установки непроверенных программных средств относится к категории потенциально опасных.

Принцип совместного использования ресурсов лежит в основе спецификации *Windows*, и в области программного обеспечения он приводит к тому, что разные приложения могут использовать общие программные ресурсы. Так, например, в большинстве приложений *Windows* можно встретить одинаковые элементы оформления и управления (окна, кнопки, раскрывающиеся списки, меню, флажки, пере-

Таблица 5.1. Структура Главного меню Windows XP

Пункт Главного меню	Назначение	Примечание
Программы	Открывает доступ к иерархической структуре, содержащей указатели для запуска приложений, установленных на компьютере. Для удобства пользования указатели объединяются в категории. Если категория имеет значок в виде треугольной стрелки, в ней имеются вложенные категории. Раскрытие вложенных категорий выполняется простым зависанием указателя мыши	Указатели, присутствующие в Главном меню, имеют статус ярлыков, а их категории – статус папок. Соответственно, указатели можно копировать и перемещать между категориями, перетаскивать на Рабочий стол и в окна папок. Это один из простейших способов создания ярлыка для недавно установленной программы.
Избранное	Открывает доступ к некоторым логическим папкам пользователя, в которых он может размещать наиболее часто используемые документы, ярлыки Web-документов и Web-узлов Интернета	Если с одним компьютером работают несколько пользователей, то каждый может иметь свою персональную группу избранных логических папок
Документы	Открывает доступ к ярлыкам последних пятнадцати документов, с которыми данный пользователь работал на компьютере	Физически эти ярлыки хранятся в скрытой папке \Recent
Настройка	Открывает доступ к основным средствам настройки Windows, в частности, к логической папке Панель управления. Служит также для доступа к папке Принтеры, через которую производится установка принтеров и настройка заданий на печать	При активной работе с компьютером приходится настолько часто использовать обращение к папке Панель управления, что целесообразно создать для нее ярлык на Рабочем столе, однако перетаскиванием из Главного меню это сделать не удается. Для создания ярлыка используйте значок Панель управления в окне Мой компьютер
Найти	Открывает доступ к средствам поиска, установленным на компьютере. Основным является средство Файлы и папки, с помощью которого производится поиск объектов в файловой структуре	При установке приложений, имеющих свои собственные средства поиска, может происходить автоматическое размещение дополнительных ярлыков в этой категории
Справка и поддержка	Пункт входа в справочную систему Windows XP	
Выполнить	Этот пункт открывает небольшое окно, имеющее командную строку для запуска приложений	Удобно использовать в тех случаях, когда необходимо в строке запуска приложения указать параметры запуска
Завершение сеанса ...	Если операционной системой зарегистрировано несколько пользователей одного компьютера, этот пункт позволяет завершить работу одного пользователя и передать компьютер другому	
Завершение работы	Корректное средство завершения работы с операционной системой. Открывает диалоговое окно Завершение работы в Windows, содержащее следующие пункты: • Ждущий режим; • Выключение; • Перезагрузка	Если закрыты все окна процессов, завершить работу с Windows можно комбинацией клавиш ALT+F4. Ждущий режим позволяет "заморозить", а затем восстановить состояние компьютера, хотя не все конфигурации оборудования это допускают и не все программы это хорошо переносят

ключатели и многое другое). Одинаковы и приемы управления ими, и методы их использования. С точки зрения приложений это означает, что их многие компоненты обрабатываются одним и тем же программным кодом. Поэтому в *Windows* принято выделять стереотипные программные фрагменты и группировать их в динамические библиотеки, к которым открыт доступ для разных программ (динамические библиотеки имеют расширение имени файла .DLL).

При установке новых приложений вместе с ними устанавливаются только те программные ресурсы, которые нужны для работы данного приложения, но отсутствуют на данном компьютере (то есть не зарегистрированы в его операционной системе). Поэтому для установки новых приложений очень важно, чтобы они проходили правильную регистрацию. Несмотря на то что в состав дистрибутивных комплектов большинства современных приложений входят специальные устанавливающие программы (Setup.exe), полагаться на то, что они правильно выполнят регистрацию, в общем случае не следует. Установку программ следует выполнять стандартными средствами. Этим обеспечивается надежная работа ранее установленных приложений и закладывается основа для корректной установки последующих приложений.

Стандартное средство установки приложений

Стандартное средство установки (и удаления) приложений *Windows* запускают командой Пуск ▸ Настройка ▸ Панель управления ▸ Установка и удаление программ. После двойного щелчка на указанном значке открывается диалоговое окно Свойства: Установка и удаление программ. Для установки произвольного программного обеспечения надо щелкнуть на значке Установка программ в левой части окна.

Установка приложения начинается с щелчка на кнопке CD или дискета. После этого запускается вспомогательная программа-мастер Установка программ с дискеты или компакт-диска. После щелчка на кнопке Далее мастер пытается автоматически запустить программу установки, найденную на съемном носителе.

Если ему это не удается, можно найти местоположение программы Setup.exe, которая входит в дистрибутивный комплект устанавливаемого приложения, с помощью кнопки Обзор. После этого надо щелкнуть на кнопке Готово.

После установки приложения нередко требуется перезагрузить компьютер. В системе *Windows XP* необходимость перезагрузки возникает реже, чем в предыдущих версиях операционных систем семейства *Windows*, но тоже может потребоваться. Это одна из причин, по которой до начала установки закрывают все открытые приложения и документы.

Необходимость перезагрузки связана с особенностью операционной системы. Некоторые операции выполняются удобно и безопасно только в момент завершения работы системы или на начальном этапе ее загрузки, когда большинство модулей системы еще не активированы.

Удаление приложений Windows

Удаление ранее установленных приложений *Windows* производится средствами того же диалогового окна Установка и удаление программ. Открыв его, следует щелк-

нуть на значке Изменение или удаление программ в левой части окна. Далее надо выбрать удаляемый объект. В зависимости от типа программы вы увидите две отдельные кнопки Изменить и Удалить или общую кнопку Заменить/Удалить. Щелчок на соответствующей кнопке запускает автоматическое средство удаления программы.

Удаление редко бывает полным. Скорее всего, какие-то компоненты останутся. Чаще всего остаются некоторые папки (как правило, пустые). Компоненты, не удаленные автоматически, следует удалить вручную. Рекомендуется удалять их в Корзину и наблюдать за компьютером в течение нескольких дней. Если после этого работоспособность прочих программ не нарушается, эти компоненты можно удалить и из Корзины.

5.6. Установка оборудования

В общем случае оборудование подключается к компьютеру дважды: аппаратно и программно. Под *аппаратным подключением* понимают физическое соединение с компьютером либо с помощью гнезд на материнской плате, либо с помощью внешних разъемов стандартных портов на задней стенке системного блока. Бывает и смешанное подключение, когда *интерфейсная плата* нового устройства вставляется в слот материнской платы и при этом создается новый (нестандартный) порт, разъем которого выходит на заднюю стенку. Таким способом подключают, как правило, устройства, требующие высокой скорости передачи данных, например сканеры или сетевые устройства.

Под *программным подключением* понимают установку программы-драйвера, являющейся посредником между операционной системой и устройством. При установке драйвера происходит выделение операционной системой части ресурсов новому устройству, а также регистрация устройства и его драйвера в реестре операционной системы.

Однако в общем правиле есть и исключения. Такие «стандартные» устройства, как жесткие диски, дисководы гибких дисков и клавиатура, не требуют драйверов, поскольку сведения о том, как с ними работать, уже имеются в базовой системе ввода-вывода (*BIOS*). Они должны распознаваться и работать еще до загрузки операционной системы. То же относится и к монитору, и к видеоадаптеру, но без драйверов они распознаются только как простейшие стандартные модели. Для того чтобы использовать все функциональные возможности конкретной модели, драйвер установить необходимо.

Несколько менее «стандартными» устройствами считаются мышь и дисковод *CD-ROM*. Они не всегда распознаются средствами *BIOS*, но после загрузки операционной системы *Windows XP* уже считаются стандартными устройствами и обслуживаются драйверами, имеющимися в ее составе; однако если речь идет о необычных моделях, особый драйвер для них может потребоваться.

Абсолютное большинство прочих устройств требуют наличия программного драйвера. При продаже аппаратного обеспечения общепринято прикладывать к устройству программные драйверы на компакт-диске. В отсутствие такой возможности

можно воспользоваться библиотекой драйверов, входящей в состав операционной системы. Если библиотека не поддерживает конкретную модель устройства, необходимый драйвер можно получить в Интернете на сервере фирмы, изготовившей оборудование, или на сервере компании *Microsoft*, где имеется коллекция драйверов устройств для операционных систем, выпускаемых этой компанией. Даже для старых и надежно работающих устройств рекомендуется периодически (два раза в год) посещать сервер изготовителя и получать обновленную версию драйвера. Своевременное обновление драйверов устройств повышает эффективность работы оборудования, улучшает совместимость с программным обеспечением и повышает общую надежность системы.

Средства программной установки оборудования

Базовое программное средство установки оборудования запускается двойным щелчком на значке Установка оборудования в окне папки Панель управления. С его помощью можно установить большую часть оборудования, хотя в общем правиле есть исключения.

Драйвер монитора можно установить в диалоговом окне свойств видеосистемы: Пуск ▸ Настройка ▸ Панель управления ▸ Экран ▸ Параметры ▸ Дополнительно ▸ Монитор ▸ Свойства ▸ Драйвер ▸ Обновить. Там же можно установить или заменить драйвер видеоадаптера: Пуск ▸ Настройка ▸ Панель управления ▸ Экран ▸ Параметры ▸ Дополнительно ▸ Адаптер ▸ Свойства ▸ Драйвер ▸ Обновить.

Специальные средства существуют для установки принтеров: Пуск ▸ Настройка ▸ Принтеры и факсы ▸ Установка принтера, а также для установки модемов Пуск ▸ Настройка ▸ Панель управления ▸ Телефон и модем.

Однако наиболее универсальным средством для большей части оборудования все-таки остается Мастер установки оборудования, который запускается двойным щелчком на значке Установка оборудования в окне папки Панель управления.

Порядок установки оборудования

Новое оборудование подключается при выключенном питании компьютера. Если устройство является самоустанавливающимся (соответствует спецификации *plug-and-play*), то после включения питания его наличие выявляется автоматически, и после сообщения Обнаружено неизвестное устройство операционная система приступает к подбору драйвера для него. В этот момент может потребоваться вставить дистрибутивный диск с операционной системой в дисковод *CD-ROM* или использовать компакт-диск с драйвером, полученным вместе с устройством. Иногда необходимы оба диска.

Если устройство не было опознано при запуске, надо воспользоваться Мастером установки оборудования. Мастер запускается командой Пуск ▸ Настройка ▸ Установка оборудования. На первом этапе он разыскивает устройства, соответствующие спецификации *plug-and-play*, и выдает список обнаруженных устройств. Если нужное устройство не входит в список, надо выбрать пункт Добавление нового устройства и щелкнуть на кнопке Далее. Мастер выполнит более тщательный поиск. Если нужное устройство вновь не удалось отыскать, остается возможность указать его

тип самостоятельно. После этого откроется диалоговое окно, в котором можно выбрать производителя и конкретную модель. При наличии нужной модели драйвер можно установить из базы данных *Windows* или с компакт-диска. Если абсолютного совпадения по модели достичь не удается, возможна только установка драйвера с диска, что выполняется после щелчка на кнопке Установить с диска.

По окончании процесса установки оборудования компьютер следует перезагрузить и выполнить проверку на наличие конфликтов. Для проверки наличия конфликтов используют значок Система в окне папки Панель управления или пункт Свойства контекстного меню значка Мой компьютер.

И в том и в другом случае открывается диалоговое окно Свойства: Система. На вкладке Оборудование необходимо щелкнуть на кнопке Диспетчер устройств. В окне Диспетчер устройств отображается список установленных устройств. Нераспознанные устройства в списке обозначены знаком «?», а конфликтующие — знаком «!». Простейший способ устранения конфликтов — удалить конфликтующие устройства с помощью кнопки Удалить и заново провести распознавание оборудования и установку драйверов обоих устройств. Во многих случаях это автоматически снимает проблемы. Более сложная технология устранения конфликтов предполагает назначение аппаратных ресурсов (номера прерывания, адреса порта, адреса канала прямого доступа к памяти) каждому из конфликтующих устройств вручную командой Свойства ▸ Ресурсы.

Практическое занятие

Упражнение 3.1. Отработка приемов управления с помощью мыши

15 мин

1. **Зависание.** Слева на Панели задач имеется кнопка Пуск. Это элемент управления *Windows*, называемый *командной кнопкой*. Наведите на нее указатель мыши и задержите на некоторое время — появится *всплывающая подсказка*: Начните работу с нажатия этой кнопки.

   Справа на Панели задач расположена *панель индикации*. На этой панели, в частности, расположен индикатор *системных часов*. Наведите на него указатель мыши и задержите на некоторое время — появится всплывающая подсказка с показаниями *системного календаря*.

2. **Щелчок.** Наведите указатель мыши на кнопку Пуск и щелкните левой кнопкой — над ней откроется *Главное меню Windows*. Меню — это один из элементов управления, представляющий собой список возможных команд. Команды, представленные в меню, выполняются щелчком на соответствующем пункте. Все команды, связанные с элементами управления, выполняются одним обычным щелчком.

   Однако у щелчка есть и другое назначение. Его применяют также для *выделения* объектов. Разыщите на Рабочем столе значок Мой компьютер и щелкните на нем. Значок и подпись под ним изменят цвет. Это произошло выделение объекта. Объекты выделяют, чтобы подготовить их к дальнейшим операциям.

Щелкните на другом объекте, например на значке Корзина. Выделение значка Мой компьютер снимется, а вместо него выделится значок Корзина. Если нужно снять выделение со всех объектов, для этого достаточно щелкнуть на свободном от объектов месте Рабочего стола.

3. **Двойной щелчок.** Двойной щелчок применяют для *использования* объектов. Например, двойной щелчок на значке, связанном с приложением, приводит к запуску этого приложения, а двойной щелчок на значке документа приводит к открытию данного документа в том приложении, в котором он был создан. При этом происходит одновременно и запуск этого приложения. Относительно документа оно считается *родительским.*

   В системе *Windows XP* с одним и тем же объектом можно выполнить много разных действий. Например, файл с музыкальной записью можно воспроизвести (причем в разных приложениях), его можно отредактировать, можно скопировать на другой носитель или удалить. Сколько бы действий ни было возможно с объектом, всегда существует одно *основное действие.* Оно и выполняется двойным щелчком.

   Выполните двойной щелчок на значке Мой компьютер, и на экране откроется одноименное окно Мой компьютер, в котором можно увидеть значки дисков, подключенных к компьютеру, значок Панели управления и другие значки.

   Если нужно закрыть окно, надо щелкнуть один раз на *закрывающей кнопке*, которая находится в правом верхнем углу окна. Закрывающая кнопка — это элемент управления, и для работы с ним достаточно одного щелчка.

4. **Щелчок правой кнопкой.** Щелкните правой кнопкой на значке Мой компьютер, и рядом с ним откроется элемент управления, который называется *контекстным меню.* У каждого объекта *Windows* свое контекстное меню. Состав его пунктов зависит от свойств объекта, на котором произошел щелчок. Для примера сравните содержание контекстного меню объектов Мой компьютер и Корзина, обращая внимание на их различия.

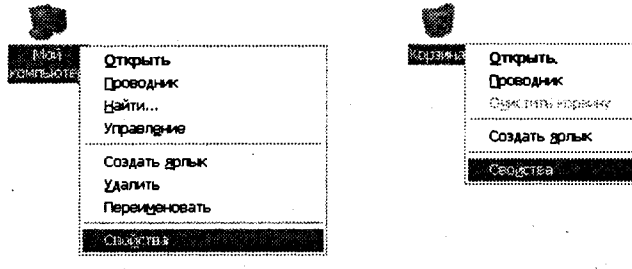


Рис. 5.4. Контекстные меню разных объектов имеют разный состав

Доступ к контекстному меню — основное назначение щелчка правой кнопкой. В работе с объектами *Windows* (особенно с незнакомыми) щелчок правой кнопкой используется очень часто.

Контекстное меню чрезвычайно важно для работы с объектами операционной системы. Выше мы говорили, что двойной щелчок позволяет выполнить только то действие над объектом, которое считается *основным*. В противоположность этому в контекстном меню приведены *все действия*, которые можно выполнить над данным объектом. Более того, во всех контекстных меню любых объектов имеется пункт Свойства. Он позволяет просматривать и изменять свойства объектов, то есть выполнять настройки программ, устройств и самой операционной системы.

5. **Перетаскивание.** Перетаскивание — очень мощный прием для работы с объектами операционной системы. Наведите указатель мыши на значок Мой компьютер. Нажмите левую кнопку и, не отпуская ее, переместите указатель — значок Мой компьютер переместится по поверхности Рабочего стола вместе с ним.

   Откройте окно Мой компьютер. Окно можно перетаскивать с одного места на другое, если «подцепить» его указателем мыши за строку заголовка. Так прием перетаскивания используют для оформления рабочей среды.

6. **Протягивание.** Откройте окно Мой компьютер. Наведите указатель мыши на одну из рамок окна и дождитесь, когда он изменит форму, превратившись в двунаправленную стрелку. После этого нажмите левую кнопку и переместите мышь. Окно изменит размер. Если навести указатель мыши на правый нижний угол окна и выполнить протягивание, то произойдет изменение размера сразу по двум координатам (по вертикали и горизонтали).

   Изменение формы объектов *Windows* — полезное, но не единственное использование протягивания. Нередко этот прием используют для *группового выделения* объектов. Наведите указатель мыши на поверхность Рабочего стола, нажмите кнопку мыши и протяните мышь вправо-вниз — за указателем потянется прямоугольный контур выделения. Все объекты, которые окажутся внутри этого контура, будут выделены одновременно.

7. **Специальное перетаскивание.** Наведите указатель мыши на значок Мой компьютер, нажмите правую кнопку мыши и, не отпуская ее, переместите мышь. Этот прием отличается от обычного перетаскивания только используемой кнопкой, но дает иной результат. При отпускании кнопки не происходит перемещение объекта, а вместо этого открывается так называемое *меню специального перетаскивания*. Содержимое этого меню зависит от перемещаемого объекта. Для большинства объектов в нем четыре пункта (Копировать, Переместить, Создать ярлыки и Отменить). Для таких *уникальных* объектов, как Мой компьютер или Корзина, в этом меню только два пункта: Создать ярлыки и Отменить.

Мы убедились, что, несмотря на то что стандартная мышь имеет только две кнопки, с их помощью можно реализовать весьма разнообразные приемы управления. Мы узнали наиболее характерные особенности этих приемов и их общепринятое назначение. В основе идеологии Windows лежит принцип, согласно которому базовые приемы управления операционной системой должны использоваться и при управлении ее приложениями. Знание общесистемных приемов пригодится в работе с любыми приложениями данной системы.

Упражнение 3.2. Изучение приемов работы с объектами

1. Откройте папку \Мои документы (Пуск ▸ Документы ▸ Мои документы).
2. Щелчком на раскрывающей кнопке разверните окно на полный экран.
3. В строке меню дайте команду Файл ▸ Создать ▸ Папку. Убедитесь в том, что в рабочей области окна появился значок папки с присоединенной надписью Новая папка.
4. Щелкните правой кнопкой мыши на свободной от значков рабочей области окна текущей папки. В открывшемся контекстном меню выберите команду Создать ▸ Папку. Убедитесь в том, что в пределах окна появился значок папки с надписью Новая папка (2).
5. Щелкните правой кнопкой мыши на значке Новая папка. В открывшемся контекстном меню выберите пункт Переименовать. Дайте папке содержательное имя, например Экспериментальная. Аналогично переименуйте папку Новая папка (2). Убедитесь в том, что операционная система не допускает существования в одной папке (\Мои документы) двух объектов с одинаковыми именами. Дайте второй папке имя Мои эксперименты.
6. Восстановите окно папки \Мои документы до нормального размера щелчком на восстанавливающей кнопке.
7. Откройте окно Мой компьютер. В нем откройте окно с содержимым жесткого диска (C:). Пользуясь полосами прокрутки, разыщите в нем папку \Windows и откройте ее двойным щелчком. Ознакомьтесь с текстом предупреждающего сообщения о том, что изменение содержания этой системной папки может быть потенциально опасным. Включите отображение содержимого папки щелчком на ссылке Отображать содержимое этой папки. В открывшемся содержимом разыщите значок папки \Temp и откройте ее (эта папка считается папкой временного хранения данных, и экспериментировать с ее содержимым можно без опасений). Перетаскиванием переместите папку \Экспериментальная из папки \Мои документы в папку C:\Windows\Temp. Специальным перетаскиванием переместите папку \Мои эксперименты в папку C:\Windows\Temp и по окончании перетаскивания выберите пункт Переместить в открывшемся контекстном меню.
8. Откройте окно C:\Windows\Temp. Щелчком выделите значок папки \Экспериментальная. При нажатой клавише CTRL щелчком выделите значок папки \Мои эксперименты. Убедитесь в том, что в рабочей области одновременно выделено два объекта (групповое выделение).
9. Заберите выделенные объекты в буфер обмена комбинацией клавиш CTRL+X. Убедитесь в том, что их значки исчезли в рабочей области папки.
10. Откройте окно папки \Мои документы. Вставьте в него объекты, находящиеся в буфере обмена (CTRL+V).
11. Выделите значки папок \Экспериментальная и \Мои эксперименты в папке \Мои документы. Щелкните правой кнопкой мыши и в открывшемся контекстном

меню выберите пункт Удалить. В открывшемся диалоговом окне подтвердите необходимость удаления объектов. Закройте окно папки \Мои документы.

12. Двойным щелчком на значке откройте окно Корзина. Убедитесь, что в нем находятся значки удаленных папок \Экспериментальная и \Мои эксперименты. Выделите оба значка. Щелкните правой кнопкой мыши и в открывшемся контекстном меню выберите пункт Восстановить. Закройте Корзину.
13. Откройте окно папки \Мои документы. Убедитесь в том, что в нем восстановились значки папок \Экспериментальная и \Мои эксперименты. Выделите оба значка. Удалите их с помощью клавиши DELETE при нажатой клавише SHIFT. В открывшемся диалоговом окне подтвердите необходимость удаления объектов. Закройте окно папки \Мои документы.
14. Откройте окно Корзины. Убедитесь в том, что объекты, удаленные при нажатой клавише SHIFT, не поступили в Корзину. Закройте Корзину.

Мы научились создавать новые папки с помощью строки меню и контекстного меню, научились давать папкам осмысленные имена, познакомились с тремя приемами копирования и перемещения объектов между окнами папок (перетаскиванием, специальным перетаскиванием и с использованием буфера обмена). Мы освоили приемы группового выделения объектов, удаления объектов в Корзину и окончательного удаления, минуя Корзину.

Упражнение 3.3. Работа с файловой структурой в программе Проводник


30 мин

1. Включите персональный компьютер, дождитесь окончания загрузки операционной системы.
2. Запустите программу Проводник с помощью Главного меню (Пуск ▸ Программы ▸ Проводник). Обратите внимание на то, какая папка открыта на левой панели Проводника в момент запуска. Это должна быть папка \Мои документы.
3. На правой панели Проводника создайте новую папку \Экспериментальная.
4. На левой панели разверните папку \Мои документы одним щелчком на значке узла «+». Обратите внимание на то, что *раскрытие* и *разворачивание* папок на левой панели — это разные операции. Убедитесь в том, что на левой панели в папке \Мои документы образовалась вложенная папка \Экспериментальная.
5. Откройте папку \Экспериментальная на левой панели Проводника. На правой панели не должно отображаться никакое содержимое, поскольку эта папка пуста.
6. Создайте на правой панели Проводника новую папку \Мои эксперименты внутри папки \Экспериментальная. На левой панели убедитесь в том, что рядом со значком папки \Экспериментальная образовался узел «+», свидетельствующий о том, что папка имеет вложенные папки. Разверните узел и рассмотрите образовавшуюся структуру на левой панели Проводника.
7. На левой панели Проводника разыщите папку \Windows и разверните ее.

8. На левой панели Проводника внутри папки \Windows разыщите папку для временного хранения объектов — \Temp, но не раскрывайте ее.
9. Методом перетаскивания переместите папку \Экспериментальная с правой панели Проводника на левую — в папку C:\Windows\Temp. Эту операцию надо выполнять аккуратно. Чтобы «попадание» было точным, следите за цветом надписи папки-приемника. При точном наведении надпись меняет цвет — в этот момент можно отпускать кнопку мыши при перетаскивании. Еще труднее правильно «попасть в приемник» при перетаскивании групп выделенных объектов. Метод контроля тот же — по выделению надписи.
10. На левой панели Проводника откройте папку C:\Windows\Temp. На правой панели убедитесь в наличии в ней папки \Экспериментальная.
11. Разыщите на левой панели Корзину и перетащите папку \Экспериментальная на ее значок. Раскройте Корзину и проверьте наличие в ней только что удаленной папки. Закройте окно программы Проводник.

Мы научились выполнять навигацию с помощью левой панели программы Проводник и изучили приемы копирования и перемещения объектов методом перетаскивания между панелями. Те, кто считает, что с левой панелью Проводника работать не очень удобно, могут исполнять все операции, пользуясь только правой панелью. При этом используют следующие свойства Проводника:

- возможность копирования и перемещения объектов через буфер обмена;
- программу Проводник можно запустить несколько раз — соответственно, на Рабочем столе можно иметь несколько правых панелей, между которыми удобно выполняются все операции обмена.

Исследовательская работа

Задание 3.1. Исследование методов запуска программы Проводник

В операционной системе *Windows XP* большинство операций можно выполнить многими разными способами. На примере программы Проводник мы исследуем различные приемы запуска программ.

1. Щелкните правой кнопкой мыши на кнопке Пуск и в открывшемся контекстном меню используйте пункт Проводник. Обратите внимание на то, какая папка открыта на левой панели в момент запуска.
2. Щелкните правой кнопкой мыши на значке Мой Компьютер и в открывшемся контекстном меню используйте пункт Проводник. Обратите внимание на то, какая папка открыта на левой панели в момент запуска.
3. Проверьте контекстные меню всех значков, открытых на Рабочем столе. Установите, для каких объектов контекстное меню имеет средства запуска Проводника, и выясните, какая папка открывается на левой панели в момент запуска.
4. Выполните запуск Проводника через пункт Программы Главного меню.
5. Выполните запуск Проводника через пункт Выполнить Главного меню.

6. Выполните запуск Проводника через ярлык папки \Мои документы (Пуск ▸ Документы ▸ Мои документы ▸ *щелчок правой кнопкой мыши* ▸ Проводник).
7. Выполните запуск Проводника с Рабочего стола (предварительно на Рабочем столе следует создать ярлык Проводника).
8. Выполните запуск Проводника с Панели быстрого запуска (предварительно на этой панели следует создать ярлык Проводника).
9. Заполните отчетную таблицу по образцу:

Метод запуска Проводника	**Используемый элемент управления**	**Папка открытия**
Через контекстное меню кнопки Пуск	Кнопка Пуск	\Главное меню

Глава 7. Стандартные приложения Windows XP

Основное назначение операционных систем — обеспечение взаимодействия человека, оборудования и программ. От операционных систем не требуется наличия средств, предназначенных для исполнения конкретных прикладных задач, — для этого есть прикладное программное обеспечение. Тем не менее, в операционную систему *Windows XP* входит ограниченный набор прикладных программ, с помощью которых можно решать некоторые простейшие повседневные задачи, пока на компьютере не установлены более мощные программные средства. Такие программы, входящие в поставку *Windows*, называют *стандартными приложениями*. В силу особой простоты их принято также рассматривать в качестве учебных. Знание приемов работы со стандартными приложениями позволяет ускорить освоение специализированных программных средств.

7.1. Стандартные прикладные программы

Программа Блокнот

Блокнот — это простейший текстовый редактор, который можно также использовать в качестве удобного *средства просмотра* текстовых файлов (формат .TXT и некоторые другие). Для создания текстовых документов его применяют редко (только для небольших записок), но данную программу удобно использовать для отработки навыков работы с клавиатурой.

Программа запускается командой Пуск ▸ Программы ▸ Стандартные ▸ Блокнот. Пример ее рабочего окна показан на рис. 7.1.

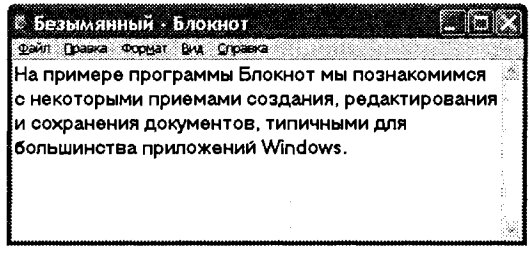


Рис. 7.1. Окно программы Блокнот

На примере программы Блокнот мы познакомимся с некоторыми приемами создания, редактирования и сохранения документов, типичными для большинства приложений *Windows*.

Ввод текста с помощью клавиатуры. Текст вводят с помощью алфавитно-цифровых клавиш. Для ввода прописных букв используют клавишу SHIFT. Если нужно ввести длинный ряд (поток) прописных символов, клавиатуру можно переключить с помощью клавиши CAPS LOCK.

Когда текст достигает правой границы окна, он может автоматически перетекать на новую строку, но может продолжаться далее, пока не будет нажата клавиша ENTER. Чтобы включить (или отключить) режим автоматического перетекания текста, используют команду Формат ▸ Перенос по словам.

Понятие курсора. Место документа, в которое происходит ввод текста (*точка ввода*), отмечается на экране вертикальной чертой, которую называют *курсором*. Не надо путать курсор с указателем мыши — это два разных понятия. Указатель мыши — это активный элемент управления, а курсор — это только маркер, не выходящий за пределы документа.

В прошлом, до появления графических операционных систем, указатель мыши называли курсором, но сегодня эти понятия различают. В редакторе Блокнот нетрудно убедиться в том, что, когда курсор находится в одном месте в тексте документа, указатель мыши можно свободно перемещать по полю документа и даже вне окна программы.

Переключение между русскими и латинскими символами. При наборе текста иногда приходится переключаться между русскими и латинскими символами. Это делается общесистемным способом, то есть метод переключения между символьными наборами не зависит от конкретной программы, а выполняется во всех программах одинаково. Это функция операционной системы.

Для того чтобы узнать, какой комбинацией клавиш на данном компьютере выполняется переключение раскладок клавиатуры, надо посмотреть, как настроены свойства языка (Пуск ▸ Настройка ▸ Панель управления ▸ Язык и региональные стандарты). Выбрав вкладку Языки, щелкните на кнопке Подробнее. Далее надо щелкнуть на кнопке Параметры клавиатуры. Для выбора способа переключения раскладок надо выбрать нужное действие в списке и щелкнуть на кнопке Смена сочетания клавиш. Обычно для переключения между русской и английской раскладками используют комбинацию клавиш CTRL+SHIFT. Если включено отображение языковой панели на Панели задач, то индикатор текущего языка отображается рядом с Панелью индикации. В этом случае для переключения между языками достаточно щелкнуть мышью на данном индикаторе.

Выбор шрифта. Размер и форма символов языка определяются использованным шрифтом. Редактор Блокнот слишком прост для того, чтобы позволить использование разных шрифтов в документе, но выбрать один шрифт, используемый для отображения документа, он позволяет. Это выполняется командой Формат ▸ Шрифт, которая открывает системное диалоговое окно Выбор шрифта, представленное на рис. 7.2.

В списке Шрифт можно выбрать один из возможных шрифтов. Здесь представлены все шрифты, установленные на компьютере. Не все шрифтовые наборы могут иметь

в своем составе символы русского языка, поэтому при выборе шрифта требуется либо предварительное знание, либо свободное экспериментирование.

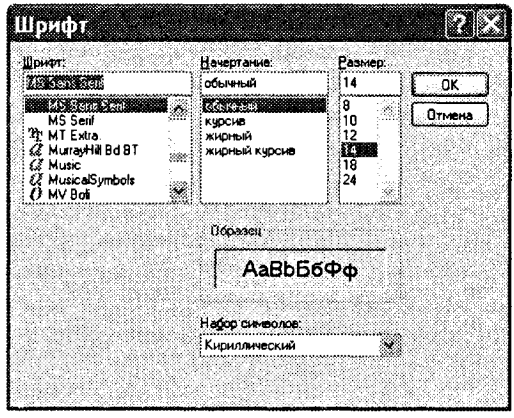


Рис. 7.2. Выбор шрифта в программе Блокнот

В списке Начертание можно задать начертание для избранного шрифта. Обычно используют четыре основных типа начертания: прямое светлое (обычное), *наклонное* (*курсив*), **полужирное** и ***полужирный курсив***. Выбор начертания, как и выбор шрифта, относится только к способу отображения документа (в более мощных текстовых редакторах и процессорах в одном документе можно применять разные шрифты и разные начертания).

В списке Размер выбирают размер шрифта. Размеры шрифтов измеряются в *пунктах*. Пункт — это типографская единица измерения, равная 1/72 дюйма (0,353 мм). Для того чтобы документ хорошо читался на экране, обычно используют шрифт размером 12 пунктов.

Сохранение созданного документа. Созданный документ сохраняют на жестком или гибком магнитном диске в виде нового файла. При сохранении следует указать имя файла. Программа Блокнот не осуществит сохранение, пока имя не задано. Для сохранения нового документа служит команда Файл ▸ Сохранить как. По этой команде открывается диалоговое окно Сохранение, представленное на рис. 7.3.

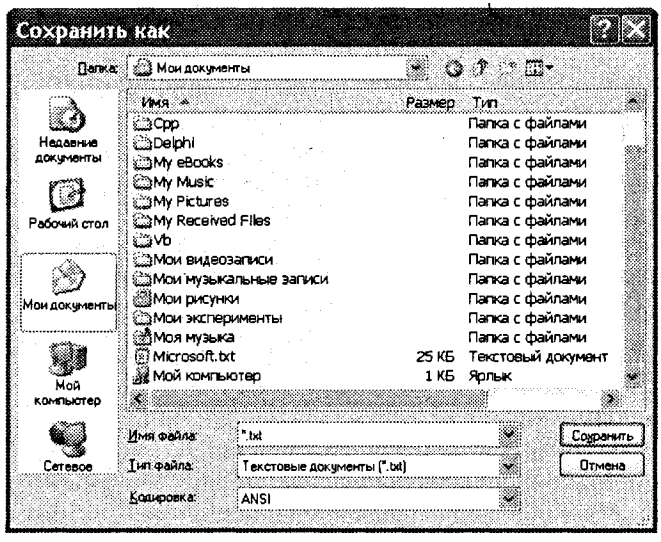


Рис. 7.3. Сохранение файла документа

В этом окне выбирают папку, в которую будет сохраняться файл, и дают ему имя. Приемы сохранения файлов одинаковы для всех приложений *Windows*. Освоив их один раз, далее можно пользоваться ими в любых программах. В качестве папки, в которую редактор Блокнот сохраняет документы *по умолчанию*, служит папка \Мои документы. Большинство приложений *Windows* предлагают по умолчанию использовать для сохранения документов именно эту папку. В ней можно создать несколько папок для раздельного хранения документов, относящихся к разным темам (проектам). Папка \Мои документы удобна еще и тем, что если с одним компьютером работают несколько человек и при запуске операционной системы каждый пользователь проходит регистрацию, то система создает каждому свою особую папку \Мои документы, чтобы документы разных людей не перемешивались между собой.

Если предложенная папка \Мои документы соответствует желанию автора, то остается только ввести имя файла в поле Имя файла и щелкнуть на кнопке Сохранить. Если в этой папке нужно создать новую папку, надо использовать кнопку Создание новой папки и дать новой папке содержательное имя.

Если же для сохранения документа надо использовать произвольную папку, отличную от папки \Мои документы, ее надо разыскать. Поиск по файловой структуре начинается с щелчка на раскрывающей кнопке справа у поля Папка. К некоторым папкам имеется удобный доступ через панель, расположенную в левой части окна. Особого внимания заслуживает значок Недавние документы. С его помощью легко перейти в любую папку, к которой недавно уже обращались.

Приемы редактирования документов. Под *редактированием* понимают изменение уже существующих документов. Редактирование начинают с загрузки (открытия) документа. Для этого служит команда Файл ▸ Открыть. По этой команде на экране появляется стандартное диалоговое окно Открыть. Как и окно Сохранить как, оно одинаково во всех приложениях *Windows*. По умолчанию окно Открыть указывает на папку \Мои документы. Если нужный документ находится в другой папке, ее надо разыскать и раскрыть.

Для редактирования текстовых документов следует научиться управлять курсором. Его перемещают с помощью специальных *клавиш управления курсором*. Для перемещения курсора на экранную страницу вверх или вниз используют клавиши PAGE UP и PAGE DOWN. Для перевода курсора в начало текущей строки используют клавишу HOME, а в конец строки — клавишу END. В большинстве приложений *Windows* работают также комбинации клавиш CTRL+HOME и CTRL+END, переводящие курсор в начало или конец документа, соответственно. Для произвольного размещения курсора используют указатель мыши — курсор устанавливается в место щелчка в рабочей области.

Удаление ошибочных символов выполняют клавишами BACKSPACE или DELETE. Разница между ними состоит в том, что первая удаляет символы, стоящие слева от курсора, а вторая — справа. Для удаления большого блока текста пользоваться клавишами редактирования неудобно. В таких случаях сначала выделяют текстовый блок, а потом нажимают клавишу DELETE (один раз). При этом удаляется весь выделенный блок.

Выделение больших блоков производят методом протягивания мыши. В этом случае для удаления удобно использовать команду Удалить контекстного меню. Существует и прием выделения текстовых фрагментов с помощью клавиатуры. Он выполняется клавишами управления курсором при нажатой клавише SHIFT.

Выделенные фрагменты текста можно не только удалять, но и копировать или перемещать. Эти приемы очень часто применяются при редактировании. Копирование и перемещение происходит через буфер обмена *Windows*. Напомним комбинации клавиш, которые следует запомнить:

CTRL+C — копировать в буфер;

CTRL+X — вырезать в буфер;

CTRL+V — вставить из буфера.

Программа Блокнот не позволяет работать более чем с одним документом, но ее можно запустить два и более раз. В этом случае на экране можно иметь несколько окон программы с разными документами. Поставив такой эксперимент, нетрудно убедиться, что перенос текстовых фрагментов через буфер обмена возможен не только внутри одного окна, но и между окнами.

Сохранение отредактированного документа. Сохранение документа, прошедшего редактирование, отличается от сохранения нового документа хотя бы тем, что файл этого документа уже существует и не надо выбирать папку и давать файлу имя. Для его сохранения достаточно дать команду Файл ▸ Сохранить, и новая копия документа заместит старую. Однако бывают случаи, когда старую копию документа не следует замещать. В этом случае документ сохраняют либо в другую папку, либо под другим именем. В этом случае порядок действий тот же, что и при сохранении нового документа командой Сохранить как.

Средства автоматизации. Программа Блокнот слишком проста, чтобы иметь серьезные средства автоматизации. В более мощных текстовых редакторах и процессорах эти средства надо изучать специально, поскольку от них зависит эффективность работы. В этой же программе единственное средство автоматизации состоит в том, что при нажатии на клавишу F5 в документ автоматически впечатывается текущее время и дата. Это удобно для ведения деловых записей и дневников.

Графический редактор Paint

Графическими называют редакторы, предназначенные для создания и редактирования изображений (рисунков). Программа *Paint* — простейший графический редактор. По своим возможностям она не соответствует современным требованиям, но в силу простоты и доступности остается необходимым компонентом операционной системы. Не разобравшись с принципами управления этой программой, трудно осваивать другие, более мощные средства работы с графикой.

Программа запускается командой Пуск ▸ Программы ▸ Стандартные ▸ Paint.

Основные понятия. Программа *Paint* является редактором *растровой* графики. Это важное замечание, поскольку кроме редакторов растровой графики существуют еще редакторы *векторной* графики. Приемы и методы работы с этими двумя различ-

ными классами программ совершенно различны. В растровой графике мельчайшим элементом изображения является точка, которой на экране соответствует экранная точка (*пиксел*). Мельчайшим элементом векторной графики является линия, описываемая математическим выражением.

Рабочее окно программы *Paint* представлено на рис. 7.4. В состав его элементов управления, кроме строки меню, входят панель инструментов, палитра настройки инструмента и цветовая палитра. Кнопки панели инструментов служат для вызова чертежно-графических инструментов. На палитре настройки можно выбрать параметры инструмента (толщину линии, форму оттиска, метод заполнения фигуры и т. п.). Элементы цветовой палитры служат для выбора основного цвета изображения (щелчком левой кнопки) и фонового цвета (щелчком правой кнопки).

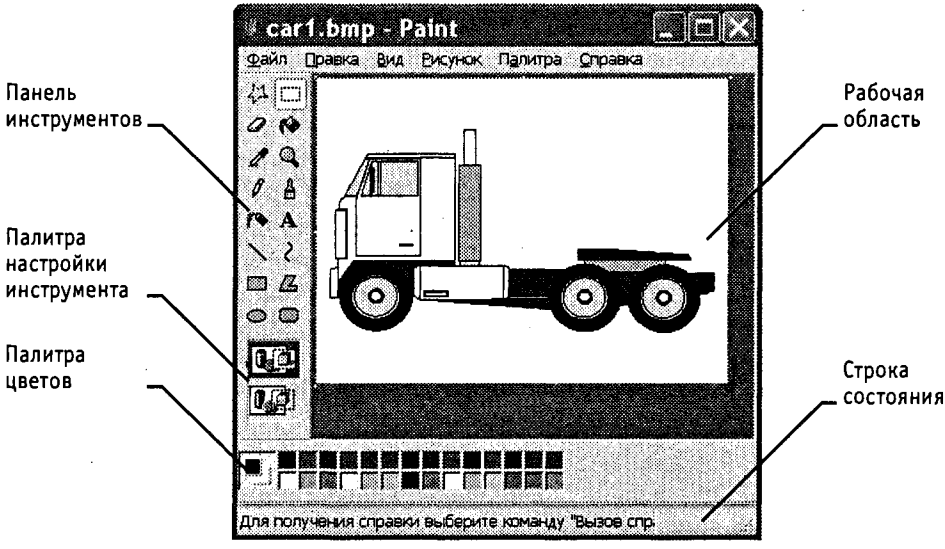


Рис. 7.4. Графический редактор Paint

Задание размера рабочей области. Перед началом работы следует хотя бы приблизительно задать размер будущего рисунка. Размеры задают в полях Ширина и Высота диалогового окна Атрибуты (Рисунок ▸ Атрибуты). До ввода размеров следует выбрать принятую единицу измерения с помощью одного из переключателей:

- Дюймы;
- См (сантиметры);
- Точки (пикселы).

В России не принято задавать размеры документов в дюймах. Размер в сантиметрах задают в тех случаях, когда предполагается вывод работы на печатающее устройство (принтер) или встраивание изображения на страницу с текстовым документом. В тех случаях, когда рисунок предназначен для воспроизведения на экране, в качестве единицы измерения выбирают Точки (пикселы). Так, например, если рисунок

готовится для использования в качестве фона Рабочего стола, его размеры следует принять равными величине экранного разрешения монитора (640×480; 800×600; 1024×768 точек и т. д.).

Основные чертежно-графические инструменты. Все инструменты, кроме Ластика, выполняют рисование основным цветом (выбирается щелчком левой кнопки в палитре красок). Ластик стирает изображение, заменяя его фоновым цветом (выбирается щелчком правой кнопки мыши в палитре красок).

Инструмент Линия предназначен для вычерчивания прямых. Толщину линии выбирают в палитре настройки. Линии вычерчивают методом протягивания мыши. Чтобы линия получилась «строгой» (вертикальной, горизонтальной или наклонной под углом 45°), при ее вычерчивании следует держать нажатой клавишу SHIFT.

Инструмент Карандаш предназначен для рисования произвольных линий. Толщину линии выбирают в палитре настройки.

Инструмент Кривая служит для построения гладких кривых линий. Толщину предварительно выбирают в палитре настройки. Построение производится в три приема. Сначала методом протягивания проводят прямую линию, затем щелчком и протягиванием в стороне от линии задают первый и второй радиусы кривизны. Математически, данная кривая, имеющая два радиуса кривизны и одну точку перегиба, является частным случаем кривой третьего порядка (*кривой Безье*).

Инструмент Кисть можно использовать для свободного рисования произвольных кривых, как Карандаш, но чаще его используют для рисования методом *набивки*. Сначала выбирают форму кисти в палитре настройки, а потом щелчками левой кнопки мыши наносят оттиски на рисунок без протягивания мыши.

Инструмент Распылитель используют как для свободного рисования, так и для рисования методом набивки. Форму пятна выбирают в палитре настройки. При свободном рисовании вид рисунка зависит и от скорости движения указателя мыши.

Инструмент Прямоугольник применяют для рисования прямоугольных фигур. Рисование выполняется протягиванием мыши. В палитре настройки можно выбрать метод заполнения прямоугольника. Возможны три варианта: Без заполнения (рисуется только рамка), Заполнение фоновым цветом и Заполнение основным цветом.

Если при создании прямоугольника держать нажатой клавишу SHIFT, образуется правильная фигура. Для прямоугольника правильной фигурой является квадрат.

Аналогичный инструмент Скругленный прямоугольник действует точно так же, но при этом получается прямоугольник со скругленными углами.

Инструмент Многоугольник предназначен для рисования произвольных многоугольников. Рисование выполняют серией последовательных щелчков с протягиванием. Если конечная точка многоугольника совпадает с начальной, то многоугольник считается замкнутым. Замкнутые фигуры могут автоматически заливаться краской в соответствии с вариантом заполнения, выбранным в палитре настройки.

Инструмент Эллипс служит для изображения эллипсов и окружностей. Окружность — это частный случай «правильного» эллипса. Она получается при рисовании с нажатой клавишей SHIFT.

Инструмент Заливка служит для заполнения замкнутых контуров основным или фоновым цветом. Заполнение основным цветом производится щелчком левой кнопки мыши, а заполнение фоновым цветом — щелчком правой кнопки. Если контур не замкнут, инструмент работает неправильно. В этом случае ошибочное действие надо немедленно отменить командой Правка ▸ Отменить или комбинацией клавиш CTRL+Z.

Комбинацию CTRL+Z следует запомнить. Она отменяет последнее действие в большинстве приложений Windows и является удобным общесистемным приемом.

Инструмент Выбор цветов позволяет точно выбрать основной или дополнительный цвет не из палитры красок, а непосредственно из рисунка. Это важно, когда надо обеспечить тождественность цвета в разных областях изображения. После выбора инструмента наводят указатель на участок рисунка с нужным цветом и щелкают кнопкой мыши. Если произошел щелчок левой кнопкой, текущий цвет становится основным, а если правой — фоновым.

Инструменты выделения областей. Два инструмента предназначены для работы с выделенными областями: Выделение и Выделение произвольной области. Действуют они одинаково, разница лишь в том, что инструмент Выделение формирует не произвольную, а прямоугольную выделенную область. С выделенной областью можно поступать так, как это принято во всех приложениях *Windows*: ее можно удалить клавишей DELETE, скопировать в буфер обмена (CTRL+C), вырезать в буфер обмена (CTRL+X) и вставить из буфера обмена (CTRL+V). Прием копирования и вставки выделенной области применяют для размножения повторяющихся фрагментов.

При размножении выделенных областей возможны два режима вставки: с сохранением фоновой графики или без нее (точки фонового цвета во вставляемой области игнорируются). Переключение режима выполняют в палитре настройки.

Масштабирование изображений. Для точной доводки рисунка иногда необходимо увеличить его масштаб. Максимальное увеличение — восьмикратное. Для изменения масштаба служит команда Вид ▸ Масштаб. То же можно сделать с помощью инструмента Масштаб, в этом случае величину масштаба выбирают в палитре настройки.

В режиме шестикратного или восьмикратного увеличения на рисунок можно наложить вспомогательную сетку (Вид ▸ Масштаб ▸ Показать сетку). Каждая ячейка этой сетки представляет собой одну увеличенную точку изображения. В этом режиме удобно редактировать изображение по отдельным точкам.

Трансформация изображений. *Трансформациями* называют автоматические изменения формы, расположения или размеров графических объектов. В программе *Paint* не слишком много инструментов трансформации, но все-таки они есть. Их можно найти в меню Рисунок.

Команда Рисунок ▸ Отразить/повернуть вызывает диалоговое окно Отражение и поворот, содержащее элементы управления для симметричного отображения рисунка относительно вертикальной или горизонтальной оси симметрии, а также для поворота на фиксированный угол, кратный 90°.

Команда Рисунок ▸ Растянуть/наклонить вызывает диалоговое окно Растяжение и наклон. Его элементы управления позволяют растянуть рисунок по горизонтали и вертикали или наклонить относительно горизонтальной или вертикальной оси. Параметры растяжения задают в процентах, а параметры наклона — в угловых градусах.

Команда Рисунок ▸ Обратить цвета действует как переключатель. При использовании этой команды цвет каждой точки изображения меняется на «противоположный». В данном случае мы назвали «противоположным» тот цвет, который дополняет данный цвет до белого.

Ввод текста. Программа *Paint* — графический редактор и не предназначена для работы с текстом. Поэтому ввод текста в этой программе является исключением, а не правилом. Поскольку редактор относится к растровым, он строит изображение по точкам. Следовательно, текст после ввода станет «рисунком» и будет состоять из достаточно крупных точек растра. Поэтому избегайте использования мелких символов, которые смотрятся неопрятно. Рассматривайте режим работы с текстом в программе *Paint* только как средство для создания кратких и крупных заголовков.

Для ввода текста используют инструмент Надпись. Выбрав инструмент, щелкните на рисунке примерно там, где надпись должна начинаться, — на рисунке откроется поле ввода. В это поле вводится текст с клавиатуры. О типе шрифта, его размере и начертании заботиться пока не надо — главное набрать текст без ошибок, а остальное все можно изменить позже. Размер поля ввода изменяют путем перетаскивания *маркеров области ввода* — небольших прямоугольных узлов, расположенных по сторонам и углам области ввода.

Закончив ввод, вызывают панель атрибутов текста (Вид ▸ Панель атрибутов текста). Элементами управления этой панели можно выбрать форму шрифта, его начертание и размер.

О том, чего нет в редакторе Paint. В работе с вычислительной техникой безусловно важно знать возможности программных средств и приемы их использования. Но не менее важно знать и ограничения программных средств. Это позволяет двигаться вперед, осваивать новые продукты и приемы. Как мы уже говорили, графический редактор *Paint* — простейший, поэтому в нем нет многого из того, что есть в других современных графических редакторах.

1. *Автоматическое выделение областей.* Мы видели, как в редакторе *Paint* выполняется выделение прямоугольных и произвольных областей. В более мощных редакторах есть средства для автоматического выделения. Например, они могут работать по принципу подобия цвета: все элементы изображения, имеющие цвет, близкий к заданному, выделяются автоматически. Это позволяет точно выделять очень сложные контуры (операция называется *обтравкой контура*).
2. *Специальные методы заливки.* В программе *Paint* работает только простейшая заливка одним цветом. В более мощных программах обычно имеются также

средства выполнения градиентной заливки (*градиентной заливкой* или *растяжкой* называют заливку с плавным переходом от одного цвета к другому) и множество вариантов текстурной заливки (при текстурной заливке замкнутый контур заполняется узором или рисунком, имитирующим фактуру материала, например дерева, металла, ткани и т. п.).

3. *Применение фильтров.* Фильтрами называют специальные методы автоматической обработки изображений или выделенного фрагмента. Например, с помощью фильтров можно управлять яркостью или контрастностью изображения. Существуют искажающие фильтры, например имитирующие просмотр рисунка через стекло, смоченное водой и т. п. В редакторе *Paint* нет фильтров, но в других графических редакторах могут насчитываться десятки и сотни фильтров для создания специальных эффектов.
4. *Использование слоев.* В редакторе *Paint* мы работаем только с одним слоем изображения. Это не слишком удобно. В тех программах, где предусмотрена возможность создания слоев, можно разные объекты располагать на разных слоях, а потом объединять их. Слои могут быть прозрачными или полупрозрачными. С помощью слоев создают эффекты туманной дымки на фотографиях или эффекты, когда объект как бы парит над поверхностью фона и отбрасывает тень на поверхность (особенно часто этот эффект применяют для создания «парящих надписей»).
5. *Трансформации.* На примере программы *Paint* мы познакомились с простейшими трансформациями изображения: *наклоном* и *растягиванием*. Существуют и более сложные трансформации, например *скручивание*. Особенно много трансформаций существует для преобразования трехмерных объектов.
6. *Использование подключаемых расширений* (*plug-ins*). Ни одна графическая программа не может содержать все мыслимые инструменты, фильтры, средства заливки и трансформации. Поэтому современные графические редакторы позволяют подключать дополнительные компоненты, называемые *расширениями*. Возможность модернизации программного обеспечения путем подключения дополнительных блоков, сделанных посторонними программистами, называют *принципом открытой программной архитектуры*. В последние годы этот принцип получил широкое развитие. Программы, обладающие открытой архитектурой, развиваются и совершенствуются быстрее, чем программы с *закрытой архитектурой*, модернизация которых посторонними лицами не предусмотрена.

Текстовый процессор WordPad

Текстовые процессоры, как и текстовые редакторы, служат для создания, редактирования и просмотра текстовых документов. Однако они выполняют еще одну важную функцию — *форматирование* документов. Под форматированием понимают оформление документов применением нескольких шрифтовых наборов, использованием методов выравнивания текста, встраиванием в текстовый документ объектов иной природы, например рисунков, а также контролем за обтеканием графики текстом.

В стандартную поставку *Windows XP* входит текстовый процессор *WordPad*, который фактически является «облегченной» версией гораздо более мощной программы

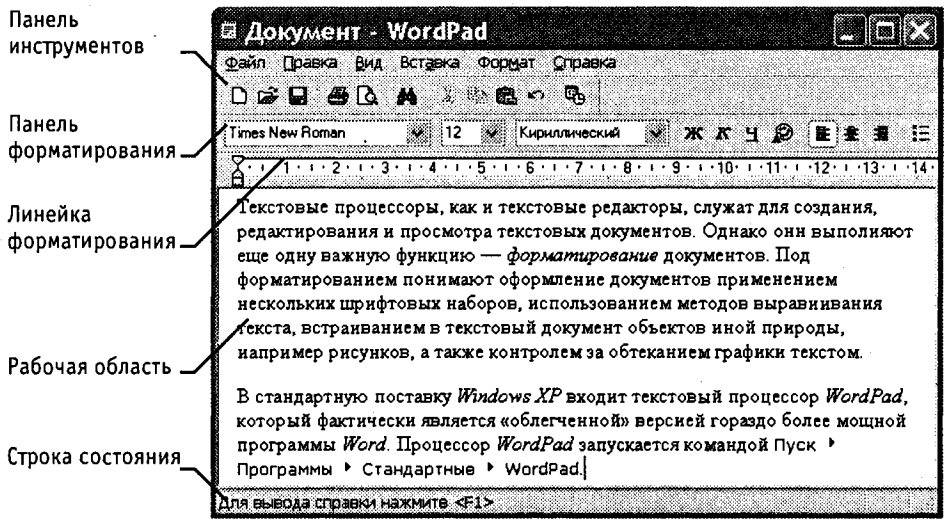


Рис. 7.5. Окно текстового процессора WordPad

Word. Процессор *WordPad* запускается командой Пуск ▸ Программ ▸ Стандартные ▸ WordPad. Рабочее окно программы представлено на рис. 7.5. Как видно из этого рисунка, в отличие от текстового редактора Блокнот окно текстового процессора содержит дополнительную панель элементов управления — панель форматирования.

Поскольку с приемами создания и редактирования документа мы знакомы по текстовому редактору Блокнот (см. выше), то на примере текстового процессора *WordPad* мы ознакомимся с простейшими приемами форматирования документов.

Настройка параметров печатной страницы. Форматирование документа предполагает получение полноценного бумажного оттиска на печатающем устройстве. Поэтому работа в текстовых процессорах начинается с задания параметров печатной страницы. Параметры страницы задают в диалоговом окне Параметры страницы (Файл ▸ Параметры страницы).

Настройку параметров печатной страницы следует выполнять в соответствии с тем типом принтера, который предполагается использовать для печати. Для выбора принтера служит кнопка Принтер в диалоговом окне Параметры страницы. В раскрывающемся списке Принтер приведены те модели принтеров, на которые настроена конфигурация данного компьютера.

После выбора модели принтера выбирают параметры печатной страницы. Размеры листа бумаги выбирают в раскрывающемся списке Размер. В России в качестве стандартного машинописного листа принято использовать лист формата А4, имеющий размер 210×297 мм.

При печати принято ориентировать лист так, чтобы его высота была больше ширины (Книжная ориентация). Альбомная ориентация применяется в особых случаях, например при печати двух страниц на одном листе.

Печатное поле документа составляет не весь бумажный лист, поскольку со всех сторон документа должны оставаться белые поля. При выборе размеров полей следует учитывать следующие обстоятельства:

- если левое поле используют для брошюровки, оно должно иметь увеличенный размер;
- если при брошюровке предполагается обрезка блока, правое и нижнее поля должны иметь увеличенный размер;
- если при оформлении документа используются колонтитулы (верхние или нижние), для них следует предусмотреть увеличение размера соответствующих полей.

Конкретные значения размеров полей следует выяснить у заказчика документа (работодателя, администрации предприятия). Если никаких рекомендаций нет, можно задать для всех полей, кроме левого, по 15 мм, а для левого поля — 25 мм.

Настройка параметров абзаца. Абзац является минимальным элементом форматирования. Настройка параметров абзаца выполняется в диалоговом окне Абзац, открываемом командой Формат ▸ Абзац. Здесь можно задать следующие параметры:

- величину отступа от левого поля;
- величину отступа от правого поля;
- величину специального отступа для первой строки абзаца (используется для создания «красной строки»);
- метод выравнивания: по левому полю, по центру и по правому полю. К сожалению, текстовый процессор *WordPad* не имеет средств для выравнивания текста «по ширине» — так называется метод выравнивания, при котором текст выравнивается и по левому, и по правому полям одновременно. Для большинства документов, написанных на русском языке, этот метод является стандартным.

Настройка параметров шрифтового набора. Тип используемого шрифта, его размер и начертание можно задать как с помощью строки меню (команда Формат), так и с помощью элементов управления, представленных на панели форматирования. В отличие от редактора Блокнот, текстовый процессор *WordPad* позволяет применять шрифтовое оформление как ко всему документу в целом, так и к отдельным, предварительно выделенным фрагментам.

Создание маркированных списков. Создание маркированных списков — характерная возможность большинства текстовых процессоров. В программе *WordPad* первая строка маркированного списка создается командой (Формат ▸ Маркер) или щелчком на кнопке Маркеры на панели форматирования.

Последующие строки автоматически получают маркер после нажатия клавиши ENTER. Для прекращения маркировки надо просто повторить команду еще раз.

Управление табуляцией. Режим табуляции определяет характер линейного смещения текстового курсора в строке при последовательных нажатиях клавиши TAB. Табуляцией пользуются в тех случаях, когда есть необходимость оформления текста ровными столбцами, что в большинстве случаев необходимо при создании таблиц.

Позиции табуляции задают в диалоговом окне Табуляция (Формат ▸ Табуляция). Координаты позиции табуляции задаются в сантиметрах и измеряются от левого поля. Например, если задать три позиции (5 см, 10 см и 15 см), то при нажатии клавиши TAB текстовый курсор в зависимости от текущего положения смещается вправо к ближайшей позиции табуляции.

Поиск и замена текстовых фрагментов. Наличие средства поиска и замены текстового фрагмента — обязательный элемент текстовых процессоров. В программе *WordPad* средство поиска запускают командой Правка ▸ Найти. Текстовый фрагмент, подлежащий поиску, вводят в поле Что, а процесс поиска запускают щелчком на кнопке Найти далее. Установкой флажков Только слово целиком и С учетом регистра настраивают особенности поиска.

Поиск с одновременной заменой запускают командой Правка ▸ Заменить. Разыскиваемый фрагмент вводят в поле Что, а замещающий фрагмент — в поле Чем. Поиск выполняют командой Найти далее, замену фрагмента — командой Заменить, а глобальную замену по всему тексту — командой Заменить все.

Возможность автоматической замены используют для автоматизации ввода текста и редактирования. Так, например, если при вводе текста набирать слова в сокращенном виде: к-р, к-ра, к-ров и т. п., а потом по всему тексту заменить к-р на компьютер, то можно значительно сократить объем ввода с клавиатуры. Таким же образом правят систематические ошибки, обнаруженные в ходе редактирования.

7.2. Принципы внедрения и связывания объектов

Операционная система *Windows* позволяет:

- создавать комплексные документы, содержащие несколько разных типов данных;
- обеспечивать совместную работу нескольких приложений при подготовке одного документа;
- переносить и копировать объекты между приложениями.

Так, например, рисунок, созданный в графическом редакторе *Paint*, можно скопировать в текстовый документ, разрабатываемый в текстовом процессоре *WordPad*. То же можно делать и с фрагментами звукозаписи и видеозаписи. Разумеется, звуковой объект нельзя отобразить на печатной странице, но если документ электронный, то его можно вставить в текст в виде значка. Щелчок на этом значке во время просмотра документа позволит прослушать связанную с ним звукозапись.

Возможность использования в одном документе объектов различной природы является очень мощным инструментом *Windows*. Она основана на так называемой *концепции внедрения и связывания объектов* (*OLE — Object Linking and Embedding*).

Внедрение объектов

Под внедрением объектов подразумевается создание комплексного документа, содержащего два или более автономных объектов. Обычным средством внедрения объектов в документ является их *импорт* из готового файла, в котором данный объект хранится. Так, например, если в графическом редакторе *Paint* был создан

и сохранен на диске файл рисунка ABCD.BMP, то в текстовом процессоре *WordPad* этот рисунок можно вставить в текстовый документ с помощью команды Вставка ▸ Объект. При этом открывается диалоговое окно Вставка объекта.

Импорт вставляемого объекта обеспечивается переключателем Создать из файла, а его выбор на диске — кнопкой Обзор. Кроме графических объектов в текстовый документ можно внедрять и объекты другой природы — тексты, фрагменты звукозаписи и видеозаписи.

При сохранении комплексного документа происходит сохранение и текста, и всех внедренных в него объектов. Рисунок, ранее существовавший в виде отдельного графического файла, теперь внедрен в текстовый документ и располагается внутри него. Разумеется, при этом размер исходного текстового документа возрастает на величину внедренных объектов.

Связывание объектов

Однако мы могли поместить рисунок в текстовый документ и другим способом. В том же диалоговом окне Вставка объекта есть флажок, который называется Связь. Если установить этот флажок перед вставкой объекта, то происходит другой тип вставки, который называется *связыванием*. Связывание отличается от внедрения тем, что сам объект не вставляется в документ, а вместо этого вставляется только указатель на местоположение объекта. Когда при просмотре документа читатель дойдет до этого указателя, текстовый процессор обратится по адресу, имеющемуся в указателе, и отобразит рисунок в тексте документа.

При использовании связывания объектов, а не внедрения, размер результирующего комплексного документа практически не увеличивается, так как указатель занимает очень мало места. Однако, если не принять специальные меры, то при передаче такого документа заказчику не произойдет передача связанных объектов, поскольку они останутся в своих местах хранения. Это явление называется *разрывом*, или *потерей связи*. Потерянные связи надо восстанавливать. Потеря связи может происходить даже при простом перемещении связанных объектов из одной папки в другую. Таким образом, при использовании метода связывания объектов необходимо специально контролировать целостность связей между объектами и выполнять операции обслуживания этих связей (обновления и восстановления).

Сравнение методов внедрения и связывания

И тот и другой методы имеют свои области применения. Все зависит от формы и назначения документа. Внедряя объекты, мы избавляемся от необходимости поддерживать и обслуживать связи, но при этом можем получать файлы огромных размеров, с которыми трудно оперировать. Связывая объекты, мы резко уменьшаем размеры файлов и значительно повышаем производительность компьютера, но вынуждены следить за тем, чтобы все связанные объекты хранились строго в тех папках, в которые они были помещены в момент создания связи.

С принципами связывания и внедрения объектов непосредственно соприкасается принцип *совместного использования объектов*. В корпоративных вычислительных

системах нередко используют стандартизированные объекты (бланки документов, логотипы предприятий и т. п.), доступ к которым (без права изменения) имеют большие группы сотрудников.

Такие объекты удобно вставлять в результирующий документ методом связывания. Во-первых, это позволяет значительно сократить объем документации предприятия, так как один и тот же объект может использоваться во всех документах без размножения. Во-вторых, такой подход позволяет администрации предприятия легко изменять (в случае необходимости) стандартный объект и иметь уверенность в том, что при использовании любого документа, имеющего с ним связь, произойдет автоматическая подмена объекта. При таком подходе за пределы предприятия не выйдет ни один документ, напечатанный на устаревшем бланке, имеющем устаревшие реквизиты и т. п.

Итак, на практике обычно поступают следующим образом. Если документ готовится для печати на принтере или для просмотра на экране в пределах локальной сети предприятия, то объекты в него вставляют методом связывания. Если же документ готовится для передачи в электронном виде во внешние структуры, в него объекты внедряются.

OLE-серверы и OLE-клиенты

Объект — это очень специфическое образование, и не каждое приложение может его создать. Те приложения, которые способны создавать объекты для передачи другим приложениям, называются *OLE-серверами*, а те, которые позволяют внедрять или связывать чужие объекты в свои документы, называются *OLE-клиентами*. Например, при вставке рисунка в текстовый документ графический редактор играет роль *OLE*-сервера, а текстовый процессор — роль *OLE*-клиента.

7.3. Служебные приложения Windows XP

Служебные приложения *Windows XP* предназначены для обслуживания персонального компьютера и самой операционной системы. Они позволяют находить и устранять дефекты файловой системы, оптимизировать настройки программного и аппаратного обеспечения, а также автоматизировать некоторые рутинные операции, связанные с обслуживанием компьютера.

В Главном меню служебные приложения *Windows XP* сосредоточены в категории Пуск ▸ Программы ▸ Стандартные ▸ Служебные. Они поставляются в составе операционной системы и устанавливаются вместе с ней (полностью или выборочно). Ниже приведена краткая характеристика основных служебных приложений.

Буфер обмена

Приложение Буфер обмена предназначено для просмотра текущего содержания буфера обмена *Windows*. С его помощью можно выполнить сохранение содержимого буфера обмена в виде файла специального формата (.CLP) или его загрузку. Соответствующие команды — Файл ▸ Сохранить как и Файл ▸ Открыть.

Дефрагментация диска

Дефрагментация диска — служебное приложение, предназначенное для повышения эффективности работы жесткого диска путем устранения фрагментированности файловой структуры.

Наименьшей единицей хранения данных на диске является *кластер*. Если свободного места на диске достаточно, то файлы записываются так, что кластеры, в которые происходит запись, располагаются последовательно. В этом случае обращения к файлу происходят достаточно быстро, поскольку затраты времени на поиск очередных кластеров минимальны.

Если диск заполнен до отказа, запись на него возможна только после освобождения некоторого количества кластеров путем удаления файлов. При этом свободные области, образующиеся на диске, в общем случае не образуют одну большую непрерывную область. При попытке записать длинный файл на диск, имеющий прерывистую структуру свободных областей, файл делится на фрагменты, которые записываются туда, где для них нашлось место. Длительная работа с заполненным жестким диском приводит к постепенному увеличению фрагментированности файлов и значительному замедлению работы.

Программа Дефрагментация диска выполняет перекомпоновку файлов таким образом, что длинные файлы собираются из коротких фрагментов. В результате доступ к файлам заметно упрощается и эффективность работы компьютера возрастает.

Сведения о системе

Сведения о системе — это специальный пакет программных средств, собирающих сведения о настройке операционной системы *Windows XP*, ее приложений и оборудования компьютерной системы. Средства этого пакета предназначены для специалистов, выполняющих ремонтно-восстановительные работы. Его дополнительное преимущество состоит в том, что он позволяет провести диагностику компьютера с удаленного сервера.

Таблица символов

Кроме шрифтов с алфавитно-цифровыми символами в операционной системе *Windows XP* можно использовать и специальные символьные наборы с дополнительными элементами оформления текстовых документов. В любом текстовом процессоре этими символьными наборами можно пользоваться точно так же, как обычными шрифтами. Однако если для обычных шрифтов раскладка клавиш понятна, то для символьных наборов нужны специальные средства, чтобы установить закрепление символов за клавишами клавиатуры.

Программа Таблица символов позволяет увидеть на экране все символы заданного набора и установить, какой символ какой клавише соответствует. Рабочее окно программы Таблица символов показано на рис. 7.6. В качестве примера в нее загружен символьный набор *Wingdings*, входящий в комплект поставки *Windows XP*.

Выбор просматриваемого шрифта выполняется в раскрывающемся списке Шрифт. Если навести указатель мыши на один из символов, входящих в набор, и щелкнуть

Рис. 7.6. Окно программы Таблица символов

левой кнопкой, этот символ отображается в увеличенном виде. В строке состояния программы при этом появляется запись, указывающая код данного символа, а для некоторых стандартных шрифтовых наборов — также его описание и клавиатурная комбинация для ввода.

Восстановление системы

Операционная система *Windows XP* имеет в своем составе средства, позволяющие восстановить ее безошибочную работу в случае повреждения каких-либо системных файлов. Механизм такого восстановления основан на создании так называемых *контрольных точек*, содержащих сведения о состоянии системы и копии важных системных файлов. При возникновении каких-либо неполадок можно воспользоваться существующей контрольной точкой и вернуть систему в прежнее работоспособное состояние.

Созданием контрольных точек и восстановлением системы руководит специальная программа — Восстановление системы. Операционная система автоматически создает контрольные точки, когда выполняются «опасные», на ее взгляд, операции. Кроме того, контрольную точку можно создать в любой момент по инициативе пользователя. Восстановление системы выполняется только по явной команде пользователя.

Недостатком механизма восстановления являются значительные затраты дискового пространства на хранение контрольных точек. Если аппаратная или программная конфигурация компьютера подвержена регулярным изменениям, то объем дискового пространства, используемый для хранения данных контрольных точек, может выйти за любые разумные пределы. В этом случае средство Восстановление системы можно отключить, отдавая себе отчет, что надежность работы компьютера в этом случае может снизиться. Снова активировать это средство можно в любой момент.

Мастер переноса файлов и параметров

Переход от одной версии операционной системы к другой или замена устаревшего компьютера новым сопровождается большим объемом технической работы. Необходимо убедиться в надежном и безошибочном переносе всех данных, а также необходимых настроек, привычных для пользователя.

Сэкономить время и автоматизировать процесс переноса данных в системе *Windows XP* помогает Мастер переноса файлов и параметров. Он позволяет перенести такие личные настройки, как характеристики экрана (Рабочего стола), параметры папок и Панели задач. В число копируемых данных входят архив сообщений электронной почты, а также данные из стандартных пользовательских папок, например \Мои документы и \Избранное.

Наблюдение за функционированием компьютера и операционной системы

Операционная система *Windows XP* содержит средства для визуального или протокольного наблюдения за функционированием компьютера и операционной системы. В предыдущих версиях *Windows* для этой цели использовалась служебная программа Системный монитор. В *Windows XP* для этой цели применяется специальная административная программа Производительность.

Чтобы запустить ее, откройте папку Панель управления (Пуск ▸ Настройка ▸ Панель управления). Теперь дважды щелкните на значке Администрирование, а затем — на значке Производительность.

Это средство позволяет контролировать загрузку процессора, распределение оперативной памяти, обмен данными между дисками и другие параметры вычислительной системы. Результаты наблюдения можно отображать на экране в виде графиков или записывать в протокольный файл. Исследование компьютера с применением такого средства позволяет находить «узкие места» в производительности компьютерной системы, сравнивать между собой варианты настройки аппаратных и программных средств.

Средства командной строки

Ряд средств специфической настройки компьютера недоступны в системе *Windows XP* через Главное меню. Как правило, такие ограничения наложены на потенциально опасные и редко используемые средства. Предполагается, что этими средствами в случае необходимости воспользуется специалист по обслуживанию компьютера.

Запуск подобных средств осуществляется с командной строки. Команда и параметры вводятся либо в диалоговом окне Запуск программы (Пуск ▸ Выполнить) либо в специальном окне Командная строка (Пуск ▸ Программы ▸ Стандартные ▸ Командная строка). Вот некоторые из полезных программ, запускаемых таким образом.

- regedit.exe — программа для ручного редактирования Реестра *Windows*, служебной базы данных, содержащей сведения об аппаратно-программной конфигурации компьютера.

- convert.exe — программа для преобразования файловой системы диска к более совершенному формату без уничтожения данных. Поддерживает преобразование от *FAT16* или *FAT32* к *NTFS*. Обратное преобразование невозможно.
- msconfig.exe — программа, задающая настройки, управляющие процессом начальной загрузки операционной системы. Позволяет редактировать системные файлы конфигурации и на временной основе отключать и подключать драйверы и команды, используемые в ходе начальной загрузки системы.

7.4. Стандартные средства мультимедиа

Мультимедиа — понятие комплексное. С одной стороны, оно подразумевает особый тип документов, а с другой стороны — особый класс программного и аппаратного обеспечения. *Мультимедийные документы* отличаются от обычных тем, что кроме традиционных текстовых и графических данных могут содержать звуковые и музыкальные объекты, анимированную графику (мультипликацию), видеофрагменты. *Мультимедийное программное обеспечение* — это программные средства, предназначенные для создания и/или воспроизведения мультимедийных документов и объектов. *Мультимедийное аппаратное обеспечение* — это оборудование, необходимое для создания, хранения и воспроизведения мультимедийного программного обеспечения. Исторически к нему относятся звуковая карта, дисковод *CD-ROM* и звуковые колонки. Эту группу оборудования называют также *базовым мультимедийным комплектом*.

В последние годы класс аппаратных средств мультимедиа бурно развивается. Так, в него вошли устройства для обработки телевизионных сигналов и воспроизведения телепрограмм (*ТВ-тюнеры*), аппаратные средства для обработки сжатой видеоинформации (*MPEG-декодеры*), дисководы для воспроизведения цифровых видеодисков (*DVD*), оборудование для записи компакт-дисков (*CD-R* и *CD-RW*) и многое другое.

При наличии мультимедийного аппаратного обеспечения (хотя бы в объеме базового мультимедийного комплекта) операционная система *Windows XP* позволяет создавать, хранить и использовать мультимедийные объекты и документы. Программные средства, предназначенные для этой цели, находятся в категории Программы ▸ Стандартные ▸ Развлечения. К основным стандартным средствам мультимедиа относятся программы: Громкость, Звукозапись и Проигрыватель *Windows Media*.

Громкость

Программа Громкость является базовым регулятором громкости всей компьютерной системы. Это значит, что она выполняет центральную роль, и все регулировки громкости иных программ или аппаратных средств действуют только в пределах, первично заданных программой Громкость.

С помощью Панели управления (Пуск ▸ Настройка ▸ Панель управления ▸ Звуки и аудиоустройства ▸ Громкость ▸ Отображать значок на панели задач), значок средства Громкость можно отобразить на панели индикации. Щелчок левой кнопки мыши

на этом значке открывает мастер-регулятор, оказывающий влияние на все звуковые устройства, установленные в компьютере. Двойным щелчком можно открыть расширенное окно, в котором громкость, стереобаланс и установки тембра задаются для каждого из устройств отдельно.

Звукозапись

Программа Звукозапись предназначена для самостоятельного создания файлов звукозаписи. В качестве источника звука может использоваться микрофон, дисковод *CD-ROM* или внешнее устройство. Программа имеет графические элементы управления, эквивалентные органам управления обычного бытового магнитофона. Создаваемые звуковые файлы могут проходить ограниченное редактирование с наложением некоторых эффектов (изменение скорости звукозаписи, громкости, эффект «Эхо», обращение звукозаписи). Программа позволяет создавать аудиоклипы небольших размеров, которые можно использовать в звуковых схемах оформления системных событий. Ее также используют в качестве *OLE*-сервера при необходимости вставить звуковой объект в текстовый документ.

Проигрыватель Windows Media

В системе *Windows XP* Проигрыватель *Windows Media* представляет собой универсальное средство для воспроизведения на компьютере всех видов видео- и аудиозаписей. В частности, с его помощью можно воспроизводить:

- музыкальные компакт-диски;
- файлы аудио и видео;
- потоковые записи из Интернета.

В ранних версиях *Windows* эти функции возлагались на две отдельные программы: Лазерный проигрыватель (специально для музыкальных компакт-дисков) и Универсальный проигрыватель (для всех типов аудио и видео файлов). Возможность прослушивания потоковых звукозаписей Интернета появилась только в программе Проигрыватель *Windows Media*.

Экранные элементы управления Проигрывателя *Windows Media* соответствуют типичным органам управления бытовых электронных проигрывателей, магнитофонов, музыкальных центров.

При воспроизведении музыкальных компакт-дисков Проигрыватель *Windows Media* способен загрузить из Интернета описание компакт-диска: название, имя автора или исполнителя, а также список дорожек. Предусмотрена также возможность копирования записей и сохранения их в сжатом формате *MP3*.

В число дополнительных средств управления воспроизведением входят:

- средства управления объемным звучанием;
- графический эквалайзер;
- средства выбора зрительного образа;
- средства настройки видео.

Рис. 7.7. Воспроизведение аудиозаписи при помощи Проигрывателя Windows Media

Программа Проигрыватель *Windows Media* также допускает использование схем оформления («обложек»), позволяющих полностью изменить внешний вид окна программы.

Практическое занятие

Упражнение 7.1. Приемы работы с текстовым редактором Блокнот

15 мин

1. Запустите текстовый редактор Блокнот (Пуск ▸ Программы ▸ Стандартные ▸ Блокнот).
2. Убедитесь, что включена русская раскладка клавиатуры. В противном случае щелкните на указателе языка на языковой панели и выберите в открывшемся меню пункт Русский. Если языковая панель закрыта, воспользуйтесь комбинацией клавиш, выбранной на данном компьютере.
3. Введите с клавиатуры слово Конденсатор (при вводе заглавной буквы удерживайте нажатой клавишу SHIFT) и нажмите клавишу ENTER.
4. Далее введите с клавиатуры термины Резистор, Катушка индуктивности, Выключатель, Амперметр и Вольтметр, нажимая после ввода каждого термина клавишу ENTER.
5. Расставьте в документе термины по алфавиту, выделяя строки и перемещая их через буфер обмена. Дважды щелкните на слове Амперметр и убедитесь, что оно при этом выделяется (в программе Блокнот этот способ служит для выде-

ления отдельных слов). Нажмите комбинацию клавиш SHIFT+ВПРАВО, чтобы включить в выделенный фрагмент невидимый символ конца строки — курсор при этом переместится в начало следующей строки.

6. Дайте команду Правка ▸ Вырезать, чтобы забрать выделенный фрагмент в буфер обмена. Убедитесь, что он действительно удаляется из документа.
7. Нажмите комбинацию клавиш CTRL+HOME, чтобы установить курсор в начало документа. Дайте команду Правка ▸ Вставить, чтобы вставить фрагмент из буфера обмена.
8. Установите указатель мыши на начало слова Вольтметр. Нажмите левую кнопку мыши и, не отпуская ее, выделите это слово методом протягивания.
9. Нажмите комбинацию клавиш CTRL+X, переместите текстовый курсор в начало второй строки текста и вставьте новый фрагмент из буфера обмена (CTRL+V).
10. Установите текстовый курсор в начало строки, содержащей слова Катушка индуктивности. Дважды нажмите комбинацию SHIFT+CTRL+ВПРАВО и убедитесь, что при каждом нажатии выделенный фрагмент расширяется, охватывая следующее слово. Нажмите комбинацию клавиш SHIFT+ ВПРАВО. Мы выделили нужный фрагмент при помощи клавиатурных команд.
11. Нажмите комбинацию клавиш SHIFT+DELETE, переместите текстовый курсор в начало третьей строки текста и вставьте новый фрагмент из буфера обмена с помощью комбинации клавиш SHIFT+INSERT.
12. Используя описанные приемы, завершите формирование списка введенных терминов в алфавитном порядке.
13. Сохраните созданный документ под именем list.txt.

Мы научились выполнять ввод и редактирование текстов в редакторе Блокнот. Мы освоили несколько приемов выделения и перемещения фрагментов текста через буфер обмена.

Упражнение 7.2. Приемы работы с графическим редактором Paint

15 мин

В этом упражнении мы создадим условное обозначение вольтметра, принятое на электрических схемах.

1. Запустите графический редактор *Paint* (Пуск ▸ Программы ▸ Стандартные ▸ Paint).
2. Убедитесь, что на палитре задан черный цвет в качестве основного и белый — в качестве фонового.
3. Дайте команду Рисунок ▸ Атрибуты, в диалоговом окне Атрибуты задайте ширину рисунка, равную 300 точек, и высоту — 200 точек. Щелкните на кнопке OK.
4. Выберите инструмент Эллипс и в палитре настройки инструмента укажите вариант Без заполнения.
5. Нажмите и удерживайте клавишу SHIFT. Методом протягивания нарисуйте окружность в центральной части области рисунка. Диаметр окружности должен составлять около половины высоты рисунка. Отпустите клавишу SHIFT.

6. Выберите инструмент Линия. В палитре настройки инструмента выберите вариант толщины линии (второй сверху).
7. Нажмите и удерживайте клавишу SHIFT. Методом протягивания нарисуйте небольшой горизонтальный отрезок прямой в стороне от окружности. Отпустите клавишу SHIFT.
8. Выберите инструмент Выделение. В палитре настройки инструмента выберите режим с прозрачным фоном.
9. Методом протягивания выделите прямоугольный фрагмент, охватывающий нарисованный отрезок прямой, но не затрагивающий окружность. Комбинацией клавиш CTRL+X поместите его в буфер обмена.
10. Вставьте отрезок прямой на рисунок комбинацией клавиш CTRL+V. Обратите внимание, что выделение при этом сохраняется.
11. Переместите выделенный фрагмент так, чтобы отрезок прямой примыкал к окружности слева. Обратите внимание на то, что фоновая часть фрагмента не перекрывает окружность.
12. Повторите операции, описанные в пп. 10–11, чтобы создать отрезок прямой, примыкающий к окружности справа.
13. Выберите инструмент Текст. Переключитесь на английскую раскладку клавиатуры.
14. Методом протягивания создайте область ввода текста внутри окружности. Введите символ «V». С помощью панели Шрифты задайте подходящий размер и начертание шрифта.
15. Методом перетаскивания за границу области ввода текста поместите букву «V» в центре окружности.
16. Щелкните вне области ввода текста, чтобы превратить текст в часть рисунка.
17. Сохраните созданное изображение под именем scheme.bmp.

Мы научились создавать простейшие примитивы (эллипс, линия), установили, как влияет регистровая клавиша SHIFT на работу инструментов рисования, научились вводить текстовые данные и компоновать рисунок из объектов.

Упражнение 7.3. Приемы форматирования в текстовом процессоре WordPad

15 мин

В этом упражнении мы создадим иллюстрированный словарь терминов, введенных в файл list.txt в упражнении 7.1.

1. Запустите текстовый процессор *WordPad* (Пуск ▸ Программы ▸ Стандартные ▸ WordPad).
2. Откройте текстовый файл list.txt.
3. Дайте команду Файл ▸ Сохранить как, в списке Тип файла выберите пункт Файл RTF и сохраните файл под именем dict.doc.

4. Выделите первое слово документа (Амперметр). На панели форматирования задайте шрифт Arial, размер шрифта — 14 пунктов, набор символов — Кириллический, выберите полужирное начертание.
5. Нажмите клавишу END, чтобы снять выделение, а затем — клавишу ENTER.
6. Введите краткое описание термина, указанного в предыдущей строке, например так: «прибор для измерения величины электрического тока». Размножьте введенный текст таким образом, чтобы образовался абзац размером 3–4 строки (рис. 7.8).

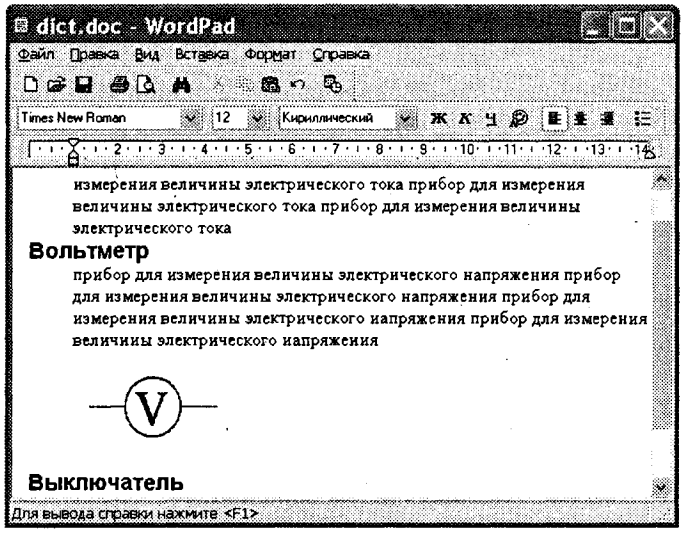


Рис. 7.8. Пример комплексного документа, содержащего встроенный объект

7. Выделите весь только что введенный абзац (можно использовать «тройной щелчок»). На панели форматирования задайте шрифт Times New Roman, размер шрифта — 12 пунктов, набор символов — Кириллический.
8. На линейке, расположенной ниже панели форматирования, перетащите маркер в виде квадратика на расстояние 1 см (по линейке) вправо. Убедитесь, что весь абзац теперь отображается с отступом от левого края.
9. Снимите выделение и установите курсор в начало первой строки того же самого абзаца. Нажмите клавишу TAB. Убедитесь, что табуляция в первой строке абзаца может использоваться для создания абзацного отступа.
10. Введите аналогичные краткие описания для последующих терминов создаваемого «словаря» и отформатируйте термины и описания так, как указано в пп. 4–9.
11. Установите курсор в конец описания термина Вольтметр и нажмите клавишу ENTER.
12. Дайте команду Вставка ▸ Объект. В диалоговом окне Вставка объекта включите переключатель Создать из файла.

13. Щелкните на кнопке Обзор, разыщите в файловой структуре ранее созданный документ scheme.bmp, щелкните на кнопке Открыть. Щелкните на кнопке OK.
14. Убедитесь, что созданное изображение схематического обозначения вольтметра вставлено в документ в качестве иллюстрации.
15. Измените масштаб отображения рисунка в документе путем перетаскивания маркеров изменения размера, расположенных на границах объекта.
16. Сохраните текущий документ dict.doc.

Мы научились выполнять форматирование текста с помощью текстового процессора WordPad. В частности, мы научились по-разному оформлять заголовки и абзацы основного текста, а также встраивать графические объекты из внешнего источника.

Упражнение 7.4. Сопоставление приемов внедрения и связывания объектов

30 мин

В предыдущем упражнении мы создали комбинированный документ dict.doc, содержащий внедренную иллюстрацию. В этом упражнении мы поместим тот же объект методом связывания.

1. Запустите текстовый процессор *WordPad* (Пуск ▸ Программы ▸ Стандартные ▸ WordPad). Откройте файл dict.doc.
2. Удалите внедренный рисунок. Выделите его щелчком и нажмите клавишу DELETE.
3. Дайте команду Вставка ▸ Объект. В диалоговом окне Вставка объекта установите переключатель Создать из файла.
4. Щелкните на кнопке Обзор и разыщите в файловой структуре документ scheme.bmp. Щелкните на кнопке Открыть — диалоговое окно Обзор закроется.
5. Установите флажок Связь, чтобы установить связь с рисунком (операция связывания). Щелкните на кнопке OK.
6. Дайте команду Файл ▸ Сохранить как и сохраните документ под именем dict1.doc. Закройте программу *WordPad*.
7. Запустите программу *Paint*. Откройте файл scheme.bmp и измените его, например, закрасив внешнюю часть рисунка другим цветом с помощью инструмента Заливка. Сохраните рисунок scheme.bmp.
8. Запустите программу *WordPad*. Откройте документ dict.doc. Проверьте, изменился ли его вид.
9. Закройте документ dict.doc. Откройте документ dict1.doc. Обратите внимание на вспомогательную операцию, выполняемую после загрузки документа. Проверьте, изменился ли вид этого документа. Чем вы объясните обнаруженные различия между документами?

Мы научились выполнять операцию связывания объектов с текстовым документом и исследовали различие между операциями внедрения и связывания.

Упражнение 7.5. Контроль загруженности процессора

1. Запустите консольную программу Производительность (Пуск ▸ Настройка ▸ Панель управления ▸ Администрирование ▸ Производительность).
2. Поочередно выбирая все показатели в списке в нижней части правой панели, щелкайте на кнопке Удалить на панели инструментов.
3. Щелкните на кнопке Добавить на панели инструментов.
4. В диалоговом окне Добавить счетчики в списке Объект выберите пункт Процессор. Установите переключатель Выбрать счетчики из списка. В списке счетчиков выберите пункт % загруженности процессора. Щелкните на кнопке Добавить, а затем на кнопке Закрыть.
5. Подождите некоторое время, чтобы оценить загруженность процессора в отсутствие каких-либо активных действий (фактически она определяется необходимостью обслуживания самой программы Производительность).
6. Двойным щелчком на значке Мой компьютер откройте окно Мой компьютер. Измените размер окна так, чтобы в нем помещалось 4–6 значков.
7. Щелкните правой кнопкой мыши на свободном от значков месте экрана и выберите в контекстном меню пункт Свойства. Откройте вкладку Оформление. Щелкните на кнопке Эффекты и установите флажок Отображать содержимое окна при перетаскивании. Закройте диалоговые окна, щелкая каждый раз на кнопке OK.
8. Наведите указатель мыши на строку заголовка окна Мой компьютер и в течение 10–20 секунд подвигайте окно по экрану, следя за показателями в окне Производительность. Запишите среднюю загрузку процессора во время этой операции.
9. Щелкните правой кнопкой мыши на свободном от значков месте экрана и выберите в контекстном меню пункт Свойства. Откройте вкладку Оформление. Щелкните на кнопке Эффекты и сбросьте флажок Отображать содержимое окна при перетаскивании. Закройте диалоговые окна, щелкая каждый раз на кнопке OK.
10. Наведите указатель мыши на строку заголовка окна Мой компьютер и в течение нескольких секунд подвигайте окно по экрану, следя за показателями в окне Производительность. Запишите среднюю загрузку процессора во время этой операции.
11. Результаты эксперимента занесите в таблицу.

Дежурный режим	Перетаскивание окна без отображения содержимого	Перетаскивание окна с отображением содержимого

8.1. Компьютерные сети

Назначение компьютерных сетей

При физическом соединении двух или более компьютеров образуется *компьютерная сеть*. В общем случае, для создания компьютерных сетей необходимо специальное аппаратное обеспечение (*сетевое оборудование*) и специальное программное обеспечение (*сетевые программные средства*). Простейшее соединение двух компьютеров для обмена данными называется *прямым соединением*. Для создания прямого соединения компьютеров, работающих в операционной системе *Windows XP*, не требуется ни специального аппаратного, ни программного обеспечения. В этом случае аппаратными средствами являются стандартные порты ввода/вывода (последовательный или параллельный), а в качестве программного обеспечения используется стандартное средство, имеющееся в составе операционной системы (Пуск ▸ Программы ▸ Стандартные ▸ Связь ▸ Мастер новых подключений ▸ Установить прямое подключение к другому компьютеру).

Все компьютерные сети без исключения имеют одно назначение — обеспечение совместного доступа к общим *ресурсам*. Слово *ресурс* — очень удобное. В зависимости от назначения сети в него можно вкладывать тот или иной смысл. Ресурсы бывают трех типов: *аппаратные*, *программные* и *информационные*. Например, устройство печати (принтер) — это аппаратный ресурс. Емкости жестких дисков — тоже аппаратный ресурс. Когда все участники небольшой компьютерной сети пользуются одним общим принтером, это значит, что они разделяют общий аппаратный ресурс. То же можно сказать и о сети, имеющей один компьютер с увеличенной емкостью жесткого диска (*файловый сервер*), на котором все участники сети хранят свои архивы и результаты работы.

Кроме аппаратных ресурсов компьютерные сети позволяют совместно использовать *программные ресурсы*. Так, например, для выполнения очень сложных и продолжительных расчетов можно подключиться к удаленной большой ЭВМ и отпра-

вить вычислительное задание на нее, а по окончании расчетов точно так же получить результат обратно.

Данные, хранящиеся на удаленных компьютерах, образуют *информационный ресурс*. Роль этого ресурса сегодня видна наиболее ярко на примере Интернета, который воспринимается, прежде всего, как гигантская информационно-справочная система.

Наши примеры с делением ресурсов на аппаратные, программные и информационные достаточно условны. На самом деле, при работе в компьютерной сети любого типа одновременно происходит совместное использование всех типов ресурсов. Так, например, обращаясь в Интернет за справкой о содержании вечерней телевизионной программы, мы безусловно используем чьи-то аппаратные средства, на которых работают чужие программы, обеспечивающие поставку затребованных нами данных.

Локальные и глобальные сети. Основные понятия

Основной задачей, решаемой при создании компьютерных сетей, является обеспечение совместимости оборудования по электрическим и механическим характеристикам и обеспечение совместимости информационного обеспечения (программ и данных) по системе кодирования и формату данных. Решение этой задачи относится к области стандартизации и основано на так называемой модели *OSI* (*модель взаимодействия открытых систем — Model of Open System Interconnections*). Она создана на основе технических предложений Международного института стандартов *ISO* (*International Standards Organization*).

Согласно модели *ISO/OSI* архитектуру компьютерных сетей следует рассматривать на разных уровнях (общее число уровней — до семи). Самый верхний уровень — *прикладной*. На этом уровне пользователь взаимодействует с вычислительной системой. Самый нижний уровень — *физический*. Он обеспечивает обмен сигналами между устройствами. Обмен данными в системах связи происходит путем их перемещения с верхнего уровня на нижний, затем транспортировки и, наконец, обратным воспроизведением на компьютере клиента в результате перемещения с нижнего уровня на верхний.

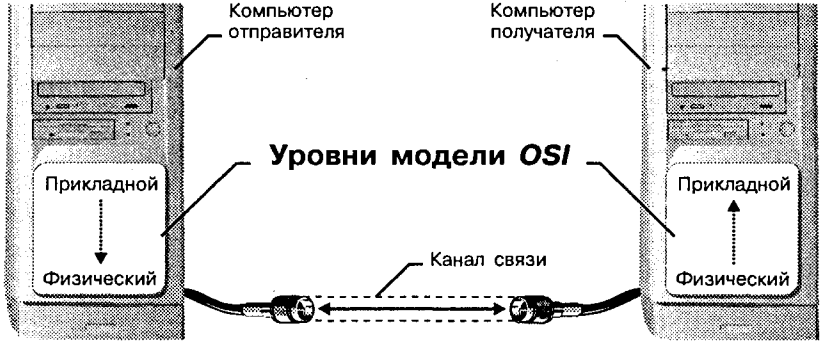


Рис. 8.1. Простейшая модель обмена данными в компьютерной сети

Для обеспечения необходимой совместимости на каждом из семи возможных уровней архитектуры компьютерной сети действуют специальные стандарты, называемые *протоколами*. Они определяют характер аппаратного взаимодействия компонентов сети (*аппаратные протоколы*) и характер взаимодействия программ и данных (*программные протоколы*). Физически функции поддержки протоколов исполняют аппаратные устройства (*интерфейсы*) и программные средства (*программы поддержки протоколов*). Программы, выполняющие поддержку протоколов, также называют *протоколами*.

Так, например, если два компьютера соединены между собой прямым соединением, то на низшем (физическом) уровне протокол их взаимодействия определяют конкретные устройства физического порта (параллельного или последовательного) и механические компоненты (разъемы, кабель и т. п.). На более высоком уровне взаимодействие между компьютерами определяют программные средства, управляющие передачей данных через порты. Для стандартных портов они находятся в базовой системе ввода/вывода (*BIOS*). На самом высоком уровне протокол взаимодействия обеспечивают приложения операционной системы.

В соответствии с используемыми протоколами компьютерные сети принято разделять на *локальные* (*LAN — Local Area Network*) и *глобальные* (*WAN — Wide Area Network*). Компьютеры локальной сети используют единый комплект протоколов для всех участников. По территориальному признаку локальные сети отличаются компактностью. Они могут объединять компьютеры одного помещения, этажа, здания, группы компактно расположенных сооружений. Глобальные сети имеют, как правило, увеличенные географические размеры. Они могут объединять как отдельные компьютеры, так и отдельные локальные сети, в том числе и использующие различные протоколы.

Группы сотрудников, работающих над одним проектом в рамках локальной сети, называются *рабочими группами*. В рамках одной локальной сети могут работать несколько рабочих групп. У участников рабочих групп могут быть разные права для доступа к общим ресурсам сети. Совокупность приемов разделения и ограничения прав участников компьютерной сети называется *политикой сети*. Управление сетевыми политиками (их может быть несколько в одной сети) называется *администрированием сети*. Лицо, управляющее организацией работы участников локальной компьютерной сети, называется *системным администратором*.

Создание локальных сетей характерно для отдельных предприятий или отдельных подразделений предприятий. Если предприятие (или отрасль) занимает обширную территорию, то отдельные локальные сети могут объединяться в глобальные сети. В этом случае локальные сети связывают между собой с помощью любых традиционных каналов связи (кабельных, спутниковых, радиорелейных и т. п.). Как мы увидим ниже, при соблюдении специальных условий для этой цели могут быть использованы даже телефонные каналы, хотя они в наименьшей степени удовлетворяют требованиям цифровой связи.

Простейшее устройство для соединения между собой двух локальных сетей, использующих одинаковые протоколы, называется *мостом*. Мост может быть аппаратным

(специализированный компьютер) или программным. Цель моста — не выпускать за пределы локальной сети данные, предназначенные для внутреннего потребления. Вне сети такие данные становятся «сетевым мусором», впустую занимающим каналы связи.

Для связи между собой нескольких локальных сетей, работающих по разным протоколам, служат специальные средства, называемые *шлюзами*. Шлюзы могут быть как аппаратными, так и программными. Например, это может быть специальный компьютер (*шлюзовый сервер*), а может быть и компьютерная программа. В последнем случае компьютер может выполнять не только функцию шлюза, но и какие-то иные функции, типичные для рабочих станций.

При подключении локальной сети предприятия к глобальной сети важную роль играет понятие *сетевой безопасности*. В частности, должен быть ограничен доступ в локальную сеть для посторонних лиц извне, а также ограничен выход за пределы локальной сети для сотрудников предприятия, не имеющих соответствующих прав. Для обеспечения сетевой безопасности между локальной и глобальной сетью устанавливают так называемые *брандмауэры*. Брандмауэром может быть специальный компьютер или компьютерная программа, препятствующая несанкционированному перемещению данных между сетями.

Сетевые службы. Основные понятия

Понятие виртуального соединения. Рассмотрим простой пример взаимодействия двух корреспондентов с помощью обычной почты. Если они регулярно отправляют друг другу письма и, соответственно, получают их, то они могут полагать, что между ними существует соединение на пользовательском (*прикладном*) уровне. Однако это не совсем так. Такое соединение можно назвать *виртуальным*. Оно было бы физическим, если бы каждый из корреспондентов лично относил другому письмо и вручал в собственные руки. В реальной жизни он бросает его в почтовый ящик и ждет ответа.

Сбором писем из общественных почтовых ящиков и доставкой корреспонденции в личные почтовые ящики занимаются местные почтовые службы. Это другой уровень модели связи, лежащий ниже. Для того чтобы наше письмо достигло адресата в другом городе, должна существовать связь между нашей местной почтовой службой и его местной почтовой службой. Это еще один пример виртуальной связи, поскольку никакой физической связью эти службы не обладают — поступившую почтовую корреспонденцию они только сортируют и передают на уровень федеральной почтовой службы.

Федеральная почтовая служба в своей работе опирается на службы очередного уровня, например на почтово-багажную службу железнодорожного ведомства. И только рассмотрев работу этой службы, мы найдем, наконец, признаки физического соединения, например железнодорожный путь, связывающий два города.

Это очень простой пример, поскольку в реальности даже доставка обычного письма может затронуть гораздо большее количество служб. Но нам важно обратить внимание на то, что в нашем примере образовалось несколько виртуальных соединений между аналогичными службами, находящимися в пунктах отправки и приема.

Не вступая в прямой контакт, эти службы взаимодействуют между собой. На каком-то уровне письма укладываются в мешки, мешки пломбируют, к ним прикладывают сопроводительные документы, которые где-то в другом городе изучаются и проверяются на аналогичном уровне.

Модель взаимодействия открытых систем. Выше мы упомянули о том, что согласно рекомендациям Международного института стандартизации *ISO* системы компьютерной связи рекомендуется рассматривать на семи разных уровнях (таблица 8.1).

Таблица 8.1. Уровни модели связи

Уровень	Аналогия
Прикладной уровень	Письмо написано на бумаге. Определено его содержание
Уровень представления	Письмо запечатано в конверт. Конверт заполнен. Наклеена марка. Клиентом соблюдены необходимые требования протокола доставки
Сеансовый уровень	Письмо опущено в почтовый ящик. Выбрана служба доставки (письмо можно было бы запечатать в бутылку и бросить в реку, но избрана другая служба)
Транспортный уровень	Письмо доставлено на почтамт. Оно отделено от писем, с доставкой которых местная почтовая служба справилась бы самостоятельно
Сетевой уровень	После сортировки письмо уложено в мешок. Появилась новая единица доставки — мешок
Уровень соединения	Мешки писем уложены в вагон. Появилась новая единица доставки — вагон
Физический уровень	Вагон прицеплен к локомотиву. Появилась новая единица доставки — состав. За доставку взялось другое ведомство, действующее по другим протоколам

Из таблицы видно, что каждый новый уровень все больше и больше увеличивает функциональность системы связи. Местная почтовая служба работает не только с письмами, но и с бандеролями и посылками. Почтово-багажная служба занимается еще и доставкой грузов. Вагоны перевозят не только почту, но и людей. По рельсам ходят не только почтово-пассажирские поезда, но и грузовые составы и т. д. То есть, чем выше уровень в модели связи, тем больше различных функциональных служб его используют.

Возвращаясь к системам компьютерной связи, рассмотрим, как в модели *ISO/OSI* происходит обмен данными между пользователями, находящимися на разных континентах.

1. На *прикладном уровне* с помощью специальных приложений пользователь создает документ (сообщение, рисунок и т. п.).
2. На *уровне представления* операционная система его компьютера фиксирует, где находятся созданные данные (в оперативной памяти, в файле на жестком диске и т. п.), и обеспечивает взаимодействие со следующим уровнем.
3. На *сеансовом уровне* компьютер пользователя взаимодействует с локальной или глобальной сетью. Протоколы этого уровня проверяют права пользователя на «выход в эфир» и передают документ к протоколам транспортного уровня.

4. На *транспортном уровне* документ преобразуется в ту форму, в которой положено передавать данные в используемой сети. Например, он может нарезаться на небольшие пакеты стандартного размера.
5. *Сетевой уровень* определяет маршрут движения данных в сети. Так, например, если на транспортном уровне данные были «нарезаны» на пакеты, то на сетевом уровне каждый пакет должен получить адрес, по которому он должен быть доставлен независимо от прочих пакетов.
6. *Уровень соединения* необходим для того, чтобы промодулировать сигналы, циркулирующие на физическом уровне, в соответствии с данными, полученными с сетевого уровня. Например, в компьютере эти функции выполняет сетевая карта или модем.
7. Реальная передача данных происходит на *физическом уровне*. Здесь нет ни документов, ни пакетов, ни даже байтов — только биты, то есть элементарные единицы представления данных. Восстановление документа из них произойдет постепенно, при переходе с нижнего на верхний уровень на компьютере клиента.

   Средства физического уровня лежат за пределами компьютера. В локальных сетях это оборудование самой сети. При удаленной связи с использованием телефонных модемов это линии телефонной связи, коммутационное оборудование телефонных станций и т. п.

На компьютере получателя информации происходит обратный процесс преобразования данных от битовых сигналов до документа.

Особенности виртуальных соединений. Разные уровни протоколов сервера и клиента не взаимодействуют друг с другом напрямую, но они взаимодействуют через физический уровень. Постепенно переходя с верхнего уровня на нижний, данные непрерывно преобразуются, «обрастают» дополнительными данными, которые анализируются протоколами соответствующих уровней на сопредельной стороне. Это и создает эффект виртуального взаимодействия уровней между собой. Однако, несмотря на виртуальность, это все-таки соединения, через которые тоже проходят данные.

Это очень важный момент с точки зрения компьютерной безопасности. Одновременно с теми запросами на поставку данных, которые клиент направляет серверу, передается масса служебной информации, которая может быть как желательной, так и нежелательной. Например, обязательно передаются данные о текущем адресе клиента, о дате и времени запроса, о версии его операционной системы, о его правах доступа к запрашиваемым данным и прочее. Передается и немало косвенной информации, например о том, по какому адресу он посылал предыдущий запрос. Известны случаи, когда даже передавались идентификационные коды процессоров компьютеров.

На использовании виртуальных соединений основаны такие позитивные свойства электронных систем связи, как возможность работать по одному физическому каналу сразу с несколькими серверами. Но на них же основаны и такие негативные средства, как «троянские программы». Троянская программа — разновидность

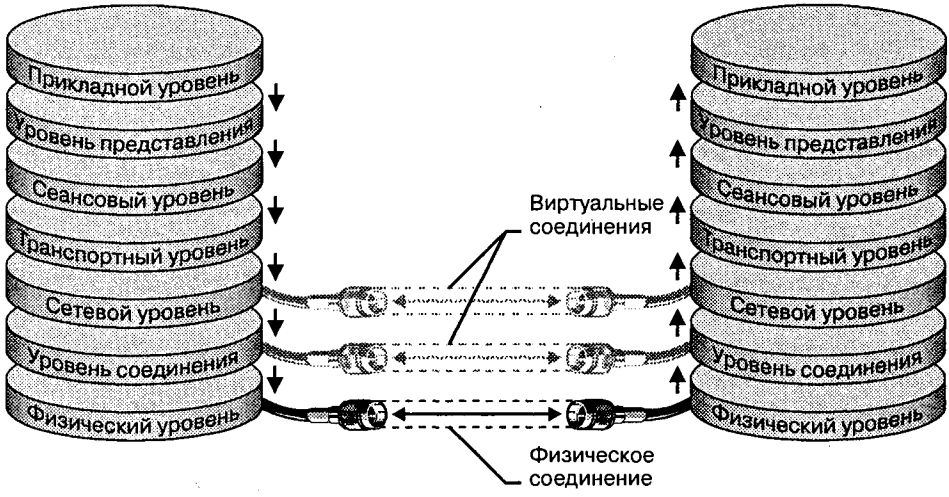


Рис. 8.2. Простейшая модель службы передачи сообщений

«компьютерного вируса», создающая во время сеансов связи виртуальные соединения для передачи данных о компьютере, на котором установлена. Среди этих данных может быть парольная информация, информация о содержании жесткого диска и т. п. В отличие от обычных компьютерных вирусов троянские программы не производят разрушительных действий на компьютере и потому лучше маскируются.

Сетевые службы. На виртуальных соединениях основаны все службы современного Интернета. Так, например, пересылка сообщения от сервера к клиенту может проходить через десятки различных компьютеров. Это совсем не означает, что на каждом компьютере сообщение должно пройти через все уровни, — ему достаточно «подняться» до сетевого уровня, (определяющего адресацию) при приеме и вновь «опуститься» до физического уровня при передаче. В данном случае служба передачи сообщений основывается на виртуальном соединении сетевого уровня и соответствующих ему протоколах (рис. 8.2).

8.2. Интернет. Основные понятия

В дословном переводе на русский язык *интернет* — это *межсеть*, то есть в узком смысле слова интернет — это объединение сетей. Однако в 90-е годы XX века у этого слова появился и более широкий смысл: Всемирная компьютерная сеть. Интернет можно рассматривать в физическом смысле как несколько миллионов компьютеров, связанных друг с другом всевозможными линиями связи, однако такой «физический» взгляд на Интернет слишком узок. Лучше рассматривать Интернет как некое информационное пространство.

Интернет — это не совокупность прямых соединений между компьютерами. Так, например, если два компьютера, находящиеся на разных континентах, обмениваются данными в Интернете, это совсем не значит, что между ними действует одно

прямое или виртуальное соединение. Данные, которые они посылают друг другу, разбиваются на пакеты, и даже в одном сеансе связи разные пакеты одного сообщения могут пройти разными маршрутами. Какими бы маршрутами ни двигались пакеты данных, они все равно достигнут пункта назначения и будут собраны вместе в цельный документ. При этом данные, отправленные позже, могут приходить раньше, но это не помешает правильно собрать документ, поскольку каждый пакет имеет свою маркировку.

Таким образом, Интернет представляет собой как бы «пространство», внутри которого осуществляется непрерывная циркуляция данных. В этом смысле его можно сравнить с теле- и радиоэфиром, хотя есть очевидная разница хотя бы в том, что в эфире никакая информация храниться не может, а в Интернете она перемещается между компьютерами, составляющими *узлы сети*, и какое-то время хранится на их жестких дисках.

Краткая история Интернета

Ранние эксперименты по передаче и приему информации с помощью компьютеров начались еще в 50-х годах и имели лабораторный характер. В США решение о создании первой глобальной сети национального масштаба было принято в 1958 году. Оно стало реакцией на запуск в СССР первого искусственного спутника Земли.

Поводом для создания глобальной компьютерной сети стала разработка Пентагоном глобальной системы раннего оповещения о пусках ракет (*NORAD — North American Aerospace Defense Command*). Станции системы *NORAD* протянулись через север Канады от Аляски до Гренландии, а подземный командный центр расположился вблизи города Колорадо-Спрингс в недрах горы Шайенн. Центр управления был введен в действие в 1964 году, и, собственно, с этого времени можно говорить о работе первой глобальной компьютерной сети, хотя и ведомственной. С середины 60-х годов к ней стали подключаться авиационные, метеорологические и другие военные и гражданские службы.

Курированием работы сети занималась специальная организация — Управление перспективных разработок министерства обороны США (*DARPA — Defense Advanced Research Project Agency*). Основным недостатком централизованной сети была недостаточная устойчивость, связанная с тем, что при выходе из строя какого-либо из узлов полностью выходил из строя и весь сектор, находившийся за ним, а при выходе из строя центра управления выходила из строя вся сеть. Во времена ядерного противостояния сверхдержав этот недостаток был критичным.

Решение проблемы устойчивости и надежности сети было поручено управлению *DARPA*. Основными направлениями исследований стали поиск новых протоколов обслуживания сети и новых принципов сетевой архитектуры. Полигоном для испытаний новых принципов стали крупнейшие университетские и научные центры США, между которыми были проложены линии компьютерной связи. Со стороны министерства обороны работы курировались тем же управлением *DARPA*, и первая вневедомственная национальная компьютерная сеть получила название *ARPANET*. Ее внедрение состоялось в 1969 году.

В 70-е годы сеть *ARPANET* развивалась медленно. В основном развитие происходило за счет подключения региональных сетей, воссоздающих общую архитектуру *ARPANET* на более низком уровне (в региональном или локальном масштабе). Основной объявленной задачей *ARPANET* стала координация групп коллективов, работающих над едиными научно-техническими проектами, а основным назначением стал обмен электронной почтой и файлами с научной и проектно-конструкторской документацией. В то же время не прекращались работы над основной необъявленной задачей — разработкой новых сетевых протоколов, способных обеспечить живучесть глобальной сети даже в ядерном конфликте.

Всякий раз, когда мы говорим о вычислительной технике, нам надо иметь в виду принцип единства аппаратного и программного обеспечения. Пока глобальное расширение *ARPANET* происходило за счет механического подключения все новых и новых аппаратных средств (узлов и сетей), до Интернета в современном понимании этого слова было еще очень далеко.

Второй датой рождения Интернета принято считать 1983 год. В этом году произошли революционные изменения в программном обеспечении компьютерной связи. Проблема устойчивости глобальной сети была решена внедрением протокола *TCP/IP*, лежащего в основе всемирной сети по нынешний день. Решив, наконец, эту задачу, управление *DARPA* прекратило свое участие в проекте и передало управление сетью Национальному научному фонду (*NSF*), который в США играет роль нашей Академии наук. Так в 1983 году образовалась глобальная сеть *NSFNET*. В середине 80-х к ней начали активно подключаться академические и научные сети других стран, например академическая сеть Великобритании *JANET* (*Joint Academic Network*).

Годы, когда глобальной сетью руководил Национальный научный фонд США, вошли в историю как эпоха решительной борьбы с попытками коммерциализации сети. Сеть финансировалась на правительственные средства. Национальный научный фонд распределял их между узлами и материально наказывал тех, кто пытался иметь от сети побочные доходы. В то же время, развитие сети после внедрения протокола *TCP/IP* значительно ускорилось, *NSF* уже не успевал отслеживать деятельность каждого узла, а с подключением иностранных секторов его роль стала чисто символической.

Во второй половине 80-х годов произошло деление всемирной сети на домены по принципу принадлежности. Домен gov финансировался на средства правительства, домен sci — на средства научных кругов, домен edu — на средства системы образования, а домен com (коммерческий) не финансировался никем, то есть его узлы должны были развиваться за счет собственных ресурсов. Национальные сети других государств стали рассматриваться как отдельные домены, например uk — домен Великобритании, su — домен Советского Союза, ru — домен России.

Когда во второй половине 80-х годов сложилась и заработала система доменных имен (*DNS, Domain Name System*), Национальный научный фонд США утратил контроль над развитием сети. Тогда и появилось понятие *Интернета* как саморазвивающейся децентрализованной иерархической структуры. Если во времена *ARPANET* и *NSFNET* сеть финансировалась сверху вниз, то теперь она финансируется от периферии, снизу вверх — от конечных пользователей к владельцам опорных сетей.

Основы функционирования Интернета

В техническом понимании *TCP/IP* — это не один сетевой протокол, а два протокола, лежащих на разных уровнях (это так называемый *стек протоколов*). Протокол *TCP* — протокол *транспортного уровня*. Он управляет тем, *как происходит передача информации*. Протокол *IP* — *адресный*. Он принадлежит *сетевому уровню* и определяет, *куда происходит передача*.

Протокол TCP. Согласно протоколу *TCP*, отправляемые данные «нарезаются» на небольшие пакеты, после чего каждый пакет маркируется таким образом, чтобы в нем были данные, необходимые для правильной сборки документа на компьютере получателя.

Для понимания сути протокола *TCP* можно представить игру в шахматы по переписке, когда двое участников разыгрывают одновременно десяток партий. Каждый ход записывается на отдельной открытке с указанием номера партии и номера хода. В этом случае между двумя партнерами через один и тот же почтовый канал работает как бы десяток соединений (по одному на партию). Два компьютера, связанные между собой одним физическим соединением, могут точно так же поддерживать одновременно несколько *TCP*-соединений. Так, например, два промежуточных сетевых сервера могут одновременно по одной линии связи передавать друг другу в обе стороны множество *TCP*-пакетов от многочисленных клиентов.

Когда мы работаем в Интернете, то по одной-единственной телефонной линии можем одновременно принимать документы из Америки, Австралии и Европы. Пакеты каждого из документов поступают порознь, с разделением во времени, и по мере поступления собираются в разные документы.

Протокол IP. Теперь рассмотрим адресный протокол – *IP* (*Internet Protocol*). Его суть состоит в том, что у каждого участника Всемирной сети должен быть свой уникальный адрес (*IP-адрес*). Без этого нельзя говорить о точной доставке *TCP*-пакетов на нужное рабочее место. Этот адрес выражается очень просто — четырьмя байтами, например: 195.38.46.11. Структуру *IP*-адреса мы рассматривать в этом пособии не будем, но она организована так, что каждый компьютер, через который проходит какой-либо *TCP*-пакет, может по этим четырем числам определить, кому из ближайших «соседей» надо переслать пакет, чтобы он оказался «ближе» к получателю. В результате конечного числа перебросок *TCP*-пакет достигает адресата.

Выше мы не случайно взяли в кавычки слово «ближе». В данном случае оценивается не географическая «близость». В расчет принимаются условия связи и пропускная способность линии. Два компьютера, находящиеся на разных континентах, но связанные высокопроизводительной линией космической связи, считаются более «близкими» друг к другу, чем два компьютера из соседних поселков, связанные простым телефонным проводом. Решением вопросов, что считать «ближе», а что «дальше», занимаются специальные средства — *маршрутизаторы*. Роль маршрутизатора в сети может выполнять как специализированный компьютер, так и специальная программа, работающая на узловом сервере сети.

Поскольку один байт содержит до 256 различных значений, то теоретически с помощью четырех байтов можно выразить более четырех миллиардов уникальных *IP*-

адресов (256^4 за вычетом некоторого количества адресов, используемых в качестве служебных). На практике же из-за особенностей адресации к некоторым типам локальных сетей количество возможных адресов составляет порядка двух миллиардов, но и это по современным меркам достаточно большая величина.

Службы Интернета

Когда говорят о работе в Интернете или об использовании Интернета, то на самом деле речь идет не об Интернете в целом, а только об одной или нескольких из его многочисленных служб. В зависимости от конкретных целей и задач клиенты Сети используют те службы, которые им необходимы.

В простейшем понимании *служба* — это пара программ, взаимодействующих между собой согласно определенным правилам, называемым *протоколами*. Одна из программ этой пары называется *сервером*, а вторая — *клиентом*. Соответственно, когда говорят о работе служб Интернета, речь идет о взаимодействии серверного оборудования и программного обеспечения с клиентским оборудованием и программным обеспечением.

Разные службы имеют разные протоколы. Они называются *прикладными протоколами*. Их соблюдение обеспечивается и поддерживается работой специальных программ. Таким образом, чтобы воспользоваться какой-то из служб Интернета, необходимо установить на компьютере программу, способную работать по протоколу данной службы. Такие программы называют *клиентскими* или просто *клиентами*.

Так, например, для передачи файлов в Интернете используется специальный прикладной протокол *FTP* (*File Transfer Protocol*). Соответственно, чтобы получить из Интернета файл, необходимо:

- иметь на компьютере программу, являющуюся клиентом *FTP* (*FTP-клиент*);
- установить связь с сервером, предоставляющим услуги *FTP* (*FTP-сервером*).

Другой пример: чтобы воспользоваться электронной почтой, необходимо соблюсти протоколы отправки и получения сообщений. Для этого надо иметь программу (*почтовый клиент*) и установить связь с *почтовым сервером*. Так же обстоит дело и с другими службами.

Терминальный режим. Исторически одной из ранних является служба удаленного управления компьютером *Telnet*. Подключившись к удаленному компьютеру по протоколу этой службы, можно управлять его работой. Такое управление еще называют *консольным* или *терминальным*. В прошлом эту службу широко использовали для проведения сложных математических расчетов на удаленных вычислительных центрах. Так, например, если для очень сложных вычислений на персональном компьютере требовались недели непрерывной работы, а на удаленной супер-ЭВМ всего несколько минут, то персональный компьютер применяли для удаленного ввода данных в ЭВМ и для приема полученных результатов.

В наши дни в связи с быстрым увеличением мощности персональных компьютеров необходимость в подобной услуге сократилась, но, тем не менее, службы *Telnet* в Интернете продолжают существовать. Часто протоколы *Telnet* применяют для

дистанционного управления техническими объектами, например телескопами, видеокамерами, промышленными роботами.

Каждый сервер, предоставляющий *Telnet*-услуги, обычно предлагает свое клиентское приложение. Его надо получить по сети (например, по протоколу *FTP*, см. ниже), установить на своем компьютере, подключиться к серверу и работать с удаленным оборудованием. Простейший клиент *Telnet* входит в состав операционной системы *Windows XP* (файл telnet.exe).

Электронная почта (E-Mail). Эта служба также является одной из наиболее ранних. Ее обеспечением в Интернете занимаются специальные *почтовые серверы*. Обратите внимание на то, что когда мы говорим о каком-либо сервере, не имеется в виду, что это специальный выделенный компьютер. Здесь и далее под *сервером* может пониматься программное обеспечение. Таким образом, один узловой компьютер Интернета может выполнять функции нескольких серверов и обеспечивать работу различных служб, оставаясь при этом универсальным компьютером, на котором можно выполнять и другие задачи, характерные для средств вычислительной техники.

Почтовые серверы получают сообщения от клиентов и пересылают их по цепочке к почтовым серверам адресатов, где эти сообщения накапливаются. При установлении соединения между адресатом и его почтовым сервером происходит автоматическая передача поступивших сообщений на компьютер адресата.

Почтовая служба основана на двух прикладных протоколах: *SMTP* и *POP3*. По первому происходит отправка корреспонденции с компьютера на сервер, а по второму — прием поступивших сообщений. Существует большое разнообразие клиентских почтовых программ. К ним относится, например, программа *Microsoft Outlook Express*, входящая в состав операционной системы *Windows XP* как стандартная. Более мощная программа, интегрирующая в себе кроме поддержки электронной почты и другие средства делопроизводства, *Microsoft Outlook*, входит в состав известного пакета *Microsoft Office XP*. Из специализированных почтовых программ хорошую популярность имеют программы *The Bat!* и *Eudora Pro*.

Списки рассылки (Mail List). Обычная электронная почта предполагает наличие двух партнеров по переписке. Если же партнеров нет, то достаточно большой поток почтовой информации в свой адрес можно обеспечить, подписавшись на *списки рассылки*. Это специальные тематические серверы, собирающие информацию по определенным темам и переправляющие ее подписчикам в виде сообщений электронной почты.

Темами списков рассылки может быть что угодно, например вопросы, связанные с изучением иностранных языков, научно-технические обзоры, презентация новых программных и аппаратных средств вычислительной техники (рис. 8.3). Большинство телекомпаний создают списки рассылки на своих узлах, через которые рассылают клиентам аннотированные обзоры телепрограмм. Списки рассылки позволяют эффективно решать вопросы регулярной доставки данных.

Служба телеконференций (Usenet). Служба телеконференций похожа на циркулярную рассылку электронной почты, в ходе которой одно сообщение отправляется

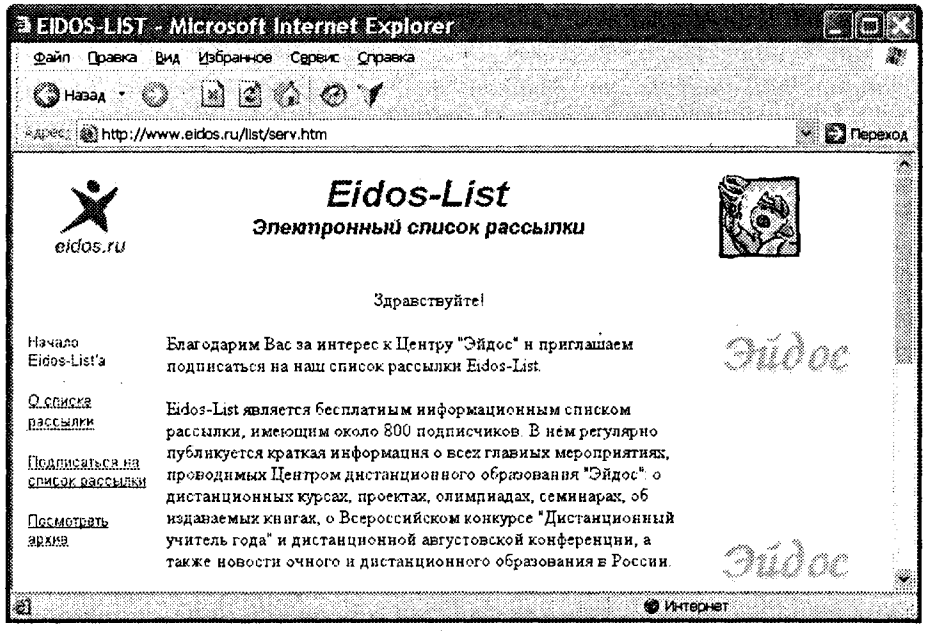


Рис. 8.3. Список рассылки, посвященный вопросам дистанционного образования

не одному корреспонденту, а большой группе (такие группы называются *телеконференциями* или *группами новостей*).

Обычное сообщение электронной почты пересылается по узкой цепочке серверов от отправителя к получателю. При этом не предполагается его хранение на промежуточных серверах. Сообщения, направленные на сервер группы новостей, отправляются с него на все серверы, с которыми он связан, если на них данного сообщения еще нет. Далее процесс повторяется. Характер распространения каждого отдельного сообщения напоминает лесной пожар.

На каждом из серверов поступившее сообщение хранится ограниченное время (обычно неделю), и все желающие могут в течение этого времени с ним ознакомиться. Распространяясь во все стороны, менее чем за сутки сообщения охватывают весь земной шар. Далее распространение затухает, поскольку на сервер, который уже имеет данное сообщение, повторная передача производиться не может.

Ежедневно в мире создается порядка миллиона сообщений для групп новостей. Выбрать в этом массиве действительно полезную информацию практически невозможно. Поэтому вся система телеконференций разбита на тематические группы. Сегодня в мире насчитывают порядка 100 000 тематических групп новостей. Они охватывают большинство тем, интересующих массы. Особой популярностью пользуются группы, посвященные вычислительной технике.

Основной прием использования групп новостей состоит в том, чтобы задать вопрос, обращаясь ко всему миру, и получить ответ или совет от тех, кто с этим вопросом уже разобрался. При этом важно следить за тем, чтобы содержание вопроса соот-

ветствовало теме данной телеконференции. Многие квалифицированные специалисты мира (конструкторы, инженеры, ученые, врачи, педагоги, юристы, писатели, журналисты, программисты и прочие) регулярно просматривают сообщения телеконференций, проходящие в группах, касающихся их сферы деятельности. Такой просмотр называется *мониторингом информации*. Регулярный мониторинг позволяет специалистам точно знать, что нового происходит в мире по их специальности, какие проблемы беспокоят большие массы людей и на что надо обратить особое внимание в своей работе.

В современных промышленных и проектно-конструкторских организациях считается хорошим тоном, если специалисты высшего эшелона периодически (один-два раза в месяц) отвечают через систему телеконференций на типовые вопросы пользователей своей продукции. Так, например, в телеконференциях, посвященных легковым автомобилям, нередко можно найти сообщения от главных конструкторов крупнейших промышленных концернов.

При отправке сообщений в телеконференции принято указывать свой адрес электронной почты для обратной связи. В тех случаях, когда есть угроза переполнения электронного «почтового ящика» корреспонденцией, не относящейся к непосредственной производственной деятельности, вместо основного адреса, используемого для деловой переписки, указывают дополнительный адрес. Как правило, такой адрес арендуют на сервере одной из бесплатных анонимных почтовых служб, например www.hotmail.com.

Огромный объем сообщений в группах новостей значительно затрудняет их целенаправленный мониторинг, поэтому в некоторых группах производится предварительный «отсев» бесполезной информации (в частности, рекламной), не относящейся к теме конференции. Такие конференции называют *модерируемыми*. В качестве *модератора* может выступать не только человек, но и программа, фильтрующая сообщения по определенным ключевым словам. В последнем случае говорят об *автоматической модерации*.

Для работы со службой телеконференций существуют специальные клиентские программы. Так, например, приложение *Microsoft Outlook Express*, указанное выше как почтовый клиент, позволяет работать также и со службой телеконференций. Для начала работы надо настроить программу на взаимодействие с сервером групп новостей, оформить «подписку» на определенные группы и периодически, как и электронную почту, получать все сообщения, проходящие по теме этой группы. В данном случае слово «подписка» не предполагает со стороны клиента никаких обязательств или платежей — это просто указание серверу о том, что сообщения по указанным темам надо доставлять, а по прочим — нет. Отменить подписку или изменить ее состав можно в любой удобный момент.

Служба World Wide Web (WWW). Безусловно, это самая популярная служба современного Интернета. Ее нередко отождествляют с Интернетом, хотя на самом деле это лишь одна из его многочисленных служб.

World Wide Web — это единое информационное пространство, состоящее из сотен миллионов взаимосвязанных электронных документов, хранящихся на *Web-сер-*

верах. Отдельные документы, составляющие *пространство Web*, называют *Web-страницами*. Количество существующих *Web*-страниц уже измеряется миллиардами, причем энергичный рост объема *World Wide Web* продолжается.

Группы тематически объединенных *Web*-страниц называют *Web-узлами* (альтернативный термин — *Web-сайт* или просто *сайт*). Один физический *Web*-сервер может содержать достаточно много *Web*-узлов, каждому из которых, как правило, отводится отдельный каталог на жестком диске сервера.

От обычных текстовых документов *Web*-страницы отличаются тем, что они оформлены без привязки к конкретному носителю. Например, оформление документа, напечатанного на бумаге, привязано к параметрам печатного листа, который имеет определенную ширину, высоту и размеры полей. Электронные *Web*-документы предназначены для просмотра на экране компьютера, причем заранее не известно, на каком. Не известны ни размеры экрана, ни параметры цветового и графического разрешения, не известна даже операционная система, с которой работает компьютер клиента. Поэтому *Web*-документы не могут иметь «жесткого» форматирования. Оформление выполняется непосредственно во время их воспроизведения на компьютере клиента и происходит оно в соответствии с настройками программы, выполняющей просмотр.

Программы для просмотра *Web*-страниц называют *браузерами*. В период «неустойчивости» терминологии применялись также термины *броузер* или *обозреватель*, которые еще можно встретить в литературе. Во всех случаях речь идет о некотором *средстве просмотра Web-документов*.

Браузер выполняет отображение документа на экране, руководствуясь командами, которые автор документа внедрил в его текст (если автор применяет автоматические средства подготовки *Web*-документов, необходимые команды внедряются автоматически). Такие команды называются *тегами*. От обычного текста они отличаются тем, что заключены в угловые скобки. Большинство тегов используются парами: *открывающий* тег и *закрывающий*. Закрывающий тег начинается с символа «/».

```
<CENTER> Этот текст должен выравниваться по центру экрана </CENTER>

<LEFT> Этот текст выравнивается по левой границе экрана </LEFT>

<RIGHT> Этот текст выравнивается по правой границе экрана </RIGHT>
```

Сложные теги имеют кроме *ключевого слова* дополнительные *атрибуты* и *параметры*, детализирующие способ их применения. Правила записи тегов содержатся в спецификации особого *языка разметки*, близкого к языкам программирования. Он называется *языком разметки гипертекста — HTML (HyperText Markup Language)*. Таким образом, *Web*-документ представляет собой обычный текстовый документ, размеченный тегами *HTML*. Такие документы также называют *HTML-документами* или *документами в формате HTML*.

При отображении *HTML*-документа на экране с помощью браузера теги не показываются, и мы видим только текст, составляющий документ. Однако оформление этого текста (выравнивание, цвет, размер и начертание шрифта и прочее) выполняется в соответствии с тем, какие теги имплантированы в текст документа.

Существуют специальные теги для внедрения графических и мультимедийных объектов (звук, музыка, видеоклипы). Встретив такой тег, браузер делает запрос к серверу на доставку файла, связанного с тегом, и воспроизводит его в соответствии с заданными атрибутами и параметрами тега — мы видим иллюстрацию или слышим звук. Более подробно вопросы создания *Web*-страниц и использования тегов *HTML* рассмотрены в главе «Подготовка и публикация *Web*-документов».

В последние годы в *Web*-документах находят широкое применение так называемые *активные компоненты*. Это тоже объекты, но они содержат не только текстовые, графические и мультимедийные данные, но и программный код, то есть могут не просто отображаться на компьютере клиента, но и выполнять на нем работу по заложенной в них программе. Для того чтобы активные компоненты не могли выполнить на чужом компьютере разрушительные операции (что характерно для «компьютерных вирусов»), они исполняются только под контролем со стороны браузера. Браузер не должен допустить исполнения команд, несущих потенциальную угрозу: например, он пресекает попытки осуществить операции с жестким диском.

Возможность внедрения в текст графических и других объектов, реализуемая с помощью тегов *HTML*, является одной из самых эффектных с точки зрения оформления *Web*-страниц, но не самой важной с точки зрения самой идеи *World Wide Web*. Наиболее важной чертой *Web*-страниц, реализуемой с помощью тегов *HTML*, являются *гипертекстовые ссылки*. С любым фрагментом текста или, например, с рисунком с помощью тегов можно связать иной *Web*-документ, то есть установить *гиперссылку*. В этом случае при щелчке левой кнопкой мыши на тексте или рисунке, являющемся гиперссылкой, отправляется запрос на доставку нового документа. Этот документ, в свою очередь, тоже может иметь гиперссылки на другие документы.

Тем самым, совокупность огромного числа гипертекстовых электронных документов, хранящихся на серверах *WWW*, образует своеобразное *гиперпространство документов*, между которыми возможно перемещение. Произвольное перемещение между документами в *Web*-пространстве называют *Web-серфингом* (выполняется с целью ознакомительного просмотра). Целенаправленное перемещение между *Web*-документами называют *Web-навигацией* (выполняется с целью поиска нужной информации).

Гипертекстовая связь между сотнями миллионов документов, хранящихся на физических серверах Интернета, является основой существования логического пространства *World Wide Web*. Однако такая связь не могла бы существовать, если бы каждый документ в этом пространстве не обладал своим уникальным адресом. Выше мы говорили, что каждый файл одного локального компьютера обладает *уникальным полным именем*, в которое входит собственное имя файла (включая расширение имени) и путь доступа к файлу, начиная от имени устройства, на котором он хранится. Теперь мы можем расширить представление об уникальном имени файла и развить его до Всемирной сети. Адрес любого файла во всемирном масштабе определяется *унифицированным указателем ресурса — URL*.

Адрес *URL* состоит из трех частей.

1. Указание службы, которая осуществляет доступ к данному ресурсу (обычно обозначается именем прикладного протокола, соответствующего данной

службе). Так, например, для службы *WWW* прикладным является протокол *HTTP* (*HyperText Transfer Protocol — протокол передачи гипертекста*). После имени протокола ставится двоеточие (:) и два знака «/» (косая черта):

http://...

2. Указание *доменного имени* компьютера (сервера), на котором хранится данный ресурс:

http://**www.abcde.com**...

3. Указания полного пути доступа к файлу на данном компьютере. В качестве разделителя используется символ «/» (косая черта):

http://www.abcde.com/**Files/New/abcdefg.zip**

При записи *URL*-адреса важно точно соблюдать регистр символов. В отличие от правил работы в *MS-DOS* и *Windows*, в Интернете строчные и прописные символы в именах файлов и каталогов считаются разными.

Именно в форме *URL* и связывают адрес ресурса с гипертекстовыми ссылками на *Web*-страницах. При щелчке на гиперссылке браузер посылает запрос для поиска и доставки ресурса, указанного в ссылке. Если по каким-то причинам он не найден, выдается сообщение о том, что ресурс недоступен (возможно, что сервер временно отключен или изменился адрес ресурса).

Служба имен доменов (DNS). Когда мы говорили о протоколах Интернета, то сказали, что адрес любого компьютера или любой локальной сети в Интернете может быть выражен четырьмя байтами, например так:

195.28.132.97

А только что мы заявили, что каждый компьютер имеет уникальное доменное имя, например такое:

www.abcdef.com

Нет ли здесь противоречия?

Противоречия здесь нет, поскольку это просто две разные формы записи адреса одного и того же *сетевого компьютера*. Человеку неудобно работать с числовым представлением *IP*-адреса, зато доменное имя запоминается легко, особенно если учесть, что, как правило, это имя имеет содержание. Например, *Web*-сервер компании *Microsoft* имеет имя www.microsoft.com, а *Web*-сервер компании «Космос ТВ» имеет имя www.kosmostv.ru (суффикс .ru в конце имени говорит о том, что сервер компании принадлежит российскому сектору Интернета). Нетрудно «реконструировать» и имена для других компаний.

С другой стороны, автоматическая работа серверов сети организована с использованием четырехзначного числового адреса. Благодаря ему промежуточные серверы могут осуществлять передачу запросов и ответов в нужном направлении, не зная, где конкретно находятся отправитель и получатель. Поэтому необходим перевод доменных имен в связанные с ними *IP*-адреса. Этим и занимаются серверы службы имен доменов *DNS*. Наш запрос на получение одной из страниц сервера www.abcde.com

сначала обрабатывается сервером *DNS*, и далее он направляется по *IP*-адресу, а не по доменному имени.

Служба передачи файлов (FTP). Прием и передача файлов составляют значительный процент от прочих Интернет-услуг. Необходимость в передаче файлов возникает, например, при приеме файлов программ, при пересылке крупных документов (например, книг), а также при передаче архивных файлов, в которых запакованы большие объемы информации.

Служба *FTP* имеет свои серверы в мировой сети, на которых хранятся архивы данных. Со стороны клиента для работы с серверами *FTP* может быть установлено специальное программное обеспечение, хотя в большинстве случаев браузеры *WWW* обладают встроенными возможностями для работы и по протоколу *FTP*.

Протокол *FTP* работает одновременно с двумя *TCP*-соединениями между сервером и клиентом. По одному соединению идет передача данных, а второе соединение используется как управляющее. Протокол *FTP* также предоставляет серверу средства для идентификации обратившегося клиента. Этим часто пользуются коммерческие серверы и серверы ограниченного доступа, поставляющие информацию только зарегистрированным клиентам, — они выдают запрос на ввод имени пользователя и связанного с ним пароля. Однако существуют и десятки тысяч *FTP*-серверов с *анонимным доступом* для всех желающих. В этом случае в качестве имени пользователя надо ввести слово: anonymous, а в качестве пароля задать адрес электронной почты. В большинстве случаев программы-клиенты *FTP* делают это автоматически.

IRC. Служба *IRC* (*Internet Relay Chat*) предназначена для прямого общения нескольких человек в режиме реального времени. Иногда службу *IRC* называют *чат-конференциями* или просто *чатом*. В отличие от системы телеконференций, в которой общение между участниками обсуждения темы открыто всему миру, в системе *IRC* общение происходит только в пределах одного *канала*, в работе которого принимают участие обычно лишь несколько человек. Каждый пользователь может создать собственный канал и пригласить в него участников «беседы» или присоединиться к одному из открытых в данный момент каналов.

Существует несколько популярных клиентских программ для работы с серверами и сетями, поддерживающими сервис *IRC*. Одна из наиболее популярных — программа mIRC.exe.

ICQ. Эта служба — одна из нескольких существующих в Интернете служб для мгновенного обмена сообщениями. Если два человека подключены к Интернету одновременно, то, в принципе, им почти ничто не мешает общаться друг с другом напрямую. Единственная проблема — знание сетевого *IP*-адреса человека, подключенного в данный момент к Интернету. Большинство пользователей не имеет постоянного *IP*-адреса — такой адрес выдается им на временной основе в момент установки соединения. Название службы *ICQ* является акронимом выражения *I seek you — я тебя ищу*. Для пользования этой службой надо зарегистрироваться на ее центральном сервере (http://www.icq.com) и получить персональный идентификационный номер *UIN* (*Universal Internet Number*). Данный номер можно сообщить партнерам по контактам. Зная номер *UIN* партнера, но не зная его текущий *IP*-адрес, можно

через центральный сервер службы отправить ему сообщение с предложением установить соединение.

Как было указано выше, каждый компьютер, подключенный к Интернету, должен иметь четырехзначный *IP*-адрес. Этот адрес может быть *постоянным* или *динамически* временным. Те компьютеры, которые включены в Интернет на постоянной основе, имеют постоянные *IP*-адреса. Большинство же пользователей подключаются к Интернету лишь на время сеанса. Им выдается динамический *IP*-адрес, действующий только в течение данного сеанса. Этот адрес выдает тот сервер, через который происходит подключение. В разных сеансах динамический *IP*-адрес может быть различным, причем заранее не известно, каким.

При каждом подключении к Интернету программа *ICQ*, установленная на нашем компьютере, определяет текущий *IP*-адрес и сообщает его центральной службе, которая, в свою очередь, оповещает наших партнеров по контактам. Далее наши партнеры (если они тоже являются клиентами данной службы) могут установить с нами прямую связь. Программа предоставляет возможность выбора режима связи («готов к контакту»; «прошу не беспокоить, но готов принять срочное сообщение»; «закрыт для контакта» и т. п.). После установления контакта связь происходит в режиме, аналогичном сервису *IRC*.

8.3. Подключение к Интернету

Основные понятия

Для работы в Интернете необходимо:

- физически подключить компьютер к одному из узлов Всемирной сети;
- получить *IP*-адрес на постоянной или временной основе;
- установить и настроить программное обеспечение — программы-клиенты тех служб Интернета, услугами которых предполагается пользоваться.

Организации, предоставляющие возможность подключения к своему узлу и выделяющие *IP*-адреса, называются *поставщиками услуг Интернета* (используется также термин *сервис-провайдер*, или просто *провайдер*). Они оказывают подобную услугу на договорной основе.

Физическое подключение может быть *выделенным* или *коммутируемым*. Для выделенного соединения необходимо, как правило, проложить новую или арендовать готовую физическую линию связи (кабельную, оптоволоконную, радиоканал, спутниковый канал и т. п.). Такое подключение используют организации и предприятия, нуждающиеся в передаче больших объемов данных. От типа линии связи зависит ее *пропускная способность* (измеряется в единицах *бит в секунду*). В настоящее время пропускная способность мощных линий связи (оптоволоконных и спутниковых) составляет сотни мегабит в секунду (Мбит/с).

В противоположность выделенному соединению коммутируемое соединение — временное. Оно не требует специальной линии связи и может быть осуществлено, например, по телефонной линии. Коммутацию (подключение) выполняет автома-

тическая телефонная станция (АТС) по сигналам, выданным в момент набора телефонного номера.

Для телефонных линий связи характерна низкая пропускная способность. В зависимости от того, какое оборудование использовано на станциях АТС по пути следования сигнала, различают *аналоговые* и *цифровые* телефонные линии. Основную часть телефонных линий во многих городах России составляют устаревшие аналоговые линии. Их предельная пропускная способность не превосходит 50 Кбит/с (примерно две страницы текста в секунду или одна-две фотографии стандартного размера в минуту). Пропускная способность цифровых телефонных линий составляет 60–120 Кбит/с, то есть в 2–4 раза выше. По аналоговым телефонным линиям связи можно передавать и видеоинформацию (что используется в видеоконференциях), но размер окна, в котором отображаются видеоданные, обычно невелик (порядка 150×150 точек) и частота смены кадров мала для получения качественного видеоряда (1–2 кадра в секунду). Для сравнения: в обычном телевидении частота кадров — 25 кадров в секунду.

Телефонные линии связи никогда не предназначались для передачи цифровых сигналов — их характеристики подходят только для передачи голоса, причем в достаточно узком диапазоне частот — 300–3000 Гц. Поэтому для передачи цифровой информации несущие сигналы звуковой частоты *модулируют* по амплитуде, фазе и частоте. Такое преобразование выполняет специальное устройство — *модем* (название образовано от слов *модулятор* и *демодулятор*).

Установка модема

По способу подключения различают *внешние* и *внутренние* модемы. Внешние модемы подключают к разъему *последовательного порта*, выведенному на заднюю стенку системного блока. Внутренние модемы устанавливают в один из разъемов расширения материнской платы.

Поток данных, проходящих через модем, очень мал по сравнению с потоками, проходящими через другие устройства компьютера. Поэтому модемы, рассчитанные на подключение в разъем (слот) устаревшей малопроизводительной шины *ISA*, по производительности практически не уступают более современным устройствам. Однако в настоящее время все выпускаемые модели внутренних модемов рассчитаны на подключение к шине *PCI*.

Как и другие устройства компьютера, модем требует не только аппаратной, но и программной установки. В операционной системе *Windows XP* ее можно выполнить стандартными средствами Пуск ▸ Настройка ▸ Панель управления ▸ Установка оборудования, хотя для модемов есть и специальное средство: Пуска ▸ Настройка ▸ Панель управления ▸ Телефон и модем ▸ Модемы ▸ Добавить.

Для модемов, подключаемых к шине *PCI*, проблем с установкой обычно не возникает, поскольку они соответствуют стандарту на *самоустанавливающееся оборудование* (*plug-and-play*). Модемы, подключаемые к шине *ISA* (как и другие устройства, подключаемые к этой шине), не всегда являются самоустанавливающимися, и операционная система может некорректно выполнять их автоматическую про-

граммную установку и настройку. Если при этом возникают аппаратные конфликты, они чаще всего приводят к неправильной работе самого модема или мыши. Для устранения конфликта изменяют назначение последовательного порта для мыши и/или модема и повторяют установку. Проверить правильность подключения модема можно командой Пуск ▸ Настройка ▸ Панель управления ▸ Телефон и модем ▸ Модемы ▸ Свойства ▸ Диагностика ▸ Опросить модем.

Подключение к компьютеру поставщика услуг Интернета

Операционная система *Windows XP*, в отличие от предыдущих версий *Windows*, рассматривает все виды соединения компьютера с другими системами одинаково. Прямое соединение с соседним компьютером, подключение к локальной сети, удаленный доступ к Интернету — для *Windows XP* все это всего лишь разные виды *сетевых подключений*.

Для подключения к компьютеру поставщика услуг Интернета создать новое подключение (Мой компьютер ▸ Настройка ▸ Сетевые подключения ▸ Мастер новых подключений). При настройке программы необходимы данные, которые должен сообщить поставщик услуг:

- номер телефона, по которому производится соединение;
- имя пользователя (*login*);
- пароль (*password*);
- *IP*-адрес сервера *DNS*. На всякий случай вводят два адреса — основной и дополнительный, используемый, если основной сервер *DNS* по каким-то причинам временно не работает. В некоторых случаях адрес сервера *DNS* назначается поставщиком услуг автоматически и его указание необязательно.

Этих данных достаточно для подключения к Интернету, хотя при заключении договора с поставщиком услуг можно получить и дополнительную информацию, например номера телефонов службы поддержки. Вводить собственный *IP*-адрес для настройки программы не надо. Сервер поставщика услуг выделит его автоматически на время проведения сеанса работы.

Порядок создания и настройки подключения к Интернету рассмотрен в упражнениях 8.1 и 8.2.

8.4. Вопросы компьютерной безопасности

Понятие о компьютерной безопасности

В вычислительной технике понятие безопасности является весьма широким. Оно подразумевает и надежность работы компьютера, и сохранность ценных данных, и защиту информации от внесения в нее изменений неуполномоченными лицами, и сохранение тайны переписки при электронной связи. Разумеется, во всех цивилизованных странах на страже безопасности граждан стоят законы, но в сфере вычислительной техники правоприменительная практика пока развита недостаточно, а законотворческий процесс не успевает за развитием технологий, поэтому надежность работы компьютерных систем во многом опирается на меры самозащиты.

Компьютерные вирусы

Компьютерный вирус — это программный код, встроенный в другую программу, или в документ, или в определенные области носителя данных и предназначенный для выполнения несанкционированных действий на несущем компьютере.

Основными типами компьютерных вирусов являются:

- программные вирусы;
- загрузочные вирусы;
- макровирусы.

К компьютерным вирусам примыкают и так называемые *троянские кони* (*троянские программы*, *троянцы*).

Программные вирусы. Программные вирусы — это блоки программного кода, целенаправленно внедренные внутрь других прикладных программ. При запуске программы, несущей вирус, происходит запуск имплантированного в нее вирусного кода. Работа этого кода вызывает скрытые от пользователя изменения в файловой системе жестких дисков и/или в содержании других программ. Так, например, вирусный код может воспроизводить себя в теле других программ — этот процесс называется *размножением*. По прошествии определенного времени, создав достаточное количество копий, программный вирус может перейти к разрушительным действиям — нарушению работы программ и операционной системы, удалению информации, хранящейся на жестком диске. Этот процесс называется *вирусной атакой*.

Самые разрушительные вирусы могут инициировать форматирование жестких дисков. Поскольку форматирование диска — достаточно продолжительный процесс, который не должен пройти незамеченным со стороны пользователя, во многих случаях программные вирусы ограничиваются уничтожением данных только в системных секторах жесткого диска, что эквивалентно потере таблиц файловой структуры. В этом случае данные на жестком диске остаются нетронутыми, но воспользоваться ими без применения специальных средств нельзя, поскольку неизвестно, какие сектора диска каким файлам принадлежит. Теоретически восстановить данные в этом случае можно, но трудоемкость этих работ исключительно высока.

Считается, что никакой вирус не в состоянии вывести из строя аппаратное обеспечение компьютера. Однако бывают случаи, когда аппаратное и программное обеспечение настолько взаимосвязаны, что программные повреждения приходится устранять заменой аппаратных средств. Так, например, в большинстве современных материнских плат базовая система ввода-вывода (*BIOS*) хранится в перезаписываемых постоянных запоминающих устройствах (так называемая *флэш-память*). Возможность перезаписи информации в микросхеме флэш-памяти используют некоторые программные вирусы для уничтожения данных *BIOS*. В этом случае для восстановления работоспособности компьютера требуется либо замена микросхемы, хранящей *BIOS*, либо ее перепрограммирование на специальных устройствах, называемых *программаторами*.

Программные вирусы поступают на компьютер при запуске непроверенных программ, полученных на внешнем носителе (гибкий диск, компакт-диск и т. п.) или принятых из Интернета. Особое внимание следует обратить на слова *при запуске*. При обычном копировании зараженных файлов заражение компьютера произойти не может. В связи с этим все данные, принятые из Интернета, должны проходить обязательную проверку на безопасность, а если получены незатребованные данные из незнакомого источника, их следует уничтожать, не рассматривая. Обычный прием распространения «троянских» программ — приложение к электронному письму с «рекомендацией» извлечь и запустить якобы полезную программу.

Загрузочные вирусы. От программных вирусов загрузочные вирусы отличаются методом распространения. Они поражают не программные файлы, а определенные системные области магнитных носителей (гибких и жестких дисков). Кроме того, на включенном компьютере они могут временно располагаться в оперативной памяти.

Обычно заражение происходит при попытке загрузки компьютера с магнитного носителя, системная область которого содержит загрузочный вирус. Так, например, при попытке загрузить компьютер с гибкого диска происходит сначала проникновение вируса в оперативную память, а затем в загрузочный сектор жестких дисков. Далее этот компьютер сам становится источником распространения загрузочного вируса.

Макровирусы. Эта особая разновидность вирусов поражает документы, выполненные в некоторых прикладных программах, имеющих средства для исполнения так называемых *макрокоманд*. В частности, к таким документам относятся документы текстового процессора *Microsoft Word* (они имеют расширение .DOC). Заражение происходит при открытии файла документа в окне программы, если в ней не отключена возможность исполнения макрокоманд. Как и для других типов вирусов, результат атаки может быть как относительно безобидным, так и разрушительным.

Методы защиты от компьютерных вирусов

Существуют три рубежа защиты от компьютерных вирусов:

- предотвращение поступления вирусов;
- предотвращение вирусной атаки, если вирус все-таки поступил на компьютер;
- предотвращение разрушительных последствий, если атака все-таки произошла.

Существуют три метода реализации защиты:

- программные методы защиты;
- аппаратные методы защиты;
- организационные методы защиты.

В вопросе защиты ценных данных часто используют бытовой подход: «болезнь лучше предотвратить, чем лечить». К сожалению, именно он и вызывает наиболее разрушительные последствия. Создав бастионы на пути проникновения вирусов в компьютер, нельзя положиться на их прочность и остаться неготовым к действиям после разрушительной атаки. К тому же вирусная атака — далеко не единственная

и даже не самая распространенная причина утраты важных данных. Существуют программные сбои, которые могут вывести из строя операционную систему, а также аппаратные сбои, способные сделать жесткий диск неработоспособным. Всегда существует вероятность утраты компьютера вместе с ценными данными в результате кражи, пожара или иного стихийного бедствия.

Поэтому создавать систему безопасности следует в первую очередь «с конца» — с предотвращения разрушительных последствий любого воздействия, будь то вирусная атака, кража в помещении или физический выход жесткого диска из строя. Надежная и безопасная работа с данными достигается только тогда, когда любое неожиданное событие, в том числе и полное физическое уничтожение компьютера, не приведет к катастрофическим последствиям.

Средства антивирусной защиты

Основным средством защиты информации является резервное копирование наиболее ценных данных. В случае утраты информации по любой из вышеперечисленных причин жесткие диски переформатируют и подготавливают к новой эксплуатации. На «чистый» отформатированный диск устанавливают операционную систему с дистрибутивного компакт-диска, затем под ее управлением устанавливают все необходимое программное обеспечение, которое тоже берут с дистрибутивных носителей. Восстановление компьютера завершается восстановлением данных, которые берут с резервных носителей.

При резервировании данных следует также иметь в виду и то, что надо отдельно сохранять все регистрационные и парольные данные для доступа к сетевым службам Интернета. Их не следует хранить на компьютере. Обычное место хранения — служебный дневник в сейфе руководителя подразделения.

Создавая план мероприятий по резервному копированию информации, необходимо учитывать, что резервные копии должны храниться отдельно от компьютера. То есть, например, резервирование информации на отдельном жестком диске того же компьютера только создает иллюзию безопасности. Относительно новым и достаточно надежным приемом хранения ценных, но неконфиденциальных данных является их хранение в *Web*-папках на удаленных серверах в Интернете. Есть службы, бесплатно предоставляющие пространство (до нескольких Мбайт) для хранения данных пользователя.

Резервные копии конфиденциальных данных сохраняют на внешних носителях, которые хранят в сейфах, желательно в отдельных помещениях. При разработке организационного плана резервного копирования учитывают необходимость создания не менее двух резервных копий, сохраняемых в разных местах. Между копиями осуществляют *ротацию*. Например, в течение недели ежедневно копируют данные на носители резервного комплекта «А», а через неделю их заменяют комплектом «Б» и т. д.

Вспомогательными средствами защиты информации являются антивирусные программы и средства аппаратной защиты. Так, например, простое отключение перемычки на материнской плате не позволит осуществить стирание перепрограмми-

руемой микросхемы ПЗУ (*флэш-BIOS*), независимо от того, кто будет пытаться это сделать: компьютерный вирус, злоумышленник или неаккуратный пользователь.

Существует достаточно много программных средств антивирусной защиты. Они предоставляют следующие возможности.

1. *Создание образа жесткого диска на внешних носителях* (например, на гибких дисках). В случае выхода из строя данных в системных областях жесткого диска сохраненный «образ диска» может позволить восстановить если не все данные, то по крайней мере их большую часть. Это же средство может защитить от утраты данных при аппаратных сбоях и при неаккуратном форматировании жесткого диска.
2. Регулярное сканирование жестких дисков в поисках компьютерных вирусов. Сканирование обычно выполняется автоматически при каждом включении компьютера и при размещении внешнего диска в считывающем устройстве. При сканировании следует иметь в виду, что антивирусная программа ищет вирус путем сравнения кода программ с кодами известных ей вирусов, хранящимися в базе данных. Если база данных устарела, а вирус является новым, сканирующая программа его не обнаружит. Для надежной работы следует регулярно обновлять антивирусную программу. Желательная периодичность обновления — один раз в две недели; допустимая — один раз в три месяца. Для примера укажем, что разрушительные последствия атаки вируса W95.CIH.1075 («Чернобыль»), вызвавшего уничтожение информации на сотнях тысяч компьютеров 26 апреля 1999 года, были связаны не с отсутствием средств защиты от него, а с длительной задержкой (более года) в обновлении этих средств.
3. Контроль изменения размера и других атрибутов файлов. Поскольку некоторые компьютерные вирусы на этапе размножения изменяют параметры зараженных файлов, контролирующая программа может обнаружить их деятельность и предупредить пользователя.
4. Контроль обращений к жесткому диску. Поскольку наиболее опасные операции, связанные с работой компьютерных вирусов, так или иначе обращены на модификацию данных, записанных на жестком диске, антивирусные программы могут контролировать обращения к нему и предупреждать пользователя о подозрительной активности.

Защита информации в Интернете

При работе в Интернете следует иметь в виду, что насколько ресурсы Всемирной сети открыты каждому клиенту, настолько же и ресурсы его компьютерной системы могут быть при определенных условиях открыты всем, кто обладает необходимыми средствами.

Для частного пользователя этот факт не играет особой роли, но знать о нем необходимо, чтобы не допускать действий, нарушающих законодательства тех стран, на территории которых расположены серверы Интернета. К таким действиям относятся вольные или невольные попытки нарушить работоспособность компьютерных

систем, попытки взлома защищенных систем, использование и распространение программ, нарушающих работоспособность компьютерных систем (в частности, компьютерных вирусов).

Работая во Всемирной сети, следует помнить о том, что абсолютно все действия фиксируются и протоколируются специальными программными средствами и информация как о законных, так и о незаконных действиях обязательно где-то накапливается. Таким образом, к обмену информацией в Интернете следует подходить как к обычной переписке с использованием почтовых открыток. Информация свободно циркулирует в обе стороны, но в общем случае она доступна всем участникам информационного процесса. Это касается всех служб Интернета, открытых для массового использования.

Однако даже в обычной почтовой связи наряду с открытками существуют и почтовые конверты. Использование почтовых конвертов при переписке не означает, что партнерам есть, что скрывать. Их применение соответствует давно сложившейся исторической традиции и устоявшимся морально-этическим нормам общения. Потребность в аналогичных «конвертах» для защиты информации существует и в Интернете. Сегодня Интернет является не только средством общения и универсальной справочной системой — в нем циркулируют договорные и финансовые обязательства, необходимость защиты которых как от просмотра, так и от фальсификации очевидна. Начиная с 1999 года Интернет становится мощным средством обеспечения розничного торгового оборота, а это требует защиты данных кредитных карт и других электронных платежных средств.

Принципы защиты информации в Интернете опираются на определение информации, сформулированное нами в первой главе этого пособия. *Информация — это продукт взаимодействия данных и адекватных им методов.* Если в ходе коммуникационного процесса данные передаются через открытые системы (а Интернет относится именно к таковым), то исключить доступ к ним посторонних лиц невозможно даже теоретически. Соответственно, системы защиты сосредоточены на втором компоненте информации — на методах. Их принцип действия основан на том, чтобы исключить или, по крайней мере, затруднить возможность подбора *адекватного* метода для преобразования данных в информацию. Одним из приемов такой защиты является *шифрование* данных.

Понятие о несимметричном шифровании информации

Системам шифрования столько же лет, сколько письменному обмену информацией. Обычный подход состоит в том, что к документу применяется некий метод шифрования, основанный на использовании *ключа*, после чего документ становится недоступен для чтения обычными средствами. Его можно прочитать только тот, кто знает ключ, — только он может применить адекватный метод чтения. Аналогично происходит шифрование и ответного сообщения. Если в процессе обмена информацией для шифрования и чтения пользуются одним и тем же ключом, то такой криптографический процесс является *симметричным*.

Основной недостаток симметричного процесса заключается в том, что, прежде чем начать обмен информацией, надо выполнить передачу ключа, а для этого опять-

таки нужна защищенная связь, то есть проблема повторяется, хотя и на другом уровне. Если рассмотреть оплату клиентом товара или услуги с помощью кредитной карты, то получается, что торговая фирма должна создать по одному ключу для каждого своего клиента и каким-то образом передать им эти ключи. Это крайне неудобно.

Поэтому в настоящее время в Интернете используют *несимметричные* криптографические системы, основанные на использовании не одного, а двух ключей. Происходит это следующим образом. Компания для работы с клиентами создает два ключа: один *открытый* (*public — публичный*), а другой *закрытый* (*private — личный*). На самом деле это как бы две «половинки» одного целого ключа, связанные друг с другом.

Ключи устроены так, что сообщение, зашифрованное одной половинкой, можно расшифровать только другой половинкой (не той, которой оно было закодировано). Создав пару ключей, торговая компания широко распространяет *публичный ключ* (открытую половинку) и надежно сохраняет *закрытый ключ* (свою половинку).

Как публичный, так и закрытый ключи представляют собой некую кодовую последовательность. Публичный ключ компании может быть опубликован на ее сервере, откуда каждый желающий может его получить. Если клиент хочет сделать фирме заказ, он возьмет ее публичный ключ и с его помощью закодирует свое сообщение о заказе и данные о своей кредитной карте. После кодирования это сообщение может прочесть только владелец закрытого ключа. Никто из участников цепочки, по которой пересылается информация, не в состоянии это сделать. Даже сам отправитель не может прочитать собственное сообщение, хотя ему хорошо известно содержание. Лишь получатель сможет прочесть сообщение, поскольку только у него есть закрытый ключ, дополняющий использованный публичный ключ.

Если фирме надо будет отправить клиенту квитанцию о том, что заказ принят к исполнению, она закодирует ее своим закрытым ключом. Клиент сможет прочитать квитанцию, воспользовавшись имеющимся у него публичным ключом данной фирмы. Он может быть уверен, что квитанцию ему отправила именно эта фирма, поскольку никто иной доступа к закрытому ключу фирмы не имеет.

Принцип достаточности защиты

Защита публичным ключом (впрочем, как и большинство других видов защиты информации) не является абсолютно надежной. Дело в том, что поскольку каждый желающий может получить и использовать чей-то публичный ключ, то он может сколь угодно подробно изучить алгоритм работы механизма шифрования и пытаться установить метод расшифровки сообщения, то есть *реконструировать закрытый ключ*.

Это настолько справедливо, что алгоритмы кодирования публичным ключом даже нет смысла скрывать. Обычно к ним есть доступ, а часто они просто широко публикуются. Тонкость заключается в том, что знание алгоритма еще не означает возможности провести реконструкцию ключа в *разумно приемлемые сроки*. Так, например, правила игры в шахматы известны всем, и нетрудно создать алгоритм для перебора всех возможных шахматных партий, но он никому не нужен, поскольку

даже самый быстрый современный суперкомпьютер будет работать над этой задачей дольше, чем существует жизнь на нашей планете.

Количество комбинаций, которое надо проверить при реконструкции закрытого ключа, не столь велико, как количество возможных шахматных партий, однако защиту информации принято считать достаточной, если затраты на ее преодоление превышают ожидаемую ценность самой информации. В этом состоит *принцип достаточности защиты*, которым руководствуются при использовании несимметричных средств шифрования данных. Он предполагает, что защита не абсолютна и приемы ее снятия известны, но она все же достаточна для того, чтобы сделать это мероприятие нецелесообразным. При появлении иных средств, позволяющих таки получить зашифрованную информацию в разумные сроки, изменяют принцип работы алгоритма, и проблема повторяется на более высоком уровне.

Разумеется, не всегда реконструкцию закрытого ключа производят методами простого перебора комбинаций. Для этого существуют специальные методы, основанные на исследовании особенностей взаимодействия открытого ключа с определенными структурами данных. Область науки, посвященная этим исследованиям, называется *криптоанализом*, а средняя продолжительность времени, необходимого для реконструкции закрытого ключа по его опубликованному открытому ключу, называется *криптостойкостью* алгоритма шифрования.

Для многих методов несимметричного шифрования криптостойкость, полученная в результате криптоанализа, существенно отличается от величин, заявляемых разработчиками алгоритмов на основании теоретических оценок. Поэтому во многих странах вопрос применения алгоритмов шифрования данных находится в поле законодательного регулирования. В частности, в России к использованию в государственных и коммерческих организациях разрешены только те программные средства шифрования данных, которые прошли государственную сертификацию в административных органах, в частности, в Федеральном агентстве правительственной связи и информации при Президенте Российской Федерации (ФАПСИ).

Понятие об электронной подписи

Мы рассмотрели, как клиент может переслать организации свои конфиденциальные данные (например, номер электронного счета). Точно так же он может общаться и с банком, отдавая ему распоряжения о перечислении своих средств на счета других лиц и организаций. Ему не надо ездить в банк и стоять в очереди — все можно сделать, не отходя от компьютера. Однако здесь возникает проблема: как банк узнает, что распоряжение поступило именно от данного лица, а не от злоумышленника, выдающего себя за него? Эта проблема решается с помощью так называемой *электронной подписи*.

Принцип ее создания тот же, что и рассмотренный выше. Если нам надо создать себе электронную подпись, следует с помощью специальной программы (полученной от банка) создать те же два ключа: *закрытый* и *публичный*. Публичный ключ передается банку. Если теперь надо отправить поручение банку на операцию с расчетным счетом, оно кодируется *публичным* ключом банка, а своя подпись под ним кодируется собственным *закрытым* ключом. Банк поступает наоборот. Он читает

поручение с помощью своего *закрытого* ключа, а подпись — с помощью *публичного* ключа поручителя. Если подпись читаема, банк может быть уверен, что поручение ему отправили именно мы, и никто другой.

Понятие об электронных сертификатах

Системой несимметричного шифрования обеспечивается делопроизводство в Интернете. Благодаря ей каждый из участников обмена может быть уверен, что полученное сообщение отправлено именно тем, кем оно подписано. Однако здесь возникает еще ряд проблем, например проблема регистрации даты отправки сообщения. Такая проблема возникает во всех случаях, когда через Интернет заключаются договоры между сторонами. Отправитель документа может легко изменить текущую дату средствами настройки операционной системы. Поэтому обычно дата и время отправки электронного документа не имеют юридической силы. В тех же случаях, когда это важно, выполняют сертификацию даты/времени.

Сертификация даты. Сертификация даты выполняется при участии третьей, независимой стороны. Например, это может быть сервер организации, авторитет которой в данном вопросе признают оба партнера. В этом случае документ, зашифрованный открытым ключом партнера и снабженный своей электронной подписью, отправляется сначала на сервер сертифицирующей организации. Там он получает «приписку» с указанием точной даты и времени, зашифрованную закрытым ключом этой организации. Партнер декодирует содержание документа, электронную подпись отправителя и отметку о дате с помощью своих «половинок» ключей. Вся работа автоматизирована.

Сертификация Web-узлов. Сертифицировать можно не только даты. При заказе товаров в Интернете важно убедиться в том, что сервер, принимающий заказы и платежи от имени некоей фирмы, действительно представляет эту фирму. Тот факт, что он распространяет ее открытый ключ и обладает ее закрытым ключом, строго говоря, еще ничего не доказывает, поскольку за время, прошедшее после создания ключа, он мог быть скомпрометирован. Подтвердить действительность ключа тоже может третья организация путем выдачи сертификата продавцу. В сертификате указано, когда он выдан и на какой срок. Если добросовестному продавцу станет известно, что его закрытый ключ каким-либо образом скомпрометирован, он сам уведомит сертификационный центр, старый сертификат будет аннулирован, создан новый ключ и выдан новый сертификат.

Прежде чем выполнять платежи через Интернет или отправлять данные о своей кредитной карте кому-либо, следует проверить наличие действующего сертификата у получателя путем обращения в сертификационный центр. Это называется *сертификацией Web-узлов*.

Сертификация издателей. Схожая проблема встречается и при распространении программного обеспечения через Интернет. Так, например, мы указали, что браузеры, служащие для просмотра *Web*-страниц, должны обеспечивать механизм защиты от нежелательного воздействия активных компонентов на компьютер клиента. Можно представить, что произойдет, если кто-то от имени известной компании начнет распространять модифицированную версию ее браузера, в которой специально

оставлены бреши в системе защиты. Злоумышленник может использовать их для активного взаимодействия с компьютером, на котором работает такой браузер.

Это относится не только к браузерам, но и ко всем видам программного обеспечения, получаемого через Интернет, в которое могут быть имплантированы «троянские кони», «компьютерные вирусы», «часовые бомбы» и прочие нежелательные объекты, в том числе и такие, которые невозможно обнаружить антивирусными средствами. Подтверждение того, что сервер, распространяющий программные продукты от имени известной фирмы, действительно уполномочен ею для этой деятельности, осуществляется путем *сертификации издателей*. Она организована аналогично сертификации *Web*-узлов.

Средства для проверки сертификатов обычно предоставляют браузеры. В частности, в браузере *Microsoft Internet Explorer* 6.0, работа с которым более подробно будет рассмотрена в следующей главе, доступ к центрам сертификации осуществляется командой Сервис ▸ Свойства обозревателя ▸ Содержание ▸ Сертификаты ▸ Доверенные корневые центры сертификации.

Практическое занятие

Упражнение 8.1. Создание соединения удаленного доступа


15 мин

1. Запустите Мастер новых подключений: Пуск ▸ Настройка ▸ Сетевые подключения ▸ Мастер новых подключений.
2. В окне мастера щелкните на кнопке Далее. Затем выберите тип подключения, установив переключатель Подключить к Интернету. Щелкните на кнопке Далее.
3. На следующем этапе работы мастера установите переключатель Установить подключение вручную и щелкните на кнопке Далее.
4. На следующем этапе работы мастера установите переключатель Через обычный модем и щелкните на кнопке Далее.
5. Введите произвольное название нового соединения в поле Имя поставщика услуг. Щелкните на кнопке Далее.
6. Заполните поле телефонного номера (номер должен быть получен от поставщика услуг). Щелкните на кнопке Далее.
7. Укажите имя пользователя и (дважды) пароль доступа. Эти данные должны быть получены от поставщика услуг. Щелкните на кнопке Далее.

В некоторых случаях имя пользователя и пароль предоставляются на Web-сайте поставщика услуг в ходе интерактивной регистрации. В этом случае для первичного подключения к Интернету и выполнения регистрации поставщик предоставляет имя пользователя и пароль для бесплатного гостевого доступа. В этом упражнении допустимо создать такое соединение для гостевого доступа к сайту поставщика услуг.

8. На завершающем этапе работы мастера можно установить флажок, который позволит создать ярлык подключения на Рабочем столе. Если переместить этот ярлык на панель быстрого запуска, установка соединения с Интернетом становится очень простым делом.

Упражнение 8.2. Настройка подключения для удаленного доступа

1. Откройте папку Сетевые подключения (Пуск ▸ Настройка ▸ Сетевые подключения). В этой папке находятся значки подключений, имеющихся на данном компьютере. Их может быть несколько.
2. Выберите настраиваемое подключение. Щелкните на его значке правой кнопкой мыши. В открывшемся контекстном меню выберите пункт Свойства — откроется диалоговое окно свойств данного подключения.
3. На вкладке Общие проверьте правильность ввода телефонного номера поставщика услуг Интернета и правильность выбора и настройки модема. В случае необходимости внесите необходимые изменения.

   Если поставщик услуг Интернета предоставил несколько телефонных номеров для подключения к его серверу, щелкните на кнопке Другие и введите дополнительные номера телефонов, используя кнопку Добавить.
4. На вкладке Сеть в списке Компоненты, используемые этим подключением флажки у пунктов Служба доступа к файлам и принтерам сетей Microsoft и Клиент для сетей Microsoft должны быть сброшены из соображений безопасности. Выберите в этом списке пункт Протокол Интернета (TCP/IP) и щелкните на кнопке Свойства.
5. Включите переключатель ввода *IP*-адреса в соответствии с указаниями поставщика услуг (для коммутируемого соединения обычно включают переключатель Получить IP-адрес автоматически).
6. Введите адреса серверов *DNS*. Если эти адреса получены от поставщика услуг, включите переключатель Использовать следующие адреса DNS-серверов и введите по четыре числа для первичного и вторичного серверов *DNS*. Если адреса не получены, возможно, что они вводятся автоматически. В этом случае включите переключатель Получить адрес DNS-сервера автоматически.
7. Щелчком на кнопке OK закройте диалоговое окно настройки свойств протокола *TCP/IP*.
8. Щелчком на кнопке OK закройте диалоговое окно настройки свойств подключения.

Упражнение 8.3. Установление соединения с сервером поставщика услуг

1. Запустите программу установки соединения двойным щелчком на значке настроенного соединения — откроется диалоговое окно Установка связи.
2. Проверьте правильность записи номера телефона.
3. Введите имя пользователя, согласованное с поставщиком услуг Интернета.
4. В поле Пароль введите пароль, полученный от поставщика услуг. При вводе пароля его символы заменяются подстановочными символами «*» и на экране не

видны. Предварительно убедитесь, что клавиатура находится в нужном регистре (строчные символы) и правильно выбрана раскладка клавиш (англоязычная). Чтобы при каждом сеансе связи не заниматься вводом имени пользователя и пароля, установите флажок Сохранить пароль.

Сохранение информации об имени пользователя и о его пароле происходит только при условии, что соединение успешно состоялось. Если оно не состоялось, эта информация не сохраняется и ее надо вводить заново.

5. Запустите программу щелчком на кнопке Подключиться. Если все сделано правильно, произойдет подключение к серверу поставщика услуг. По окончании процесса установки на панели индикации (справа на Панели задач) образуется значок работающего соединения.

6. Щелкните правой кнопкой мыши на значке работающего соединения на панели индикации. В открывшемся диалоговом окне узнайте параметры соединения, в частности скорость обмена данными с сервером поставщика услуг Интернета.

Сохранять информацию о пароле можно только на компьютерах, находящихся в личном пользовании. На компьютерах, предназначенных для коллективного использования, эту информацию не сохраняют. В операционных системах семейства Windows защита конфиденциальных данных организована не идеально. Подготовленному пользователю доступны косвенные данные, дающие возможности извлечь зашифрованные сведения обходными приемами.

9.1. Основные понятия World Wide Web

Сегодня Интернет используется как источник разносторонней информации по различным областям знаний. Большинство документов, доступных на серверах Интернета, имеют *гипертекстовый формат*. Службу Интернета, управляющую передачей таких документов, называют *World Wide Web* (*Web*, *WWW*). Этим же термином, или *средой WWW*, называют обширную совокупность *Web*-документов, между которыми существуют гипертекстовые связи.

Среда *WWW* не имеет централизованной структуры. Она пополняется теми, кто желает разместить в Интернете свои материалы, и может рассматриваться как *информационное пространство*. Как правило, документы *WWW* хранятся на постоянно подключенных к Интернету компьютерах — *Web-серверах*. Обычно на *Web*-сервере размещают не отдельный документ, а группу взаимосвязанных документов. Такая группа представляет собой *Web-узел* (альтернативный термин — *Web-сайт*). Размещение подготовленных материалов на *Web*-узле называется *Web-изданием* или *Web-публикацией*.

Web-страница. Отдельный документ *World Wide Web* называют *Web-страницей*. Обычно это комбинированный документ, который может содержать текст, графические иллюстрации, мультимедийные и другие вставные объекты. Для создания *Web*-страниц используется *язык HTML* (*HyperText Markup Language* — язык разметки гипертекста), который при помощи вставленных в документ *тегов* описывает логическую структуру документа, управляет форматированием текста и размещением вставных объектов. *Интерактивные Web*-узлы получают информацию от пользователя через *формы* и генерируют запрошенную *Web*-страницу с помощью специальных программ (*сценариев CGI*), *динамического HTML* и других средств.

Гиперссылки. Отличительной особенностью среды *World Wide Web* является наличие средств перехода от одного документа к другому, тематически с ним связанному, без явного указания адреса. Связь между документами осуществляется при помощи *гипертекстовых ссылок* (или просто *гиперссылок*). Гиперссылка — это выделенный

фрагмент документа (текст или иллюстрация), с которым ассоциирован адрес другого *Web*-документа. При использовании гиперссылки (обычно для этого требуется навести на нее указатель мыши и один раз щелкнуть) происходит *переход по гиперссылке* — открытие *Web*-страницы, на которую указывает ссылка. Механизм гиперссылок позволяет организовать тематическое путешествие по *World Wide Web* без использования (и даже без знания) адресов конкретных страниц.

Адресация документов. Для записи адресов документов Интернета (*Web*-страниц) используется форма, называемая *адресом URL*. Адрес *URL* содержит указания на прикладной протокол передачи, адрес компьютера и путь поиска документа на этом компьютере. Адрес компьютера состоит из нескольких частей, разделенных точками, например www.intel.ru. Части адреса, расположенные справа, определяют сетевую принадлежность компьютера, а левые элементы указывают на конкретный компьютер данной сети. Преобразование адреса *URL* в цифровую форму *IP*-адреса производит *служба имен доменов* (*Domain Name Service*, *DNS*). В качестве разделителя в пути поиска документа Интернета всегда используется символ косой черты.

Средства просмотра Web. Документы Интернета предназначены для отображения в *электронной форме*, причем автор документа не знает, каковы возможности компьютера, на котором документ будет отображаться. Поэтому язык *HTML* обеспечивает не столько форматирование документа, сколько описание его логической структуры. Форматирование и отображение документа на конкретном компьютере производится специальной программой — *браузером* (от английского слова *browser*).

Основные функции браузеров следующие:

- установление связи с *Web*-сервером, на котором хранится документ, и загрузка всех компонентов комбинированного документа;
- интерпретация тегов языка *HTML*, форматирование и отображение *Web*-страницы в соответствии с возможностями компьютера, на котором браузер работает;
- предоставление средств для отображения мультимедийных и других объектов, входящих в состав *Web*-страниц, а также механизма расширения, позволяющего настраивать программу на работу с новыми типами объектов;
- обеспечение автоматизации поиска *Web*-страниц и упрощение доступа к *Web*-страницам, посещенным ранее.
- предоставление доступа к встроенным или автономным средствам для работы с другими службами Интернета.

9.2. Работа с программой Internet Explorer 6.0

Со стороны Интернета работу службы *World Wide Web* обеспечивают серверные программные средства — *Web-серверы*. Со стороны пользователя работа обеспечивается клиентскими программами — *Web*-браузерами. Существует несколько разных браузеров, выпускаемых разными компаниями.

В принципе, все браузеры выполняют одни и те же функции, и выбор конкретного средства просмотра — дело вкуса и привычки пользователя. Однако у браузера

Microsoft Internet Explorer есть преимущество перед остальными, заключающееся в том, что, начиная с операционной системы *Windows* 98, он поставляется вместе с системой и интегрирован в нее так, что является ее неотъемлемым компонентом.

С последней версией операционной системы *Windows XP* поставляется версия браузера *Internet Explorer* 6.0. Эта программа предоставляет единый метод доступа к локальным документам компьютера, ресурсам корпоративной сети *intranet* и к информации, доступной в Интернете. Она обеспечивает работу с *World Wide Web*, предоставляет идентичные средства работы с локальными папками компьютера и файловыми архивами *FTP*, дает доступ к средствам связи через Интернет. Соответствующие программы (*Outlook Express*, Проигрыватель *Windows Media* и другие) автономны, но рассматриваются как часть пакета *Internet Explorer* 6.0. Схема использования Интернета через *Internet Explorer* представлена на рис. 9.1.

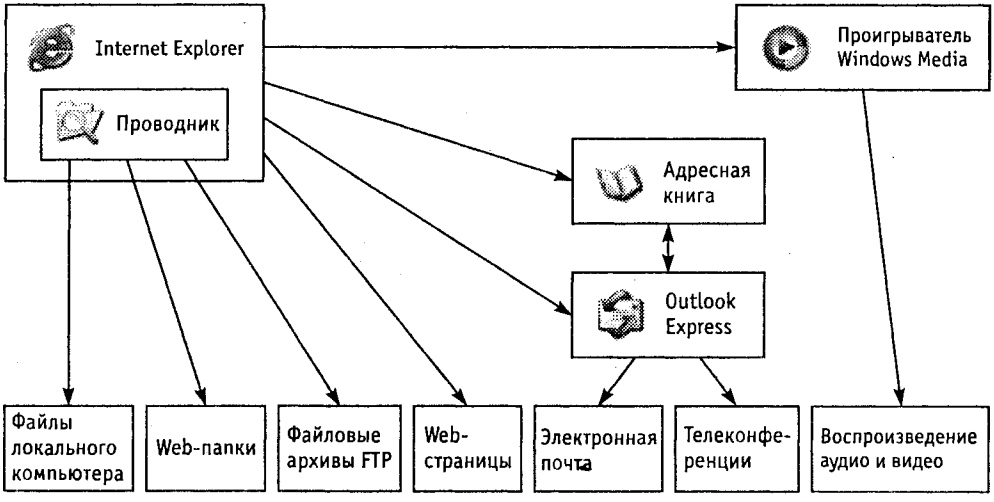


Рис. 9.1. Организация доступа к ресурсам Интернета

Для запуска браузера *Internet Explorer* можно использовать значок Internet Explorer на Рабочем столе или на Панели быстрого запуска, а также Главное меню (Пуск ▸ Программы ▸ Internet Explorer). Кроме того, программа запускается автоматически при попытке открыть документ Интернета или локальный документ в формате *HTML*. Для этой цели можно использовать ярлыки *Web*-страниц, папку Избранное (Пуск ▸ Избранное или пункт меню Избранное в строке меню окна папки или программы Проводник), панель инструментов Рабочего стола Адрес или поле ввода в диалоговом окне Запуск программы (Пуск ▸ Выполнить).

Если соединение с Интернетом отсутствует, то после запуска программы на экране появится диалоговое окно для управления установкой соединения. При невозможности установить соединение сохраняется возможность просмотра в *автономном режиме* ранее загруженных *Web*-документов. При наличии соединения после запуска программы на экране появится так называемая «домашняя», или *основная*, страница, выбранная при настройке программы.

Открытие и просмотр Web-страниц

Просматриваемая *Web*-страница отображается в рабочей области окна. По умолчанию воспроизводится все ее содержимое, включая графические иллюстрации и встроенные мультимедийные объекты. Управление просмотром осуществляется при помощи строки меню, панелей инструментов, а также активных элементов, имеющихся в открытом документе, например гиперссылок.

Если *URL*-адрес *Web*-страницы известен, его можно ввести в поле панели Адрес и щелкнуть на кнопке Переход. Страница с указанным адресом открывается вместо текущей. Наличие средства автозаполнения адресной строки упрощает повторный ввод адресов. Вводимый адрес автоматически сравнивается с адресами ранее просматривавшихся *Web*-страниц. Все подходящие адреса отображаются в раскрывающемся списке панели Адрес. Если нужный адрес есть в списке, его можно выбрать клавишами ВВЕРХ и ВНИЗ, после чего щелкнуть на кнопке Переход. При отсутствии нужного адреса ввод продолжают как обычно.

Работа с гиперссылками. Навигация по Интернету чаще выполняется не путем ввода адреса *URL*, а посредством использования *гиперссылок*. При отображении *Web*-страницы на экране гиперссылки выделяются цветом (обычно синим) и подчеркиванием. Обычно подчеркивание применяют *только* для выделения гиперссылок. Более надежным признаком является форма указателя мыши. При наведении на гиперссылку он принимает форму кисти руки с вытянутым указательным пальцем, а сама гиперссылка при соответствующей настройке браузера изменяет цвет. Адрес *URL*, на который указывает ссылка, отображается в строке состояния. При щелчке на гиперссылке соответствующая *Web*-страница загружается вместо текущей. Если гиперссылка указывает на произвольный файл, его загрузка происходит по протоколу *FTP*.

На *Web*-страницах могут также встречаться графические ссылки (то есть, гиперссылки, представленные рисунком) и изображения-карты, объединяющие несколько ссылок в рамках одного изображения. Для просмотра ссылок на открытой *Web*-странице удобно использовать клавишу TAB. При нажатии этой клавиши фокус ввода (пунктирная рамка) перемещается к следующей ссылке. Перейти по ссылке можно, нажав клавишу ENTER. При таком подходе последовательно перебираются текстовые и графические ссылки, а также отдельные области изображений-карт.

Дополнительные возможности использования гиперссылок предоставляет их контекстное меню. Чтобы открыть новую страницу, не закрывая текущей, применяют команду Открыть в новом окне. В результате открывается новое окно браузера. Адрес *URL*, заданный ссылкой, можно поместить в буфер обмена при помощи команды Копировать ярлык. Его можно вставить в поле панели Адрес или в любой другой документ для последующего использования.

Другие операции, относящиеся к текущей странице и ее элементам, также удобно осуществлять через контекстное меню. Так, например, рисунок, имеющийся на странице, можно:

- сохранить как файл (Сохранить рисунок как);

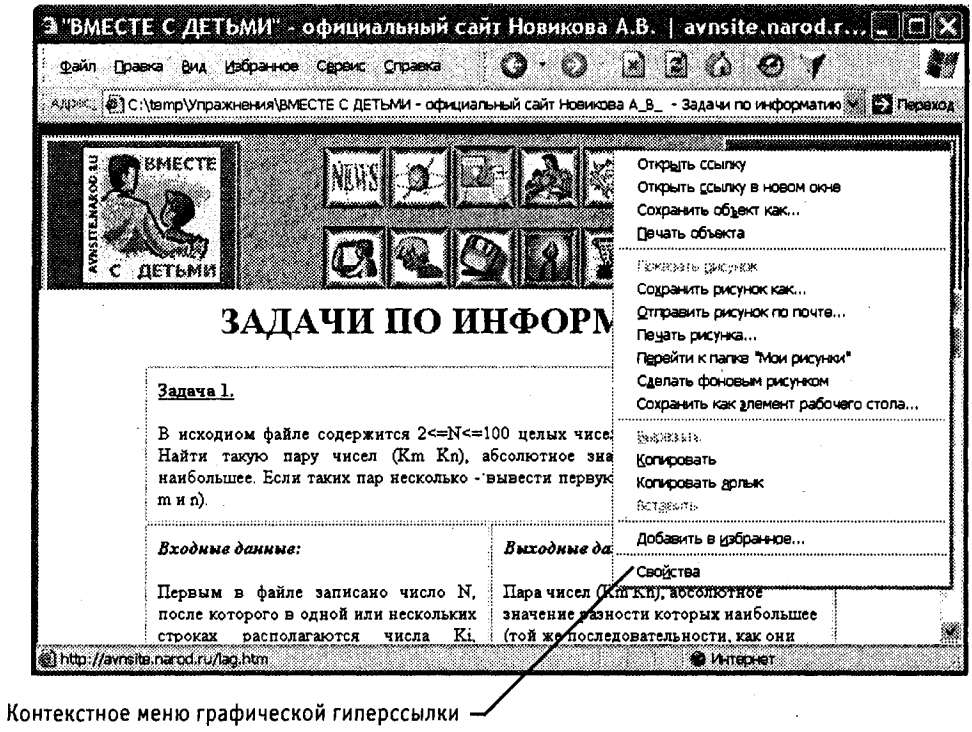


Рис. 9.2. Web-страница в ходе просмотра

- использовать как фоновый рисунок (Сделать фоновым рисунком) или как активный элемент (Сохранить как элемент рабочего стола).

Если рисунок выполняет функции графической ссылки, к нему можно применять как команды, относящие к изображению, так и команды, относящиеся к ссылке.

Приемы управления броузером

Необходимость определенных действий в ходе просмотра документов *World Wide Web* часто диктуется самим ходом работы. В таких случаях удобно использовать кнопки панели инструментов Обычные кнопки. Для того чтобы вернуться к странице, которая просматривалась некоторое время назад, используют кнопку Назад. Чтобы возвратиться на несколько страниц назад, можно использовать присоединенную к ней кнопку раскрывающегося списка. Отменить действия, выполненные при помощи кнопки Назад, позволяет кнопка Вперед.

Если процесс загрузки страницы затянулся или надобность в ней отпала, используют кнопку Остановить. Заново загрузить *Web*-страницу, если ее загрузка была прервана или содержание документа изменилось, позволяет кнопка Обновить. Чтобы немедленно загрузить «домашнюю» (основную) страницу, с которой браузер обычно начинает работу, пользуются кнопкой Домой.

Создать новое окно, сохранить открытый документ на своем компьютере, распечатать его, включить или выключить режим автономной работы, а также завершить работу с программой позволяют команды меню Файл.

Копирование фрагментов документа в буфер обмена, поиск текста на *Web*-странице осуществляются при помощи команд меню Правка.

Включение и выключение отображения служебных элементов окна (панелей инструментов, дополнительных панелей, строки состояния), выбора шрифта и кодировки символов осуществляются через меню Вид.

Ведение списка регулярно посещаемых страниц и быстрый доступ к ним осуществляются через меню Избранное. Переход к использованию программ для работы с другими службами Интернета, а также настройка браузера осуществляются через меню Сервис.

Работа с несколькими окнами

Нередко возникает необходимость открыть новый *Web*-документ, не закрывая текущий, например в тех случаях, когда текущий документ содержит список интересных ссылок. Чтобы открыть новое окно программы *Internet Explorer*, применяют команду Файл ▸ Создать ▸ Окно. Каждое окно отображает свой *Web*-документ и может использоваться самостоятельно. В частности, списки кнопок Назад и Вперед обновляются в каждом окне индивидуально.

Закрывать окна программы *Internet Explorer* можно в любом порядке, а не только в том, в каком они открывались. Однако при закрытии последнего окна на компьютере может больше не остаться открытых программ, использующих Интернет. В такой ситуации на экран выдается предупреждающее сообщение, позволяющее разорвать соединение, если оно действительно больше не нужно.

Настройка свойств броузера

Для эффективной и комфортной работы в Интернете необходима настройка браузера. Параметры оптимальной настройки зависят от многих факторов:

- свойств видеосистемы компьютера;
- производительности действующего соединения с Интернетом;
- содержания текущего *Web*-документа;
- личных предпочтений индивидуального пользователя.

Начать настройку программы *Internet Explorer* можно как из самой этой программы (Сервис ▸ Свойства обозревателя), так и через общесистемное средство *Windows* — Панель управления (значок Свойства обозревателя). Открывшееся диалоговое окно отличается в этом случае только названием (Свойства обозревателя и Свойства: Интернет). Оно содержит семь вкладок, предназначенных для настройки разных групп параметров.

Общие параметры работы браузера задают на вкладке Общие (рис. 9.3). Здесь можно указать, какую страницу следует использовать в качестве основной, задать объем дискового пространства для хранения временных файлов Интернета и удалить

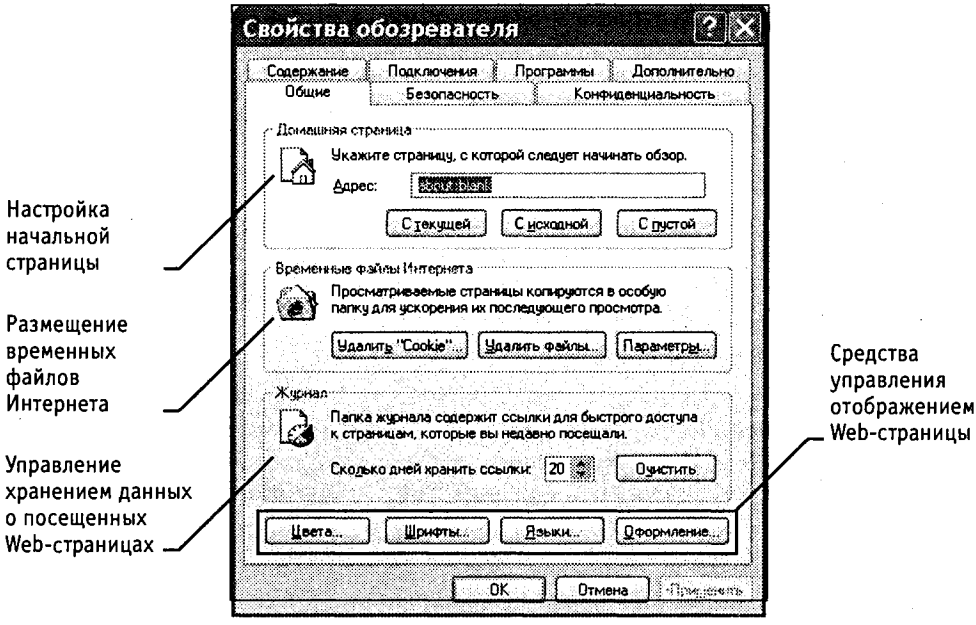


Рис. 9.3. Управление основными параметрами отображения Web-страниц

такие временные файлы, а также страницы, подготовленные для чтения в автономном режиме. Правила хранения временных файлов задаются с помощью кнопки Настройка. Чем реже программа проверяет соответствие версий давно загруженной страницы и реального документа, тем больше экономится времени на загрузке страниц, но увеличивается риск их устаревания. Кнопка Обновить на панели инструментов Обычные кнопки позволит получить самую последнюю версию документа независимо от настроек.

Управление оформлением отображаемых *Web*-страниц также осуществляется элементами управления вкладки Общие. Используемые цвета настраиваются при помощи кнопки Цвета, а шрифты — при помощи кнопки Шрифты. Эти настройки подчинены тому, что задано в самом *Web*-документе.

Если по какой-либо причине необходим полный контроль над оформлением отображаемых документов, используют кнопку Оформление. С ее помощью можно задать принудительное использование параметров форматирования, заданных в свойствах браузера. Это может относиться к используемым цветам (флажок Не учитывать цвета, указанные на веб-страницах), начертаниям шрифтов (Не учитывать шрифты, указанные на веб-страницах) и размерам шрифтов (Не учитывать размеры шрифтов, указанные на веб-страницах).

Настройка свойств соединения с Интернетом осуществляется при помощи вкладки Подключение. Здесь доступны те же операции, что и при непосредственном использовании папки Сетевые подключения. Кроме того, можно указать, какое именно соединение должно использоваться при работе браузера. С помощью переключа-

телей можно задать режим отказа от автоматического подключения, стандартный режим подключения при отсутствии соединения или режим использования только одного соединения.

Выбор программ, используемых для работы в Интернете, осуществляется с помощью вкладки Программы. Все виды программ, кроме календаря (для ведения списка дел, встреч, праздников и прочего), входят непосредственно в дистрибутивный пакет *Internet Explorer* 6.0.

Средства защиты от потенциально опасного содержимого *Web*-документов предоставляет вкладка Безопасность. Она позволяет указать *Web*-узлы, взаимодействие с которыми следует считать опасным, и запретить прием с них информации, которая может оказаться разрушительной.

Вкладка Конфиденциальность позволяет ограничить доступ *Web*-узлов к личной информации, хранимой на компьютере. *Web*-узлы имеют потенциальную возможность сохранять на компьютере пользователя небольшие информационные файлы («маркеры *cookie*») и читать их при последующих обращениях к тому же узлу. На вкладке Конфиденциальность можно задать ограничения на прием маркеров *cookie*.

Для ограничения доступа к узлам с неприемлемым содержанием, а также для управления использованием электронных сертификатов служат элементы управления вкладки Содержание.

Прочие настройки сосредоточены на вкладке Дополнительно. Они позволяют:

- соблюдать конфиденциальность работы с помощью средств шифрования, использования электронных сертификатов и своевременного удаления временных файлов;
- контролировать использование средств языка *Java*;
- управлять отображением мультимедийных объектов;
- использовать дополнительные настройки оформления;
- управлять режимом поиска *Web*-страниц, содержащих нужную информацию.

Прием файлов из Интернета

Гиперссылки, имеющиеся на *Web*-страницах, могут указывать на документы разных типов. Если браузер не способен отображать файлы определенного типа (например, исполняемые файлы с расширением .EXE, архивы .ZIP и прочие), инициируется процесс загрузки данного файла на компьютер.

Программа *Internet Explorer* 6.0 запускает мастер загрузки файла, на первом этапе работы которого требуется указать, следует ли открыть файл или сохранить его на диске. «Открытие» файла подразумевает загрузку его в каталог временных файлов и немедленный запуск (если это исполняемый файл) или открытие с помощью программы, которая предназначена для работы с файлами этого типа. Такой подход открывает путь на компьютер для небезопасной информации. Надежнее выбрать сохранение файла на диске. В этом случае требуется выбрать папку, в которой следует сохранить файл, и задать имя файла.

Ход загрузки файла отображается в специальном окне (рис. 9.4). Шкала хода работы появляется только в том случае, когда мастер управления загрузкой может получить информацию о полной длине файла, а это возможно только когда файл загружается непосредственно с *Web*-узла. При загрузке файла с узла *FTP* такие данные предоставляются не всегда. За ходом загрузки можно также следить по строке заголовка окна или, если окно свернуто или скрыто другими окнами, по надписи на кнопке Панели задач. Процесс загрузки файла не препятствует параллельному просмотру *Web*-страниц или другим операциям в Интернете.

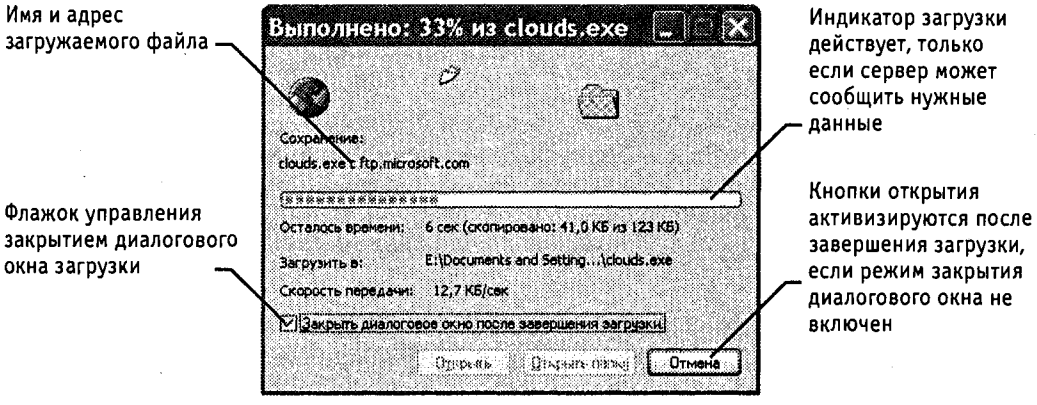


Рис. 9.4. Загрузка файла с узла FTP

После окончания загрузки окно загрузки закрывается автоматически, если установлен флажок Закрыть диалоговое окно после завершения загрузки. В противном случае после окончания загрузки активизируются кнопки Открыть и Открыть папку, которые позволяют, соответственно, открыть только что загруженный файл или папку, которая его содержит.

Загрузку файла можно прервать в любой момент при помощи кнопки Отмена. После прерывания загрузки пользователем или вследствие разрыва соединения, эту операцию необходимо начать заново. В операционной системе *Windows XP* нет средств, способных возобновить загрузку файла, прерванную по какой-либо причине. Это возможно только при использовании специальных служебных программ.

Файлы, доступные для загрузки любым пользователям, чаще всего хранятся на *FTP*-узлах. Для доступа к *FTP*-узлу можно указать его адрес *URL* на панели Адрес. Браузер *Internet Explorer* 6.0 обеспечивает по умолчанию *анонимное подключение* к узлу *FTP*, при котором разрешены только просмотр каталогов и загрузка файлов. Если анонимный доступ не разрешен, на экране отображается диалоговое окно для ввода имени и пароля (разумеется, их следует знать).

Окно *FTP*-узла выглядит на экране как обычное окно папки, но с использованием значка удаленной папки. Для загрузки файла надо щелкнуть на его значке правой кнопкой мыши и выбрать в контекстном меню команду Копировать в папку. Если для данного каталога *FTP* разрешены все файловые операции, то с ним можно работать точно так же, как с окном папки. Невозможен только прямой перенос файлов

с одного узла на другой. Чтобы осуществить такую операцию, надо сначала перенести файл в локальную папку компьютера, а затем отправить ее оттуда на другой *FTP*-узел или в другой каталог того же *FTP*-узла.

9.3. Поиск информации в World Wide Web

Интернет имеет три функции: *коммуникационную, информационную* и *управленческую*. Разные службы могут обеспечивать разные функции. Хотя в рамках службы *World Wide Web* есть сервисы, исполняющие коммуникационные и управленческие функции, основное назначение этой службы — информационное. Когда нам нужно разыскать какие-то сведения, мы обращаемся за данными в первую очередь в информационное пространство *Web*.

Это пространство отличается гигантскими размерами и содержит несколько миллиардов *Web*-документов. Найти среди них именно то, что нужно, — это особая, отнюдь не простая задача. Разумеется, можно пользоваться рекомендациями знакомых, коллег по работе, адресами *URL*, опубликованными в средствах массовой информации, но службе *WWW* совершенно необходимы свои поисковые сервисы, и они существуют.

Поисковая система представляет собой специализированный *Web*-узел. Пользователь сообщает поисковой системе данные о содержании искомой *Web*-страницы, а поисковая система выдает список гиперссылок на страницы, на которых упоминаются соответствующие сведения. Существует несколько моделей, на которых основана работа поисковых систем, но исторически две модели приобрели наибольшую популярность — это *поисковые каталоги* и *поисковые указатели*.

Поисковые каталоги

Поисковые каталоги устроены по тому же принципу, что и тематические каталоги крупных библиотек. Обратившись к поисковому каталогу, мы находим на его основной странице сокращенный список крупных тематических категорий, например таких, как Наука (Science), как показано на примере поискового каталога *Yahoo!* (рис. 9.5).

Каждая запись в списке категорий — это гиперссылка. Щелчок на ней открывает следующую страницу поискового каталога, на котором данная тема представлена подробнее, например по предметам: Астрономия, Биология, География, Математика, Физика и многие другие. Щелчок на названии темы (например, Физика) открывает страницу со списком разделов (Астрофизика, Атомная физика, Гидродинамика, Механика и т. д.). Продолжая погружение в тему, можно дойти до списка конкретных *Web*-страниц и выбрать себе тот ресурс, который лучше подходит для решения задачи.

Работа с поисковыми каталогами интуитивно проста. В них поиск информации практически всегда завершается более или менее плодотворно. Однако за этой простотой скрывается высочайшая сложность создания и ведения каталога. Поисковые каталоги создаются вручную, коллективом высококвалифицированных редакторов. При этом общий объем каталогизированных *Web*-ресурсов невелик, а степень охвата общего объема ресурсов *WWW* непрерывно уменьшается.

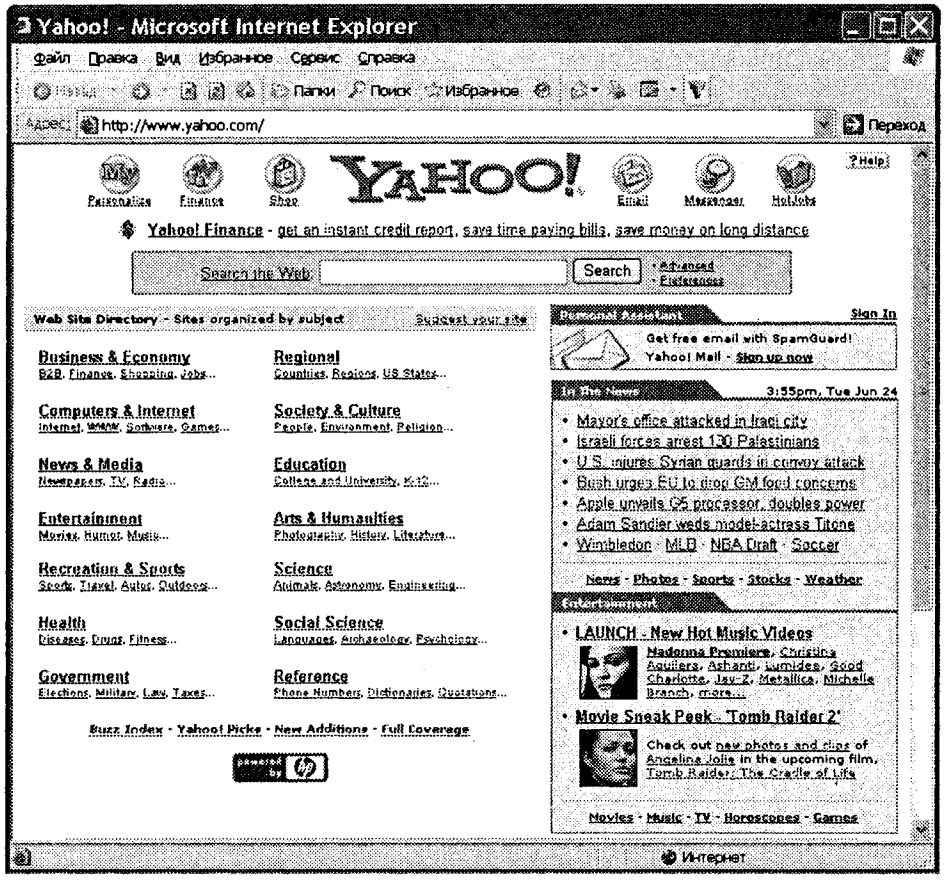


Рис. 9.5. Основная страница поискового каталога Yahoo!

Несмотря на низкий *коэффициент охвата*, поисковые каталоги пользуются огромной популярностью. Их принято использовать для первичного, реферативного поиска информации по заданной теме. Если для пользователя тема является совершенно новой и неисследованной, то ему, прежде всего, нужны указатели на классические, наиболее содержательные ресурсы, а именно это и обеспечивают поисковые каталоги. Человеческий фактор, связанный с тем, что над составлением каталога работают люди, а не программы, обеспечивает качественный отбор наиболее важных ресурсов по каждой теме.

Поисковые указатели

Автоматическую каталогизацию *Web*-ресурсов и удовлетворение запросов клиентов выполняют так называемые *поисковые указатели*. Из процесса наполнения базы данных поисковой системы исключается человеческий фактор. При этом значительно падает качество ссылок, предоставляемых системой по результатам поиска, но одновременно увеличивается их количество.

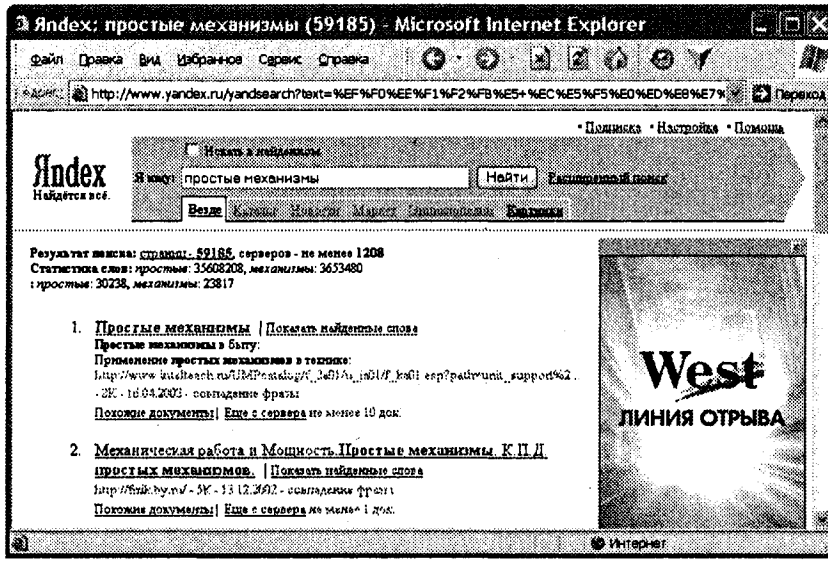


Рис. 9.6. Поиск информации по ключевым словам с помощью поисковой системы Яндекс

Основной принцип работы поискового указателя заключается в поиске *Web*-ресурсов по *ключевым словам*. Пользователь описывает искомый ресурс с помощью ключевых слов, после чего дает задание на поиск. Поисковая система анализирует данные, хранящиеся в своей базе, и выдает список *Web*-страниц, соответствующих запросу. Вместе с гиперссылками выдаются краткие сведения о найденных ресурсах, на основании которых пользователь может выбрать нужные ему ресурсы (рис. 9.6).

Разные поисковые указатели применяют разные информационные технологии для обработки запросов пользователей. Чтобы эффективно выполнять поиск информации в *WWW*, надо хотя бы в общих чертах понимать принципы их работы.

Три этапа работы поискового указателя. Работу поискового указателя можно условно разделить на три этапа. Из них два этапа являются подготовительными — они незаметны для клиента, и лишь на третьем этапе происходит взаимодействие с пользователем, но от каждого из этапов зависят функциональные свойства поисковой системы и эффективность работы с ней.

Сбор первичной базы данных. На первом этапе поисковая система занимается сканированием информационного пространства *World Wide Web*. Для этого используют специальные агентские программы — *черви*. Не следует путать агентов поисковых систем с разновидностью сетевых компьютерных вирусов, тоже именуемых *червями*. Черви поисковых систем совершенно безобидны для серверов и клиентов *WWW*. По своей сути это очень эффективные малоразмерные браузеры. Им не надо выполнять функции просмотра и воспроизведения содержимого — их задача состоит только в том, чтобы автоматически разыскивать *Web*-ресурсы, следуя по гиперссылкам, и, убедившись, что этот ресурс системе еще не известен, копировать его

в свою базу данных. Так же происходит и обновление ранее принятых документов, но измененных за время после предыдущего копирования.

Индексация базы данных. Собрать базу данных сетевых *Web*-ресурсов — еще не значит получить функционирующую поисковую систему. Поиск ключевых слов, введенных пользователем, в обширной базе — это весьма продолжительная операция. Чтобы не задерживать клиента более чем на доли секунды, собранные базы данных проходят предварительную обработку, называемую *индексацией*. На этапе индексации создаются специализированные документы — *поисковые указатели*.

Рафинирование результирующего списка. Это третий этап работы, в ходе которого осуществляется взаимодействие с пользователем. На этом этапе создается список ссылок, который будет передан пользователю в качестве результирующего. Пользовательское представление о качестве работы поисковой системы напрямую зависит от технологий, использованных на этом этапе.

Рафинирование заключается в *фильтрации* и *ранжировании* результатов поиска. Под фильтрацией понимается отсев ссылок, которые выдавать пользователю нецелесообразно. Прежде всего проверяется наличие дубликатов. Если система в одном списке выдает множество ссылок, ведущих к одному и тому же *Web*-ресурсу, это говорит о том, что ее средства добросовестно отработали два первых этапа, но ничего не сделали на третьем этапе. Дублирующиеся ссылки перегружают результирующий список и затрудняют выбор действительно полезных ресурсов.

Ранжирование заключается в создании специального порядка представления результирующего списка, при котором наиболее «полезные» (с точки зрения поисковой системы) ссылки приводятся в вершине списка, а наименее полезные — в его конце. Понимание критерия «полезности» для клиента той или иной ссылки может быть самым разнообразным. Именно поэтому разные поисковые системы, даже работающие с одинаковыми базами ресурсов, выдают разные результаты поиска.

Новые поисковые технологии

Автоматическая каталогизация. Для поисковых каталогов вопрос несоответствия между размерами исследованного и неисследованного *Web*-пространства стоит особенно остро. Перспективные направления развития основаны на внедрении так называемых *SMART*-технологий автоматической каталогизации.

Существует множество теоретических изысканий в области *SMART*-технологий, но наиболее перспективной является модель векторного информационного пространства. Представим себе эксперта в какой-то области, например в физике. Если ему поставить задачу, то, наверное, он сможет составить словари, характерные для таких областей, как Механика, Термодинамика, Оптика и т. п. Проанализировав множество документов, относящихся к этим научным областям, он сможет не только указать характерные термины и понятия, но и дать им весовые оценки. Так, например, достаточно очевидно, что слово «перемещение» имеет больший вес в механике, чем в термодинамике. Комбинируя термины и весовые коэффициенты, можно строить многомерные системы координат, в которых различные области знания описывались бы разными многомерными векторами.

Автоматически получив новую *Web*-страницу, поисковая система может построить для нее математический вектор, основанный на формальном анализе содержания. Сравнивая этот вектор с уже рассчитанными векторами для различных областей знания, система может без участия человека предположить, к какой категории, теме и разделу относится тот или иной документ.

При таком подходе не обязательно хранить копии всех известных *Web*-страниц, как не надо хранить и их поисковые указатели. Вполне достаточно для каждого *Web*-документа хранить лишь его *URL*-адрес и число, соответствующее вектору. В настоящее время конкретные алгоритмы *SMART*-технологий не публикуются, но можно предположить, что они уже работают, например в поисковых системах реального времени.

Поисковые системы реального времени. Это новое направление в технологиях поиска. Для работы с такой службой пользователь должен подключиться к ее центральному серверу, получить оттуда и установить на своем компьютере клиентскую программу. Эта программа подключается к браузеру и работает как дополнительная панель.

При каждом запуске браузера клиентская программа устанавливает соединение со своим центральным сервером и далее работает с ним в паре. Она передает серверу копии всех *Web*-страниц, которые посещает пользователь, то есть выполняет те же функции, что и автоматический *червь*, копирующий *Web*-ресурсы на сервер традиционной поисковой системы. Однако при этом есть два существенных различия:

- во-первых, человек в ходе навигации в *WWW* руководствуется не теми принципами, что автоматическая программа, поэтому сервер получает копии не всех *Web*-ресурсов, а только тех, что заинтересовали кого-то из его клиентов;
- во-вторых, если поставкой *Web*-ресурсов занимаются несколько миллионов постоянных клиентов, индексация *Web*-пространства происходит намного быстрее.

В свою очередь, пользователь тоже имеет важное преимущество. На какой бы *Web*-странице он ни находился, система всегда готова предложить ему список других *Web*-страниц, имеющих близкое по тематике содержание. Она готовит этот список на основании предшествующего опыта, полученного в работе с другими людьми. Так можно получить рекомендации, которые было бы очень трудно (а зачастую и невозможно) разыскать в *WWW* традиционными поисковыми средствами (рис. 9.7).

Рекомендации по приемам эффективного поиска

При проведении первичного реферативного поиска, когда тема задана достаточно широко, целесообразно использовать поисковые каталоги. Это позволит быстро установить местоположение основных первоисточников. При ознакомлении с первоисточниками следует, прежде всего, уделять внимание понятийной базе. Знание основных понятий и терминов позволит перейти к углубленному поиску в поисковых указателях с использованием ключевых слов, наиболее точно характеризующих тему.

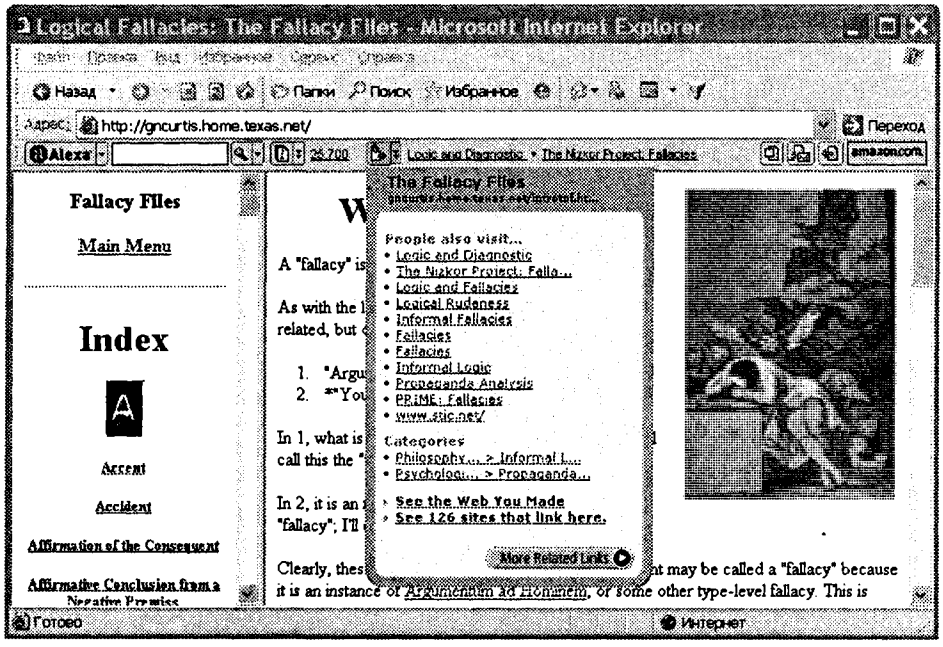


Рис. 9.7. При просмотре Web-страницы, посвященной логическим ошибкам в рассуждениях, система Alexa предлагает ссылки на другие Web-страницы аналогичной тематики

При наличии первичных сведений по теме поиска документы можно разыскивать в поисковых указателях. При этом следует различать приемы *простого, расширенного, контекстного* и *специального поиска.*

- Под *простым поиском* понимается поиск *Web*-ресурсов по одному или нескольким ключевым словам. Недостаток простого поиска заключается в том, что обычно он выдает слишком много документов, среди которых трудно выбрать наиболее подходящие.
- При использовании *расширенного поиска* ключевые слова связывают между собой операторами логических отношений. Расширенный поиск применяют в тех случаях, когда приемы простого поиска дают слишком много результатов. С помощью логических отношений поисковое задание формируют так, чтобы более точно детализировать задание и ограничить область отбора, например по дате публикации или по типу данных.
- *Контекстный поиск* — это поиск по точной фразе. Он удобен для реферативного поиска информации, но доступен далеко не во всех поисковых системах. Прежде всего, чтобы обеспечивать такую возможность, система должна работать не только с индексированными файлами, но и с полноценными образами *Web*-страниц. Эта операция достаточно медленная, и ее выполняют лишь немногие поисковые системы.

- *Специальный поиск* применяют при розыске Web-страниц, содержащих ссылки на заданные адреса *URL*, а также содержащих заданные данные в служебных полях, например в поле заголовка.

Рекомендации по использованию поисковых систем

Для проведения научных поисков рекомендуется пользоваться поисковой системой *Northern Light* (www.northernlight.com). Эта система имеет один из лучших коэффициентов охвата *Web*-пространства, и ее администрация прилагает специальные усилия для поддержания актуальности своих указателей. Кроме того, система удачно сочетает свойства поискового указателя и каталога. По наиболее популярным темам в ней можно найти специальные разделы каталожного типа — они называются *Special Editions* и подготавливаются вручную. Дополнительно система предоставляет платные услуги по поставке актуальных научных документов. Они находятся в разделе *Special Collection*.

Самым большим поисковым указателем обладает поисковая система *Fast Search* (www.alltheweb.com).

В России в настоящее время наиболее эффективно использовать поисковую систему Яндекс (www.yandex.ru), обеспечивающую максимальный охват российского сектора *WWW*. Она сочетает в себе возможности поискового каталога и поискового указателя. Особенно удобно использовать ее при формировании сложных поисковых заданий, поскольку она обладает очень гибким языком для расширенного поиска.

Специальные возможности поиска в программе Internet Explorer

Программа *Internet Explorer* 6.0 имеет специальные средства организации поиска без явного обращения к поисковым системам. Проще всего дать задание на поиск непосредственно с панели Адрес. Для этого надо ввести туда ключевое слово go, find или ? и ключевую фразу или набор ключевых слов. Поиск будет произведен с помощью поисковой системы, заданной по умолчанию. Результаты поиска отображаются в виде списка ссылок.

Другая возможность поиска состоит в обращении к мини-порталу, поддерживаемому компанией *Microsoft*. Он организует поиск с помощью существующих систем в соответствии с предпочтениями пользователя. Для такого поиска следует открыть в браузере дополнительную панель Поиск, щелкнув на кнопке Поиск на панели инструментов Обычные кнопки. Содержание панели Поиск загружается с *Web*-узла компании *Microsoft*. Ключевые слова или ключевая фраза вводятся в текстовое поле на этой панели.


При открытия панели Поиск из окна папки она открывается в режима поиска файлов и папок на компьютере. В режиме просмотра Web-страниц эта панель предполагает поиск в Интернете.

Поиск начинается по щелчку на кнопке на панели Поиск. Результаты представляются на этой же панели в виде упрощенной страницы результатов, полученных от реально использованной поисковой системы. Чтобы с результатами было удобнее работать, можно расширить панель Поиск, перетащив правую границу, или предста-

вить результаты поиска в окне с помощью команды контекстного меню Открыть в отдельном окне.

Выбрать используемый способ поиска можно с помощью кнопки Настроить на панели Поиск. В открывшемся диалоговом окне каждая группа элементов управления соответствует определенному типу поиска и позволяет указать, какие поисковые системы должны использоваться.

9.4. Отправка и получение сообщений

Для работы с электронной почтой и телеконференциями обычно используют единую программу, так как и в том и в другом случае речь идет об отправке и приеме сообщений. Часто оказывается удобным объединение средств работы с этими службами в рамках одной программы. Например, так сделано в программе *Outlook Express*, которая позволяет получать и отправлять сообщения электронной почты и телеконференций, используя аналогичные средства.

Возможность использования электронной почты сегодня не рассматривается как самостоятельная услуга и автоматически предоставляется тем, кто подключается к Интернету без дополнительной оплаты. Адрес электронной почты состоит из двух частей. Доменный адрес условно соответствует двум последним частям обозначения компьютера в адресе *URL* и фактически представляет собой адрес локальной сети, к которой принадлежит конкретный пользователь. Вторая часть адреса (которая в записи идет перед первой и отделяется от нее символом «@») указывает конкретного пользователя в этой локальной сети. Сообщения для данного адресата накапливаются на *почтовом сервере*, а затем передаются на компьютер адресата по запросу.

Например, пользователь, подключающийся к Интернету через поставщика услуг *ABCDE*, может иметь адрес типа myname@abcde.ru.

Телеконференции (или *группы новостей*) представляют собой средства распространения сообщений, не предназначенных для конкретного адресата. Информация о наличии сообщения постепенно распространяется от одного *сервера новостей* к другому. Сообщение хранится на сервере в течение некоторого времени (от нескольких дней до нескольких недель) после чего сбрасывается. Пользователь имеет доступ ко всем сообщениям, имеющимся на данном сервере новостей.

Авторы сообщений направляют их в тематические телеконференции. Имена телеконференций образуют иерархическую структуру, не имеющую единого корня. Элементы имени разделяются точками, старшие элементы располагаются слева, младшие — правее. Чем больше элементов в имени телеконференции, тем более узкой теме она посвящена.

Например, телеконференция news.announces.newusers содержит регулярно обновляемый набор сообщений (на английском языке), предназначенный для ознакомления начинающих с правилами использования телеконференций и сетевым этикетом. А скажем, с элементов comp.hardware... начинается целое семейство телеконференций, посвященных различным темам, связанным с аппаратным обеспечением компьютеров.

При обращении к телеконференции сервер новостей передает на компьютер пользователя заголовки имеющихся в ней и не прочитанных пользователем сообщений. Текст сообщений передается позже в соответствии с указаниями пользователя и настройками программы чтения сообщений телеконференций. Можно также отправить в телеконференцию новое сообщение или отклик.

Хотя электронная почта и служба новостей — разные службы, для пользователя они почти одинаковы, так как и в том и в другом случае речь идет об отправке и получении сообщений.

Сообщение, отправляемое в телеконференцию, носит общественный характер, а частную информацию следует пересылать по электронной почте. Однако ни одна из этих служб не годится для пересылки *конфиденциальной* информации, которая не должна быть доступна посторонним.

Работа с программой Outlook Express

Создание учетной записи. Сообщения электронной почты и телеконференций накапливаются, соответственно, на *почтовом сервере* и *сервере новостей*. Для работы с этими службами предназначена программа *Microsoft Outlook Express* (Пуск ▸ Программы ▸ Outlook Express). Из браузера *Internet Explorer* 6.0 она запускается командой Сервис ▸ Почта и новости ▸ Читать почту.

Так как сообщения поступают и отправляются через сервер, программе требуется указать информацию об используемом сервере. Эта информация хранится в виде *учетной записи*.

В программе *Outlook Express* учетную запись создают командой Сервис ▸ Учетные записи. В диалоговом окне Учетные записи в Интернете надо щелкнуть на кнопке Добавить и выбрать в открывшемся меню службу, для которой создается учетная запись. Последующая информация вводится под управлением мастера и включает имя, указываемое как имя отправителя, адрес электронной почты, имя используемого сервера и, в случае необходимости, имя пользователя и пароль.

Создание сообщения электронной почты. Чтобы отправить сообщение электронной почты, его надо создать. Для этого следует щелкнуть на кнопке Создать сообщение на панели инструментов. При этом открывается окно Создать сообщение, рабочая область которого разбивается на две основные части. В верхней части располагаются поля для ввода служебной информации, а в нижней — собственно текст сообщения. В поле Тема вводится краткое описание вопроса, которому посвящено сообщение. После того как тема указана, соответствующий текст становится заголовком окна. В поле Кому вводится адрес основного получателя письма, в поле Копия — адреса получателей копии. Если необходимо отправить копию письма, о которой ничего не известно другим адресатам, соответствующий адрес вводится в поле Скрытая (если такое поле отсутствует, надо дать команду Вид ▸ Все заголовки).

В ходе создания и редактирования сообщения наличие связи с почтовым сервером не требуется. Такая связь нужна только в момент отправки (получения) сообщений. Программа *Outlook Express* устроена таким образом, что отправка и получение сообщений осуществляются одновременно. Так, получение и доставка почты

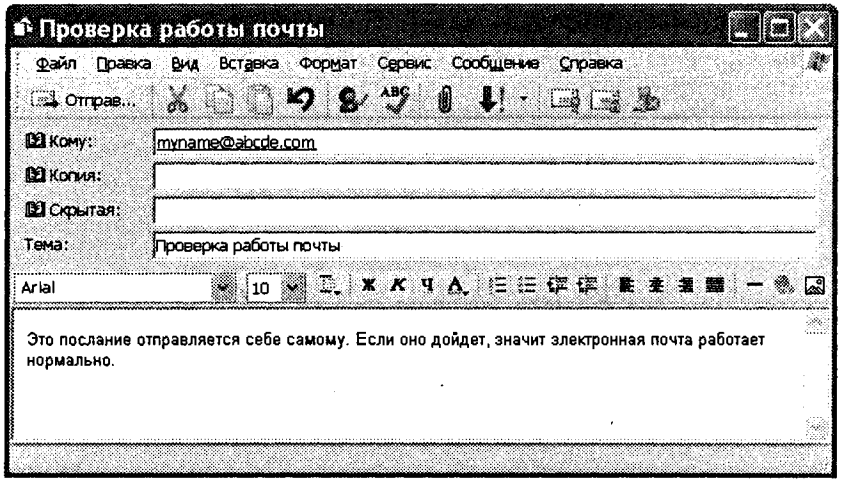


Рис. 9.8. Создание сообщения для отправки по электронной почте

осуществляются по щелчку на кнопке Отправить в окне создания сообщения или по щелчку на кнопке Доставить почту в основном окне программы *Outlook Express*.

Сообщения электронной почты размещаются в системе «внутренних» папок программы *Outlook Express*. Поступившие сообщения заносятся в папку Входящие. Открыв эту папку щелчком на ее значке на панели Папки, можно увидеть в правой области список поступивших сообщений. Если выбрать щелчком любое из сообщений, его содержание отобразится в области, расположенной ниже списка. Двойной щелчок позволяет открыть и прочитать сообщение в отдельном окне.

Подготовка ответов на сообщения. Как правило, использование любых средств коммуникации подразумевает диалог. В случае электронной почты речь идет об отправке ответов на полученные сообщения. Программа *Outlook Express* включает средства, упрощающие подготовку таких ответов. Открыв полученное сообщение в отдельном окне, можно использовать кнопки на панели инструментов.

- Кнопка Ответить отправителю служит для ответа автору письма. При этом в окне создания сообщения автоматически заполняются поля Кому и Тема, а в «тело» сообщения заносится текст исходного сообщения, что позволяет привязать комментарии непосредственно к отдельным фразам полученного письма.
- Кнопка Ответить всем служит для отправки ответа автору письма, а также всем, кто получил исходное сообщение. В окне создания сообщения автоматически заполняются поля Кому, Копия и Тема. Текст исходного сообщения копируется в тело сообщения.
- Кнопка Переслать позволяет отправить полученное сообщение (вместе с комментариями, если необходимо) другому корреспонденту. В данном случае автоматически заполняется только поле Тема, так как нового адресата необходимо указать дополнительно.

Чтение сообщений телеконференций. Механизм чтения сообщений телеконференций примерно тот же, что и при использовании электронной почты. После создания учетной записи для сервера новостей на панели Папки появляется значок, соответствующий выбранному серверу. После выбора этого значка автоматически открывается диалоговое окно Подписка на группу новостей, а программа получает список телеконференций, поддерживаемых данным сервером. Выбрав телеконференцию, следует щелкнуть на кнопке Подписаться. Телеконференции с подпиской отображаются непосредственно на панели Папки, и для доступа к ним не требуется открывать диалоговое окно Подписка на группу новостей.

Работа с сообщениями телеконференций осуществляется примерно так же, как с сообщениями электронной почты. При просмотре сообщения в отдельном окне можно Ответить в группу (отправить отклик в телеконференцию), Ответить автору (сообщение отправляется непосредственно автору по электронной почте) или Переслать сообщение по электронной почте другому корреспонденту.

Работа с адресной книгой

При активном использовании электронной почты общее число корреспондентов может достигать многих сотен. Помнить все электронные адреса просто немыслимо. Облегчить эту работу позволяет специальная программа Адресная книга.

С ее помощью можно:

- запоминать адреса корреспондентов, от которых поступили сообщения;
- автоматизировать ввод адресов корреспондентов;
- организовать проверку правильности введенных адресов;
- упростить отправку сообщений группам адресатов.

Открывать Адресную книгу вручную (Пуск ▸ Программы ▸ Стандартные ▸ Адресная книга) требуется только для ее редактирования. Чтобы добавить нового адресата, следует щелкнуть на кнопке Контакты и выбрать в открывшемся меню пункт Создать контакт. Откроется диалоговое окно Свойства (позже в заголовке будет указано имя корреспондента), содержащее многочисленные вкладки, предназначенные для ввода разнообразной информации об адресате. Имя и адрес электронной почты задаются на вкладке Имя. Удобно использовать также поле Псевдоним: данные, введенные в это поле, можно указывать вместо адреса в ходе создания сообщения.

Если информация о корреспонденте поступила вместе с полученным от него сообщением, то занести эти данные в Адресную книгу можно непосредственно из программы *Outlook Express*. Для этого надо щелкнуть правой кнопкой мыши на имени адресата в поле От в списке сообщений или в окне сообщения и выбрать в контекстном меню команды Добавить отправителя в адресную книгу или Добавить в адресную книгу соответственно.

Чтобы воспользоваться Адресной книгой для ввода адреса, надо в ходе создания сообщения щелкнуть на заголовке соответствующего поля (Кому, Копия или Скрытая). Адреса, помещаемые в каждое из этих полей, выбираются в диалоговом окне Выбрать получателей.

Адрес, взятый из Адресной книги, выделяется в соответствующем поле подчеркиванием. Если какие-то из адресов вводились вручную, но должны быть в Адресной книге, их можно проверить при помощи команды Сервис ▸ Проверить имена. Найденные адреса также будут подчеркнуты, ненайденные можно исправить, выбрав один из нескольких подходящих адресов, или занести в Адресную книгу.

Если необходимо регулярно отправлять сообщение одной и той же группе корреспондентов, Адресная книга позволяет создать и использовать *группу адресов*. Для этого используется команда Создать ▸ Создать группу. При добавлении участников в группу их адреса могут выбираться из Адресной книги или создаваться на месте. При указании в поле адреса имени группы сообщение отправляется всем выбранным корреспондентам.

Включение корреспондента в группу не влияет на возможность индивидуального использования его адреса. Один корреспондент может быть включен в несколько групп.

Практическое занятие

Упражнение 9.1. Поиск информации по ключевым словам

15 мин

1. Запустите программу *Internet Explorer* (Пуск ▸ Программы ▸ Internet Explorer).
2. На панели Адрес введите: http://www.yandex.ru/ и щелкните на кнопке Переход.
3. Внимательно рассмотрите загруженную страницу, найдите поле для ввода ключевых слов и кнопку запуска поиска. Мы собираемся искать *Web*-страницы, посвященные простым механизмам.
4. В поле для ввода ключевых слов введите простые механизмы.
5. Щелкните на кнопке Найти.
6. Просмотрите результаты поиска.
7. Щелкните на гиперссылке с номером 1.
8. Просмотрите загруженную страницу.
9. Поисковая система Яндекс всегда открывает найденную страницу в отдельном окне. Закройте это окно и вернитесь к результатам поиска.
10. Повторяя действия пп. 7–9, просмотрите всю первую группу из десяти ссылок на найденные страницы. Сколько из этих страниц все еще существуют? Сколько из них можно считать полезными?
11. Щелкните на кнопке Поиск на панели инструментов.
12. Введите набор ключевых слов из п. 4 в поле панели Поиск.
13. Щелкните на кнопке начала поиска.
14. Сравните результаты поиска.
15. На панель Адрес введите слово find и набор ключевых слов из п. 4. Щелкните на кнопке Переход.

16. Объясните, что произошло.

Мы научились проводить поиск информации в Интернете тремя разными способами: с помощью поисковой системы, с помощью панели Поиск и непосредственно с панели Адрес. Мы узнали, в чем состоят особенности поиска по ключевым словам.

Упражнение 9.2. Использование папки Избранное

1. Запустите программу *Internet Explorer*.
2. На панели Адрес введите: http://www.parispourvous.net/index.php?wpe=a16 (или другой адрес по указанию преподавателя) и щелкните на кнопке Переход.
3. Просмотрите загруженную страницу.
4. Щелкните в рабочей области программы правой кнопкой мыши и выберите в контекстном меню команду Добавить в Избранное.
5. В поле Имя введите: Экспериментальная страница.
6. Щелкните на кнопке OK.
7. Щелкните на кнопке Домой на панели инструментов.
8. Дайте команду Избранное ▸ Экспериментальная страница.
9. Убедитесь, что в папке Избранное действительно была сохранена информация о загружаемой странице.
10. Дайте команду Избранное ▸ Упорядочить избранное. Щелкните на кнопке Создать папку. Дайте новой папке имя Материалы.
11. Выберите пункт Экспериментальная страница. Щелкните на кнопке Переместить.
12. В диалоговом окне Обзор папок выберите папку Материалы, после чего щелкните на кнопке OK.
13. Закройте диалоговое окно Упорядочить избранное и программу *Internet Explorer*. Разрывать соединение с Интернетом не следует!
14. Дайте команду Пуск ▸ Избранное ▸ Материалы ▸ Экспериментальная страница.
15. Ознакомьтесь с тем, какая страница при этом загружается.
16. Уничтожьте папку Материалы и все ее содержимое.

Мы научились сохранять информацию о полезных Web-страницах в папке Избранное. Мы также узнали, как изменять структуру папок, вложенных в папку Избранное, и познакомились с различными способами загрузки избранных Web-страниц.

Упражнение 9.3. Загрузка файла из Интернета

1. Запустите программу *Internet Explorer*.
2. На панели Адрес введите: ftp://ftp.microsoft.com/ и щелкните на кнопке Переход.

3. Внимательно рассмотрите способ представления каталога архива *FTP* в программе *Internet Explorer*. Обратите внимание на то, как выглядит значок в строке адреса.
4. Двойными щелчками на значках папок откройте папку /Products/Windows/Windows95/CDRomExtras/FunStuff/.
5. Дважды щелкните на значке clouds.exe.
6. В открывшемся диалоговом окне Загрузка файла щелкните на кнопке Сохранить.
7. В диалоговом окне Сохранить как выберите папку, специально отведенную для хранения загруженных файлов, и задайте имя файла.
8. Сбросьте в диалоговом окне загрузки файла флажок Закрыть диалоговое окно после завершения загрузки.
9. Следите за ходом загрузки файла по этому диалоговому окну.
10. Когда загрузка файла завершится, закройте диалоговое окно, информирующее о завершении загрузки, с помощью кнопки Закрыть.
11. Откройте папку, в которой был сохранен загруженный файл, при помощи программы Проводник.
12. Убедитесь, что загруженный файл можно использовать в соответствии с его назначением.

Мы научились просматривать каталоги FTP и загружать файлы из Интернета. Механизм загрузки файлов работает практически одинаково при загрузке с Web-узла и из архива FTP.

Упражнение 9.4. Настройка отображения объектов

15 мин

1. Запустите программу *Internet Explorer*.
2. На панели Адрес введите: http://elfwood.lysator.liu.se/ (или другой адрес по указанию преподавателя).
3. Щелкните на гиперссылке Click here to enter...
4. Зафиксируйте с помощью секундомера время загрузки страницы.
5. Посмотрите, как выглядит загруженная страница.
6. Щелкните на кнопке Назад на панели инструментов.
7. Дайте команду Сервис ▸ Свойства обозревателя.
8. Откройте вкладку Дополнительно.
9. Сбросьте флажки Воспроизводить анимацию на веб-страницах, Воспроизводить звуки на веб-страницах, Воспроизводить видео на веб-страницах и Отображать рисунки.
10. Выберите вкладку Общие.
11. Щелкните на кнопке Удалить файлы.
12. Щелкните на кнопке OK.

13. Опять щелкните на гиперссылке Click here to enter...
14. Еще раз зафиксируйте с помощью секундомера время загрузки страницы.
15. Сравните результаты измерений.
16. Сравните внешний вид страницы при предыдущей и нынешней загрузке.
17. Щелкните на одной из пустых рамок для рисунков правой кнопкой мыши и выберите в контекстном меню команду Показать рисунок.

Мы научились ускорять загрузку Web-страниц ценой отказа от отображения рисунков и других объектов. Мы узнали, как индивидуально загружать нужные объекты. В ходе упражнения мы также выяснили, как очистить пространство на диске, занятое временными файлами Интернета.

Упражнение 9.5. Создание учетной записи электронной почты

15 мин

1. Запустите программу *Outlook Express*.
2. Дайте команду Сервис ▸ Учетные записи.
3. Щелкните на кнопке Добавить и выберите в открывшемся меню пункт Почта.
4. В поле Введите имя введите свои имя и фамилию.
5. Щелкните на кнопке Далее.
6. Введите в поле Электронная почта заданный адрес электронной почты.

Необходимую информацию предоставляет преподаватель.

7. Щелкните на кнопке Далее.
8. Введите заданные имена серверов для входящей и исходящей почты. Если используется один сервер, введите одно и то же имя в оба поля.
9. Щелкните на кнопке Далее.
10. Введите заданные имя пользователя и пароль для доступа к электронной почте. Установите флажок Запомнить пароль.
11. Щелкните на кнопке Далее. Щелкните на кнопке Готово.
12. Откройте вкладку Почта. Убедитесь, что учетная запись действительно создана. Щелкните на кнопке Закрыть.

Мы научились создавать учетную запись для электронной почты, используемую при отправке и получении корреспонденции. Мы узнали, какие данные потребуются для создания учетной записи.

Упражнение 9.6. Отправка и получение сообщения электронной почты

15 мин

1. Запустите программу *Outlook Express*.
2. На панели Папки выберите папку Входящие.
3. Щелкните на кнопке Создать сообщение на панели инструментов.

4. В поле Тема введите слова: Проверка работы электронной почты.
5. В поле Кому введите заданный (свой собственный) адрес электронной почты.
6. В тело сообщения введите произвольный легко запоминающийся текст.
7. Щелкните на кнопке Отправить на панели инструментов.
8. Щелкните на кнопке Доставить почту на панели инструментов.
9. Проследите за процессом отправки созданного сообщения и поиском на сервере поступивших сообщений. Ход этих действий отображается в открывшемся окне.
10. Убедитесь, что только что отправленное сообщение появилось в списке поступивших сообщений.
11. Выберите это сообщение в списке и ознакомьтесь с его содержанием на нижней панели.
12. Дважды щелкните на заголовке сообщения, чтобы открыть его в отдельном окне.
13. Закройте окно сообщения.

Мы научились создавать сообщения электронной почты, отправлять их, а также получать и читать поступившие сообщения. Мы также познакомились с тем, как организуется процесс передачи сообщений с компьютера пользователя на почтовый сервер и в обратную сторону.

Упражнение 9.7. Подписка на телеконференцию и чтение сообщений

15 мин

1. Запустите программу *Outlook Express*.
2. Щелкните на значке сервера новостей на панели Папки.
3. Если диалоговое окно Подписка на группу новостей не откроется автоматически, щелкните на кнопке Группы новостей на панели инструментов или на правой панели.
4. В поле Отобразить группы новостей, содержащие введите: comp.os.
5. В общем списке телеконференций выберите телеконференцию:

   comp.os.ms-windows.programmer.tools.misc

6. Щелкните на кнопке Подписаться.
7. Щелкните на кнопке OK.
8. Щелкните на значке выбранной телеконференции на панели Папки.
9. Дождитесь загрузки блока сообщений.
10. Включите режим группировки по обсуждениям командой Вид ▸ Текущее представление ▸ Сгруппировать сообщения по теме обсуждения.
11. Выберите какое-либо сообщение, чтобы просмотреть его. Щелкните на сообщении дважды, чтобы открыть его в отдельном окне.
12. Закройте окно сообщения.

13. Дайте команду Сервис ▸ Следующие 300 заголовка (-ов), чтобы загрузить следующую порцию сообщений телеконференции.

Мы научились производить подписку на телеконференцию (группу новостей). Мы также узнали, как получать заголовки сообщений с сервера и читать имеющиеся сообщения.

Упражнение 9.8. Использование Адресной книги

1. Запустите программу *Outlook Express*.
2. Создайте вручную запись в Адресной книге для своего адреса электронной почты. Для этого на панели Контакты дайте команду Контакты ▸ Создать контакт.

Занесение собственного адреса в Адресную книгу применяют при использовании средств шифрования переписки и механизма цифровой подписи.

3. На вкладке Имя введите свою фамилию, имя, отчество.
4. Укажите адрес электронной почты и щелкните на кнопке Добавить.
5. По желанию занесите данные о себе также на вкладки Домашние, Служебные и Личные.
6. Щелкните на кнопке OK.
7. Убедитесь, что данные занесены в Адресную книгу, — на панели Контакты должен добавиться новый значок.
8. Откройте папку Входящие.
9. Щелкните на заголовке входящего сообщения правой кнопкой мыши и выберите в контекстном меню пункт Добавить отправителя в адресную книгу.
10. Убедитесь, что данные занесены в Адресную книгу, — на панели Контакты должен добавиться новый значок.
11. Щелкните на кнопке Создать сообщение на панели инструментов.
12. Щелкните на кнопке Кому рядом с полем ввода адреса.
13. Выберите собственный адрес в списке слева и щелкните на кнопке Кому.
14. Щелкните на кнопке OK.
15. Убедитесь, что имя адресата внесено в поле Кому. Обратите внимание, что использовано именно имя, а не адрес. Обратите внимание, что имя подчеркнуто. Это означает, что данный адрес считается «проверенным».
16. Произвольным образом заполните поле Тема, а также введите текст сообщения.
17. Отправьте созданное сообщение и убедитесь, что оно доставляется правильно (см. упражнение 9.6).

Мы научились заносить адреса электронной почты в Адресную книгу, вручную и извлекая их из поступивших сообщений. Мы также узнали, как использовать данные из Адресной книги при отправке сообщений, и выяснили, что Адресная книга позволяет использовать имена корреспондентов вместо адресов электронной почты.

В этой и следующей главе рассматриваются понятия, методы и приемы, относящиеся к созданию текстовых документов с помощью персонального компьютера. Условно (из чисто методических соображений) мы выделим две группы создаваемых документов — *простые* и *комплексные*. Первые представляют собой форматированный текст, а вторые содержат кроме текста объекты иной природы (чертежи, рисунки, формулы, таблицы, объекты мультимедиа и прочие).

10.1. Общие сведения о текстовом процессоре Microsoft Word

Общее название программных средств, предназначенных для создания, редактирования и форматирования простых и комплексных текстовых документов, — *текстовые процессоры*. В настоящее время в России наибольшее распространение имеет текстовый процессор *Microsoft Word*. Это связано, прежде всего, с тем, что его создатели относительно давно предусмотрели *локализацию* программы в России путем включения в нее средств поддержки работы с документами, исполненными на русском языке.

Основные версии текстового процессора Microsoft Word

Первоначальные версии текстового процессора *Microsoft Word* относятся к восьмидесятым годам и, соответственно, к операционной системе *MS-DOS*. Последней версией процессора для неграфической операционной среды была версия *Microsoft Word 5.0*. Она позволяла создавать, редактировать и распечатывать форматированные текстовые документы.

Поскольку операционная система *MS-DOS* не является графической, данная версия программы не могла соблюдать принятый ныне принцип соответствия экранного изображения печатному (принцип *WYSIWYG*) и операции форматирования документа выполнялись в известной степени «вслепую». Однако возможность просмотра документа в «натуральном» виде все-таки была. Она реализовывалась

специальным режимом *предварительного просмотра* (*preview*), который сохранился и в современных версиях программы, хотя и не имеет уже решающего значения.

Основным преимуществом текстового процессора *Word 5.0*, отличавшим эту программу от конкурентных продуктов, была возможность встраивания в текст графических объектов, правда, без взаимодействия текста и графики (*обтекания графических изображений текстом*). Сегодня текстовым процессором *Word 5.0* еще иногда пользуются при работе на устаревшем оборудовании (*IBM PC AT/286*).

Принцип *WYSIWYG* впервые был реализован в следующей версии программы, которая называлась *Microsoft Word for Windows* (*Word 6.0*). Благодаря этому принципу значительно упростились и стали наглядными приемы форматирования документов. Будучи приложением *Windows* 3.1, программа получила возможность использовать системный буфер обмена, а пользователи получили мощное и удобное средство для создания комплексных документов.

Следующая версия программы называлась *Microsoft Word 95* (*Word 7.0*). Она была ориентирована на графическую операционную систему *Windows 95*. Основным достижением этой версии стало то, что после нее текстовый процессор уже не рассматривается только как отдельное приложение. В состав мощного офисного пакета *Microsoft Office* входит несколько приложений (с каждой новой версией пакета этот состав расширяется), и на процессор *Microsoft Word* возлагаются дополнительные функции интеграции прочих приложений. Он занимает центральное положение в системе и позволяет организовать эффективный обмен данными между составляющими приложениями, что позволило в значительной степени автоматизировать разработку офисных документов разной содержательности и сложности.

Еще одним важным нововведением седьмой версии стало управление взаимодействием текста со встроенными объектами, что значительно расширило набор возможностей при форматировании документов. А особенный успех этой версии программы в России (она очень широко используется и сегодня) завоевали встроенные средства поддержки русского языка (автоматическая проверка орфографии и грамматики).

Восьмая версия программы *Microsoft Word 97* (*Word 8.0*), вошедшая в состав пакета *Microsoft Office 97*, внесла относительно мало практически полезных изменений для повседневной офисной работы. Так, например, ее жесткая ориентация на использование шрифтов *UNICODE* затруднила обмен данными с большинством приложений, выпущенных «третьими» фирмами, и создала пользователям проблемы при печати материалов на большинстве печатающих устройств. Дополнительные средства оформления текстовых документов, представленные в этой версии, имели практическое значение только при разработке электронных (экранных) документов. Возможность сохранения документов в «электронных» форматах *HTML* и *PDF*, рассчитанная на публикацию документов в Интернете, осталась проработанной не до конца и не вошла в практику *Web*-дизайнеров.

Начиная с этой версии текстовый процессор *Microsoft Word* можно рассматривать как *средство автоматизации авторской деятельности* (*authoring system*). При использовании этой программы следует четко определять целевой объект — документ *электронный* или *печатный*. Для разных типов документов используют раз-

ные средства, приемы и методы. Применение неадекватных средств значительно усложняет последующие этапы работы с документом. В итоге в качестве средства разработки электронных документов *Word 97* не заменил *Web*-редакторы, а в качестве средства создания печатных документов внес ряд неудобств по сравнению с предыдущей версией *Word 95*.

Очередной стала версия текстового процессора *Microsoft Word 2000* (*Word 9.0*), входящая в состав пакета *Microsoft Office 2000*. В ней устранены основные недостатки предыдущей версии, заметно улучшена система управления и введены мощные средства поддержки сетевых режимов работы. Предполагается, что основным стилем производительной работы с текстовым процессором *Word* 2000 должна стать совместная деятельность рабочих групп над общими проектами в рамках корпоративных сетей.

Последней (ко времени подготовки данного пособия) является версия текстового процессора *Microsoft Word XP* (*Word 10.0*). Они входит в состав пакета *Microsoft Office XP*. В ней заметно расширены средства работы со стилями и шаблонами, введены механизмы, позволяющие автоматически обеспечить единство оформления документа.

Рабочее окно процессора Microsoft Word 2000

Рабочее окно процессора *Microsoft Word XP* представлено на рис. 10.1. Его основные элементы управления: строка меню, панель инструментов, рабочее поле и строка состояния, включающая индикаторы. Начиная с процессора *Microsoft Word 95*, панель инструментов является настраиваемой.

Режимы отображения документов

Начиная с шестой версии, текстовый процессор *Microsoft Word* поддерживает несколько режимов представления документов.

В *обычном режиме* представляется только содержательная часть документа без реквизитных элементов оформления, относящихся не к тексту, а к печатным страницам (колонтитулы, колонцифры, подстраничные сноски и т. п.). Этот режим удобен на ранних этапах разработки документа (ввод текста, редактирование, рецензирование), а также во всех случаях, когда содержательная часть документа имеет более высокое значение, чем внешнее представление. В этом режиме операции с объемными документами проходят быстрее, что важно при работе на малопроизводительных компьютерах.

В *режиме Web-документа* экранное представление не совпадает с печатным. Это отступление от принципа *WYSIWYG*, но оно характерно для электронных публикаций в *World Wide Web*, поскольку заранее не известно, каким средством просмотра и на каком оборудовании будет отображаться документ. Понятие печатной страницы для электронных документов не имеет смысла, поэтому назначенные параметры страницы не учитываются, а форматирование документа на экране является *относительным*. В этом режиме разрабатывают электронные публикации.

В *режиме разметки* экранное представление документа полностью соответствует печатному, вплоть до назначенных параметров печатной страницы. Этот режим

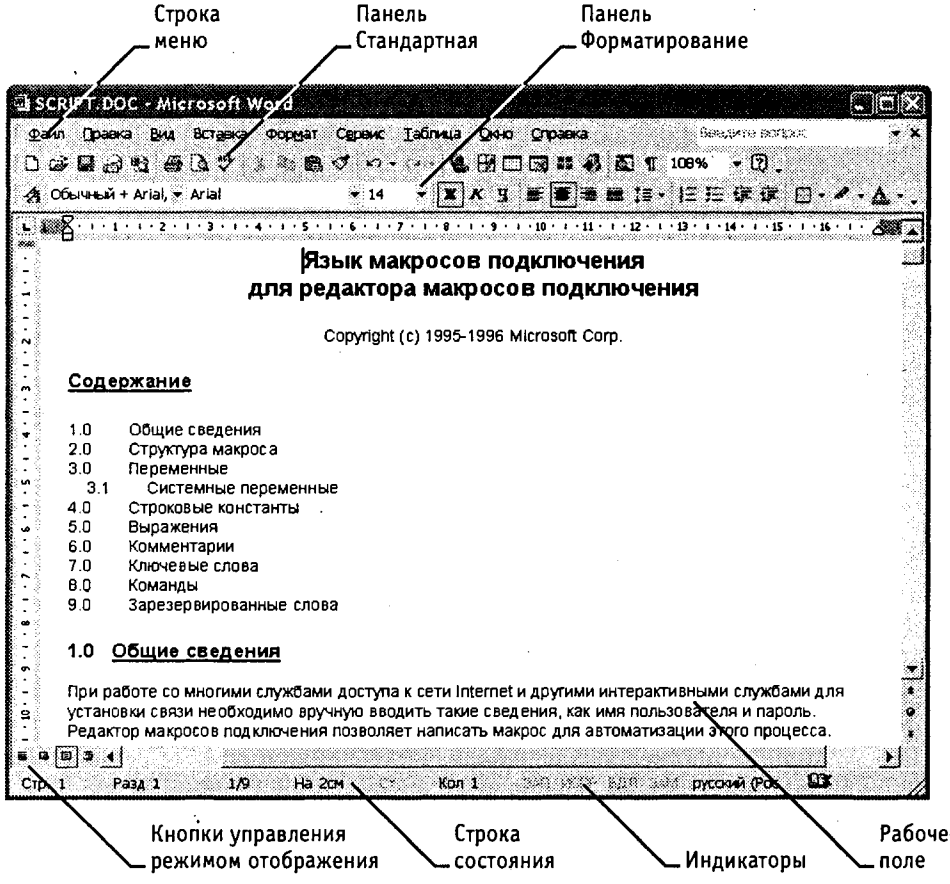


Рис. 10.1. Рабочее окно программы Word XP

удобен для большинства работ, связанных с форматированием текста, предназначенного для печати.

В *режиме структуры* можно отобразить только заголовки документа. Режим полезен в тех случаях, когда разработку документа начинают с создания плана содержания. Если предполагаемый размер документа превышает 5–7 печатных страниц, следует начинать работу именно с создания первичного плана. Режим структуры отличается тем, что при его включении автоматически открывается вспомогательная панель инструментов Структура, элементы управления которой позволяют править структуру документа.

Выбор одного из четырех указанных режимов представления документа выполняют с помощью командных кнопок, расположенных в левом нижнем углу окна приложения, или командами меню Вид.

Через меню Вид доступно также специальное представление (пятый режим) Схема документа, при котором окно приложения имеет две рабочие панели. На левой панели

представляется структура документа, а на правой — сам документ. Этот режим, сочетающий достоинства режима разметки и режима структуры, полезен при навигации по объемному документу — его удобно использовать не при создании, а при просмотре документов сложной структуры.

Через меню Файл доступны еще два режима представления документа, используемые для предварительного просмотра. Для электронных документов используют команду Файл ▸ Предварительный просмотр Web-страницы, а для печатных документов — Файл ▸ Предварительный просмотр. В первом случае созданный документ отображается как *Web*-страница в окне браузера, зарегистрированного операционной системой в качестве принятого по умолчанию (желательно, чтобы это был браузер *Microsoft Internet Explorer* 6.0). Во втором случае документ представляется в специальном окне.

Приемы работы с командами строки меню

Как и в большинстве других приложений, корректно соблюдающих идеологию *Windows*, строка меню текстового процессора *Microsoft Word XP* как элемент управления отличается тем, что обеспечивает доступ ко всем функциональным возможностям программы. Не всегда этот доступ самый удобный, во многих случаях другие элементы управления использовать проще, но строка меню удовлетворяет *принципу функциональной полноты*.

Меню, открывающиеся из строки меню, обладают свойством *функциональной автонастройки*. Расширенные возможности приложения не могли не отразиться в изобилии элементов управления, открываемых через строку меню. В нем не всегда удобно ориентироваться. Поэтому пункты строки меню открываются в два приема. На первом этапе открывают *сокращенное меню*, и, если необходимого элемента управления в нем нет, открывают *расширенное меню* наведением указателя мыши на *пункт раскрытия*. Использованные пункты расширенной части меню далее открываются в составе сокращенного меню (рис. 10.2).

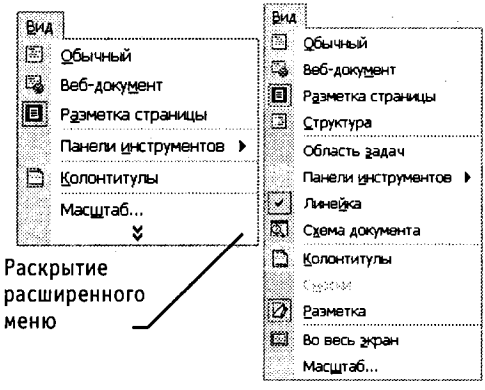


Рис. 10.2. Команды меню Вид, сокращенный и расширенный вариант

Панели инструментов Microsoft Word XP

Начиная с седьмой версии, программа *Microsoft Word* поддерживает возможность самостоятельной настройки панелей инструментов. Настройку выполняет пользователь путем подключения функциональных панелей, необходимых ему по роду деятельности (Вид ▸ Панели инструментов). Расширение общей панели инструментов сопровождается некоторым уменьшением площади рабочего окна документа. Перемещение функциональных панелей производят методом перетаскивания за рубчик, расположенный на левом краю панели.

В последних версиях текстового процессора панели инструментов не только допускают настройку, но и обладают контекстной чувствительностью. Так, при выделении в поле документа какого-либо объекта, автоматически открывается панель инструментов, предназначенная для его редактирования. Назначение панелей инструментов приведено в таблице 10.1.

Таблица 10.1. Панели инструментов программы Word XP

Панель инструментов	Состав, назначение	Примечание
Стандартная	Элементы управления файловыми операциями, редактированием, экранным отображением	Устанавливается по умолчанию
Форматирование	Элементы управления форматированием документа	Устанавливается по умолчанию
Visual Basic	Доступ к средствам создания и редактирования макросов и Web-сценариев, а также к настройке средств обеспечения безопасности при запуске макросов	Макросы служат для автоматизации типовых операций. Web-сценарии обеспечивают динамичный характер просмотра Web-страниц
Word-Art	Элементы управления для создания художественных заголовков	
Автотекст	Средство быстрого доступа к настройке функции автотекста	Одновременно предоставляет быстрый доступ к средствам настройки функций автозамены и автоформата
База данных	Элементы управления, характерные для работы с базами данных (сортировка, поиск, управление структурой таблиц и прочее)	В качестве базы данных могут выступать как таблицы Access, так и собственные таблицы Word
Веб-компоненты	Комплект готовых компонентов для создания элементов управления Web-страницы или электронной формы	Применяются для создания обратной связи с потребителем документа (опросные листы, анкеты, бланки заказов и заявок и прочее)
Веб-узел	Элементы управления для навигации в Web-структурах данных	В качестве Web-структур могут выступать World Wide Web, корпоративные сети intranet, системы Web-документов локального компьютера
Настройка изображения	Элементы управления для основных функций настройки растровых изображений	Позволяют настраивать яркость, контрастность, размер, рамку, режимы обтекания текстом и прочие параметры выделенного растрового объекта
Рамки	Элементы управления для создания фреймов (не путать с рамками, создаваемыми с помощью панели инструментов Таблицы и границы)	Текстовый процессор Word XP поддерживает два типа фреймов. Фреймы в электронных документах представляют собой особые прямоугольные области, предназначенные для вывода нескольких Web-документов в рамках одной Web-страницы. Фреймы в печатных документах представляют особые области печатной страницы для вывода специальной информации, например колонтитулов

Таблица 10.1. Панели инструментов программы Word 2000 (окончание)

Панель инструментов	Состав, назначение	Примечание
Рецензирование	Элементы управления для проведения редактирования и комментирования документов без искажения исходного текста	Измененные данные сохраняются в том же документе на правах новых версий. Автор исходного текста имеет возможность просмотреть замечания и предлагаемые изменения, после чего принять их или отвергнуть
Рисование	Элементы управления и инструменты для выполнения простейших чертежно-графических работ	Графические объекты, создаваемые инструментами данной панели, имеют характер векторных объектов
Слияние	Инструменты для работы с документами слияния, содержащими постоянную и переменную части	Используется при использовании программы Word XP, например, для массовой подготовки писем аналогичного содержания
Статистика	Позволяет получить информацию об объеме документа	Сведения о числе знаков, слов, строк, абзацев, страниц
Структура	Инструменты для работы с логической структурой документа	Позволяет управлять заголовками и порядком следования логических частей текста. Активно используется при работе с документом в режиме структуры
Таблицы и границы	Элементы управления для создания таблиц и оформления текстовых блоков рамками	Дополнительно предоставляет средства для сортировки данных и проведения итоговых расчетов в таблицах (функция Автосумма)
Формы	Элементы управления для разработки стандартных форм	Программа Word XP позволяет создавать три типа форм: Web-формы, являющиеся объектами Web-страниц; формы Word, распространяемые и заполняемые как электронный документ; печатные формы
Элементы управления	Набор готовых компонентов ActiveX для создания элементов управления Web-страниц и Web-форм	Средства данной панели инструментов позволяют не только использовать около 150 готовых компонентов, но и проводить установку и регистрацию дополнительных компонентов ActiveX

Кроме того, в программе *Word XP* роль специальной контекстно-зависимой информационно-инструментальной панели играет область задач. Эта область открывается на правах панели инструментов и обычно располагается у правого края окна программы. Область задач может использоваться для выполнения разных функций, в зависимости от выбранного режима. Выбор режима осуществляется из меню, открывающегося при щелчке на треугольной кнопке в верхней строке Область задач. Кроме того, существуют специальные команды для открытия области задач в нужном режиме (например, Правка ▸ Буфер обмена Office или Формат ▸ Показать форматирование). Режимы работы области задач описаны в таблице 10.2.

Таблица 10.2. Режимы области задач

Режим	Команда	Содержание Области задач	Назначение
Создание документа	Файл ▸ Создать	Список недавно открывавшихся документов, команды создания новых документов	Открытие существующих и создание новых документов
Буфер обмена	Правка ▸ Буфер обмена Office	Содержание буфера обмена Office (до 24 объектов)	Выбор объектов для вставки в документ
Поиск	Файл ▸ Найти	Команды для поиска и информация о результатах поиска	Поиск текста в форматированных файлах
Вставка картинки	Вставка ▸ Рисунок ▸ Картинки	Команды для поиска клипартов и информация о результатах поиска	Выбор клипартов и других графических изображений для вставки в документ
Стили и форматирование	Формат ▸ Стили и форматирование	Сведения о стилях и форматировании текста и средства для их изменения	Выбор и создание стилей на основе существующего оформления текста
Показать форматирование	Формат ▸ Показать форматирование	Сведения о характеристиках форматирования в месте расположения курсора	Информация о форматировании, сравнение форматов разных фрагментов
Слияние	Сервис ▸ Письма и рассылки ▸ Мастер слияния	Этапы создания документа слияния	Создание документов слияния (например, писем), содержащих постоянную и переменную части
Перевод	Сервис ▸ Язык ▸ Перевод	Исходный текст и результат перевода	Перевод отдельных слов и коротких фраз

Основные принципы практической работы с текстовым процессором Microsoft Word

Основные принципы практической работы зависят от используемой версии программы. Базовый принцип здесь состоит в том, что чем больше возможностей имеет программа, тем строже надо подходить к выбору тех функций, которыми можно пользоваться в каждом конкретном случае. Удобен подход, когда набор допустимых средств оформления и форматирования документа определяет его заказчик.

Заказчик есть у каждого документа. Даже если документ готовится для личного употребления, условным заказчиком является сам автор. Заказчиком можно считать и исполнителя, которому передается документ для последующих операций, например, для рецензирования или вывода на печать. К категории «заказчиков» относятся и предполагаемые клиенты, для которых данный документ разрабатывается. При этом возникает ряд вопросов, которые надо решить до начала работы с документом.

К какому типу относится документ? Современные текстовые процессоры позволяют создавать документы трех типов. Во-первых, это *печатные документы*, кото-

рые создаются и распечатываются на одном рабочем месте или в одной рабочей группе. Дальнейшее движение документа происходит только в бумажной форме. Состав допустимых средств оформления в данном случае определяется только техническими возможностями печатающего устройства.

Второй тип — *электронные документы в формате текстового процессора*, например *Microsoft Word*. Такие документы передаются заказчику в виде файлов. Электронный документ, как правило, не является окончательным. В большинстве случаев заказчик может его дорабатывать, редактировать, форматировать, распечатывать или использовать его компоненты для подготовки своих документов (книг, журналов, сборников статей и т. п.). Набор разрешенных средств в данном случае, как правило, минимален и определяется заказчиком.

Третий тип — *Web-документы*. Предполагается, что в этом качестве они останутся навсегда, и их преобразование в печатные документы не планируется. В *Web*-документах большую роль играет управление цветом. Для этой категории документов наиболее широк выбор средств форматирования и оформления.

Кто является заказчиком документа? Самый типичный случай — когда заказчиком документа является работодатель, то есть администрация предприятия или учреждения. Надо выяснить правила оформления документов, принятые в данной организации, и строго их придерживаться. Если существуют готовые шаблоны, их надо использовать, а если их нет, то разработать свои и согласовать с руководством.

Самый простой случай — когда заказчика нет, и автор делает документ для себя. Он может использовать любые средства, которые ему подскажет фантазия и которые поддерживаются его устройствами вывода (экран для *Web*-документов или принтер для печатных документов).

Самый трудный случай — когда заказчик внешний, особенно если он не вполне определен. В этом случае исполнители часто путают понятия *представление* документа и *предоставление* документа. Для *представления* документа они стремятся использовать все средства форматирования, которые наилучшим образом подчеркивают достоинства документа. При *предоставлении* документа ситуация обратная. Здесь не автор, а заказчик определяет форму и средства форматирования и оформления. Использование этих средств в данном случае имеет *разрешительный характер*. Если же требования заказчика еще не известны, следует предполагать, что нежелательны большинство средств форматирования документов, передаваемых для дальнейшей обработки. В частности, необходимо:

- ограничить используемые наборы шрифтов только теми, которые входят в состав операционной системы (не более двух наборов: один — для основного текста, другой — для заголовков и вспомогательного текста);
- минимизировать использование средств форматирования абзацев: отказаться от выравнивания по ширине и от переноса слов, ограничить число используемых шрифтовых начертаний (не более двух: основного и дополнительного);
- отключить все автоматические средства форматирования: расстановку колонтитулов, нумерацию страниц, маркировку и нумерацию списков и прочие;

- не использовать встроенные средства текстового процессора для создания встроенных объектов (художественные заголовки, векторные рисунки, рамки и прочие) — все объекты должны создаваться специальными программами, храниться в отдельных файлах, вставляться в текст документа методом связывания и прилагаться к файлу документа;
- исключить использование приемов взаимодействия встроенных объектов с текстом;
- сохранять готовые документы в простейших форматах, несущих минимум информации о форматировании (для документов *Microsoft Word* таковыми являются форматы Только текст, Текст в формате RTF или Word 6.0/95);
- в каждом случае отступления от этих правил, например при необходимости использовать формулы, таблицы и специальные символы, согласовывать свои действия с заказчиком.

Эти требования к документам, предоставляемым для дальнейшей технологической обработки, связаны с тем, что большинство средств оформления и форматирования текстового процессора являются «вещью в себе». Достоинства этих средств проявляются только при выводе окончательного документа средствами того же самого процессора, будь то вывод на печать, просмотр на экране или публикация в *Web*-структуре. При обработке данных, содержащихся в документе, другими программными средствами преимущества форматирования и оформления могут оборачиваться тяжкими проблемами.

Первичная настройка текстового процессора Microsoft Word

Приступая к первому знакомству с текстовым процессором *Microsoft Word*, следует выполнить ряд первичных настроек. Некоторые средства автоматизации, имеющиеся в программе, могут отвлекать начинающего пользователя от главной задачи — освоения основных приемов. В ряде случаев из-за работы автоматических средств результаты операций получаются неожиданными — это препятствует установлению обратной связи и эффективному усвоению практических приемов.

Комплекс настроек, рекомендуемых перед началом освоения текстового процессора, приведен в упражнении 10.1.

10.2. Приемы работы с текстами в процессоре Microsoft Word

К базовым приемам работы с текстами в текстовом процессоре *Microsoft Word* относятся следующие:

- создание документа;
- ввод текста;
- редактирование текста;
- рецензирование текста;
- форматирование текста;
- сохранение документа;
- печать документа.

Создание документа

В текстовом процессоре *Word XP* принято использовать два метода создания нового документа: на основе готового шаблона или на основе существующего документа. Второй метод проще, но первый методически более корректен.

Создание документа на основе имеющегося документа. Этот метод потенциально опасен, и потому его использование категорически не рекомендуется! Тем не менее, им очень широко пользуются, и мы его рассматриваем, чтобы предупредить о возможных опасностях и обратить внимание на правильный порядок действий.

При создании документа на основе существующего документа:

- открывают готовый документ (Файл ▸ Открыть);
- сохраняют его под новым именем (Файл ▸ Сохранить как);
- выделяют в нем все содержимое (Правка ▸ Выделить все);
- удаляют его нажатием клавиши DELETE;
- в результате получают пустой документ, имеющий собственное имя и сохраняющий все настройки, ранее принятые для исходного документа.

Этот метод характерен для начинающих пользователей, не умеющих создавать шаблоны и пользоваться ими. Получив задание у руководителя, они запрашивают образец и приступают к его правке. Метод интуитивно прост, но чреват весьма неприятными ошибками. Если забыть сохранить новый файл под другим именем, можно легко уничтожить ценный документ, даже не успев создать новый. Кроме того, при небрежной правке содержание документа, взятого за основу, может переходить в новый документ. Для рабочих мест, на которых создаются десятки документов в сутки, этот метод весьма опасен.

Создание документа на основе шаблона. Шаблоны — это те же образцы документов, но защищенные от досадных неприятностей. Создание документа на основе готового шаблона выполняется следующим образом.

1. Команда Файл ▸ Создать открывает Область задач в режиме создания документа. Щелкните на этой панели на ссылке Общие шаблоны — откроется диалоговое окно Шаблоны. Надо включить переключатель Создать документ и выбрать подходящий шаблон. Если никаких предпочтений нет, следует выбрать шаблон Новый документ на вкладке Общие. Созданный документ приобретает имя Документ1, принятое по умолчанию. Его целесообразно сразу же сохранить под «правильным» именем, выбрав для него соответствующую папку и дав команду Файл ▸ Сохранить как.
2. Диалоговое окно Сохранение документа в текстовом процессоре *Microsoft Word XP*, представленное на рис. 10.3, практически не отличается от аналогичного окна ранее рассмотренных нами стандартных приложений. Оно обычно предполагает сохранение файла в папку \Мои документы, но обеспечивает быстрый доступ и к некоторым иным папкам.
3. В левой части окна Сохранение имеется пять кнопок, позволяющих быстро выбрать место для сохранения файла.

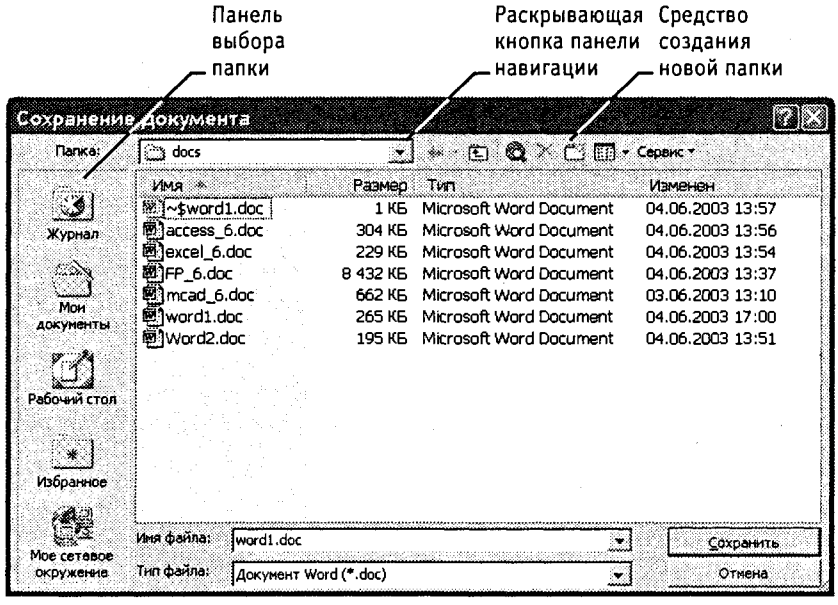


Рис. 10.3. Диалоговое окно Сохранение документа

Журнал — логическая папка. Если нужно сохранить документ в одну из папок, которой пользовались в последнее время, это очень удобное средство доступа.

Мои документы — традиционная папка для хранения авторских документов в операционных системах семейства *Windows*.

Рабочий стол — не слишком удобное место для хранения документов, поскольку его принято содержать «в чистоте». Есть два случая, когда Рабочий стол используют для хранения документов:

- если документ временный и после просмотра будет удален в Корзину;
- если документом предполагается пользоваться особенно часто (например, это список номеров телефонов коллег по работе).

Избранное — особая логическая папка пользователя, предназначенная для хранения ярлыков *Web*-страниц. Ее нецелесообразно использовать для сохранения текстовых документов, но для открытия документов она может использоваться активно.

Мое сетевое окружение — этот значок обеспечивает быстрый доступ к сохранению документа не на своем компьютере, а в локальной сети, например на файловом сервере. При этом требуются дополнительные операции по навигации, связанные с выбором конкретной папки на конкретном сетевом компьютере.

При необходимости сохранить документ в произвольную папку, не представленную в данном списке, следует выполнить навигацию по файловой структуре с использованием раскрывающей кнопки на правом краю поля Папка.

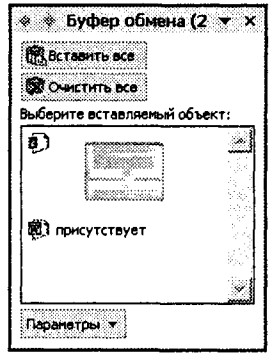
Специальные средства ввода текста

Технология ввода текста и переключения языковых раскладок клавиатуры, применение регистровых клавиш и буфера обмена *Windows* были представлены выше при описании стандартного приложения Блокнот. В данном разделе мы остановимся на особенностях текстового процессора *Microsoft Word XP*, позволяющих автоматизировать ввод текста.

Средства отмены и возврата действий. Все операции ввода, редактирования и форматирования текста протоколируются текстовым процессором, и потому необходимое количество последних действий можно отменить. Последнее действие отменяют комбинацией клавиш CTRL+Z. Эта команда имеет кумулятивный эффект: серия команд отменяет серию последних действий. Другие аналогичные средства — команда Правка ▸ Отменить действие и кнопка Отменить действие на панели инструментов Стандартная. Длинные последовательности действий можно отменять также с помощью списка действий (кнопка, раскрывающая список, присоединена к кнопке Отменить действие).

После отмены ряда действий существует возможность вернуться к состоянию, предшествовавшему отмене. Для этого служит команда Правка ▸ Вернуть действие или кнопка Вернуть действие на панели инструментов Стандартная. (К ней также присоединена кнопка, раскрывающая список действий, допускающих возврат.)

Расширенный буфер обмена. При компиляции документа путем использования фрагментов текста, взятых из разных первоисточников, удобно пользоваться расширенным буфером обмена. Впервые расширенный буфер обмена появился в версии *Microsoft Word 2000*, в *Word XP* его объем был удвоен. Расширенный буфер обмена может хранить до 24 объектов (имеются также ограничения на общий объем используемой памяти).



Если между двумя последовательными операциями копирования текста в буфер обмена не было операции вставки, программа автоматически открывает Область задач в режиме Буфер обмена. Содержание конкретного элемента буфера также указывается в Области задач. При переполнении расширенного буфера самый старый элемент теряется, а очередной поступает в освобожденную ячейку.

Область задач позволяет вставить любой из имеющихся элементов, а также скомпоновать их в единый объект и вставить все сразу.

Автотекст. Автотекст — это режим автоматического ввода фрагментов текста. Он представлен двумя функциями: *автозавершением* и собственно *автотекстом*. Их принцип действия состоит в следующем.

Текстовый процессор хранит *словарь автотекста*, состоящий из слов и фраз, встречающихся в документах достаточно часто. При вводе первых четырех символов словарного элемента на экране появляется всплывающая подсказка с полным текстом слова или фразы. Если это то, что имел в виду пользователь, он завершает

ввод всего фрагмента нажатием клавиши ENTER — так работает функция *автозавершения*. Однако пользователь может самостоятельно выбрать необходимый элемент текста из списка с иерархической структурой — это функция *автотекста*. Список элементов автотекста открывается с помощью панели инструментов Автотекст (Вид ▸ Панели инструментов ▸ Автотекст).

Настройку словаря автотекста выполняют в диалоговом окне Автозамена (Вставка ▸ Автотекст ▸ Автотекст). Простейший способ наполнения словаря новым содержанием — выделить текст на экране, щелкнуть на кнопке Автотекст на панели инструментов Автотекст и в открывшемся диалоговом окне использовать кнопку Добавить.

Использование средства автозамены при вводе. Последние версии текстового процессора *Microsoft Word* позволяют эффективно сократить объем вводимого текста за счет использования средства Автозамена. Оно позволяет заменить ввод длинных последовательностей символов произвольным (желательно коротким) сочетанием других символов. Например, если в тексте очень часто встречается словосочетание «диалоговое окно», его можно заменить коротким сочетанием «.до». Соответственно вместо «диалоговых окон» использовать «.дн», а вместо «диалогового окна» — «.да». Точка перед символами стоит специально, чтобы отличать их от двухбуквенных предлогов или союзов, таких как «да».

Настройку средства Автозамена выполняют в диалоговом окне Сервис ▸ Параметры автозамены. Для этого надо установить флажок Заменять при вводе, ввести заменяемую комбинацию в поле Заменить, а замещающую комбинацию в поле На, после чего пополнить список автозамены щелчком на кнопке Добавить.

Как будет показано ниже, средство автоматической замены символов при вводе используется также для ввода специальных символов. Например, выполнив соответствующие настройки, можно вводить греческие буквы π и ρ обычным русским текстом: «пи» или «ро».

Ввод специальных и произвольных символов. При вводе текста часто существует необходимость ввода специальных символов, не имеющих соответствующей клавиши в раскладке клавиатуры, а также произвольных символов, раскладка для которых неизвестна. Основным средством для ввода специальных и произвольных символов, а также для закрепления их за избранными клавишами является диалоговое окно Символ (Вставка ▸ Символ). Данное диалоговое окно имеет две вкладки: Символы и Специальные знаки.

На вкладке Специальные знаки присутствует список специальных символов, таких как «длинное» («полиграфическое») тире, «копирайт», «торговая марка» и других. Для вставки такого символа достаточно щелкнуть на кнопке Вставить. Вместе с тем, для большинства специальных символов существуют клавиатурные комбинации — они приведены в списке, и их стоит запомнить. На первых порах, пока навык их ввода не закреплен, это окно используют для получения справки. В том же окне имеются кнопки Автозамена и Сочетание клавиш, позволяющие либо выполнять ввод специальных символов обычными символами и автоматически производить замену, либо закрепить специальный символ за избранной комбинацией клавиш.

На вкладке Символы представлены элементы управления для ввода произвольных символов любых символьных наборов (рис. 10.4). Центральное положение в окне занимает таблица символов текущего набора. Выбор шрифта выполняют в раскрывающемся списке Шрифт. Если шрифт относится к категории универсальных шрифтов *UNICODE*, то для него имеется и возможность выбора символьного набора в раскрывающемся списке Набор.

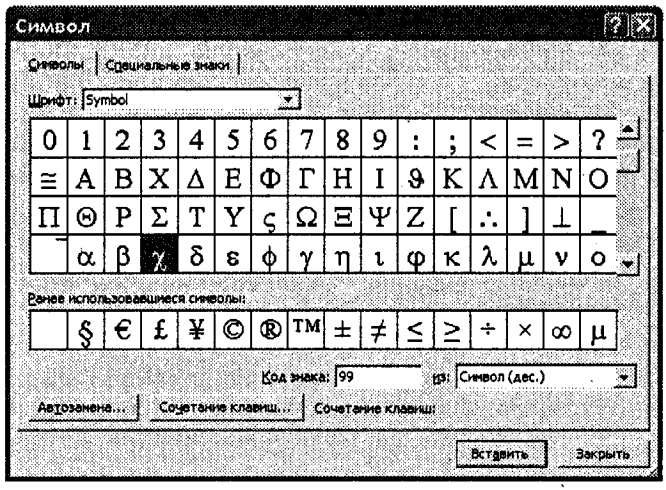


Рис. 10.4. Средство ввода специальных символов

Если символ надо вставить только один раз, достаточно щелкнуть на командной кнопке Вставить. Если предполагается многократное использование данного символа, за ним можно закрепить постоянную комбинацию клавиш (кнопка Сочетание клавиш) или создать элемент для списка Автозамена с помощью одноименной кнопки.

Специальные средства редактирования текста

Базовые приемы редактирования текста мы рассмотрели в разделе, посвященном стандартному приложению Блокнот. В данном разделе мы рассмотрим специальные средства редактирования, характерные для текстового процессора *Microsoft Word*, на примере последней версии *Microsoft Word XP*.

Режимы вставки и замены символов. Текстовый процессор предоставляет возможность выбора между двумя режимами редактирования текста: *режимом вставки* и *режимом замены*. В режиме вставки вводимый текст «раздвигает» существующий текст, а в режиме замены новые символы замещают символы предшествующего текста, находившиеся в точке ввода. Режим вставки применяют при разработке основных содержательных блоков текстовых документов, а режим замены — при редактировании стандартных форм и стандартных элементов (колонтитулов, реквизитных элементов в письмах, служебных записках, бланках).

Текущий режим правки текста индицируется в строке состояния. В режиме замены индикатор ЗАМ в строке состояния окна программы включен, в противном случае

он выключен. Двойной щелчок на этом индикаторе позволяет переключать режимы. Настройка режима правки выполняется на вкладке Правка диалогового окна Параметры (Сервис ▸ Параметры ▸ Правка).

Если установлены флажки Режим замены и Использовать клавишу INS для вставки, правка осуществляется в режиме замены символов. Если оба эти флажка сброшены, то режим можно выбирать с помощью клавиши INSERT. Если флажок Режим замены сброшен, а флажок Использовать клавишу INS для вставки установлен, то правка осуществляется в режиме вставки. Возможность изменять режим путем двойного щелчка на индикаторе в строке состояния сохраняется в любом случае.

Использование Тезауруса. Тезаурус представляет собой словарь смысловых синонимов. При подготовке технической документации особую роль играют смысловые синонимы к используемым глаголам. Для выделенного слова тезаурус удобно вызывать через пункт Синонимы контекстного меню. Однако этот прием срабатывает далеко не для всех слов (преимущественно для глаголов в неопределенной форме). Общий прием вызова тезауруса состоит в использовании команды строки меню Сервис ▸ Язык ▸ Тезаурус.

Окно Тезаурус имеет две панели. Его интересная особенность состоит в том, что в то время, как на левой панели отображаются синонимы выделенного слова, на правой панели могут отображаться синонимы к выбранному синониму, то есть поиск синонима является как бы двухуровневым. Заменяющий синоним можно выбирать как на левой панели, так и на правой. Замена производится щелчком на командной кнопке Заменить. Кроме синонимов в некоторых случаях тезаурус позволяет находить *антонимы* слов и *связанные* (как правило, однокоренные) слова.

Средства автоматизации проверки правописания. Средства автоматизации проверки правописания включают средства проверки орфографии и грамматики. Текстовый процессор позволяет реализовать два режима проверки правописания — *автоматический* и *командный.*

Для работы в автоматическом режиме надо установить флажки Автоматически проверять орфографию и Автоматически проверять грамматику на вкладке Правописание диалогового окна Параметры (Сервис ▸ Параметры ▸ Правописание). В автоматическом режиме слова, содержащие орфографические ошибки, подчеркиваются красным цветом, а выражения, содержащие грамматические ошибки, — зеленым. Для того чтобы узнать характер ошибки, надо щелкнуть правой кнопкой мыши на помеченном фрагменте. В зависимости от характера ошибки контекстное меню содержит пункт Орфография или Грамматика. С их помощью открывается диалоговое окно, в котором имеются элементы управления для получения более точной справки о том, какое правило нарушено, и предложены варианты исправления предполагаемой ошибки.

Встроенное автоматическое средство проверки правописания является, по существу, экспертной системой и допускает настройку. Так, например, если рекомендации экспертной системы неточны или неприемлемы, от них можно отказаться командой Пропустить (обычно такое бывает при проверке грамматики). Если же слово отмечено как орфографическая ошибка только потому, что оно отсутствует

в словаре системы автоматической проверки (например, слово браузер), то его можно добавить в словарь.

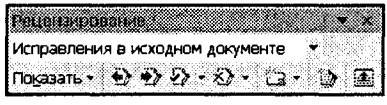
Встроенный словарь системы проверки правописания не подлежит правке. Все дополнения и изменения вносятся в специальный подключаемый *пользовательский словарь*. Каждый пользователь может создать несколько специализированных пользовательских словарей, ориентированных на различные области знаний (автомобильное дело, машиностроение, вычислительная техника и т. п.). Подключение нужного словаря для работы с конкретным документом выполняется выбором словарного файла в диалоговом окне Вспомогательные словари. Чтобы открыть его, надо щелкнуть на кнопке Словари на вкладке Сервис ▸ Параметры ▸ Правописание. Постепенно наполняясь конкретным содержанием, вспомогательные словари пользователя становятся мощным средством повышения производительности его труда.

В командном режиме проверка правописания выполняется независимо от установки элементов управления на вкладке Сервис ▸ Параметры ▸ Правописание. Запуск средства проверки выполняют командой Сервис ▸ Правописание. Проверка начинается от местоположения курсора и продолжается до появления первой ошибки. После исправления ошибки проверка продолжается дальше, а по достижении конца документа проверка может быть продолжена с его начала. Естественное завершение проверки происходит, когда документ просмотрен целиком.

В тех случаях, когда пользователь отказывается от предлагаемых исправлений и дает команду Пропустить, в документе накапливается *список пропускаемых слов*, то есть слов и выражений, не подлежащих проверке. Для того чтобы очистить этот список и начать проверку заново, используют командную кнопку Сервис ▸ Параметры ▸ Правописание ▸ Повторная проверка.

Средства рецензирования текста

Под рецензированием можно понимать два процесса: *редактирование текста с регистрацией изменений* и *комментирование* текста. В отличие от обычного редактирования при рецензировании текст документа изменяется не окончательно — новый вариант и старый «сосуществуют» в рамках одного документа на правах различных *версий*.



Основным средством рецензирования является панель Рецензирование (Вид ▸ Панели управления ▸ Рецензирование). На ней представлены несколько групп элементов управления, предназначенных для:

- создания, просмотра и удаления примечаний;
- регистрации, просмотра, принятия и отмены изменений;
- выбора цвета выделения примечаний;
- сохранения версий документа.

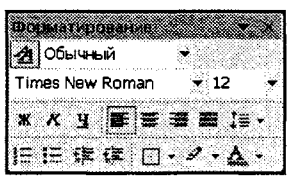
Для создания примечания служит кнопка Создать примечание. При ее использовании местоположение курсора выделяется заданным цветом, а на правом поле

страницы открывается область ввода текста примечания. Созданное примечание отображается только при просмотре документа, но не при его печати.

Для регистрации изменений в тексте служит кнопка Исправления. Все редактирование текста в режиме регистрации исправлений считается неавторским и выделяется особым методом (метод выделения можно задать на вкладке Исправления диалогового окна Сервис ▸ Параметры). Прочие элементы управления данной панели позволяют выполнять переходы между исправлениями, принимать их или отвергать.

Если документ проходит многоступенчатое редактирование, часто возникает необходимость хранить его промежуточные версии. Текстовый процессор *Microsoft Word XP* позволяет хранить несколько версий документа в одном файле. Это удобное средство отличается тем, что при сохранении нескольких версий (в отличие от нескольких копий) эффективно используется рабочее место на диске. Дело в том, что при сохранении очередной версии не происходит повторного сохранения всего документа — сохраняются только отличия текущей версии от предшествующей. Для сохранения текущей версии и для загрузки одной из промежуточных версий применяют команду Файл ▸ Версии.

Форматирование текста



Форматирование текста осуществляется средствами меню Формат или панели Форматирование. Основные приемы форматирования включают:

- выбор и изменение гарнитуры шрифта;
- управление размером шрифта;
- управление начертанием и цветом шрифта;
- управление методом выравнивания;
- создание маркированных и нумерованных списков (в том числе многоуровневых);
- управление параметрами абзаца.

Возможно, вы обратили внимание на то, что внешний вид панели Форматирование, представленной на рисунке справа, отличается от того, что вы видите в своей программе. Это связано с тем, что все инструментальные панели программы Word можно сделать плавающими. Перетаскиванием за рубчик, имеющийся на левом краю панели, их можно переместить в любое место экрана, в том числе и вне пределов основного рабочего окна. Правда, преимущества такого приема ощутят не все пользователи, а только те, у кого мониторы имеют достаточно большой размер.

Настройка шрифта. При выборе гарнитуры шрифта следует иметь в виду следующие обстоятельства:

- Выбор гарнитуры шрифта действует на выделенный текстовый фрагмент. Если ни один фрагмент не выделен, он действует на весь вводимый текст до очередной смены гарнитуры.
- Начиная с версии *Microsoft Word* 97, текстовые процессоры *Word* ориентированы на работу с многоязычными шрифтовыми наборами (*UNICODE*). При использовании других шрифтовых наборов возможны проблемы, которые воз-

никают при переключении раскладки клавиатуры с основной (английской) на дополнительную (русскую). В таком случае возможен неконтролируемый автоматический возврат к использованию одного из стандартных шрифтов *UNICODE*, зарегистрированных в операционной системе.

Напомним, что как операционная система Windows XP, так и сам текстовый процессор Microsoft Word поставляются с наборами шрифтов UNICODE, то есть использование шрифтов, входящих в стандартную поставку, является гарантией от непредвиденных осложнений.

Настройку шрифта выполняют в диалоговом окне Шрифт (Формат ▸ Шрифт). В версии *Microsoft Word XP* данное диалоговое окно имеет три вкладки: Шрифт, Интервал и Анимация.

На вкладке Шрифт выбирают:

- гарнитуру шрифта;
- его размер (измеряется в полиграфических пунктах);
- вариант начертания;
- цвет символов;
- наличие подчеркивания;
- характер видоизменения.

При выборе гарнитуры шрифта следует иметь в виду, что существует две категории шрифтов: с засечками и без засечек (*рубленые*). Характерными представителями первой категории являются шрифты семейства *Times*, а второй категории — шрифты семейства *Arial*. Шрифты, имеющие засечки, легче читаются в больших текстовых блоках — их рекомендуется применять для оформления основного текста.

Шрифты, не имеющие засечек, рекомендуется использовать для заголовков в технических текстах, а также для оформления дополнительных материалов (врезок, примечаний и прочего).

Кроме того, считается, что шрифты с засечками лучше воспринимаются в документах, напечатанных на бумаге. Для электронных документов, которые предполагается читать с экрана, многие предпочитают применять рубленые шрифты.

Большинство гарнитур шрифтов являются *пропорциональными*. Это означает, что и ширина отдельных символов, и расстояние между соседними символами не являются постоянными величинами и динамически меняются так, чтобы сопряжение символов было наиболее благоприятным для чтения.

Особую группу представляют так называемые *моноширинные* шрифты. В них каждый символ вместе с окаймляющими его интервалами имеет строго определенную ширину. Такие шрифты применяют в тех случаях, когда надо имитировать шрифт пишущей машинки, а также при вводе текстов, представляющих листинги программ. Характерными представителями таких шрифтов являются шрифты семейства *Courier*.

При выборе размера шрифта руководствуются назначением документа, а также вертикальным размером печатного листа. Для документов, имеющих формат типо-

вой книжной страницы, обычно применяют шрифт размером 10 пунктов. Для документов, готовящихся для печати на стандартных листах формата А4 (210×297 мм), выбирают размер 12 пунктов. При подготовке документов, предназначенных для передачи средствами факсимильной связи, применяют увеличенный размер — 14 пунктов и больше (факсимильные документы часто воспроизводятся с искажениями, и увеличенный размер шрифта улучшает удобство их чтения).

При подготовке электронных документов, распространяемых в формате *Microsoft Word*, размер шрифта выбирают, исходя из разрешения экрана. В настоящее время наиболее распространены компьютеры, видеоподсистема которых настроена на экранное разрешение 800×600 точек или 1024×768 точек. Для этих параметров целесообразно готовить электронные документы с размером шрифта 12 пунктов. На этот размер по умолчанию настроены последние версии процессора *Microsoft Word XP*. (Версия *Microsoft Word* 97 была настроена по умолчанию на размер экранного шрифта 10 пунктов, но практика показала, что он неудобен.)

Использование прочих средств управления шрифтом (выбор начертания, подчеркивания и других видоизменений) определяется стилевым решением документа, которое задает заказчик или работодатель. Приступая к первому заданию, следует выяснить, какие стилевые решения уже существуют в данной организации, каковы ограничения на использование средств оформления и форматирования. По возможности, надо получить от заказчика готовые шаблоны документов или хотя бы печатные образцы.

Из прочих, не рассмотренных здесь средств управления шрифтами надо отметить управление интервалом между символами и возможность использования эффектов анимации. Интервал задается путем выбора одного из трех значений (Обычный, Разреженный, Уплотненный) на вкладке Формат ▸ Шрифт ▸ Интервал.

Эффекты анимации используют очень редко и только при подготовке электронных документов, распространяемых в формате текстового процессора. В печатных документах эти эффекты невоспроизводимы по очевидным причинам, а в *Web*-документах их нет смысла применять, так как они пока не поддерживаются *Web*-браузерами.

Настройка метода выравнивания. Все последние версии текстового процессора *Microsoft Word* поддерживают четыре типа выравнивания:

- по левому краю;
- по центру;
- по правому краю;
- по ширине.

Выбор метода выполняют соответствующими кнопками панели инструментов Форматирование или из раскрывающегося списка Формат ▸ Абзац ▸ Отступы и интервалы ▸ Выравнивание. Избранный метод действует на текущий и последующие вводимые абзацы. Выбор метода выравнивания определяется назначением документа. Так, например, для *Web*-страниц нет смысла выполнять выравнивание по ширине, поскольку все равно неизвестна ширина окна браузера, в котором документ будет просматриваться, однако выравнивание по центру использовать можно.

Для документов, передаваемых на последующую обработку, все методы выравнивания, кроме тривиального выравнивания по левому краю, являются излишними. Для печатных документов, выполненных на русском или немецком языках, рекомендуется в основном тексте использовать выравнивание по ширине с одновременным включением функции переноса, а для документов на английском языке основной метод выравнивания — по левому полю.

Настройка параметров абзаца. Кроме режима выравнивания настраиваются следующие параметры абзаца:

- величина отступа слева (от левого поля);
- величина отступа справа (от правого поля);
- величина отступа первой строки абзаца («красная строка»);
- величина интервала (отбивки между абзацами) перед абзацем и после него.

Для печатных документов величину отступа для основного текста, как правило, не задают (необходимое положение текста определяется шириной полей), но ее задают для дополнительных материалов и заголовков, если они не выравниваются по центру. В то же время, для *Web*-страниц величина отступа для абзацев имеет большое значение. Это один из весьма немногих параметров форматирования, допускаемых для *Web*-документов, поэтому его используют очень широко.

Роль отбивок между абзацами, как и роль отступа первой строки абзаца, состоит в том, чтобы визуально выделить абзацы. При этом следует помнить, что эти средства несовместимы. То есть, применяя отступ первой строки абзаца, не следует применять отбивки между абзацами, и наоборот. Комбинация этих стилей допускается только для маркированных и нумерованных списков (основной текст оформляется с отступом первой строки, а списки — без него, но с отбивкой между абзацами).

Обычная практика назначения формата состоит в том, что для документов простой структуры (художественных) используют отступ первой строки (это особенно важно для текстов на русском и немецком языках), а для документов сложной структуры (технических) и документов на английском языке используют отбивки между абзацами. Промежуточное положение занимают документы, относящиеся к естественнонаучным и гуманитарным дисциплинам, — при их подготовке кроме точки зрения автора руководствуются сложившейся практикой и устоявшимися традициями.

В *Web*-документах применяют только отбивки между абзацами. Отступ первой строки в них обычно не используют в связи с повышенными трудностями его создания.

Средства создания маркированных и нумерованных списков. Специальное оформление маркированных и нумерованных списков редко применяют в художественных документах и персональной переписке, но в служебных документах и особенно в *Web*-документах оно используется очень широко. В *Web*-документах оформление маркированных списков особо усиливают за счет применения специальных графических маркеров, стиль которых должен тематически сочетаться с содержанием и оформлением документов.

Для создания нумерованных и маркированных списков нужно сначала выполнить *настройку*, затем *вход* в список и, наконец, *выход* из него. Настройку выполняют в диалоговом окне Список, открываемом командой Формат ▸ Список. Данное окно имеет четыре вкладки: Маркированный список, Нумерованный список, Многоуровневый список и Список стилей. В качестве элементов управления здесь представлены образцы оформления списков. Для выбора нужного достаточно щелкнуть на избранном образце.

Вход в список может осуществляться автоматически или по команде. Чтобы автоматически создать маркированный список, достаточно начать запись строки с ввода символа «*». По завершении строки и нажатии клавиши ENTER символ «*» автоматически преобразуется в маркер, а на следующей строке маркер будет установлен автоматически. Для автоматического создания нумерованного списка достаточно начать строку с цифры, после которой стоят точка и пробел, например «1. », «2. » и т. д. Этот метод позволяет начать нумерацию с любого пункта (не обязательно с единицы).

Для создания списка по команде служат кнопки Нумерация и Маркеры, представленные на панели Форматирование. Как маркированный, так и нумерованный список легко превратить в многоуровневый. Для перехода на новые (или возврата на предшествующие уровни) служат кнопки Увеличить отступ и Уменьшить отступ на панели Форматирование.

Для списков с очень глубоким вложением уровней (более трех) можно настроить стиль оформления каждого из уровней. Для этого служит командная кнопка Изменить на вкладке Многоуровневый диалогового окна Список (Формат ▸ Список).

Вкладка Список стилей также предназначена для оформления многоуровневых списков. Она позволяет выбрать или определить для каждого уровня списка особый стиль. О работе со стилями мы поговорим чуть позже.

Характерной особенностью процессора *Microsoft Word XP*, связанной с его ориентацией на создание *Web*-документов, является возможность использования графических маркеров. Для выбора такого маркера на вкладке Маркированный диалогового окна Список (Формат ▸ Список) выберите один из образцов списка и щелкните на кнопке Изменить. Откроется диалоговое окно Изменение маркированного списка, в котором надо щелкнуть на кнопке Рисунок. Эта кнопка открывает диалоговое окно Рисованный маркер, в котором можно выбрать подходящее изображение.

Для завершения маркированного или нумерованного списка и выхода из режима его создания достаточно по завершении ввода последней строки дважды нажать клавишу ENTER.

10.3. Приемы и средства автоматизации разработки документов

С рядом приемов автоматизации ввода и редактирования текста мы познакомились выше. К ним относятся средства Автотекст, Автозамена, средства проверки правописания, средства расстановки переносов, средства поиска и замены фрагментов текста.

В этом разделе мы познакомимся с наиболее общими средствами автоматизации разработки и оформления документов, к числу которых относятся *стили оформления абзацев*, *шаблоны документов* и *темы оформления*.

Работа со стилями

Абзац — элементарный элемент оформления любого документа. Каждый заголовок документа тоже рассматривается как отдельный абзац. Выше мы видели, что в меню Формат ▸ Абзац имеется немало различных элементов управления, и выполнять их настройку для каждого абзаца отдельно — неэффективная и утомительная задача. Она автоматизируется путем использования понятия *стиль*.

Стиль оформления — это именованная совокупность настроек параметров шрифта, абзаца, языка и некоторых элементов оформления абзацев (линий и рамок). Благодаря использованию стилей обеспечивается простота форматирования абзацев и заголовков текста, а также единство их оформления в рамках всего документа.

Особенностью текстовых процессоров *Microsoft Word* является то, что они поддерживают четыре типа стилей: *стили абзаца*, *стили знаков* (символов), *стили списков* и *стили таблиц*. С помощью стилей абзаца выполняют форматирование абзацев, а с помощью знаковых стилей можно изменять оформление выделенных фрагментов текста внутри абзаца. Стиль списка предполагает наличие в начале абзаца номера или маркера. Стиль таблицы обеспечивает согласование границ, заливки, выравнивания и шрифтов в таблицах.

Наличие разных типов стилей позволяет реализовать довольно сложные приемы форматирования. Например, внутри абзаца, оформленного одним шрифтом, могут содержаться фрагменты текста, оформленные другим шрифтом. В данной книге, например, специальный шрифт использован для записи названий элементов управления.

Работа со стилями состоит в создании, настройке и использовании стилей. Некоторое количество стандартных стилей присутствует в настройке программы по умолчанию, сразу после ее установки. Их используют путем выбора нужного стиля из раскрывающегося списка на панели управления Форматирование.

Все работы по созданию новых стилей и изменению существующих выполняют с помощью Области задач в режиме Стили и форматирование. Если Область задач закрыта или находится в ином режиме, надо дать команду Формат ▸ Стили и форматирование.

Настройка стиля. Настройку стиля (рис. 10.5) выполняют в диалоговом окне Стиль (Формат ▸ Стиль). Настраиваемый стиль выбирают в списке Стили (при этом на панелях Абзац и Знаки отображаются образцы применения данного стиля). Для изменения стиля служит командная кнопка Изменить, открывающая диалоговое окно Изменение стиля. Каждый из компонентов стиля настраивается в отдельном диалоговом окне. Выбор компонента выполняют в меню, открываемом с помощью командной кнопки Формат.

При проведении настройки стиля важно правильно выбрать исходный стиль. Он должен быть как можно ближе к желаемому, чтобы минимизировать количество необходимых настроек.

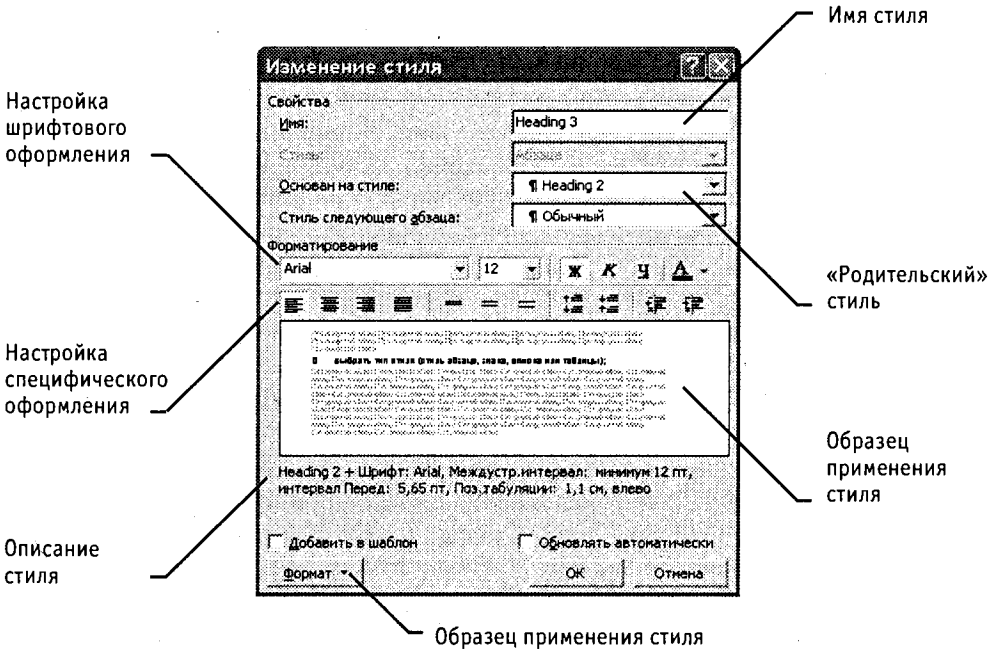


Рис. 10.5. Настройка стиля

Создание стиля. Для создания нового стиля надо щелкнуть на кнопке Создать стиль в Области задач. Откроется диалоговое окно Создание стиля.

В данном окне следует:

- ввести название нового стиля в поле Имя;
- выбрать тип стиля (стиль абзаца, знака, списка или таблицы);
- выбрать стиль, на котором основан новый стиль;
- указать стиль следующего абзаца;
- настроить основные элементы стиля, используя средства данного диалогового окна;
- настроить дополнительные элементы стиля с помощью кнопки Формат.

Важной чертой программы является принцип *наследования стилей*. Он состоит в том, что любой стиль может быть основан на каком-то из существующих стилей. Это позволяет, во-первых, сократить настройку стиля до минимума, сосредоточившись только на отличиях от базового, а во-вторых, обеспечить принцип единства оформления всего документа в целом. Так, например, при изменении базового стиля автоматически произойдут и изменения наследуемых элементов в стилях, созданных на его основе.

Стиль следующего абзаца указывают для обеспечения автоматического применения стиля к следующему абзацу, после того как предыдущий абзац закрывается клавишей ENTER.

Разработка новых стилей и их настройка являются достаточно сложными технологическими операциями. Они требуют тщательного планирования, внимательности и аккуратности, особенно в связи с тем, что согласно принципу наследования свойств стилей желаемые изменения в одном стиле могут приводить к нежелательным изменениям во многих других стилях.

В связи с трудоемкостью изучения и освоения приемов практической работы со стилями начинающие пользователи часто ими пренебрегают. Действительно, при разработке небольших документов (одна-две страницы) можно обойтись без настройки и использования стилей, выполнив все необходимое форматирование вручную средствами меню Формат. Однако при разработке объемных документов вручную очень трудно обеспечить единство оформления, особенно если разные разделы документа разрабатывались разными авторами.

☑ Прийти к использованию стилей надо как можно раньше. Правильное и рациональное использование этого средства является залогом высокой эффективности работы с процессором Microsoft Word и высокого качества разрабатываемых документов. На изучение средств управления стилями может потребоваться несколько часов, но полученные при этом навыки останутся на всю жизнь и пригодятся многократно.

Шаблоны

Совокупность удачных стилевых настроек сохраняется вместе с готовым документом, но желательно иметь средство, позволяющее сохранить их и вне документа. Тогда их можно использовать для подготовки новых документов. Такое средство есть — это шаблоны, причем некоторое количество универсальных шаблонов поставляется вместе с текстовым процессором и устанавливается на компьютере вместе с ним.

По своей сути, шаблоны — это тоже документы, а точнее говоря, заготовки будущих документов. От обычных документов шаблоны отличаются тем, что в них приняты специальные меры, исключающие возможность их повреждения. Открывая шаблон, мы начинаем новый документ и вносим изменения в содержание шаблона. При сохранении же мы записываем новый документ, а шаблон, использованный в качестве его основы, остается в неизменном виде и пригоден для дальнейшего использования.

Использование шаблона для создания документа. По команде Файл ▸ Создать открывается Область задач в режиме Создание документа. Здесь можно выбрать шаблон, на базе которого документ будет разрабатываться. В этом случае документ сразу получает несколько готовых стилей оформления, которые содержатся в шаблоне. Основные шаблоны перечислены в области задач в разделе Создание. Если их недостаточно, надо щелкнуть на ссылке Общие шаблоны и выбрать подходящий шаблон на одной из вкладок открывшегося диалогового окна Шаблоны.

Изменение шаблона готового документа. Эта достаточно редкая операция выполняется с помощью диалогового окна Шаблоны и настройки (Сервис ▸ Шаблоны и настройки). Для смены текущего шаблона следует использовать кнопку Присоединить и в открывшемся диалоговом окне Присоединение шаблона выбрать нужный шаблон в папке C:\Program Files\Microsoft Office\Шаблоны.

Создание нового шаблона на базе шаблона. Открыв диалоговое окно Шаблон щелчком на ссылке Общие шаблоны в Области задач (режим Создание документа), включите переключатель Шаблон. Теперь надо выбрать стандартный шаблон, на базе которого создается новый (рис. 10.6). После настройки стилей и редактирования содержания выполняется сохранение шаблона командой Сохранить как с включением пункта Шаблон документа в поле Тип файла.

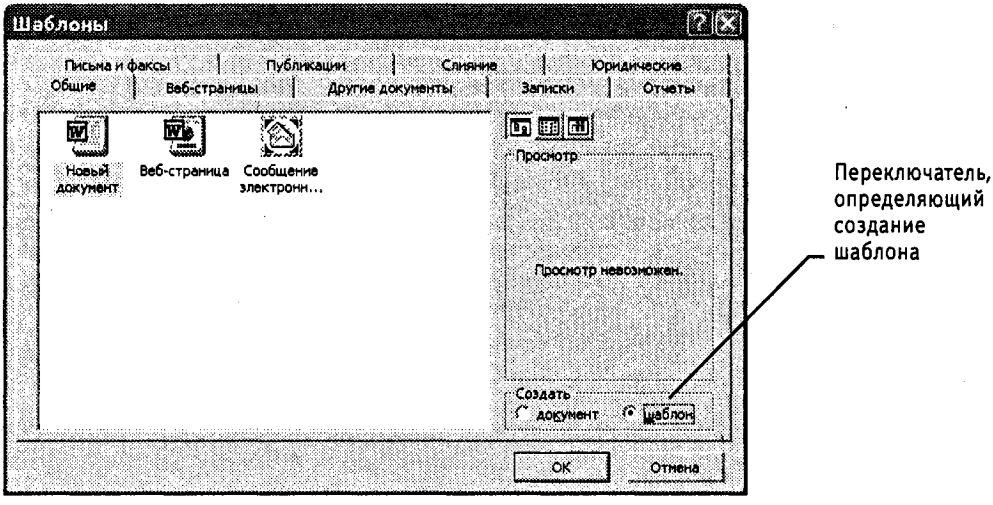


Рис. 10.6. Диалоговое окно Создание документа

Создание нового шаблона на базе документа. Если готовый документ может быть использован в качестве заготовки для создания других документов, его целесообразно сохранить как шаблон. Командой Файл ▸ Открыть открывают готовый документ, в нем правят содержание и настраивают стили, а потом сохраняют документ как шаблон командой Сохранить как с включением пункта Шаблон документа в поле Тип файла.

Темы

Последние версии текстового процессора *Microsoft Word* (начиная с *Word 2000*) имеют специальное средство автоматического оформления, предназначенное в первую очередь для электронных документов (для *Web*-документов и документов, распространяемых в формате процессора). Это средство называется *темы*. Тема представляет собой совокупность следующих элементов оформления:

- фоновый узор;
- стили оформления основного текста и заголовков;
- стиль оформления маркированных списков;
- стиль графических элементов оформления (линий).

Доступ к выбору тем выполняется командой Формат ▸ Темы.

Практическое занятие

Упражнение 10.1. Первичные настройки текстового процессора Microsoft Word XP

1. Запустите текстовый процессор командой Пуск ▸ Программы ▸ Microsoft Word.
2. Откройте заранее подготовленный файл (произвольный).
3. Откройте меню настройки панелей управления (Вид ▸ Панели управления) и убедитесь в том, что включено отображение только двух панелей: Стандартная и Форматирование.
4. В качестве режима отображения документа выберите Режим разметки. Для этого используйте соответствующую кнопку в левом нижнем углу окна документа или команду Вид ▸ Разметка страницы.
5. Если шрифт на экране выглядит слишком мелким, настройте масштаб отображения командой Вид ▸ Масштаб. Можно также использовать раскрывающийся список Масштаб на панели инструментов Стандартная. Если желаемого масштаба нет в списке (например, 125%), введите нужное значение непосредственно в поле списка и нажмите клавишу ENTER. Для эффективного использования площади окна документа при достаточном разрешении экрана можно использовать пункты По ширине страницы или По ширине текста.
6. В качестве единицы измерения для настройки параметров документа выберите миллиметры (Сервис ▸ Параметры ▸ Общие ▸ Единицы измерения).
7. Настройте список быстрого открытия документов. После запуска программы в меню Файл можно найти список из нескольких документов, открывавшихся в текстовом процессоре в последнее время. Это удобно для быстрого открытия нужного документа. Количество документов, отображаемых в этом списке, задайте счетчиком Сервис ▸ Параметры ▸ Общие ▸ Помнить список из ... файлов.
8. Отключите замену выделенного фрагмента при правке текста, сбросив флажок Сервис ▸ Параметры ▸ Правка ▸ Заменять выделенный фрагмент. Это несколько снижает производительность труда при редактировании текста, но страхует начинающих от нежелательных ошибок. С набором опыта практической работы этот флажок можно установить вновь.
9. Включите контекстно-чувствительное переключение раскладки клавиатуры (Сервис ▸ Параметры ▸ Правка ▸ Автоматическая смена клавиатуры). Эта функция удобна при редактировании текста. При помещении курсора в английский текст автоматически включается англоязычная раскладка, а при помещении его в текст на русском языке — русскоязычная.
10. Запретите «быстрое» сохранение файлов, сбросив флажок Сервис ▸ Параметры ▸ Сохранение ▸ Разрешить быстрое сохранение. При «быстром» сохранении сохраняется не сам файл, а только его изменения по сравнению с предыдущей сохраненной версией. Это действительно сокращает время операции сохранения, но замедляет другие операции с документами. При этом также заметно возрастают размеры итогового файла.

11. Настройте функцию *автосохранения* с помощью счетчика Сервис ▸ Параметры ▸ Сохранение ▸ Автосохранение каждые ... минут. Имейте в виду следующие обстоятельства:
    - при автосохранении данные записываются в специальный файл, который в аварийных ситуациях может быть использован для восстановления несохраненных данных, но только однократно(!);
    - функция автосохранения не отменяет необходимости периодически во время работы и после ее завершения сохранять файл прямыми командами Сохранить или Сохранить как.
12. Временно отключите средства проверки правописания. На вкладке Сервис ▸ Параметры ▸ Правописание сбросьте флажки Автоматически проверять орфографию и Автоматически проверять грамматику. На ранних этапах работы с документом надо сосредоточиться на его содержании, а средства проверки правописания могут отвлекать от этого. Завершая работу над документом, необходимо вновь подключить и использовать эти средства.
13. Временно отключите функцию *автозамены при вводе* сбросом флажка Сервис ▸ Параметры автозамены ▸ Автозамена ▸ Заменять при вводе.
14. Включите автоматическую замену «прямых» кавычек парными: Сервис ▸ Параметры автозамены ▸ Автоформат при вводе ▸ Заменять при вводе «прямые» кавычки парными. В русскоязычных текстах прямые кавычки не применяются. Для подготовки англоязычных текстов и листингов программ отключите эту функцию.
15. Временно отключите ряд средств автоматического форматирования, в частности автоматическую маркировку и нумерацию списков. На вкладке Сервис ▸ Параметры автозамены ▸ Автоформат при вводе сбросьте флажки Применять при вводе стили маркированных списков и Применять при вводе стили нумерованных списков. После приобретения первичных навыков работы с текстами вновь подключите эти средства.
16. Отключите Помощника. Помощник — удобное интерактивное средство для получения конкретной справки, но справочная система программы в целом обладает более высокой методической ценностью. В текстовом процессоре *Microsoft Word XP* Помощник «перехватывает» все запросы к справочной системе, поэтому для полноценной работы со справочной системой его надо принудительно отключить.
    - Вызовите Помощника: Справка ▸ Справка: Microsoft Word.
    - Щелкните на изображении Помощника правой кнопкой мыши и выберите в контекстном меню пункт Параметры — откроется диалоговое окно Помощник.
    - На вкладке Параметры сбросьте флажок Использовать Помощника.
    - Закройте диалоговое окно Помощник щелчком на кнопке OK.

    Проверьте, как работает вход в справочную систему: Справка ▸ Справка: Microsoft Word. Вместо Помощника должно открываться окно справочной системы.
17. Отключите автоматическую расстановку переносов. В абсолютном большинстве случаев на ранних этапах работы с документами она не нужна. Для *Web*-документов, для документов, распространяемых в формате текстового процес-

сора, и для документов, передаваемых на последующую обработку, расстановка переносов не только бесполезна, но и вредна. Для документов, которые окончательно форматируются и распечатываются в одной рабочей группе, расстановка переносов может быть полезной, но и в этом случае ее применяют только на заключительных этапах форматирования и при этом очень тщательно проверяют соответствие переносов, расставленных автоматически, нормам и правилам русского языка.

Расстановку переносов отключают сбросом флажка Сервис ▸ Язык ▸ Расстановка переносов ▸ Автоматическая расстановка переносов.

18. Включите запрос на подтверждение изменения шаблона «Обычный»: Сервис ▸ Параметры ▸ Сохранение ▸ Запрос на сохранение шаблона Normal.dot. Шаблон «Обычный» является первоосновой для всех остальных шаблонов (они создаются на его базе и наследуют его свойства). При обычной работе с программой необходимость его изменения не возникает (если надо что-то изменить в этом шаблоне, достаточно создать его копию под другим именем и работать с ней). Включением данного флажка предупреждаются случайные внесения изменений в шаблон со стороны пользователя, а также попытки макровирусов сохранить свой код в данном шаблоне (для дальнейшего размножения в документах, создаваемых на его основе).

Мы научились выполнять первичные настройки текстового процессора и узнали, что доступ к ним осуществляется следующими командами:

- Сервис ▸ Параметры;
- Сервис ▸ Параметры автозамены;
- Сервис ▸ Язык;
- Вид ▸ Панели инструментов;
- Вид ▸ Масштаб.

Упражнение 10.2. Первичные настройки параметров печатного документа

Форматирование документов, предназначенных для печати на принтере, выполняется в «привязке» к параметрам печатной страницы. Поэтому создание документов этой категории необходимо начинать с настройки параметров страницы. К этим параметрам относятся прежде всего размер листа бумаги и величина полей.

Характерная ошибка начинающих заключается в том, что они начинают подготовку документов с ввода текста. Интуитивно понятно, что текст — это важнейший компонент документа, но для ввода текста служат программы иного класса — текстовые редакторы. Имея дело с текстовым процессором, начинать надо не с ввода текста документа, а с настройки параметров печатной страницы, поскольку от нее зависят все используемые приемы форматирования. Тем, кому утомительно начинать создание каждого документа с настройки параметров страницы, можно порекомендовать чаще пользоваться заранее заготовленными шаблонами.

1. Запустите текстовый процессор командой Пуск ▸ Программы ▸ Microsoft Word.
2. Дайте команду для создания нового документа: Файл ▸ Создать.

3. Щелкните на ссылке Новый документ в Области задач, которая открылась в режиме Создание документа.
4. Откройте диалоговое окно Параметры страницы (Файл ▸ Параметры страницы).
5. На вкладке Размер бумаги выберите в раскрывающемся списке Размер бумаги пункт A4 210×297 mm (этот формат принят в России в качестве стандартного). При использовании нестандартного формата выбирают пункт Другой и с помощью кнопок счетчиков Ширина и Высота задают его параметры.
6. На вкладке Поля задайте ориентацию бумаги (Книжная или Альбомная). При «альбомной» ориентации бумага располагается длинной стороной по горизонтали.
7. На этой же вкладке задайте размеры полей:

   Верхнее — 15 мм Нижнее — 20 мм

   Левое — 25 мм Правое — 15 мм
8. На вкладке Источник бумаги задайте для нижнего поля интервал от края до колонтитула 12 мм (в нижнем колонтитуле будет размещаться номер печатной страницы).
9. Если предполагается двусторонняя печать (четные страницы печатаются на оборотной стороне нечетных страниц), выберите на вкладке Поля пункт Зеркальные поля в списке Несколько страниц. Восстановите обычную настройку.
10. Проверьте, как действует настройка печати двух страниц на одном листе. Выберите в списке Несколько страниц пункт 2 страницы на листе. На панели Образец рассмотрите результат настройки. Установите «альбомную» ориентацию страниц. Оцените результат настройки. Восстановите «книжную» ориентацию и печать одной страницы на листе.
11. Создайте нижний колонтитул для размещения номера печатной страницы. Дайте команду Вид ▸ Колонтитулы — откроется панель инструментов Колонтитулы. Пользуясь кнопкой Верхний/нижний колонтитулы, создайте область нижнего колонтитула. Вставьте в нее номер страницы щелчком на кнопке Номер страницы на панели инструментов Колонтитулы. Отцентрируйте номер страницы щелчком на кнопке По центру на панели инструментов Форматирование. Закройте панель Колонтитулы. Убедитесь в том, что в документе появились нижние колонтитулы с номерами страниц.

Прямой команды для удаления колонтитулов нет. Чтобы удалить колонтитулы по всему документу, надо очистить область колонтитула на одной из страниц. Колонтитул, лишенный содержимого, удаляется автоматически. Для удаления содержимого колонтитула откройте панель Колонтитулы (Вид ▸ Колонтитулы), переключитесь на верхний или нижний колонтитул (по ситуации) кнопкой Верхний/нижний колонтитулы, выделите элемент содержимого и нажмите клавишу DELETE.

12. Закройте панель инструментов Колонтитулы. Сохраните документ командой Сохранить как, дав ему имя Эксперимент и использовав для сохранения папку \Мои документы.

Мы научились создавать и настраивать печатные документы. В порядке эксперимента мы создали «пустой» документ, имеющий настроенные параметры страницы, стили, соответствующие шаблону «Обычный», и нижний колонтитул для размещения номеров печатных страниц. Мы готовы к наполнению данного документа текстовым содержанием с последующим редактированием и форматированием.

Упражнение 10.3. Ввод специальных символов

30 мин

В этом упражнении мы рассмотрим пять приемов ввода символов греческого алфавита. Особо отметим, что это еще далеко не все возможные приемы для текстового процессора *Microsoft Word*. Упражнение будем выполнять вводом фразы: Длина окружности равна 2πR. Для подготовки к упражнению запустите текстовый процессор и создайте пустой документ, взяв за основу шаблон Обычный.

1. *Замена шрифта*. Введите текст: Длина окружности равна 2pR. Выделите букву «p». На панели Форматирование раскройте список шрифтов и выберите символьный набор Symbol. Символ «p» заменится символом «π».

   Если панель Форматирование скрыта, то доступ к списку шрифтов можно получить командой Формат ▸ Шрифт. Это наиболее стандартный прием. Им можно пользоваться во всех программах, имеющих средства для изменения шрифта, но для его применения нужно заранее знать, какой символ латинского шрифта соответствует нужному символу греческого шрифта, а это не всегда возможно.

2. *Классический подход*. Введите текст: Длина окружности равна 2xR. Выделите символ «x». Откройте программу Таблица символов (Пуск ▸ Программы ▸ Стандартные ▸ Служебные ▸ Таблица символов). В окне этой программы выберите шрифт Symbol. В поле таблицы разыщите символ π, выделите его, щелкните на кнопке Выбрать и на кнопке Копировать. Вернитесь в окно *Microsoft Word* и комбинацией клавиш CTRL+V вставьте из буфера обмена скопированный символ на место выделенного.

   Этот прием действует в большинстве программ. Его применяют, если заранее не известно, какому символу латинского шрифта соответствует необходимый символ.

3. *Использование стиля*. Если документ содержит много символов греческого алфавита, имеет смысл создать для них специальный *знаковый стиль*. На базе существующего знакового стиля, например стиля Основной шрифт абзаца создайте новый знаковый стиль, например Греческий. Для этого откройте Область задач в режиме Стили и форматирование (Формат ▸ Стили и форматирование) и щелкните на кнопке Создать стиль. В диалоговом окне Создание стиля в поле Имя введите имя нового стиля, в раскрывающемся списке Стиль выберите пункт Знака и в списке Основан на стиле выберите базовый стиль. Если предполагается и в дальнейшем создание аналогичных документов, созданный стиль можно сохранить в шаблоне, установив флажок Добавить в шаблон. После этого выберите символьный набор Symbol в раскрывающемся списке на панели Форматирование. В дальнейшем при необходимости ввода греческих букв достаточно на панели форматирование выбрать стиль Греческий.

Этот прием специфичен для программы *Microsoft Word*. Далеко не все текстовые редакторы и процессоры позволяют создавать знаковые стили — большинство используют только стили абзаца, применение которых изменяет шрифт во всем абзаце целиком.

4. *Применение «горячих клавиш»*. Это самый эффективный прием. Нет более быстрого способа ввода нестандартных символов, чем ввод с помощью заранее назначенных клавиатурных комбинаций. Так, например, мы можем закрепить символ π за комбинацией клавиш CTRL+ALT+P и использовать ее всюду, где в этом возникает необходимость.

   Дайте команду Вставка ▸ Символ — откроется диалоговое окно Символ. В списке Шрифт выберите шрифт Symbol. В таблице символов разыщите и выберите символ π. Щелкните на кнопке Сочетание клавиш — откроется диалоговое окно Настройка клавиатуры. Убедитесь в том, что текстовый курсор находится в поле Новое сочетание клавиш (в таких случаях говорят, что *фокус ввода* принадлежит элементу управления Новое сочетание клавиш). Если это не так, переместите фокус ввода в нужное поле последовательными нажатиями клавиши TAB.

   Когда фокус ввода находится в нужном поле, нажмите желаемую комбинацию клавиш, например CTRL+ALT+P. Обратите внимание на запись, появившуюся в поле, и щелкните на кнопке Назначить. Закройте открытые диалоговые окна и проверьте работу данной комбинации.

   Обратите внимание на то, что для одного и того же символа можно назначать несколько комбинаций клавиш. Если нужно изменить назначение, следует в диалоговом окне Настройка клавиатуры выделить назначенную комбинацию и щелкнуть на кнопке Удалить. Если нужно, чтобы назначенная комбинация действовала во всех вновь создаваемых документах, ее можно сохранить в текущем шаблоне, выбрав его в раскрывающемся списке Сохранить изменения.

5. *Использование средства автозамены*. У метода «горячих клавиш» есть существенный недостаток: надо запоминать, какому символу какая комбинация соответствует. Если предполагается ввод множества нестандартных символов, удобно использовать средство автоматической замены символов при вводе.

   Дайте команду Вставка ▸ Символ — откроется диалоговое окно Символ. В списке Шрифт выберите шрифт Symbol. В таблице символов разыщите и выберите символ π. Щелкните на кнопке Автозамена — откроется диалоговое окно Автозамена. В поле Заменить введите заменяемую комбинацию «.пи.». (Зачем символы «пи» оконтурены точками с двух сторон, выясните самостоятельно, экспериментируя с вводом выражения 2πR). Аналогичным образом можно организовать ввод и других символов: «.фи.», «.тау.», «.кси.» и т. д. Как видите, ничего не надо специально запоминать.

В текстовом процессоре Microsoft Word, как и во многих других приложениях Windows, одну и ту же операцию можно выполнить множеством разных способов. У каждого способа есть достоинства и недостатки. Пользователи опытным путем подбирают наиболее удобные для себя приемы. Выбор приема зависит от объема и характера исполняемой работы, а также от периодичности ее исполнения.

В предыдущей главе мы рассмотрели приемы создания *простых* текстовых документов средствами текстового процессора *Microsoft Word*. К условной категории *простых* эти документы были отнесены только потому, что не содержали объектов, встроенных в текст. Соответственно, нами не были рассмотрены вопросы взаимодействия текста и встроенных объектов.

В этой главе мы рассмотрим приемы создания *комплексных* текстовых документов, содержащих специальные элементы оформления и встроенные объекты нетекстовой природы (формулы, таблицы, диаграммы, художественные заголовки, растровые и векторные иллюстрации, а также объекты мультимедиа).

11.1. Приемы управления объектами Microsoft Word

Особенности объектов Word

Текстовый процессор *Word XP* обладает развитой функциональностью по работе с объектами нетекстовой природы. Среди встроенных объектов могут быть стандартные объекты, созданные другими программами (рисунки, анимационные и звуковые клипы и многое другое), а также объекты, созданные средствами самого текстового процессора. В частности, программа позволяет создавать и встраивать геометрические фигуры, художественные заголовки, диаграммы, формульные выражения, заготовленные векторные иллюстрации (клипарты), то есть в ней имеются средства, отдаленно напоминающие средства специализированных графических редакторов. Правда, среди этих средств нет ничего для создания и обработки растровых иллюстраций — их можно только импортировать из других программ, но зато есть средства для управления их визуализацией, например для изменения яркости, контрастности и масштаба изображения.

Несмотря на столь разностороннюю природу объектов, с которыми может работать текстовый процессор *Word XP*, у них есть общие свойства, например такие, как размер, положение на странице, характер взаимодействия с текстом. Сначала мы остановимся на изучении самых общих свойств встроенных объектов, не обсуждая

их природу, — это поможет освоить базовые приемы работы с объектами. А с конкретными свойствами конкретных объектов мы познакомимся чуть позже. Но перед тем как приступать к изучению приемов работы с объектами *Word XP*, необходимо сделать важное замечание о целесообразности их применения. На этот счет существуют весьма противоречивые мнения.

1. Все объекты *Microsoft Word XP* безусловно можно использовать, если документ готовится для печати, то есть предполагается, что он будет передаваться заказчику или распространяться в виде бумажной копии, выполненной на принтере. Оформление документов с помощью встроенных объектов позволяет сделать их *представительными*.
2. Если документ предполагается передать в виде файла для последующей обработки (а именно так передают рукописи в редакции), то все собственные средства программы по созданию и размещению встроенных объектов не только бесполезны, но и вредны. Это связано с тем, что объекты *Microsoft Word XP* не стандартны и не поддерживаются профессиональными программами. Компания *Microsoft* имеет лидирующее положение в отрасли и может не считаться с общепринятыми стандартами и правилами, а внедрять свои. Поэтому объекты, созданные в программах этой компании, могут полноценно использоваться только в других программах той же компании.
3. Из последнего замечания вытекает еще одно направление для использования объектов, созданных в *Microsoft Word*. Их можно успешно экспортировать через буфер обмена *Windows* в другие программные продукты, входящие в пакет *Microsoft Office XP*, например такие, как система управления электронными таблицами *Excel*, система управления базами данных *Access* и другие.

Взаимодействие объектов Word с текстом и страницей

Управление размером и положением объекта. Взгляните на рис. 11.1. Здесь представлен графический объект, встроенный в текст документа. Этот объект обладает рядом свойств. Самое очевидное свойство — его размер. Когда объект выделен,

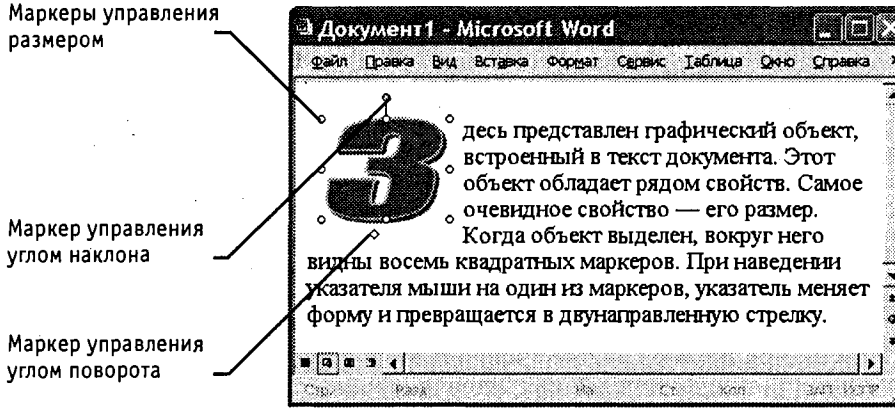


Рис. 11.1. Пример объекта, встроенного в текст

вокруг него видны восемь квадратных маркеров. При наведении указателя мыши на один из маркеров указатель меняет форму и превращается в двунаправленную стрелку. В этот момент размер объекта можно менять методом протягивания мыши. Угловые маркеры позволяют пропорционально изменять размер объекта как по горизонтали, так и по вертикали. Четыре маркера, расположенные на сторонах воображаемого прямоугольника, позволяют управлять размером по одному направлению (по вертикали или горизонтали).

При наведении указателя мыши на сам объект указатель меняет форму и превращается в четырехнаправленную стрелку. В таком состоянии объект можно перетаскивать с помощью мыши по рабочему полю документа. Он займет новое положение в тот момент, когда левая кнопка мыши будет отпущена после перетаскивания.

Расширенное управление свойствами объектов. Вручную мы можем только управлять размером, поворотом и положением объекта на странице. Для управления всеми остальными свойствами объектов нужны дополнительные средства — их можно найти в двух местах:

- на панели инструментов, соответствующей типу объекта (она открывается автоматически, когда объект выделен);
- в диалоговом окне Формат объекта (рис. 11.2), которое открывают из контекстного меню объекта (после щелчка правой кнопкой мыши на объекте).

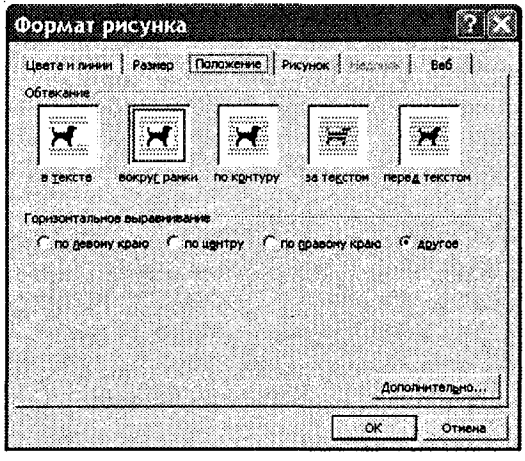


Рис. 11.2. Основное средство управления общими параметрами встроенного объекта

С помощью панели инструментов управляют индивидуальными свойствами объектов (у разных типов объектов они различны), а с помощью диалогового окна Формат объекта управляют наиболее общими свойствами, имеющимися у объектов любых типов.

Взаимодействие объекта с окружающим текстом. Вставив объект в текст, следует задать характер его взаимодействия с текстом. Средства для этого представлены на вкладке Положение диалогового окна Формат объекта. Возможны следующие варианты.

1. Вариант В тексте используют для графических объектов малого размера, сопоставимого с размерами символов текста. В этом случае объект вставляется в текстовую строку на правах графического символа и далее перемещается по странице только вместе с текстом.
2. Вариант Вокруг рамки использован в примере на рис. 11.2. В этом случае текст располагается вокруг воображаемой прямоугольной рамки, охватывающей весь контур объекта.
3. Вариант По контуру отличается от предыдущего тем, что воображаемая прямоугольная рамка не проводится и текст плавно обтекает контур объекта (если он криволинейный).
4. Вариант Перед текстом — это прием вставки объекта без обтекания. Текст и объект лежат на разных слоях, причем объект лежит выше и загораживает часть текста. Этим приемом пользуются, когда оформление важнее содержания.
5. Вариант За текстом — это еще один прием вставки объекта без обтекания. Текст и объект тоже лежат на разных слоях, но в данном случае объект лежит на нижнем слое и загораживается текстом. Этот вариант используют для размещения текста на тематическом художественном фоне.

Дополнительные варианты взаимодействия текста со встроенным объектом можно найти в диалоговом окне Дополнительная разметка, которое открывают с помощью кнопки Дополнительно.

6. Вариант Сквозное — это прием обтекания, аналогичный обтеканию По контуру, но в данном случае текст обтекает объект не только снаружи, но и изнутри.
7. Там же, в диалоговом окне Дополнительная разметка можно выбрать вариант обтекания Сверху и снизу. Этот прием используют наиболее часто — его считают основным для объектов, ширина которых составляет более половины ширины страницы.

Прочие параметры взаимодействия объекта с окружающим текстом. Более тонкую настройку взаимодействия объектов с текстом выполняют с помощью элементов управления, имеющихся в диалоговом окне Дополнительная разметка. В частности, здесь можно с помощью переключателей конкретно указать, с каких сторон объекта происходит обтекание, а с каких — нет. Здесь же можно указать величину интервала в миллиметрах между текстом и объектом.

Управление горизонтальным положением объекта относительно элементов печатной страницы. Завершив настройку взаимодействия объекта с текстом, приступают к размещению объекта на странице. Как уже говорилось выше, это можно сделать вручную методом перетаскивания объекта с помощью мыши, но более точную настройку выполняют с помощью рассмотренного диалогового окна Формат объекта ▸ Положение.

Варианты горизонтального размещения объекта:

- по левому краю;
- по правому краю;
- по центру;
- другое.

Варианты По левому краю и По правому краю обычно используют при обтекании По контуру или Вокруг рамки. Вариант По центру часто сочетают с обтеканием Сверху и снизу, а последний вариант соответствует ручному размещению объекта перетаскиванием с помощью мыши.

Управление вертикальным положением объекта относительно элементов печатной страницы. К объекту, встроенному в текст, можно подходить с двух позиций: как к элементу оформления страницы или как к элементу оформления содержания, то есть текста. Разница заключается в том, что происходит с объектом во время редактирования текста: он перемещается вместе с ним (с абзацами, к которым он примыкает) или он неподвижен, а текст перемещается, обтекая объект по заданным правилам.

В первом случае объект надо закрепить относительно абзаца, а во втором случае — относительно страницы. Необходимую настройку выполняют элементами управления вкладки Положение рисунка в диалоговом окне Дополнительная разметка. Вертикальное положение объекта относительно элементов страницы задают установкой переключателя Выравнивание и выбором метода выравнивания и элемента, относительно которого происходит выравнивание. Вертикальное положение относительно текста задают установкой переключателя Положение и выбором объекта, относительно которого положение задается, например абзаца.

Чтобы объект был связан с элементом страницы и не перемещался вместе с текстом, устанавливают флажок Установить привязку. Чтобы объект мог перемещаться вместе с текстом, устанавливают флажок Перемещать вместе с текстом.

Управление свойствами объектов Microsoft Word

Управление размерами объекта. Мы знаем, что размерами встроенных объектов можно управлять перетаскиванием графических маркеров с помощью мыши. Это прием ручного управления. Однако существуют и приемы автоматического управления. Их реализуют с помощью элементов управления вкладки Размер рассмотренного выше диалогового окна Формат объекта. Счетчиками Высота, Ширина и Поворот задают вертикальные и горизонтальные размеры объекта, а также его угол поворота по часовой стрелке.

Размерами объектов можно управлять не только в абсолютном исчислении, но и в относительном (в процентах относительно исходного). Для этого служат счетчики группы Масштаб. Чтобы размеры объекта синхронно изменялись по вертикали и горизонтали, надо установить флажок Сохранить пропорции.

Управление свойствами линии. Большинство объектов, создаваемых средствами самой программы *Word XP*, имеют векторную природу, то есть в их основе лежат простейшие геометрические фигуры — линии. Эти линии, в свою очередь, имеют собственные свойства: толщину, цвет и тип. Управление этими свойствами выполняют с помощью средств вкладки Формат объекта ▸ Цвета и линии.

Управление свойствами замкнутых линий. Замкнутые линии, в отличие от обычных, обладают дополнительным свойством — *заливкой*. Свойство заливки задают

на вкладке Формат объекта ▸ Цвета и линии. Заливка может быть *простой* и *комбинированной*. Вид заливки выбирают в раскрывающейся палитре Цвет.

Простая заливка — одноцветная. Цвет заливки может быть одним из сорока стандартных, имеющихся в палитре, или одним из дополнительных (выбирается в палитре с помощью кнопки Другие цвета). Простые цвета отличаются тем, что их можно назначить полупрозрачными, — тогда через закрашенные контуры может просвечивать текст или объект нижележащего слоя (рис. 11.3).

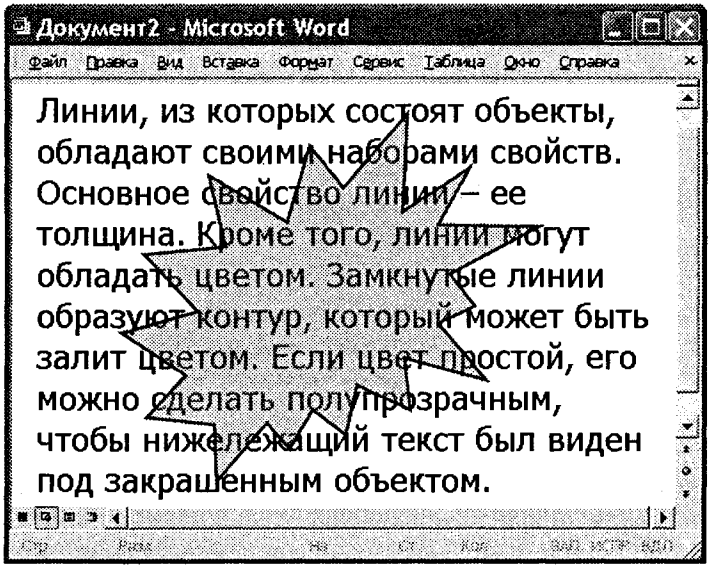


Рис. 11.3. Объекты, залитые сплошным цветом, можно сделать полупрозрачными

Комбинированная заливка имеет более сложный характер. В программе *Word XP* реализовано четыре метода комбинированной заливки:

- градиентная заливка;
- текстурная заливка;
- заливка узором;
- заливка рисунком (изображением-картой).

Для выбора метода комбинированной заливки в палитре цветов имеется кнопка Способы заливки. Она открывает диалоговое окно Способы заливки, имеющее четыре вкладки: Градиентная, Текстура, Узор и Рисунок.

Градиентная заливка — это многоцветная заливка, при которой осуществляется плавный переход между заданными цветами. Количество исходных цветов, сами цвета и направление градиента произвольно выбираются на вкладке Градиентная.

Текстурная заливка — это заливка, воспроизводящая нерегулярную текстуру. Обычно используется для имитации поверхности материала. Выбор текстуры выполняют на вкладке Текстура (рис. 11.4). Если представленных там текстур недостаточно, с помощью кнопки Другая текстура можно загрузить графический файл с изображением дополнительной текстуры.

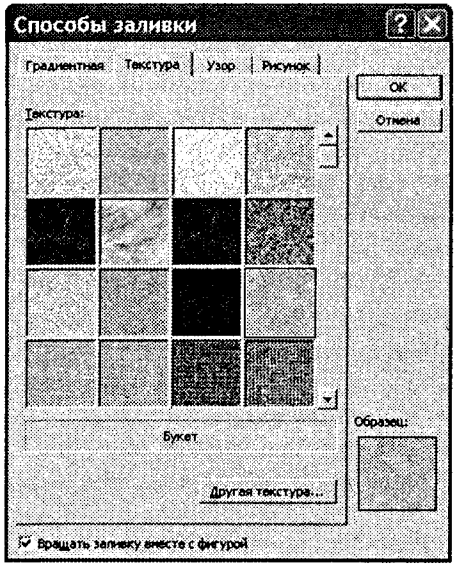


Рис. 11.4. Выбор текстуры для заливки замкнутых контуров

Заливка узором, как и заливка текстурой, — это заливка заранее подготовленным изображением, но имеющим регулярный характер. Выбор узора выполняют на вкладке Узор. Там же можно настроить цвет переднего плана рисунка узора и цвет его фона.

Заливка изображением-картой — это аналог текстурной заливки, при котором замкнутый контур заполняется специально подготовленным графическим изображением. Выбор изображения выполняют выбором файла, в котором оно хранится. Для этого служит вкладка Рисунок.

Взаимодействие объектов друг с другом

Мы рассмотрели, как происходит взаимодействие объектов с текстом и с элементами печатной страницы, но если на одной странице имеется несколько встроенных объектов, то они могут взаимодействовать и друг с другом. Характером этого взаимодействия тоже нужно управлять.

Первое, что нужно решить, — это разрешено ли объектам перекрывать друг друга. Для тех объектов, которым перекрытие разрешено, следует установить флажок Формат объекта ▸ Положение ▸ Дополнительно ▸ Положение объекта ▸ Разрешить перекрытие. Напомним, что доступ к диалоговому окну Формат объекта открыва-

ется командой (для разных объектов она может называться по-разному) контекстного меню объекта.

Управление взаимным положением объектов выполняют с помощью операций:

- группирования;
- задания порядка следования;
- выравнивания;
- распределения.

Группирование объектов. Если на странице представлено несколько объектов и при этом важно строго зафиксировать их взаимное расположение, то их объединяют в один комплексный (групповой) объект с помощью операции группирования. После этой операции свойства группового объекта можно настраивать точно так же, как мы настраивали свойства простейших объектов, — ему может быть задан характер обтекания текстом, метод привязки к абзацу или к элементам печатной страницы и т. п.

Для группирования нескольких объектов их следует выделить (выделение нескольких объектов выполняют при нажатой клавише SHIFT), щелкнуть на любом из объектов группы правой кнопкой мыши и выбрать в контекстном меню команду Группировка ▸ Группировать. Сгруппированные объекты можно перемещать как единое целое. Чтобы разгруппировать объекты и получить доступ к индивидуальным свойствам каждого из них, надо выделить группу и дать команду Группировка ▸ Разгруппировать.

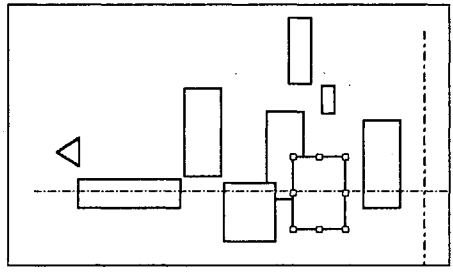
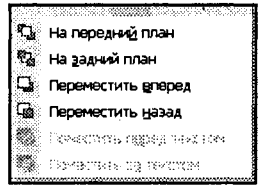


Рис. 11.5. Разгруппированный комплексный объект

Управление порядком следования объектов. Если на странице документа размещается несколько объектов, то предполагается, что у каждого объекта есть свой *слой*. По умолчанию порядок следования слоев связан с порядком создания объектов, то есть те объекты, которые были созданы раньше, лежат на слоях ниже, чем объекты, созданные позже. Если между объектами нет перекрытия, то мы не замечаем, что существует некий порядок следования объектов, однако, когда объекты перекрывают друг друга, этот порядок становится заметен.

Управляют порядком следования объектов с помощью команды Порядок контекстного меню. Она открывает вложенное меню, средствами которого можно поднять объект на передний план, опустить на задний план, сместить на один слой вверх или вниз и задать положение объекта относительно текста.



Выравнивание объектов. Если объекты, составляющие композицию, не перекрывают друг друга, важно иметь средство их относительного выравнивания между собой. Выравнивание объектов выполняют до группирования, ведь после него

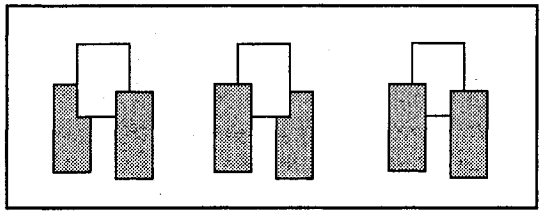


Рис. 11.6. Управление порядком следования

объекты уже нельзя сдвинуть друг относительно друга. В этом случае операция группирования закрепляет взаимное расположение объектов. После нее объекты уже не могут сдвинуться друг относительно друга, и положением всей группы на странице можно управлять как единым целым. Чтобы выполнить выравнивание, необходимо предварительно открыть дополнительную панель инструментов Рисование (Вид ▸ Панели инструментов ▸ Рисование).

Для выравнивания нескольких объектов между собой их следует выделить при нажатой клавише SHIFT, а затем дать команду Действия ▸ Выровнять/распределить (с помощью кнопки Действия панели инструментов Рисование). Существует шесть методов выравнивания. Им соответствуют три команды горизонтального выравнивания (По левому краю, По правому краю, По центру) и три команды выравнивания вертикального (По верхнему краю, По нижнему краю, По середине). Следует обратить внимание на особенность действия команд выравнивания. Так, например, если два объекта выравниваются по *нижнему* полю, значит, они выравниваются по *нижнему* полю *нижнего* объекта. Выравнивание по *правому* полю — это выравнивание по *правому* полю самого *правого* объекта из числа выделенных и так далее. Если необходимо выполнить выравнивание относительно полей страницы, следует предварительно установить флажок меню Действия ▸ Выровнять/распределить ▸ Относительно страницы.

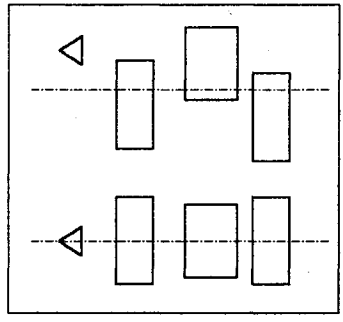


Рис. 11.7. Выравнивание «по середине»

Распределение объектов. Эта операция родственна выравниванию. Ее суть в том, что между объектами устанавливаются равные интервалы по горизонтали или (и) вертикали. Соответственно, в меню команды Действия ▸ Выровнять/распределить имеются команды: Распределить по горизонтали и Распределить по вертикали.

Равномерное распределение объектов обычно выполняют после выравнивания, но, разумеется, до группирования. Нередко объекты выравнивают по вертикали и одновременно равномерно распределяют по горизонтали или, соответственно, наоборот. Дополнительное отличие команд распределения от команд выравнивания заключается еще и в том, что для взаимного выравнивания достаточно иметь два выделенных объекта, а для команд распределения должно быть выделено не менее трех объектов.

11.2. Ввод формул

Необходимость в наличии средства для ввода математических выражений в текстовый документ характерна для научно-технической документации. Одним из таких средств является специальное приложение *Mathcad*, представленное в главе 18. Но функции системы *Mathcad* намного шире, и есть немало оснований для того, чтобы иметь простое средство ввода формул в самом текстовом процессоре.

В программе *Microsoft Word* таким средством является редактор формул *Microsoft Equation* 3.0. Он позволяет создавать формульные объекты и вставлять их в текстовый документ. При необходимости вставленный объект можно редактировать непосредственно в поле документа.

Запуск и настройка редактора формул

Для запуска редактора формул служит команда Вставка ▸ Объект. В открывшемся диалоговом окне Вставка объекта следует выбрать пункт Microsoft Equation 3.0 в списке Тип объекта на вкладке Создание. Откроется панель управления Формула, представленная на рис. 11.8. При этом строка меню текстового процессора замещается строкой меню редактора формул.

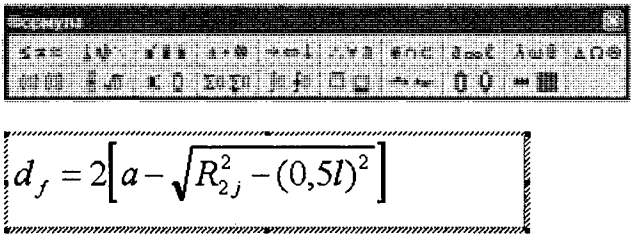


Рис. 11.8. Панель управления редактора формул

Прежде чем пользоваться редактором формул, следует выполнить его настройку. Настройка состоит в назначении шрифтов для различных элементов, входящих в формулы. Она выполняется в диалоговом окне Стили, открываемом командой Стиль ▸ Определить (рис. 11.9). Эта настройка является обязательной — без нее редактор формул работать не будет, но выполнить ее достаточно только один раз.

Прочие (необязательные) настройки редактора формул выполняют в диалоговом окне Интервал (Формат ▸ Интервал). Многочисленные средства настройки, присутствующие в нем, предназначены для задания размеров различных элементов формул.

Панель инструментов редактора формул содержит два ряда кнопок. Кнопки нижнего ряда создают своеобразные шаблоны, содержащие поля для ввода символов. Так, например, для ввода обыкновенной дроби следует выбрать соответствующий шаблон, имеющий два поля: числитель и знаменатель. Заполнение этих полей может производиться как с клавиатуры, так и с помощью элементов управления верхней строки. Переходы между полями выполняются с помощью клавиш управления курсором.

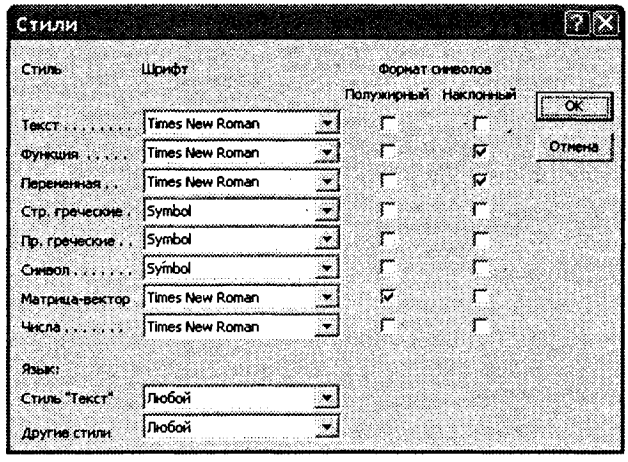


Рис. 11.9. Пример обязательных настроек редактора формул

Ввод и редактирование формул завершается нажатием клавиши ESC или закрытием панели редактора формул. Можно также щелкнуть левой кнопкой мыши где-либо в поле документа вне области ввода формулы. Введенная формула автоматически вставляется в текст в качестве объекта. Далее ее можно переместить в любое иное место документа через буфер обмена (CTRL+X — вырезать; CTRL+V — вставить). Для редактирования формулы непосредственно в документе достаточно выполнить на ней двойной щелчок. При этом автоматически открывается окно редактора формул.

Особенности редактора формул

1. Редактор формул *Microsoft Equation* 3.0 представляет собой отдельный компонент, поэтому при установке текстового процессора требуется специально указать необходимость его подключения.
2. При работе с редактором формул следует стремиться к максимальной полноте вводимых выражений. Так, например, выражение (формула) может содержать компоненты, ввод которых возможен и без использования редактора формул, но для удобства работы и простоты дальнейшего редактирования следует вводить всю формулу целиком только в редакторе формул, не используя иные средства.

$$S=\boxed{\frac{at^2}{2}}$$ — неправильно; $\boxed{S=\frac{at^2}{2}}$ — правильно.

3. При вводе формул и выражений не рекомендуется использовать символы русского алфавита. В тех случаях, когда они необходимы, например, в качестве описательных индексов переменных, им следует назначать стиль Текст.

$$E_{\text{кинет.}}=\frac{mV^2}{2}$$

4. В редакторе формул не работает клавиша ПРОБЕЛ, поскольку необходимые интервалы между символами создаются автоматически. Однако если необходимость ввода пробелов все-таки возникнет, то их можно вводить с помощью кнопки Пробелы и многоточия панели инструментов Формула. Всего предусмотрено пять разновидностей пробелов различной ширины.

11.3. Работа с таблицами

Данные, представленные в табличной форме, отличаются наглядностью. Таблицы всегда были неотъемлемым атрибутом печатной научно-технической документации, а в последние годы стали и эффективным средством оформления *Web*-страниц Интернета. Это связано с тем, что в силу естественных причин возможности форматирования *Web*-страниц весьма ограниченны. Поэтому многие *Web*-дизайнеры используют таблицы (в том числе и скрытые), чтобы принудительно управлять отображением данных на экране клиента и не доверять этот ответственный процесс средству просмотра *Web* (браузеру). Так, например, таблицы — это простейшее средство для имитации на *Web*-странице газетного или журнального текста, имеющего две и более колонок.

Ячейки таблиц могут содержать не только текст, но и графические и прочие объекты. Благодаря этому можно размещать несколько иллюстраций по ширине *Web*-страницы (обычные средства форматирования *Web*-страниц не позволяют это сделать).

При создании страниц можно управлять методом представления ячеек и рамок, как внешних, так и внутренних. При создании печатных документов таблицы оформляют так, чтобы они соответствовали стилю и содержанию документа. При создании *Web*-страниц существует прием, когда рамки вообще не отображают, а между ячейками делают зазор. В результате этого объекты, находящиеся в ячейках, образуют ровные регулярные структуры на экране, в то время как никаких следов таблиц на экране не видно (рис. 11.10).

Текстовый процессор *Microsoft Word* обладает удивительно гибкими и мощными средствами создания таблиц как для печатных, так и для электронных документов.

Три основных средства создания таблиц — это:

- кнопка Добавить таблицу на панели инструментов Стандартная;
- диалоговое окно Вставка таблицы (Таблица ▸ Вставить ▸ Таблица);
- средство рисования таблиц Таблицы и границы (Таблица ▸ Нарисовать таблицу).

Создание таблиц

Кнопку Добавить таблицу используют для создания простейших таблиц небольшого размера. Созданные таким методом таблицы можно в дальнейшем развивать, по мере необходимости увеличивая в них количество строк и столбцов командами меню Таблица ▸ Вставить.

Команду Таблица ▸ Вставить ▸ Таблица используют для создания более сложных таблиц. Она открывает диалоговое окно Вставка таблицы, в котором задают число

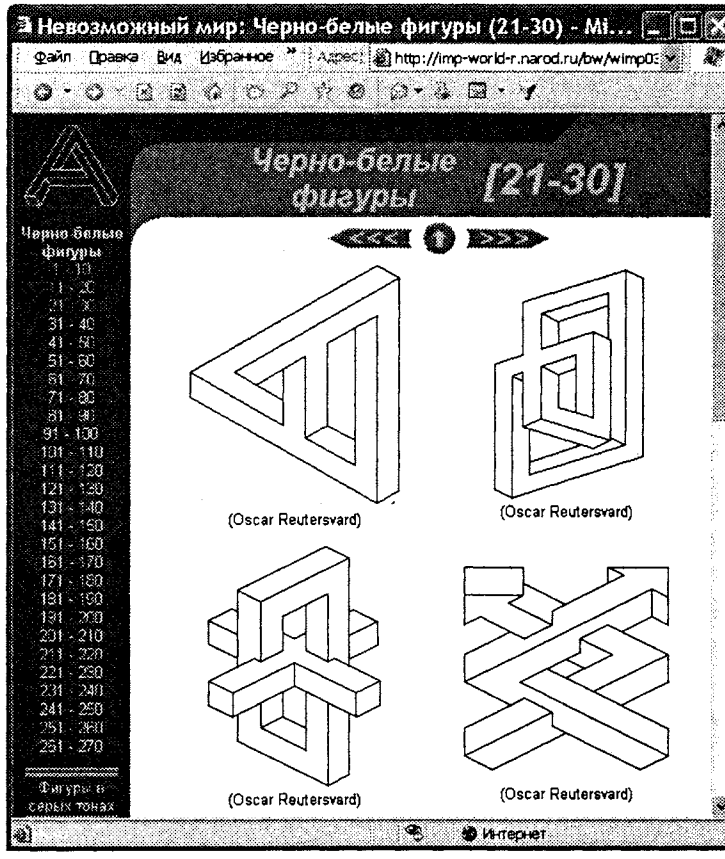


Рис. 11.10. Подобное оформление Web-страниц достигается путем использования таблиц

строк и столбцов, а также ширину столбцов. Если вместо конкретного размера задать параметр Авто, включается режим Автоподбор, благодаря которому столбцы могут эластично форматироваться в соответствии с имеющимся содержанием. Режим автоподбора задают соответствующим переключателем:

- постоянная ширина — общая ширина таблицы равна ширине поля набора документа, а ширина каждого столбца постоянна и зависит от количества столбцов (режим удобен при создании печатных документов);
- по содержимому — ширина каждого столбца пропорциональна объему данных, содержащихся в нем (режим удобен при создании электронных документов, распространяемых в формате текстового процессора);
- по ширине окна — специальный режим для таблиц, размещаемых на *Web*-страницах (окончательное форматирование таблицы происходит не в момент ее создания, а во время просмотра).

Таблицы сложной структуры удобно создавать методом «рисования». Необходимые для этого элементы управления сосредоточены на панели инструментов Таблицы и границы (открывается командой Таблица ▸ Нарисовать таблицу). Порядок действий, необходимых для создания таблиц этим методом, рассмотрен в упражнении 11.1.

Редактирование таблиц

Говоря о редактировании таблиц, мы имеем в виду не редактирование их содержимого, а только редактирование их структуры. Редактирование содержимого осуществляется обычными средствами, рассмотренными в предыдущей главе. Фактически редактирование структуры таблиц сводится к следующим операциям:

- добавление заданного количества строк;
- добавление заданного количества столбцов;
- удаление выделенных ячеек, строк и столбцов;
- слияние выделенных ячеек;
- разбиение выделенных ячеек.

Комбинируя вышеуказанные операции, можно на базе таблиц с простой структурой готовить таблицы, имеющие сложную структуру. Средства для выполнения этих операций находятся в меню Таблица (возможно, потребуется раскрыть *расширенное меню*) или доступны через контекстные меню выделенных объектов.

Форматирование таблиц

При работе с таблицами следует различать *форматирование таблиц* и *форматирование содержимого*. В первом случае происходит управление размерами структурных элементов таблицы (ячеек, строк, столбцов и т. п.), а во втором — управление размещением содержимого ячеек.

Форматирование таблиц можно выполнять в командном или интерактивном режиме. В командном режиме для этой цели используют диалоговое окно Свойства таблицы (Таблица ▸ Свойства таблицы). Его можно открыть и из контекстного меню таблицы, если щелкнуть в ее пределах правой кнопкой мыши. Элементы управления вкладок диалогового окна Свойства таблицы позволяют:

- задать метод выравнивания таблицы относительно страницы документа (Таблица ▸ Свойства таблицы ▸ Таблица ▸ Выравнивание);
- задать метод взаимодействия таблицы с окружающим текстом (Таблица ▸ Свойства таблицы ▸ Таблица ▸ Обтекание);
- определить или переопределить вариант оформления внешних и внутренних рамок таблицы, а также настроить характер оформления ячеек (Таблица ▸ Свойства таблицы ▸ Таблица ▸ Границы и заливка);
- задать размеры внутренних полей в ячейках и интервалы между ячейками (Таблица ▸ Свойства таблицы ▸ Таблица ▸ Параметры);
- назначить параметры текущей строки или выделенных строк (Таблица ▸ Свойства таблицы ▸ Строка);

- назначить параметры текущего столбца или выделенных столбцов (Таблица ▸ Свойства таблицы ▸ Столбец);
- назначить параметры текущей ячейки или выделенных ячеек (Таблица ▸ Свойства таблицы ▸ Ячейка).

В интерактивном режиме таблицу форматируют с помощью маркеров, появляющихся при наведении указателя мыши на таблицу или ее элементы. Маркер в левом верхнем углу таблицы позволяет перемещать таблицу по рабочему полю документа. Маркер в правом нижнем углу позволяет управлять общими размерами таблицы. Маркеры изменения размера, появляющиеся при наведении указателя мыши на рамки таблицы, позволяют интерактивно изменять размеры столбцов и строк методом перетаскивания.

Ввод и форматирование содержимого таблиц

Выделение нужной ячейки для ввода текста выполняют с помощью мыши. Отдельную ячейку выделяют тройным щелчком левой кнопки. Перемещение между ячейками выполняют клавишей TAB (к следующей ячейке) или комбинацией SHIFT+TAB (к предыдущей ячейке). Для навигации по ячейкам таблицы можно также использовать клавиши управления курсором. В тексте курсорные клавиши выполняют перемещение курсора внутри ячейки, но по достижении границы текста они позволяют переходить к соседним ячейкам.

Все команды форматирования текста относятся к выделенному элементу. Выделенным элементом может быть любая ячейка, строка (группа строк), столбец (группа столбцов) или вся таблица в целом. Группы ячеек выделяют методом протягивания мыши. Большинство команд, связанных с форматированием элементов таблицы и содержащихся в них объектов, можно выполнить с помощью панели инструментов Форматирование.

Автоматическое форматирование таблиц

Автоматическое форматирование таблиц выполняют с помощью встроенного средства Автоформат (рис. 11.11), которое запускается командой Таблица ▸ Автоформат таблицы (при наличии выделенной таблицы). Набор предлагаемых форматов представлен в списке Стили таблиц, а результат, получающийся при их использовании, — в поле Образец. Работа по форматированию таблицы полностью автоматизирована и сводится к тому, чтобы выбрать такой формат и так установить сопутствующие элементы управления, чтобы представленный образец наиболее соответствовал запланированному результату.

11.4. Работа с диаграммами

Диаграммы являются удобным средством визуального представления данных и наряду с таблицами очень широко используются в научно-технической документации. Для создания диаграмм текстовый процессор *Microsoft Word* имеет подключаемое средство *Microsoft Graph*. Как и описанный выше редактор формул *Microsoft Equation 3.0*, эта программа является внешним компонентом, и ее установка должна специально заказываться при установке текстового процессора.

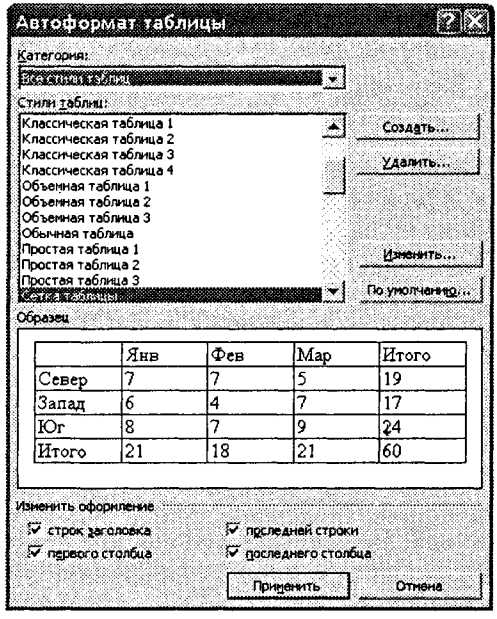


Рис. 11.11. Средство автоматического форматирования таблиц

Текстовый процессор *Microsoft Word XP* предоставляет два метода для вставки диаграмм в документ. Более общий метод основан на том, что сначала в документ вставляется некая произвольная диаграмма, с которой связана некая произвольная *базовая таблица* данных. Далее производится настройка диаграммы, которая состоит в настройке внешнего вида и в редактировании содержания. Поскольку содержание основано на базовой таблице, то оно редактируется путем заполнения этой таблицы нужными данными.

Второй, частный метод основан на том, что диаграмма создается на базе конкретной таблицы, имеющейся в документе. В этом случае настройка диаграммы состоит только в настройке внешнего вида. Этот метод очевидно более удобен, но злоупотреблять им не следует, поскольку данные в таблице и диаграмме дублируют друг друга, а не во всяком документе это оправдано. Приемы создания диаграмм на базе таблиц документа мы рассмотрим в упражнении 11.2.

Создание базовой диаграммы

Создание диаграммы начинается с создания базовой диаграммы командой Вставка▸Объект. В открывшемся диалоговом окне Вставка объекта следует выбрать пункт Microsoft Graph Chart, после чего в документ вставляется диаграмма, с которой связана некая *базовая таблица* (рис. 11.12). Рассматривайте эту таблицу как шаблон. Ее ячейки следует заполнить собственными данными, причем заполнение можно автоматизировать путем импорта данных из какой-либо иной таблицы, например из таблицы *Microsoft Excel*.

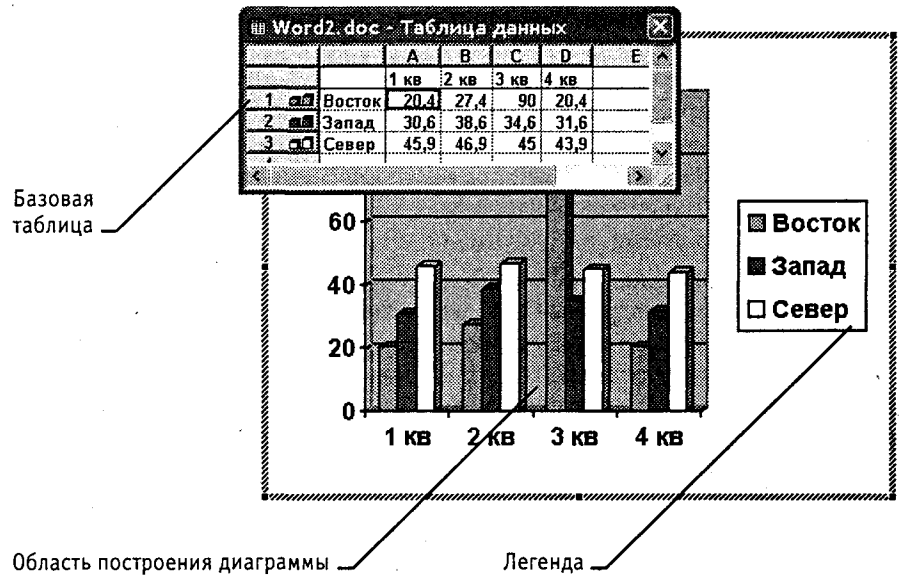


Рис. 11.12. Сначала в документ вставляется произвольная диаграмма и связанная с ней таблица. Далее диаграмма и таблица редактируются по месту

Настройка внешнего вида диаграммы

Существует множество различных типов диаграмм и графиков, отличающихся способом визуального представления связанных с ними данных. Выбор типа диаграммы производят в диалоговом окне Тип диаграммы (Диаграмма ▸ Тип диаграммы), которое имеет пару вкладок (для стандартных и нестандартных типов диаграмм).

Тип диаграммы выбирают в поле Тип, просматривая при этом внешний вид образцов в поле Вид. Выбрав форму диаграммы, приступают к ее настройке. Настройка диаграммы состоит в выборе *элементов оформления диаграммы* и *элементов представления данных* и выполняется в диалоговом окне Параметры диаграммы (Диаграмма ▸ Параметры диаграммы).

Элементы представления данных — это точки на графиках, столбцы гистограмм, секторы круговых диаграмм — в общем, все то, что служит для непосредственного отображения данных. *Элементы оформления* — это название диаграммы, названия ее осей, «легенда» (специальное поле, в котором приведены условные обозначения для групп элементов данных), подписи к элементам данных и линии координатной сетки. Настройку выполняют подключением или отключением тех или иных элементов.

Элементы диаграммы бывают *связанными* или *присоединенными*. Так, например, название диаграммы, названия ее осей и легенду можно редактировать отдельно — это *присоединенные элементы оформления*. Подписи к элементам данных редактировать на диаграмме нельзя — они связаны со значениями в базовой таблице и потому считаются *связанными элементами*.

Для каждого из присоединенных элементов оформления можно выполнить индивидуальное форматирование. Для этого надо в поле диаграммы щелкнуть дважды на поле присоединенного элемента — откроется соответствующее диалоговое окно форматирования (Формат легенды, Формат оси, Формат названия диаграммы, Формат области диаграммы и т. д.). Состав вкладок и других элементов управления этих диалоговых окон зависит от свойств конкретного присоединенного элемента. Так, например, средства форматирования осей диаграммы отличаются от средств форматирования ее названия.

Настройка элементов данных и элементов оформления — это как бы внутренние средства настройки диаграмм. Они определяют свойства диаграммы как объекта. Однако возможно также и редактирование объекта в целом в составе документа. Так, например, для выделенной диаграммы можно с помощью мыши изменять горизонтальный и вертикальный размеры объекта путем перетаскивания маркеров. При изменении размера диаграммы возможно автоматическое перемасштабирование ее элементов оформления.

11.5. Работа с графическими объектами

В документах *Microsoft Word* можно использовать два типа графических объектов: *рисунки* и *изображения*. На русском языке разница между этими терминами неочевидна, и мы поясним, что под ними понимается в текстовом процессоре *Word*. *Рисунки* — объекты векторной природы (линии, прямые и кривые, геометрические фигуры, стандартные и нестандартные). Простейшие средства для их создания есть в самом текстовом процессоре.

Изображения — растровые объекты. Текстовый процессор не имеет средств для их создания, поэтому они вставляются как внешние объекты из файла, подготовленного другими средствами (графическим редактором, с помощью сканера, цифровой камеры, графического планшета).

Рисунки всегда внедрены в документ — их можно редактировать непосредственно по месту. Изображения вставляют в документ методом связывания или внедрения. Их редактирование средствами текстового процессора возможно, но только в ограниченных пределах.

Работа с рисунками

Создание и редактирование рисунков. Для работы с векторными рисунками служит панель инструментов Рисование (Вид ▸ Панели инструментов ▸ Рисование). Основным средством этой панели, предназначенным для создания простейших объектов, является раскрывающийся список Автофигуры. В его категориях представлены заготовки для создания линий, прямых и кривых, простейших геометрических фигур, фигурных стрелок и выносных линий, чертежных элементов для блок-схем и функциональных схем и прочего. При создании и редактировании векторных объектов используют следующие приемы и средства.

1. Векторные объекты создают путем их выбора из категорий списка Автофигуры.
2. Их размер редактируют путем перетаскивания маркеров выделенного объекта в поле документа.

3. Удобным средством, упрощающим создание геометрических фигур, является вспомогательная координатная сетка. Командой Действия ▸ Сетка открывают диалоговое окно Привязка к сетке. В нем задают шаг сетки и способ отображения горизонтальных и вертикальных линий. Флажок Привязать к сетке обеспечивает точное позиционирование узловых точек фигур в узлах координатной сетки. Он удобен, если создаются простые (преимущественно прямолинейные) геометрические фигуры. При редактировании готовых фигур привязка к узлам сетки может создавать неудобства — в этом случае ее отключают или выполняют перемещение объектов при нажатой клавише ALT.
4. Толщина контурной линии и цвет заливки объекта относятся к свойствам объекта. Все свойства объектов можно редактировать в диалоговом окне Формат автофигуры, которое открывают командой Формат ▸ Автофигура, или через контекстное меню объекта, или двойным щелчком на самом объекте. В частности, для управления толщиной и формой контурных линий, а также параметрами заливки служат элементы управления вкладки Цвета и линии данного диалогового окна.
5. Поворотом объекта можно управлять дискретно и непрерывно. Для произвольного поворота фигуры используют команду Действия ▸ Повернуть/отразить ▸ Свободное вращение с панели инструментов Рисование. Для поворота на фиксированный угол значение угла вводят в поле счетчика Поворот на вкладке Размер диалогового окна Формат автофигуры.
6. Взаимодействие рисованного объекта с окружающим текстом может быть достаточно сложным. Так, например, текст может обтекать рисунок по заданной схеме, но он может лежать и поверх рисунка, и под ним. Выбор метода взаимодействия рисунка с текстом выполняют на вкладке Положение в диалоговом окне Формат автофигуры.

Создание надписей в поле рисунка. Рисованные объекты могут содержать текстовые элементы, например заголовки, буквенные или цифровые обозначения на схемах и чертежах. В принципе, необходимые надписи можно создать и основными средствами текстового процессора, но в этом случае очень трудно обеспечить точное положение рисунка относительно связанного с ним текста, особенно если текст не окончателен и может далее редактироваться и форматироваться. Для *Web*-страниц этот метод вообще неприемлем, поскольку они форматируются при каждом просмотре, причем непредсказуемым образом.

Для создания текстовых элементов, присоединенных к автофигурам или рисункам, служит специальное средство Надпись (Вставка ▸ Надпись). Создав автофигуру, рядом создают элемент Надпись. В поле надписи вводят необходимый текст, после чего надпись можно редактировать. Ее размер подгоняют под размер содержащегося в ней текста перетаскиванием маркеров. Прочие свойства надписи задают в диалоговом окне Формат надписи, которое для выделенной надписи открывают командой Формат ▸ Надпись. Элементы управления, представленные на вкладках этого окна, позволяют настроить:

- фоновый цвет (если задать параметр Нет заливки, надпись будет лежать на прозрачном фоне);

- цвет, тип и толщину обрамляющих линий (если при выборе цвета задать параметр Нет линий, то прочие параметры не имеют смысла);
- размеры внутренних полей между текстом и внешней рамкой поля Надпись (назначаются на вкладке Надпись).

Создав объект Надпись, его можно сгруппировать с рисунком, и тогда они будут представлять цельную композицию.

Для автофигур есть особое средство создания текстового оформления — текст может размещаться в поле автофигуры. Это выполняют командой Добавить текст в контекстном меню автофигуры. Если текст слишком велик, можно либо изменить размер автофигуры путем перетаскивания ее маркеров, либо изменить формат текста, уменьшив размер шрифта средствами панели Форматирование. Этот прием используют при создании блок-схем и функциональных схем устройств.

Работа с клипартами. Создание достаточно сложных композиций может быть очень трудоемким. В таких случаях используют готовые библиотеки (*коллекции*) рисунков (*клипартов*), в том числе и тематических. Такие библиотеки распространяются на отдельных компакт-дисках, их можно найти в Интернете, но базовая, простейшая коллекция может быть установлена вместе с текстовым процессором — она входит в комплект поставки пакета *Microsoft Office*.

Для вставки клипартов используют команду Вставка ▸ Рисунок ▸ Картинки. Соответствующая кнопка (Добавить картинку) имеется и на панели инструментов Рисование. При этом открывается Область задач в режиме Вставка картинки. Это название достаточно условное, поскольку клипарт — понятие расширенное. К клипартам относят не только графические объекты, но и звуковые клипы и видеоклипы — их тоже можно вставить в документ с помощью этого средства.

Для поиска графических клипартов раскройте список Искать объекты и оставьте флажки только в нужных категориях. Затем щелкните на кнопке Найти. На панели появятся изображения всех найденных клипартов (рис. 11.13). Разыскав нужный клипарт, его можно вставить в документ простым щелчком.

При работе с клипартами следует иметь в виду, что подобрать именно тот клипарт, который наилучшим образом соответствует характеру документа, можно далеко не всегда. Поэтому клипарты следует рассматривать не как готовые средства оформления, а как заготовки для их создания. Клипарты — это композиционные объекты. Их можно «разбирать» на составляющие, редактировать их элементы по отдельности, создавать композиции из объектов, взятых из разных клипартов. Все это выполняется путем редактирования клипартов, вставленных в документ.

Обычный порядок редактирования клипартов — следующий:

- клипарт выделяют щелчком левой кнопки мыши;
- открывают его контекстное меню щелчком правой кнопки;
- в контекстном меню выбирают команду Изменить рисунок — он открывается в режиме редактирования;
- в этом режиме работают с отдельными объектами, составляющими рисунок.

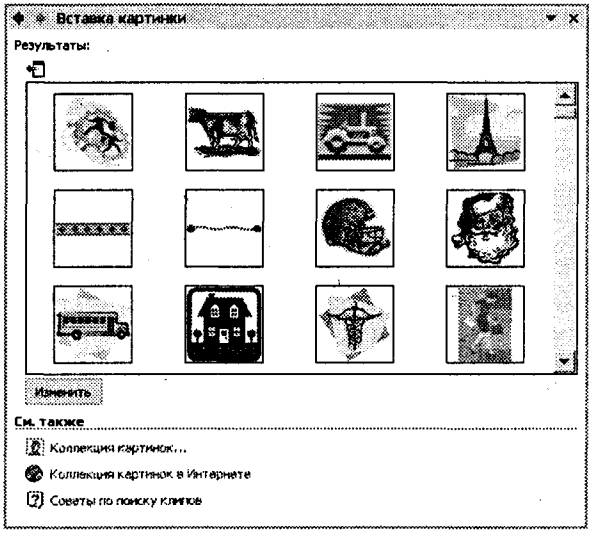


Рис. 11.13. Поиск и вставка клипартов

При работе с объектами клипарта используют команды разгруппировки и изменения порядка. Если из сложной композиции надо выделить один составляющий объект, то простейший прием состоит не в том, чтобы выделить все элементы, которые в него входят, а в том, чтобы удалить те, которые в него не входят. После каждого из удалений можно подавать отменяющую команду CTRL+Z, проверяя, что изменилось в составе рисунка. Если изменения желательны, их восстанавливают командой CTRL+Y, а если нет — переходят к выбору и удалению других элементов.

Комбинирование объектов, принадлежащих разным клипартам, выполняют путем копирования через буфер обмена *Windows* (CTRL+C и CTRL+V). При создании новых объектов из готовых клипартов часто приходится изменять размер итогового рисунка. Простейший способ для этого — воспользоваться кнопкой Подобрать размер на панели инструментов Полотно. При этой операции происходит подгонка границ рисунка по размеру содержимого.

Специальные средства оформления. Эти средства оформления представлены кнопками на панели инструментов Рисование. Они позволяют:

- управлять цветом заливки, цветом контура и цветом текста;
- управлять толщиной сплошных линий и параметрами штриха для штриховых линий;
- преобразовывать линии в стрелки и управлять формой их концов;
- создавать теневые эффекты;
- создавать трехмерные эффекты.

Для каждой из указанных кнопок открывается палитра, позволяющая настроить результат действия эффекта. Если к объекту применен теневой или трехмерный эффект, то редактировать результат этого эффекта непосредственно в поле доку-

мента нельзя, поскольку в отличие от контуров плоских объектов контуры трехмерных эффектов не являются объектами и не имеют управляющих маркеров. Поэтому для объектов, имеющих теневое или трехмерное оформление, используют иные приемы редактирования:

- выделяют объект в поле документа;
- используют кнопку Тень или Объем на панели инструментов Рисование;
- в открывшейся палитре выбирают элемент управления Настройка тени или Настройка объема;
- при этом открывается одноименная панель инструментов, посредством которых и редактируют специальные объекты.

Работа с изображениями

Под *изображениями* понимаются растровые графические объекты, исполненные посторонними программными средствами или полученные из внешнего источника. Они вставляются в документ методом связывания или внедрения. Общая команда для вставки таких объектов — Вставка ▸ Рисунок ▸ Из файла. По этой команде открывается стандартное диалоговое окно Добавление рисунка, в котором и производится выбор файла, содержащего изображение.

Выбор метода вставки. В текстовом процессоре *Microsoft Word XP* избранный рисунок можно вставить в документ тремя способами: *внедрением*, *связыванием* и *внедрением со связыванием*.

1. В первом случае объект войдет в документ и может передаваться вместе с ним.
2. Во втором случае он останется по месту своего хранения, а в документ войдет только указатель на первоисточник.
3. В третьем случае объект войдет в документ, но его связь с первоисточником сохранится. Это полезно, если предполагается возможность редактирования первоисточника и надо обеспечить синхронное редактирование и внедренного объекта.

Выбор метода вставки выполняют в диалоговом окне Добавление рисунка. В его правом нижнем углу есть раскрывающийся список, в котором следует выбрать один метод из трех возможных.

Изменение метода вставки. Если в качестве метода вставки было избрано внедрение, то ничего изменить уже нельзя. Пользователь документа, в который внедрено изображение, естественным образом лишен доступа к оригиналу. Если же при вставке был использован один из двух методов, подразумевающих связь с оригиналом, то метод изменить можно.

При выделении объекта, имеющего связь с оригиналом, в меню Правка активизируется пункт Связи, открывающий диалоговое окно Связи (рис. 11.14).

Элементы управления этого диалогового окна позволяют:

- обновить связь (если оригинал изменился);
- разорвать связь (и перейти к хранению объекта в документе);

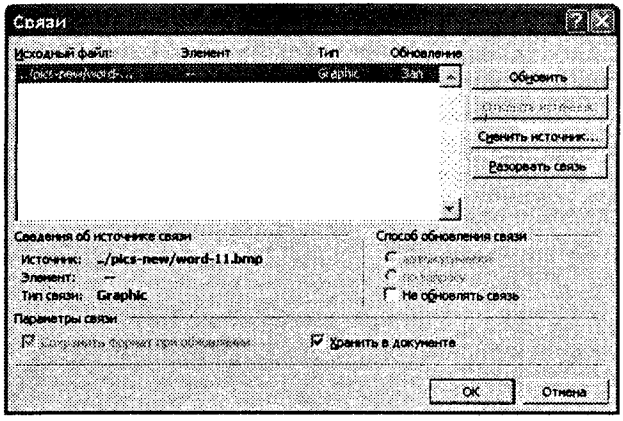


Рис. 11.14. Диалоговое окно Связи

- сменить источник (установить связь с другим объектом или с тем же объектом, но хранящимся в другом месте);
- перейти к методу одновременного внедрения и связывания путем установки флажка Хранить в документе.

Взаимодействие изображения с текстом. Основная часть инструментов для настройки свойств изображений в текстовом документе сосредоточена на панели инструментов Настройка изображения (Вид ▸ Панели инструментов ▸ Настройка изображения). Как правило, при выборе рисунка в тексте документа эта панель открывается автоматически.

По способу взаимодействия с текстом выделяют два основных типа изображений: *внедренные в строку* (*inline*) и *свободные* (*floating*). Изображения первого типа можно условно рассматривать как отдельные символы: при движении текста в процессе редактирования изображение перемещается вместе с ним и остается в том месте текста, куда его поместили. Положение свободного изображения на странице не связано с позицией ввода. Изображение взаимодействует с текстом посредством обтекания.

Для управления методом взаимодействия изображения с текстом служит вкладка Положение в диалоговом окне Формат рисунка, которое открывают командой Формат ▸ Рисунок или кнопкой Формат рисунка на панели инструментов Настройка изображения. Элемент управления В тексте обеспечивает внедрение изображения в текстовую строку. Прочие элементы служат для выбора одного из методов обтекания. Если изображение вставлено в документ как свободное, дополнительные средства настройки обтекания можно получить из меню, которое открывается кнопкой Обтекание текстом на панели инструментов Настройка изображения. В частности, здесь присутствует пункт Изменить контур обтекания, который позволяет создавать интересные варианты обтекания изображения по криволинейному контуру.

Приемы редактирования изображения. В текстовом процессоре *Microsoft Word XP* имеются два средства редактирования встроенного растрового изображения.

Первое средство — внутреннее, а второе — внешнее, подключаемое при установке процессора. Внутреннее средство представлено элементами управления панели инструментов Настройка изображения (Вид ▸ Панели инструментов ▸ Настройка изображения). Внешним средством редактирования изображений является редактор *Microsoft Photo Editor* 3.0. Он должен быть подключен при установке *Microsoft Word XP* точно так же, как редактор формул *Microsoft Equation* 3.0 и редактор диаграмм и графиков *Microsoft Chart*.

Внутреннее средство редактирования изображений имеет относительно малые возможности, и, если говорить строго, его не вполне корректно считать средством редактирования изображений. При его использовании оригинал изображения не меняется, а меняется только способ его отображения в документе. Фактически здесь редактируется не изображение, а фильтр, управляющий тем, как оно выглядит в документе.

На панели инструментов Настройка изображения средства настройки изображения представлены следующими кнопками:

- Увеличить контрастность;
- Уменьшить контрастность;
- Увеличить яркость;
- Уменьшить яркость;
- Обрезка;
- Установить прозрачный цвет.

Функция установки прозрачного цвета имеет особое значение для создания *Web*-страниц. Она позволяет назначить один (любой) из цветов изображения в качестве «прозрачного». При размещении такого графического объекта поверх других объектов (это выполняется настройкой метода обтекания) все объекты нижележащего слоя видны через те участки верхнего изображения, которые имеют цвет, назначенный прозрачным. Разумеется, изображения, используемые для такого представления, надо готовить особо. Они должны иметь большие участки, окрашенные однородным фоновым цветом. Для этого изображение либо предварительно обрабатывают в графическом редакторе, либо сразу снимают цифровой фотокамерой на однородном фоне (как правило, синего цвета).

Внешнее средство редактирования изображений (редактор *Microsoft Photo Editor 3.0*) рассчитано на изменение файла оригинала и потому применимо только к изображениям, внедренным в документ, но не связанным. Более того, вставку изображения в документ в этом случае надо выполнять не как обычно (Вставка ▸ Рисунок ▸ Из файла), а другим способом — Вставка ▸ Объект ▸ Microsoft Photo Editor 3.0 Photo. При этом открывается окно создания нового изображения Создание рисунка, в котором следует включить переключатель Открыть имеющийся.

Заранее подготовленное изображение открывается из файла и может редактироваться средствами редактора *Microsoft Photo Editor* 3.0. По окончании редактирования окно редактора закрывают, и изображение автоматически встраивается в текстовый документ. Если в дальнейшем потребуется продолжить его редактирование,

то при двойном щелчке на объекте изображение откроется непосредственно в редакторе *Microsoft Photo Editor 3.0*.

Практическое занятие

Упражнение 11.1. Создание сложных таблиц методом рисования

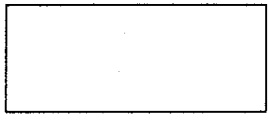
На рис. 11.15 представлен фрагмент технологической карты механической обработки детали. По своей сути технологическая карта является табличной формой сложной структуры. В данном упражнении мы рассмотрим процесс ее создания средствами текстового процессора *Microsoft Word*.

Переход	Содержание перехода	Инструмент (код и наименование)			Режим обработки					T_o	$T_в$
		вспомогательный	режущий	измерительный	T	i	S	n	V		
А											
1											
2											
3											

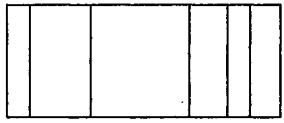
Рис. 11.15. Фрагмент карты механической обработки детали

1. Запустите текстовый процессор.
2. Создайте новый документ на базе обычного шаблона.
3. В качестве режима представления документа включите Режим разметки (Вид ▸ Разметка страницы), чтобы четко видеть границы полосы набора.
4. Откройте панель инструментов Таблицы и границы (Вид ▸ Панели инструментов ▸ Таблицы и границы).
5. Выберите инструмент Нарисовать таблицу.
6. Методом протягивания нарисуйте с его помощью прямоугольник, ширина которого равна ширине полосы набора. Высота прямоугольника может быть произвольной — его можно будет растянуть или сжать впоследствии. Для этого достаточно навести указатель мыши на нижнюю границу рамки и, когда указатель сменит форму, переместить рамку методом перетаскивания.

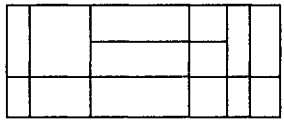
   Полученный прямоугольник представляет собой внешнюю границу таблицы. Для прочих границ она будет *опорной*, то есть они должны начинаться и заканчиваться на опорной границе.



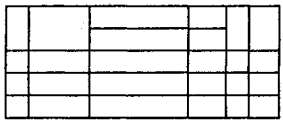
7. Проведите пять вертикальных линий. Это внутренние границы. Они опираются на внешние границы. Для горизонтальных границ, которые будут на них опираться, они будут выполнять функции опорных. На ширину столбцов не обращайте внимания — ее можно будет изменить впоследствии. Сейчас мы разрабатываем только структуру таблицы.



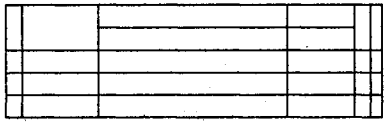
8. Убедитесь, что с помощью инструмента Ластик можно удалить любую из только что проведенных границ. Удаление выполняется одним щелчком. Внешние границы удалить нельзя.
9. Проведите две горизонтальные линии, как показано на рисунке.



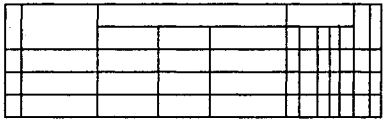
10. Убедитесь с помощью Ластика в том, что вертикальные линии, ставшие опорными для первой горизонтальной линии, не могут быть удалены.
11. Выделите всю таблицу. Для этого введите в нее указатель мыши и дайте команду Таблица ▸ Выделить ▸ Таблица.
12. Когда таблица выделена, можно задать высоту ее строк элементом управления Таблица ▸ Свойства таблицы ▸ Строка ▸ Высота. Добавьте в нижней части таблицы несколько строк командой Таблица ▸ Вставить ▸ Строки ниже. При необходимости впоследствии можно добавить столько строк, сколько надо.



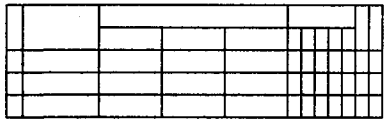
13. Методом перетаскивания вертикальных границ создайте нужное соотношение между шириной столбцов.



14. Проведите дополнительные вертикальные линии инструментом Нарисовать таблицу.



15. Выделите группы столбцов, которые должны иметь равную ширину. Для этого установите указатель мыши над верхней рамкой таблицы и в тот момент, когда он примет форму стрелки, направленной вниз, щелкните левой кнопкой.
16. Выделенные столбцы станут равными по ширине, если щелкнуть на кнопке Выровнять ширину столбцов на панели инструментов Таблицы и границы.



17. Если необходимо выровнять высоту строк, их следует выделить и использовать кнопку Выровнять высоту строк.
18. Заполните заголовки столбцов таблицы. Гарнитуру шрифта, его размер и начертание задайте с помощью инструментов панели Форматирование.
19. Обратите внимание на то, что в ячейках таблицы имеет значение не только горизонтальное выравнивание, но и вертикальное, поэтому для задания выравнивания заголовков средств панели Форматирование недостаточно. Нужный метод выравнивания (один из девяти) выбирают в палитре, которая открывается щелчком на раскрывающей кнопке Выравнивание в ячейке в центре панели Таблицы и границы.
20. При вводе заголовка первого столбца в образце использовано вертикальное расположение текста. Это типичный прием для оформления заголовков узких столбцов. Изменение направления текста выполняют с помощью кнопки Изменить направление текста на панели инструментов Таблицы и границы.
21. Завершив создание таблицы, сохраните документ *Word* в папке \Мои документы.

Мы научились создавать таблицы сложной структуры методом «рисования» и использовать автоматические средства управления шириной столбцов, высотой ячеек и их выравниванием.

Упражнение 11.2. Создание диаграмм на основе таблиц

Ниже представлена таблица с итогами испытания на износ образцов легированных сталей при трении скольжения под нагрузкой в условиях недостаточной смазки. Замеры величины износа образца производились восемь раз через каждые пятнадцать минут.

Пара трения	Износ верхнего образца, мг							
	15 мин	30 мин	45 мин	60 мин	75 мин	90 мин	105 мин	120 мин
40Х13/95Х18	11,2	7,6	4,2	1,8	1,1	1,2	1,1	1,2
40Х13/40ХН	17,4	12,5	9,5	7,4	5,3	4,8	4,5	4,4
40ХН/95Х18	12,1	6,4	3,1	2,2	1,7	1,6	1,6	1,6

В этом упражнении мы построим диаграмму на базе данной таблицы.

1. Запустите текстовый процессор.
2. Создайте новый документ на базе стандартного шаблона.
3. В качестве режима представления документа включите Режим разметки (Вид ▸ Разметка страницы), чтобы четко видеть границы полосы набора.
4. Командой Таблица ▸ Вставить ▸ Таблица создайте базовую таблицу, имеющую 5 строк и 9 столбцов.
5. Выделите две верхние ячейки первого столбца и объедините их командой Таблица ▸ Объединить ячейки.
6. Выделите ячейки первой строки для столбцов со второго по девятый и объедините их.
7. Заполните таблицу согласно прилагаемому образцу.
8. Установите указатель мыши в поле таблицы и выделите таблицу командой Таблица ▸ Выделить ▸ Таблица. Скопируйте выделенную таблицу в буфер обмена (Правка ** Копировать).
9. Вставьте базовую диаграмму командой Вставка ▸ Объект ▸ Microsoft Graph Chart. Рядом с диаграммой развернется ее базовая таблица.
10. Выделите содержимое базовой таблицы диаграммы щелчком на ячейке, образованной на пересечении заголовков строк и столбцов в левом верхнем углу.
11. Замените содержимое базовой таблицы содержимым своей таблицы командой вставки содержимого из буфера обмена (Правка ▸ Вставить).
12. Обратите внимание на то, как изменилась диаграмма: она пришла в соответствие с содержимым таблицы.
13. На диаграмме выделите область построения. Щелкните правой кнопкой мыши и в контекстном меню выберите пункт Тип диаграммы. Средствами открывшегося диалогового окна проверьте, как выглядят диаграммы других (стандартных и нестандартных) типов.
14. Закройте диалоговое окно Тип диаграммы. Сохраните документ *Word* в папке \Мои документы.

В этом упражнении мы освоили один из двух основных методов создания диаграмм — метод, основанный на использовании базовой таблицы, которая содержащится в документе.

Упражнение 11.3. Изучение эффективных приемов работы с графическими объектами

1. Запустите текстовый процессор.
2. Создайте новый документ на базе стандартного шаблона.
3. В качестве режима представления документа включите Режим разметки (Вид ▸ Разметка страницы), чтобы четко видеть границы полосы набора.
4. Введите несколько строк произвольного текста.

5. Командой Вставка ▸ Рисунок ▸ Из файла вставьте ниже текста рисунок из произвольного файла, например из файла \Windows\Японский мотив.bmp.
6. Выделите рисунок щелчком левой кнопки мыши — откроется панель инструментов Настройка изображения. Используя кнопку Формат рисунка, откройте одноименное диалоговое окно.
7. На вкладке Положение выберите вариант размещения В тексте. Передвиньте изображение методом перетаскивания, оценивая происходящее взаимодействие с текстом.
8. На вкладке Положение диалогового окна Формат рисунка выберите вариант размещения По контуру. Проверьте, как происходит взаимодействие с текстом при перемещении изображения.
9. Выделите изображение, скопируйте его в буфер обмена (CTRL+C) и создайте рядом его копию (CTRL+V).
10. Перемещая оба изображения, добейтесь их положения рядом, с выравниванием по верхнему краю.
11. Повторите перемещение изображений с выравниванием при нажатой клавише ALT. Убедитесь в том, что перемещение изображений происходит дискретно, с привязкой к узлам невидимой сетки, что позволяет выполнить выравнивание абсолютно точно.
12. Выделите одно из изображений. Используя угловой маркер, измените его размер методом перетаскивания.
13. Восстановите прежний размер изображения.
14. Повторите перетаскивание углового маркера, но при нажатой клавише CTRL. Обратите внимание на то, что характер изменения размера изображения изменился. В данном случае оно перемасштабируется «от центра».
15. Сохраните итоговый документ *Word* в папке \Мои документы.

Мы освоили два основных приема вставки изображения в текст — с внедрением в строку и со свободным размещением. Мы убедились, что использование клавиш CTRL и ALT при работе с изображениями в документе открывает дополнительные возможности оформления.

Упражнение 11.4. Создание графических заголовков

15 мин

Для создания художественных графических надписей, например заголовков, текстовый процессор *Microsoft Word XP* имеет специальное программное средство *WordArt*. Доступ к нему осуществляется двумя способами: либо через панель инструментов WordArt (Вид ▸ Панели инструментов ▸ WordArt), либо с помощью кнопки Добавить объект WordArt на панели инструментов Рисование.

Графические объекты, вставленные в текстовый документ средством *WordArt*, могут распечатываться вместе с документом на выводном печатающем устройстве, могут отображаться в составе электронного документа, распространяемого в формате *Microsoft Word*, и могут отображаться на *Web*-страницах. Однако при экспорте доку-

мента в форматы других программ, предназначенных для обработки документов, объекты *WordArt* не всегда воспроизводятся правильно, то есть при создании документов, в которых содержание играет более высокую роль, чем оформление, использовать художественные заголовки, выполненные средствами *WordArt*, не рекомендуется.

1. Запустите текстовый процессор.
2. Создайте новый документ на базе стандартного шаблона.
3. В качестве режима представления документа включите Режим разметки (Вид ▸ Разметка страницы), чтобы четко видеть границы полосы набора.
4. Введите несколько строк произвольного текста.
5. Командой Вид ▸ Панели инструментов ▸ WordArt включите отображение панели инструментов WordArt.
6. Щелкните на кнопке Добавить объект WordArt — произойдет запуск мастера создания объекта *WordArt*.
7. В окне Коллекция WordArt выберите желаемый стиль оформления надписи.
8. В диалоговом окне Изменение текста WordArt выберите желаемый шрифт, его размер, начертание и введите текст создаваемого заголовка (надписи).
9. После щелчка на кнопке OK произойдет вставка созданного объекта в текущий документ *Microsoft Word*.
10. Дальнейшее управление формой и расположением созданного объекта выполняют элементами управления панели инструментов WordArt. Проверьте, как протекают следующие операции (после каждой команды возвращайтесь к исходному состоянию комбинацией CTRL+Z):
    - изменение содержания надписи (Изменить текст);
    - изменение стиля оформления (Коллекция WordArt);
    - изменение характера взаимодействия с основным текстом (Формат объекта ▸ Положение);
    - изменение формы надписи (Форма WordArt);
    - выравнивание букв надписи по высоте (Выровнять буквы WordArt по высоте);
    - расположение текста надписи по вертикали (Вертикальный текст WordArt);
    - управление интервалом между символами (Межсимвольный интервал WordArt).
11. Закончив эксперименты, создайте заголовок по своему вкусу и сохраните документ *Word* в папке \Мои документы.

Мы научились создавать художественные заголовки, внедрять их в документы и редактировать «по месту». В то же время мы узнали, что для документов, передаваемых на последующую обработку, пользоваться этим средством не рекомендуется.

Глава 12. Обработка данных средствами электронных таблиц

Для представления данных в удобном виде используют таблицы. Компьютер позволяет представлять их в электронной форме, а это дает возможность не только отображать, но и обрабатывать данные. Класс программ, используемых для этой цели, называется *электронными таблицами*.

Особенность электронных таблиц заключается в возможности применения формул для описания связи между значениями различных ячеек. Расчет по заданным формулам выполняется автоматически. Изменение содержимого какой-либо ячейки приводит к пересчету значений всех ячеек, которые с ней связаны формульными отношениями и, тем самым, к обновлению всей таблицы в соответствии с изменившимися данными.

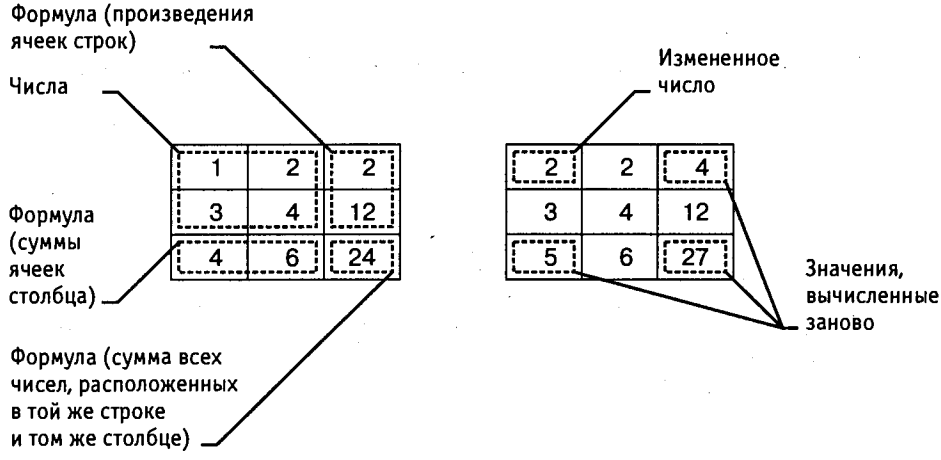


Рис. 12.1. При изменении содержания одной из ячеек таблицы все формулы пересчитываются и значения в ячейках, которые прямо или косвенно зависят от измененных, автоматически обновляются

Применение электронных таблиц упрощает работу с данными и позволяет получать результаты без проведения расчетов вручную или специального программирования. Наиболее широкое применение электронные таблицы нашли в экономических и бухгалтерских расчетах, но и в научно-технических задачах электронные таблицы можно использовать эффективно, например для:

- проведения однотипных расчетов над большими наборами данных;
- автоматизации итоговых вычислений;
- решения задач путем подбора значений параметров, табулирования формул;
- обработки результатов экспериментов;
- проведения поиска оптимальных значений параметров;
- подготовки табличных документов;
- построения диаграмм и графиков по имеющимся данным.

Одним из наиболее распространенных средств работы с документами, имеющими табличную структуру, является программа *Microsoft Excel.*

12.1. Основные понятия электронных таблиц

Программа *Microsoft Excel* предназначена для работы с таблицами данных, преимущественно числовых. При формировании таблицы выполняют ввод, редактирование и форматирование текстовых и числовых данных, а также формул. Наличие средств автоматизации облегчает эти операции. Созданная таблица может быть выведена на печать.

Рабочая книга и рабочий лист. Строки, столбцы, ячейки

Документ *Excel* называется *рабочей книгой.* Рабочая книга представляет собой набор *рабочих листов,* каждый из которых имеет табличную структуру и может содержать одну или несколько таблиц. В окне документа в программе *Excel* отображается только *текущий* рабочий лист, с которым и ведется работа (рис. 12.2). Каждый рабочий лист имеет *название,* которое отображается на ярлычке *листа,* отображаемом в его нижней части. С помощью ярлычков можно переключаться к другим рабочим листам, входящим в ту же самую рабочую книгу. Чтобы переименовать рабочий лист, надо дважды щелкнуть на его ярлычке.

Рабочий лист состоит из *строк* и *столбцов.* Столбцы озаглавлены прописными латинскими буквами и, далее, двухбуквенными комбинациями. Всего рабочий лист может содержать до 256 столбцов, пронумерованных от A до IV. Строки последовательно нумеруются цифрами, от 1 до 65 536 (максимально допустимый номер строки).

Ячейки и их адресация. На пересечении столбцов и строк образуются *ячейки* таблицы. Они являются минимальными элементами для хранения данных. Обозначение отдельной ячейки сочетает в себе номера столбца и строки (в этом порядке), на пересечении которых она расположена, например: A1 или DE234. Обозначение ячейки (ее номер) выполняет функции ее адреса. Адреса ячеек используются при

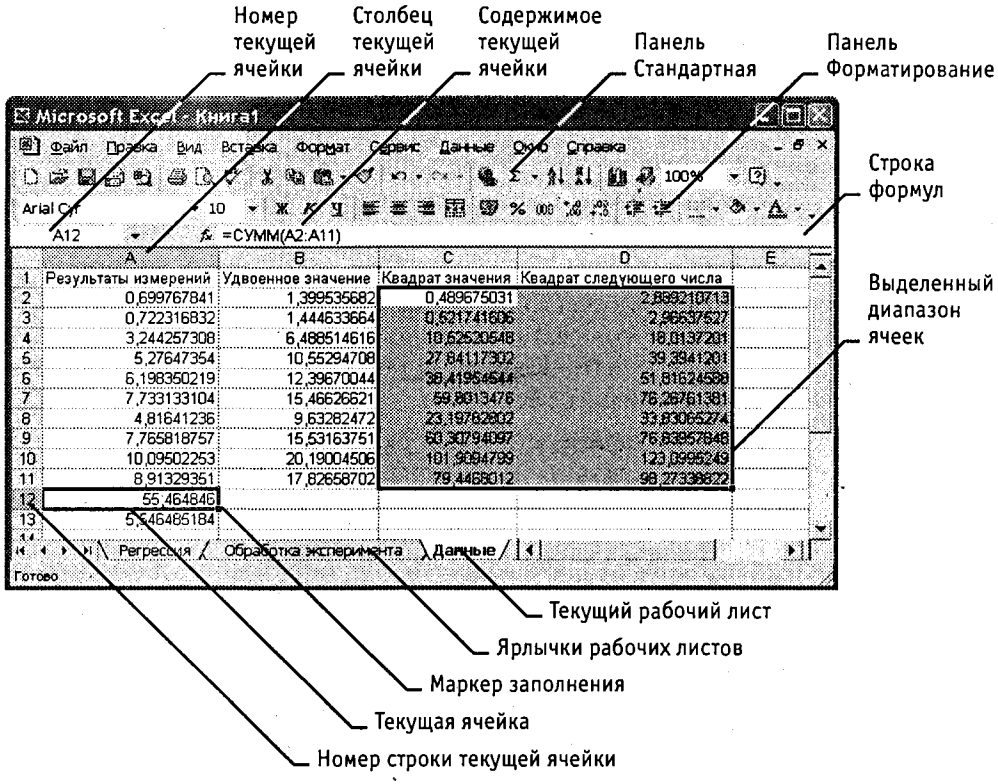


Рис. 12.2. Рабочий лист электронной таблицы Excel

записи формул, определяющих взаимосвязь между значениями, расположенными в разных ячейках.

Одна из ячеек всегда является *активной* и выделяется *рамкой активной ячейки*. Эта рамка в программе *Excel* играет роль курсора. Операции ввода и редактирования всегда производятся в активной ячейке. Переместить рамку активной ячейки можно с помощью курсорных клавиш или указателя мыши.

Диапазон ячеек. На данные, расположенные в соседних ячейках, можно ссылаться в формулах как на единое целое. Такую группу ячеек называют *диапазоном*. Наиболее часто используют прямоугольные диапазоны, образующиеся на пересечении группы последовательно идущих строк и группы последовательно идущих столбцов. Диапазон ячеек обозначают, указывая через двоеточие номера ячеек, расположенных в противоположных углах прямоугольника, например: A1:C15.

Если требуется выделить прямоугольный диапазон ячеек, это можно сделать протягиванием указателя от одной угловой ячейки до противоположной по диагонали. Рамка текущей ячейки при этом расширяется, охватывая весь выбранный диапазон. Чтобы выбрать столбец или строку целиком, следует щелкнуть на заголовке

столбца (строки). Протягиванием указателя по заголовкам можно выбрать несколько идущих подряд столбцов или строк.

Ввод, редактирование и форматирование данных

Отдельная ячейка может содержать данные, относящиеся к одному из трех типов: *текст*, *число* или *формула*, — а также оставаться пустой. Программа *Excel* при сохранении рабочей книги записывает в файл только прямоугольную область рабочих листов, примыкающую к левому верхнему углу (ячейка A1) и содержащую все заполненные ячейки.

Тип данных, размещаемых в ячейке, определяется автоматически при вводе. Если эти данные можно интерпретировать как число, программа *Excel* так и делает. В противном случае данные рассматриваются как текст. Ввод формулы всегда начинается с символа «=» (знака равенства).

Ввод текста и чисел. Ввод данных осуществляют непосредственно в текущую ячейку или в *строку формул*, располагающуюся в верхней части окна программы под панелями инструментов (см. рис. 12.2). Место ввода отмечается текстовым курсором. Если начать ввод нажатием алфавитно-цифровых клавиш, данные из текущей ячейки заменяются вводимым текстом. Если щелкнуть на строке формул или дважды на текущей ячейке, старое содержимое ячейки не удаляется и появляется возможность его редактирования. Вводимые данные в любом случае отображаются как в ячейке, так и в строке формул.

Чтобы завершить ввод, сохранив введенные данные, используют кнопку Ввод в строке формул или клавишу ENTER. Чтобы отменить внесенные изменения и восстановить прежнее значение ячейки, используют кнопку Отмена в строке формул или клавишу ESC. Для очистки текущей ячейки или выделенного диапазона проще всего использовать клавишу DELETE.

Форматирование содержимого ячеек. Текстовые данные по умолчанию выравниваются по левому краю ячейки, а числа — по правому. Чтобы изменить формат отображения данных в текущей ячейке или выбранном диапазоне, используют команду Формат ▸ Ячейки. Вкладки этого диалогового окна позволяют выбирать формат записи данных (количество знаков после запятой, указание денежной единицы, способ записи даты и прочее), задавать направление текста и метод его выравнивания, определять шрифт и начертание символов, управлять отображением и видом рамок, задавать фоновый цвет.

12.2. Содержание электронной таблицы

Формулы

Вычисления в таблицах программы *Excel* осуществляются при помощи *формул*. Формула может содержать числовые константы, ссылки на ячейки и *функции Excel*, соединенные знаками математических операций. Скобки позволяют изменять стандартный порядок выполнения действий. Если ячейка содержит формулу, то в рабочем листе отображается текущий результат вычисления этой формулы. Если сделать ячейку текущей, то сама формула отображается в строке формул.

Правило использования формул в программе *Excel* состоит в том, что, если значение ячейки *действительно* зависит от других ячеек таблицы, *всегда* следует использовать формулу, даже если операцию легко можно выполнить в «уме». Это гарантирует, что последующее редактирование таблицы не нарушит ее целостности и правильности производимых в ней вычислений.

Ссылки на ячейки

Формула может содержать *ссылки*, то есть адреса ячеек, содержимое которых используется в вычислениях. Это означает, что результат вычисления формулы зависит от числа, находящегося в другой ячейке. Ячейка, содержащая формулу, таким образом, является *зависимой*. Значение, отображаемое в ячейке с формулой, пересчитывается при изменении значения ячейки, на которую указывает ссылка.

Ссылку на ячейку можно задать разными способами. Во-первых, адрес ячейки можно ввести вручную. Другой способ состоит в щелчке на нужной ячейке или выборе диапазона, адрес которого требуется ввести. Ячейка или диапазон при этом выделяются пунктирной рамкой.

Все диалоговые окна программы *Excel*, которые требуют указания номеров или диапазонов ячеек, содержат кнопки, присоединенные к соответствующим полям. При щелчке на такой кнопке диалоговое окно сворачивается до минимально возможного размера, что облегчает выбор нужной ячейки (диапазона) с помощью щелчка или протягивания (рис. 12.3).

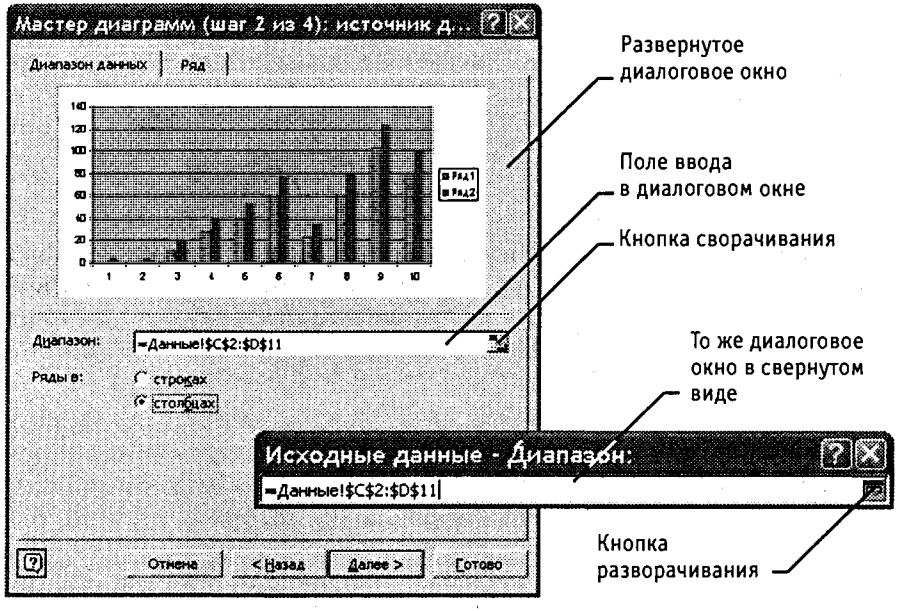


Рис. 12.3. Диалоговое окно в развернутом и свернутом виде

Для редактирования формулы следует дважды щелкнуть на соответствующей ячейке. При этом ячейки (диапазоны), от которых зависит значение формулы, выде-

ляются на рабочем листе цветными рамками, а сами ссылки отображаются в ячейке и в строке формул тем же цветом. Это облегчает редактирование и проверку правильности формул.

Абсолютные и относительные ссылки

По умолчанию, ссылки на ячейки в формулах рассматриваются как *относительные*. Это означает, что при копировании формулы адреса в ссылках автоматически изменяются в соответствии с относительным расположением исходной ячейки и создаваемой копии.

Пусть, например, в ячейке B2 имеется ссылка на ячейку A3. В относительном представлении можно сказать, что ссылка указывает на ячейку, которая располагается на один столбец левее и на одну строку ниже данной. Если формула будет скопирована в другую ячейку, то такое относительное указание ссылки сохранится. Например, при копировании формулы в ячейку EA27 ссылка будет продолжать указывать на ячейку, располагающуюся левее и ниже, в данном случае на ячейку DZ28.

При *абсолютной адресации* адреса ссылок при копировании не изменяются, так что ячейка, на которую указывает ссылка, рассматривается как нетабличная. Для изменения способа адресации при редактировании формулы надо выделить ссылку на ячейку и нажать клавишу F4. Элементы номера ячейки, использующие абсолютную адресацию, предваряются символом $. Например, при последовательных нажатиях клавиши F4 номер ячейки A1 будет записываться как A1, $A$1, A$1 и $A1. В двух последних случаях один из компонентов номера ячейки рассматривается как абсолютный, а другой — как относительный.

Копирование содержимого ячеек

Копирование и перемещение ячеек в программе *Excel* можно осуществлять методом перетаскивания или через буфер обмена. При работе с небольшим числом ячеек удобно использовать первый метод, при работе с большими диапазонами — второй.

Метод перетаскивания. Чтобы методом перетаскивания скопировать или переместить текущую ячейку (выделенный диапазон) вместе с содержимым, следует навести указатель мыши на рамку текущей ячейки (он примет вид стрелки с дополнительными стрелочками). Теперь ячейку можно перетащить в любое место рабочего листа (точка вставки помечается всплывающей подсказкой).

Для выбора способа выполнения этой операции, а также для более надежного контроля над ней рекомендуется использовать *специальное перетаскивание* с помощью правой кнопки мыши. В этом случае при отпускании кнопки мыши появляется специальное меню, в котором можно выбрать конкретную выполняемую операцию.

Применение буфера обмена. Передача информации через буфер обмена имеет в программе *Excel* определенные особенности, связанные со сложностью контроля над этой операцией. Вначале необходимо выделить копируемый (вырезаемый) диапазон и дать команду на его помещение в буфер обмена: Правка ▸ Копировать или Правка ▸ Вырезать. Вставка данных в рабочий лист возможна лишь немедленно после их помещения в буфер обмена. Попытка выполнить любую другую опера-

цию приводит к отмене начатого процесса копирования или перемещения. Однако утраты данных не происходит, поскольку «вырезанные» данные удаляются из места их исходного размещения только в момент выполнения вставки.

Место вставки определяется путем указания ячейки, соответствующей верхнему левому углу диапазона, помещенного в буфер обмена, или путем выделения диапазона, который по размерам в точности равен копируемому (перемещаемому). Вставка выполняется командой Правка ▸ Вставить. Для управления способом вставки можно использовать команду Правка ▸ Специальная вставка. В этом случае правила вставки данных из буфера обмена задаются в открывшемся диалоговом окне.

Автоматизация ввода

Так как таблицы часто содержат повторяющиеся или однотипные данные, программа *Excel* содержит средства автоматизации ввода. К числу предоставляемых средств относятся: *автозавершение*, *автозаполнение числами* и *автозаполнение формулами*.

Автозавершение. Для автоматизации ввода текстовых данных используется метод *автозавершения*. Его применяют при вводе в ячейки одного столбца рабочего листа текстовых строк, среди которых есть повторяющиеся. В ходе ввода текстовых данных в очередную ячейку программа *Excel* проверяет соответствие введенных символов строкам, имеющемся в этом столбце выше. Если обнаружено однозначное совпадение, введенный текст автоматически дополняется. Нажатие клавиши ENTER подтверждает операцию автозавершения, в противном случае ввод можно продолжать, не обращая внимания на предлагаемый вариант.

Можно прервать работу средства автозавершения, оставив в столбце пустую ячейку. И наоборот, чтобы использовать возможности средства автозавершения, заполненные ячейки должны идти подряд, без промежутков между ними.

Автозаполнение числами. При работе с числами используется метод *автозаполнения*. В правом нижнем углу рамки текущей ячейки имеется черный квадратик — *маркер заполнения*. При наведении на него указатель мыши (он обычно имеет вид толстого белого креста) приобретает форму тонкого черного крестика. Перетаскивание маркера заполнения рассматривается как операция «размножения» содержимого ячейки в горизонтальном или вертикальном направлении.

Если ячейка содержит число (в том числе дату, денежную сумму), то при перетаскивании маркера происходит копирование ячеек или их заполнение арифметической прогрессией. Для выбора способа автозаполнения следует производить специальное перетаскивание с использованием правой кнопки мыши.

Пусть, например, ячейка A1 содержит число 1. Наведите указатель мыши на маркер заполнения, нажмите правую кнопку мыши и перетащите маркер заполнения так, чтобы рамка охватила ячейки A1, B1 и C1, и отпустите кнопку мыши. Если теперь выбрать в открывшемся меню пункт Копировать ячейки, все ячейки будут содержать число 1. Если же выбрать пункт Заполнить, то в ячейках окажутся числа 1, 2 и 3.

Чтобы точно сформулировать условия заполнения ячеек, следует дать команду Правка ▸ Заполнить ▸ Прогрессия. В открывшемся диалоговом окне Прогрессия выбирается тип прогрессии, величина шага и предельное значение. После щелчка

на кнопке OK программа *Excel* автоматически заполняет ячейки в соответствии с заданными правилами.

Автозаполнение формулами. Эта операция выполняется так же, как автозаполнение числами. Ее особенность заключается в необходимости копирования ссылок на другие ячейки. В ходе автозаполнения во внимание принимается характер ссылок в формуле: относительные ссылки изменяются в соответствии с относительным расположением копии и оригинала, абсолютные остаются без изменений.

Для примера предположим, что значения в третьем столбце рабочего листа (столбце C) вычисляются как суммы значений в соответствующих ячейках столбцов A и B. Введем в ячейку C1 формулу =A1+B1. Теперь скопируем эту формулу методом автозаполнения во все ячейки третьего столбца таблицы. Благодаря относительной адресации формула будет правильной для всех ячеек данного столбца.

В таблице 12.1 приведены правила обновления ссылок при автозаполнении вдоль строки или вдоль столбца.

Таблица 12.1. Правила обновления ссылок при автозаполнении

Ссылка в исходной ячейке	Ссылка в следующей ячейке	
	При заполнении вправо	При заполнении вниз
A1 (относительная)	B1	A2
$A1 (абсолютная по столбцу)	$A1	$A2
A$1 (абсолютная по строке)	B$1	A$1
$A$1 (абсолютная)	$A$1	$A$1

Использование стандартных функций

Стандартные функции используются в программе *Excel* только в формулах. *Вызов функции* состоит в указании в формуле *имени функции*, после которого в скобках указывается *список параметров*. Отдельные параметры разделяются в списке точкой с запятой. В качестве параметра может использоваться число, адрес ячейки или произвольное выражение, для вычисления которого также могут использоваться функции.

В режиме ввода формулы в левой части строки формул, где раньше располагался номер текущей ячейки, появляется раскрывающийся список функций. Он содержит десять функций, которые использовались последними, а также пункт Другие функции.

Использование мастера функций. При выборе пункта Другие функции запускается Мастер функций, облегчающий выбор нужной функции. В раскрывающемся списке Категория выбирается категория, к которой относится функция (если определить категорию затруднительно, используют пункт Полный алфавитный перечень), а в списке Выберите функцию — конкретная функция данной категории. После щелчка на кнопке OK имя функции заносится в строку формул вместе со скобками, ограничивающими список параметров. Текстовый курсор устанавливается между этими скобками. Вызвать Мастер функций можно и проще, щелчком на кнопке Вставка функции в строке формул.

Аргументы функции. Как только имя функции выбрано, на экране появляется диалоговое окно Аргументы функции (в предыдущих версиях *Excel* это окно рассматривалось как *палитра формул*). Это окно, в частности, содержит значение, которое получится, если немедленно закончить ввод формулы (рис. 12.4).

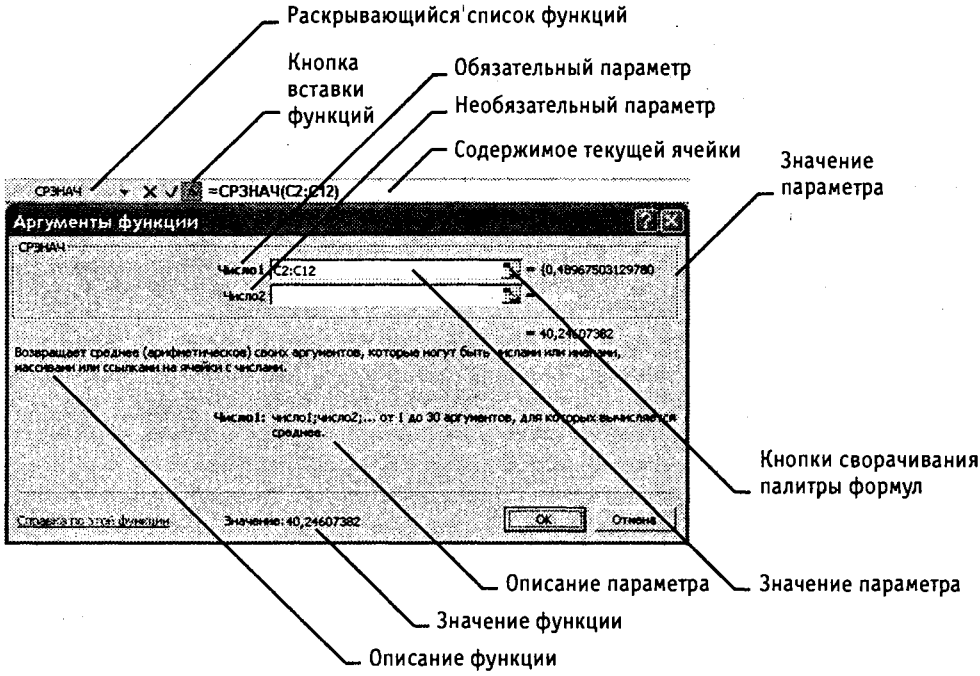


Рис. 12.4. Строка формул и диалоговое окно Аргументы функции

Правила вычисления формул, содержащих функции, не отличаются от правил вычисления более простых формул. Ссылки на ячейки, используемые в качестве параметров функции, также могут быть относительными или абсолютными, что учитывается при копировании формул методом автозаполнения.

12.3. Печать документов Excel

Экранное представление электронной таблицы в *Excel* значительно отличается от того, которое получилось бы при выводе данных на печать. Это связано с тем, что единый рабочий лист приходится разбивать на фрагменты, размер которых определяется форматом печатного листа. Кроме того, элементы оформления рабочего окна программы: номера строк и столбцов, условные границы ячеек — обычно не отображаются при печати.

Предварительный просмотр

Перед печатью рабочего листа следует перейти в режим *предварительного просмотра* (кнопка Предварительный просмотр на стандартной панели инструментов). Режим предварительного просмотра (рис. 12.5) не допускает редактирования документа,

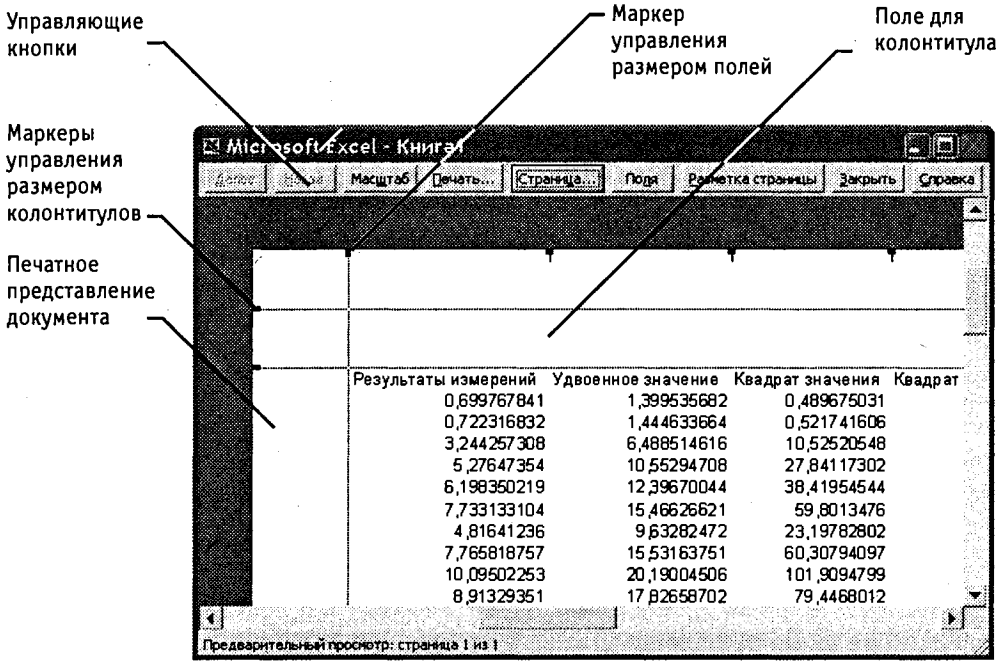


Рис. 12.5. Предварительный просмотр документа перед печатью

но позволяет увидеть его на экране точно в таком виде, в каком он будет напечатан. Кроме того, режим предварительного просмотра позволяет изменить свойства печатной страницы и параметры печати.

Управление в режиме предварительного просмотра осуществляется при помощи кнопок, расположенных вдоль верхнего края окна. Кнопка Страница открывает диалоговое окно Параметры страницы, которое служит для задания параметров страницы: ориентации листа, масштаба страницы (изменение масштаба позволяет управлять числом печатных страниц, необходимых для документа), размеров полей документа. Здесь же можно задать верхние и нижние колонтитулы для страницы. На вкладке Лист включается или отключается печать сетки и номеров строк и столбцов, а также выбирается последовательность разбиения на страницы рабочего листа, превосходящего размеры печатной страницы как по длине, так и по ширине.

Изменить величину полей страницы, а также ширину ячеек при печати можно также непосредственно в режиме предварительного просмотра, при помощи кнопки Поля. При щелчке на этой кнопке на странице появляются маркеры, указывающие границы полей страницы и ячеек. Изменить положение этих границ можно методом перетаскивания.

Завершить работу в режиме предварительного просмотра можно тремя способами, в зависимости от того, что планируется делать дальше. Щелчок на кнопке Закрыть позволяет вернуться к редактированию документа. Щелчок на кнопке Разметка

страницы служит для возврата к редактированию документа, но в режиме *разметки страницы*. В этом режиме документ отображается таким образом, чтобы наиболее удобно показать не содержимое ячеек таблицы, а *область печати* и границы страниц документа. Переключение между режимом разметки и обычным режимом можно также осуществлять через меню Вид (команды Вид ▸ Обычный и Вид ▸ Разметка страницы). Третий способ — начать печать документа.

Печать документа

Щелчок на кнопке Печать открывает диалоговое окно Печать, используемое для распечатки документа (его можно открыть и без предварительного просмотра — с помощью команды Файл ▸ Печать). Это окно содержит стандартные средства управления, применяемые для печати документов в любых приложениях.

Выбор области печати

Область печати — эта часть рабочего листа, которая должна быть выведена на печать. По умолчанию область печати совпадает с заполненной частью рабочего листа и представляет собой прямоугольник, примыкающий к верхнему левому углу рабочего листа и захватывающий все заполненные ячейки. Если часть данных не должна выводиться на бумагу, область печати можно задать вручную. Для этого надо выделить ячейки, которые должны быть включены в область печати, и дать команду Файл ▸ Область печати ▸ Задать. Если текущей является одна-единственная ячейка, то программа предполагает, что область печати не выделена, и выдает предупреждающее сообщение.

Если область печати задана, то программа отображает в режиме предварительного просмотра и распечатывает только ее. Границы области печати выделяются на рабочем листе крупным пунктиром (сплошной линией в режиме разметки). Для изменения области печати можно задать новую область или при помощи команды Файл ▸ Область печати ▸ Убрать вернуться к параметрам, используемым по умолчанию.

Границы отдельных печатных страниц отображаются на рабочем листе мелким пунктиром. В некоторых случаях требуется, чтобы определенные ячейки располагались вместе на одной и той же печатной странице или, наоборот, разделение печатных страниц происходило в определенном месте рабочего листа. Такая возможность реализуется путем задания границ печатных страниц вручную. Чтобы вставить разрыв страницы, надо сделать текущей ячейку, которая будет располагаться в левом верхнем углу печатной страницы, и дать команду Вставка ▸ Разрыв страницы. Программа *Excel* вставит принудительные разрывы страницы перед строкой и столбцом, в которых располагается данная ячейка. Если выбранная ячейка находится в первой строке или столбце A, то разрыв страницы задается только по одному направлению.

12.4. Применение электронных таблиц для расчетов

В научно-технической деятельности программу *Excel* трудно рассматривать как основной вычислительный инструмент. Однако ее удобно применять в тех случаях, когда требуется быстрая обработка больших объемов данных. Она полезна для выполнения таких операций, как статистическая обработка и анализ данных, реше-

ние задач оптимизации, построение диаграмм и графиков. Для такого рода задач применяют как основные средства программы *Excel*, так и дополнительные (надстройки).

Итоговые вычисления

Итоговые вычисления предполагают получение числовых характеристик, описывающих определенный набор данных в целом. Например, возможно вычисление суммы значений, входящих в набор, среднего значения и других статистических характеристик, количества или доли элементов набора, удовлетворяющих определенных условиям. Проведение итоговых вычислений в программе *Excel* выполняется при помощи встроенных функций. Особенность использования таких *итоговых функций* состоит в том, что при их задании программа пытается «угадать», в каких ячейках заключен обрабатываемый набор данных, и задать параметры функции автоматически.

В качестве параметра итоговой функции обычно задается некоторый диапазон ячеек, размер которого определяется автоматически. Выбранный диапазон рассматривается как отдельный параметр («массив»), и в вычислениях используются все ячейки, составляющие его.

Суммирование. Для итоговых вычислений применяют ограниченный набор функций, наиболее типичной из которых является функция суммирования (СУММ). Это единственная функция, для применения которой есть отдельная кнопка на стандартной панели инструментов (кнопка Автосумма). Диапазон суммирования, выбираемый автоматически, включает ячейки с данными, расположенные над текущей ячейкой (предпочтительнее) или слева от нее и образующие непрерывный блок. При неоднозначности выбора используется диапазон, непосредственно примыкающий к текущей ячейке.

Автоматический подбор диапазона не исключает возможности редактирования формулы. Можно переопределить диапазон, который был выбран автоматически, а также задать дополнительные параметры функции.

Функции для итоговых вычислений. Прочие функции для итоговых вычислений выбираются обычным образом, с помощью раскрывающегося списка в строке формул или с использованием мастера функций. Все эти функции относятся к категории Статистические. В их число входят функции ДИСП (вычисляет дисперсию), МАКС (максимальное число в диапазоне), СРЗНАЧ (среднее арифметическое значение чисел диапазона), СЧЕТ (подсчет ячеек с числами в диапазоне) и другие.

Функции, предназначенные для выполнения итоговых вычислений, часто применяют при использовании таблицы *Excel* в качестве базы данных, а именно на фоне фильтрации записей или при создании сводных таблиц.

Использование надстроек

Надстройки — это специальные средства, расширяющие возможности программы *Excel*. На практике именно надстройки делают программу *Excel* удобной для использования в научно-технической работе. Хотя эти средства считаются внешними,

дополнительными, доступ к ним осуществляется при помощи обычных команд строки меню (обычно через меню Сервис или Данные). Команда использования настройки обычно открывает специальное диалоговое окно, оформление которого не отличается от стандартных диалоговых окон программы *Excel*.

Подключить или отключить установленные надстройки можно с помощью команды Сервис ▸ Надстройки (рис. 12.6). Подключение надстроек увеличивает нагрузку на вычислительную систему, поэтому обычно рекомендуют подключать только те надстройки, которые реально используются.

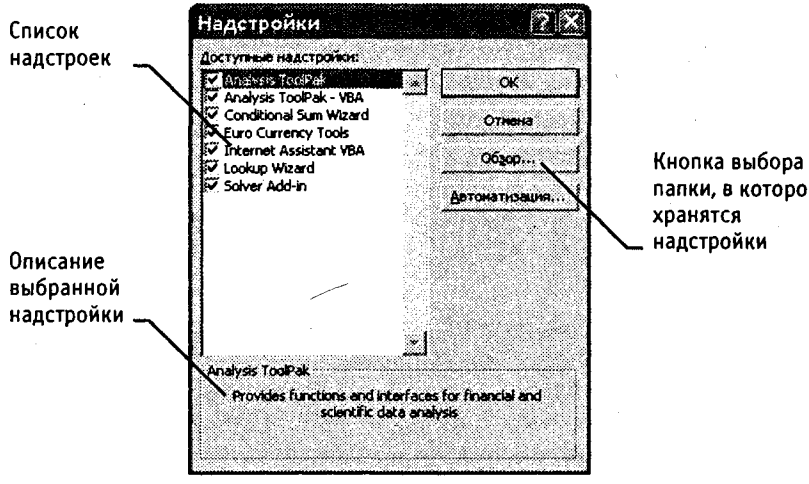


Рис. 12.6. Диалоговое окно для подключения и отключения надстроек

Вот основные надстройки, поставляемые вместе с программой *Excel*.

Пакет анализа (*Analysis ToolPak*). Обеспечивает дополнительные возможности анализа наборов данных. Выбор конкретного метода анализа осуществляется в диалоговом окне Data Analysis (Анализ данных), которое открывается командой Сервис ▸ Data Analysis (Анализ данных).

Мастер суммирования (*Conditional Sum Wizard*). Позволяет автоматизировать создание формул для суммирования данных в столбце таблицы. При этом ячейки могут включаться в сумму только при выполнении определенных условий. Запуск мастера осуществляется с помощью команды Сервис ▸ Conditional Sum (Частичная сумма).

Мастер подстановок (*Lookup Wizard*). Автоматизирует создание формулы для поиска данных в таблице по названию столбца и строки. Мастер позволяет произвести однократный поиск или предоставляет возможность ручного задания параметров, используемых для поиска. Вызывается командой Сервис ▸ Lookup (Поиск).

Поиск решения (*Solver Add-in*). Эта надстройка используется для решения задач оптимизации. Ячейки, для которых подбираются оптимальные значения и задаются ограничения, выбираются в диалоговом окне Solver Parameters (Поиск решения), которое открывают при помощи команды Сервис ▸ Solver (Поиск решения).

12.5. Построение диаграмм и графиков

В программе *Excel* термин «диаграмма» используется для обозначения всех видов графического представления числовых данных. Построение графического изображения производится на основе *ряда данных*. Так называют группу ячеек с данными в пределах отдельной строки или столбца. На одной диаграмме можно отображать несколько рядов данных.

Диаграмма представляет собой вставной объект, внедренный на один из листов рабочей книги. Она может располагаться на том же листе, на котором находятся данные, или на любом другом листе (часто для отображения диаграммы отводят отдельный лист). Диаграмма сохраняет связь с данными, на основе которых она построена, и при обновлении этих данных немедленно изменяет свой вид.

Для построения диаграммы обычно используют Мастер диаграмм, запускаемый щелчком на кнопке Мастер диаграмм на стандартной панели инструментов. Часто удобно заранее выделить область, содержащую данные, которые будут отображаться на диаграмме, но задать эту информацию можно и в ходе работы мастера.

Выбор типа диаграммы

На первом этапе работы мастера выбирают форму диаграммы. Доступные формы перечислены в списке Тип на вкладке Стандартные. Для выбранного типа диаграммы справа указывается несколько вариантов представления данных (палитра Вид), из которых следует выбрать наиболее подходящий. На вкладке Нестандартные отображается набор полностью сформированных типов диаграмм с готовым форматированием. После задания формы диаграммы следует щелкнуть на кнопке Далее.

Выбор данных

Второй этап работы мастера служит для выбора данных, по которым будет строиться диаграмма (рис. 12.7). Если диапазон данных был выбран заранее, то в области предварительного просмотра в верхней части окна мастера появится приблизительное отображение будущей диаграммы. Если данные образуют единый прямоугольный диапазон, то их удобно выбирать при помощи вкладки Диапазон данных. Если данные не образуют единой группы, то информацию для отрисовки отдельных рядов данных задают на вкладке Ряд. Предварительное представление диаграммы автоматически обновляется при изменении набора отображаемых данных.

Оформление диаграммы

Третий этап работы мастера (после щелчка на кнопке Далее) состоит в выборе оформления диаграммы. На вкладках окна мастера задаются:

- название диаграммы, подписи осей (вкладка Заголовки);
- отображение и маркировка осей координат (вкладка Оси);
- отображение сетки линий, параллельных осям координат (вкладка Линии сетки);
- описание построенных графиков (вкладка Легенда);

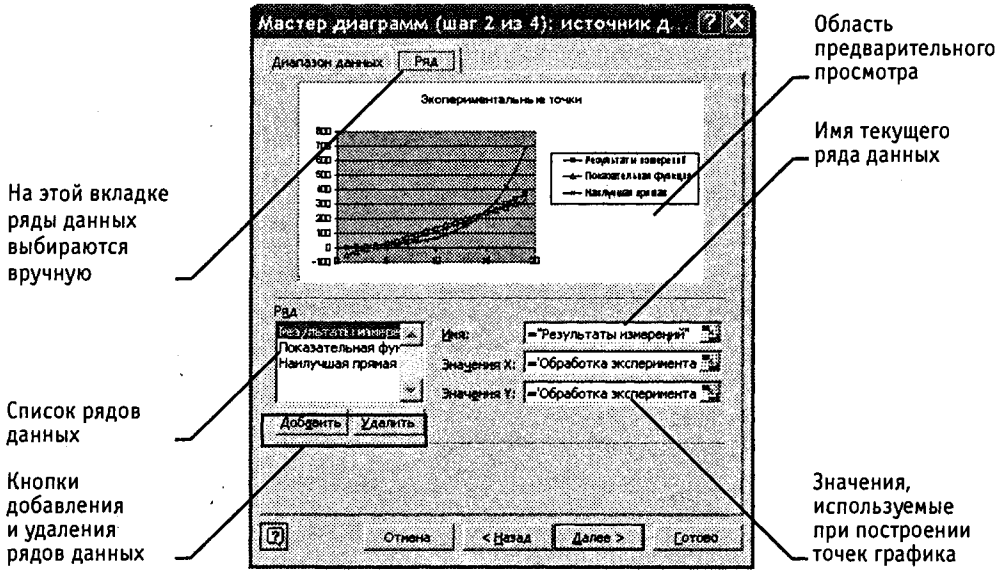


Рис. 12.7. Выбор данных, отображаемых на диаграмме

- отображение надписей, соответствующих отдельным элементам данных на графике (вкладка Подписи данных);
- представление данных, использованных при построении графика, в виде таблицы (вкладка Таблица данных).

В зависимости от типа диаграммы некоторые из перечисленных вкладок могут отсутствовать.

Размещение диаграммы

На последнем этапе работы мастера (после щелчка на кнопке Далее) указывается, следует ли использовать для размещения диаграммы новый рабочий лист или один из имеющихся. Обычно этот выбор важен только для последующей печати документа, содержащего диаграмму. После щелчка на кнопке Готово диаграмма строится автоматически и вставляется на указанный рабочий лист (рис. 12.8).

Редактирование диаграммы

Готовую диаграмму можно изменить. Она состоит из набора отдельных элементов, таких, как сами графики (ряды данных), оси координат, заголовок диаграммы, область построения и прочее. При щелчке на элементе диаграммы он выделяется маркерами, а при наведении на него указателя мыши — описывается всплывающей подсказкой. Открыть диалоговое окно для форматирования элемента диаграммы можно через меню Формат (для выделенного элемента) или через контекстное меню (команда Формат). Различные вкладки открывшегося диалогового окна позволяют изменять параметры отображения выбранного элемента данных.

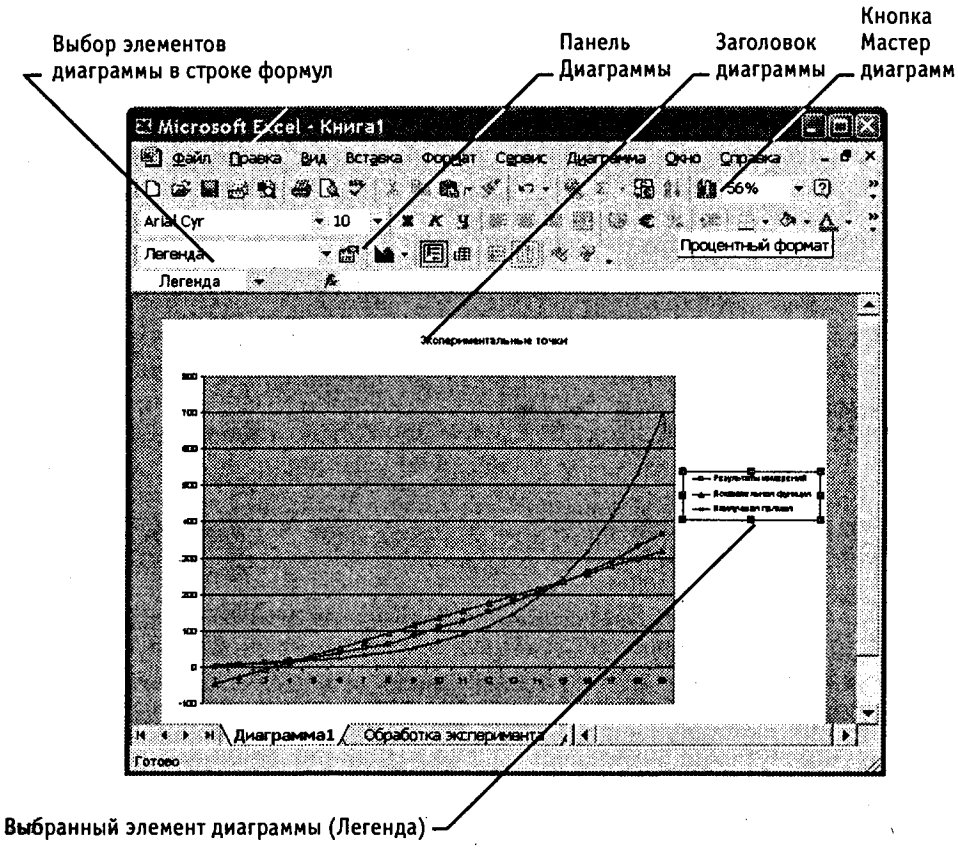


Рис. 12.8. Готовая диаграмма Excel

Если требуется внести в диаграмму существенные изменения, следует вновь воспользоваться мастером диаграмм. Для этого следует открыть рабочий лист с диаграммой или выбрать диаграмму, внедренную в рабочий лист с данными. Запустив мастер диаграмм, можно изменить текущие параметры, которые рассматриваются в окнах мастера как заданные по умолчанию.

Чтобы удалить диаграмму, можно удалить рабочий лист, на котором она расположена (Правка ▸ Удалить лист), или выбрать диаграмму, внедренную в рабочий лист с данными, и нажать клавишу DELETE.

Практическое занятие

Упражнение 12.1. Обработка данных

1. Запустите программу *Excel* (Пуск ▸ Программы ▸ Microsoft Excel).
2. Создайте новую рабочую книгу (кнопка Создать на стандартной панели инструментов).

3. Дважды щелкните на ярлычке текущего рабочего листа и дайте этому рабочему листу имя Данные.
4. Дайте команду Файл ▸ Сохранить как и сохраните рабочую книгу под именем book.xls.
5. Сделайте текущей ячейку A1 и введите в нее заголовок Результаты измерений.
6. Введите произвольные числа в последовательные ячейки столбца A, начиная с ячейки A2.
7. Введите в ячейку B1 строку Удвоенное значение.
8. Введите в ячейку C1 строку Квадрат значения.
9. Введите в ячейку D1 строку Квадрат следующего числа.
10. Введите в ячейку B2 формулу =2*A2.
11. Введите в ячейку C2 формулу =A2*A2.
12. Введите в ячейку D2 формулу =B2+C2+1.
13. Выделите протягиванием ячейки B2, C2 и D2.
14. Наведите указатель мыши на маркер заполнения в правом нижнем углу рамки, охватывающей выделенный диапазон. Нажмите левую кнопку мыши и перетащите этот маркер, чтобы рамка охватила столько строк в столбцах B, C и D, сколько имеется чисел в столбце A.
15. Убедитесь, что формулы автоматически модифицируются так, чтобы работать со значением ячейки в столбце A текущей строки.
16. Измените одно из значений в столбце A и убедитесь, что соответствующие значения в столбцах B, C и D в этой же строке были автоматически пересчитаны.
17. Введите в ячейку E1 строку Масштабный множитель.
18. Введите в ячейку E2 число 5.
19. Введите в ячейку F1 строку Масштабирование.
20. Введите в ячейку F2 формулу =A2*E2.
21. Используйте метод автозаполнения, чтобы скопировать эту формулу в ячейки столбца F, соответствующие заполненным ячейкам столбца A.
22. Убедитесь, что результат масштабирования оказался неверным. Это связано с тем, что адрес E2 в формуле задан относительной ссылкой.
23. Щелкните на ячейке F2, затем в строке формул. Установите текстовый курсор на ссылку E2 и нажмите клавишу F4. Убедитесь, что формула теперь выглядит как =A2*$E$2, и нажмите клавишу ENTER.
24. Повторите заполнение столбца F формулой из ячейки F2.
25. Убедитесь, что благодаря использованию абсолютной адресации значения ячеек столбца F теперь вычисляются правильно. Сохраните рабочую книгу book.xls.

Мы научились вводить текстовые и числовые данные в электронные таблицы Excel. Мы узнали, как производится ввод и вычисление формул. Мы также выяснили, как осуществляется копирование формул методом автозаполнения, и определили, в каких случаях следует использовать относительные и абсолютные ссылки.

Упражнение 12.2. Применение итоговых функций

1. Запустите программу *Excel* (Пуск ▸ Программы ▸ Microsoft Excel) и откройте рабочую книгу book.xls, созданную ранее.
2. Выберите рабочий лист Данные.
3. Сделайте текущей первую свободную ячейку в столбце A.
4. Щелкните на кнопке Автосумма на стандартной панели инструментов.
5. Убедитесь, что программа автоматически подставила в формулу функцию СУММ и правильно выбрала диапазон ячеек для суммирования. Нажмите клавишу ENTER.
6. Сделайте текущей следующую свободную ячейку в столбце A.
7. Щелкните на кнопке Вставка функции в строке формул.
8. В раскрывающемся списке Категория выберите пункт Статистические.
9. В списке Функция выберите функцию СРЗНАЧ и щелкните на кнопке OK.
10. Переместите методом перетаскивания окно Аргументы функции, если оно заслоняет нужные ячейки. Обратите внимание, что автоматически выбранный диапазон включает все ячейки с числовым содержимым, включая и ту, которая содержит сумму. Выделите правильный диапазон методом протягивания и нажмите клавишу ENTER.
11. Используя порядок действий, описанный в пп. 6–10, вычислите минимальное число в заданном наборе (функция МИН), максимальное число (МАКС), количество элементов в наборе (СЧЕТ).
12. Сохраните рабочую книгу book.xls.

Мы познакомились с некоторыми итоговыми функциями. Мы научились использовать итоговые функции для вычисления значений, характеризующих набор данных. Мы выяснили, как автоматически определяется диапазон значений, обрабатываемых функцией, и как изменить его вручную.

Упражнение 12.3. Подготовка и форматирование прайс-листа

1. Запустите программу *Excel* (Пуск ▸ Программы ▸ Microsoft Excel) и откройте рабочую книгу book.xls.
2. Выберите щелчком на ярлычке неиспользуемый рабочий лист или создайте новый (Вставка ▸ Лист). Дважды щелкните на ярлычке нового листа и переименуйте его как Прейскурант.
3. В ячейку A1 введите текст Прейскурант и нажмите клавишу ENTER.
4. В ячейку A2 введите текст Курс пересчета: и нажмите клавишу ENTER. В ячейку B2 введите текст 1 у.е.= и нажмите клавишу ENTER. В ячейку C2 введите текущий курс пересчета и нажмите клавишу ENTER.

5. В ячейку A3 введите текст Наименование товара и нажмите клавишу ENTER. В ячейку B3 введите текст Цена (у.е.) и нажмите клавишу ENTER. В ячейку C3 введите текст Цена (руб.) и нажмите клавишу ENTER.
6. В последующие ячейки столбца A введите названия товаров, включенных в прейскурант.
7. В соответствующие ячейки столбца B введите цены товаров в условных единицах.
8. В ячейку C4 введите формулу: =B4*$C$2, которая используется для пересчета цены из условных единиц в рубли.
9. Методом автозаполнения скопируйте формулы во все ячейки столбца C, которым соответствуют заполненные ячейки столбцов A и B. Почему при таком копировании получатся верные формулы?
10. Измените курс пересчета в ячейке C2. Обратите внимание, что все цены в рублях при этом обновляются автоматически.
11. Выделите методом протягивания диапазон A1:C1 и дайте команду Формат ▸ Ячейки. На вкладке Выравнивание задайте выравнивание по горизонтали По центру и установите флажок Объединение ячеек.
12. На вкладке Шрифт задайте размер шрифта равный 14 пунктам и в списке Начертание выберите вариант Полужирный. Щелкните на кнопке OK.
13. Щелкните правой кнопкой мыши на ячейке B2 и выберите в контекстном меню команду Формат ячеек. Задайте выравнивание по горизонтали По правому краю и щелкните на кнопке OK.
14. Щелкните правой кнопкой мыши на ячейке C2 и выберите в контекстном меню команду Формат ячеек. Задайте выравнивание по горизонтали По левому краю и щелкните на кнопке OK.
15. Выделите методом протягивания диапазон B2:C2. Щелкните на раскрывающей кнопке рядом с кнопкой Границы на панели инструментов Форматирование и задайте для этих ячеек толстую внешнюю границу (кнопка в правом нижнем углу открывшейся палитры).
16. Дважды щелкните на границе между заголовками столбцов A и B, B и C, C и D. Обратите внимание, как при этом изменяется ширина столбцов A, B и C.
17. Посмотрите, устраивает ли вас полученный формат таблицы. Щелкните на кнопке Предварительный просмотр на стандартной панели инструментов, чтобы увидеть, как документ будет выглядеть при печати.
18. Щелкните на кнопке Печать и напечатайте документ.
19. Сохраните рабочую книгу book.xls.

Мы научились форматировать документ Excel. При этом мы использовали такие средства, как изменение ширины столбцов, объединение ячеек, управление выравниванием текста, создание рамок ячеек. Мы выяснили, что в готовом документе заданные и вычисленные ячейки отображаются одинаково. Мы познакомились с использованием средства предварительного просмотра и произвели печать документа.

Упражнение 12.4. Построение экспериментального графика

1. Запустите программу *Excel* (Пуск ▸ Программы ▸ Microsoft Excel) и откройте рабочую книгу book.xls, созданную ранее.
2. Выберите щелчком на ярлычке неиспользуемый рабочий лист или создайте новый (Вставка ▸ Лист). Дважды щелкните на ярлычке листа и переименуйте его как Обработка эксперимента.
3. В столбец A, начиная с ячейки A1, введите произвольный набор значений независимой переменной.
4. В столбец B, начиная с ячейки B1, введите произвольный набор значений функции.
5. Методом протягивания выделите все заполненные ячейки столбцов A и B.
6. Щелкните на значке Мастер диаграмм на стандартной панели инструментов.
7. В списке Тип выберите пункт Точечная (для отображения графика, заданного парами значений). В палитре Вид выберите средний пункт в первом столбце (маркеры, соединенные гладкими кривыми). Щелкните на кнопке Далее.
8. Так как диапазон ячеек был выделен заранее, мастер диаграмм автоматически определяет расположение рядов данных. Убедитесь, что данные на диаграмме выбраны правильно. На вкладке Ряд в поле Имя укажите: Результаты измерений. Щелкните на кнопке Далее.
9. Выберите вкладку Заголовки. Убедитесь, что заданное название ряда данных автоматически использовано как заголовок диаграммы. Замените его, введя в поле Название диаграммы заголовок Экспериментальные точки. Щелкните на кнопке Далее.
10. Установите переключатель Отдельном. По желанию, задайте произвольное имя добавляемого рабочего листа. Щелкните на кнопке Готово.
11. Убедитесь, что диаграмма построена и внедрена в новый рабочий лист. Рассмотрите ее и щелкните на построенной кривой, чтобы выделить ряд данных.
12. Дайте команду Формат ▸ Выделенный ряд. Откройте вкладку Вид.
13. На панели Линия откройте палитру Цвет и выберите красный цвет. В списке Тип линии выберите пунктир.
14. На панели Маркер выберите в списке Тип маркера треугольный маркер. В палитрах Цвет и Фон выберите зеленый цвет.
15. Щелкните на кнопке OK, снимите выделение с ряда данных и посмотрите, как изменился вид графика.
16. Сохраните рабочую книгу.

Мы научились строить графики на основе данных, содержащихся на рабочем листе, настраивать формат диаграммы, задавать отображаемые данные и оформлять получающуюся диаграмму. Мы также узнали, как можно изменить формат готовой диаграммы.

Упражнение 12.5. Анализ данных с использованием метода наименьших квадратов

Задача. Для заданного набора пар значений независимой переменной и функции определить наилучшие линейное приближение в виде прямой с уравнением $y = ax + b$ и показательное приближение в виде линии с уравнением $y = b \cdot a^x$.

1. Запустите программу *Excel* (Пуск ▸ Программы ▸ Microsoft Excel) и откройте рабочую книгу book.xls, созданную ранее.
2. Щелчком на ярлычке выберите рабочий лист Обработка эксперимента.
3. Сделайте ячейку C1 текущей и щелкните на кнопке Вставка функции в строке формул.
4. В окне мастера функций выберите категорию Ссылки и массивы и функцию ИНДЕКС. В новом диалоговом окне выберите первый вариант набора параметров.
5. Установите текстовый курсор в первое поле для ввода параметров в окне Аргументы функции и выберите в раскрывающемся списке в строке формул пункт Другие функции.
6. С помощью мастера функций выберите функцию ЛИНЕЙН категории Статистические.
7. В качестве первого параметра функции ЛИНЕЙН выберите диапазон, содержащий значения функции (столбец B).
8. В качестве второго параметра функции ЛИНЕЙН выберите диапазон, содержащий значения независимой переменной (столбец A).
9. Переместите текстовый курсор в строке формул, чтобы он стоял на имени функции ИНДЕКС. В качестве второго параметра функции ИНДЕКС задайте число 1. Щелкните на кнопке OK в окне Аргументы функции.

Функция ЛИНЕЙН возвращает коэффициенты уравнения прямой в виде массива из двух элементов. С помощью функции ИНДЕКС выбирается нужный элемент.

10. Сделайте текущей ячейку D1. Повторите операции, описанные в пп. 3–9, чтобы в итоге в этой ячейке появилась формула: =ИНДЕКС(ЛИНЕЙН(B1:B20;A1:A20);2). Ее можно ввести и вручную (посимвольно). Теперь в ячейках C1 и D1 вычислены, соответственно, коэффициенты a и b уравнения наилучшей прямой.
11. Сделайте текущей ячейку C2. Повторите операции, описанные в пп. 3–9, или введите вручную следующую формулу:

    =ИНДЕКС(ЛГРФПРИБЛ(B1:B20;A1:A20);1)

12. Сделайте текущей ячейку D2. Повторите операции, описанные в пп. 3–9, или введите вручную следующую формулу:

    =ИНДЕКС(ЛГРФПРИБЛ(B1:B20;A1:A20);2)

    Теперь ячейки C2 и D2 содержат, соответственно, коэффициенты a и b уравнения наилучшего показательного приближения.

Для интерполяции или экстраполяции оптимальной кривой без явного определения ее параметров можно использовать функции ТЕНДЕНЦИЯ (для линейной зависимости) и РОСТ (для показательной зависимости).

13. Для построения наилучшей прямой другим способом дайте команду Сервис ▸ Data Analysis (Анализ данных).
14. Откроется одноименное диалоговое окно. В списке Analysis Tools (Инструменты анализа) выберите пункт Regression (Регрессия), после чего щелкните на кнопке OK.
15. В поле Input Y Range (Входной интервал Y) укажите методом протягивания диапазон, содержащий значения функции (столбец B).
16. В поле Input X Range (Входной интервал X) укажите методом протягивания диапазон, содержащий значения независимой переменной (столбец A).
17. Установите переключатель New Worksheet (Новый рабочий лист) и задайте для него имя Результат расчета.
18. Щелкните на кнопке OK и по окончании расчета откройте рабочий лист Результат расчета. Убедитесь, что вычисленные коэффициенты (см. ячейки B17 и B18) совпали с полученными первым методом.
19. Сохраните рабочую книгу book.xls.

Мы научились анализировать с помощью программы Excel экспериментальные данные с использованием метода наименьших квадратов. Мы применили для вычислений разные средства программы Excel. Мы получили информацию, необходимую для построения графиков нужных приближений.

Упражнение 12.6. Применение таблиц подстановки

Задача. Построить графики функций, коэффициенты которых определены в предыдущем упражнении.

1. Запустите программу *Excel* (Пуск ▸ Программы ▸ Microsoft Excel) и откройте рабочую книгу book.xls.
2. Выберите щелчком на ярлычке рабочий лист Обработка эксперимента.
3. Так как программа *Excel* не позволяет непосредственно строить графики функций, заданных формулами, необходимо сначала табулировать формулу, то есть создать таблицу значений функций для заданных значений переменной. Сделайте текущей ячейку C3 и занесите в нее значение 0. Эта ячейка будет использоваться как ячейка ввода, на которую будут ссылаться формулы.
4. Методом протягивания выделите значения в столбце A. Дайте команду Правка ▸ Копировать, чтобы перенести эти данные в буфер обмена. Сделайте текущей ячейку F2 и дайте команду Правка ▸ Вставить, чтобы скопировать заданные значения независимой переменной в столбец F, начиная со второй строки.
5. В ячейку G1 введите формулу =C3*$C$1+$D$1. Здесь C3 — ячейка ввода, а в качестве других ссылок используются вычисленные методом наименьших квадратов коэффициенты уравнения прямой.

6. В ячейку H1 введите формулу =$D$2*$C$2^C3 для вычисления значения показательной функции. В программе *Excel* можно табулировать несколько функций одной переменной в рамках единой операции.
7. Выделите прямоугольный диапазон, включающий столбцы F, G и H и строки от строки 1, содержащей формулы, до последней строки с данными в столбце F.
8. Дайте команду Данные ▸ Таблица подстановки. Выберите поле Подставлять значения по строкам в и щелкните на ячейке ввода C3.
9. Щелкните на кнопке OK, чтобы заполнить пустые ячейки в столбцах G и H выделенного диапазона значениями формул в ячейках первой строки для значений независимой переменной, выбранных из столбца F.
10. Переключитесь на рабочий лист Диаграмма1 (если используемое по умолчанию название листа с диаграммой было изменено, используйте свое название).
11. Щелкните на кнопке Мастер диаграмм на стандартной панели инструментов и пропустите первый этап щелчком на кнопке Далее.
12. Выберите вкладку Ряд и щелкните на кнопке Добавить. В поле Имя укажите: Наилучшая прямая. В поле Значения X укажите диапазон ячеек с данными в столбце F, а в поле Значения Y укажите диапазон ячеек в столбце G.
13. Еще раз щелкните на кнопке Добавить. В поле Имя укажите: Показательная функция. В поле Значения X укажите диапазон ячеек с данными в столбце F, а в поле Значения Y укажите диапазон ячеек в столбце H.
14. Щелкните на кнопке Готово, чтобы перестроить диаграмму в соответствии с новыми настройками.
15. Сохраните рабочую книгу book.xls.

Мы научились создавать таблицу подстановки, содержащую значения заданных формул для нужных значений независимой переменной. Мы применили эту возможность программы Excel для построения графиков функций, заданных формулами. Мы также научились редактировать ранее построенную диаграмму, нанося на нее дополнительные графики.

Упражнение 12.7. Решение уравнений средствами программы Excel

Задача. Найти решение уравнения $x^3 - 3x^2 + x = -1$.

1. Запустите программу *Excel* (Пуск ▸ Программы ▸ Microsoft Excel) и откройте рабочую книгу book.xls, созданную ранее.
2. Создайте новый рабочий лист (Вставка ▸ Лист), дважды щелкните на его ярлычке и присвойте ему имя Уравнение.
3. Занесите в ячейку A1 значение 0.
4. Занесите в ячейку B1 левую часть уравнения, используя в качестве независимой переменной ссылку на ячейку A1. Соответствующая формула может, например, иметь вид =A1^3-3*A1^2+A1.

5. Дайте команду Сервис ▸ Подбор параметра.
6. В поле Установить в ячейке укажите B1, в поле Значение задайте –1, в поле Изменяя значение ячейки укажите A1.
7. Щелкните на кнопке OK и посмотрите на результат подбора, отображаемый в диалоговом окне Результат подбора параметра. Щелкните на кнопке OK, чтобы сохранить полученные значения ячеек, участвовавших в операции.
8. Повторите расчет, задавая в ячейке A1 другие начальные значения, например 0,5 или 2. Совпали ли результаты вычислений? Чем можно объяснить различия?
9. Сохраните рабочую книгу book.xls.

Мы научились численно решать с помощью программы Excel уравнения, содержащие одно неизвестное и задаваемые формулой. Мы выяснили, что при наличии нескольких корней результат решения уравнения зависит от того, какое число было выбрано в качестве начального приближения.

Упражнение 12.8. Решение задач оптимизации

Задача. Завод производит электронные приборы трех видов (прибор A, прибор B и прибор C), используя при сборке микросхемы трех типов (тип 1, тип 2 и тип 3). Расход микросхем задается следующей таблицей:

	Прибор A	Прибор B	Прибор C
Тип 1	2	5	1
Тип 2	2	0	4
Тип 3	2	1	1

Стоимость изготовленных приборов одинакова.

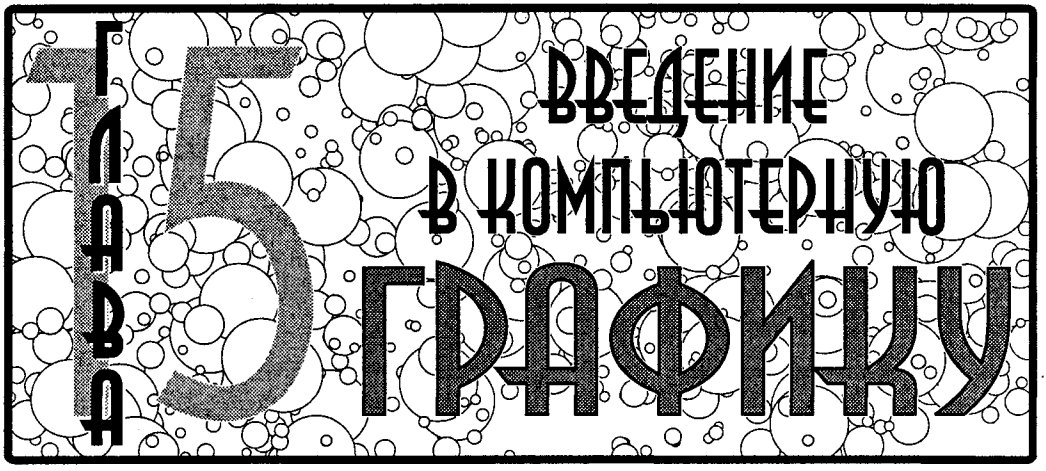
Ежедневно на склад завода поступает 400 микросхем типа 1 и по 500 микросхем типов 2 и 3. Каково оптимальное соотношение дневного производства приборов различного типа, если производственные мощности завода позволяют использовать запас поступивших микросхем полностью?

1. Запустите программу *Excel* (Пуск ▸ Программы ▸ Microsoft Excel) и откройте рабочую книгу book.xls, созданную ранее.
2. Создайте новый рабочий лист (Вставка ▸ Лист), дважды щелкните на его ярлычке и присвойте ему имя Организация производства.
3. В ячейки A2, A3 и A4 занесите дневной запас комплектующих — числа 400, 500 и 500 соответственно.
4. В ячейки C1, D1 и E1 занесите нули — в дальнейшем значения этих ячеек будут подобраны автоматически.
5. В ячейках диапазона C2:E4 разместите таблицу расхода комплектующих.
6. В ячейках B2:B4 нужно указать формулы для расчета расхода комплектующих по типам. В ячейке B2 формула будет иметь вид =$C$1*C2+$D$1*D2+$E$1*E2,

а остальные формулы можно получить методом автозаполнения (обратите внимание на использование абсолютных и относительных ссылок).

7. В ячейку F1 занесите формулу, вычисляющую общее число произведенных приборов: для этого выделите диапазон C1:E1 и щелкните на кнопке Автосумма на стандартной панели инструментов.
8. Дайте команду Сервис ▸ Solver (Поиск решения) — откроется диалоговое окно Solver Parameters (Поиск решения).
9. В поле Set Target Cell (Установить целевую) укажите ячейку, содержащую оптимизируемое значение (F1). Установите переключатель Equal To Max (Равной максимальному значению) (требуется максимальный объем производства).
10. В поле By Changing Cells (Изменяя ячейки) задайте диапазон подбираемых параметров — C1:E1.
11. Чтобы определить набор ограничений, щелкните на кнопке Add (Добавить). В диалоговом окне Add Constraint (Добавление ограничения) в поле Cell Reference (Ссылка на ячейку) укажите диапазон B2:B4. В качестве условия задайте <=. В поле Constraint (Ограничение) задайте диапазон A2:A4. Это условие указывает, что дневной расход комплектующих не должен превосходить запасов. Щелкните на кнопке OK.
12. Снова щелкните на кнопке Add (Добавить). В поле Cell Reference (Ссылка на ячейку) укажите диапазон C1:E1. В качестве условия задайте >=. В поле Constraint (Ограничение) задайте число 0. Это условие указывает, что число производимых приборов неотрицательно. Щелкните на кнопке OK.
13. Снова щелкните на кнопке Add (Добавить). Cell Reference (Ссылка на ячейку) укажите диапазон C1:E1. В качестве условия выберите пункт int (цел). Это условие не позволяет производить доли приборов. Щелкните на кнопке OK.
14. Щелкните на кнопке Solve (Выполнить). По завершении оптимизации откроется диалоговое окно Solver Results (Результаты поиска решения).
15. Установите переключатель Keep Solver Solution (Сохранить найденное решение), после чего щелкните на кнопке OK.
16. Проанализируйте полученное решение. Кажется ли оно очевидным? Проверьте его оптимальность, экспериментируя со значениями ячеек C1:E1. Чтобы восстановить оптимальные значения, можно в любой момент повторить операцию поиска решения.
17. Сохраните рабочую книгу book.xls.

Мы узнали, как использовать программу Excel для решения сложных задач оптимизации. Мы научились формулировать условия задачи табличным образом, формировать ограничения, которым должно удовлетворять решение, и производить поиск оптимального набора переменных. Мы также выяснили, что даже для несложной задачи оптимизации найти оптимальное решение подбором практически невозможно.



15.1. Основы представления графических данных

Виды компьютерной графики

Представление данных на мониторе компьютера в графическом виде впервые было реализовано в середине 50-х годов для больших ЭВМ, применявшихся в научных и военных исследованиях. С тех пор графический способ отображения данных стал неотъемлемой принадлежностью подавляющего числа компьютерных систем, в особенности персональных. Графический интерфейс пользователя сегодня является стандартом «де-факто» для программного обеспечения разных классов, начиная с операционных систем.

Существует специальная область информатики, изучающая методы и средства создания и обработки изображений с помощью программно-аппаратных вычислительных комплексов, — *компьютерная графика*. Она охватывает все виды и формы представления изображений, доступных для восприятия человеком либо на экране монитора, либо в виде копии на внешнем носителе (бумага, кинопленка, ткань и прочее). Без компьютерной графики невозможно представить себе не только компьютерный, но и обычный, вполне материальный мир. Визуализация данных находит применение в самых разных сферах человеческой деятельности. Для примера назовем медицину (компьютерная томография), научные исследования (визуализация строения вещества, векторных полей и других данных), моделирование тканей и одежды, опытно-конструкторские разработки.

В зависимости от способа формирования изображений компьютерную графику принято подразделять на *растровую, векторную и фрактальную*.

Отдельным предметом считается *трехмерная (3D) графика*, изучающая приемы и методы построения объемных моделей объектов в виртуальном пространстве. Как правило, в ней сочетаются векторный и растровый способы формирования изображений.

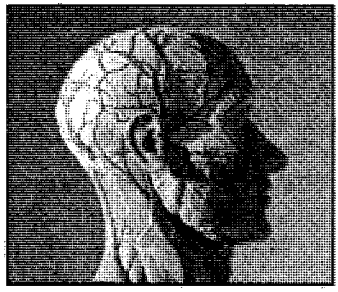
Особенности цветового охвата характеризуют такие понятия, как *черно-белая и цветная графика*. На специализацию в отдельных областях указывают названия

некоторых разделов: *инженерная графика, научная графика, Web-графика, компьютерная полиграфия* и прочие.

На стыке компьютерных, телевизионных и кинотехнологий зародилась и стремительно развивается сравнительно новая область *компьютерной графики и анимации*.

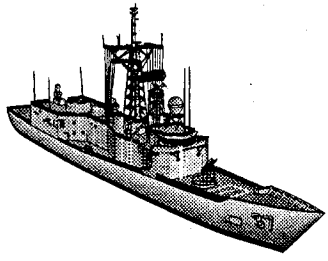
Заметное место в компьютерной графике отведено развлечениям. Появилось даже такое понятие, как механизм графического представления данных (*Graphics Engine*) в играх. Рынок игровых программ имеет оборот в десятки миллиардов долларов и часто инициализирует очередной этап совершенствования графики и анимации.

Хотя компьютерная графика служит всего лишь инструментом, ее структура и методы основаны на передовых достижениях фундаментальных и прикладных наук: математики, физики, химии, биологии, статистики, программирования и множества других. Это замечание справедливо как для программных, так и для аппаратных средств создания и обработки изображений на компьютере. Поэтому компьютерная графика является одной из наиболее бурно развивающихся отраслей информатики и во многих случаях выступает «локомотивом», тянущим за собой всю компьютерную индустрию.

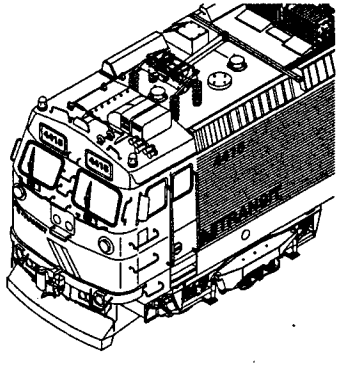


Растровая графика

Векторная графика



Трехмерная графика



Инженерная графика

Растровая графика

Для растровых изображений, состоящих из точек, особую важность имеет понятие *разрешения*, выражающее количество точек, приходящихся на единицу длины. При этом следует различать:

- разрешение оригинала;
- разрешение экранного изображения;
- разрешение печатного изображения.

Разрешение оригинала. Разрешение оригинала измеряется в *точках на дюйм* (*dots per inch — dpi*) и зависит от требований к качеству изображения и размеру файла, способу оцифровки или методу создания исходной иллюстрации, избранному формату файла и другим параметрам. В общем случае действует правило: чем выше требования к качеству, тем выше должно быть разрешение оригинала.

Разрешение экранного изображения. Для экранных копий изображения элементарную точку растра принято называть *пикселом*. Размер пиксела варьируется в зависимости от выбранного *экранного разрешения* (из диапазона стандартных значений), *разрешения оригинала* и масштаба отображения.

Мониторы для обработки изображений с диагональю 19–24 дюйма (профессионального класса), как правило, обеспечивают стандартные экранные разрешения 640×480, 800×600, 1024×768, 1280×1024, 1600×1200, 1600×1280, 1920×1440, 1920×1600, 2048×1536 точек. Расстояние между соседними точками люминофора у качественного монитора составляет 0,22–0,25 мм.

Для экранной копии достаточно разрешения 72 *dpi*, для распечатки на цветном или лазерном принтере 150–200 *dpi*, для вывода на фотоэкспонирующем устройстве 200–300 *dpi*. Установлено эмпирическое правило, что при распечатке величина разрешения оригинала должна быть в 1,5 раза больше, чем *линиатура растра* устройства вывода. В случае, если твердая копия будет увеличена по сравнению с оригиналом, эти величины следует умножить на коэффициент масштабирования.

Разрешение печатного изображения и понятие линиатуры. Размер точки растрового изображения как на твердой копии (бумага, пленка и т. д.), так и на экране зависит от примененного метода и параметров *растрирования* оригинала. При растрировании на оригинал как бы накладывается сетка линий, ячейки которой образуют *элемент растра*. Частота сетки растра измеряется числом *линий на дюйм* (*lines per inch — lpi*) и называется *линиатурой*.

Размер точки растра рассчитывается для каждого элемента и зависит от интенсивности тона в данной ячейке. Чем больше интенсивность, тем плотнее заполняется элемент растра. То есть, если в ячейку попал абсолютно черный цвет, размер точки растра совпадет с размером элемента растра. В этом случае говорят о 100% заполняемости. Для абсолютно белого цвета значение заполняемости составит 0%. На практике заполняемость элемента на отпечатке обычно составляет от 3 до 98%. При этом все точки растра имеют одинаковую оптическую плотность, в идеале приближающуюся к абсолютно черному цвету. Иллюзия более темного тона создается за счет увеличения размеров точек и, как следствие, сокращения пробельного поля между ними при одинаковом расстоянии между центрами элементов растра (рис. 15.1). Такой метод называют растрированием с *амплитудной модуляцией (АМ)*.

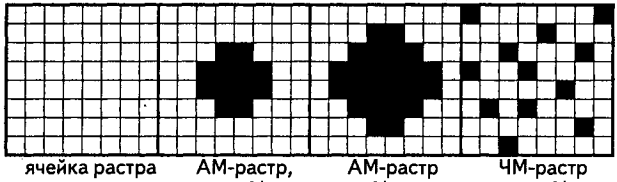


Рис. 15.1. Примеры амплитудной и частотной модуляции растра

Существует и метод растрирования с *частотной модуляцией (ЧМ)*, когда интенсивность тона регулируется изменением расстояния между соседними точками одинакового размера. Таким образом, при частотно-модулированном растрировании

в ячейках растра с разной интенсивностью тона находится разное число точек (см. рис. 15.1). Изображения, растрированные ЧМ-методом, выглядят более качественно, так как размер точек минимален и, во всяком случае, существенно меньше, чем средний размер точки при АМ-растрировании. Еще более повышает качество изображения разновидность ЧМ-метода, называемая *стохастическим растрированием*. В этом случае рассчитывается число точек, необходимое для отображения требуемой интенсивности тона в ячейке растра. Затем эти точки располагаются внутри ячейки на расстояниях, вычисленных квазислучайным методом (на самом деле используется специальный математический алгоритм). То есть регулярная структура растра внутри ячейки, как и на изображении в целом, вообще отсутствует (рис. 15.2). Поэтому при стохастическом ЧМ-растрировании теряет смысл понятие линиатуры растра, имеет значение лишь разрешающая способность устройства вывода. Такой способ требует больших затрат вычислительных ресурсов и высокой точности полиграфического оборудования; он применяется в основном для художественных работ, при печати с числом красок, превышающим четыре.

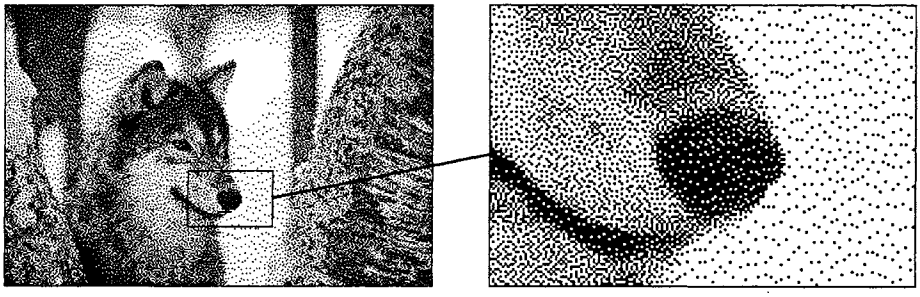


Рис. 15.2. Пример использования стохастического растра

Интенсивность тона (так называемую *светлоту*) принято подразделять на 256 уровней. Большее число градаций не воспринимается зрением человека и является избыточным. Меньшее число ухудшает восприятие изображения (минимально допустимым для качественной полутоновой иллюстрации принято значение 150 уровней). Нетрудно подсчитать, что для воспроизведения 256 уровней тона достаточно иметь размер ячейки растра $256 = 16 \times 16$ точек.

Между разрешением оригинала, частотой растра и градацией уровней существует зависимость, описываемая формулой:

$$N = \left(\frac{dpi}{lpi}\right)^2 + 1;\ lpi = \frac{dpi}{\sqrt{N-1}},$$

где N — число градаций уровней тона (оттенков), dpi — разрешение устройства вывода (отображения), lpi — линиатура растра. Единица в формуле соответствует абсолютно белому цвету, когда ячейка растра вообще не заполнена.

При выводе копии изображения на принтере или полиграфическом оборудовании линиатуру растра выбирают, исходя из компромисса между требуемым качеством, возможностями аппаратуры и параметрами печатных материалов. Для лазерных

принтеров рекомендуемая линиатура составляет 65–100 *lpi*, для газетного производства — 65–85 *lpi*, для книжно-журнального — 85–133 *lpi*, для художественных и рекламных работ — 133–300 *lpi*.

При печати изображений с наложением растров друг на друга, например многоцветных, каждый последующий растр поворачивается на определенный угол. Традиционными для цветной печати считаются углы поворота: 105 градусов для голубой печатной формы, 75 градусов для пурпурной, 90 градусов для желтой и 45 градусов для черной. При этом ячейка растра становится косоугольной, и для воспроизведения 256 градаций тона с линиатурой 150 *lpi* уже недостаточно разрешения 16×150=2400 *dpi*. Поэтому для фотоэкспонирующих устройств профессионального класса принято минимальное стандартное разрешение 2540 *dpi*, обеспечивающее качественное растрирование при разных углах поворота растра. Таким образом, коэффициент, учитывающий поправку на угол поворота растра, для цветных изображений составляет 1,06.

Динамический диапазон. Качество воспроизведения тоновых изображений принято оценивать *динамическим диапазоном* (*D*). Это *оптическая плотность*, численно равная десятичному логарифму величины, обратной *коэффициенту пропускания* τ (для оригиналов, рассматриваемых «на просвет», например слайдов) или *коэффициенту отражения* ρ (для прочих оригиналов, например полиграфических отпечатков):

$$D = \lg\frac{1}{\tau}; \quad D = \lg\frac{1}{\rho}; \qquad \rho = \frac{F_\rho}{F_0}; \quad \tau = \frac{F_\tau}{F_0},$$

где F_0 — падающий световой поток, F_ρ — отраженный световой поток, F_τ — пропущенный световой поток.

Для оптических сред, пропускающих свет, динамический диапазон лежит в пределах от 0 до 4. Для поверхностей, отражающих свет, значение динамического диапазона составляет от 0 до 2. Чем выше динамический диапазон, тем большее число полутонов присутствует в изображении и тем лучше качество его восприятия.

Связь между параметрами изображения и размером файла. Средствами растровой графики принято иллюстрировать работы, требующие высокой точности в передаче цветов и полутонов. Однако размеры файлов растровых иллюстраций стремительно растут с увеличением разрешения. Фотоснимок, предназначенный для домашнего прочтения (стандартный размер 10×15 см, оцифрованный с разрешением 200–300 *dpi*, цветовое разрешение 24 бита), занимает в формате *TIFF* с включенным режимом сжатия около 4 Мбайт. Оцифрованный с высоким разрешением слайд занимает 45–50 Мбайт. Цветоделенное цветное изображение формата А4 занимает 120–150 Мбайт.

Масштабирование растровых изображений. Одним из недостатков растровой графики является так называемая *пикселизация* изображений при их увеличении (если не приняты специальные меры). Раз в оригинале присутствует определенное количество точек, то при большем масштабе увеличивается и их размер, становятся заметны элементы растра, что искажает саму иллюстрацию (рис. 15.3). Для противодействия пикселизации принято заранее оцифровывать оригинал с разре-

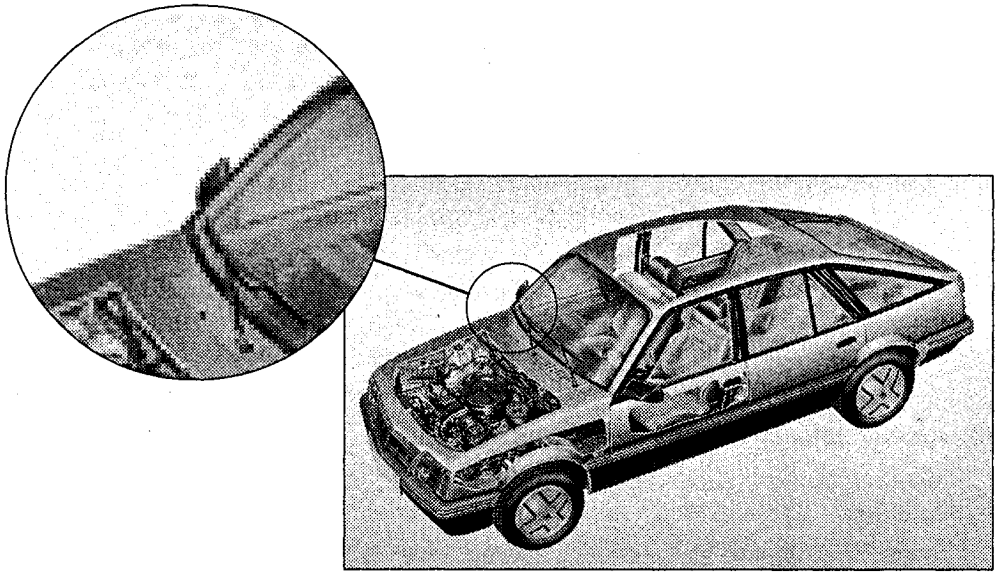


Рис. 15.3. Эффект пикселизации при масштабировании растрового изображения

шением, достаточным для качественной визуализации при масштабировании. Другой прием состоит в применении стохастического растра, позволяющего уменьшить эффект пикселизации в определенных пределах. Наконец, при масштабировании используют метод интерполяции, когда увеличение размера иллюстрации происходит не за счет масштабирования точек, а путем добавления необходимого числа промежуточных точек.

Векторная графика

Если в растровой графике базовым элементом изображения является точка, то в векторной графике — *линия*. Линия описывается математически как единый объект, и потому объем данных для отображения объекта средствами векторной графики существенно меньше, чем в растровой графике.

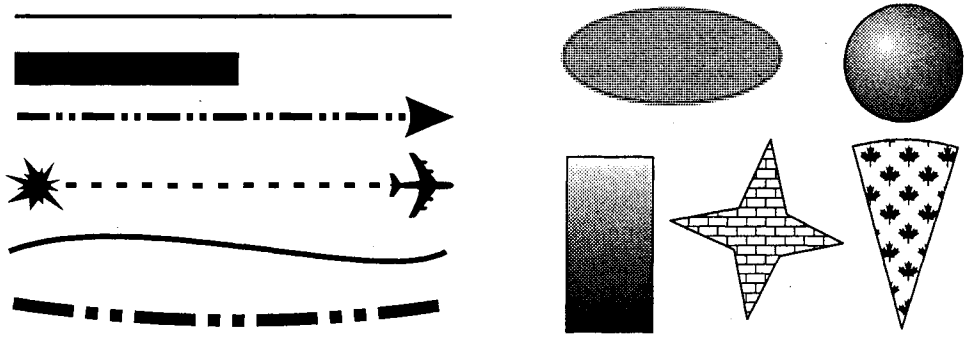


Рис. 15.4. Объекты векторной графики

Линия — элементарный *объект* векторной графики. Как и любой объект, линия обладает свойствами: формой (прямая, кривая), толщиной, цветом, начертанием (сплошная, пунктирная). Замкнутые линии приобретают свойство *заполнения*. Охватываемое ими пространство может быть заполнено другими объектами (*текстуры, карты*) или выбранным цветом.

Простейшая незамкнутая линия ограничена двумя точками, именуемыми *узлами*. Узлы также имеют свойства, параметры которых влияют на форму конца линии и характер сопряжения с другими объектами.

Все прочие объекты векторной графики составляются из линий. Например, куб можно составить из шести связанных прямоугольников, каждый из которых, в свою очередь, образован четырьмя связанными линиями. Возможно представить куб и как двенадцать связанных линий, образующих ребра.

Математические основы векторной графики

Рассмотрим подробнее способы представления различных объектов в векторной графике.

Точка. Этот объект на плоскости представляется двумя числами (x, y), указывающими его положение относительно начала координат.

Прямая линия. Ей соответствует уравнение $y = kx + b$. Указав параметры k и b, всегда можно отобразить бесконечную прямую линию в известной системе координат, то есть для задания прямой достаточно двух параметров.

Отрезок прямой. Он отличается тем, что требует для описания еще двух параметров — например, координат x_1 и x_2 начала и конца отрезка.

Кривая второго порядка. К этому классу кривых относятся параболы, гиперболы, эллипсы, окружности, то есть все линии, уравнения которых содержат степени не выше второй. Кривая второго порядка не имеет *точек перегиба*. Прямые линии являются всего лишь частным случаем кривых второго порядка. Формула кривой второго порядка в общем виде может выглядеть, например, так:

$$x^2 + a_1y^2 + a_2xy + a_3x + a^4y + a_5 = 0.$$

Таким образом, для описания бесконечной кривой второго порядка достаточно пяти параметров. Если требуется построить отрезок кривой, понадобятся еще два параметра.

Кривая третьего порядка. Отличие этих кривых от кривых второго порядка состоит в возможном наличии точки перегиба. Например, график функции $y = x^3$ имеет точку перегиба в начале координат (рис. 15.5). Именно эта особенность позволяет сделать кривые третьего порядка основой отображения природных объектов в векторной графике. Например, линии изгиба человеческого тела весьма близки к кривым третьего порядка. Все кривые второго порядка, как и прямые, являются частными случаями кривых третьего порядка.

В общем случае уравнение кривой третьего порядка можно записать так:

$$x^3 + a_1y^3 + a_2x^2y + a_3xy^2 + a_4x^2 + a_5y^2 + a_6xy + a_7x + a_8y + a_9 = 0$$

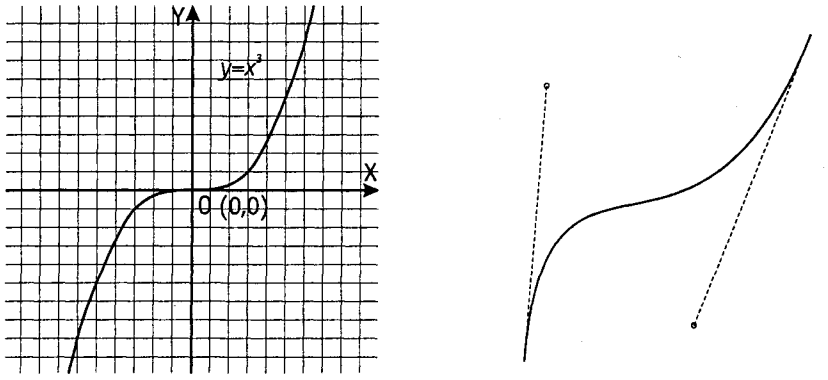


Рис. 15.5. Кривая третьего порядка (слева) и кривая Безье (справа)

Таким образом, кривая третьего порядка описывается девятью параметрами. Описание ее отрезка потребует на два параметра больше.

Кривые Безье. Это особый, упрощенный вид кривых третьего порядка (см. рис. 15.5). Метод построения кривой Безье основан на использовании пары касательных, проведенных к отрезку линии в ее окончаниях. Отрезки кривых Безье описываются восемью параметрами, поэтому работать с ними удобнее. На форму линии влияет угол наклона касательной и длина ее отрезка. Таким образом, касательные играют роль виртуальных «рычагов», с помощью которых управляют кривой.

Фрактальная графика

Фрактальная графика, как и векторная, основана на математических вычислениях. Однако базовым элементом фрактальной графики является сама математическая формула, то есть никаких объектов в памяти компьютера не хранится и изображение строится исключительно по уравнениям. Таким способом строят как простейшие регулярные структуры, так и сложные иллюстрации, имитирующие природные ландшафты и трехмерные объекты (рис. 15.6).

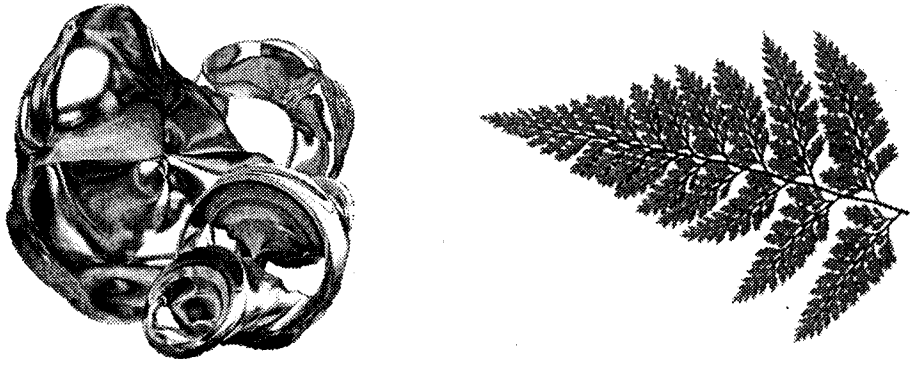


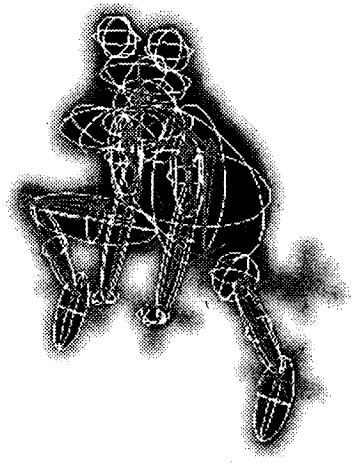
Рис. 15.6. Примеры фрактальных объектов

Основные понятия трехмерной графики

Трехмерная графика нашла широкое применение в таких областях, как научные расчеты, инженерное проектирование, компьютерное моделирование физических объектов. В качестве примера рассмотрим наиболее сложный вариант трехмерного моделирования — создание подвижного изображения реального физического тела.

В упрощенном виде для пространственного моделирования объекта требуется:

- спроектировать и создать виртуальный каркас («скелет») объекта, наиболее полно соответствующий его реальной форме;
- спроектировать и создать виртуальные материалы, по физическим свойствам визуализации похожие на реальные;
- присвоить материалы различным частям поверхности объекта (на профессиональном жаргоне — «спроектировать текстуры на объект»);
- настроить физические параметры пространства, в котором будет действовать объект, — задать освещение, гравитацию, свойства атмосферы, свойства взаимодействующих объектов и поверхностей;
- задать траектории движения объектов;
- рассчитать результирующую последовательность кадров;
- наложить поверхностные эффекты на итоговый анимационный ролик.



Для создания реалистичной модели объекта используют геометрические примитивы (прямоугольник, куб, шар, конус и прочие) и гладкие, так называемые *сплайновые поверхности*. В последнем случае применяют чаще всего метод *бикубических рациональных B-сплайнов на неравномерной сетке* (*NURBS*). Вид поверхности при этом определяется расположенной в пространстве сеткой опорных точек. Каждой точке присваивается коэффициент, величина которого определяет степень ее влияния на часть поверхности, проходящей вблизи точки. От взаимного расположения точек и величины коэффициентов зависит форма и «гладкость» поверхности в целом. Специальный инструментарий позволяет обрабатывать примитивы, составляющие объект, как единое целое, с учетом их взаимодействия на основе заданной физической модели.

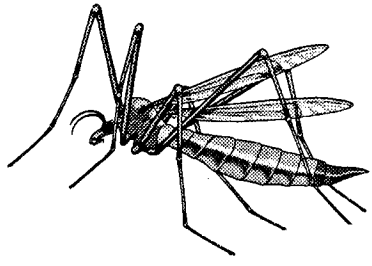
Деформация объекта обеспечивается перемещением контрольных точек, расположенных вблизи. Каждая контрольная точка связана с близлежащими опорными точками, степень ее влияния на них определяется удаленностью. Другой метод называют *сеткой деформации*. Вокруг объекта или его части размещается трехмерная сетка, перемещение любой точки которой вызывает упругую деформацию как самой сетки, так и окруженного объекта.

Еще одним способом построения объектов из примитивов служит *твердотельное моделирование*. Объекты представлены твердыми телами, которые при взаимодействии с другими телами различными способами (объединение, вычитание, слияние и другие) претерпевают необходимую трансформацию. Например, вычитание из прямоугольного параллелепипеда шара приведет к образованию в параллелепипеде полукруглой лунки.

После формирования «скелета» объекта необходимо покрыть его поверхность материалами. Все многообразие свойств материалов в компьютерном моделировании сводится к визуализации поверхности, то есть к расчету коэффициента прозрачности поверхности и угла преломления лучей света на границе материала и окружающего пространства. Для построения поверхностей материалов используют пять основных физических моделей:

- *Bouknight* — поверхности с диффузным отражением без бликов (например, матовый пластик);
- *Phong* — поверхности со структурированными микронеровностями (например, металлические);
- *Blinn* — поверхности со специальным распределением микронеровностей с учетом взаимных перекрытий (например, глянец);
- *Whitted* — модель, позволяющая дополнительно учитывать поляризацию света;
- *Hall* — модель, позволяющая корректировать направления отражения и параметры преломления света.

Закраска поверхностей осуществляется методами Гуро (*Gouraud*) или Фонга (*Phong*). В первом случае цвет примитива рассчитывается лишь в его вершинах, а затем линейно интерполируется по поверхности. Во втором случае строится нормаль к объекту в целом, ее вектор интерполируется по поверхности составляющих примитивов и освещение рассчитывается для каждой точки.

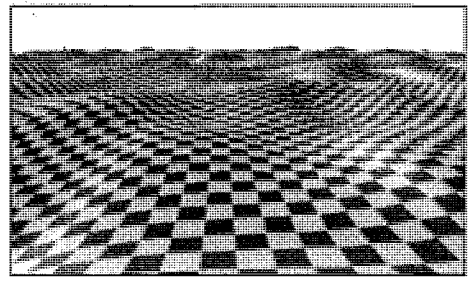


Свет, уходящий с поверхности в конкретной точке в сторону наблюдателя, представляет собой сумму компонентов, умноженных на коэффициент, связанный с материалом и цветом поверхности в данной точке. К таковым компонентам относятся:

- свет, пришедший с обратной стороны поверхности, то есть преломленный свет (*Refracted*);
- свет, равномерно рассеиваемый поверхностью (*Diffuse*);
- зеркально отраженный свет (*Reflected*);
- блики, то есть отраженный свет источников (*Specular*);
- собственное свечение поверхности (*Self Illumination*).

Свойства поверхности описываются в создаваемых массивах текстур (двух- или трехмерных). Таким образом, в массиве содержатся данные о степени прозрачности

материала; коэффициенте преломления; коэффициентах смещения компонентов (их список указан выше); цвете в каждой точке, цвете блика, его ширине и резкости; цвете рассеянного (фонового) освещения; локальных отклонениях векторов от нормали (то есть учитывается шероховатость поверхности).



Следующим этапом является наложение («проектирование») текстур на определенные участки каркаса объекта. При этом необходимо учитывать их взаимное влияние на границах примитивов. Проектирование материалов на объект — задача трудно формализуемая, она сродни художественному процессу и требует от исполнителя хотя бы минимальных творческих способностей.

Из всех параметров пространства, в котором действует создаваемый объект, с точки зрения визуализации самым важным является определение источников света. В трехмерной графике принято использовать виртуальные эквиваленты физических источников.

- Аналогом равномерного светового фона служит так называемый *растворенный свет* (*Ambient Light*). Он не имеет геометрических параметров и характеризуется только цветом и интенсивностью. Пример в природе — естественная освещенность вне видимости Солнца и Луны.
- Удаленный не точечный источник называют *удаленным светом* (*Distant Light*). Ему присваиваются конкретные геометрические параметры (координаты). Аналог в природе — Солнце.
- Точечный источник света (*Point Light Source*) равномерно испускает свет во всех направлениях и также имеет координаты. Аналог в технике — электрическая лампочка.
- Направленный источник света (*Direct Light Source*) кроме местоположения характеризуется направлением светового потока, углами раствора полного конуса света и его наиболее яркого пятна. Аналог в технике — прожектор.

После завершения конструирования и визуализации объекта приступают к его «оживлению», то есть заданию параметров движения. Компьютерная анимация базируется на ключевых кадрах. В первом кадре объект выставляется в исходное положение. Через определенный промежуток (например, в восьмом кадре) задается новое положение объекта и так далее до конечного положения. Промежуточные значения вычисляет программа по специальному алгоритму. При этом происходит не просто линейная аппроксимация, а плавное изменение положения опорных точек объекта в соответствии с заданными условиями (рис. 15.7).

Эти условия определяются иерархией объектов (то есть законами их взаимодействия между собой), разрешенными плоскостями движения, предельными углами поворотов, величинами ускорений и скоростей. Такой подход называют методом

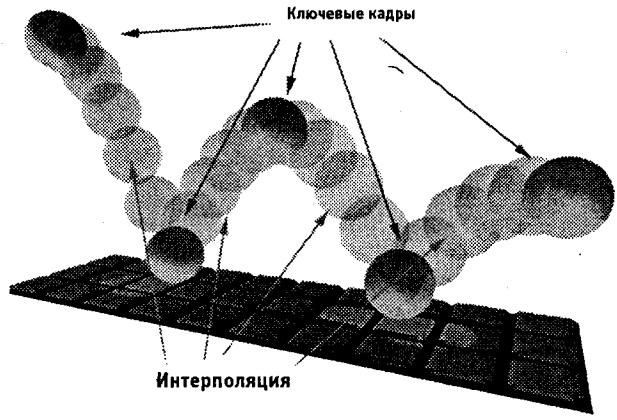


Рис. 15.7. Построение видеоряда по ключевым кадрам

инверсной кинематики движения. Он хорошо работает при моделировании механических устройств. В случае с имитацией живых объектов используют так называемые *скелетные модели*. То есть создается некий каркас, подвижный в точках, характерных для моделируемого объекта. Движения точек просчитываются предыдущим методом. Затем на каркас накладывается оболочка, состоящая из смоделированных поверхностей, для которых каркас является набором контрольных точек, то есть создается *каркасная модель*. Каркасная модель визуализируется наложением поверхностных текстур с учетом условий освещения. В ходе перемещения объекта получается весьма правдоподобная имитация движений живых существ.

Наиболее совершенный метод анимации заключается в фиксации реальных движений физического объекта. Например, на человеке закрепляют в контрольных точках яркие источники света и снимают заданное движение на видео- или кинопленку. Затем координаты точек по кадрам переводят с пленки в компьютер и присваивают соответствующим опорным точкам каркасной модели. В результате движения имитируемого объекта практически неотличимы от живого прототипа.

Процесс расчета реалистичных изображений называют *рендерингом* (визуализацией). Большинство современных программ рендеринга основаны на *методе обратной трассировки лучей* (*Backway Ray Tracing*). Его суть заключается в следующем.

1. Из точки наблюдения сцены посылается в пространство виртуальный луч, по траектории которого должно прийти изображение в точку наблюдения.
2. Для определения параметров приходящего луча все объекты сцены проверяются на пересечение с траекторией наблюдения. Если пересечения не происходит, считается, что луч попал в фон сцены и приходящая информация определяется только параметрами фона. Если траектория пересекается с объектом, то в точке соприкосновения рассчитывается свет, уходящий в точку наблюдения в соответствии с параметрами материала.

3. Сначала просчитывается преломленный и отраженный свет, затем проверяется видимость из точки пересечения всех источников света и интенсивность светового потока. Также вычисляются наличие, резкость и ширина бликов от каждого источника света.
4. Полученные в результате итоговые значения цвета и интенсивности обрабатываются с учетом траектории луча и параметров атмосферы, и присваиваются точке объекта как значения визуализации для наблюдателя. Затем процесс повторяется для всех элементов сцены. С целью упрощения расчетов пересечение проверяют не для каждой точки, а для примитива в целом. Иногда вокруг объекта создают простую виртуальную геометрическую фигуру (параллелепипед, шар), расчет пересечений для объекта выполняют только при пересечении траектории наблюдения с фигурой в целом.

Применение сложных математических моделей позволяет имитировать такие физические эффекты, как взрывы, дождь, огонь, дым, туман. Существуют методы расчета процедурных эффектов (*Procedural Effects*) и взаимодействия систем частиц (*Particle System*). Однако их применение в полном объеме требует громадных вычислительных ресурсов, и потому в персональных компьютерах обычно используют упрощенные варианты. По завершении рендеринга компьютерную трехмерную анимацию используют либо как самостоятельный продукт, либо в качестве отдельных частей или кадров готового продукта (рис. 15.8).

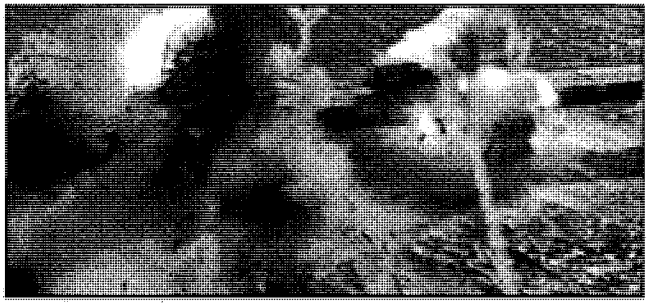


Рис. 15.8. Моделирование взрыва с помощью систем частиц

Особую область трехмерного моделирования в режиме реального времени составляют тренажеры технических средств — автомобилей, судов, летательных и космических аппаратов. В них необходимо очень точно реализовывать технические параметры объектов и свойства окружающей физической среды. В более простых вариантах, например при обучении вождению наземных транспортных средств, тренажеры реализуют на персональных компьютерах.

Самые совершенные на сегодняшний день устройства созданы для обучения пилотированию космических кораблей и военных летательных аппаратов. Моделированием и визуализацией объектов в таких тренажерах заняты несколько специализированных графических станций, построенных на мощных *RISC*-процессорах и скоростных видеоадаптерах с аппаратными ускорителями трехмерной графики. Общее управление системой и просчет сценариев взаимодействия возложены на

суперкомпьютер, состоящий из десятков и сотен процессоров. Стоимость таких комплексов выражается девятизначными цифрами, но их применение окупается достаточно быстро, так как обучение на реальных аппаратах в десятки раз дороже.

Программные средства обработки трехмерной графики

На персональных компьютерах основную долю рынка программных средств обработки трехмерной графики занимают три пакета. Эффективней всего они работают на самых мощных машинах (в двух- или четырехпроцессорных конфигурациях *Pentium* 4 и *Xeon*) под управлением операционной системы *Windows*.

Программа создания и обработки трехмерной графики *3D Studio Max* фирмы *Kinetix* изначально создавалась для платформы *Windows*. Этот пакет считается «полупрофессиональным». Однако его средств вполне хватает для разработки качественных трехмерных изображений объектов неживой природы (рис. 15.9). Отличительными особенностями пакета являются поддержка большого числа аппаратных ускорителей трехмерной графики, мощные световые эффекты, большое число дополнений, созданных сторонними фирмами. Сравнительная нетребовательность к аппаратным ресурсам позволяет работать даже на компьютерах среднего уровня. Вместе с тем по средствам моделирования и анимации пакет *3D Studio Max* уступает более развитым программным средствам.

Рис. 15.9. Трехмерное моделирование ландшафта средствами 3D Studio Max

Программа *Softimage 3D* компании *Microsoft* изначально создавалась для рабочих станций *SGI* и лишь позднее была конвертирована под операционную систему *Windows*. Программу отличают богатые возможности моделирования, наличие большого числа регулируемых физических и кинематографических параметров. Для рендеринга применяется качественный и достаточно быстрый модуль *Mental Ray*. Существует множество дополнений, выпущенных «третьими» фирмами, значительно расширяющих функции пакета. Эта программа считается стандартом «де-факто» в мире специализированных графических станций *SGI*, а на платформе *IBM PC* выглядит несколько тяжеловато и требует мощных аппаратных ресурсов.

Наиболее революционной с точки зрения интерфейса и возможностей является программа *Maya*, разработанная консорциумом известных компаний (*Alias Wavefront*). Пакет существует в вариантах для разных операционных систем, в том числе и *Windows*. Он имеет модульное построение и включает следующие блоки.

- *Base* — содержит ядро программы. Обеспечивает поддержку основных инструментов моделирования, инверсной кинематики, обработки звука, имитации физических твердых тел, захвата движения, рендеринга и основных наборов эффектов.
- *Maya F/X* — набор дополнительных модулей, поддерживающих эффекты обработки систем частиц и моделирования физики взаимодействия мягких тел.
- *Maya Power Modeler* — в основном содержит мощные средства полигонального и сплайнового моделирования объектов.
- *Maya Artisan* — наиболее передовой модуль, позволяющий обрабатывать виртуальные модели методами, характерными для реальной работы скульпторов и художников. Позволяет, к примеру, рисовать по поверхности объекта «кистями», сглаживать поверхности или делать их более шероховатыми «скульптурными резцами».
- *Maya Cloth* — предназначен для моделирования одежды.
- *Maya Fur* — модуль для имитации поверхностей, покрытых шерстью или мехом (рис. 15.10).

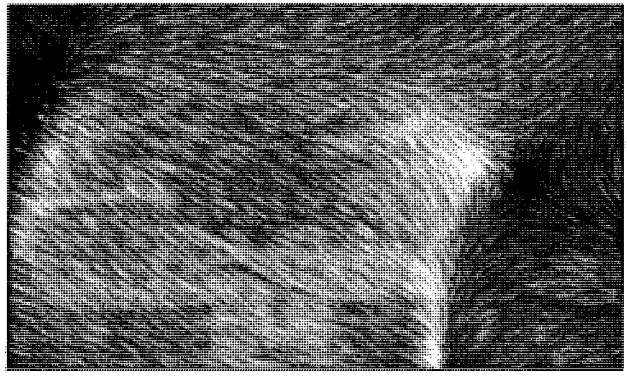


Рис. 15.10. Моделирование меховой поверхности средствами пакета Maya

- *Maya Live* — сценарный модуль, обеспечивающий сопряжение реальных съемок (на «натуре») с компьютерной анимацией.

Инструментарий *Maya* сведен в четыре группы: Animation (анимация), Modeling (моделирование), Dynamic (физическое моделирование), Rendering (визуализация). Удобный настраиваемый интерфейс выполнен в соответствии с современными требованиями. На сегодняшний день *Maya* является наиболее передовым пакетом в классе средств создания и обработки трехмерной графики для персональных компьютеров.

15.2. Представление графических данных

Форматы графических данных

В компьютерной графике применяют по меньшей мере три десятка форматов файлов для хранения изображений. Но лишь часть из них стала стандартом «де-факто» и применяется в подавляющем большинстве программ. Как правило, несовместимые форматы имеют файлы растровых, векторных, трехмерных изображений, хотя существуют форматы, позволяющие хранить данные разных классов. Многие приложения ориентированы на собственные «специфические» форматы, перенос их файлов в другие программы вынуждает использовать специальные фильтры или экспортировать изображения в «стандартный» формат.

TIFF (Tagged Image File Format). Формат предназначен для хранения растровых изображений высокого качества (расширение имени файла .TIF). Относится к числу широко распространенных, отличается переносимостью между платформами (*IBM PC* и *Apple Macintosh*), обеспечен поддержкой со стороны большинства графических, верстальных и дизайнерских программ. Предусматривает широкий диапазон цветового охвата — от монохромного черно-белого до 32-разрядной модели цветоделения *CMYK*. Начиная с версии 6.0 в формате *TIFF* можно хранить сведения о масках (контурах обтравки) изображений. Для уменьшения размера файла применяется встроенный алгоритм сжатия *LZW*.

PSD (PhotoShop Document). Собственный формат программы *Adobe Photoshop* (расширение имени файла .PSD), один из наиболее мощных по возможностям хранения растровой графической информации. Позволяет запоминать параметры слоев, каналов, степени прозрачности, множества масок. Поддерживаются 48-разрядное кодирование цвета, цветоделение и различные цветовые модели. Основной недостаток выражен в том, что отсутствие эффективного алгоритма сжатия информации приводит к большому объему файлов.

PCX. Формат появился как формат хранения растровых данных программы *PC PaintBrush* фирмы *Z-Soft* и в свое время был одним из наиболее распространенных (расширение имени файла .PCX). Отсутствие возможности хранить цветоделенные изображения, недостаточность цветовых моделей и другие ограничения привели к утрате популярности формата. К настоящему времени устарел.

PhotoCD. Формат разработан фирмой *Kodak* для хранения цифровых растровых изображений высокого качества (расширение имени файла .PCD). Сам формат хранения данных в файле называется *Image Pac*. Файл имеет внутреннюю структуру, обеспечивающую хранение изображения с фиксированными величинами разрешений, и потому размеры любых файлов лишь незначительно отличаются друг от друга и находятся в диапазоне 4–5 Мбайт. Каждому разрешению присвоен собственный уровень, отсчитываемый от так называемого базового (*Base*), составляющего 512×768 точек. Всего в файле пять уровней — от *Base*/16 (128×192 точек) до *Base*16 (2048×3072 точек). При первичном сжатии исходного изображения применяется метод субдискретизации, практически без потери качества. Затем вычисляются разности *Base* – *Base*×4 и *Base*×4 – *Base*×16. Итоговый результат записы-

вается в файл. Чтобы воспроизвести информацию с высоким разрешением, производится обратное преобразование. Для хранения информации о цвете использована цветовая модель *YCC*.

Windows Bitmap. Формат хранения растровых изображений в операционной системе *Windows* (расширение имени файла .BMP). Соответственно, поддерживается всеми приложениями, работающими в этой среде.

JPEG (Joint Photographic Experts Group). Формат предназначен для хранения растровых изображений (расширение имени файла .JPG). Позволяет регулировать соотношение между степенью сжатия файла и качеством изображения. Применяемые методы сжатия основаны на удалении «избыточной» информации, поэтому формат рекомендуют использовать только для электронных публикаций.

GIF (Graphics Interchange Format). Стандартизирован в 1987 году как средство хранения сжатых изображений с фиксированным (256) количеством цветов (расширение имени файла .GIF). Получил популярность в Интернете благодаря высокой степени сжатия. Последняя версия формата *GIF89a* позволяет выполнять чересстрочную загрузку изображений и создавать рисунки с прозрачным фоном. Ограниченные возможности по количеству цветов обусловливают его применение исключительно в электронных публикациях.

PNG (Portable Network Graphics). Сравнительно новый (1995 год) формат хранения изображений, предназначенный для их публикации в Интернете (расширение имени файла .PNG). Создавался как замена для форматов *GIF* и *JPEG*. Поддерживаются три типа изображений — цветные с глубиной 8 или 24 бита и черно-белое с градацией 256 оттенков серого. Сжатие информации происходит практически без потерь, предусмотрены 254 уровня альфа-канала, чересстрочная развертка. Массового распространения так и не получил.

WMF (Windows MetaFile). Формат хранения векторных изображений операционной системы *Windows* (расширение имени файла .WMF). По определению поддерживается всеми приложениями этой системы. Однако отсутствие средств для работы со стандартизированными цветовыми палитрами, принятыми в полиграфии, и другие недостатки ограничивают его применение.

EPS (Encapsulated PostScript). Формат описания как векторных, так и растровых изображений на языке *PostScript* фирмы *Adobe*, фактическом стандарте в области допечатных процессов и полиграфии (расширение имени файла .EPS). Так как язык *PostScript* является универсальным, в файле могут одновременно храниться векторная и растровая графика, шрифты, контуры обтравки (маски), параметры калибровки оборудования, цветовые профили. Для отображения на экране векторного содержимого используется формат *WMF*, а растрового — *TIFF*. Но экранная копия лишь в общих чертах отображает реальное изображение, что является существенным недостатком *EPS*. Действительное изображение можно увидеть лишь на выходе выводного устройства, с помощью специальных программ просмотра или после преобразования файла в формат *PDF* в приложениях *Acrobat Reader*, *Acrobat Exchange*.

PDF (Portable Document Format). Формат описания документов, разработанный фирмой *Adobe* (расширение имени файла .PDF). Хотя этот формат в основном предназначен для хранения документа целиком, его впечатляющие возможности позволяют обеспечить эффективное представление изображений. Формат является аппаратно-независимым, поэтому вывод изображений допустим на любых устройствах — от экрана монитора до фотоэкспонирующего устройства. Мощный алгоритм сжатия со средствами управления итоговым разрешением изображения обеспечивает компактность файлов при высоком качестве иллюстраций.

Понятие цвета

Цвет чрезвычайно важен в компьютерной графике как средство усиления зрительного впечатления и повышения информационной насыщенности изображения. Ощущение цвета формируется человеческим мозгом в результате анализа светового потока, попадающего на сетчатку глаза от излучающих или отражающих объектов. Считается, что цветовые рецепторы (колбочки) подразделяются на три группы, каждая из которых воспринимает только единственный цвет — красный, зеленый или синий. Нарушения в работе любой из групп приводит к явлению *дальтонизма* — искаженного восприятия цвета.

Световой поток формируется излучениями, представляющими собой комбинацию трех «чистых» спектральных цветов (красный, зеленый, синий — КЗС) и их производных (в англоязычной литературе используют аббревиатуру *RGB — Red, Green, Blue*). Для излучающих объектов характерно *аддитивное цветовоспроизведение* (световые излучения суммируются), для отражающих объектов — *субтрактивное цветовоспроизведение* (световые излучения вычитаются). Примером объекта первого типа является электронно-лучевая трубка монитора, второго типа — полиграфический отпечаток.

Физические характеристики светового потока определяются параметрами *мощности, яркости* и *освещенности*. Визуальные параметры ощущения цвета характеризуются *светлотой*, то есть различимостью участков, сильнее или слабее отражающих свет. Минимальную разницу между яркостью различимых по светлоте объектов называют *порогом*. Величина порога пропорциональна логарифму отношения яркостей. Последовательность оптических характеристик объекта (расположенная по возрастанию или убыванию), выраженная в оптических плотностях или логарифмах яркостей, составляет *градацию* и является важнейшим инструментом для анализа и обработки изображения.

Для точного цветовоспроизведения изображения на экране монитора важным является понятие *цветовой температуры*. В классической физике считается, что любое тело с температурой, отличной от 0 градусов *по шкале Кельвина*, испускает излучение. С повышением температуры спектр излучения смещается от инфракрасного до ультрафиолетового диапазона, проходя через оптический.

Для идеального черного тела легко находится зависимость между длиной волны излучения и температурой тела. На основе этого закона, например, была дистанционно вычислена температура Солнца — около 6500 К. Для целей правильного цветовоспроизведения характерна обратная задача. То есть монитор с выставлен-

ной цветовой температурой 6500 К должен максимально точно воспроизвести спектр излучения идеального черного тела, нагретого до такой же степени. Таким образом, стандартные значения цветовых температур используют в качестве всеобщего эталона, обеспечивающего одинаковое цветовоспроизведение на разных излучающих устройствах.

На практике зрение человека непрерывно подстраивается под спектр, характерный для цветовой температуры источника излучения. Например, на улице в яркий солнечный день цветовая температура составляет около 7000 К. Если с улицы зайти в помещение, освещенное только лампами накаливания (цветовая температура около 2800 К), то в первый момент свет ламп покажется желтым, белый лист бумаги тоже приобретет желтый оттенок. Затем происходит адаптация зрения к новому соотношению КЗС, характерному для цветовой температуры 2800 К, и свет лампы и лист бумаги начинают восприниматься как белые.

Насыщенность цвета показывает, насколько данный цвет отличается от монохроматического («чистого») излучения того же цветового тона. В компьютерной графике за единицу принимается насыщенность цветов спектральных излучений.

Ахроматические цвета (белый, серый, черный) характеризуется только светлотой. *Хроматические цвета* имеют параметры насыщенности, светлоты и цветового тона.

Способы описания цвета

В компьютерной графике применяют понятие *цветового разрешения* (другое название — *глубина цвета*). Оно определяет метод кодирования цветовой информации для ее воспроизведения на экране монитора. Для отображения черно-белого изображения достаточно двух бит (белый и черный цвета). Восьмиразрядное кодирование позволяет отобразить 256 градаций цветового тона. Два байта (16 бит) определяют 65 536 оттенков (такой режим называют *High Color*). При 24-разрядном способе кодирования возможно определить более 16,5 миллионов цветов (режим называют *True Color*).

С практической точки зрения цветовому разрешению монитора близко понятие *цветового охвата*. Под ним подразумевается диапазон цветов, который можно воспроизвести с помощью того или иного устройства вывода (монитор, принтер, печатная машина и прочие).

В соответствии с принципами формирования изображения аддитивным или субтрактивным методами разработаны способы разделения цветового оттенка на составляющие компоненты, называемые *цветовыми моделями*. В компьютерной графике в основном применяют модели *RGB* и *HSB* (для создания и обработки аддитивных изображений) и *CMYK* (для печати копии изображения на полиграфическом оборудовании).

Цветовые модели расположены в трехмерной системе координат, образующей *цветовое пространство*, так как из *законов Грассмана* следует, что цвет можно выразить точкой в трехмерном пространстве.

Первый закон Грассмана (закон трехмерности). *Любой цвет однозначно выражается тремя составляющими, если они линейно независимы. Линейная независи-*

мость заключается в невозможности получить любой из этих трех цветов сложением двух остальных.

Второй закон Грассмана (закон непрерывности). *При непрерывном изменении излучения цвет смеси также меняется непрерывно. Не существует такого цвета, к которому нельзя было бы подобрать бесконечно близкий.*

Третий закон Грассмана (закон аддитивности). *Цвет смеси излучений зависит только от их цвета, но не спектрального состава.* То есть цвет (C) смеси выражается суммой цветовых уравнений излучений:

$$C_1 = R_1R + G_1G + B_1B;$$

$$C_2 = R_2R + G_2G + B_2B;$$

$$C_n = R_nR + G_nG + B_nB;$$

$$C_{\text{сумм}} = (R_1 + R_2 + ... + R_n)\,R + (G_1 + G_2 + ... + G_n)\,G + (B_1 + B_2 + ... + B_n)\,B$$

Таким образом, прямоугольная трехмерная координатная система цветового пространства для аддитивного способа формирования изображения имеет точку начала координат, соответствующую абсолютно черному цвету (цветовое излучение отсутствует), и три оси координат, соответствующих основным цветам. Любой цвет (C) может быть выражен в цветовом пространстве вектором, который описывается уравнением:

$$\overrightarrow{C_n} = R_n\overrightarrow{R} + G_n\overrightarrow{G} + B_n\overrightarrow{B};$$

которое практически идентично уравнению свободного вектора в пространстве, рассматриваемому в векторной алгебре. Направление вектора характеризует цветность, а его модуль выражает яркость.

Так как величина излучения основных цветов является основой цветовой модели, ее максимальное значение принято считать за единицу. Тогда в трехмерном цветовом пространстве можно построить *плоскость единичных цветов*, образованную *треугольником цветности*. Каждой точке плоскости единичных цветов соответствует след цветового вектора, пронизывающего ее в этой точке (рис. 15.11). Следовательно, цветность любого излучения может быть представлена единственной точкой внутри треугольника цветности, в вершинах которого находятся точки основных цветов. То есть положение точки любого цвета можно задать двумя координатами, а третья легко находится по двум другим.

Если на плоскости единичных цветов указать значения координат, соответствующих реальным спектральным излучениям оптического диапазона (от 380 до 700 нм), и соединить их кривой, то мы получим линию, являющуюся геометрическим местом точек цветности монохроматических излучений, называемую *локусом*. Внутри локуса находятся все реальные цвета (рис. 15.12).

Чтобы избежать отрицательных значений координат, была выбрана колориметрическая система *XYZ*, полученная путем пересчета из *RGB*. В этой системе точке белого соответствуют координаты (0,33; 0,33). Колориметрическая система *XYZ* является универсальной, в ней можно выразить цветовой охват как аддитивных, так и субтрактивных источников цвета. Для аддитивных источников цветовой охват

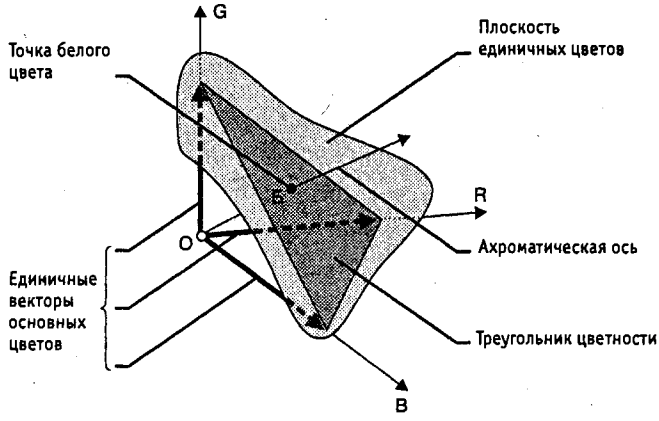


Рис. 15.11. Плоскость единичных цветов

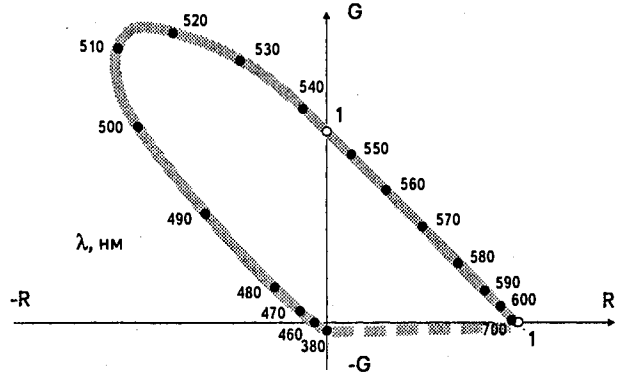


Рис. 15.12. Цветовой локус

выражается треугольником с координатами вершин, соответствующими излучению основных цветов *R*, *G*, *B*.

Для субтрактивных источников (полученных в процессе печати красками, чернилами, красителями) используется модель *CMYK*, поэтому цветовой охват описывается шестиугольником, когда помимо точек синтеза основной триады (желтая, пурпурная, голубая) добавляются точки попарных наложений, соответствующие основным цветам: желтая + голубая = зеленая, желтая + пурпурная = красная, голубая + пурпурная = синяя (рис. 15.13).

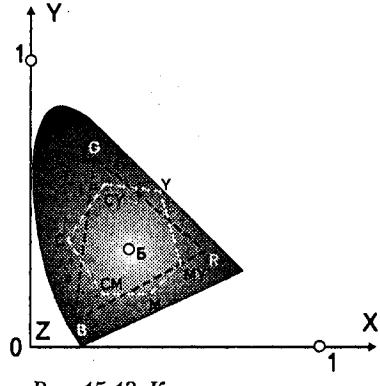


Рис. 15.13. Колориметрическая система XYZ

Цветовая модель CIE Lab

В 1920 году была разработана цветовая пространственная модель *CIE Lab* (*Communication Internationale de l'Eclairage — международная комиссия по освещению*; *L*, *a*, *b* — обозначения осей координат в этой системе). Система является аппаратно независимой и потому часто применяется для переноса данных между устройствами. В модели *CIE Lab* любой цвет определяется светлотой (*L*) и хроматическими компонентами: параметром *a*, изменяющимся в диапазоне от зеленого до красного, и параметром *b*, изменяющимся в диапазоне от синего до желтого. Цветовой охват модели *CIE Lab* значительно превосходит возможности мониторов и печатных устройств, поэтому перед выводом изображения, представленного в этой модели, его приходится преобразовывать. Данная модель была разработана для согласования цветных фотохимических процессов с полиграфическими. Сегодня она является принятым по умолчанию стандартом для программы *Adobe Photoshop*.

Цветовая модель RGB

Цветовая модель *RGB* является аддитивной, то есть любой цвет представляет собой сочетание в различной пропорции трех основных цветов — красного (*Red*), зеленого (*Green*), синего (*Blue*). Она служит основой при создании и обработке компьютерной графики, предназначенной для электронного воспроизведения (на мониторе, телевизоре). При наложении одного компонента основного цвета на другой яркость суммарного излучения увеличивается. Совмещение трех компонентов одинаковой яркости дает ахроматический серый цвет, который при увеличении яркости приближается к белому цвету. При 256 градационных уровнях тона черному цвету соответствуют нулевые значения *RGB*, а белому — максимальные, с координатами (255, 255, 255).

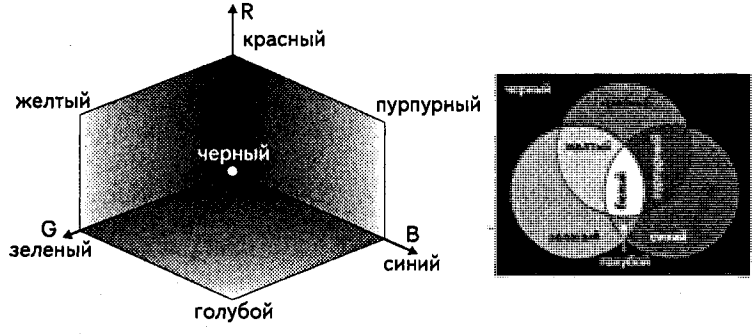


Рис. 15.14. Аддитивная цветовая модель RGB

Цветовая модель HSB

Цветовая модель *HSB* разработана с максимальным учетом особенностей восприятия цвета человеком. Она построена на основе цветового круга Манселла. Цвет описывается тремя компонентами: оттенком (*Hue*), насыщенностью (*Saturation*) и яркостью (*Brightness*). Значение цвета выбирается как вектор, исходящий из центра окружности. Точка в центре соответствует белому цвету, а точки по периметру

окружности — чистым спектральным цветам. Направление вектора задается в градусах и определяет цветовой оттенок. Длина вектора определяет насыщенность цвета. На отдельной оси, называемой *ахроматической*, задается яркость, при этом нулевая точка соответствует черному цвету. Цветовой охват модели *HSB* перекрывает все известные значения реальных цветов.

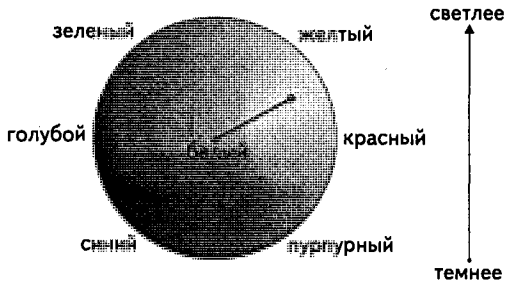


Рис. 15.15. Цветовая модель HSB

Модель *HSB* принято использовать при создании изображений на компьютере с имитацией приемов работы и инструментария художников. Существуют специальные программы, имитирующие кисти, перья, карандаши. Обеспечивается имитация работы с красками и различными полотнами. После создания изображения его рекомендуется преобразовать в другую цветовую модель, в зависимости от предполагаемого способа публикации.

Цветовая модель CMYK, цветоделение

Цветовая модель *CMYK* относится к субтрактивным, и ее используют при подготовке публикаций к печати. Цветовыми компонентами *CMY* служат цвета, полученные вычитанием основных из белого:

голубой (*cyan*) = белый − красный = зеленый + синий;

пурпурный (*magenta*) = белый − зеленый = красный + синий;

желтый (*yellow*) = белый − синий = красный + зеленый.

Такой метод соответствует физической сущности восприятия отраженных от печатных оригиналов лучей. Голубой, пурпурный и желтый цвета называются *дополнительными*, потому что они дополняют основные цвета до белого. Отсюда вытекает и главная проблема цветовой модели *CMY* — наложение друг на друга дополнительных цветов на практике не дает чистого черного цвета. Поэтому в цветовую модель был включен компонент чистого черного цвета. Так появилась четвертая буква в аббревиатуре цветовой модели *CMYK* (*Cyan*, *Magenta*, *Yellow*, *blacK*).

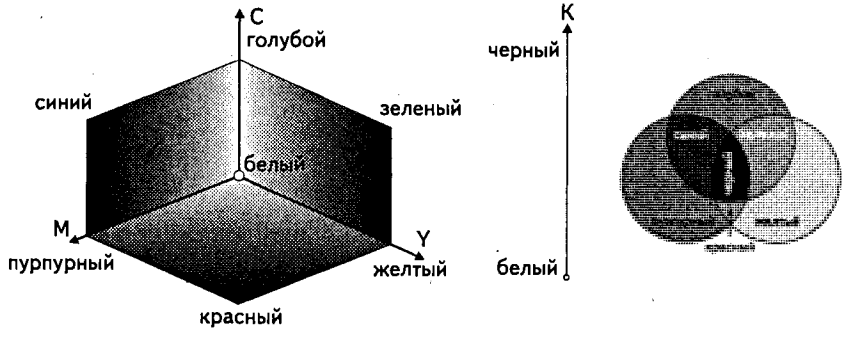


Рис. 15.16. Цветовая модель CMYK

Для печати на полиграфическом оборудовании цветное компьютерное изображение необходимо разделить на составляющие, соответствующие компонентам цветовой модели *CMYK*. Этот процесс называют *цветоделением*. В итоге получают четыре отдельных изображения, содержащих одноцветное содержимое каждого компонента в оригинале. Затем в типографии с форм, созданных на основе цветоделенных пленок, печатают многоцветное изображение, получаемое наложением цветов *CMYK*.

Цветовая палитра

Электронная *цветовая палитра* в компьютерной графике по предназначению подобна палитре художника, но включает гораздо большее число цветов. Электронная палитра состоит из определенного числа ячеек, каждая из которых содержит отдельный цветовой тон. Конкретная цветовая палитра соотносится с определенной цветовой моделью, так как ее цвета созданы на основе цветового пространства этой модели. Но если в цветовой модели возможно воспроизвести любой из описываемых ею цветов, цветовая палитра содержит ограниченный набор цветов, называемых *стандартными*.

Примером стандартных цветовых палитр являются наборы фирмы *Pantone*, ориентированные на полиграфическую публикацию изображений. Программы создания и обработки компьютерной графики, как правило, предоставляют на выбор несколько цветовых палитр в цветовых моделях *RGB*, *HSB*, *CIE Lab*, *CMYK*.

Состав цветовых палитр *RGB* зависит от выбранного цветового разрешения — 24, 16 или 8 бит. В последнем случае цветовая палитра называется *индексной*, потому что каждый цветовой оттенок кодируется одним числом, которое выражает не цвет пиксела, а *индекс* (номер) цвета. Таким образом, к файлу цветного изображения, созданного в индексной палитре, должна быть приложена сама палитра, так как программе обработки компьютерной графики неизвестно, какая именно палитра была использована.

Изображения, подготавливаемые для публикации в Интернете, принято создавать в так называемой *безопасной палитре* цветов. Она является вариантом рассмотренной выше индексной палитры. Но так как файлы изображений в *Web*-графике должны иметь минимальный размер, необходимо было отказаться от включения в их состав индексной палитры. Для этого была принята единая фиксированная палитра цветов, названная *«безопасной»*, то есть обеспечивающей правильное отображение цветов на любых устройствах (в программах), поддерживающих единую палитру. Безопасная палитра содержит всего 216 цветов, что связано с ограничениями, накладываемыми требованиями совместимости с компьютерами, не относящимися к классу *IBM PC*.

Системы управления цветом

При создании и обработке элементов компьютерной графики необходимо добиться, чтобы изображение выглядело практически одинаково на всех стадиях процесса, на любом устройстве отображения, при любом методе визуализации (аддитивном или субтрактивном). Иначе, чем больше переходных этапов будет содержать про-

цесс обработки, тем большие искажения будут вноситься в оригинал, и конечный результат может совершенно не удовлетворять даже минимальным требованиям к качеству. Для согласования цветов на всех стадиях обработки компьютерной графики применяют *системы управления цветом* (*Color Management System — CMS*).

Такие системы содержат набор объективных параметров, обязательных для всех устройств при обмене цветовыми данными. Универсальность *CMS* достигается введением трех типов переменных, каждая из которых управляет представлением цвета на своем уровне:

Цветовая гамма. Каждый тип устройства имеет свою *цветовую гамму*, область которой всегда меньше, чем цветовой охват практически любой цветовой модели. *CMS* управляет преобразованием цвета между различными цветовыми моделями с учетом цветовой гаммы конкретных устройств.

Профиль. Каждое устройство воспроизводит цвета особенным образом, что зависит от технических и программных решений, принятых изготовителем. Для согласования отображения цветов на различных устройствах они должны иметь собственный *профиль*, описывающий различия в представлении цвета между устройством и определенной цветовой моделью. Международным консорциумом по цвету (*International Color Consortium — ICC*) установлен промышленный стандарт на параметры описания характеристик воспроизведения цвета. Устройства, имеющие профиль *ICC*, напрямую управляются *CMS*. В противном случае возможна генерация профиля в некоторых системах *CMS*.

Калибровка. Даже устройства одной модели от одного производителя имеют отличия в реализации профиля *ICC*, обусловленные допусками при изготовлении компонентов, условиями эксплуатации, внешними помехами. Поэтому *CMS*, как правило, включает средства *калибровки*, то есть настройки конкретного экземпляра в соответствии с требованиями профиля *ICC* и фиксации неустранимых отклонений (с целью их программной компенсации). Средства калибровки могут быть аппаратно-программными и чисто программными. Сам процесс калибровки выполняется с периодичностью, установленной изготовителем, или автоматически, при выходе параметров *ICC* за границы допусков.

Не существует идеальной системы управления цветом, одинаково пригодной для всех устройств, одинаково работающей на всех платформах и во всех программных средах. Наиболее близко к идеалу подходят *CMS*, реализованные на уровне операционной системы. Впервые *CMS* под названием *ColorSync* в операционную систему встроила фирма *Apple*, что предопределило успех компьютеров *Macintosh* в сфере издательской деятельности, допечатной подготовки и полиграфии. В операционной системе *Windows XP* используется система *Image Color Management*.

Из *CMS*, являющихся внешними по отношению к операционной системе, наибольшее распространение получили программы фирм, давно работающих в области цветной фотографии, печати, цифровых графических технологий.

Agfa Foto Tune. Эта система управления цветом работает на платформах *Windows* и *Apple*. Включает множество профилей *ICC* для мониторов, цветных принтеров, сканеров, цифровых фотокамер, полиграфического оборудования. Имеются сред-

ства создания заказных профилей для устройств, не попавших в список. Преобразования между цветовыми профилями устройств (например, сканер — монитор) могут производиться напрямую, без промежуточного конвертирования в цветовую модель *CIE Lab* и обратно.

Kodak DayStar ColorMatch. Система предназначена для пользователей пакетов *Adobe Photoshop* и *QuarkXPress*. Отличается модульным построением, поэтому базовая поставка содержит ограниченное число профилей, а остальные необходимо приобретать дополнительно. Система имеет средства поддержки формата *Kodak PhotoCD* с учетом вывода изображений на фотопринтеры. Средства калибровки включают стандартный шаблон *IT8* для сканеров и устройство *Digital Colorimeter* для мониторов.

Практическое занятие

Упражнение 15.1. Расчет требуемых линиатур растра

15 мин

Рассчитать, какая линиатура потребуется для печати изображения на черно-белом принтере с разрешением 1200 *dpi* при требовании к качеству 100 уровней серого цвета. Можно ли распечатать на таком принтере изображение с 256 градациями тона?

1. Подставим исходные параметры в формулу $lpi = \frac{dpi}{\sqrt{N-1}}$:

$$lpi = \frac{1200}{\sqrt{100-1}} = \frac{1200}{\sqrt{99}} = \frac{1200}{9{,}94} = 120{,}72.$$

2. Так как линиатуру лучше представлять целым числом, округляем полученное значение в меньшую сторону и получаем значение 120 *lpi*.
3. Проверим, можно ли вывести изображение с 256 градациями тона:

$$lpi = \frac{1200}{\sqrt{256-1}} = \frac{1200}{\sqrt{255}} = \frac{1200}{15{,}96} = 75{,}18.$$

4. Округляем результат до ближайшего целого, получаем 75 *lpi*. Полученный оттиск может быть использован для размножения полиграфическими средствами с качеством, соответствующим газетному производству.

При требовании к уровню качества 100 градаций тона возможна печать с линиатурой 120 lpi, характерной для книжно-журнального производства. При воспроизведении оригинала с 256 градациями серого возможна печать с линиатурой 75 lpi, характерной для газетного производства.

Упражнение 15.2. Расчет требуемого разрешения оцифровки

15 мин

Рассчитать требуемое разрешение оцифровки цветного 35-мм слайда, если предполагается печать его цветной копии размером А3 альбомного формата с линиатурой растра 133 *lpi*. Справка: ширина листа А3 альбомной ориентации равна 420 мм.

1. Сначала определим коэффициент масштабирования:

$$420 : 35 = 12$$

2. Определим исходное разрешение оцифровки:

   $dpi = 133 \cdot 1{,}5 = 199{,}5$, округленно = 200 *dpi*

3. Вычисляем разрешение оцифровки с учетом масштабирования:

   $200 \cdot 12 = 2400$ *dpi*

4. Устанавливаем окончательное разрешение оцифровки с учетом угла поворота растра:

   $2400 \cdot 1{,}06 = 2540$ *dpi*

Для качественного полиграфического воспроизведения изображений с цветных слайдов необходимо использовать сканеры профессионального класса, обеспечивающие разрешение не ниже 2540 dpi.

Упражнение 15.3. Расчет угла поворота растра

15 мин

Вам поручено подготовить на компьютере цветоделенные изображения для вывода на фотоавтомате, с учетом печати двумя дополнительными применительно к модели *CMYK* красками — зеленой и синей. Указать углы поворота растра для всех пленок.

1. Выберем для модели *CMYK* традиционные углы поворота растра, которые и подставим в таблицу.
2. Зеленый цвет является в модели *CMYK* производным, то есть суммой желтого и голубого, поэтому прибавим стандартный угол поворота 45° к значению угла для желтого цвета:

   $90 + 45 = 135°$

3. Синий цвет также является в модели *CMYK* производным, то есть суммой голубого и пурпурного. Прибавим стандартный угол поворота к значению угла для голубого цвета:

   $105 + 45 = 150°$

Печатная форма для краски	Угол поворота растра, градусов
Голубой	105
Пурпурной	75
Желтой	90
Черной	45
Зеленой	135
Синей	150

В полиграфии возможна печать числом красок более четырех (из стандартного набора *CMYK*). Такой прием позволяет добиться на отпечатке ярких, сочных тонов именно того цвета, который необходимо подчеркнуть.

15.3. Средства для работы с растровой графикой

Программные средства создания растровых изображений

Среди программ, предназначенных для создания компьютерной двумерной живописи, самыми популярными считаются *Painter* компании *Fractal Design*, *FreeHand* компании *Macromedia* и *Fauve Matisse*. Пакет *Painter* обладает достаточно широким спектром средств рисования и работы с цветом. В частности, он моделирует различные инструменты (кисти, карандаш, перо, уголь, аэрограф и др.), позволяет имитировать материалы (акварель, масло, тушь), а также добиться эффекта натуральной среды. В свою очередь, последние версии программы *FreeHand* обладают богатыми средствами редактирования изображений и текста, содержат библиотеку спецэффектов и набор инструментов для работы с цветом, в том числе средства многоцветной градиентной заливки.

Среди программ для создания изображений на платформе *Macintosh* стоит отметить пакет для редактирования растровой живописи и изображений *PixelPaint Pro* компании *Pixel Resources*.

Среди программ компьютерной живописи для графических станций *Silicon Graphics* (*SGI*) особое место занимает пакет *StudioPaint 3D* компании *Alias Wavefront*, который позволяет рисовать различными инструментами («кистями») в режиме реального времени прямо на трехмерных моделях. Пакет работает с неограниченным количеством слоев изображения и предоставляет 30 уровней отмены предыдущего действия (*undo*), включает операции цветокоррекции и «сплайновые кисти», «мазок» которых можно редактировать по точкам как сплайновую кривую. *StudioPaint 3D* поддерживает планшет с чувствительным пером, что дает возможность художнику сделать традиционный эскиз от руки, а затем позволяет перенести рисунок в трехмерные пакеты для моделирования или анимации и построить по эскизу трехмерную модель.

Аппаратные средства получения растровых изображений

К аппаратным средствам получения цифровых растровых оригиналов в основном относятся *сканеры* и *цифровые фотокамеры*. Другие устройства, например цифровые видеокамеры, адаптеры захвата телевизионных кадров, в компьютерной графике играют чаще вспомогательную роль. Для создания изображений «от руки» предназначены *графические планшеты*, на которых рисуют специальным электронным пером.

Сканеры по способу восприятия изображения делятся на две группы: устройства с электронными фотоумножителями (ФЭУ) и устройства на приборах с зарядовой связью (ПЗС, английская аббревиатура *CCD*). Сканеры с фотоумножителями называют *барабанными* — внутри аппарата помещен прозрачный барабан, на который крепится оригинал (отражающий или просветный). Затем барабан начинает вращаться с большой скоростью. Сканирующая головка имеет мощный источник света с фокусированным лучом и ФЭУ, которые движутся вдоль продольной оси барабана. Отраженный или проходящий световой поток попадает на ФЭУ (обычно имеется по одному ФЭУ на каждый канал) через прецизионную зеркальную сис-

тему развертки. Накопленный ФЭУ заряд преобразуется в цифровое значение аналого-цифровым преобразователем высокой разрядности. Так как процесс до этого момента по сути аналоговый, удается добиться очень высоких значений динамического диапазона. То есть оригинал правильно оцифровывается и в светлых, и в темных участках. Выходное разрешение оригинала достигает 5000–6000 точек на дюйм. За совершенное качество приходится платить — барабанные сканеры чрезвычайно дорогостоящи и требовательны к условиям эксплуатации.

Прочие сканеры относятся к *устройствам на ПЗС*. В отличие от ФЭУ, приборы с зарядовой связью представляют собой фотоприемник, выполненный на кремниевых элементах, объединенных в линейку. Каждый светочувствительный элемент обладает способностью накапливать заряды пропорционально числу попавших на него фотонов. За время экспозиции возникает матрица зарядов, пропорциональных яркости исходного изображения. По вертикали развертка осуществляется передвижением либо всей линейки ПЗС с помощью шагового электродвигателя, либо перемещением оригинала.

Разрешающая способность определяется числом оптических элементов на единицу длины. В устройствах бытового класса это 300–600 элементов на дюйм, профессионального — 1200–3000. Программная интерполяция оптического разрешения никакого реального повышения качества оцифровки не дает. Динамический диапазон устройств на ПЗС ниже, чем у ФЭУ, потому что кремниевые элементы имеют худшее соотношение сигнал/шум.

В высокоточных сканерах на ПЗС дополнительно применяются: система зеркальной развертки по обоим координатам с компенсацией искажений по краям оригинала, несколько линеек ПЗС, стабильные по цветовой температуре осветительные лампы, многоразрядные цифро-аналоговые преобразователи, элементы, выполненные на *CMOS*-пластинах. Такие устройства по качеству оцифровки приближаются к барабанным сканерам, а по стоимости значительно доступнее.

Конструктивно барабанные сканеры выполняют с вертикальным или горизонтальным барабаном, съемным или несъемным. Сканеры на ПЗС бывают листовые, планшетные, проекционные, ручные и так называемые слайдовые (для сканирования оригиналов «на просвет»).

Для целей компьютерной графики важно не столько разрешение сканера (оно может не превышать 300 *dpi*), сколько хороший динамический диапазон. Для сканирования в отраженном свете желательно иметь динамический диапазон не ниже 2, «на просвет» — не ниже 3,5.

Основой *цифровых фотокамер* служит матрица ПЗС, состоящая из двумерного массива элементов. Для целей электронной публикации и непрофессионального применения достаточное число элементов на матрице около 1,5 миллионов. Полупрофессиональные камеры должны иметь разрешение матрицы не ниже 4 миллионов элементов, профессиональные аппараты — не менее 5 миллионов. Оцифрованные с их помощью изображения можно использовать для подготовки полиграфических публикаций. Оптическая система цифровых камер профессионального класса должна обеспечивать разрешение не ниже 110–120 пар линий на дюйм.

Графические планшеты представляют собой координатную двумерную электронную сетку, каждый элемент которой способен воспринимать и передавать ряд сигналов от электронного пера. К таковым сигналам относятся: координаты точки контакта пера с планшетом, сила нажима, угол наклона, скорость прохода (то есть время экспозиции) и ряд других. Затем за счет программного преобразования полученные данные отображаются на экране в виде линий, мазков и других художественных средств создания изображений. Обладая достаточным навыком работы с графическим планшетом, удается очень точно имитировать различную живописную технику — письмо маслом, рисунок углем, аэрографом, карандашом и т. д.

Программа обработки растровой графики Adobe Photoshop

В обширном классе программ для обработки растровой графики особое место занимает пакет *Photoshop* компании *Adobe*. По сути дела, сегодня он является стандартом в компьютерной графике, и все другие программы неизменно сравнивают именно с ним (рис. 15.17).

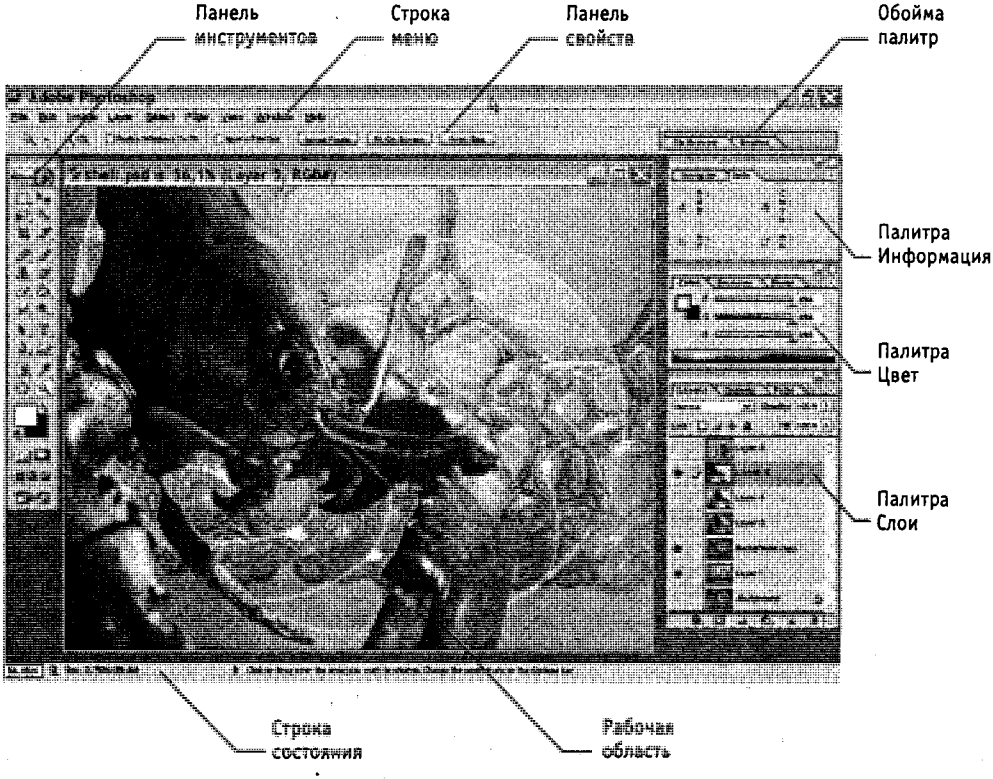


Рис. 15.17. Рабочее окно графического редактора Adobe Photoshop

Главные элементы управления программы Adobe Photoshop сосредоточены в строке меню, панели инструментов и панели свойств. Особую группу составляют диалоговые окна — *инструментальные палитры*. Далее мы рассмотрим функции перечисленных средств.

Первичное получение оригинала происходит через меню File (Файл) либо командой Open (Открыть), либо командой Import (Импорт). *Импортом* называют получение изображения от внешнего источника — сканера, цифровой фотокамеры или из документа Adobe PDF со встроенной графикой. Связь графического редактора с внешними устройствами обеспечивается через программный интерфейс *TWAIN*, устанавливающий стандарт на параметры обмена данными с источниками изображений.

Прежде чем начать операции с оригиналом изображения, следует уяснить его параметры. Для этого командой Image ▸ Image Size (Изображение ▸ Размер изображения) открывают диалоговое окно Image Size (Размер изображения). В группах Pixel Dimensions (Размерность) и Document Size (Размер печатного оттиска) приведены ширина и высота оригинала в пикселах и сантиметрах соответственно, а также разрешение (в пикселах на дюйм — *ppi*). От установленных значений зависят размер и качество изображения. Для целей электронной публикации лучше установить разрешение 72 *ppi*, для последующей распечатки выбирают разрешение, исходя из формулы $ppi = 1{,}5 \cdot lpi$, где *lpi* — линиатура растра, установленная на устройстве вывода. Для устройств, поддерживающих стохастическое ЧМ-растрирование, вместо *lpi* надо знать их разрешающую способность в *dpi*. Размер изображения лучше устанавливать в масштабе 1:1 по отношению к тому, что будет использовано в публикации, или несколько больше.

Панель Tools (Инструменты) (рис. 15.18) является одним из основных средств для работы с изображениями. Большинство инструментов, представленных на панели, имеют альтернативные варианты. Их значки помечены маленьким треугольником (разворачивающая кнопка). Если при нажатой кнопке мыши задержать указатель на таком значке, откроется линейка значков с вариантами инструмента.

Панель инструментов разделена на области, в которых сгруппированы средства для редактирования определенных свойств изображения и элементы управления некоторыми параметрами программы. Всего таких областей восемь: инструменты для работы с объектами, инструменты для рисования и ретуши, инструменты для создания новых объектов, вспомогательные инструменты, средства выбора цветов, средства управления представлением маски, средства управления интерфейсом программы, кнопка перехода в программу Adobe Image Ready.

Для работы с объектами предназначена группа значков, объединяющая инструменты Marquee (Область), Move (Перемещение), Lasso (Лассо), Magic Wand (Волшебная палочка), Crop (Обрезка) и Slice (Разделение на фрагменты). Инструментами Область и Лассо выделяют участок изображения, ограниченный геометрической фигурой. Инструмент Волшебная палочка осуществляет выборку области по принципу цветового (или ахроматического) совпадения в рамках границ охвата, установленных пользователем. Эти инструменты применяют для выполнения операций *обтравки* — обводки контуров объектов на изображении. Инструментом Перемещение передвигают выделенные области и копируют их. Инструмент Обрезка позволяет удалить области (поля) рисунка, а инструмент Разделение на фрагменты используется, когда большой рисунок готовится для отображения в Интернете. С помощью этого инструмента его можно автоматически разделить на фрагменты приемлемого размера, каждый из которых будет загружаться через сеть по отдельности.

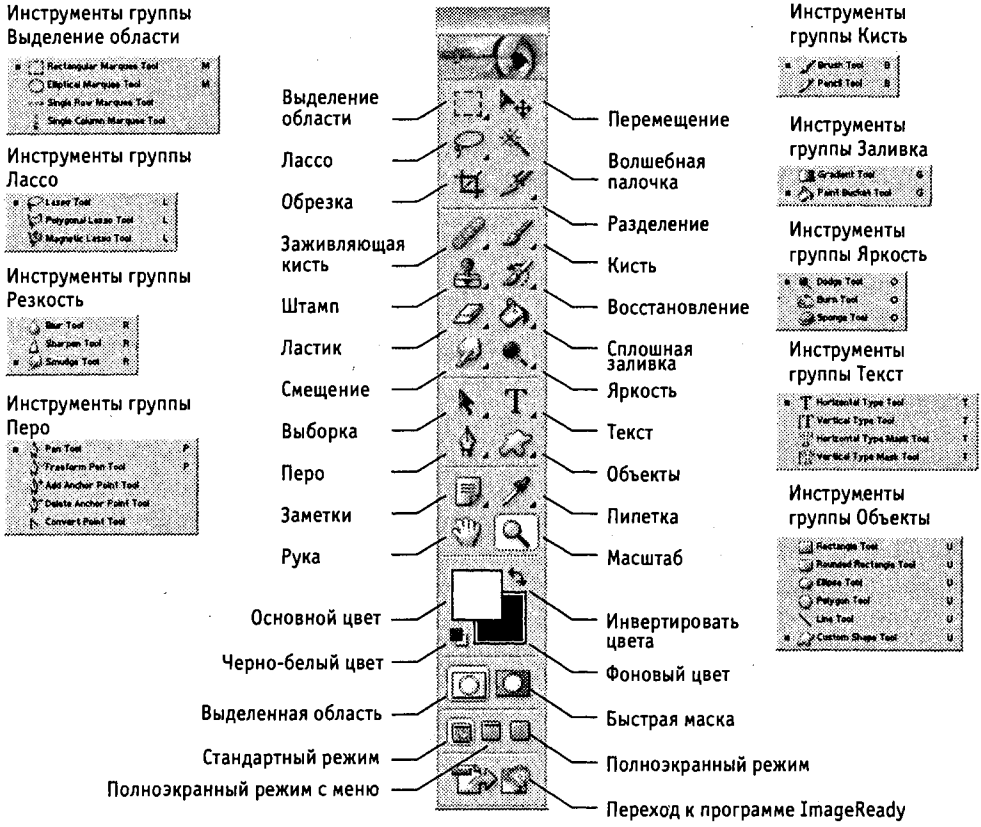


Рис. 15.18. Панель инструментов Adobe Photoshop 7.0

Следующая группа инструментов предназначена для рисования и ретуши. Она включает Healing Brush (Заживляющая кисть) и Patch (Заплатка), Brush (Кисть) и Pencil (Карандаш), Clone Stamp (Клонирующий штамп) и Pattern Stamp (Узорный штамп), History Brush (Восстанавливающая кисть) и Art History Brush (Художественная восстанавливающая кисть), Eraser (Ластик), Paint Bucket (Сплошная заливка) и Gradient (Градиентная заливка). Для ретуши также используют инструменты манипуляций с резкостью Blur (Размытие), Sharpen (Резкость), Smudge (Смещение) или яркостью Dodge (Осветлитель), Burn (Затемнитель), Sponge (Губка).

Инструмент Клонирующий штамп позволяет выполнять *набивку* — копирование выбранных участков изображения по каждому щелчку мыши. Узорный штамп в качестве наполнителя набивки использует узоры из палитры Pattern. Заживляющая кисть похожа по способу действия на Клонирующий штамп, однако приближает яркость и цвет копируемой области к параметрам пикселов целевой области. Инструмент Заплатка позволяет выбрать произвольную область-источник протягиванием мыши, а затем переместить ее в нужное место. При этом цвета и яркость области-источника приводятся в соответствие с параметрами пикселов целевой

области. Восстанавливающая кисть приводит параметры пикселов в области действия к исходному состоянию, которое они имели до редактирования.

С помощью инструмента Сплошная заливка выполняют заполнение выделенных участков изображения одним цветом. Плавный переход между цветами обеспечивает инструмент Градиентная заливка. Инструменты с альтернативным выбором Резкость/Размытие позволяют изменять эти параметры на отдельных участках изображения, а инструменты Осветлитель/Затемнитель/Губка служат для местной коррекции яркости и цветовой насыщенности.

Третья группа инструментов предназначена для создания новых объектов и редактирования существующих. Инструмент Pen (Перо) позволяет рисовать плавные криволинейные контуры. Инструментом Text (Текст) выполняют надписи. Инструменты для создания фигур и линий служат для рисования векторных объектов. Для выборки объектов целиком или узлов контура служат инструменты Path Selection (Выборка контура) и Direct Selection (Выборка узлов) соответственно.

Группа вспомогательных инструментов объединяет подсобные средства. Инструментом Hand (Рука) перемещают видимую область по изображению, а инструмент Zoom (Масштаб) предназначен для увеличения/уменьшения изображения в видимой области. Инструмент Eyedropper (Пипетка) служит для точного определения цвета в любой точке изображения и принятия его как образца. В этом же наборе размещен инструмент Measure (Измерительная линейка). Инструмент Notes (Заметки) позволяет привязать к рисунку текстовые замечания. Звуковые комментарии создают с помощью альтернативного инструмента Audio Annotation (Звуковые аннотации).

В нижней части инструментальной панели помещены средства для работы с цветом, масками, формой отображения элементов управления программы. Средство управления цветом показывает основные цвета фона и переднего плана. В левом нижнем углу расположен значок, щелчок на котором устанавливает цвета, принятые по умолчанию для фона и переднего плана. Элемент управления Mask (Маска) позволяет работать в режимах Standard (Стандартный) или Quick (Быстрая маска). Средство управления режимом отображения позволяет переключаться между Стандартным режимом, Расширенным (скрывается строка заголовка окна программы) и Полным (панель меню сворачивается и помещается в виде кнопки в верхней части панели инструментов). Последним элементом управления на панели инструментов является кнопка Jump to ImageReady (Переход к программе Image Ready).

Важнейшим средством управления параметрами инструментов является интерактивная Панель свойств. При выборе любого инструмента автоматически появляется соответствующая его свойствам палитра с необходимыми элементами управления.

Помимо Панели свойств, многими параметрами инструментов или изображения в целом «заведуют» *инструментальные палитры*, которые представляют собой диалоговые окна особого вида. Всего в программе *Adobe Photoshop* больше десяти инструментальных палитр.

Управление отображением палитр осуществляется из меню Window (Окно) с помощью флажков меню, соответствующих существующим палитрам. Неиспользуемые палитры можно также удалить с экрана щелчком на закрывающей кнопке. Щелч-

ком на сворачивающей кнопке палитру сокращают до размера строки с корешками вкладок. Справа под строкой заголовка окно палитры имеет кнопку, щелчок на которой открывает доступ к меню, содержащему команды работы с объектами палитры и настройки параметров палитры. Некоторые палитры имеют командные кнопки, раскрывающиеся списки, поля ввода и другие элементы управления. Назначение конкретного элемента управления поясняет всплывающая подсказка, появляющаяся при задержке указателя мыши на интересующем элементе.

Палитры можно перемещать по экрану методом перетаскивания. Новые палитры создают «сборкой» из имеющихся элементов. Для этого, подцепив указателем мыши корешок одной из вкладок палитры, его перетаскивают в окно другой палитры. Если вкладку разместить на свободном поле экрана, она преобразуется в независимую палитру.

Палитры можно размещать (методом перетаскивания за корешок вкладки) в специальной области Панели свойств, названной Palettes Well (Обойма палитр). В этой области отображаются только корешки вкладок, однако при активизации палитры ее окно демонстрируется целиком.

Палитра Brushes (Кисти) управляет настройкой параметров инструментов редактирования. Она имеет увеличенный размер, так что ее удобно держать в обойме палитр.

Палитра File Browser (Обозреватель файлов) предоставляет средства навигации по файловой системе и просмотра изображений различного формата.

Палитра Info (Инфо) обеспечивает информационную поддержку средств отображения. На ней представлены: текущие координаты указателя мыши, размер текущей выделенной области, цветовые параметры элемента изображения и другие данные.

Палитра Navigator (Навигатор) позволяет просмотреть различные фрагменты изображения и изменить масштаб просмотра. В окне палитры помещена миниатюра изображения с выделенной областью просмотра.

Палитра Color (Синтез) отображает цветовые значения текущих цветов переднего плана и фона. Ползунки на цветовой линейке соответствующей цветовой системы позволяют редактировать эти параметры.

Палитра Swatches (Каталог) содержит набор доступных цветов. Такой набор можно загрузить и отредактировать, добавляя и удаляя цвета. Цветовой тон переднего плана и фона выбирают из состава набора. В стандартном комплекте поставки программы предусмотрено несколько цветовых наборов, в основном компании *Pantone*.

Палитра Layers (Слои) служит для управления отображением всех слоев изображения, начиная с самого верхнего. Возможно определение параметров слоев, изменение их порядка, операции со слоями с применением разных методов.

Палитру Channels (Каналы) используют для выделения, создания, дублирования и удаления цветовых каналов, определения их параметров, изменения порядка, преобразования каналов в самостоятельные объекты и формирования совмещенных изображений из нескольких каналов.

Палитра Paths (Контуры) содержит список всех созданных контуров. Контуры можно использовать для обтравки — программа *Adobe Photoshop* позволяет преобразовать контур в границу выделенной области.

Палитра Actions (Операции) позволяет создавать *макрокоманды* — заданную последовательность операций с изображением. Макрокоманды можно записывать, выполнять, редактировать, удалять, сохранять в виде файлов.

Особую группу программных средств обработки изображений представляют *фильтры*. Это подключаемые к программе модули, часто третьих фирм, позволяющие обрабатывать изображение по заданному алгоритму. Иногда такие алгоритмы бывают очень сложными, а окно фильтра может иметь множество настраиваемых параметров. Из групп фильтров популярны продукты серий *Kai's Power Tools*, *Alien Skin*, *Andromeda* и другие.

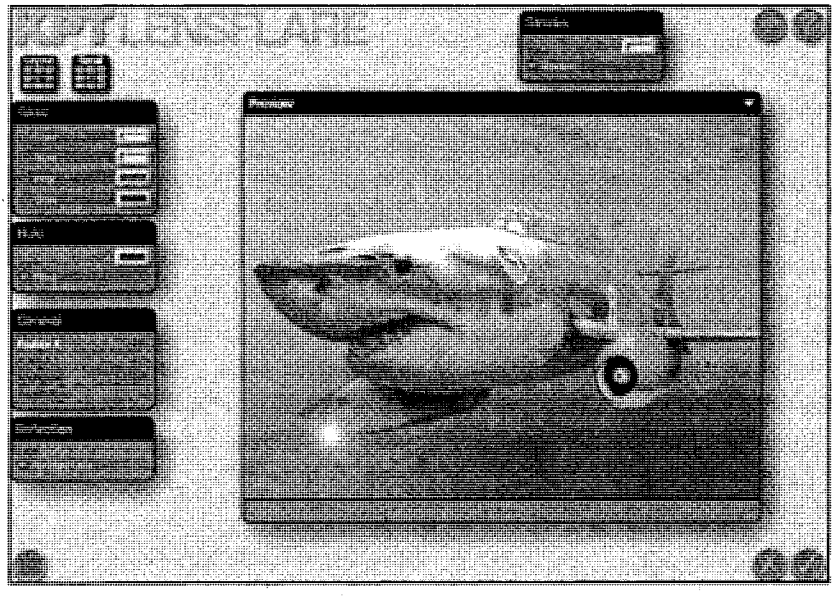


Рис. 15.19. Фильтр KPT Lens Flare из пакета Kai's Power Tools 6 предназначен для создания эффекта блика линзы фотоаппарата

15.4. Средства для работы с векторной графикой

Средства создания и обработки векторной графики

К программным средствам создания и обработки векторной графики относятся графические редакторы (например: *Adobe Illustrator*, *Macromedia Freehand*, *CorelDraw*) и векторизаторы (трассировщики) — специализированные пакеты преобразования растровых изображений в векторные (например, *Adobe StreamLine*, *CorelTrace*).

Векторный редактор Adobe Illustrator является одним из общепризнанных лидеров среди программ этого класса. Его особое преимущество заключается в хорошо

отлаженном взаимодействии с другими продуктами компании *Adobe*, прежде всего с пакетами *Photoshop, InDesign*. Эти приложения выполнены в едином стиле и образуют законченный пакет.

Векторный редактор Macromedia Freehand с простым и дружественным интерфейсом служит удобным инструментом работы для начинающих. Программа отличается небольшим размером и хорошим быстродействием. Нетребовательность к аппаратным ресурсам позволяет работать на компьютерах среднего уровня. Инструментальные средства программы достаточны для разработки сложных документов и лишь в некоторых элементах уступают более мощным средствам *Adobe Illustrator* и *CorelDraw*. Пакет специально адаптирован для совместной работы с программой компьютерной верстки *QuarkXPress*.

Векторный редактор CorelDraw исторически, особенно в России, считается основным пакетом создания и обработки векторной графики на платформе *Windows*. К его преимуществам относятся развитая система управления и обширные средства настройки параметров инструментов. По возможностям создания самых сложных художественных композиций *CorelDraw* превосходит конкурентов. Однако интерфейс программы считается непростым для освоения.

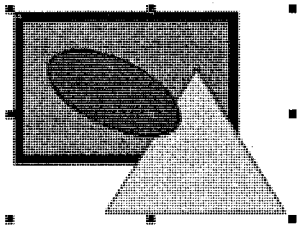
Трассировщик Adobe StreamLine по праву занимает ведущее место в своем классе программ. Хотя имеются более мощные пакеты, ориентированные на обработку чертежей, они очень требовательны к аппаратным ресурсам, да и по стоимости много дороже. *StreamLine* позволяет проводить тонкую настройку параметров векторизации, что улучшает ее точность. Более всего векторизация удобна для преобразования чертежей, черно-белых рисунков и другой простой графики без полутонов. Полутоновые и цветные изображения обрабатываются хуже, и для приближения к оригиналу результат требует значительной доработки.

Основные понятия векторной графики

Основным объектом векторной графики является *линия*. При этом *прямая* линия рассматривается как частный случай *кривой*. Иногда вместо понятия линии используется понятие *контур*. Этот термин более полно отражает суть, поскольку контур может иметь любую форму — прямой, кривой, ломаной линии, фигуры.

Каждый контур имеет две или более *опорных точек*, также именуемых *узлами*. Элемент контура, заключенный между двумя смежными опорными точками, называют *сегментом контура*. *Форму* контура меняют перемещением опорных точек, изменением их свойств, добавлением новых и удалением имеющихся узлов. Контур может быть *открытым* или *замкнутым* — когда последняя опорная точка одновременно является и первой. Свойства замкнутых и открытых контуров различны.

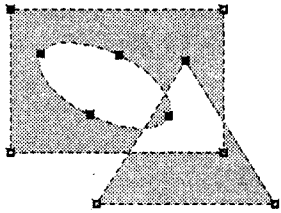
Контур является элементарным графическим *объектом*. Из контуров создают новые объекты или их группы. С несколькими контурами выполняют операции *группировки, комбинирования, объединения*. В результате образуются соответственно: *группа объектов, составной контур, новый контур*. После операции группировки каждый контур сохраняет свои свойства и принадлежащие ему узлы. После операции комбинирования составной контур приобретает новые свойства, но узлы оста-



Группировка объектов

Операция не влияет ни на свойства объектов, ни на узловые точки контуров

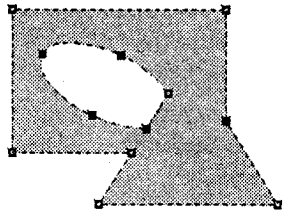
Разгруппировка восстанавливает независимость всех объектов



Составной контур

Операция не влияет на узловые точки, но все объекты приобретают единые свойства

Разгруппировка не восстанавливает исходные свойства



Объединенный контур

Все объекты приобретают единые свойства, образуются новые узловые точки

Разгруппировка не восстанавливает ни исходные свойства, ни узловые точки

Рис. 15.20. Действия с группой векторных объектов

ются прежними. После операции объединения образуются новые узлы и меняются свойства исходных контуров (рис. 15.20).

Параметры обводки контура определяют его вид при отображении. К ним относятся:

- толщина линии;
- цвет линии;
- тип линии (сплошная, пунктирная и прочие);
- форма концов (со стрелкой, закругленные и прочие).

Замкнутые контуры обладают особым свойством — *заливкой*, то есть параметрами заполнения охватываемой области. Заливка также является объектом и обладает собственным набором свойств. Различают несколько типов заливки:

- заливка основным цветом, то есть заполнение внутренней области избранным цветом;
- градиентная заливка — заполнение двумя или более цветами с плавным переходом между ними;
- текстурная заливка — заполнение узором с регулярной структурой;
- заливка изображением-картой — заполнение готовым растровым изображением, называемым *картой*;
- заливка векторным изображением (текстуры *PostScript*).

Векторный редактор Adobe Illustrator

Редактор *Adobe Illustrator* удобен для изучения начинающими пользователями по причине наличия официальных локализованных версий (только устаревших, русский язык в версии 10.0 не поддерживается), сравнительно простого интерфейса и

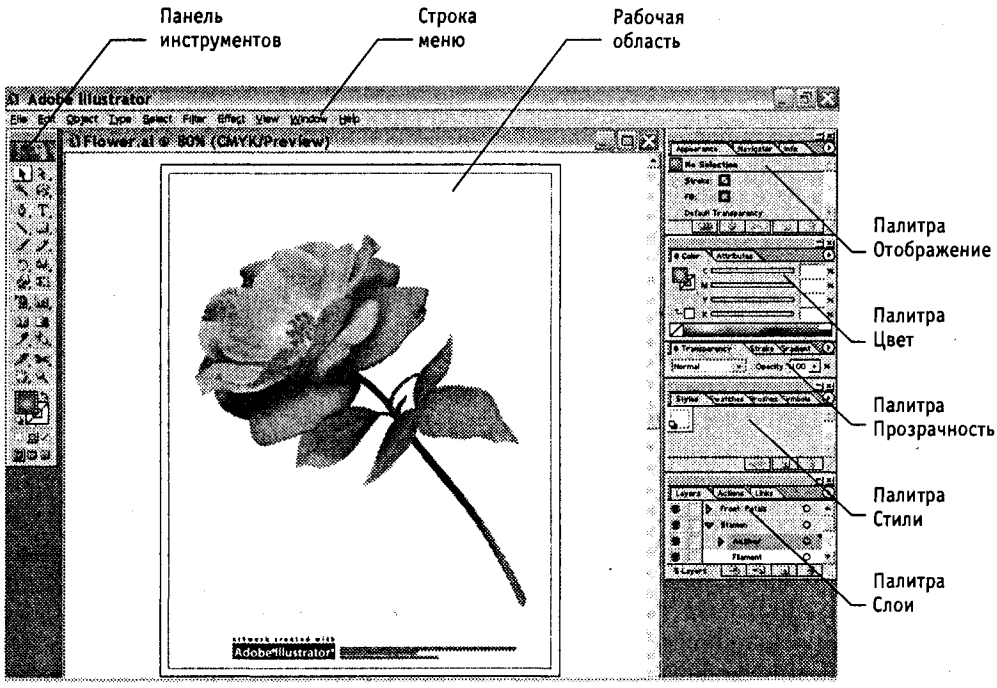


Рис. 15.21. Рабочее окно векторного редактора Adobe Illustrator

развитых функциональных возможностей (рис. 15.21). Главным недостатком *Adobe Illustrator* считается невозможность создания многостраничных документов.

Основные элементы управления программы *Adobe Illustrator* сосредоточены в строке меню, на панели инструментов и в инструментальных палитрах.

Панель инструментов выполнена подобно рассмотренной нами ранее для *Adobe Photoshop* и включает девять групп значков (рис. 15.22). Первая группа объединяет инструменты выделения объектов. Инструмент Selection (Выделение) позволяет выделить объект целиком щелчком на его контуре или построением рамки вокруг объекта. Инструмент Direct Selection (Частичное выделение) служит для выделения части контура, например одного сегмента. При нажатой клавише SHIFT этими инструментами выделяют несколько объектов. Инструментом Лассо (Lasso) выделяют объекты, хотя бы частично попавшие внутрь очерченной указателем области. Инструмент Magic Wand (Волшебная палочка) позволяет выделять объекты с одинаковыми свойствами контура и заливки.

Для рисования предназначены инструменты второй группы, объединяющие средства создания фигур, линий и текста. Инструмент Pen (Перо) и его альтернативные варианты используют для рисования кривых и редактирования их точек. Назначение инструмента Text (Текст) понятно из его названия. Альтернативные варианты инструмента позволяют разместить текст в окне (текстовом блоке), по вертикали, под наклоном и вдоль кривой. Линии и сегменты заранее заданной формы рисуют

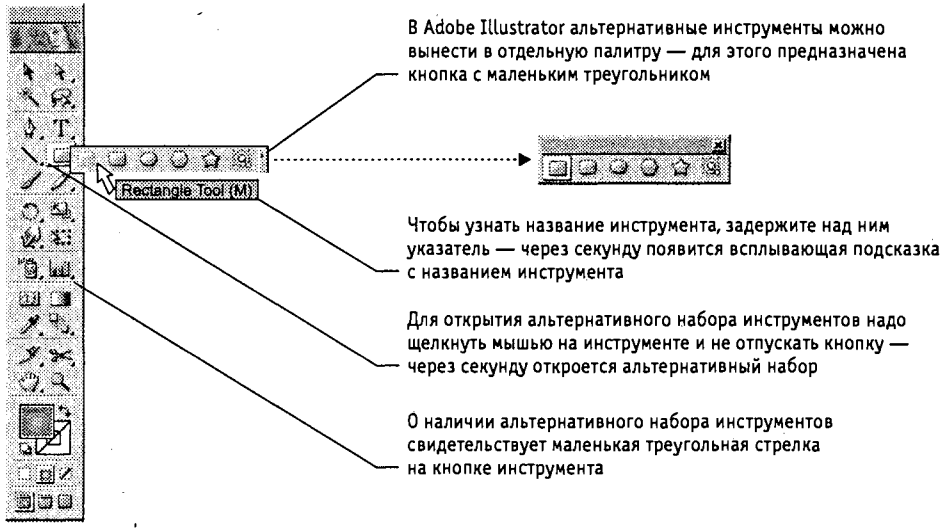


Рис. 15.22. Панель инструментов Adobe Illustrator

с помощью инструментов Line Segment (Сегмент прямой линии), Arc (Дуга), Spiral (Спираль), Rectangular Grid (Прямоугольная сетка), Polar Grid (Сетка в полярных координатах). Для этих инструментов (кроме Спирали) можно заранее задать свойства создаваемого объекта в диалоговом окне, открываемом при двойном щелчке на значке инструмента или одинарном щелчке на рабочем поле.

Инструменты Rectangle (Прямоугольник), Rounded Rectangle (Прямоугольник со скругленными углами), Ellipse (Эллипс), Polygon (Многоугольник), Star (Звезда), Flare (Вспышка) служат для создания соответствующих геометрических фигур. Инструмент Paintbrush (Кисть) расставляет вдоль рисуемого контура объекты, образующие Узор (Pattern) или Каллиграфическую линию (Calligraphic Line). Инструментом Pencil (Карандаш) рисуют контуры произвольной формы (с автоматической расстановкой узлов). Альтернативными вариантами карандаша служат инструменты Smooth (Сглаживание) и Erase (Ластик). Первый позволяет улучшить гладкость контура, а Ластик удаляет опорные точки контура в области своего воздействия.

Инструменты третьей группы предназначены для различных преобразований объектов. Суть этих операций ясна из названий инструментов: Rotate (Поворот), Reflect (Зеркало), Twist (Завихрение), Scale (Масштаб), Shear (Перекос), Reshape (Конфигурирование), Warp (Деформация), Twirl (Скручивание), Pucker (Сжатие), Bloat (Раздувание), Scallop (Раковина), Crystallize (Кристаллизация), Wrinkle (Сморщивание), Free Transform (Свободное преобразование).

Следующая группа содержит инструмент Symbol Sprayer (Распылитель символов) и его альтернативные варианты, а также средства построения диаграмм (Graph) различной формы.

Пятая группа в основном объединяет средства работы с заливкой. В нее входят инструменты: Mesh (Сетка), Gradient (Градиентная заливка), Eyedropper (Пипетка),

Paint Bucket (Сплошная заливка). В эту же группу по странной логике разработчиков попали инструменты Measure (Измерительная линейка), Blend (Смешивание) и Auto Trace (Автоматическая трассировка).

В следующей группе объединены инструменты вспомогательного назначения: Slice (Разделитель), Scissors (Ножницы), Knife (Нож), Hand (Рука), Page (Страница) и Zoom (Масштаб). Разделитель автоматически разбивает документ на блоки для публикации в *World Wide Web*. Ножницы позволяют разрезать контур в определенной точке. Разрезать объект на части позволяет инструмент Нож.

Элементы управления в нижней части Панели инструментов практически совпадают с рассмотренными выше для программы *Adobe Photoshop*.

В векторном редакторе *Adobe Illustrator* также применяются *палитры инструментов*. Их общие свойства и методы управления отображением, настройки параметров совпадают с таковыми в редакторе *Adobe Photoshop*. Всего в редакторе *Adobe Illustrator* версии 10 предусмотрено 24 палитры. К сожалению, параметры и свойства объекта разнесены по многим палитрам и потому настройка нужных свойств вызывает определенные сложности у начинающих пользователей.

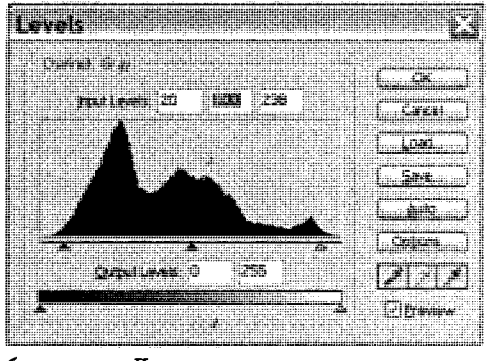
В целом графический редактор *Adobe Illustrator* обладает обширными возможностями и достаточно прост в освоении. Однако некоторые особенности программы позволяют характеризовать ее как средство художественного творчества, слабо приспособленное для технической графики и, тем более, создания технической документации.

Практическое занятие

Упражнение 15.4. Изменение динамического диапазона изображения

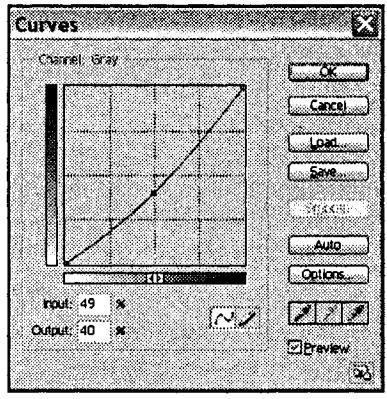
15 мин

1. Запустите программу *Adobe Photoshop*, откройте файл Old Image.jpg. Этот файл входит в состав образцов, поставляемых с программой, и находится в папке \Adobe Photoshop 7.0\Samples.
2. Оцените динамический диапазон изображения, то есть разброс между минимальной и максимальной яркостью. Для этого откройте окно Image ▸ Adjustments ▸ Levels (Изображения ▸ Коррекция ▸ Уровни). По гистограмме снимка видно, что диапазон яркостей сдвинут в область темных полутонов, то есть изображение не охватывает доступный динамический диапазон.



3. Расширьте динамический диапазон изображения. Для этого щелкните на кнопке Auto (Авто). Граничные движки уровней сместятся к центру — левый на уровень 20, правый на уровень 238. Снимок станет более контрастным.

4. По гистограмме видно, что область темных тонов значительно превосходит область светлых тонов — снимок выглядит темным. Для исправления изображения выполните *гамма-коррекцию*. Откройте диалоговое окно Image ▸ Adjustments ▸ Curves (Изображения ▸ Коррекция ▸ Кривые). Указателем мыши перетащите середину гамма-кривой вниз, «подтягивая» полутона к более светлым. При этом окончания кривой (то есть границы динамического диапазона) остаются неизменными. Регулируя кривизну линии, добейтесь наиболее сбалансированной яркости и контрастности. Зафиксируйте результат щелчком на кнопке OK.



5. Испытаем другой способ гамма-коррекции — с помощью диалогового окна Levels (Уровни). В среднем окне поля Input Levels установим гамма-коэффициент, больший единицы. Или переместим средний движок влево, отслеживая изменения гамма-коэффициента. Таким способом добиваемся повышения качества изображения.

Мы установили, что управление динамическим диапазоном изображения позволяет существенно улучшить качество растровой графики. Мы также выяснили, что существует зависимость между яркостью и контрастностью изображения.

Упражнение 15.5. Ретушь изображения

1. Запустите программу *Adobe Photoshop*, откройте файл Old Image.jpg.
2. Выделите слишком темный участок снимка. Выберите инструмент Lasso (Лассо), на панели свойств установите флажок Anti-aliased (Сглаживание) и задайте значение растушевки (Feather) 4 пиксела. Инструментом Lasso (Лассо) выделите темную область в левом верхнем углу изображения.
3. Для осветления выделенной области откройте диалоговое окно Image ▸ Adjustments ▸ Levels (Изображения ▸ Коррекция ▸ Уровни) и щелкните на кнопке Auto (Авто). Выделенный участок изображения осветляется и на нем прорисовываются невидимые ранее детали. Перемещая движки, добейтесь такой яркости и контрастности изображения, чтобы тень исчезла.
4. Удалите повреждения на изображении. Для этого воспользуйтесь инструментами Stamp (Штамп) и Healing Brush (Заживляющая кисть). Процесс заключается в забивке ненужной детали переносом фрагментов соседних областей.

   В палитре Layers (Слои) щелкните правой кнопкой мыши на слое Background (Фон) и в открывшемся меню выберите пункт Duplicate Layer (Дублировать слой). Щелчком на новом слое сделайте его активным.

   Выберите инструмент Stamp (Штамп). На панели свойств в разделе Brush (Кисть) щелкните на раскрывающей кнопке, в открывшемся диалоговом окне устано-

вите диаметр кисти движком Master Diameter (Установка диаметра). В списке типов кисти выберите один из вариантов с мягким краем (Soft Round).

Установите указатель мыши на исходный участок изображения и, удерживая клавишу ALT, щелкните мышью, фиксируя зону, с которой будет происходить перенос изображения. Исходная точка в момент щелчка отмечается крестообразным маркером. Переместите указатель мыши на ближайшую область фона и щелчками закрасьте ее. При необходимости изменяйте исходную точку щелчком с нажатой клавишей ALT. С помощью инструмента Stamp удалите примерно половину повреждений.

5. Действуя аналогичным образом, с помощью инструмента Healing Brush (Заживляющая кисть) удалите остальные повреждения на снимке.

Рис. 15.23. Рисунок до ретуши (слева) и после нее (справа)

6. Выполните местную коррекцию резкости. Выберите инструмент Sharpen (Резкость). На панели свойств выберите подходящий размер области действия инструмента. Проведите улучшение резкости лица для усиления рельефности деталей — изображение станет выразительнее.

Инструменты местной коррекции и ретуши изображения позволяют восстановить поврежденные и старые фотоснимки, улучшить восприятие деталей изображения, убрать ненужные детали, подчеркнуть важные элементы изображения.

Упражнение 15.6. Использование фильтров

30 мин

1. Запустите программу *Adobe Photoshop*, откройте файл Old Image.jpg.
2. Удаляем пыль и царапины со снимка. Дадим команду Filter ▸ Noise ▸ Dust & Scratches (Фильтр ▸ Шум ▸ Царапины). Обратите внимание, что фильтр сглаживает гра-

ницы элементов изображения. Он удобен при ретуши старых, поврежденных снимков и воздействует на всю поверхность изображения.

3. Повышаем резкость изображения. Дадим команду Filter ▸ Sharpen ▸ Sharpen Edges (Фильтр ▸ Резкость ▸ Края). Обратите внимание, что фильтр воздействует только на границы перехода между элементами изображения разной яркости.
4. Проведите эксперименты с фильтрами группы Stylize (Стилизация).
5. Проведите эксперименты с фильтрами группы Pixelate (Пикселизация).

Мы освоили приемы применения фильтров для улучшения качества изображения за счет ретуши и акцента на важных деталях. Возможна имитация размещения изображений на разных материалах, изменение условий освещенности и другие художественные эффекты. Применять фильтры надо осторожно, чтобы не испортить, а усилить впечатление, которое должно вызывать изображение.

Упражнение 15.7. Обтравка изображения

30 мин

1. Запустите программу *Adobe Photoshop*, откройте файл Old Image.jpg.
2. Процесс точного выделения элемента изображения называют *обтравкой*. Выберите инструмент Polygonal Lasso (Полигональное лассо). На панели свойств установите нулевую величину Feather (Растушевка).
3. Обведите как можно точнее инструментом Polygonal Lasso (Полигональное лассо) контур фигуры. За один прием это сделать практически невозможно, поэтому поправьте контур инструментом Polygonal Lasso (Полигональное лассо) при нажатых клавишах SHIFT (для добавления области выделения к первоначальной) или ALT (для вычитания области выделения из первоначальной).
4. Испытайте альтернативный способ выделения области на изображении. Выберите инструмент Magic Wand (Волшебная палочка). Установите на панели свойств в поле Tolerance (Чувствительность) значение 24. Щелкая инструментом, выделите фон вокруг фигуры. Для добавления или исключения областей выделения удерживайте в момент щелчка клавиши SHIFT или ALT.
5. Выполните окончательную корректировку выделенной области с помощью инструмента Lasso (Лассо).
6. Обтравочный контур может храниться в том же файле, но отдельно от изображения. Для этого предназначены каналы. В палитре Channels (Каналы) щелкните на кнопке сохранения выделенной области в новом канале. В окне палитры появляется миниатюра с изображением обтравочного контура. Загрузку контура из канала производят щелчком на его изображении при нажатой клавише CTRL.

Каналы и обтравочные контуры служат мощным средством композиции и редактирования изображений. Их умелое применение позволяет создавать качественную рекламную и художественную продукцию, в которой использована вся сила инструментария Adobe Photoshop.

Исследовательская работа

Задание 15.1. Составление композиции в программе Adobe Photoshop

1. Запустите программу *Adobe Photoshop*, откройте файлы Ducky.tif и Dune.tif.
2. Улучшите динамический диапазон изображения Dune.tif командой Image ▸ Adjustments ▸ Levels ▸ Auto (Изображение ▸ Коррекция ▸ Уровни ▸ Авто).
3. Выберите инструмент Polygonal Lasso (Полигональное лассо), в файле Ducky.tif выполните обтравку фигуры.
4. Создайте канал для контура обтравки в изображении Ducky.tif. Для этого щелкните в палитре Channels (Каналы) на кнопке сохранения контура в новом канале.
5. Улучшите динамический диапазон выделенной фигуры командой Image ▸ Adjustments ▸ Levels ▸ Auto (Изображение ▸ Коррекция ▸ Уровни ▸ Авто).
6. Выполните гамма-коррекцию выделенного изображения, задав коэффициент 0,85 в окне Image ▸ Adjustments ▸ Levels (Изображение ▸ Коррекция ▸ Уровни).
7. Примените растушевку краев области выделения, установив в поле Feather (Растушевка) значение 4 пх.
8. Скопируйте выделенную область в буфер обмена командой Edit ▸ Copy (Редактирование ▸ Копирование).
9. Вклейте скопированную область на новый слой изображения Dune.tif командой Edit ▸ Paste (Редактирование ▸ Вклеивание).
10. Установите на палитре Layers (Слои) уровень прозрачности нового слоя 50%. Получится изображение, представленное на рис. 15.24.

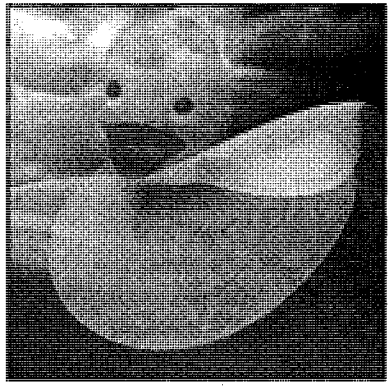


Рис. 15.24. Композиция, выполненная с использованием дополнительного слоя

11. Поэкспериментируйте с режимом совмещения слоев с помощью раскрывающегося списка в палитре Layers (Слои).
12. Попробуйте действие различных фильтров, применяя их раздельно к слоям. Запишите названия и параметры примененных фильтров.

Слой	Название фильтра	Параметры фильтра

Мы выяснили, что программа Adobe Photoshop позволяет достаточно легко создавать сложные композиции с применением специальных эффектов.

Практическое занятие

Упражнение 15.8. Создание простейших объектов в редакторе Adobe Illustrator


15 мин

1. Запустите векторный редактор *Adobe Illustrator*.
2. Создайте новый документ командой File ▸ New (Файл ▸ Создать).
3. Установите параметры страницы в диалоговом окне: в поле Size (Размер) значение A4, в поле Units (Единица) значение Millimeters (Миллиметры), переключателем Orientation (Ориентация) — книжную ориентацию.
4. *Рисование прямой линии.* Выберите инструмент Pen (Перо). Первым щелчком задайте начальную опорную точку, вторым щелчком после смещения указателя мыши задайте конечную опорную точку. На палитре Stroke (Обводка) установите толщину 8 пунктов. При нажатой клавише CTRL щелкните на свободном поле. Далее создайте строго вертикальную линию. При нажатой клавише SHIFT сделайте первый щелчок, удерживая клавишу, переместите указатель вниз и сделайте второй щелчок.
5. *Рисование замкнутого контура.* Выберите инструмент Pen (Перо). Щелчком задайте начальную опорную точку, затем сделайте четыре щелчка вдоль воображаемого контура многоугольника. Подведите указатель к начальной опорной точке так, чтобы рядом с его значком появился кружок. В этот момент сделайте последний щелчок.
6. *Рисование эллипса и окружности.* Выберите инструмент Ellipse (Эллипс). Щелкните на рабочем поле и протягиванием задайте форму и размеры эллипса. Удерживая клавишу SHIFT, вновь щелкните на рабочем поле и протягиванием создайте правильную окружность. Для рисования фигуры «от центра» удерживайте комбинацию клавиш ALT+SHIFT, щелкните на рабочем поле и создайте правильную окружность, начиная от ее центра.
7. *Рисование спирали.* Щелкните на значке Line Segment (Сегмент) инструментальной панели и удерживайте кнопку до появления линейки с альтернативным набором инструментов. Выберите инструмент Spiral (Спираль). Щелкните на рабочем поле и протягиванием от центра создайте спираль. Следующую спираль создайте методом задания параметров. Щелкните на рабочем поле и в открывшемся диалоговом окне Spiral (Спираль) задайте необходимые параметры.

8. *Рисование многоугольника.* Выберите инструмент Polygon (Многоугольник). Щелкните на рабочем поле и, удерживая нажатой кнопку мыши, с помощью клавиш управления курсором (ВВЕРХ и ВНИЗ) установите число его вершин равным двенадцати.

Мы научились создавать простейшие векторные объекты с помощью векторного графического редактора Adobe Illustrator. Мы установили, как с помощью клавиш SHIFT и ALT можно модифицировать действие чертежных инструментов.

Упражнение 15.9. Создание криволинейных контуров

1. Запустите векторный редактор *Adobe Illustrator.*
2. Создайте новый документ командой File ▸ New (Файл ▸ Создать).
3. Установите параметры страницы в диалоговом окне: в поле Size (Размер) значение A4, в поле Units (Единица) значение Millimeters (Миллиметры), переключателем Orientation (Ориентация) — книжную ориентацию.
4. *Рисование контура произвольной формы.* Выберите инструмент Pencil (Карандаш). Удерживая нажатой кнопку мыши, протягивайте указатель по рабочему полю, рисуя волнистую линию. Выберите инструмент (Кисть). Удерживая нажатой кнопку мыши, протягивайте указатель по рабочему полю, рисуя волнистую линию.
5. *Построение кривых.* Выбираем инструмент Pen (Перо). Создайте три кривые, основанные на разных математических формулах (рис. 15.25):

   кривая первого порядка — щелчок + щелчок;

   кривая второго порядка — щелчок + протягивание + щелчок;

   кривая третьего порядка — щелчок + протягивание + щелчок + протягивание.

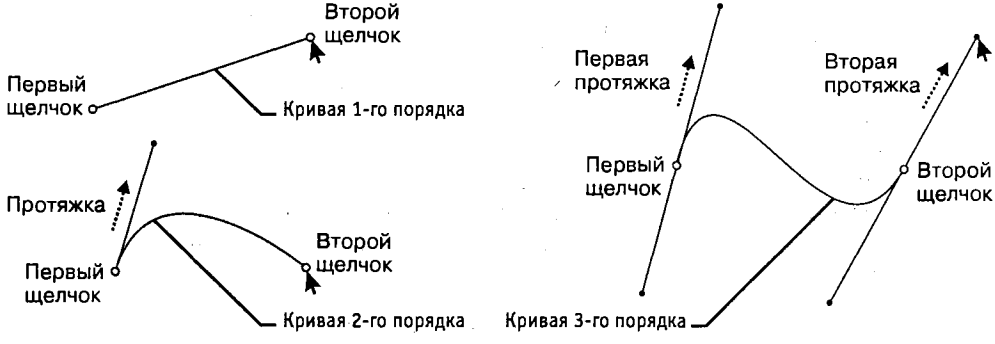


Рис. 15.25. Построение кривых первого, второго и третьего порядка

6. *Создание замкнутого криволинейного контура.* Выберите инструмент Pen (Перо). Пользуясь приемами, описанными в предыдущем пункте, создайте контур. Последний щелчок выполните, подведя указатель к начальной опорной точке (после появления кружка у его значка).

Упражнение 15.10. Редактирование контуров

1. Запустите векторный редактор *Adobe Illustrator*.
2. Создайте новый документ командой File ▸ New (Файл ▸ Создать).
3. Установите параметры страницы в диалоговом окне: в поле Size (Размер) значение A4, в поле Units (Единица) значение Millimeters (Миллиметры), переключателем Orientation (Ориентация) — книжную ориентацию.
4. Создайте замкнутый криволинейный контур. Выберите инструмент Pen (Перо). Щелчками с последующим протягиванием создайте на рабочем поле замкнутый криволинейный контур.
5. *Перемещение опорных точек.* Выберите инструмент Direct Select (Точная выборка). Выделите опорную точку на криволинейном контуре и протягиванием сместите ее так, чтобы изменить форму контура. Повторите операцию для двух-трех опорных точек.

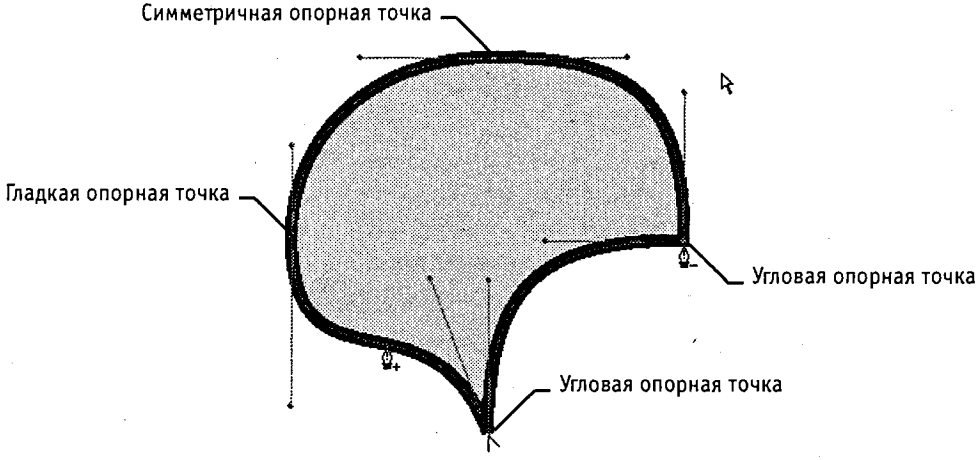


Рис. 15.26. Редактирование контура

6. *Изменение свойств опорных точек.* Выберите инструмент Convert Anchor Point (Преобразовать опорную точку). Щелчком на *гладкой* опорной точке контура преобразуйте ее в *угловую*. Щелчком на *гладкой* опорной точке с последующим протягиванием преобразуйте ее в *симметричную*. Установите указатель на *управляющую линию*, изменением ее длины и угла наклона касательной измените форму криволинейного контура.
7. *Создание и удаление опорных точек.* Выберите инструмент Add Anchor Point (Добавить опорную точку). Выберите сегмент на криволинейном контуре и щелчком добавьте новую опорную точку. Выберите инструмент Delete Anchor Point (Удалить опорную точку). Щелчком на опорной точке криволинейного контура удалите ее.

Упражнение 15.11. Обработка замкнутых контуров

1. Запустите векторный редактор *Adobe Illustrator*.
2. Создайте новый документ командой File ▸ New (Файл ▸ Создать).
3. Установите параметры страницы в диалоговом окне: в поле Size (Размер) значение A4, в поле Units (Единица) значение Millimeters (Миллиметры), переключателем Orientation (Ориентация) — книжную ориентацию.
4. Создайте замкнутый криволинейный контур. Выберите инструмент Pen (Перо). Щелчками с последующим протягиванием создайте на рабочем поле замкнутый криволинейный контур. Откройте палитры Color, Gradient, Swatches (Window ▸ *Имя палитры*).
5. *Выполнение сплошной заливки.* Инструментом Selection (Выделение) выберите криволинейный контур. На панели инструментов щелкните по кнопке механизма заливки Color (Цвет). Выберите цвет заливки щелчком на образце на палитре Swatches (Образцы). Меняйте цвет заливки щелчками на цветовой линейке палитры Color (Цвет). Меняйте цвет заливки перемещением движков (R, G, B) на палитре Color.
6. *Выполнение градиентной заливки.* Инструментом Selection (Выделение) выберите криволинейный контур. На панели инструментов щелкните на кнопке механизма заливки Gradient (Градиент). Выберите исходный цвет в палитрах Swatches (Образцы) или Color (Цвет) методами, описанными в предыдущем пункте. На палитре Gradient (Градиент) в раскрывающемся списке Type укажите Linear (Линейный). В поле Angle (Угол) задайте направление 45 градусов. На *градиентной линейке* щелчком выделите *маркер концевого цвета*, затем на палитре Color определите его цветовой тон. Перемещением маркеров концевых цветов и *маркера срединной точки* задайте необходимую градиентную растяжку.
7. *Выполнение текстурной заливки.* Инструментом Selection (Выделение) выберите криволинейный контур. В палитре Swatches (Образцы) щелчком на кнопке Show Pattern Swatches (Показать палитру образцов) откройте комплект образцов. Щелчком на образце назначьте параметры текстурной заливки.

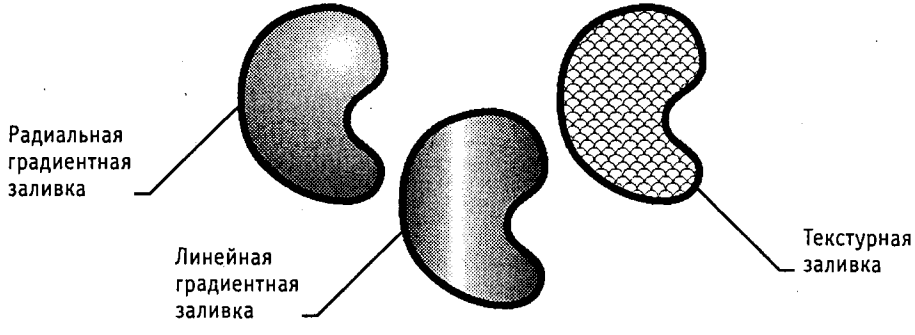


Рис. 15.27. Различные виды заливок замкнутых контуров

8. *Размыкание замкнутого контура.* Выберите инструмент Scissors (Ножницы). Щелчком на сегменте криволинейного контура разомкните его в избранной точке.

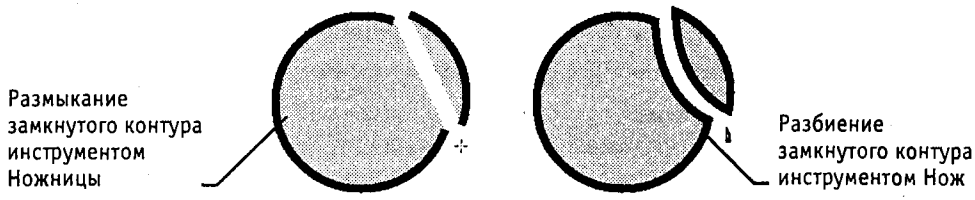


Рис. 15.28. Разделение контуров методом размыкания и разбиения

9. *Разбиение замкнутого контура.* Выберите инструмент Knife (Нож). При нажатой кнопке мыши протащите указатель поперек криволинейного контура. Выделите один из получившихся объектов инструментом Direct Selection и перетащите в сторону.

В векторной графике замкнутые контуры обладают особым свойством — заливкой. В этом упражнении мы научились управлять этим свойством средствами редактора Adobe Illustrator. Мы изучили несколько методов создания заливки разных типов и познакомились с приемами размыкания и разбиения замкнутых контуров.

Исследовательская работа

Задание 15.2. Создание сложных композиций средствами Adobe Illustrator

1. Запустите векторный редактор *Adobe Illustrator*.
2. Создайте новый документ командой File ▸ New (Файл ▸ Создать).
3. Установите параметры страницы в диалоговом окне: в поле Size (Размер) значение A4, в поле Units (Единица) значение Millimeters (Миллиметры), переключателем Orientation (Ориентация) — книжную ориентацию.
4. Выберите инструмент Pen (Перо). Щелчками с последующим протягиванием создайте на рабочем поле замкнутый криволинейный контур. На палитре Stroke (Обводка) установите толщину обводки 10 пунктов.
5. Выберите инструмент Ellipse (Эллипс). Щелкните на рабочем поле и протягиванием задайте форму и размеры эллипса. На палитре Stroke (Обводка) установите толщину обводки 4 пункта. Удерживая клавишу SHIFT, вновь щелкните на рабочем поле и протягиванием создайте правильную окружность. На палитре Stroke (Обводка) установите толщину обводки 7 пунктов.
6. Откройте палитры Color, Gradient, Swatches (Window ▸ *Имя палитры*).
7. Инструментом Selection (Выделение) выберите окружность. На панели инструментов щелкните по кнопке механизма заливки Gradient (Градиент). Выберите исходный цвет в палитре Swatches (Образцы). На палитре Gradient (Градиент) в раскрывающемся списке Type (Тип) укажите Linear (Линейный).

В поле Angle (Угол) задайте направление 0 градусов. На градиентной линейке щелчком выделите маркер концевого цвета, затем на палитре Color (Цвет) определите его цветовой тон. Перемещением маркеров концевых цветов и маркера срединной точки задайте необходимую градиентную растяжку.

8. Инструментом Selection (Выделение) выберите эллипс. На палитре Swatches (Образцы) щелчком на кнопке Show Pattern Swatches (Показать палитру образцов) откройте комплект образцов. Щелчком на образце назначьте параметры текстурной заливки.
9. Инструментом Selection (Выделение) выберите криволинейный контур. На панели инструментов щелкните на кнопке механизма заливки Color (Цвет). Выберите цвет заливки щелчком на образце в палитре Swatches (Образцы).
10. *Группировка объектов.* Инструментом Selection (Выделение) при нажатой клавише SHIFT выберите все объекты. Дайте команду Object ▸ Group (Объект ▸ Сгруппировать). Запишите, изменились ли свойства объектов:

    а) обводка _____;

    б) заливка _____.
11. *Разгруппировка.* Инструментом Selection (Выделение) выберите сгруппированные объекты. Дайте команду Object ▸ Ungroup (Объект ▸ Разгруппировать).
12. *Объединение контуров.* Инструментом Selection (Выделение) выберите окружность и перетащите ее до частичного наложения на эллипс. Окружность должна находиться сверху эллипса, так как создавалась последней. При нажатой клавише SHIFT выберите эллипс. На панели Pathfinder (Путь) щелкните на кнопке Add to shape area (Добавить к области). Запишите, как изменились свойства объектов.

Объект	Обводка, толщина в пунктах		Заливка, тип	
	До операции	После операции	До операции	После операции
Окружность				
Эллипс				
Результирующий				

13. *Пересечение контуров.* Отмените предыдущую операцию командой Edit ▸ Undo (Правка ▸ Отменить). Инструментом Selection (Выделение) выберите окружность. При нажатой клавише SHIFT выберите эллипс. На панели Pathfinder (Путь) щелкните на кнопке Intersect shape areas (Пересечение). Запишите, как изменились свойства объектов.

Объект	Обводка, толщина в пунктах		Заливка, тип	
	До операции	После операции	До операции	После операции
Окружность				
Эллипс				
Результирующий				

14. *Исключение контуров.* Отмените предыдущую операцию командой Edit ▸ Undo (Правка ▸ Отменить). Инструментом Selection (Выделение) выберите окруж-

ность. При нажатой клавише SHIFT выберите эллипс. На панели Pathfinder (Путь) щелкните на кнопке Exclude overlapping shape areas (Исключение). Запишите, как изменились свойства объектов.

Объект	Обводка, толщина в пунктах		Заливка, тип	
	До операции	После операции	До операции	После операции
Окружность				
Эллипс				
Результирующий				

15. *Операция Минус верхний.* Отмените предыдущую операцию командой Edit ▸ Undo (Правка ▸ Отменить). Инструментом Selection (Выделение) выберите окружность. При нажатой клавише SHIFT выберите эллипс. На панели Pathfinder (Путь) щелкните на кнопке Subtract from shape area (Вычитание). Запишите, как изменились свойства объектов.

Объект	Обводка, толщина в пунктах		Заливка, тип	
	До операции	После операции	До операции	После операции
Окружность				
Эллипс				
Результирующий				

16. *Операция Минус нижний.* Отмените предыдущую операцию командой Edit ▸ Undo (Правка ▸ Отменить). Инструментом Selection (Выделение) выберите эллипс. При нажатой клавише SHIFT выберите окружность. На панели Pathfinder (Путь) щелкните на кнопке Minus Back (Минус нижний). Запишите, как изменились свойства объектов.

Объект	Обводка, толщина в пунктах		Заливка, тип	
	До операции	После операции	До операции	После операции
Окружность				
Эллипс				
Результирующий				

17. *Комбинирование контуров.* Отмените предыдущую операцию командой Edit ▸ Undo (Правка ▸ Отменить). Инструментом Selection (Выделение) выберите окружность. При нажатой клавише SHIFT выберите эллипс. Дайте команду Object ▸ Compound Path ▸ Make. Запишите, как изменились свойства объектов.

Объект	Обводка, толщина в пунктах		Заливка, тип	
	До операции	После операции	До операции	После операции
Окружность				
Эллипс				
Результирующий				

18.1. Компьютер как инструмент научной работы

Вычислительная мощь компьютера позволяет использовать его как средство автоматизации научной работы. Для решения сложных расчетных задач используют программы, написанные специально. В то же время, в научной работе встречается широкий спектр задач ограниченной сложности, для решения которых можно использовать универсальные средства.

К такого рода задачам относятся, например, следующие:

- подготовка научно-технических документов, содержащих текст и формулы, записанные в привычной для специалистов форме;
- вычисление результатов математических операций, в которых участвуют числовые константы, переменные и размерные физические величины;
- операции с векторами и матрицами;
- решение уравнений и систем уравнений (неравенств);
- статистические расчеты и анализ данных;
- построение двумерных и трехмерных графиков;
- тождественные преобразования выражений (в том числе упрощение), аналитическое решение уравнений и систем;
- дифференцирование и интегрирование, аналитическое и численное;
- решение дифференциальных уравнений;
- проведение серий расчетов с разными значениями начальных условий и других параметров.

К универсальным программам, пригодным для решения таких задач, относится, например, программа *Mathcad*. Это автоматизированная система, позволяющая динамически обрабатывать данные в числовом и аналитическом (формульном) виде. Программа *Mathcad* сочетает в себе возможности проведения расчетов и подготовки форматированных научных и технических документов.

Научно-технические документы обычно содержат формулы, результаты расчетов в виде таблиц данных или графиков, текстовые комментарии или описания, другие иллюстрации. В программе *Mathcad* им соответствуют два вида объектов: *формулы* и *текстовые блоки*. Формулы вычисляются с использованием числовых констант, переменных, функций (стандартных и определенных пользователем), а также общепринятых обозначений математических операций. Введенные в документ *Mathcad* формулы автоматически приводятся к стандартной научно-технической форме записи. Графики, которые автоматически строятся на основе результатов расчетов, также рассматриваются как формулы. Комментарии, описания и иллюстрации размещаются в текстовых блоках, которые игнорируются при проведении расчетов.

Чтобы буквенные обозначения можно было использовать при расчетах по формулам, этим обозначениям должны быть сопоставлены числовые значения. В программе *Mathcad* буквенные обозначения рассматриваются как переменные, и их значения задаются при помощи оператора присваивания (вводится символом «:»). Таким же образом можно задать числовые последовательности, аналитически определенные функции, матрицы и векторы.

Если все значения переменных известны, то для вычисления числового значения выражения (скалярного, векторного или матричного) надо подставить все числовые значения и произвести все заданные действия. В программе *Mathcad* для этого применяют оператор вычисления (вводится символом «=»). В ходе вычисления автоматически используются значения переменных и определения функций, заданные в документе ранее. Удобно задать значения известных параметров, провести вычисления с использованием аналитических формул, результат присвоить некоторой переменной, а затем использовать оператор вычисления для вывода значения этой переменной. Например:

$$g := 9.8$$
$$M := 3$$
$$F := M \cdot g$$
$$F = 29.4$$

Изменение значения любой переменной, коррекция любой формулы означает, что все расчеты, зависящие от этой величины, необходимо проделать заново. Такая необходимость возникает при выборе подходящих значений параметров или условий, поиске оптимального варианта, исследовании зависимости результата от начальных условий. Электронный документ, подготовленный в программе MathCad, готов к подобной ситуации. При изменении какой-либо формулы программа автоматически производит необходимые вычисления, обновляя изменившиеся значения и графики. Например, если документ содержит формулы $x := 4$; $\sqrt{x} = 2$, то, изменив значение переменной x, мы сразу же увидим, что изменился и результат расчета: $x := 9$; $\sqrt{x} = 3$.

При проведении расчетов с использованием реальных физических величин учитывают их размерность. Чтобы расчет был корректен, все данные должны быть приведены в одну систему единиц — в этом случае результат расчетов получится в этой же системе. Здесь скрывается характерный источник ошибок при расчетах вручную. В программе *Mathcad* единицы измерения (в любой системе) присоеди-

няют к значению величины с помощью знака умножения. Данные автоматически преобразуются в одну и ту же систему единиц (по умолчанию СИ) и обрабатываются в этом виде. Размерный результат выдается вместе с полученной единицей измерения. Например:

$v := 100 \cdot kph \quad t := 0.5 \cdot yr$ (*kph* — километры в час, *yr* — годы)

$s := v \cdot t \quad s = 4.383 \cdot 10^8\,\mathrm{m}$ (результат получен в метрах)

При работе с матрицами приходится применять такие операции, как сложение матриц, умножение, транспонирование. Часто возникает необходимость в обращении матриц и в декомпозиции (разложении в произведение матриц специального вида). Для квадратных матриц представляет интерес поиск собственных значений и собственных векторов. Программа *Mathcad* позволяет выполнить все эти операции с помощью стандартных обозначений математических операторов (сложение, умножение) или встроенных функций. Например:

$$\begin{pmatrix} 1 & 2 \\ 1 & 1 \end{pmatrix}^T = \begin{pmatrix} 1 & 1 \\ 2 & 1 \end{pmatrix}$$

$$\left|\begin{pmatrix} 1 & 2 \\ 1 & 1 \end{pmatrix}\right| = -1$$

$$\begin{pmatrix} 1 & 2 \\ 1 & 1 \end{pmatrix}^{-1} = \begin{pmatrix} -1 & 2 \\ 1 & -1 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 2 \\ 1 & 1 \end{pmatrix}^T \cdot \begin{pmatrix} 1 & 1 \\ 2 & 1 \end{pmatrix} = \begin{pmatrix} 2 & 3 \\ 3 & 5 \end{pmatrix}$$

Уравнения и системы уравнений, возникающие в практических задачах, обычно можно решить только численно. Методы численного решения реализованы и в программе *Mathcad*. Блок уравнений и неравенств, требующих решения, записывается после ключевого слова *given* (*дано*). При записи уравнений используется знак логического равенства (комбинация клавиш CTRL+=). Значения переменных, удовлетворяющие системе уравнений и неравенств, находятся с помощью стандартной функции *find*.

$x := 1 \quad y := 0$

given

$x - y = 2$

$\sin(x) = \sin(y)$

$$\mathrm{find}(x,y) = \begin{pmatrix} 2{,}571 \\ 0{,}571 \end{pmatrix}$$

При обработке результатов экспериментов часто встречаются задачи статистического анализа серий данных. Для такого рода задач программа *Mathcad* предостав-

ляет средства интерполяции данных, предсказания дальнейшего поведения функции, а также построения функций заданного вида, наилучшим образом соответствующих имеющемуся набору данных.

При статистическом анализе можно также использовать стандартные функции распределения вероятности и генераторы случайных величин с заданным распределением.

При аналитических вычислениях результат получают в нечисловой форме в результате тождественных преобразований выражений. Простейшие преобразования — это раскрытие скобок, приведение подобных членов, применение тригонометрических тождеств.

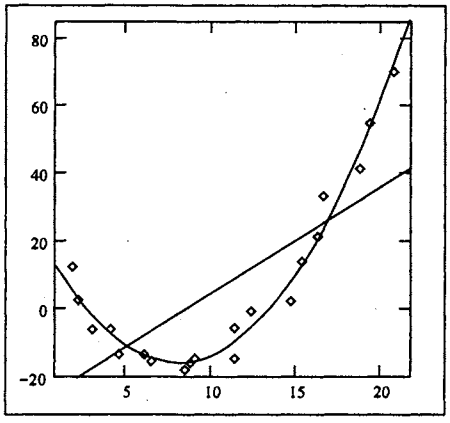


Рис. 18.1. Набор точек аппроксимирован с помощью многочленов первого и второго порядка

Например, выражение $\cos(3\cdot\mathrm{atan}(x))$ преобразуется в $\frac{4}{\left(1+x^2\right)^{\frac{3}{2}}} - \frac{3}{\left(1+x^2\right)^{\frac{1}{2}}}$.

Проверку тождественности этого преобразования выполните самостоятельно.

Более сложные преобразования позволяют находить аналитические решения некоторых уравнений и систем. Для такого рода вычислений в программе *Mathcad* используют оператор аналитического вычисления (клавиатурная комбинация CTRL+.), а также команды меню Symbolics (Аналитические вычисления). Переменные при аналитических вычислениях рассматриваются как неопределенные параметры. Результат можно использовать для анализа решения при различных значениях этих переменных. При аналитическом решении уравнений и систем за одну операцию можно найти все существующие решения. Например:

given

$z^3 + 3\cdot z^2 + 2\cdot z - 6=0$

$\mathrm{find}(z)\rightarrow[1 \quad -2+i\cdot\sqrt{2} \quad -2-i\cdot\sqrt{2}\ ]$

Дифференцирование и интегрирование заданных функций вручную — обычно несложная, но трудоемкая операция. В программе *Mathcad* для вычисления производной, а также неопределенных и определенных интегралов могут использоваться символические вычисления с помощью меню Symbolics ▸ Variable (Аналитические вычисления ▸ Переменная). Если функция не задана аналитически или не позволяет получить первообразную в виде формулы, имеется возможность численного дифференцирования и численного расчета определенных интегралов. Например, при вычислении интеграла

$$\int x\cdot e^{-x}\cdot\cos(x)dx$$

получается правильный результат:

$$-\frac{1}{2}\cdot x\cdot exp(-x)\cdot cos(x)+\frac{1}{2}\cdot x\cdot exp(-x)\cdot sin(x)+\frac{1}{2}\cdot exp(-x)\cdot sin(x)$$

Численные методы используют и для решения дифференциальных уравнений. С помощью программы *Mathcad* можно решать уравнения и системы уравнений первого порядка с заданными начальными условиями. Уравнение более высокого порядка надо сначала преобразовать в систему уравнений первого порядка.

18.2. Приемы работы с системой Mathcad

Документ программы *Mathcad* называется *рабочим листом*. Он содержит объекты: *формулы* и *текстовые блоки*. В ходе расчетов формулы обрабатываются последовательно, слева направо и сверху вниз, а текстовые блоки игнорируются.

Ввод информации осуществляется в месте расположения курсора. Программа *Mathcad* использует три вида курсоров. Если ни один объект не выбран, используется *крестообразный курсор*, определяющий место создания следующего объекта. При вводе формул используется *уголковый курсор*, указывающий текущий элемент выражения. При вводе данных в текстовый блок применяется *текстовый курсор* в виде вертикальной черты.

Ввод формул

Формулы — основные объекты рабочего листа. Новый объект по умолчанию является формулой. Чтобы начать ввод формулы, надо установить крестообразный курсор в нужное место и начать ввод букв, цифр, знаков операций. При этом создается область формулы, в которой появляется уголковый курсор, охватывающий текущий элемент формулы, например имя переменной (функции) или число. При вводе бинарного оператора по другую сторону знака операции автоматически появляется заполнитель в виде черного прямоугольника. В это место вводят очередной операнд.

Для управления порядком операций используют скобки, которые можно вводить вручную. Уголковый курсор позволяет автоматизировать такие действия. Чтобы выделить элементы формулы, которые в рамках операции должны рассматриваться как единое целое, используют клавишу ПРОБЕЛ. При каждом ее нажатии уголковый курсор «расширяется», охватывая элементы формулы, примыкающие к данному. После ввода знака операции элементы в пределах уголкового курсора автоматически заключаются в скобки.

Элементы формул можно вводить с клавиатуры или с помощью специальных панелей управления. Панели управления (рис. 18.2) открывают с помощью меню View (Вид) или кнопками панели управления Math (Математика). Для ввода элементов формул предназначены следующие панели:

- панель управления Calculator (Счет) для ввода чисел, знаков типичных математических операций и наиболее часто употребляемых стандартных функций;
- панель управления Evaluation (Вычисление) для ввода операторов вычисления;
- панель управления Boolean (Логика) для ввода знаков отношения и логических операций;

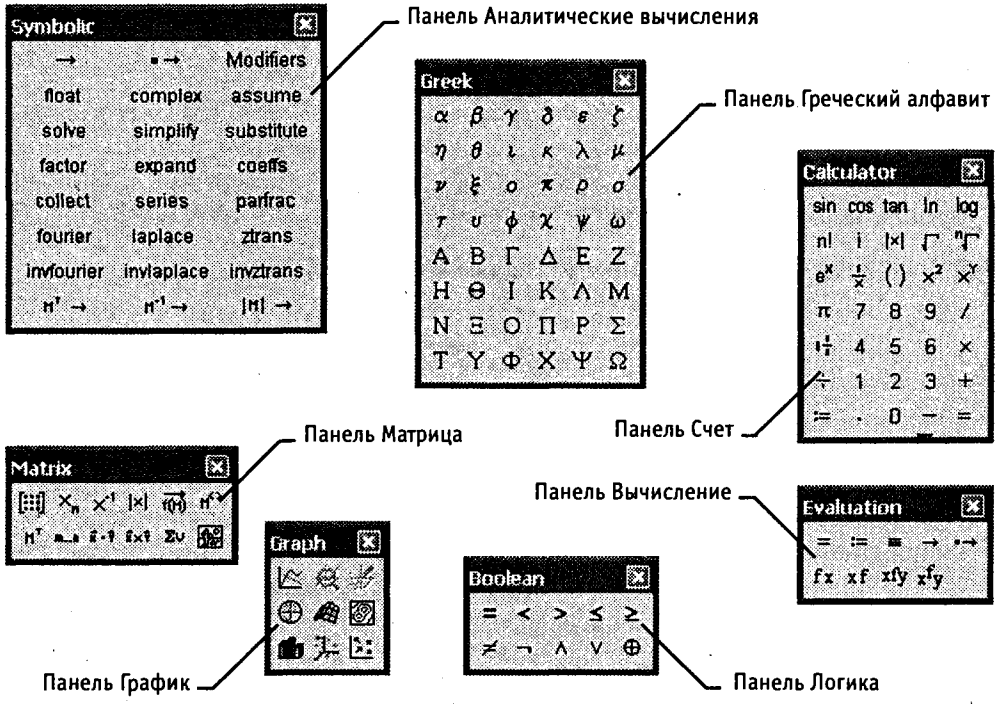


Рис. 18.2. Панели инструментов программы Mathcad для ввода формул

- панель управления Graph (График) для построения графиков;
- панель управления Matrix (Матрица) для ввода векторов и матриц и задания матричных операций;
- панель управления Calculus (Исчисление) для задания операций, относящихся к математическому анализу;
- панель управления Greek (Греческий алфавит) для ввода греческих букв (их можно также вводить с клавиатуры, если сразу после ввода соответствующего латинского символа нажимать сочетание клавиш CTRL+G, например [a][CTRL+G] — α, [W][CTRL+G] — Ω);
- панель управления Symbolic (Аналитические вычисления) для управления аналитическими преобразованиями.

Введенное выражение обычно вычисляют или присваивают переменной. Для вывода результата выражения используют знак вычисления, который выглядит как знак равенства и вводится при помощи кнопки Evaluate Numerically (Вычислить выражение) на панели инструментов Evaluation (Вычисление).

Знак присваивания изображается как «:=», а вводится при помощи кнопки Definition (Определить) на панели инструментов Evaluation (Вычисление). Слева от знака присваивания указывают имя переменной. Оно может содержать латинские и грече-

ские буквы, цифры, символы «′», «_» и «∞», а также описательный индекс. Описательный индекс вводится с помощью символа «.» и изображается как нижний индекс, но является частью имени переменной, например V_{init}. «Настоящие» индексы, определяющие отдельный элемент вектора или матрицы, задаются по-другому.

Переменную, которой присвоено значение, можно использовать далее в документе в вычисляемых выражениях. Чтобы узнать значение переменной, следует использовать оператор вычисления.

В следующем примере вычислена площадь круга с радиусом 2 (использованы переменные r и s, значение постоянной p определено в программе *Mathcad* по умолчанию).

$$r := 2 \quad s := \pi \cdot r^2 \quad s = 12.566$$

Ввод текста

Текст, помещенный в рабочий лист, содержит комментарии и описания и предназначен для ознакомления, а не для использования в расчетах. Программа *Mathcad* определяет назначение текущего блока автоматически при первом нажатии клавиши ПРОБЕЛ. Если введенный текст не может быть интерпретирован как формула, блок преобразуется в текстовый и последующие данные рассматриваются как текст. Создать текстовый блок без использования автоматических средств позволяет команда Insert ▸ Text Region (Вставка ▸ Текстовый блок).

Иногда требуется встроить формулу внутрь текстового блока. Для этого служит команда Insert ▸ Math Region (Вставка ▸ Формула).

Форматирование формул и текста

Для форматирования формул и текста в программе *Mathcad* используется панель инструментов Formatting (Форматирование). С ее помощью можно индивидуально отформатировать любую формулу или текстовый блок, задав гарнитуру и размер шрифта, а также полужирное, курсивное или подчеркнутое начертание символов. В текстовых блоках можно также задавать тип выравнивания и применять маркированные и нумерованные списки.

В качестве средств автоматизации используются стили оформления. Выбрать стиль оформления текстового блока или элемента формулы можно из списка Style (Стиль) на панели инструментов Formatting (Форматирование). Для формул и текстовых блоков применяются разные наборы стилей.

Чтобы изменить стиль оформления формулы или создать новый стиль, используется команда Format ▸ Equation (Формат ▸ Выражение). Изменение стандартных стилей Variables (Переменные) и Constants (Константы) влияет на отображение формул по всему документу. Стиль оформления имени переменной учитывается при ее определении. Так, переменные $\mathbf{x}$ и $\boldsymbol{x}$ рассматриваются как различные и не взаимозаменяемы.

При оформлении текстовых блоков можно использовать более широкий набор стилей. Настройка стилей текстовых блоков производится при помощи команды Format ▸ Style (Формат ▸ Стиль).

Работа с матрицами

Векторы и матрицы рассматриваются в программе *Mathcad* как одномерные и двумерные массивы данных. Число строк и столбцов матрицы задается в диалоговом окне Insert Matrix (Вставка матрицы), которое открывают командой Insert ▸ Matrix (Вставка ▸ Матрица). Вектор задается как матрица, имеющая один столбец.

После щелчка на кнопке OK в формулу вставляется матрица, содержащая вместо элементов заполнители. Вместо каждого заполнителя надо вставить число, переменную или выражение.

Для матриц определены следующие операции: сложение, умножение на число, перемножение и прочие. Допустимо использование матриц вместо скалярных выражений: в этом случае предполагается, что указанные действия должны быть применены к каждому элементу матрицы, и результат также представляется в виде матрицы. Например, выражение $M + 3$, где M — матрица, означает, что к каждому элементу матрицы прибавляется число 3. Если требуется явно указать необходимость поэлементного применения операции к матрице, используют знак векторизации, для ввода которого служит кнопка Vectorize (Векторизация) на панели инструментов Matrix (Матрица). Например:

$\begin{pmatrix}1 & 3\\ -1 & 1\end{pmatrix}\cdot\begin{pmatrix}1 & -1\\ 2 & 1\end{pmatrix}=\begin{pmatrix}7 & 2\\ 1 & 2\end{pmatrix}$ — обычное произведение матриц;

$\overrightarrow{\begin{pmatrix}1 & 3\\ -1 & 1\end{pmatrix}\cdot\begin{pmatrix}1 & -1\\ 2 & 1\end{pmatrix}}=\begin{pmatrix}1 & -3\\ -2 & 1\end{pmatrix}$ — поэлементное произведение матриц с использованием векторизации.

Для работы с элементами матрицы используют индексы элементов. Нумерация строк и столбцов матрицы начинается с нуля. Индекс элемента задается числом, переменной или выражением и отображается как нижний индекс. Он вводится после щелчка на кнопке Subscript (Индекс) на панели инструментов Matrix (Матрица).

Пара индексов, определяющих элемент матрицы, разделяется запятой. Иногда (например, при построении графиков) требуется выделить вектор, представляющий собой столбец матрицы. Номер столбца матрицы отображается как верхний индекс, заключенный в угловые скобки, например $M^{\langle 0\rangle}$. Для его ввода используется кнопка Matrix Column (Столбец) на панели инструментов Matrix (Матрица).

Чтобы задать общую формулу элементов матрицы, типа $M_{i,j} := i + j$, используют *диапазоны*. Диапазон фактически представляет собой вектор, содержащий арифметическую прогрессию, определенную первым, вторым и последним элементами. Чтобы задать диапазон, следует указать значение первого элемента, через запятую значение второго и через точку с запятой значение последнего элемента. Точка с запятой при задании диапазона отображается как две точки (..). Диапазон можно использовать как значение переменной, например $x := 0{,}0.01..1$.

Если разность прогрессии равна единице (то есть, элементы просто нумеруются), значение второго элемента и соответствующую запятую опускают. Например, чтобы сформировать по приведенной выше формуле матрицу размером 6×6, перед

этой формулой надо указать $i := 0..5$ $j := 0..5$. При формировании матрицы путем присвоения значения ее элементам размеры матрицы можно не задавать заранее. Всем неопределенным элементам автоматически присваиваются нулевые значения. Например, формула $M_{5,5} := 1$ создает матрицу M размером 6×6, у которой все элементы, кроме расположенного в правом нижнем углу, равны 0.

Стандартные и пользовательские функции

Произвольные зависимости между входными и выходными параметрами задаются при помощи функций. Функции принимают набор параметров и возвращают значение, скалярное или векторное (матричное). В формулах можно использовать стандартные встроенные функции, а также функции, определенные пользователем.

Чтобы использовать функцию в выражении, надо определить значения входных параметров в скобках после имени функции. Имена простейших математических функций можно ввести с панели инструментов Calculator (Счет). Информацию о других функциях можно почерпнуть в справочной системе. Вставить в выражение стандартную функцию можно при помощи команды Insert ▸ Function (Вставка ▸ Функция). В диалоговом окне Insert Function (Вставка функции) слева выбирается категория, к которой относится функция, а справа — конкретная функция. В нижней части окна выдается информация о выбранной функции. При вводе функции через это диалоговое окно автоматически добавляются скобки и заполнители для значений параметров.

Пользовательские функции должны быть сначала определены. Определение задается при помощи оператора присваивания. В левой части указывается имя пользовательской функции и, в скобках, формальные параметры — переменные, от которых она зависит. Справа от знака присваивания эти переменные должны использоваться в выражении. При использовании пользовательской функции в последующих формулах ее имя вводят вручную. В диалоговом окне Insert Function (Вставка функции) оно не отображается.

Решение уравнений и систем

Для численного поиска корней уравнения в программе *Mathcad* используется функция *root*. Она служит для решения уравнений вида $f(x) = 0$, где $f(x)$ — выражение, корни которого нужно найти, а x — неизвестное. Для поиска корней с помощью функции *root* надо присвоить искомой переменной начальное значение, а затем вычислить корень при помощи вызова функции:

$$\text{root}(f(x),x)$$

Здесь $f(x)$ — функция переменной x, используемой в качестве второго параметра. Функция *root* возвращает значение независимой переменной, обращающее функцию $f(x)$ в 0. Например:

$$x := 1$$
$$\text{root}(2\cdot\sin(x) - x,x) = 1.895$$

Если уравнение имеет несколько корней (как в данном примере), то результат, выдаваемый функцией *root*, зависит от выбранного начального приближения.

Если надо решить систему уравнений (неравенств), используют так называемый *блок решения*, который начинается с ключевого слова *given* (*дано*) и заканчивается вызовом функции *find* (*найти*). Между ними располагают «логические утверждения», задающие ограничения на значения искомых величин, иными словами, уравнения и неравенства. Всем переменным, используемым для обозначения неизвестных величин, должны быть заранее присвоены начальные значения.

Чтобы записать уравнение, в котором утверждается, что левая и правая части равны, используется знак логического равенства — кнопка Equal to (Равно) на панели инструментов Boolean (Логика). Другие знаки логических условий также можно найти на этой панели.

Заканчивается блок решения вызовом функции *find*, у которой в качестве аргументов должны быть перечислены искомые величины. Эта функция возвращает вектор, содержащий вычисленные значения неизвестных. Например:

$x := 0 \quad y := 0$

given

$x + y = 1$

$x^2 + y^2 = 4$

$$\text{find}(x,y) = \begin{pmatrix} 1{,}823 \\ 0{,}823 \end{pmatrix}$$

Построение графиков

Чтобы построить двумерный график в координатных осях X–Y, надо дать команду Insert ▸ Graph ▸ X-Y Plot (Вставка ▸ График ▸ Декартовы координаты). В области размещения графика находятся заполнители для указания отображаемых выражений и диапазона изменения величин. Заполнитель у середины оси координат предназначен для переменной или выражения, отображаемого по этой оси. Обычно используют диапазон или вектор значений. Граничные значения по осям выбираются автоматически в соответствии с диапазоном изменения величины, но их можно задать и вручную.

В одной графической области можно построить несколько графиков. Для этого надо у соответствующей оси перечислить несколько выражений через запятую.

Разные кривые изображаются разным цветом, а для форматирования графика надо дважды щелкнуть на области графика. Для управления отображением построенных линий служит вкладка Traces (Линии) в открывшемся диалоговом окне. Текущий формат каждой линии приведен в списке, а под списком расположены элементы управления, позволяющие изменять формат. Поле Legend Label (Описание) задает описание линии, которое отображается только при сбросе флажка Hide Legend (Скрыть описание). Список Symbol (Символ) позволяет выбрать маркеры для отдельных точек, список Line (Тип линии) задает тип линии, список Color (Цвет) — цвет. Список Type (Тип) определяет способ связи отдельных точек, а список Weight (Толщина) — толщину линии.

Точно так же можно построить и отформатировать график в полярных координатах. Для его построения надо дать команду Insert ▸ Graph ▸ Polar Plot (Вставка ▸ График ▸ Полярные координаты).

Для построения простейшего трехмерного графика, необходимо задать матрицу значений. Отобразить эту матрицу можно в виде поверхности — Insert ▸ Graph ▸ Surface Plot (Вставка ▸ График ▸ Поверхность), столбчатой диаграммы — Insert ▸ Graph ▸ 3D Bar Plot (Вставка ▸ График ▸ Столбчатая диаграмма) или линий уровня — Insert ▸ Graph ▸ Contour Plot (Вставка ▸ График ▸ Линии уровня).

Для отображения векторного поля при помощи команды Insert ▸ Graph ▸ Vector Field Plot (Вставка ▸ График ▸ Поле векторов) значения матрицы должны быть комплексными. В этом случае в каждой точке графика отображается вектор с координатами, равными действительной и мнимой частям элемента матрицы. Во всех этих случаях после создания области графика необходимо указать вместо заполнителя имя матрицы, содержащей необходимые значения.

Для построения параметрического точечного графика командой Insert ▸ Graph ▸ 3D Scatter Plot (Вставка ▸ График ▸ Точки в пространстве) необходимо задать три вектора с одинаковым числом элементов, которые соответствуют x-, y- и z-координатам точек, отображаемых на графике. В области графика эти три вектора указываются внутри скобок через запятую.

Аналогичным образом можно построить поверхность, заданную параметрически. Для этого надо задать три матрицы, содержащие, соответственно, x-, y- и z-координаты точек поверхности. Теперь надо дать команду построения поверхности Insert ▸ Graph ▸ Surface Plot (Вставка ▸ График ▸ Поверхность) и указать в области графика эти три матрицы в скобках и через запятую. Таким образом можно построить практически любую криволинейную поверхность (например, представленную на рис. 18.3), в том числе с самопересечениями.

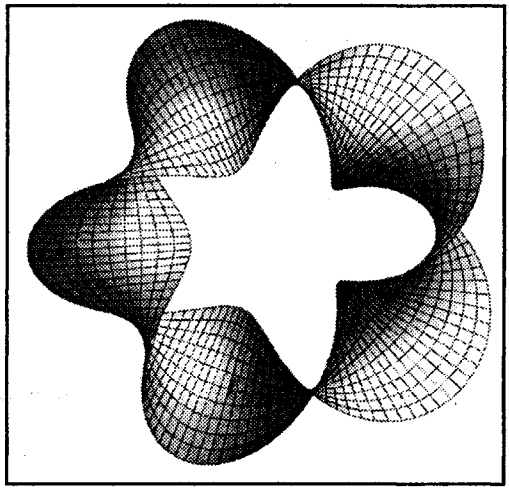


Рис. 18.3. Пятикратно перекрученная замкнутая лента, заданная параметрически

Диалоговое окно для форматирования трехмерных графиков также открывают двойным щелчком на области графика.

Аналитические вычисления

С помощью аналитических вычислений находят аналитические или полные решения уравнений и систем, а также проводят преобразования сложных выражений (например, упрощение). Иначе говоря, при таком подходе можно получить нечисловой результат. В программе *Mathcad* конкретные значения, присвоенные переменным, при этом игнорируются — переменные рассматриваются как неопределенные параметры. Команды для выполнения аналитических вычислений в основном сосредоточены в меню Symbolics (Аналитические вычисления).

Чтобы упростить выражение (или часть выражения), надо выбрать его при помощи уголкового курсора и дать команду Symbolics ▸ Simplify (Аналитические вычисления ▸ Упростить). При этом выполняются арифметические действия, сокращаются общие множители и приводятся подобные члены, применяются тригонометрические тождества, упрощаются выражения с радикалами, а также выражения, содержащие прямую и обратную функции (типа $e^{\ln x}$). Некоторые действия по раскрытию скобок и упрощению сложных тригонометрических выражений требуют применения команды Symbolics ▸ Expand (Аналитические вычисления ▸ Раскрыть).

Команду Symbolics ▸ Simplify (Аналитические вычисления ▸ Упростить) применяют и в более сложных случаях. Например, с ее помощью можно:

- вычислить предел числовой последовательности, заданной общим членом;
- найти общую формулу для суммы членов числовой последовательности, заданной общим членом;
- вычислить производную данной функции;
- найти первообразную данной функции или значение определенного интеграла.

Другие возможности меню Symbolics (Аналитические вычисления) состоят в выполнении аналитических операций, ориентированных на переменную, использованную в выражении. Для этого надо выделить в выражении переменную и выбрать команду из меню Symbolics ▸ Variable (Аналитические вычисления ▸ Переменная). Команда Solve (Решить) ищет корни функции, заданной данным выражением, например, если выделить уголковым курсором переменную x в выражении $a \cdot x^2 + b \cdot x + c$, то в результате применения команды Symbolics ▸ Variable ▸ Solve (Аналитические вычисления ▸ Переменная ▸ Решить), будут найдены все корни:

$$\begin{bmatrix} \frac{1}{(2 \cdot a)} \cdot \left[-b + \sqrt{(b^2 - 4 \cdot a \cdot c)} \right] \\ \frac{1}{(2 \cdot a)} \cdot \left[-b - \sqrt{(b^2 - 4 \cdot a \cdot c)} \right] \end{bmatrix}$$

Другие возможности использования этого меню включают:

- аналитическое дифференцирование и интегрирование: Symbolics ▸ Variable ▸ Differentiate (Аналитические вычисления ▸ Переменная ▸ Дифференцировать) и Symbolics ▸ Variable ▸ Integrate (Аналитические вычисления ▸ Переменная ▸ Интегрировать);
- замена переменной: Symbolics ▸ Variable ▸ Substitute (Аналитические вычисления ▸ Переменная ▸ Подставить) — вместо переменной подставляется содержимое буфера обмена;
- разложение в ряд Тейлора: Symbolics ▸ Variable ▸ Expand to Series (Аналитические вычисления ▸ Переменная ▸ Разложить в ряд);
- представление дробно-рациональной функции в виде суммы простых дробей с линейными и квадратичными знаменателями: Symbolics ▸ Variable ▸ Convert to Partial Fraction (Аналитические вычисления ▸ Переменная ▸ Преобразовать в простые дроби).

Наконец, самым мощным инструментом аналитических вычислений является оператор аналитического вычисления, который вводится с помощью кнопки Evaluate Symbolically (Вычислить аналитически) на панели инструментов Evaluation (Вычисление). Его можно, например, использовать для аналитического решения системы уравнений и неравенств. Блок решения задается точно так же, как при численном решении (хотя начальные значения переменных можно не задавать), а последняя формула блока должна выглядеть как *find(x, y...)*®, где в скобках приведен список искомых величин, а далее следует знак аналитического вычисления, отображаемый в виде стрелки, направленной вправо.

Любое аналитическое вычисление можно применить с помощью ключевого слова. Для этого используют кнопку Symbolic Keyword Evaluation (Вычисление с ключевым словом) на панели инструментов Evaluation (Вычисление). Ключевые слова вводятся через панель инструментов Symbolics (Аналитические вычисления). Они полностью охватывают возможности, заключенные в меню Symbolics (Аналитические вычисления), позволяя также задавать дополнительные параметры.

Практическое занятие

Упражнение 18.1. Простые вычисления с использованием программы Mathcad

30 мин

Задача. Найти ребро куба, равновеликого шару, площадь поверхности которого равна площади боковой поверхности прямого кругового конуса, у которого высота вдвое меньше, чем длина образующей. Объем этого конуса равен 1.

Анализ. Основные геометрические формулы, используемые при расчете.

Объем конуса — $V = \frac{1}{3}\pi r^2 h$.

Площадь боковой поверхности конуса — $S = \pi r l$.

Соотношение в конусе между радиусом основания, высотой и длиной образующей — $r^2 + h^2 = l^2$.

Площадь поверхности шара — $V = 4\pi R^2$.

Объем шара — $V = \frac{4}{3}\pi r^3$. Объем куба — $V = a^3$.

1. Запустите программу *Mathcad* через Главное меню (Пуск ▸ Программы ▸ MathSoft Apps ▸ Mathcad).
2. Откройте панель инструментов Calculator (Счет) щелчком на кнопке Calculator Toolbar (Панель инструментов Счет) на панели инструментов Math (Математика) или с помощью команды View ▸ Toolbars ▸ Calculator (Вид ▸ Панели инструментов ▸ Счет).
3. Для удобства расчета будем обозначать каждую из вычисляемых величин отдельной переменной. Объем конуса обозначим как V и присвоим ему значение 1. Оператор присваивания вводится символом «:» или кнопкой Definition

(Определить) на панели инструментов Calculator (Счет). Итак, надо ввести V:1. В документе появится полноценный оператор присваивания.

$$V := 1$$

4. Путем несложных преобразований получим, что радиус основания конуса можно вычислить по формуле

$$r = \sqrt[3]{\frac{V \cdot \sqrt{3}}{\pi}}.$$

Вводить эту формулу следует слева направо. Порядок ввода этой формулы следующий. Сначала введите знак корня произвольной степени: кнопка Nth Root (Корень данной степени) на панели инструментов Calculator (Счет) или комбинация клавиш CTRL+\. Щелкните на черном квадратике, стоящем на месте показателя степени, и введите цифру 3. Щелкните на квадратике, замещающем подкоренное выражение, нажмите клавиши V*. Введите знак квадратного корня: кнопка Square Root (Квадратный корень) на панели инструментов Calculator (Счет) или клавиша \ — и цифру 3. Прежде чем вводить знаменатель, дважды нажмите клавишу ПРОБЕЛ. Обратите внимание на синий уголок, который указывает на текущее выражение. Предполагается, что знак операции связывает выбранное выражение со следующим. В данном случае это безразлично, но в целом этот прием позволяет вводить сложные формулы, избегая ручного ввода дополнительных скобок. Нажмите клавишу /. Чтобы ввести число π, можно воспользоваться комбинацией клавиш CTRL+SHIFT+P или соответствующей кнопкой на панели инструментов Calculator (Счет).

На экране появится следующая надпись: $r = \sqrt[3]{\frac{V \cdot \sqrt{3}}{\pi}}$.

5. Введите формулы для вычисления длины образующей и площади боковой поверхности конуса:

$$l = \frac{r \cdot 2}{\sqrt{3}}; \quad S = \pi \cdot r \cdot l.$$

Указание знака умножения между переменными обязательно, так как иначе программа *Mathcad* сочтет, что указана одна переменная с именем из нескольких букв.

6. Для вычисления радиуса шара R введите формулу $R = \sqrt{\frac{S}{4\pi}}$.

7. Для вычисления объема шара введите формулу $W = \frac{4}{3}\pi R^3$.

Использовать переменную V во второй раз не следует, так как теперь мы определяем совершенно другой объем.

8. Заключительная формула $a = \sqrt[3]{W}$ позволит получить окончательный результат. После этого снова наберите имя переменной a и нажмите клавишу = или щелкните на кнопке Evaluate Expression (Вычислить выражение) на панели инст-

рументов Calculator (Счет). После формулы появится знак равенства и вычисленный результат.

$$a = 0.7102$$

Вычислять можно как действительные, так и комплексные выражения. Обозначение мнимой единицы (i) следует вводить непосредственно после числового коэффициента, который нельзя опускать, даже если он равен единице.

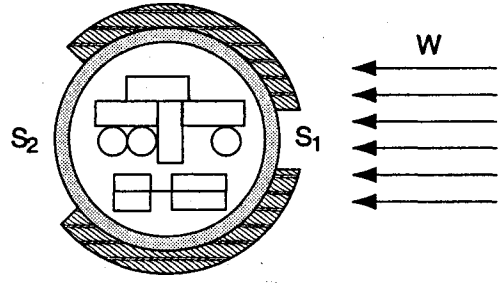
9. Вернитесь к самому первому выражению и отредактируйте его. Вместо значения 1 присвойте переменной значение 8. Сразу же перейдите к последней введенной формуле и обратите внимание, что результат расчета сразу же стал отражать новые начальные данные.

Мы познакомились с методикой простейших вычислений в программе MathCad. Описанная техника позволяет использовать эту программу как «интеллектуальный калькулятор» для автоматического расчета по известным формулам. Особенностью программы Mathcad является возможность практически мгновенного перерасчета с другими начальными данными.

Упражнение 18.2. Физические вычисления с использованием единиц измерения

30 мин

Постановка задачи. Теплоизолированный космический аппарат, находящийся на орбите Земли, имеет на борту приборы с электрической мощностью, которая может изменяться в ходе работы от $N_1 = 75$ Вт (дежурный режим) до $N_2 = 200$ Вт (сеанс связи). С целью обеспечения предсказуемого теплового режима в теплоизоляции сделано отверстие площадью S_1, на которое попадает поток солнечной энергии $W = 1400$ Вт/м^2. Полученная энергия излучается аппаратом через это и дополнительное отверстие в теплоизоляции с площадью S_2 в режиме «черного тела». Каковы должны быть площади отверстий, если допустимый диапазон температур для оборудования, расположенного в аппарате, составляет 20–30°С?



Анализ задачи. Минимальная температура аппаратуры соответствует режиму минимального тепловыделения. В этом случае поступающая мощность $Q_1 = WS_1 + N_1$. Излучаемая мощность $Q_2 = \sigma T_1^4 (S_1 + S_2)$, где T_1 — минимальная допустимая температура в градусах Кельвина. В условиях теплового баланса эти мощности должны быть равны.

Режим максимального тепловыделения соответствует максимальной температуре аппаратуры. В этом случае $WS_1 + N_2 = \sigma T_2^4 (S_1 + S_2)$.

Используя два полученных уравнения, получаем:

$$S_1 = \frac{(N_2 T_1^4 - N_1 T_2^4)}{W \cdot (T_2^4 - T_1^4)}; \qquad S_2 = \frac{W \cdot (N_2 - N_1) - \sigma \cdot (N_2 T_1^4 - N_1 T_2^4)}{\sigma W \cdot (T_2^4 - T_1^4)}$$

1. Запустите программу *Mathcad*.
2. Введите значения известных величин, присвоив их переменным с соответствующими именами. Вместо нижних индексов используйте просто дополнительную цифру в названии переменной.

$$W := 1400\ \frac{\text{watt}}{\text{m}^2}$$

$$N1 := 75\cdot\text{watt}\quad N2 := 200\cdot\text{watt}$$

$$T1 := (20 + 273)\cdot\text{K}\quad T2 := (30 + 273)\cdot\text{K}$$

3. Обозначения физических единиц присоединяйте к соответствующим значениям через знак умножения. Если нужное обозначение неизвестно, используйте команду Insert ▸ Unit (Вставка ▸ Единица измерения). Измеряемая величина выбирается в списке Dimension (Размерность), а нужная единица измерения — в списке Unit (Единица измерения).
4. Присвойте переменной σ значение постоянной Стефана-Больцмана $(5{,}67\cdot 10^{-8}\ \frac{\text{Вт}}{\text{м}^2\cdot\text{К}^4})$.

   Чтобы ввести греческую букву, используйте панель инструментов Greek (Греческий алфавит) или введите соответствующую латинскую букву (в данном случае «*s*») и сразу же нажмите комбинацию клавиш CTRL+G. Так как специальной единицы для размерности этой константы не существует, ее следует составить из стандартных единиц с помощью умножения и деления.
5. Введите полученные в ходе анализа формулы для вычисления площадей отверстий, присвоив полученные значения переменным *S*1 и *S*2.

$$S1 := \frac{(N2\cdot T1^4 - N1\cdot T2^4)}{W\cdot(T2^4 - T1^4)};\qquad S2 := \frac{W\cdot(N2 - N1) - \sigma\cdot(N2\cdot T1^4 - N1\cdot T2^4)}{\sigma W\cdot(T2^4 - T1^4)}$$

6. Изменение значений параметров, заданных в условии задачи, приводит к автоматическому перерасчету формул. В частности, исследуйте, изменяя значение переменной *W*, как изменяются требования к такому методу терморегуляции при удалении аппарата от Солнца и приближении к нему

   $W = 2700\ \frac{\text{Вт}}{\text{м}^2}$; на орбите Марса $W = 500\ \frac{\text{Вт}}{\text{м}^2}$).
7. Обратите внимание, что результат содержит единицы измерения в соответствии с системой единиц СИ. Используемая система единиц отображается в диалоговом окне Insert Unit (Вставка единиц измерения).
8. Чтобы изменить используемую систему единиц, дайте команду Math ▸ Options (Математика ▸ Параметры) и в открывшемся диалоговом окне Math Options (Параметры расчета) выберите вкладку Unit System (Система единиц). Выберите систему *CGS* и посмотрите, как изменились результаты (они теперь выражаются в квадратных сантиметрах). Если, например, выбрать американскую систему единиц (*U.S.*), то результат будет выражен в квадратных футах.

Мы научились производить вычисления с использованием реальных размерных физических величин, а также производить преобразование данных из одной системы единиц в другую. Это позволяет немедленно получать результат в наиболее удобной форме.

Упражнение 18.3. Векторы и матрицы

Задача. Разложить вектор $\vec{V}=\begin{bmatrix}1\\1\\1\end{bmatrix}$ по нормированным собственным векторам матрицы $M=\begin{bmatrix}1&2&2\\2&6&1\\2&1&1\end{bmatrix}$.

Анализ. Первый этап решения задачи состоит в нахождении собственных значений и собственных векторов данной матрицы. Затем необходимо найти вектор $\vec{T}$, такой, что $S\cdot\vec{T}=\vec{V}$, где S — матрица, столбцы которой представляют собой собственные вектора матрицы M.

1. Запустите программу *Mathcad*.
2. Создайте матрицу M. Начните запись оператора присваивания, а для ввода правой части нажмите комбинацию клавиш CTRL+M, воспользуйтесь командой Insert ▸ Matrix (Вставка ▸ Матрица) или щелкните на кнопке Matrix or Vector (Матрица или вектор) на панели инструментов Matrix (Матрица).
3. В открывшемся диалоговом окне Insert Matrix (Вставка матрицы) укажите число строк и столбцов (по три) и щелкните на кнопке OK.
4. Введите значения элементов матрицы в отведенные места.
5. Аналогичным образом сформируйте вектор $\vec{V}$. Он будет представлять собой матрицу, имеющую только один столбец.
6. Собственные значения квадратной матрицы можно получить при помощи функции *eigenvals*. Результатом ее работы является вектор собственных значений, присвойте его переменной L.
7. Функция *eigenvec* позволяет получить собственный вектор, соответствующий данному собственному значению. Ей нужны два параметра: матрица, для которой ищется собственный вектор, и собственное значение, которому он соответствует. Чтобы записать собственные вектора в качестве столбцов матрицы S, надо присвоить вычисленное значение столбцу матрицы. Столбцы матрицы в программе *Mathcad* выбираются специальным верхним индексом, заключенным в угловые скобки. Чтобы ввести номер столбца, нажмите комбинацию клавиш CTRL+6 или щелкните на кнопке Matrix Column (Столбец) на панели инструментов Matrix (Матрица), после чего введите номер нужного столбца матрицы. Будьте внимательны — столбцы и строки матрицы нумеруются начиная с нуля.

$S^{<0>}$

8. В правой части оператора присваивания надо указать собственное значение матрицы. Собственные значения являются элементами вектора L. Номер элемента указывается как нижний индекс. Для ввода нижнего индекса нажмите клавишу [или воспользуйтесь кнопкой Subscript (Индекс) на панели инструментов Matrix (Матрица). Итоговый оператор для первого собственного вектора будет выглядеть следующим образом:

$$S^{<0>} := \text{eigenvec}(M, L_0)$$

Аналогично задайте операторы для второго и третьего собственных значений.

9. Для нахождения коэффициентов при собственных векторах в разложении необходимо решить систему линейных уравнений. Ее удобно записать в матричной форме. Создайте вектор T с тремя элементами. Величины этих элементов значения не имеют.
10. Запишите ключевое слово *given*.
11. Ниже запишите матричное уравнение $S \cdot T = V$. Знак логического равенства введите с помощью комбинации клавиш CTRL+=.
12. Найдите коэффициенты в разложении при помощи функции *find*.

$$\text{find}(T) = \begin{bmatrix} 0{,}836 \\ 0{,}334 \\ 0{,}148 \end{bmatrix}$$

Мы научились производить операции с векторами и матрицами, использовать соответствующие функции, выделять столбцы матриц и отдельные элементы. Матричная запись часто позволяет представить задачу в более удобной форме.

Упражнение 18.4. Аналитические вычисления

45 мин

Задача 1. На приведенной схеме сопротивление RR является переменным. Определить, как меняется ток между точками A и B в зависимости от величины этого сопротивления.

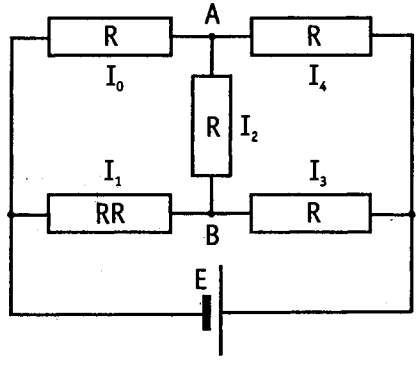


Рис. 18.4. Исследуемая электрическая схема

Анализ. Перенумеровав сопротивления в указанном порядке и воспользовавшись законами Кирхгофа, получим систему уравнений, позволяющую найти величины токов.

$$\begin{cases} I_0 + I_2 = I_4 \\ I_1 = I_2 + I_3 \\ RR \cdot I_1 + R \cdot I_2 - R \cdot I_0 = 0 \\ R \cdot I_2 + R \cdot I_4 - R \cdot I_3 = 0 \\ R \cdot I_0 + R \cdot I_4 = E \end{cases}$$

Эту систему надо решить, не подставляя конкретных значений вместо параметров *R*, *RR* и *E*.

1. Запустите программу *Mathcad*.
2. Введите ключевое слово *given*.
3. Введите уравнения системы, полученной в ходе анализа. Обозначьте неизвестные токи переменными *I*0, *I*1, *I*2, *I*3, *I*4. Фиксированное сопротивление *R* обозначьте переменной *R*0. Обратите внимание, что присваивать начальные значения токов или задавать значения переменных *R*0, *RR* и *E* не требуется.
4. Введите функцию *find*, перечислив в качестве параметров неизвестные *I*0, *I*1, *I*2, *I*3, *I*4. Затем введите оператор аналитического вычисления, который выглядит как стрелка, направленная вправо, и вводится комбинацией клавиш CTRL+. или кнопкой Evaluate Symbolically (Вычислить аналитически) на панели инструментов Evaluation (Вычисление).
5. Щелкните за пределами данного блока, и программа *Mathcad* произведет аналитическое решение системы уравнений.

$$\text{find}(I0, I1, I2, I3, I4) \rightarrow \begin{bmatrix} E \cdot \dfrac{(3 \cdot RR + R0)}{(R0 \cdot (5 \cdot RR + 3 \cdot R0))} \\ 4 \cdot \dfrac{E}{(5 \cdot RR + 3 \cdot R0)} \\ E \cdot \dfrac{(-RR + R0)}{(R0 \cdot (5 \cdot RR + 3 \cdot R0))} \\ E \cdot \dfrac{(RR + 3 \cdot R0)}{(R0 \cdot (5 \cdot RR + 3 \cdot R0))} \\ 2 \cdot E \cdot \dfrac{(RR + R0)}{(R0 \cdot (5 \cdot RR + 3 \cdot R0))} \end{bmatrix}$$

Полученный результат позволяет провести полный анализ схемы.

Задача 2. Найти все корни уравнения:

$$(1 + y - y^2)^2 + y = 2$$

Анализ. Это уравнение четвертого порядка. Легко подобрать один корень ($y = 1$). Остающееся уравнение третьего порядка не имеет рациональных корней, так что поиск других корней этого уравнения — дело непростое. Неясно даже, сколько еще действительных корней имеет данное уравнение. Результаты численного решения зависят от подбора начального приближения и поэтому не гарантируют отыскания всех корней уравнения. Мы же решим это уравнение аналитически.

6. Введите заданное уравнение. Чтобы раскрыть скобки, дайте команду Symbolics ▸ Simplify (Аналитические вычисления ▸ Упростить).

7. Выделите в полученном уравнении независимую переменную (в данном случае y) и дайте команду Symbolics ▸ Variable ▸ Solve (Аналитические вычисления ▸ Переменная ▸ Решить).

   Программа *Mathcad* выдаст вектор, элементами которого являются корни данного уравнения.

8. Полученный результат содержит сложные комплексные выражения, и его невозможно применить с пользой (все еще непонятно, являются ли корни действительными или комплексными). Чтобы разделить действительную и мнимую части, выделите результат вычисления целиком и дайте команду Symbolics ▸ Evaluate ▸ Complex (Аналитические вычисления ▸ Вычислить ▸ В комплексном виде). Если программа *Mathcad* не справится с преобразованием всего набора корней целиком, выполните преобразование корней поочередно: дайте команду Symbolics ▸ Evaluate ▸ Complex (Аналитические вычисления ▸ Вычислить ▸ В комплексном виде), поочередно выделив каждый из корней, записанных в комплексном виде.

9. Теперь полученное выражение надо упростить. Выделив его целиком, дайте команду Symbolics ▸ Simplify (Аналитические вычисления ▸ Упростить). Выражение станет существенно проще, причем станет понятно, что все корни уравнения действительные (все мнимые компоненты сократятся).

10. Последний шаг — раскрытие скобок, в данном случае упрощение аргументов тригонометрических функций. Для этого примените команду Symbolics ▸ Expand (Аналитические вычисления ▸ Раскрыть). Полученная запись — наилучшее представление точного решения, которое можно получить с помощью программы *Mathcad*.

11. Чтобы получить результат в числовом виде, достаточно ввести в конце выражения (итогового или на любой из предыдущих стадий) команду вычисления (=).

Мы научились использовать программу Mathcad для выполнения аналитических вычислений. Это позволяет получать точные решения задач, содержащих переменные параметры, анализировать полученные результаты, а также получать полный набор решений для некоторых типов уравнений.

$$\begin{bmatrix} 1 \\ \frac{1}{3}\cdot 7^{\frac{1}{2}}\cdot\cos\left(\frac{1}{3}\cdot\operatorname{atan}\left(3\cdot 3^{\frac{1}{2}}\right)\right)+\frac{1}{3}\cdot 7^{\frac{1}{2}}\cdot\sin\left(\frac{1}{3}\cdot\operatorname{atan}\left(3\cdot 3^{\frac{1}{2}}\right)\right)\cdot 3^{\frac{1}{2}}+\frac{1}{3} \\ -\frac{2}{3}\cdot 7^{\frac{1}{2}}\cdot\cos\left(\frac{1}{3}\cdot\operatorname{atan}\left(3\cdot 3^{\frac{1}{2}}\right)\right)+\frac{1}{3} \\ \frac{1}{3}\cdot 7^{\frac{1}{2}}\cdot\cos\left(\frac{1}{3}\cdot\operatorname{atan}\left(3\cdot 3^{\frac{1}{2}}\right)\right)-\frac{1}{3}\cdot 7^{\frac{1}{2}}\cdot\sin\left(\frac{1}{3}\cdot\operatorname{atan}\left(3\cdot 3^{\frac{1}{2}}\right)\right)\cdot 3^{\frac{1}{2}}+\frac{1}{3} \end{bmatrix} = \begin{bmatrix} 1 \\ 1{,}802 \\ -1{,}247 \\ 0{,}445 \end{bmatrix}$$

Упражнение 18.5. Анализ результатов испытаний

Задача. К пружине последовательно подвешивали грузы массой 1, 2, 3, ..., 20 кг. В результате был получен список величин удлинения пружины (в миллиметрах). Определить основные статистические параметры полученного набора измерений. Рассчитать жесткость пружины и массу узла, использованного для крепления грузов к пружине, воспользовавшись методом наименьших квадратов.

Таблица измерений:

Вес, кг	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Растяжение, мм	3,5	6,9	9,2	12,2	13,5	173	22,1	24,2	27,8	29,6	31,8	37,6	39,6	42,8	45,5	46,6	52,1	52,5	56,7	62,4

Анализ. Для решения этой задачи достаточно использовать стандартные средства статистических вычислений, имеющиеся в программе *Mathcad*. Теоретически, растяжение пружины определяется формулой $k \cdot x = (m + m_0) \cdot g$. Если определить статистическими методами коэффициенты a и b в уравнении $x = a \cdot m + b$, то получим:

$$k = \frac{g}{a},\ m_0 = \frac{b}{a}.$$

1. Запустите программу *Mathcad*.
2. Введите таблицу данных, предназначенных для статистического анализа, как матрицу с двумя столбцами, первый из которых содержит веса грузов, а второй — значения растяжения пружины.
3. Определите число точек в наборах данных с помощью функции *rows*.

   $n := \text{rows(data)}\quad n = 20$

4. Вычислите среднее растяжение пружины в ходе эксперимента с помощью функции *mean*.

   $Y := \text{data}^{<1>}\quad \text{mean}(Y) = 31.695$

5. Вычислите медиану значений растяжения пружины при помощи функции *median*.

   $\text{median}(Y) = 30.7$

6. Вычислите среднеквадратичное отклонение и дисперсию величины растяжения пружины при помощи функции *stdev*.

   $\text{stdev}(Y) = 17.39625 \quad \text{stdev}(Y)^2 = 302.62947$

7. Определите коэффициенты линейного уравнения являющегося наилучшим приближением для данных наборов данных. Функция *slope* позволяет вычислить коэффициент наклона прямой, а функция *intercept* — свободный член.

   $X := \text{data}^{<0>}$

   $b_0 := \text{intercept}(X,Y) \quad b_0 = 0.07421$

   $b_1 := \text{slope}(X,Y) \quad b_1 = 3.0115$

8. Определите жесткость пружины.

   $k = 3{,}254 \cdot 10^3$ (Н/м)

9. Определите массу узла крепления.

   $m = 24{,}64$ (г)

10. Сохраните созданный документ для использования в следующем упражнении.

Мы научились применять функции, используемые для статистического анализа данных. Программа Mathcad содержит и другие функции аналогичного назначения, которые можно использовать для интерполяции и экстраполяции данных, а также их сглаживания.

Упражнение 18.6. Построение графиков

30 мин

Задача. Используя результаты, полученные в предыдущем упражнении, построить график, отображающий экспериментальные данные и аппроксимирующую зависимость. Построить другой график, отображающий величину отклонения экспериментальных значений от аппроксимирующей прямой.

Анализ. Для построения графика можно использовать функцию, заданную набором данных или формулой. Формулы для функций, полученных в результате проделанных расчетов, необходимо определить, прежде чем их можно будет использовать при построении графика.

1. Запустите программу *Mathcad*.
2. Загрузите документ, созданный в предыдущем упражнении.
3. Переместите точку ввода в нижнюю часть документа.
4. Запишите формулу функции $r(x)$ для определения координат точек, лежащих на аппроксимирующей прямой. Коэффициенты соответствующего уравнения были получены в предыдущем упражнении.

   $r(x) := b_0 + b_1 \cdot x$

5. Нажмите клавишу @, щелкните на кнопке X-Y Plot (Декартовы координаты) на панели инструментов Graph (График) или дайте команду Insert ▸ Graph ▸ X-Y Plot (Вставка ▸ График ▸ Декартовы координаты). В документе появится область для создания графика.
6. Вместо заполнителя в нижней части графика укажите в качестве независимой переменной первый столбец матрицы data (data<0> или X).
7. Вместо заполнителя слева от графика укажите, что по вертикальной оси должны откладываться значения из второго столбца матрицы data и определенная выше линейная функция $r(X)$. В качестве разделителя используется запятая. Диапазон значений для осей координат выбирается программой *Mathcad* автоматически.
8. Чтобы изменить вид автоматически построенного графика, дважды щелкните внутри него. Откроется диалоговое окно Formatting Currently Selected X-Y Plot (Форматирование графика в декартовых координатах). Первая запись в списке на вкладке Traces (Кривые) соответствует первой отображенной кривой. Для изменения записи используются поля под списком.\

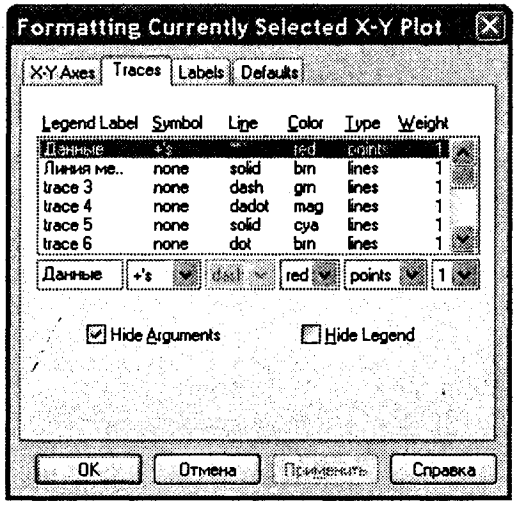


Рис. 18.5. Задание способа отображения линий графика

9. Под столбцом Legend Label (Подпись) введите название графика.
10. В раскрывающемся списке под столбцом Symbol (Маркер) выберите способ обозначения для отдельных точек.
11. Под столбцом Type (Вид линии) укажите, что необходимо пометить отдельные точки (points), а не провести непрерывную линию.
12. Выберите в списке вторую кривую и настройте ее отображение по своему вкусу.
13. Установите флажок Hide Arguments (Скрыть параметры), чтобы не отображать названия осей.

14. Сбросьте флажок Hide Legend (Скрыть подписи), чтобы включить отображение под графиком заданных подписей кривых.

15. В поле Title (Заголовок) на вкладке Labels (Надписи) задайте название графика и включите режим его отображения: флажок Show Title (Показать заголовок).

16. Постройте график, на котором отображалась бы величина отклонения экспериментальных точек от линии приближения $Y - (b_0 + b_1 \cdot X)$. Отформатируйте его, используя те же средства, что и в предыдущем случае.

Заголовок и подписи, использующие русские буквы, могут отображаться неправильно. Коррекцию обеспечивает выбор шрифта, правильно воспроизводящего кириллицу. Дайте команду Format ▸ Equation (Формат ▸ Выражение), в раскрывающемся списке Style Name (Имя стиля) выберите пункт Variables (Переменные) и щелкните на кнопке Modify (Изменить). Для задания шрифта используйте поле со списком Шрифт.

Упражнение 18.7. Построение трехмерных графиков

Задача. Изобразить на графике приблизительную форму электронных облаков в атомах.

Анализ. По современным представлениям, электронные уровни в атоме определяются четырьмя квантовыми числами. Форма электронного облака определяется двумя из этих чисел:

- число l определяет тип орбитали (значения 0–3 соответствуют *s*-, *p*-, *d*- и *f*-орбиталям);
- число m определяет магнитный момент электрона и может изменяться в диапазоне от $-l$ до l.

При m=0 форма электронного облака определяется на основе многочленов Лежандра первого рода:

$$P(x) = \frac{1}{2^l \cdot l!} \cdot \frac{d^l}{dx^l}(x^2-1)^l, \quad \text{где } l \text{ — степень многочлена.}$$

В этом случае $Y(\phi) = \sqrt{\frac{2l+1}{4\pi}} \cdot \left|P(\cos\phi)\right|$.

Параметрическое задание соответствующей поверхности имеет следующий вид:

$$x(\theta,\phi)=Y(\phi)\cdot\sin\phi\cdot\cos\theta$$
$$y(\theta,\phi)=Y(\phi)\cdot\sin\phi\cdot\sin\theta$$
$$z(\theta,\phi)=Y(\phi)\cdot\cos\phi$$

Углы θ, ϕ изменяются в диапазоне от 0 до 2π.

1. Запустите программу *Mathcad*.
2. Определите переменную l, которая укажет тип орбитали.
   $l := 3$
3. Построение поверхности будем производить по точкам. Задайте два диапазона, которые будут определять изменение параметров θ, ϕ, задающих поверхность.

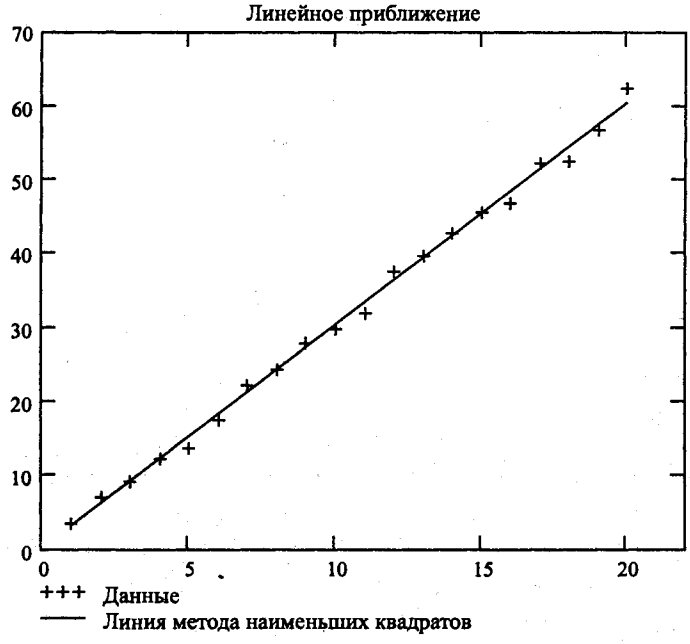


Рис. 18.6. График экспериментальных точек и аппроксимирующей прямой

Удобно определить границы диапазона в целых числах (через точку с запятой, на экране изображаются две точки), а затем произвести перемасштабирование на отрезок [0; 2π].

$i := 0..100 \quad j := 0..100$

$$\theta_i = i \cdot \frac{2 \cdot \pi}{100}, \quad \phi_j = j \cdot \frac{2 \cdot \pi}{100}$$

4. Определите двумерные матрицы, определяющие значения координат x, y и z в зависимости от значения параметров. Используйте названия переменных $X0$, $Y0$ и $Z0$.

$$P(x) = \frac{1}{2^l \cdot l!} \cdot \frac{d^l}{dx^l} (x^2 - 1)^l$$

$$Y(\phi) = \left| \sqrt{\frac{2l+1}{4\pi}} \cdot P(\cos\phi) \right|$$

$X0_{i,j} := Y(\phi_j) \cdot \sin(\phi_j) \cdot \cos(\theta_i)$

$Y0_{i,j} := Y(\phi_j) \cdot \sin(\phi_j) \cdot \sin(\theta_i)$

$Z0_{i,j} := Y(\phi_j) \cdot \cos(\phi_j)$

5. Дайте команду Insert ▸ Graph ▸ Surface Plot (Вставка ▸ График ▸ Поверхность) или воспользуйтесь кнопкой Surface Plot (Поверхность) на панели инструментов Graph (График).

6. В появившейся области графика вместо заполнителя укажите имена отображаемых матриц через запятую, заключив все их в скобки: (X0,Y0,Z0).
7. Чтобы изменить формат построенного графика, дважды щелкните на его области. Откроется диалоговое окно 3-D Plot Format (Формат трехмерного графика).
8. На вкладке General (Общие) установите флажок Equal Scales (Равный масштаб), чтобы обеспечить одинаковый масштаб по осям координат.
9. На вкладке Appearance (Оформление) установите переключатель Fill Surface (Заливка поверхности), чтобы обеспечить заливку построенной поверхности.
10. На вкладке Lighting (Подсветка) включите режим освещения поверхности. Установите флажок Enable Lighting (Включить подсветку), отключите все источники света, кроме первого.
11. На панели Light Location (Размещение источника) задайте координаты источника света. Используйте кнопку Применить, чтобы сразу видеть последствия сделанных настроек. По окончании настройки закройте диалоговое окно щелчком на кнопке OK.
12. Путем протягивания мыши в области графика измените направление осей координат, чтобы изображение было видно наиболее отчетливо.
13. Изменяя значение l, можно увидеть форму электронных облаков для разных орбиталей, в том числе и не встречающихся в природе.

Мы научились строить трехмерные графики с изображением объемных поверхностей, заданных параметрически. Это фактически означает умение изображать любые фигуры, которые могут потребоваться в ходе практической работы.

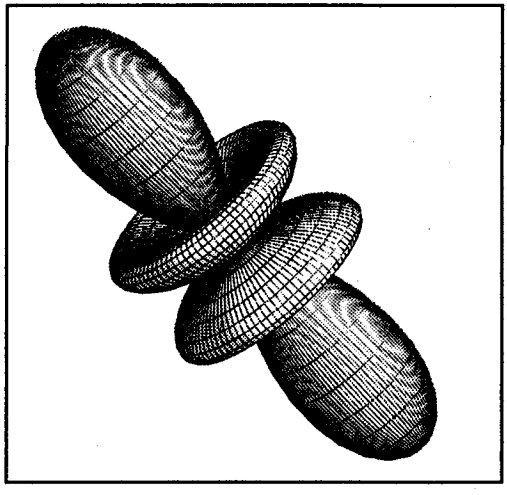


Рис. 18.7. Трехмерное изображение электронной f-орбитали

Упражнение 18.8. Решение дифференциальных уравнений

15 мин

Задача. Найти функцию $y(x)$, удовлетворяющую дифференциальному уравнению $\frac{dy}{dx} + y = x \cdot cosx$ и имеющую значение 0 при $x = 0$.

Анализ. Это простое дифференциальное уравнение допускает точное аналитическое решение. В данном упражнении предполагается использование стандартной функции программы *Mathcad*, осуществляющей численное решение данного уравнения. Результат вычислений можно после этого сравнить с точным решением.

1. Запустите программу *Mathcad*.

2. Задайте начальное значение функции как элемент вектора y, размерность которого соответствует числу решаемых уравнений (в данном случае единице): $y_0 := 0$.
3. Создайте функцию $T(x,y)$, которая вычисляет значение производной при заданных значениях независимой переменной и неизвестной функции:

   $T(x,y) := -y_0 + x \cdot \cos(x)$
4. Определите начальное (точка 0) и конечное значение отрезка интегрирования.

   $a := 0,\ b := 12 \cdot \pi$
5. Укажите число шагов интегрирования.

   $K := 20$
6. Вычислите численное решение уравнения при помощи функции *rkfixed*.

   $Z := \text{rkfixed}(y,a,b,K,T)$

   Результат вычислений — матрица Z с двумя столбцами, первый из которых содержит значения независимой переменной, а второй — соответствующие значения функции.
7. Постройте график полученного решения.
8. Определите аналитическое решение данного уравнения при тех же начальных условиях.
9. Нанесите аналитическую кривую на тот же график и сравните поведение численного и точного решения.

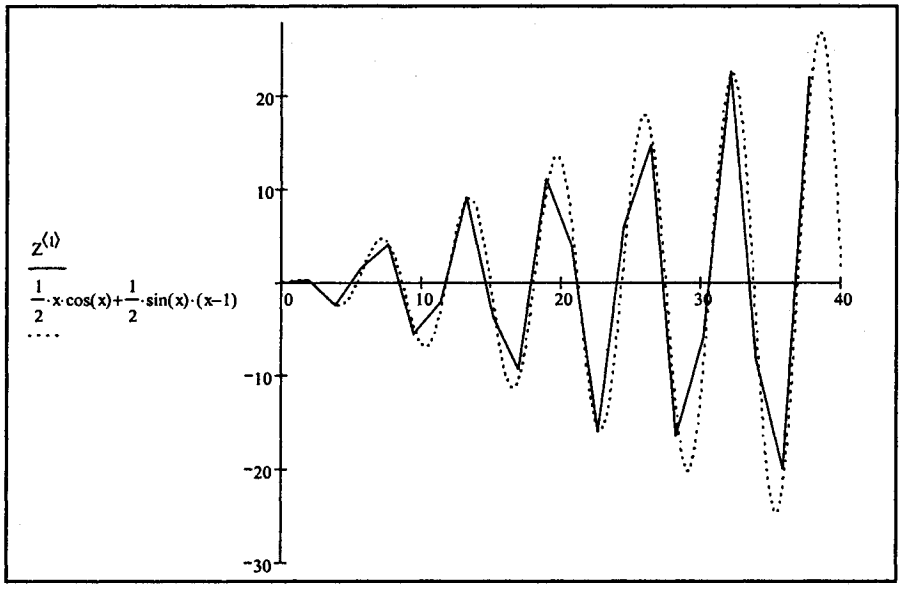


Рис. 18.8. Графики численного и точного решения дифференциального уравнения

10. Измените число шагов, на которые делится отрезок интегрирования, и исследуйте, как изменяется результат расчета при уменьшении и увеличении этого параметра.

Мы научились численно решать дифференциальные уравнения первого порядка с помощью программы Mathcad. Использованный метод без изменений переносится на системы, содержащие два или большее число дифференциальных уравнений. Увеличение величины шага интегрирования ускоряет получение результата, но снижает его точность. При слишком большой величине шага результат расчетов может вообще не соответствовать реальному решению.

Компьютерные программы создают *программисты* — люди, обученные процессу их составления (*программированию*). Мы знаем, что программа — это логически упорядоченная последовательность команд, необходимых для управления компьютером (выполнения им конкретных операций), поэтому программирование сводится к созданию последовательности команд, необходимой для решения определенной задачи.

20.1. Языки программирования

Машинный код процессора

Процессор компьютера — это большая интегральная микросхема. Все команды и данные он получает в виде электрических сигналов. Фактически процессор можно рассматривать как огромную совокупность достаточно простых электронных элементов — транзисторов. Транзистор имеет три вывода. На два крайних подается напряжение, необходимое для создания в транзисторе электрического тока, а на средний вывод — напряжение, с помощью которого можно управлять внутренним сопротивлением транзистора, а значит, управлять и током, и напряжением на его выводах.

В электронике транзисторы имеют три применения: для создания усилителей, в электронных схемах, обладающих автоколебательными свойствами, и в электронных переключателях. Последний способ и применяется в цифровой вычислительной технике. В процессоре компьютера транзисторы сгруппированы в микроэлементы, называемые *триггерами* и *вентилями*. Триггеры имеют два устойчивых состояния (*открыт — закрыт*) и переключаются из одного состояния в другое электрическими сигналами. Этим устойчивым состояниям соответствуют математические понятия 0 или 1. Вентили немного сложнее — они могут иметь несколько входов (напряжение на выходе зависит от комбинаций напряжений на входах) и служат для простейших арифметических и логических операций.

Команды, поступающие в процессор по его шинам, на самом деле являются электрическими сигналами, но и их тоже можно представить как совокупности нулей и единиц, то есть числами. Разным командам соответствуют разные числа. Поэтому реально программа, с которой работает процессор, представляет собой последовательность чисел, называемую *машинным кодом*.

Алгоритм и программа

Управлять компьютером нужно по определенному *алгоритму*. Алгоритм — это точно определенное описание способа решения задачи в виде конечной (по времени) последовательности действий. Такое описание еще называется *формальным*. Для представления алгоритма в виде, понятном компьютеру, служат *языки программирования*. Сначала всегда разрабатывается алгоритм действий, а потом он записывается на одном из таких языков. В итоге получается текст программы — полное, законченное и детальное описание алгоритма на языке программирования. Затем этот текст программы специальными служебными приложениями, которые называются *трансляторами*, либо переводится в машинный код, либо исполняется.

Что такое язык программирования

Самому написать программу в машинном коде весьма сложно, причем эта сложность резко возрастает с увеличением размера программы и трудоемкости решения нужной задачи. Условно можно считать, что машинный код приемлем, если размер программы не превышает нескольких десятков байтов и нет потребности в операциях ручного ввода/вывода данных.

Поэтому сегодня практически все программы создаются с помощью языков программирования. Теоретически программу можно написать и средствами обычного человеческого (естественного) языка — это называется программированием на *метаязыке* (подобный подход обычно используется на этапе составления алгоритма), но автоматически перевести такую программу в машинный код пока невозможно из-за высокой неоднозначности естественного языка.

Языки программирования — искусственные языки. От естественных они отличаются ограниченным числом «слов», значение которых понятно транслятору, и очень строгими правилами записи команд (*операторов*). Совокупность подобных требований образует *синтаксис* языка программирования, а *смысл* каждой команды и других конструкций языка — его *семантику*. Нарушение формы записи программы приводит к тому, что транслятор не может понять назначение оператора и выдает сообщение о синтаксической ошибке, а правильно написанное, но не отвечающее алгоритму использование команд языка приводит к семантическим ошибкам (называемым еще логическими ошибками или ошибками времени выполнения).

Процесс поиска ошибок в программе называется *тестированием*, процесс устранения ошибок — *отладкой*.

Компиляторы и интерпретаторы

С помощью языка программирования создается не готовая программа, а только ее текст, описывающий ранее разработанный алгоритм. Чтобы получить работающую

программу, надо этот текст либо автоматически перевести в машинный код (для этого служат программы-*компиляторы*) и затем использовать отдельно от исходного текста, либо сразу выполнять команды языка, указанные в тексте программы (этим занимаются программы-*интерпретаторы*).

Интерпретатор берет очередной оператор языка из текста программы, анализирует его структуру и затем сразу исполняет (обычно после анализа оператор транслируется в некоторое промежуточное представление или даже машинный код для более эффективного дальнейшего исполнения). Только после того, как текущий оператор успешно выполнен, интерпретатор перейдет к следующему. При этом, если один и тот же оператор должен выполняться в программе многократно, интерпретатор всякий раз будет выполнять его так, как будто встретил впервые. Вследствие этого, программы, в которых требуется осуществить большой объем повторяющихся вычислений, могут работать медленно. Кроме того, для выполнения такой программы на другом компьютере там также должен быть установлен интерпретатор — ведь без него текст программы является просто набором символов.

По-другому можно сказать, что интерпретатор моделирует некую виртуальную вычислительную машину, для которой базовыми инструкциями служат не элементарные команды процессора, а операторы языка программирования.

Компиляторы полностью обрабатывают весь текст программы (он иногда называется *исходный код*). Они просматривают его в поисках синтаксических ошибок (иногда несколько раз), выполняют определенный смысловой анализ и затем автоматически переводят (*транслируют*) на машинный язык — генерируют машинный код. Нередко при этом выполняется *оптимизация* с помощью набора методов, позволяющих повысить быстродействие программы (например, с помощью инструкций, ориентированных на конкретный процессор, путем исключения ненужных команд, промежуточных вычислений и т. д.). В результате законченная программа получается компактной и эффективной, работает в сотни раз быстрее программы, выполняемой с помощью интерпретатора, и может быть перенесена на другие компьютеры с процессором, поддерживающим соответствующий машинный код.

Основной недостаток компиляторов — трудоемкость трансляции языков программирования, ориентированных на обработку данных сложной структуры, часто заранее неизвестной или динамически меняющейся во время работы программы. Тогда в машинный код приходится вставлять множество дополнительных проверок, анализировать наличие ресурсов операционной системы, динамически их захватывать и освобождать, формировать и обрабатывать в памяти компьютера сложные объекты, что на уровне жестко заданных машинных инструкций осуществить довольно трудно, а для ряда задач практически невозможно.

С помощью интерпретатора, наоборот, допустимо в любой момент остановить работу программы, исследовать содержимое памяти, организовать диалог с пользователем, выполнить сколь угодно сложные преобразования данных и при этом постоянно контролировать состояние окружающей программно-аппаратной среды, благодаря чему достигается высокая надежность работы. Интерпретатор при выполнении каждого оператора проверяет множество характеристик операцион-

ной системы и при необходимости максимально подробно информирует разработчика о возникающих проблемах. Кроме того, интерпретатор очень удобен для использования в качестве инструмента изучения программирования, так как позволяет понять принципы работы любого отдельного оператора языка.

В реальных системах программирования перемешаны технологии и компиляции, и интерпретации. В процессе отладки программа может выполняться по шагам, а результирующий код не обязательно будет машинным — он даже может быть исходным кодом, написанным на другом языке программирования (это существенно упрощает процесс трансляции, но требует компилятора для конечного языка), или промежуточным машинно-независимым кодом абстрактного процессора, который в различных компьютерных архитектурах станет выполняться с помощью интерпретатора или компилироваться в соответствующий машинный код.

Уровни языков программирования

Разные типы процессоров имеют разные наборы команд. Если язык программирования ориентирован на конкретный тип процессора и учитывает его особенности, то он называется *языком программирования низкого уровня*. В данном случае «низкий уровень» не значит «плохой». Имеется в виду, что операторы языка близки к машинному коду и ориентированы на конкретные команды процессора.

Языком самого низкого уровня является *язык ассемблера*, который просто представляет каждую команду машинного кода, но не в виде чисел, а с помощью символьных условных обозначений, называемых *мнемониками*. Однозначное преобразование одной машинной инструкции в одну команду ассемблера называется *транслитерацией*. Так как наборы инструкций для каждого модели процессора отличаются, конкретной компьютерной архитектуре соответствует свой язык ассемблера, и написанная на нем программа может быть использована только в этой среде.

С помощью языков низкого уровня создаются очень эффективные и компактные программы, так как разработчик получает доступ ко всем возможностям процессора. С другой стороны, при этом требуется очень хорошо понимать устройство компьютера, затрудняется отладка больших приложений, а результирующая программа не может быть перенесена на компьютер с другим типом процессора. Подобные языки обычно применяют для написания небольших системных приложений, драйверов устройств, модулей стыковки с нестандартным оборудованием, когда важнейшими требованиями становятся компактность, быстродействие и возможность прямого доступа к аппаратным ресурсам. В некоторых областях, например в машинной графике, на языке ассемблера пишутся библиотеки, эффективно реализующие требующие интенсивных вычислений алгоритмы обработки изображений.

Языки программирования высокого уровня значительно ближе и понятнее человеку, нежели компьютеру. Особенности конкретных компьютерных архитектур в них не учитываются, поэтому создаваемые программы на уровне исходных текстов легко переносимы на другие платформы, для которых создан транслятор этого языка. Разрабатывать программы на языках высокого уровня с помощью понятных и мощных команд значительно проще, а ошибок при создании программ допускается гораздо меньше.

Поколения языков программирования

Языки программирования принято делить на пять *поколений*. В первое поколение входят языки, созданные в начале 50-х годов, когда первые компьютеры только появились на свет. Это был первый язык ассемблера, созданный по принципу «одна инструкция — одна строка».

Расцвет второго поколения языков программирования пришелся на конец 50-х — начало 60-х годов. Тогда был разработан символический ассемблер, в котором появилось понятие переменной. Он стал первым полноценным языком программирования. Благодаря его возникновению заметно возросли скорость разработки и надежность программ.

Появление третьего поколения языков программирования принято относить к 60-м годам. В это время родились универсальные языки высокого уровня, с их помощью удается решать задачи из любых областей. Такие качества новых языков, как относительная простота, независимость от конкретного компьютера и возможность использования мощных синтаксических конструкций, позволили резко повысить производительность труда программистов. Понятная большинству пользователей структура этих языков привлекла к написанию небольших программ (как правило, инженерного или экономического характера) значительное число специалистов из некомпьютерных областей. Подавляющее большинство языков этого поколения успешно применяется и сегодня.

С начала 70-х годов по настоящее время продолжается период языков четвертого поколения. Эти языки предназначены для реализации крупных проектов, повышения их надежности и скорости создания. Они обычно ориентированы на специализированные области применения, где хороших результатов можно добиться, используя не универсальные, а проблемно-ориентированные языки, оперирующие конкретными понятиями узкой предметной области. Как правило, в эти языки встраиваются мощные операторы, позволяющие одной строкой описать такую функциональность, для реализации которой на языках младших поколений потребовались бы тысячи строк исходного кода.

Рождение языков пятого поколения произошло в середине 90-х годов. К ним относятся также системы автоматического создания прикладных программ с помощью визуальных средств разработки, без знания программирования. Главная идея, которая закладывается в эти языки, — возможность автоматического формирования результирующего текста на универсальных языках программирования (который потом требуется откомпилировать). Инструкции же вводятся в компьютер в максимально наглядном виде с помощью методов, наиболее удобных для человека, не знакомого с программированием.

Обзор языков программирования высокого уровня

FORTRAN (Фортран). Это первый компилируемый язык, созданный Джимом Бэкусом в 50-е годы. Программисты, разрабатывавшие программы исключительно на ассемблере, выражали серьезное сомнение в возможности появления высокопроизводительного языка высокого уровня, поэтому основным критерием при раз-

работке компиляторов Фортрана являлась эффективность исполняемого кода. Хотя в Фортране впервые был реализован ряд важнейших понятий программирования, удобство создания программ было принесено в жертву возможности получения эффективного машинного кода. Однако для этого языка было создано огромное количество библиотек, начиная от статистических комплексов и кончая пакетами управления спутниками, поэтому Фортран продолжает активно использоваться во многих организациях, а сейчас ведутся работы над очередным стандартом Фортрана *F2k*, который появится в 2000 году. Имеется стандартная версия Фортрана *HPF* (*High Performance Fortran*) для параллельных суперкомпьютеров со множеством процессоров.

COBOL (Кобол). Это компилируемый язык для применения в экономической области и решения бизнес-задач, разработанный в начале 60-х годов. Он отличается большой «многословностью» — его операторы иногда выглядят как обычные английские фразы. В Коболе были реализованы очень мощные средства работы с большими объемами данных, хранящимися на различных внешних носителях. На этом языке создано очень много приложений, которые активно эксплуатируются и сегодня. Достаточно сказать, что наибольшую зарплату в США получают программисты на Коболе.

Algol (Алгол). Компилируемый язык, созданный в 1960 году. Он был призван заменить Фортран, но из-за более сложной структуры не получил широкого распространения. В 1968 году была создана версия Алгол 68, по своим возможностям и сегодня опережающая многие языки программирования, однако из-за отсутствия достаточно эффективных компьютеров для нее не удалось своевременно создать хорошие компиляторы.

Pascal (Паскаль). Язык Паскаль, созданный в конце 70-х годов основоположником множества идей современного программирования Никлаусом Виртом, во многом напоминает Алгол, но в нем ужесточен ряд требований к структуре программы и имеются возможности, позволяющие успешно применять его при создании крупных проектов.

Basic (Бейсик). Для этого языка имеются и компиляторы, и интерпретаторы, а по популярности он занимает первое место в мире. Он создавался в 60-х годах в качестве учебного языка и очень прост в изучении.

C (Си). Данный язык был создан в лаборатории *Bell* и первоначально не рассматривался как массовый. Он планировался для замены ассемблера, чтобы иметь возможность создавать столь же эффективные и компактные программы и в то же время не зависеть от конкретного типа процессора.

Си во многом похож на Паскаль и имеет дополнительные средства для прямой работы с памятью (*указатели*). На этом языке в 70-е годы написано множество прикладных и системных программ и ряд известных операционных систем (*Unix*).

C++ (Си++). Си++ — это объектно-ориентированное расширение языка Си, созданное Бьярном Страуструпом в 1980 году. Множество новых мощных возможностей, позволивших резко повысить производительность программистов, наложи-

лось на унаследованную от языка Си определенную низкоуровневость, в результате чего создание сложных и надежных программ потребовало от разработчиков высокого уровня профессиональной подготовки.

Java (Джава, Ява). Этот язык был создан компанией *Sun* в начале 90-х годов на основе Си++. Он призван упростить разработку приложений на основе Си++ путем исключения из него всех низкоуровневых возможностей. Но главная особенность этого языка — компиляция не в машинный код, а в платформно-независимый байт-код (каждая команда занимает один байт). Этот байт-код может выполняться с помощью интерпретатора — виртуальной *Java*-машины *JVM* (*Java Virtual Machine*), версии которой созданы сегодня для любых платформ. Благодаря наличию множества *Java*-машин программы на *Java* можно переносить не только на уровне исходных текстов, но и на уровне двоичного байт-кода, поэтому по популярности язык Ява сегодня занимает второе место в мире после Бейсика.

Особое внимание в развитии этого языка уделяется двум направлениям: поддержке всевозможных мобильных устройств и микрокомпьютеров, встраиваемых в бытовую технику (технология *Jini*) и созданию платформно-независимых программных модулей, способных работать на серверах в глобальных и локальных сетях с различными операционными системами (технология *Java Beans*). Пока основной недостаток этого языка — невысокое быстродействие, так как язык Ява интерпретируемый.

C# (Си Шарп). В конце 90-х годов в компании *Microsoft* под руководством Андерса Хейльсберга был разработан язык C#. В нем воплотились лучшие идеи Си и Си++, а также достоинства *Java*. Правда, C#, как и другие технологии *Microsoft*, ориентирован на платформу *Windows*. Однако формально он не отличается от прочих универсальных языков, а корпорация даже планирует его стандартизацию. Язык C# предназначен для быстрой разработки *.NET*-приложений, и его реализация в системе *Microsoft Visual Studio .NET* содержит множество особенностей, привязывающих C# к внутренней архитектуре *Windows* и платформы *.NET*.

Языки программирования баз данных

Эта группа языков отличается от алгоритмических языков прежде всего решаемыми задачами. База данных — это файл (или группа файлов), представляющий собой упорядоченный набор *записей*, имеющих единообразную структуру и организованных по единому шаблону (как правило, в табличном виде). База данных может состоять из нескольких таблиц. Удобно хранить в базах данных различные сведения из справочников, картотек, журналов бухгалтерского учета и т. д.

При работе с базами данных чаще всего требуется выполнять следующие операции:

- создание/модификация свойств/удаление таблиц в базе данных;
- поиск, отбор, сортировка информации по запросам пользователей;
- добавление новых записей;
- модификация существующих записей;
- удаление существующих записей.

Первые базы данных появились очень давно, как только появилась потребность в обработке больших массивов информации и выборки групп записей по определенным признакам. Для этого был создан *структурированный язык запросов SQL* (*Structured Query Language*). Он основан на мощной математической теории и позволяет выполнять эффективную обработку баз данных, манипулируя не отдельными записями, а *группами* записей.

Для управления большими базами данных и их эффективной обработки разработаны СУБД (Системы Управления Базами Данных). Практически в каждой СУБД помимо поддержки языка *SQL* имеется также свой уникальный язык, ориентированный на особенности этой СУБД и не переносимый на другие системы. Сегодня в мире насчитывается три ведущих производителя СУБД: *Microsoft* (*SQL Server*), *IBM* (*DB2*) и *Oracle*. Их продукты нацелены на поддержку одновременной работы тысяч пользователей в сети, а базы данных могут храниться в распределенном виде на нескольких серверах. В каждой из этих СУБД реализован собственный диалект *SQL*, ориентированный на особенности конкретного сервера, поэтому *SQL*-программы, подготовленные для разных СУБД, друг с другом, как правило, несовместимы.

С появлением персональных компьютеров были созданы так называемые настольные СУБД. Родоначальником современных языков программирования баз данных для ПК принято считать СУБД *dBase II*, язык которой был интерпретируемым. Затем для него были созданы компиляторы, появились СУБД *FoxPro* и *Clipper*, поддерживающие диалекты этого языка. Сегодня самой распространенной настольной СУБД стала система *Microsoft Access*.

Языки программирования для Интернета

С активным развитием глобальной сети было создано немало реализаций популярных языков программирования, адаптированных специально для Интернета. Все они отличаются характерными особенностями: языки являются интерпретируемыми, интерпретаторы для них распространяются бесплатно, а сами программы — в исходных текстах. Такие языки называют *скрипт-языками*.

HTML. Общеизвестный язык для оформления документов. Он очень прост и содержит элементарные команды форматирования текста, добавления рисунков, задания шрифтов и цветов, организации ссылок и таблиц. Все *Web*-страницы написаны на языке *HTML* или используют его расширения.

Perl. В 80-х годах Ларри Уолл разработал язык *Perl*. Он задумывался как средство эффективной обработки больших текстовых файлов, генерации текстовых отчетов и управления задачами. По мощности *Perl* значительно превосходит языки типа Си. В него введено много часто используемых функций работы со строками, массивами, всевозможные средства преобразования данных, управления процессами, работы с системной информацией и др.

PHP. Расмус Лердорф, активно использовавший *Perl*-скрипты, в 1995 году решил улучшить этот язык, упростив его и дополнив встроенными средствами доступа к базам данных. В результате появилась разработка *Personal Contents Page/Forms Interpreter* (*PHP/FI*). Уже через пару лет программы на ее основе использовались на 50 тыс. сайтов. В 1997 году ее значительно усовершенствовали Энди Гутманс и

Зив Сураски, и под названием *PHP* 3.0 этот язык быстро завоевал популярность у создателей динамических сайтов во всем мире.

Tcl/Tk. В конце 80-х годов Джон Аустираут придумал популярный скрипт-язык *Tcl* и библиотеку *Tk*. В *Tcl* он попытался воплотить видение идеального скрипт-языка. Язык *Tcl* ориентирован на автоматизацию рутинных процессов и состоит из мощных команд, предназначенных для работы с абстрактными нетипизированными объектами. Он независим от типа системы и при этом позволяет создавать программы с графическим интерфейсом.

VRML. В 1994 году был создан язык *VRML* для организации виртуальных трехмерных интерфейсов в Интернете. Он позволяет описывать в текстовом виде различные трехмерные сцены, освещение и тени, текстуры (покрытия объектов), создавать свои миры, путешествовать по ним, «облетать» со всех сторон, вращать в любых направлениях, масштабировать, регулировать освещенность и т. д.

XML. В августе 1996 года *WWW*-консорциум, ответственный за стандарты на Интернет-технологии, приступил к подготовке универсального языка разметки структуры документов, базировавшегося на достаточно давно созданной в *IBM* технологии *SGML*. Новый язык получил название *XML*. Сегодня он служит основой множества системных, сетевых и прикладных приложений, позволяя представлять в прозрачном для пользователей и программ текстовом виде различные аспекты внутренней структуры иерархически организованных документов. В недалеком будущем он может стать заменой *HTML*.

Языки моделирования

При создании программ и формировании структур баз данных нередко применяются формальные способы их представления — *формальные нотации*, с помощью которых можно визуально представить (изобразить с помощью мыши) таблицы баз данных, поля, объекты программы и взаимосвязи между ними в системе, имеющей специализированный редактор и генератор исходных текстов программ на основе созданной модели. Такие системы называются *CASE-системами*. В них активно применяются нотации *IDEF*, а в последнее время все большую популярность завоевывает язык графического моделирования *UML*.

Прочие языки программирования

PL/I (ПЛ/1). В середине 60-х годов компания *IBM* решила взять все лучшее из языков Фортран, Кобол и Алгол. В результате в 1964 году на свет появился новый компилируемый язык программирования, который получил название *Programming Language One*. В этом языке было реализовано множество уникальных решений, полезность которых удается оценить только спустя 33 года, в эпоху крупных программных систем. По своим возможностям ПЛ/1 значительно мощнее многих других языков (Си, Паскаля). Например, в ПЛ/1 присутствует уникальная возможность указания точности вычислений — ее нет даже у Си++ и Явы. Этот язык и сегодня продолжает поддерживаться компанией *IBM*.

Smalltalk (Смолток). Работа над этим языком началась в 1970 году в исследовательской лаборатории корпорации *XEROX*, а закончились спустя 10 лет, воплотив-

шись в окончательном варианте интерпретатора *SMALLTALK-80*. Данный язык оригинален тем, что его синтаксис очень компактен и базируется исключительно на понятии объекта. В этом языке отсутствуют операторы или данные. Все, что входит в Смолток, является объектами, а сами объекты общаются друг с другом исключительно с помощью сообщений (например, появление выражения I + 1 вызывает посылку объекту I сообщения «+», то есть «прибавить», с параметром 1, который считается не числом-константой, а тоже объектом). Больше никаких управляющих структур, за исключением «оператора» ветвления (на самом деле функции, принадлежащей стандартному объекту), в языке нет, хотя их можно очень просто смоделировать. Сегодня версия *VisualAge for Smalltalk* активно развивается компанией *IBM*.

LISP (Лисп). Интерпретируемый язык программирования, созданный в 1960 году Джоном Маккарти. Ориентирован на структуру данных в форме списка и позволяет организовывать эффективную обработку больших объемов текстовой информации.

Prolog (Пролог). Создан в начале 70-х годов Аланом Колмероэ. Программа на этом языке, в основу которого положена математическая модель теории исчисления предикатов, строится из последовательности фактов и правил, а затем формулируется утверждение, которое Пролог будет пытаться доказать с помощью введенных правил. Человек только описывает структуру задачи, а внутренний «мотор» Пролога сам ищет решение с помощью методов поиска и сопоставления.

Ada (Ада). Назван по имени леди Огасты Ады Байрон, дочери английского поэта Байрона и его отдаленной родственницы Анабеллы Милбэнк. В 1980 году сотни экспертов Министерства обороны США отобрали из 17 вариантов именно этот язык, разработанный небольшой группой под руководством Жана Ишбиа. Он удовлетворил на то время все требования Пентагона, а к сегодняшнему дню в его развитие вложены десятки миллиардов долларов. Структура самого языка похожа на Паскаль. В нем имеются средства строгого разграничения доступа к различным уровням спецификаций, доведена до предела мощность управляющих конструкций.

Forth (Форт). Результат попытки Чарльза Мура в 70-х годах создать язык, обладающий мощными средствами программирования, который можно эффективно реализованным на компьютерах с небольшими объемами памяти, а компилятор мог бы выдавать очень быстрый и компактный код, то есть служил заменой ассемблеру. Однако сложности восприятия программного текста, записанного в непривычной форме, сильно затрудняли поиск ошибок, и с появлением Си язык Форт оказался забытым.

Вопросы для самоконтроля

1. Что такое язык программирования?
2. В чем различие компиляторов и интерпретаторов?
3. Объясните термины «язык низкого уровня» и «язык высокого уровня».
4. Расскажите о поколениях языков программирования.
5. Какие языки программирования активно используются сегодня?

20.2. Системы программирования

Средства создания программ

В самом общем случае для создания программы на выбранном языке программирования нужно иметь следующие компоненты.

1. *Текстовый редактор.* Так как текст программы записывается с помощью ключевых слов, обычно происходящих от слов английского языка, и набора стандартных символов для записи всевозможных операций, то формировать этот текст можно в любом редакторе, получая в итоге текстовый файл с *исходным текстом* программы. Лучше использовать специализированные редакторы, которые ориентированы на конкретный язык программирования и позволяют в процессе ввода текста выделять ключевые слова и идентификаторы разными цветами и шрифтами. Подобные редакторы созданы для всех популярных языков и дополнительно могут автоматически проверять правильность синтаксиса программы непосредственно во время ее ввода.
2. Исходный текст с помощью *программы-компилятора* переводится в машинный код. Если обнаружены синтаксические ошибки, то результирующий код создан не будет.

   На этом этапе уже возможно получение готовой программы, но чаще всего в ней не хватает некоторых компонентов, поэтому компилятор обычно выдает промежуточный *объектный код* (двоичный файл, стандартное расширение .OBJ).
3. Исходный текст большой программы состоит, как правило, из нескольких *модулей* (файлов с исходными текстами), потому что хранить все тексты в одном файле неудобно — в них сложно ориентироваться. Каждый модуль компилируется в отдельный файл с объектным кодом, которые затем надо объединить в одно целое.

   Кроме того, к ним надо добавить машинный код подпрограмм, реализующих различные стандартные функции (например, вычисляющих математические функции *sin* или *ln*). Такие функции содержатся в *библиотеках* (файлах со стандартным расширением .LIB), которые поставляются вместе с компилятором. Сгенерированный код модулей и подключенные к нему стандартные функции надо не просто объединить в одно целое, а выполнить такое объединение с учетом требований операционной системы, то есть получить на выходе программу, отвечающую определенному формату.

   Объектный код обрабатывается специальной программой — *редактором связей* или *сборщиком*, который выполняет связывание объектных модулей и машинного кода стандартных функций, находя их в библиотеках, и формирует на выходе работоспособное приложение — *исполнимый код* для конкретной платформы.

   Если по каким-то причинам один из объектных модулей или нужная библиотека не обнаружены (например, неправильно указан каталог с библиотекой), то сборщик сообщает об ошибке и готовой программы не получается.

4. Исполнимый код — это законченная программа, которую можно запустить на любом компьютере, где установлена операционная система, для которой эта программа создавалась. Как правило, итоговый файл имеет расширение .EXE.

Интегрированные системы программирования

Итак, для создания программы нужны:

- текстовый редактор;
- компилятор;
- редактор связей;
- библиотеки функций.

Как правило, в стандартную поставку входят как минимум три последних компонента, но хорошая *интегрированная система* включает в себя и специализированный текстовый редактор, причем почти все этапы создания программы в ней автоматизированы: после того как исходный текст введен, его компиляция и сборка выполняются одним нажатием клавиши. Это очень удобно, так как не требует ручной настройки множества параметров запуска компилятора и редактора связей, указывания им нужных файлов вручную и т. д. Процесс компиляции обычно демонстрируется на экране: показывается, сколько строк исходного текста откомпилировано, или выдаются сообщения о найденных ошибках.

В современных интегрированных системах имеется еще один компонент — *отладчик*, который позволяет анализировать работу программы во время ее выполнения. С его помощью можно последовательно выполнять отдельные операторы исходного текста *по шагам*, наблюдая при этом, как меняются значения различных переменных. Без отладчика разработать крупное приложение очень сложно.

Среды быстрого проектирования

В последние несколько лет в программировании (особенно в программировании для операционной системы *Windows*) наметился так называемый *визуальный подход*. До этого серьезным препятствием для разработки графических приложений была сложность создания различных элементов управления и контроля их работы. Достаточно взглянуть на окно любой *Windows*-программы. В нем имеется множество стандартных элементов управления (кнопки, пункты меню, списки, переключатели и т. д.). Очень трудоемко вручную описывать процесс создания этих элементов в соответствии с требованиями *Windows*, на глазок определять координаты, отслеживать их состояние с помощью специальных команд. Например, для простой программы, складывающей два числа, потребуется один оператор (одна строка исходного текста) для выполнения нужного вычисления и сотни строк кода для подготовки приложения к работе в *Windows*, создания кнопки и пары полей ввода.

Этот процесс автоматизирован в *средах быстрого проектирования* (*Rapid Application Development*, *RAD-среды*). Все необходимые элементы оформления и управления создаются и обслуживаются не путем ручного программирования, а с помощью готовых визуальных *компонентов*, которые с помощью мыши «перетаскиваются» в проектируемое окно. Их свойства и поведение затем настраиваются с помощью

простых редакторов, визуально показывающих характеристики соответствующих элементов. При этом вспомогательный исходный текст программы, ответственный за создание и работу этих элементов, генерируется *RAD*-средой автоматически, что позволяет сосредоточиться только на логике решаемой задачи. В результате программирование во многом заменяется на проектирование — подобный подход называется еще *визуальным программированием.*

Компоненты достаточно легко создавать самостоятельно, поэтому в мире сегодня распространяются тысячи бесплатных и платных компонентов для наиболее известных *RAD*-сред, из них формируются библиотеки компонентов — *объектные репозитории.* Компоненты выступают в роли «строительных кирпичиков», позволяющих собирать готовое приложение с богатыми возможностями, написав всего десяток строк исходного кода, и такой *компонентный подход* к созданию программ считается очень перспективным, потому что без лишних усилий и на законных основаниях допускает *повторное использование* чужого труда.

Архитектура программных систем

В то время как большинство автономных приложений: офисные программы, среды разработки, системы подготовки текстов и изображений — выполняются на одном компьютере, крупные информационные комплексы (например, система автоматизации предприятия) состоят из десятков и сотен отдельных программ, которые взаимодействуют друг с другом по сети, выполняясь на разных компьютерах. В таких случаях говорят, что они работают в различной *программной архитектуре.* Она делится на следующие группы.

Автономные приложения. Работают на одном компьютере.

Приложения в файл-серверной архитектуре. Компьютеры пользователей системы объединены в сеть, при этом на каждом из них (на *клиентском месте*) запущены копии одной и той же программы, которые обращаются за данными к *серверу* — специальному компьютеру, который хранит файлы, одновременно доступные всем пользователям (как правило, это базы данных). Сервер обладает повышенной надежностью, высоким быстродействием, большим объемом памяти, на нем установлена специальная *серверная* версия операционной системы.

При одновременном обращении нескольких программ к одному файлу, например, с целью его обновления, могут возникнуть проблемы, связанные с неоднозначностью определения его содержимого. Поэтому каждое изменение общедоступного файла выделяется в *транзакцию* — элементарную операцию по обработке данных, имеющую фиксированные начало, конец (успешное или неуспешное завершение) и ряд других характеристик.

Особенность этой архитектуры в том, что все вычисления выполняются на клиентских местах, что требует наличия на них достаточно производительных ПК (это так называемые системы с *толстым клиентом* — программой, которая выполняет всю обработку получаемой от сервера информации).

Приложения в клиент-серверной архитектуре. Эта архитектура похожа на предыдущую, только сервер помимо простого обеспечения одновременного доступа к дан-

ным способен еще выполнять программы (обычно выполняются СУБД — тогда сервер называется *сервером баз данных*), которые берут на себя определенный объем вычислений (в файл-серверной архитектуре он реализуется полностью на клиентских местах). Благодаря этому удается повысить общую надежность системы, так как сервер работает значительно более устойчиво, чем ПК, и снять лишнюю нагрузку с клиентских мест, на которых удается использовать дешевые компьютеры. Запускаемые на них приложения реально осуществляют небольшие объемы вычислений, а иногда занимаются только отображением получаемой от сервера информации, поэтому они называются *тонкими клиентами*.

Приложения в многозвенной архитектуре. Недостаток предыдущей архитектуры в том, что резко возрастает нагрузка на сервер, а если он выходит из строя, то работа всей системы останавливается. Поэтому в некоторых случаях в систему добавляется так называемый *сервер приложений*, на котором выполняется вся вычислительная работа. Другой сервер баз данных обрабатывает запросы пользователей, на третьем может быть установлена специальная программа — *монитор транзакций*, которая оптимизирует обработку транзакций и балансирует нагрузку на серверы. В большинстве практических случаев все серверы соединены последовательно — *позвенно*, и выход из строя одного звена если и не останавливает всю работу, то, по крайней мере, резко снижает производительность системы.

Приложения в распределенной архитектуре. Чтобы избежать недостатков рассмотренных архитектур, были придуманы специальные технологии, позволяющие создавать программу в виде набора компонентов, которые можно запускать на любых серверах, связанных в сеть (компоненты как бы распределены по сети). Основное преимущество подобного подхода в том, что при выходе из строя любого компьютера специальные *программы-мониторы*, которые следят за корректностью работы компонентов и позволяют им «переговариваться» между собой, сразу перезапуская временно пропавший компонент на другом компьютере. При этом общая надежность всей системы становится очень высокой, а вычислительная загрузка распределяется между серверами оптимальным образом.

Доступ к возможностям любого компонента, предназначенного для общения с пользователем, осуществляется с произвольного клиентского места. При этом, так как все вычисления происходят на серверах, появляется возможность создавать *сверхтонкие клиенты* — программы, только отображающие получаемую из сети информацию и требующие минимальных компьютерных ресурсов. Благодаря этому доступ к компонентной системе возможен не только с ПК, но и с небольших мобильных устройств.

Частный случай компонентного подхода — доступ к серверным приложениям из браузеров через Интернет.

Сегодня наиболее популярны три компонентные технологии — *CORBA* консорциума *OMG*, *Java Beans* компании *Sun* и *COM+/.NET* корпорации *Microsoft*. Эти технологии будут определять развитие информационной индустрии в ближайшие десятилетия.

Основные системы программирования

Из универсальных языков программирования сегодня наиболее популярны следующие:

- Бейсик (*Basic*) — для освоения требует начальной подготовки (общеобразовательная школа);
- Паскаль (*Pascal*) — требует специальной подготовки (школы с углубленным изучением предмета и общетехнические вузы);
- Си++ (C++), Ява (*Java*), Си Шарп (C#) — требуют профессиональной подготовки (специализированные средние и высшие учебные заведения).

Для каждого из этих языков программирования сегодня имеется немало систем программирования, выпускаемых различными фирмами и ориентированных на различные модели ПК и операционные системы. Наиболее популярны следующие визуальные среды быстрого проектирования программ для *Windows*:

- *Basic*: *Microsoft Visual Basic*;
- *Pascal*: *Borland Delphi*;
- *C++*: *Microsoft Visual C++*;
- *Java*: *Borland JBuilder*;
- *C#*: *Microsoft Visual Studio .NET*, *Borland C#Builder*.

Для разработки серверных и распределенных приложений можно использовать систему программирования *Microsoft Visual C++*, продукты фирмы *Borland*, практически любые средства программирования на *Java*.

В дальнейшем будут рассматриваться возможности, характерные для Бейсика, Паскаля и Си++.

Вопросы для самоконтроля

1. Что нужно для создания программы?
2. Что такое среды быстрого проектирования?
3. Объясните понятие «архитектура программной системы».
4. Опишите основные типы программных архитектур.
5. Какая программная архитектура обеспечивает работу Интернета?

20.3. Алгоритмическое (модульное) программирование

Алгоритм — это формальное описание способа решения задачи путем разбиения ее на конечную по времени последовательность действий (элементарных операций). Под словом «формальное» подразумевается, что описание должно быть абсолютно полным и учитывать все возможные ситуации, которые могут встретиться по ходу решения. Под элементарной операцией понимается действие, которое по заранее определенным критериям (например, очевидности) не имеет смысла детализировать.

Основная идея алгоритмического программирования — разбиение программы на последовательность модулей, каждый из которых выполняет одно или несколько

действий. Единственное требование к модулю — чтобы его выполнение всегда начиналось с первой команды и всегда заканчивалось на самой последней (то есть, чтобы нельзя было попасть на команды модуля извне и передать управление из модуля на другие команды в обход заключительной).

Алгоритм на выбранном языке программирования записывается с помощью команд описания данных, вычисления значений и управления последовательностью выполнения программы.

Переменные и константы

Реальные данные, с которыми работает программа, — это *числа*, *строки* и *логические величины* (аналоги 1 и 0, «да» и «нет», «истина» и «ложь»). Эти типы данных называют *базовыми*.

Каждая единица информации хранится в ячейках памяти компьютера, имеющих свои адреса. На практике заранее неизвестно, в каких конкретно ячейках памяти во время работы программы будут записаны ее данные, поэтому в языках программирования введено понятие переменной, позволяющее отвлечься от конкретных адресов и обращаться к содержимому памяти с помощью *идентификатора* или *имени* — как правило, последовательности, содержащей английские буквы, цифры, символы подчеркивания и начинающейся не с цифры. Например:

```
Hello
_SumOfReal
x1
H8_G7_F6
```

Это имя будет указывать на *значение*, о реальном адресе и способе хранения которого можно забыть. В процессе работы программы содержимое соответствующих ячеек можно менять, обращаясь к переменной по имени. Лучше выбирать такие названия, которые отражают назначение данной переменной.

Кроме имени и значения, переменная обычно имеет *тип*, определяющий, какая информация хранится в данной переменной (число, строка и т. д.). В зависимости от объема памяти, отведенного для хранения значения переменной, оно должно укладываться в *допустимый диапазон*. Например, значение типа «байт» имеет диапазон от 0 до 255.

Переменные с указанием их типа можно вводить в программу с помощью специальных команд *описания* (*объявления*, *декларации*). Это позволяет компилятору организовать эффективное хранение и обработку данных и повышает ясность исходных тестов. Каждый тип описывается своим ключевым словом. Значения переменных разных типов допускается преобразовывать друг в друга в соответствии с соглашениями языка программирования. Такой процесс называется *приведением типов*.

Переменные могут существовать на всем протяжении работы программы — тогда они называются *статическими*, а могут создаваться и уничтожаться на разных этапах ее функционирования — такие переменные называются *динамическими*. Все

остальные данные в программе, значение которых не меняется на протяжении ее работы, называются *константами* или *постоянными*. Константы, как и переменные, обычно имеют тип. Данные можно указывать явно:

```
123
2.87
"это строка"
```

или для удобства обозначать их идентификаторами. Например, число π, равное 3,1416, можно обозначить как pi и везде вместо числа применять идентификатор. Только изменять значение pi нельзя, так как это не переменная, а константа.

Числовые данные

Числа обычно бывают двух видов: *целые* и *дробные*. Если число отрицательное, перед ним ставится знак «–»; если положительное, то знак «+» можно ставить, а можно и опускать. Вычисления над целыми числами выполняются *точно*, вычисления над дробными числами — *приближенно*. При записи дробных чисел в качестве десятичного разделителя используется точка:

```
1.28
3.333321
```

Очень большие или очень маленькие числа записываются специальным образом. Для них дополнительно указывается *мантисса* — число со знаком, являющееся степенью числа 10. Мантисса записывается справа от числа через букву e (или E). Пробелы в такой записи не допускаются.

Например, число 100 (единица, умноженная на 10 во второй степени) запишется так:

```
1e+2
```

число 0,003 (тройка, умноженная на 10 в минус третьей степени) так:

```
3e-3
```

число со 120 нулями — так:

```
1E+120
```

Допускается дробная запись числа с мантиссой:

```
31.4e-1
```

Тип числа	Бейсик	Паскаль	Си++
целое	INTEGER	integer	int
дробное	DOUBLE	real	float

Арифметические операции

Для записи арифметических действий используются арифметические операторы. В некоторых языках программирования они считаются не операторами, а *опера-*

циями, предназначенными для вычисления значения выражения, но не влияющими на другие значения и не сказывающимися на ходе выполнения программы.

К основным арифметическим операциям относятся:

+ (сложение)
- (вычитание)
* (умножение)
/ (деление)

Такая форма записи отвечает общепринятым соглашениям и принята в большинстве языков программирования.

Каждая арифметическая операция имеет свой *приоритет*. Операции с более высоким приоритетом (умножение и деление) будут выполняться раньше, чем операции с более низким приоритетом (сложение и вычитание). Изменить порядок вычисления выражения можно с помощью круглых скобок.

```
b*2 + c/3
b*(2 + c) - 3
```

Скобки допускается вкладывать друг в друга произвольное число раз. При этом использование квадратных или фигурных скобок, как правило, не допускается.

```
((y+2)*3 + 1) / 2
```

Арифметические выражения

С помощью арифметических операций формируются арифметические выражения, которые состоят из операций и *операндов* (переменных и констант).

Выражение

```
i1 + 2
```

состоит из одной операции «+» и двух операндов — переменной i1 и числовой константы 2.

Каждое выражение имеет значение, которое определяется в момент выполнения оператора, содержащего это выражение. Если на момент вычисления выражения i1+2 в переменной i1 хранится число 3, то значение этого выражения будет равно 5 (3+2).

Логические выражения

При создании программ не обойтись без *логических выражений*. Они отличаются тем, что результат их вычислений может принимать только одно из двух допустимых значений — true (истина, да, включено) и false (ложь, нет, выключено). Чаще всего значение false ассоциируется с нулем, а значение true — с числом 1 или просто ненулевым значением.

При записи логических выражений используются *операции сравнения* и *логические операции*. Операции сравнения сличают значения правого и левого операндов. Результатом сравнения является true, если оно удачно, и false в противном случае.

В таблице даны примеры записи операций сравнения для разных языков.

Операция	Варианты написания	
	Бейсик, Паскаль	Си++
Равно	=	==
Не равно	<>	!=
Меньше	<	<
Меньше или равно	<=	<=
Больше	>	>
Больше или равно	>=	>=

```
Pi == 3.14
x > 0
a1 <> b1
```

В одном выражении может потребоваться проверка нескольких подобных условий. Например, надо определить, больше ли значение переменной X, чем 0 и меньше ли, чем 10. Условия могут быть связаны с помощью логических операций, наиболее активно используемые из которых — это И и ИЛИ. В компьютерной графике также часто применяется так называемое исключающее ИЛИ и операция отрицания НЕ. Для нее требуется только один операнд, указывающийся справа от знака операции. Эта операция просто меняет значение своего операнда на противоположное.

1 операнд	2 операнд	И	ИЛИ	исключающее ИЛИ	НЕ (только первый операнд)
true	true	true	true	false	false
true	false	false	true	true	false
false	true	false	true	true	true
false	false	false	false	false	true

В следующей таблице приведен синтаксис записи логических операций.

Логическая операция	Бейсик	Паскаль	Си++
И	AND	and	&&
ИЛИ	OR	or	||
НЕ	NOT	not	!

Приоритеты всех логических операций ниже, чем приоритеты операций сравнения, поэтому сравнения всегда выполняются первыми. А логические операции вычисляются в следующем порядке: сначала НЕ, потом И, потом ИЛИ. При необходимости этот порядок может быть изменен с помощью скобок.

Примеры логических выражений:

```
x1 >= 1 && x1 <= 10
(R > 3.14) and (R < 3.149)
(Value < Oldvalue) OR (Value <> 0)
```

Логический тип

Бейсик	Паскаль	Си++
Базового типа нет. Используется числовой тип INTEGER	boolean	bool

Строчные выражения

Строки в языках программирования всегда заключаются в кавычки. В Си++ и Бейсике для этого используются двойные кавычки, в Паскале — одинарные.

```
"это строка Бейсика или Си++"
'это строка Паскаля'
```

Строка может быть *пустой* — не содержать ни одного символа.

Например:

```
''
""
```

Как правило, строки можно сравнивать друг с другом на эквивалентность (равно и не равно). В некоторых языках программирования допускаются также сравнения типа «больше» или «меньше» — при этом происходит последовательное сравнение значений символов (каждый символ представляется в компьютере конкретным числом).

Кроме того, часто допускается также *операция сцепления строк*, записываемая с помощью символа «+». Например:

```
"123" + "4567" — получится "1234567"
"абв " + "abc " + " эюя" — получится "абв abc  эюя"
```

Тип «строка»

Бейсик	Паскаль	Си++
STRING	string	Базового типа «строка» нет

Указатели

Некоторые языки программирования допускают в явном виде работу с *указателями* — адресами физической памяти. При этом в них имеется специальная *операция получения адреса* конкретной переменной, что позволяет работать с памятью напрямую, примерно так, как это происходит в языках ассемблера. Такая возможность позволяет добиваться высокой эффективности работы программы, но часто приводит к ошибкам, если указатель вдруг получает неверное значение и при его использовании начинает портиться область памяти, предназначенная совсем для других целей.

Сложные данные

Структуры. До сих пор рассматривались базовые типы данных: числа, строки, логические величины — и операции над базовыми данными. Однако для повышения производительности труда программистов и повышения качества их работы необходимо, чтобы язык программирования имел средства, позволяющие описывать данные в виде, максимально приближенном к их реальным аналогам. Например,

чтобы организовать обработку данных по студентам, в программе удобно не просто описать десяток различных переменных, а объединить их в *структуру* (или *запись*) «студент», состоящую из *полей* разного типа «имя», «пол», «год рождения», «группа» и т. д.

Современные языки программирования позволяют применять такие *сложные типы данных*, составляющиеся из базовых и определенных ранее сложных типов. В результате удается организовывать структуры данных произвольной сложности: списки, деревья и т. п. При этом структура объединяет группу разных данных под одним названием.

Получить доступ к отдельным составляющим (полям) этой структуры можно по их именам. В рассматриваемых языках программирования такой доступ осуществляется указанием имени структуры и имени поля через точку. Если подобным способом происходит обращение к полю, которое само является структурой, то выделение нужного поля продолжается приписыванием справа имени вложенного поля через точку.

Синтаксис описания структуры

Бейсик	Паскаль	Си++

TYPE имя-структуры
поле AS тип
...
END TYPE
	record
	поле: тип;
	...
	end;
		struct имя
		{
		тип поле;
		...
		};

Вот примеры описания структур.

Бейсик:

```
TYPE Student
Name AS STRING
Sex AS INTEGER
BirthYear AS INTEGER
END TYPE
```

Паскаль:

```
Record
Name: string;
Sex: boolean;
BirthYear: integer;
end;
```

Си++:

```
struct Student
{
bool Sex;
int BirthYear;
};
```

Доступ к содержимому структуры:

```
Student.BirthYear = 1980;
```

Массивы. Доступ к элементам структуры осуществляется по имени ее составляющих. В одних случаях это значительно повышает наглядность исходных текстов и упрощает процесс программирования, но имеется немало ситуаций, когда надо организовать обработку больших объемов данных одного типа, при этом создавать структуры с сотнями и тысячами полей неразумно. Поэтому в дополнение к структурам в языки программирования введено понятие *массива*, сложного типа данных, доступ к элементам которого происходит по их положению, по номеру или индексу. Например, можно описать массив, состоящий из тысячи элементов численного типа, и затем обратиться к десятому или сотому элементу по его номеру.

При описании массива обычно указывается его *размер* (число элементов) или верхняя и нижняя *границы* — диапазон, в рамках которого можно обращаться к элементам массива.

Синтаксис описания массива

Бейсик	DIM имя (число элементов) AS тип
Паскаль	array[нижняя_граница .. верхняя_граница] of тип;
Си++	тип имя[число-элементов];

В Бейсике нижней границей считается 1, в Си++ — 0, в Паскале она указывается явно.

Вот примеры описания массивов.

Бейсик:

```
DIM IntArray(1000) AS INTEGER
```

Паскаль:

```
array[1..1000] of integer
```

Си++:

```
int IntArray[1000];
```

Доступ к элементу массива осуществляется по его номеру. Этот номер указывается в круглых (Бейсик) или квадратных (Паскаль, Си++) скобках сразу за именем массива (такое действие называется *индексированием*):

```
IntArray( 12 )
IntArray[ i+1 ]
```

Массивы, границы которых явно заданы в команде описания, называются *статическими*. Их размер известен заранее и не меняется на всем протяжении работы программы.

В последних версиях компилируемых языков программирования реализуются так называемые *динамические* массивы, размер которых может меняться во время выполнения программы. В ряде случаев это весьма удобно, так как позволяет экономно расходовать память, захватывая ее по мере необходимости. Недостаток динамических массивов в том, что организовать эффективную работу с ними, исполь-

зуя компиляторы, сложно. Приходится выполнять множество проверок, связанных с расходованием памяти компьютера, что понижает общую эффективность приложения. Динамические массивы в Паскале начали поддерживаться совсем недавно, с активным распространением новых мощных ПК, а в интерпретируемых языках типа Бейсика это было сделано довольно давно.

Во многих языках программирования строки рассматриваются как массивы символов. Их допускается индексировать как обычные массивы.

Правила работы со сложными типами

Отличие базовых типов от сложных в том, что в базовых типах нельзя выделить составные части. При этом поле структуры или элемент массива считаются обычными переменными, и их использование в любых операторах ничем не отличается от использования переменных базовых типов.

В развитых языках программирования допускаются массивы, состоящие из структур, и структуры, состоящие из массивов. При этом возможны достаточно сложные формы записи, например:

```
a[0].Items.Strings[4].value
```

Массив a состоит из структур, в описании которых есть поле Items, являющееся тоже структурой, имеющей поле Strings, которое, в свою очередь, представляет собой массив структур, имеющих поле value.

Описание переменных

Чтобы переменную можно было использовать в программе, ее надо предварительно описать, указав ее тип. Пока переменная не описана, обращаться к ней нельзя (хотя в некоторых языках, например в Бейсике и Фортране, считается, что все переменные, не объявленные явно, имеют числовой тип). После того как переменная описана, к ней можно обращаться, но она обычно исходно имеет *неопределенное значение*, поэтому ее надо предварительно *инициализировать* — присвоить ей начальное значение.

Синтаксис команд описания данных

Бейсик	Паскаль	Си++
DIM имя AS тип	var имя: тип;	тип имя;

Вот примеры описания переменных.

Бейсик:

```
DIM X AS DOUBLE
```

Паскаль:

```
var x: real;
var Str: record
         P1: integer;
         S: string;
         end;
```

Си++:

```
float x;
int a[20];
```

При описании переменных одного типа в Паскале и Си++ их можно указывать через запятую.

Паскаль:

```
var xx, z2: integer;
```

Си++:

```
int xx, yy[10], z2;
```

Новые типы данных

При определении нескольких переменных со сложной структурой удобно описывать каждую переменную, многократно используя одну и ту же запись структуры. Если, например, в нее потребуется внести изменение (добавить новое поле, изменить тип существующего и т. д.), то придется делать это несколько раз, рискуя ошибиться и пропустить одно из описаний, особенно если они сделаны в разных местах программы.

Чтобы избежать этой проблемы и позволить программистам активно применять нужные структуры данных, в современных языках программирования разрешено определять собственные типы данных, которые допускается использовать в командах описания наравне с базовыми типами.

Синтаксис описания нового типа

Бейсик	Паскаль	Си++
Аналогичен описанию структуры, которое уже является описанием нового типа	type имя = описание;	typedef struct имя-структуры { поля-структуры; } имя; Имя структуры надо указывать только из-за требований синтаксиса. Реально оно нигде не применяется

Название нового типа можно использовать во всех последующих командах описания переменных.

Паскаль:

```
type TMyArray = array[0..99] of integer;
type TMyRecord = record
                 Item1: integer;
                 Item2: string;
                 end;
var MyArray: TMyArray;
var R: TMyRecord;
```

Си++:

```
typedef struct name1
{
int i;
float x;
} TNewStruct;
TNewStruct NewStruct;
```

Разделение операторов

Если записать подряд несколько операторов и не указать, где кончается один и начинается другой, то в процессе компиляции возникнет множество проблем с выделением отдельных операторов. Поэтому операторы в Паскале и Си++ отделяются друг от друга точкой с запятой «;» (каждый оператор в этих языках должен заканчиваться таким символом), а в Бейсике — двоеточием «:» или переходом на новую строку.

Блок операторов

Часто в программе возникает необходимость выполнить группу операторов (например, в зависимости от какого-либо условия). Такая группа объединяется в *блок* с помощью специальных скобок начала и конца блока, называемых *логическими скобками*.

В Бейсике явного понятия «блок операторов» нет, в Паскале для этого используются ключевые слова begin и end, а в Си++ — фигурные скобки «{» и «}».

Область действия переменных

Команды описания переменных могут встречаться в разных местах программы. При этом считается, что объявленные в них переменные являются локальными и их *область действия* — текущий блок, в котором они описаны. Как только встречается логическая скобка, закрывающая блок (например, «}»), соответствующая переменная перестает существовать, а выделенная для нее память освобождается.

Некоторые переменные описываются вне блоков и доступны из любого места программы.

Оператор присваивания

Оператор присваивания позволяет изменять текущее значение переменной. Синтаксис его очень простой. В левой части оператора присваивания указывается имя переменной, значение которой изменяется, а справа — выражение, значение которого будет записано в переменную. При этом старое значение, хранившееся в ней, безвозвратно пропадет.

Сам оператор присваивания записывается знаком «=» в Бейсике и Си++ и комбинацией двух знаков «:=» в Паскале (пробел между ними не допускается).

Например:

```
Result = 5
```

В переменную Result запишется число 5. Знак «=» означает именно присваивание, а не сравнение, которое может использоваться только в логических выражениях.

Другой пример:

```
X = X + 1
```

Сначала вычисляется значение выражения X+1, и затем оно заносится в переменную X. Допустима и такая запись:

```
X = X = X
```

Прежде всего выполняется сравнение в правой части (X = X), его значение всегда будет true, и значением переменной X, соответственно, тоже станет true. Для повышения наглядности оператора присваивания в Паскале принята специальная форма его записи:

```
X := X = X;
```

Примеры.

Бейсик:

```
a23 = a22(12) + 1: b1 = b1 - 1
```

Паскаль:

```
a := b*2 + c; d := (e[8] - f)*2.2;
```

Си++:

```
x[5] = y/3.33; y = z[0] - 0.001;
```

Комментарии

При составлении программы очень полезно комментировать различные участки кода, чтобы потом, обратившись к ним, сразу понять, что конкретно выполняется в том или ином месте программы. Забыть смысл того, что было сделано совсем недавно, можно очень быстро — за несколько недель, а в больших проектах и за несколько дней.

Эта проблема становится особенно актуальной, когда группой специалистов разрабатывается объемное приложение, и разобраться в сотнях тысяч строк своего и чужого исходного текста очень сложно.

Языки программирования допускают использование *комментариев* — частей исходных текстов, выделяемых с помощью специальных обозначений и пропускаемых компилятором при анализе текста программы.

Комментарии могут начинаться и заканчиваться особыми символами и охватывать несколько строк кода, а могут записываться только в конце строки — при этом считается, что весь остаток строки является комментарием.

Для обозначения комментариев в одном и том же языке программирования могут использоваться разные символы, поэтому возможно возникновение *вложенных комментариев*. Допустимость такого вложения задается, как правило, в настройках компилятора.

Синтаксис комментария

	Бейсик	Паскаль	Си++
Однострочный комментарий	REM или '	//	//
Многострочный комментарий	нет	{ } или (* *)	/* */

```
X = 5 'комментарий до конца строки
X := 5; // комментарий до конца строки
/*
это комментарий
языка Си++
*/
{
это комментарий
языка Паскаль
(* а это вложенный комментарий *)
}
```

Условный оператор (условные вычисления)

С помощью одного оператора присваивания можно создавать достаточно сложные расчетные программы, однако реализовать абсолютное большинство алгоритмов, просто последовательно выполняя операторы присваивания, невозможно. Постоянно приходится изменять порядок выполнения последовательности вычислений в зависимости от определенных *условий*. Эти условия записываются в виде логических выражений и всегда принимают одно из двух значений — true или false (истинно или ложно). При этом происходит *разветвление* программы — выполнение в дальнейшем может продолжиться с разных операторов.

Синтаксис условного оператора примерно одинаков во всех языках программирования — он представляет собой конструкцию:

если условие истинно

то выполнить оператор-1

иначе выполнить оператор-2

После ключевого слова IF (*если*) следует условие, и если оно истинно, то выполняется оператор или блок операторов, следующих за ключевым словом THEN (*то*); если же оно ложно, то выполняется оператор или блок операторов, следующих за ключевым словом ELSE (*иначе*).

Синтаксис условного оператора

Бейсик	Паскаль	Си++

IF условие THEN
оператор-1
ELSE
оператор-2
END IF
	if условие then
	оператор-1
	else
	оператор-2;
		if(условие)
		оператор-1
		else
		оператор-2;

Примеры.

Бейсик:

```
IF A <> 0 THEN
  A = 0
ELSE
  A = -1
END IF
```

Паскаль:

```
if a <> 0 then a := 0
          else a := -1;
```

Си++:

```
if( a <> 0 ) a = 0
else a = -1;
```

Вторую часть условного оператора, выполняющуюся в случае, если условие ложно, всегда можно опускать.

Бейсик:

```
IF x < 0 THEN
  y = x / 2
  x = 1
END IF
```

Паскаль:

```
if x < 0 then
  begin
  y := x / 2;
  x := 1;
  end
```

Си++:

```
if( x < 0 )
  {
  y = x / 2;
  x = 1;
  };
```

Повторяющиеся вычисления (операторы цикла)

С помощью условных операторов и операторов присваивания теоретически можно реализовать сколь угодно сложный алгоритм. Однако на практике при необходимости организовать обработку тысяч элементов массива (например, присвоить каж-

дому элементу начальное значение) вручную набирать тысячу операторов присваивания крайне тяжело.

Поэтому в языках программирования имеются средства для организации повторных вычислений, называемые *операторами цикла*. Они бывают двух видов: с фиксированным числом повторений и условные операторы цикла.

Каждый оператор цикла состоит из *заголовка цикла*, определяющего число повторений, и *тела цикла* — повторяемого оператора или блока операторов.

Первый вид оператора цикла

При решении задачи примерно в половине случаев заранее известно, сколько раз понадобится выполнить тело цикла. Так бывает, как правило, при обработке массивов, размер которых всегда или известен заранее, или легко определяется.

Заголовок такого оператора состоит из трех частей — *инициализации переменной-счетчика* или *параметра цикла* (присваивания ей начального значения), *определения конечного значения счетчика*, по достижении которого тело цикла надо выполнить в последний раз, и *приращения счетчика*, определяющего, на сколько будет меняться значение счетчика после каждого выполнения тела цикла.

Синтаксис оператора цикла

Бейсик	FOR счетчик = начальное_значение TO конечное_значение STEP приращение тело_цикла группа_операторов NEXT Если приращение не указывать, то считается, что оно равно 1
Паскаль	for счетчик := начальное_значение to конечное_значение do оператор или блок операторов; Приращение всегда равно 1
Си++	for(счетчик = начальное_значение; условие_завершения; счетчик = счетчик + приращение) оператор или блок операторов;

Примеры инициализации тысячи элементов массива a.

Бейсик:

```
FOR I = 1 TO 1000
   A(I) = 0
NEXT
```

Паскаль:

```
for i := 1 to 1000 do
   a[i] := 0;
```

Си++:

```
for( i = 0; i < 1000; i = i + 1 )
   a[i] = 0;
```

В последнем примере счетчик будет принимать значения от 0 до 999, потому что нумерация элементов массива в Си++ начинается с нуля.

Второй вид оператора цикла

Не менее часто встречаются ситуации, когда число повторений заранее неизвестно — надо выполнять цикл, пока не произойдет некоторое событие (пользователь нажмет на кнопку, точность вычислений уложится в заданный порог и т. д.). В таких ситуациях заголовок цикла упрощается. В нем указывается только условие (логическое выражение) — пока его значение равно true, цикл будет выполняться.

Синтаксис оператора цикла

Бейсик	Паскаль	Си++
DO WHILE условие группа операторов LOOP	while условие do оператор или группа операторов;	while(условие) оператор или группа операторов;

Бейсик:

```
DO WHILE A > B
  A = A - 0.01
LOOP
```

Паскаль:

```
while a > b do
   a := a - 0.01;
```

Си++:

```
while( a > b )
   a = a - 0.01;
```

Зацикливание

При использовании условных операторов цикла программиста подстерегает одна опасность. Как показывает практика, достаточно легко сделать ошибку и неверно задать условие окончания цикла, которое всегда будет истинным, — при этом тело цикла станет выполняться бесконечно. Подобная ситуация называется *зацикливанием*.

Например:

```
a = 0; b = 1;
while( a < b )
   a = a - 0.01;
```

Так как исходное значение переменной a меньше, чем значение переменной b, и это значение будет только уменьшаться, то подобный цикл никогда не закончится.

В некоторых случаях программисты специально применяют подобный трюк, чтобы организовать бесконечный цикл, в котором будут приниматься и обрабатываться

внешние сообщения (события). Тогда использование условного оператора цикла может выглядеть так:

```
while true do
  begin
// тело цикла
  end;
```

Контроль над выходом из цикла при наступлении определенного события при этом полностью возлагается на программиста.

В Бейсике есть специальная форма оператора цикла, позволяющая явно описывать такие бесконечные циклы:

```
DO
    ' тело цикла
LOOP
```

Исключения

Управление порядком выполнения программы может происходить не только с помощью условных операторов и операторов цикла, но и при возникновении *исключений* — ситуаций в программе или операционной системе, требующих немедленного реагирования. Например, при выполнении оператора присваивания и вычислении выражения произошло деление на ноль. Программа остановилась, так как не знает, что ей делать дальше, — ведь получено ошибочное значение. Чаще всего выполнение программы просто прекращается по ошибке, но современные системы разработки позволяют программисту явно контролировать возникновение самых разных исключений (они еще называются *исключительными ситуациями*, требующими немедленного вмешательства) и указывать, какие операторы следует выполнять при их возникновении.

Параллельные вычисления

Еще одна область программирования, в которой возможно изменение явно указанного порядка выполнения операторов, — это область параллельных вычислений. С появлением недорогих ПК с несколькими процессорами возникла возможность *распараллеливания* программы — одновременного выполнения ее независимых частей на разных процессорах, что теоретически позволяет получить выигрыш в быстродействии, линейно зависящий от числа процессоров. Однако на практике это очень сложная задача, которая требует правильного выделения независимых модулей кода (так называемых *процессов*), выполнение которых не скажется на результатах работы других процессов. Так как момент окончания работы того или иного процесса заранее неизвестен, то в программе надо предусмотреть действия, связанные с синхронизацией обработки получаемых результатов. Их выполнение может потребоваться в самые неожиданные моменты, поэтому изменение линейной последовательности работы операторов неизбежно.

Ввод и вывод

Чтобы получать от человека информацию для обработки и показывать результаты своей работы, программа должна иметь средства для организации *интерактивного* общения с пользователем (общения в реальном масштабе времени — человек щелкнул мышкой на кнопке и сразу получил ответ) и средства для ввода данных из файлов и сохранения данных в файлах. Интерактивное общение реализуется с помощью *RAD*-систем, позволяющих быстро спроектировать пользовательский интерфейс. Ввод и вывод информации осуществляется в разных языках по-разному. В Паскале и Бейсике есть операторы для такой работы, в Си++ они выделены в специальные библиотеки. Введен также специальный тип данных «файл» (FILE).

Работа с файлами всегда происходит в три этапа.

1. Файл *открывается* в одном из выбранных режимов (он рассматривается как последовательность строк или двоичных чисел, разрешается только считывать из него данные или только записывать и т. д.). Файл может состоять из последовательности одинаковых блоков, каждый из которых будет представлять собой копию структуры данных определенного типа, описанного в программе. Каждый такой блок называется *записью*.
2. Выполняется считывание, обновление или удаление записей в файле.
3. Файл *закрывается*. Если этого не сделать, то он останется открытым и в дальнейшем к нему нельзя будет обратиться из других программ.

Каждый из этих пунктов реализуется в каждом из языков программирования по-своему. Некоторые пункты требуют для своей реализации нескольких операторов.

Вопросы для самоконтроля

1. Какие типы данных считаются базовыми?
2. Приведите примеры арифметических и логических выражений.
3. Напишите формулу для вычисления среднего арифметического и среднего геометрического значений двух переменных.
4. В чем различие структуры и массива?
5. Зачем нужны комментарии?
6. С помощью условных операторов выполните проверку неравенства $x < y < z$.
7. Из каких частей состоит оператор цикла?
8. Назовите достоинства и недостатки параллельных вычислений.
9. Как организуется работа с файлами?

20.4. Структурное программирование

Подпрограммы

В предыдущем разделе рассматривались основные операторы и типы данных, необходимые для составления программ. При этом предполагалось, что текст программы

представляет собой линейную последовательность операторов присваивания, цикла и условных операторов. Таким способом можно решать не очень сложные задачи и составлять программы, содержащие несколько сот строк кода. После этого понятность исходного текста резко падает из-за того, что общая структура алгоритма теряется за конкретными операторами языка, выполняющими слишком детальные, элементарные действия. Возникают многочисленные вложенные условные операторы и операторы циклов, логика становится совсем запутанной, при попытке исправить один ошибочный оператор вносится несколько новых ошибок, связанных с особенностями работы этого оператора, результаты выполнения которого нередко учитываются в самых разных местах программы. Поэтому набрать и отладить длинную линейную последовательность операторов практически невозможно.

При создании средних по размеру приложений (несколько тысяч строк исходного кода) используется *структурное программирование*, идея которого заключается в том, что структура программы должна отражать структуру решаемой задачи, чтобы алгоритм решения был ясно виден из исходного текста. Для этого надо иметь средства для создания программы не только с помощью трех простых операторов, но и с помощью средств, более точно отражающих конкретную структуру алгоритма. С этой целью в программирование введено понятие *подпрограммы* — набора операторов, выполняющих нужное действие и не зависящих от других частей исходного кода. Программа разбивается на множество мелких подпрограмм (занимающих до 50 операторов — критический порог для быстрого понимания цели подпрограммы), каждая из которых выполняет одно из действий, предусмотренных исходным заданием. Комбинируя эти подпрограммы, удается формировать итоговый алгоритм уже не из простых операторов, а из законченных блоков кода, имеющих определенную смысловую нагрузку, причем обращаться к таким блокам можно по названиям. Получается, что подпрограммы — это новые операторы или операции языка, определяемые программистом.

Возможность применения подпрограмм относит язык программирования к классу *процедурных* языков.

Нисходящее проектирование

Наличие подпрограмм позволяет вести проектирование и разработку приложения *сверху вниз* — такой подход называется *нисходящим проектированием*. Сначала выделяется несколько подпрограмм, решающих самые глобальные задачи (например, инициализация данных, главная часть и завершение), потом каждый из этих модулей детализируется на более низком уровне, разбиваясь в свою очередь на небольшое число других подпрограмм, и так происходит до тех пор, пока вся задача не окажется реализованной.

Такой подход удобен тем, что позволяет человеку постоянно мыслить на предметном уровне, не опускаясь до конкретных операторов и переменных. Кроме того, появляется возможность некоторые подпрограммы не реализовывать сразу, а временно откладывать, пока не будут закончены другие части. Например, если имеется необходимость вычисления сложной математической функции, то выделяется

отдельная подпрограмма такого вычисления, но реализуется она временно одним оператором, который просто присваивает заранее выбранное значение (например, 5). Когда все приложение будет написано и отлажено, тогда можно приступить к реализации этой функции.

Немаловажно, что небольшие подпрограммы значительно проще отлаживать, что существенно повышает общую надежность всей программы.

Очень важная характеристика подпрограмм — это возможность их *повторного использования*. С интегрированными системами программирования поставляются большие библиотеки стандартных подпрограмм, которые позволяют значительно повысить производительность труда за счет использования чужой работы по созданию часто применяемых подпрограмм.

Рассмотрим пример, демонстрирующий методику нисходящего проектирования. Имеется массив Ocenki, состоящий из N ($N > 2$) судейских оценок (каждая оценка положительна). В некоторых видах спорта принято отбрасывать самую большую и самую маленькую оценки, чтобы избежать влияния необъективного судейства, а в зачет спортсмену идет среднее арифметическое из оставшихся оценок. Решим эту задачу, постепенно детализируя алгоритм (без привязки к конкретному языку программирования).

1. Процесс решения наиболее просто описывается подпрограммами:

   ```
   Ввести_оценки_в_массив;
   Удалить_самую_большую_оценку;
   Удалить_самую_маленькую_оценку;
   Рассчитать_среднее_арифметическое_оставшихся_оценок;
   Вывести_результаты;
   ```

   Теперь можно приступить к детализации каждой их этих подпрограмм.

2. Удалить_самую_большую_оценку;

   Как удалить самую большую оценку из статического массива? Вместо нее можно просто записать значение 0, а при подсчете среднего арифметического нулевые значения не учитывать.

   ```
   I = Номер_самого_большого_элемента_в_массиве;
   Ocenki[ I ] = 0;
   ```

3. Удалить_самую_маленькую_оценку;

   ```
   I = Номер_самого_маленького_элемента_в_массиве;
   Ocenki( I ) = 0;
   ```

   При реализации подпрограммы Номер_самого_маленького_элемента_в_массиве надо учесть, что искать придется самое маленькое из *положительных* значений (больших нуля).

4. Рассчитать_среднее_арифметическое_оставшихся_оценок;

Здесь потребуется оператор цикла, вычисляющий сумму всех элементов массива Ocenki.

```
SUM = 0
FOR I = 1 TO N
   SUM = SUM + Ocenki( I )
NEXT
SUM = SUM / (N - 2)
```

В последнем операторе происходит вычисление среднего арифметического всех оценок. Сумма элементов массива делится на число элементов, уменьшенное на 2, потому что две оценки, самую большую и самую маленькую, учитывать не надо.

Если бы эта задача решалась последовательно, то уже на этапе удаления оценок могли возникнуть определенные проблемы.

Реализацию подпрограмм Номер_самого_большого_элемента_в_массиве и Номер_самого_маленького_элемента_в_массиве выполните самостоятельно.

Процедуры и функции

Подпрограммы бывают двух видов — *процедуры* и *функции*. Отличаются они тем, что процедура просто выполняет группу операторов, а функция вдобавок вычисляет некоторое значение и передает его обратно в главную программу (*возвращает* значение). Это значение имеет определенный тип (говорят, что функция *имеет* такой-то тип).

В Си++ понятия «процедура» нет — там имеются только функции, а если никакого значения функция не вычисляет, то считается, что она возвращает значение типа «никакое» (void).

Параметры подпрограмм

Чтобы работа подпрограммы имела смысл, ей надо получить данные из внешней программы, которая эту подпрограмму *вызывает*. Данные передаются подпрограмме в виде *параметров* или *аргументов*, которые обычно описываются в ее заголовке так же, как переменные.

Управление последовательностью вызова подпрограмм

Подпрограммы вызываются, как правило, путем простой записи их названия с нужными параметрами. В Бейсике есть оператор CALL для явного указания того, что происходит вызов подпрограммы.

Подпрограммы активизируются *только* в момент их вызова. Операторы, находящиеся внутри подпрограммы, выполняются, только если эта подпрограмма явно вызвана. Пока выполнение подпрограммы полностью не закончится, оператор главной программы, следующий за командой вызова подпрограммы, выполняться не будет.

Подпрограммы могут быть *вложенными* — допускается вызов подпрограммы не только из главной программы, но и из любых других подпрограмм.

В некоторых языках программирования допускается вызов подпрограммы из себя самой. Такой прием называется *рекурсией* и потенциально опасен тем, что может привести к зацикливанию — бесконечному самовызову.

Структура подпрограммы

Подпрограмма состоит из нескольких частей: заголовка с параметрами, тела подпрограммы (операторов, которые будут выполняться при ее вызове) и завершения подпрограммы.

Локальные переменные, объявленные внутри подпрограммы, имеют областью действия только ее тело.

Функции

	Бейсик	**Паскаль**	**Си++**
Заголовок функции	FUNCTION имя (список_параметров) Тип возвращаемого значения определяется специальным символом после имени функции	function имя (список_параметров): тип_функции;	тип_функции имя(список_параметров)

Тело	Последовательность операторов	begin
		последовательность операторов
		end;
			{
			последовательность операторов
			};

Завершение	END FUNCTION	нет	нет

Процедуры

	Бейсик	**Паскаль**	**Си++**
Заголовок процедуры	SUB имя (список_параметров)	procedure имя (список_параметров);	void имя(список_параметров)

Тело	Последовательность операторов	begin
		последовательность операторов
		end;
			{
			последовательность операторов
			};

Завершение	END SUB	нет	нет

Как функция возвращает значение

После того как функция рассчитала нужное значение, ей требуется явно вернуть его в вызывающую программу. Для этого может использоваться специальный оператор (return в Си++) или особая форма оператора присваивания, когда в левой части указывается имя функции, а справа — возвращаемое значение.

Далее приведены примеры функции, вычисляющей значение квадрата аргумента.

Бейсик:

```
FUNCTION SQR% (X AS INTEGER)
SQR% = X*X
END FUNCTION
```

Паскаль:

```
function SQR(X: integer): integer;
begin
SQR := X*X
end;
```

Си++:

```
int SQR(int x)
{
return x*x;
};
```

Формальные и фактические параметры

Во время создания подпрограммы заранее не известно, какие конкретно параметры она может и будет получать. Поэтому в качестве переменных, выступающих в роли ее аргументов в заголовке, могут использоваться произвольные допустимые названия, даже совпадающие с уже имеющимися. Компилятор все равно поймет, что это не одно и то же.

Параметры, которые указываются в заголовке подпрограммы, называются *формальными*. Они нужны только для описания тела подпрограммы. А параметры (конкретные значения), которые указываются в момент вызова подпрограммы, называются *фактическими* параметрами. При выполнении операторов подпрограммы формальные параметры как бы временно заменятся на фактические.

Пример.

```
int a, y;
a = 5;
y = SQR(a);
```

Программа вызывает функцию SQR() с одним фактическим параметром a. Внутри подпрограммы формальный параметр x получает значение переменной a и возводится в квадрат. Результат возвращается обратно в программу и присваивается переменной y.

Событийно-ориентированное программирование

С активным распространением системы *Windows* и появлением визуальных *RAD*-сред широкую популярность приобрел событийный подход к созданию программ — *событийно-ориентированное программирование*.

Идеология системы *Windows* основана на событиях. Щелкнул человек на кнопке, выбрал пункт меню, нажал на клавишу или кнопку мыши — в *Windows* генерируется подходящее *сообщение*, которое отсылается окну соответствующей программы.

Структура программы, созданной с помощью событийного программирования, следующая. Главная часть представляет собой один бесконечный цикл, который опрашивает *Windows*, следя за тем, не появилось ли новое сообщение. При его обнаружении вызывается подпрограмма, ответственная за *обработку* соответствующего события (обрабатываются не все события, их сотни, а только нужные), и подобный цикл опроса продолжается, пока не будет получено сообщение «Завершить работу».

События могут быть *пользовательскими*, возникшими в результате действий пользователя, *системными*, возникающими в операционной системе (например, сообщения от таймера), и *программными*, генерируемыми самой программой (например, обнаружена ошибка и ее надо обработать).

Событийное программирование является развитием идей нисходящего проектирования, когда постепенно определяются и детализируются реакции программы на различные события.

Вопросы для самоконтроля

1. С какой целью применяют подпрограммы?
2. Чем характеризуются процедурные языки программирования?
3. В чем состоит идея нисходящего проектирования?
4. Что общего и в чем отличия процедуры и функции?
5. Определите значение выражения `F(1,2) + F(10,0.1)`, если функция `F(a,b)` рассчитывается как `a*a + b*b`.
6. В чем различие между событийным и структурным программированием?
7. Как организуется обработка программных событий?

20.5. Объектно-ориентированное программирование

Понятие объекта

Развитие идей структурного и событийного программирования существенно подняло производительность труда программистов и позволило в разумные сроки (несколько месяцев) создавать приложения объемом в сотни тысяч строк. Однако такой объем уже приблизился к пределу возможностей человека, и потребовались новые технологии разработки программ.

В начале 80-х годов в программировании возникло новое направление, основанное на понятии *объекта*. До того времени основные ограничения на возможность создания больших систем накладывала разобщенность в программе данных и методов их обработки.

Реальные объекты окружающего мира обладают тремя базовыми характеристиками: они имеют набор свойств, способны разными методами изменять эти свойства и реагировать на события, возникающие как в окружающем мире, так и внутри самого

объекта. Именно в таком виде в языках программирования и реализовано понятие *объекта* как совокупности *свойств* (структур данных, характерных для этого объекта), *методов* их обработки (подпрограмм изменения свойств) и *событий*, на которые данный объект может реагировать и которые приводят, как правило, к изменению свойств объекта.

Появление возможности создания объектов в программах качественно повлияло на производительность труда программистов. Максимальный объем приложений, которые стали доступны для создания группой программистов из 10 человек, за несколько лет увеличился до миллионов строк кода, при этом одновременно удалось добиться высокой надежности программ и, что немаловажно, повторно использовать ранее созданные объекты в других задачах.

Класс

Объекты могут иметь идентичную структуру и отличаться только значениями свойств. В таких случаях в программе создается новый тип, основанный на единой структуре объекта (по аналогии с тем, как создаются новые типы для структур данных). Он называется *классом*, а каждый конкретный объект, имеющий структуру этого класса, называется *экземпляром класса*.

Описание нового класса

Описание нового класса похоже на описание новой структуры данных, только к полям (свойствам) добавляются методы — подпрограммы.

В Си++ и Паскале для описания класса используется ключевое слово class.

Паскаль:

```
class TMyClass
Item1: integer;
Item2: string;
function GetSum(n: integer): integer;
procedure Initialize;
end;
```

Си++:

```
class TMyClass
{
int Item1;
int Item2;
int GetSum(int n);
void Initialize();
};
```

При определении подпрограмм, принадлежащих конкретному классу, его методов, в заголовке подпрограммы перед ее названием явно указывается, к какому классу

она принадлежит. Название класса от названия метода отделяют специальные символы (точка в Паскале или два двоеточия в Си++).

Паскаль:

```
procedure TMyClass.Initialize;
begin
Item1 := 1;
Item2 := «»;
end;
```

Си++:

```
void TMyClass::Initialize()
{
Item1 = 1;
Item2 = 0;
}
```

Класс — это тип данных, такой же, как любой другой базовый или сложный тип. На его основе можно описывать конкретные объекты (экземпляры классов).

Паскаль:

```
var C1, C2: TMyClass;
```

Си++:

```
TMyClass C1, C2;
```

Доступ к свойствам объектов и к их методам осуществляется так же, как к полям записей, через точку:

```
C1.Item1 := 5;
C2.Initialize;
x := C1.GetSum(21);
```

Объектно-ориентированное программирование базируется на трех ключевых концепциях — инкапсуляции, наследовании и полиморфизме. Объединение данных с методами в одном типе (классе) называется *инкапсуляцией*. Помимо объединения, инкапсуляция позволяет ограничивать доступ к данным объектов и реализации методов классов. В результате у программистов появляется возможность использования готовых классов в своих приложениях на основе только описаний этих классов.

Наследование

Важнейшая характеристика класса — возможность создания на его основе новых классов с *наследованием* всех его свойств и методов и добавлением собственных. Класс, не имеющий предшественника, называется *базовым*.

Например, класс «животное» имеет свойства «название», «размер», методы «идти» и «размножаться». Созданный на его основе класс «кошка» наследует все эти свойства и методы, к которым дополнительно добавляется свойство «окраска» и метод «пить».

Наследование позволяет создавать новые классы, повторно используя уже готовый исходный код и не тратя времени на его переписывание.

Полиморфизм

В большинстве случаев методы базового класса у классов-наследников приходится переопределять — объект класса «кошка» выполняет метод «идти» совсем не так, как объект класса «амеба». Все переопределяемые методы по написанию (названию) будут совпадать с методами базового объекта, однако компилятор по типу объекта (его классу) распознает, какой конкретно метод надо использовать, и не вызовет для объекта класса «кошка» метод «идти» класса «животное». Такое свойство объектов переопределять методы наследуемого класса и корректно их использовать называется *полиморфизмом*.

Визуальное программирование

Технологии объектного, событийного и структурного программирования сегодня объединены в *RAD*-системах, которые содержат множество готовых классов, представленных в виде визуальных *компонентов*, которые добавляются в программу одним щелчком мыши. Программисту надо только спроектировать внешний вид окон своего приложения и определить обработку основных событий — какие операторы будут выполняться при нажатии на кнопки, при выборе пунктов меню или щелчках мышкой. Весь вспомогательный исходный код среда сгенерирует сама, позволяя программисту полностью сосредоточиться только на реализации алгоритма.

Вопросы для самоконтроля

1. Для чего в языки программирования было введено понятие класса?
2. В чем различие между классом и объектом?
3. Поясните понятие инкапсуляции на бытовых примерах.
4. Для чего применяется механизм наследования?
5. Как полиморфизм модифицирует принцип наследования?
6. Опишите использование принципов объектно-ориентированного программирования в средах быстрого проектирования.

20.6. Проектирование программ

Программирование как вид деятельности

Появление первых компьютеров породило программирование как *науку*. Разрабатывались первые математические теории обработки информации, средства доказательства правильности программ, оптимизации кода, создания эффективных компиляторов, формального тестирования и т. д. Затем, с появлением универсальных языков программирования третьего поколения, эти аспекты стали менее актуальными — исследования шли и идут в основном в области автоматической генерации исходных текстов и повышения эффективности компиляторов. Программирование превратилось в *искусство* — миллионы людей, не имевших специального образования, получили возможности применять компьютеры для решения собственных

прикладных задач, что потребовало от них мастерства создавать правильно работающие программы. Искусством программирование остается и сегодня для профессиональных разработчиков и любителей, создающих программы в одиночку или в небольших компаниях, где все решает индивидуальное мастерство.

Вместе с тем, при росте спроса со стороны государственных и частных организаций на все более и более сложные системы автоматизации предприятий, надежные операционные среды, комплексы глобального телекоммуникационного управления, возникла необходимость в постановке процесса разработки программного обеспечения (ПО) на поток, превращения программирования в *ремесло*. Было разработано несколько методологий и стандартов, позволивших эффективно организовывать труд сотен программистов средней квалификации, точно укладываться в отпущенные сроки и средства и не зависеть от настроения нескольких талантливых ведущих специалистов. Отрицательная сторона подобных методологий — отсутствие творческого элемента в работе и своеобразная конвейерная «потогонная» система промышленного производства программ, которая, будучи внедренной в организации, в условиях жесточайшего дефицита программистов во всем мире может только отпугнуть сотрудников.

Потенциальные возможности человека

Объем проекта, строк исходного кода	Тип программы	Время создания	Вероятность успешного завершения	Число программистов
100	Утилиты для временных нужд	1 день	100%	1
1000	Небольшие приложения и дополнения, вносимые в готовые системы	до 1 месяца	100%	1
10 000	Типичная средняя программа, разрабатываемая на заказ	до 6 месяцев	85%	1 (предел возможностей среднего программиста)
100 000	Большинство современных коммерческих автономных и небольших клиент-серверных приложений	1 год	85% для групп, 35% для одиночки	10
1 млн	Крупные системы автоматизации	1,5–5 лет	50% для группы, 0% для одиночки	100
10 млн	Операционные системы (Microsoft Windows, IBM VMS), большие военные комплексы. Предел сегодняшних возможностей. Стоимость подобной разработки может равняться стоимости большого стадиона или крупного корабля	5–8 лет	35%	до тысячи

Экономические аспекты программирования

Когда на свет появились первые компьютеры, одна минута их работы стоила очень дорого, а задачи решались достаточно простые, поэтому в расходах на подготовку программ труд разработчиков составлял небольшую часть. С появлением ПК и ростом спроса на большие программные системы практически всю расходную часть проекта стала составлять зарплата программистов. Как видно из таблицы, большой процент таких проектов заканчивается неудачно, а расходы на них очень велики, поэтому проблемы создания качественного программного обеспечения точно в срок и в рамках бюджета сегодня самые важные и над созданием эффективных методологий производства ПО трудятся специалисты во всех развитых странах.

Этапы разработки программ

Программы небольшого и среднего размера (несколько тысяч строк) создаются, как правило, в два этапа. Сначала необходимо точно установить, что надо сделать, продумать соответствующий алгоритм, определить структуры данных, объекты и взаимодействие между ними (это этап *системного анализа*), а затем выразить этот алгоритм в виде, понятном машине (этап *кодирования*). Если же разрабатывается крупный проект объемом от десятков тысяч до миллионов строк кода, тогда приходится применять специальные методологии проектирования, охватывающие *период разработки ПО*.

Период разработки ПО

Рассмотрим классический период разработки ПО.

1. Формируются и анализируются *требования* к проекту. Этот этап самый важный, так как неправильное формулирование требований приводит к выполнению ненужной работы, а недооценка сложности вызывает перерасход средств и времени. Сегодня около 60% крупных проектов завершаются неудачей именно из-за ошибок на стадии подготовки требований.

   На основе требований по различным методикам определяется примерный объем проекта и его трудоемкость, рассчитываются будущие трудозатраты и определяется его *стоимость*. Так как требования к проекту во время работы над ним могут уточняться и меняться, а выполнение требований надо отслеживать, применяются специальные программы для управления требованиями.

   Часто заказчик не в состоянии точно выразить, чего он хочет, и задача специалистов по системному анализу — помочь ему выразить свои требования в виде, пригодном для формализации. После согласования требований подписывается *контракт на разработку ПО*. В дальнейшем любые отклонения от сформулированных требований к продукту (как со стороны заказчика, так и со стороны исполнителя) рассматриваются как нарушение контракта.

   Каждый этап требует от заказчика вложения собственных трудозатрат и привлечения высокопрофессиональных специалистов, поэтому он обязательно должен оплачиваться — бесплатно выполнять сложную работу никто не будет. Однако, хотя первый этап самый важный, заказчик редко понимает эту важ-

ность и не готов платить достаточно большие суммы. Примерный объем работ на этом этапе — 5% от объема всего проекта.

2. Начинается предпроектное обследование объекта автоматизации. С помощью *CASE*-средств составляется формальная модель его работы, модель базы данных, объектов и потоков информации. На этом этапе привлекаются специалисты заказчика и эксперты, хорошо знакомые с предметной областью, для которой составляется задача.

   Примерный объем работ второго этапа — 10% от общего.

3. На основе формальной модели составляется подробное *техническое задание* для программистов, *спецификации* отдельных модулей, таблицы баз данных, другая сопроводительная документация. Готовится подробный *календарный план работ*, где указываются все сроки, конкретные исполнители и выполняемые объемы работ (понедельно, помесячно). Для составления планов работ имеется немало хороших программ, ориентированных как на небольшие группы программистов, так и на коллективы из сотен сотрудников, выполняющих тысячи различных работ в рамках общего плана.

   Примерный объем работ третьего этапа — 10% от общего.

4. Выбирается методология разработки ПО и начинается разработка (кодирование). Крупные компании имеют собственные методологии, ориентированные на конкретные задачи (как правило, это задачи автоматизации предприятий), однако все методологии имеют общие черты и более-менее серьезно различающихся известно около двух десятков.

   Мы коснулись, в частности, методологий структурного (нисходящего) и объектного проектирования. Хотя объектный подход, безусловно, один из самых передовых, он подразумевает высокую квалификацию всех исполнителей без исключения и поэтому распространен не очень широко.

   Достаточно популярна методология *итерационного проектирования*, ориентированная на использование *RAD*-средств и систем автоматической генерации исходных текстов на основе созданной формальной модели. Такой подход хорош тем, что позволяет быстро создать первый *работающий* прототип программы, когда еще требования к ней окончательно не определены, а в дальнейшем, на следующих итерациях (их обычно требуется от двух до пяти), постепенно детализировать и реализовывать конкретные возможности, пропущенные по каким-то причинам на предыдущей итерации. Эта методология немного отличается от нисходящего проектирования тем, что применяется, когда окончательные требования неизвестны и могут меняться, а основные работающие функции нужны заказчику как можно быстрее (заказчик чаще хочет получить приложение, законченное на 80%, сегодня, чем законченное на 100% завтра). При нисходящем проектировании основная структура задачи должна быть определена заранее.

   Принятие решения о выборе подходящей методологии — очень ответственный процесс. Человек, принимающий такое решение, должен обладать богатым опытом и знаниями в области создания ПО. Многое зависит от инфраструктуры

заказчика — какие у него компьютеры, операционные системы, каковы их ресурсы. В соответствии с этим выбираются и средства разработки. Иногда бывает, что лучше всего подходит, например, итерационная методология, но для операционной системы заказчика нет хорошей *RAD*-среды и приходится остановиться на менее эффективной методологии.

В процессе разработки необходимо:

- непрерывно поддерживать *обратную связь* с заказчиком, чтобы следить за правильностью реализации требований;
- непрерывно контролировать ход работ в соответствии с планом и при отклонениях от него принимать экстренные меры.

Примерный объем этих работ — 10% от общего объема проекта.

5. Когда программа закончена (готова работоспособная *альфа-версия*), она поступает к *тестерам* компании-исполнителя, которые начинают проверять ее на наличие ошибок и сообщать о найденных ошибках программистам. Анализируется, в частности, устойчивость работы программы при вводе недопустимых или критических значений, при отсутствии информации, при неверных действиях, при сбоях аппаратуры, в стрессовых режимах и т. п. Когда число ошибок, выявляемых за определенный срок (неделя, месяц), снижается ниже экспериментально подобранного уровня (на основе аналогичных проектов), начинается *бета-тестирование* программы у заказчика. К такому тестированию привлекается максимально возможное число сотрудников, и программа уже начинает частично функционировать в рабочем режиме.

   Примерный объем этих работ — 10% от общего объема проекта.

6. После того как заказчик удовлетворен качеством продукта, начинается его *внедрение* — подготовка к окончательному запуску в эксплуатацию. Если приложение многопользовательское, нередко требуется сформировать и настроить локальную сеть, установить серверы, инсталлировать вспомогательные программы. Очень много проблем возникает при переходе со старых программ (например, бухгалтерского учета, расчета зарплаты, работы с персоналом), которые создавались разными сотрудниками и покупались в разных фирмах (так называемая *лоскутная автоматизация*), к новой интегрированной системе. Необходимо выверить, ввести или перенести множество жизненно важной информации: всю бухгалтерскую и финансовую отчетность, данные о хранимом оборудовании и множество других сведений. Этот этап самый трудоемкий и «нудный» и занимает порой до 90% времени всего проекта. Для систем автоматизации больших предприятий он растягивается на годы.

7. После того как новая система готова к работе, сотрудников организации заказчика нужно *обучить* работе с этой системой, потому что книг о ней не написано, да и содержится в такой системе, внедренной на конкретном предприятии, множество нюансов, связанных со спецификой работы.

   Примерный объем трудозатрат на обучение — 5% от общего объема проекта.

8. После того как заказчик подписывает *акт приемки*, проект считается завершенным, но связь с исполнителем не теряется. Особенно в первое время у пользователей системы постоянно будет возникать множество вопросов по работе с ней. Неизбежно и возникновение ошибок, которые требуется устранять. Кроме того, исполнитель может выпускать новые версии системы, и старая система потребует *обновления*. Сотрудничество с заказчиком по обслуживанию системы называется *сопровождением*. Оно бесплатно на определенный гарантийный срок (например, год).

   Реально объем непосредственного программирования и отладки (тестирования) в цикле разработки невелик. Он составляет 10-20% от общего объема работ.

Контроль качества

Чем крупнее проект, тем больше в нем ошибок. При этом слишком затягивать этап тестирования нельзя — нарушаются сроки, растет недовольство потребителей, и на рынок выпускается «сырая» система с множеством ошибок, которые устраняются уже в процессе эксплуатации выпуском многочисленных «заплаток».

Современные технологии создания надежного ПО предусматривают *непрерывный сквозной контроль качества* разрабатываемого продукта на всех этапах жизненного цикла — от анализа требований до внедрения и сопровождения, а не только на этапе тестирования. Качество каждой работы в плане формализуется числовой величиной с помощью специальных методик, но его можно контролировать, только оптимальным образом организовав работу большой группы аналитиков и программистов. Для этого надо иметь возможность *отслеживать* вносимые в проект *изменения* — изменения требований, формальных моделей, сопроводительной документации, версий исходных текстов, хода выполнения календарного плана по разработке, тестированию, внедрению, сопровождению, а также *контролировать и управлять* всеми этапами периода создания программы — *процессом* разработки ПО. Для этого предназначены *системы конфигурационного управления* — сложные и дорогие (десятки и сотни тысяч долларов) продукты, однако без них крупный проект, скорее всего, обречен на неудачу.

Системы не очень сложного конфигурационного управления, охватывающие контроль версий исходных текстов и ряд других аспектов работы группы программистов, встроены, в частности, в такие системы, как *Delphi* 7 и *Visual Studio .NET*.

Стандарты качества ПО

Компания может организовать у себя очень эффективный процесс разработки ПО, однако заказчик вполне обоснованно может ей не поверить. Существует международная система сертификации компаний по стандарту качества *ISO* 9000, которая гарантирует, что данная компания выполняет программные проекты в срок и с высоким качеством. Процесс сертификации сложен и требует подготовительной работы в течение нескольких лет.

В университете Карнеги-Меллона в США несколько лет назад была разработана специальная методология *CMM* (*Capability Maturity Model for Software*), позволя-

ющая сертифицировать компании по одному из 5 уровней «зрелости» процесса разработки ПО. Согласно результатам 20-летних исследований Министерства обороны США оказалось, что главная причина слишком частых неудач при разработке крупных информационных проектов заключается прежде всего в неумении менеджеров управлять процессом создания качественного ПО.

В отличие от стандарта *ISO* 9000, который просто подтверждает качественную работу компании на основании достаточно общих критериев, методология *CMM* ориентирована именно на качество управления процессом разработки и имеет множество конкретных рекомендаций и указаний по способам организации всех этапов создания ПО. Сегодня в США невозможно получить крупный государственный или военный заказ на создание программного продукта стоимостью более 2 млн. долларов, если компания не сертифицирована как минимум по третьему уровню *CMM*. Сегодня на смену этой модели приходит новая интеграционная концепция *CMMI*, дополняющая процессы разработки не менее важными вопросами организации деятельности персонала, общения с подрядчиками, выбора программного обеспечения и т. д.

Повышение индивидуального мастерства

На основе методологии *CMM* была создана методология *PSP* (*Personal Software Process*), ориентированная на индивидуальных разработчиков. Она позволяет в несколько раз повысить качество создания программ, значительно поднять собственную производительность и научиться предсказывать сроки выполнения работ.

Методология *PSP* состоит из 7 этапов самосовершенствования, и, чтобы хорошо ее освоить, надо закончить специальные курсы. Однако даже простое знакомство с ее идеологией позволит любому программисту значительно улучшить свою работу.

Перед началом проекта составляется подробный календарный план работ и производится попытка оценить его объем в строках кода и рабочих днях. Весь процесс работы детально хронометрируется, а найденные ошибки подробно описываются — накапливается статистика. По окончании проекта весь процесс тщательно анализируется и делаются выводы о том, что можно улучшить в своей работе, каких ошибок надо стараться избегать, какова реальная производительность труда и т. п. Сегодня лучшая характеристика программиста — это не просто знание Си++ или *Delphi*, а способность планировать свой труд, разрабатывать программу точно в срок и без ошибок.

Гибкие методики

В последние годы наряду с ростом интереса к «тяжелым» сертификационным методологиям значительно увеличилась роль и популярность так называемых гибких, проворных (*agile*) методик. Они не требуют значительных усилий по реорганизации компании-разработчика и могут быть внедрены не за годы, а за недели. Наиболее известные среди них — экстремальное программирование, *MDA* (архитектура системы, управляемая моделью), *Scrum* и другие.

Гибкие методики не навязывают исполнителю множество формализованных действий по разработке ПО. Они предоставляют участникам проекта большую свободу

(но и требуют от них большей ответственности) и акцентируют их внимание на важнейших задачах соблюдения требований заказчика, сроков и качества работы.

Методы маркетинга программного обеспечения

Коммерческое ПО. При создании программного продукта *издатель*, выполнив анализ рынка, заказывает у *исполнителя* разработку такого ПО, которое должно пользоваться на рынке спросом, и выделяет на его создание деньги. По окончании работ издатель получает все *имущественные права* на созданный продукт (право на тиражирование, продажу под собственной торговой маркой, право на получение дохода от программы любым способом). При этом может быть оговорено получение исполнителем некоторого процента (*роялти*) с каждой проданной копии (как правило, для программ, издающихся сотнями тысяч или миллионами копий, роялти составляет 1-3%) — тогда он получает меньшую сумму на разработку или вообще создает программу за свой счет. Если же отчисления не предусмотрены, то все расходы по подготовке программы издатель берет на себя. Он также вкладывает средства в упаковку, рекламную кампанию, организацию сетей сбыта и т. д. Издатель обеспечивает расходы, связанные с сопровождением продукта и технической поддержкой пользователей.

За исполнителем навечно остаются *авторские права* на программу — право указывать свое имя или логотип своей фирмы на начальной заставке, в документации, на упаковочной коробке.

Крупные компании имеют и подразделения разработки ПО, и отделы, занимающиеся его распространением, что помогает эффективно организовать весь процесс от производства программ до доставки их потребителю.

Условно-бесплатное ПО (shareware). В связи с активным развитием Интернета огромное число индивидуальных разработчиков получили возможность распространения своих программ по всему миру. Не имея средств на рекламные кампании, они предоставляют возможность получения *ознакомительных версий* их программ (демонстрационных или имеющих искусственные ограничения) через Интернет. Если человеку эта программа нравится, он оплачивает небольшую сумму и получает полную работоспособную версию.

В Интернете есть немало узлов, которые предлагают бесплатные услуги по размещению таких программ. Отечественным *shareware*-разработчикам можно порекомендовать сайт www.swrus.com, на котором можно найти множество материалов и форумов по всем вопросам организации *shareware*.

Бесплатное ПО (freeware, public domain). Такие программы не имеют никаких ограничений, однако автор может *попросить* заплатить ему некоторую сумму, не настаивая, впрочем, на этом (это метод *freeware*). Некоторые программы авторы называют «общественным достоянием» (*public domain*), ничего взамен не требуют и нередко распространяют такое ПО в исходных текстах.

Как правило, стимулом к созданию бесплатных программ служит стремление повысить собственную квалификацию, установить контакты с коллегами, а в случае удачно созданной программы получить известность.

Вопросы для самоконтроля

1. В чем трудности разработки крупных программных проектов?
2. Опишите организацию работы над сложной программной системой.
3. Какой этап разработки проекта является наиболее ответственным?
4. Какова роль собственно программирования в ходе работы над проектом?
5. Опишите известные вам методы контроля качества программного обеспечения.
6. Каковы основные методы распространения программного обеспечения?

20.7. Пример на Бейсике. Разведение кроликов

В данном и последующих разделах рассматриваются три примера, реализованные с помощью разных систем программирования: *QBasic* корпорации *Microsoft* (интерпретирующая версия Бейсика для операционной системы *MS-DOS*), *Borland Delphi 7* (система визуального программирования на Паскале) и *Microsoft Visual Studio .NET* (система программирования на Си++). Эти примеры включают в себя описание основных приемов работы с данными системами.

Постановка задачи

Итальянский математик Леонардо Фибоначчи придумал оригинальную числовую последовательность, названную в его честь, которая описывает рост численности поколений кроликов. Считается, что каждый год каждая пара животных приносит приплод — новую пару (самца и самку), которые в свою очередь начинают давать приплод через два года (смертность не учитывается). То есть каждый следующий член последовательности равен сумме двух предыдущих, а классическая последовательность Фибоначчи выглядит так:

1, 1, 2, 3, 5, 8, 13, 21, ...

Надо определить, через сколько лет будет достигнута популяция в N особей.

Запуск QBasic

Интерпретатор *QBasic* входит в стандартную поставку *MS-DOS* и расположен обычно в каталоге \DOS. Программа-интерпретатор называется qbasic.exe. После ее запуска на экране появится приветствие, которое пропускается нажатием на клавишу ENTER, после чего *QBasic* вызывает встроенную справочную систему на английском языке. Она закрывается нажатием клавиши ESC.

Рабочая область экрана (рис. 20.1) поделена на две части. В нижней части, в окне Immediate (Немедленное выполнение) можно вводить операторы Бейсика и тут же их выполнять.

Вывод на экран

Каждый язык программирования имеет оригинальные средства вывода информации, сильно зависящие от операционной системы. В Бейсике реализован оператор PRINT, который выводит значение следующего за ним выражения на экран, в новую строку.

Для перехода в окно Immediate (Немедленное выполнение) надо нажать клавишу F6. Чтобы сразу получить ответную реакцию от *QBasic*, достаточно набрать оператор

```
PRINT 2+2
```

и нажать клавишу ENTER, чтобы этот оператор выполнился.

На экране вывода появится число 4 (результат вычисления выражения 2+2), а в нижней строке — сообщение Press any key to continue (Нажмите любую клавишу для продолжения). Чтобы вернуться в *QBasic*, надо это сделать.

В операторе вывода можно указывать несколько значений через запятую, тогда они будут выведены в одной строке. Ранее введенный оператор можно изменить, добавив к нему текстовую подсказку:

```
PRINT "Сумма = "; 2+3
```

Если теперь нажать клавишу ENTER, то на экране вывода в новой строке (под ранее напечатанной четверкой) появится фраза

```
Сумма = 5
```

Редактор программы

Таким образом можно познакомиться с работой разных операторов Бейсика, однако выполнять их удастся только поодиночке. Чтобы выполнить группу операторов, их надо объединить в программу.

Набор и редактирование исходного текста программы осуществляется в верхнем окне интерпретатора. Для перехода в него используется клавиша F6.

Решать данную задачу удобнее всего с помощью нисходящего метода. В программе будет бесконечный главный цикл, в котором осуществляется ввод очередного значения количества особей, происходит расчет числа лет, необходимых для размножения, полученный результат печатается, и цикл повторяется снова. Если человек вводит ноль, это будет означать, что программу надо завершить.

Ввод информации от пользователя

В Бейсике ввести в переменную значение с экрана можно с помощью оператора INPUT. Сначала указывается необязательная текстовая подсказка, а потом — имя переменной. Например, оператор

```
INPUT "Введите число: ", x
```

при выполнении напечатает в новой строке подсказку

```
Введите число:
```

и будет ожидать, когда пользователь введет число и нажмет клавишу ENTER. В результате в переменную x запишется новое, введенное с клавиатуры значение.

Главная часть программы

Ввод и редактирование текста программы осуществляется во встроенном редакторе *QBasic*, правила работы с которым аналогичны правилам работы с большинством известных текстовых редакторов.

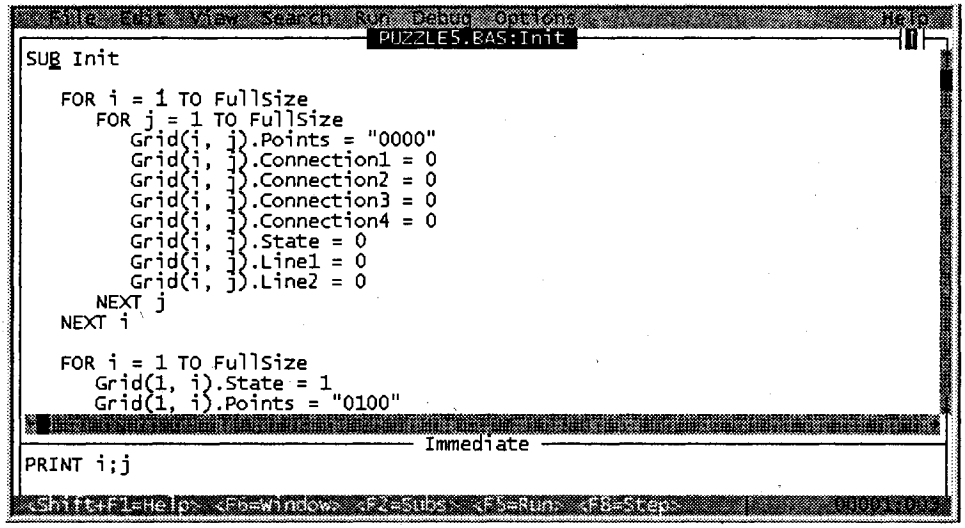


Рис. 20.1. Окно программы QBasic

Главная часть программы набирается в этом редакторе и должна выглядеть так (комментарии вводить не обязательно):

```
' описание переменной N — числа особей
DIM N AS INTEGER
' начало бесконечного цикла
DO
' ввод числа особей в переменную N
  INPUT "Введите количество особей: ", N
' если введен 0, то
   IF N = 0 THEN
' закончить программу
      END
   END IF
' напечатать результат:
   PRINT "Требуемое число лет: ", Years%(N)
' продолжить цикл с начала
LOOP
```

В тексте используется оператор **END**, который предназначен для немедленного завершения работы программы. Операторы, вложенные в цикл и в условные операторы, выделяются отступами, чтобы структура текста была более понятной и наглядной.

Основная, глобальная часть алгоритма реализована. Осталось «спуститься вниз» и запрограммировать функцию Years%(), которая в качестве аргумента получает количество особей и возвращает число лет, требуемое для их разведения.

Типы данных в Бейсике

В конце названия функции Years% указан символ %. Таким образом в Бейсике описывается тип возвращаемого функцией значения. Допустимые символы приведены в таблице.

Тип переменной	Символ в конце имени переменной
INTEGER	%
STRING	$
DOUBLE	#

Добавление новой функции

В *QBasic* имеется удобная возможность добавить в программу новую функцию, избежав при этом дополнительного ручного кодирования. Это делает команда Edit ▸ New Function (Правка ▸ Создать функцию). В появившемся диалоговом окне надо ввести название функции Years% и нажать клавишу ENTER. Основной текст программы временно пропадет, и появится автоматически сгенерированное описание новой функции:

```
FUNCTION Years%
END FUNCTION
```

Для того чтобы вернуться обратно к главному тексту, а из него — к любой введенной подпрограмме, необходимо использовать клавишу F2. При ее нажатии на экран выводится список всех созданных подпрограмм, а в первой строке — имя главного модуля.

Функции Years% надо указать список аргументов. В данном случае он будет состоять из одного параметра:

```
FUNCTION Years%(X AS INTEGER)
```

Расчет популяции

Так как для определения нового члена последовательности Фибоначчи требуется знать значения двух предыдущих членов, прежде всего надо описать три локальные переменные F1, F2 и F3, хранящие три очередных значения последовательности. Исходно первые три значения 1, 1 и 2 запишутся в переменные F1, F2 и F3 явно, а в дальнейшем новые значения будут вычисляться программно.

Сам расчет представляет собой условный цикл, который выполняется до тех пор, пока очередное значение не превысит заданное количество особей. Число таких циклов — число лет — будет подсчитываться в локальной переменной-счетчике YearsNum, первоначально имеющей значение 3.

```
FUNCTION Years%(X AS INTEGER)
' описание переменных
```

```
DIM F1 AS INTEGER, F2 AS INTEGER, F3 AS INTEGER
DIM YearsNum AS INTEGER
' задание начальных значений
F1 = 1: F2 = 1: F3 = 2: YearsNum = 3
' цикл, пока число кроликов меньше заданного
DO WHILE F3 < X
' определяем новый член последовательности
  F1 = F2: F2 = F3
  F3 = F1 + F2
' увеличиваем число лет на 1:
  YearsNum = YearsNum + 1
' повторяем цикл
LOOP
' в качестве возвращаемого значения
' используется значение переменной YearsNum
Years% = YearsNum
END FUNCTION
```

Сохранение текста программы в файле

После того как текст программы набран, его желательно сохранить в файле, чтобы потом снова обращаться к нему, улучшать, изменять или просто повторно запускать готовую программу.

Сохранение текста программы в файле осуществляется командой File ▸ Save (Файл ▸ Сохранить), в результате чего на экране показывается диалоговое окно выбора каталога и имени файла. В качестве такого имени можно указать kroliki, выбрать нужный каталог и нажать клавишу ENTER. По умолчанию к названию kroliki припишется расширение .BAS. В дальнейшем эту программу можно снова загрузить в *QBasic* командой File ▸ Open (Файл ▸ Открыть).

Запуск программы

Для запуска программы надо перейти к ее главной части (с помощью клавиши F2) — при этом в самом ее начале автоматически добавится строка с объявлением только что определенной функции:

```
DECLARE FUNCTION Years% (X AS INTEGER)
```

Теперь надо нажать клавишу F5 (Запуск). Программа начинает работать.

Возможный вариант диалога:

```
Введите количество особей: 10
Требуемое число лет: 7
Введите количество особей: 100
```

```
Требуемое число лет: 12
Введите количество особей: 1000
Требуемое число лет: 17
Введите количество особей: 10000
Требуемое число лет: 21
Введите количество особей: 0
```

Первую сотню кроликов надо разводить довольно долго, зато потом их приплод будет увеличиваться стремительными темпами.

20.8. Пример на Паскале. Раскрашивание круга

Постановка задачи

В рабочем окне программы должны находиться: изображение круга, поле ввода с подписью, кнопки Закрасить и Закрыть. В поле ввода шестью символами (шестнадцатеричными цифрами от 0 до F) задается новый цвет круга. Первые два символа определяют интенсивность синего цвета, третий и четвертый — интенсивность зеленого, пятый и шестой — интенсивность красного. Чистый синий опишется строкой ff0000, чистый зеленый — строкой 00ff00, чистый красный — строкой 0000ff, черный — строкой 000000, белый — строкой ffffff и т. д.

В самой программе надо дополнительно проверить, правильно ли введена эта строка. При нажатии на кнопку Закрасить цвет круга должен измениться.

Знакомство с Delphi

Для работы со средой *Delphi* необходимо, чтобы она была предварительно установлена на компьютере.

После ее запуска (например, Пуск ▸ Программы ▸ Borland Delphi 7 ▸ Delphi 7) на экране появятся следующие окна (рис. 20.2).

Главное окно Delphi 7. Здесь расположено основное меню, командные кнопки, а в правой части — *палитра компонентов*. Она состоит из набора панелей, на которых компоненты сгруппированы по решаемым задачам: панель Standard (Стандартная) — стандартные элементы управления, панель Win32 — элементы управления для версии *Windows 9x*, панель Internet (Интернет) — компоненты для организации работы в Интернете и т. д.

Визуальный проектировщик. С его помощью проектируется будущее окно программы. Оно представлено в виде формы, у которой можно менять свойства и размеры. На ней будут располагаться нужные элементы управления.

Редактор исходных тестов. Предназначен для набора и редактирования текстов программы. Ключевые слова и различные идентификаторы выделяются в этом редакторе особыми цветами и разным шрифтом. Принцип его работы аналогичен принципам работы большинства редакторов *Windows*.

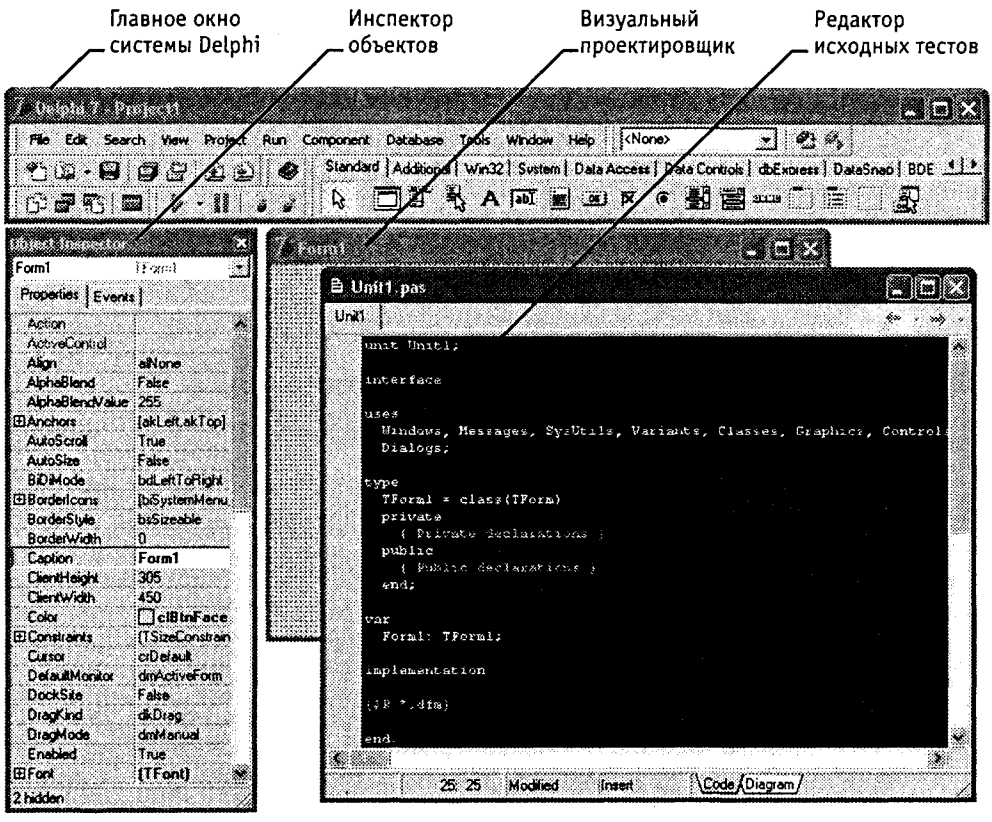


Рис. 20.2. Рабочие окна программы Delphi 7

Инспектор объектов. Используется для визуальной (без программирования) настройки свойств различных объектов на этапе проектирования.

Заголовок окна

Понять, как работает Инспектор объектов, лучше всего на примере. Пока что автоматически создана только одна пустая форма (называющаяся Form1), но она обладает множеством различных свойств. Заголовок формы (будущего окна программы) задается в свойстве Caption (Заголовок). Чтобы его изменить, надо в Инспекторе объектов найти строку, в левой части которой написано Caption, и в правой части этой строки, небольшом поле ввода, указать новое название, например Раскрашивание. Тут же изменится и заголовок формы в визуальном проектировщике.

Аналогично меняются и любые другие свойства. Сначала в выпадающем списке в верхней части Инспектора выбирается нужный объект (или он выделяется на форме щелчком мыши), затем находится название свойства и в правой части его значение меняется на новое. Изменение может происходить как путем простого ввода нового значения с клавиатуры, так и выбором одного из предопределенных значе-

ний из списка. В некоторых случаях для редактирования свойства вызывается специальный редактор.

Свойства могут быть многосоставными, например, свойство Font (Шрифт) — слева от названия такого свойства ставится символ «±». Двойным щелчком мыши на его названии оно раскрывается и показывает все свои вложенные подсвойства.

Размещение компонентов на форме

Прежде всего разместим на форме поле ввода. Для этого на палитре компонентов с помощью закладки выбирается панель Standard (Стандартная) и нажимается кнопка с всплывающей подсказкой Edit (выбран компонент «поле ввода»). Затем надо щелкнуть мышкой на форме, и в месте щелчка появится элемент управления Edit1. Его можно перетаскивать по форме и менять размеры.

В свойстве Text (Содержимое) этого объекта исходно надо задать пустую строку, чтобы при запуске программы в данном поле ничего не показывалось.

Рядом с полем ввода надо разместить свободное поле с комментарием. Для этого на панели компонентов выбирается компонент, называющийся Label (Подпись), и помещается на форме так, как это было сделано с полем ввода. Новый объект автоматически получит название Label1. Чтобы указать в нем текст «Цвет:», его надо ввести в свойство Caption.

Изменить название любого объекта на форме можно, изменив его свойство Name в Инспекторе объектов.

В нижней части формы надо разместить кнопку — компонент Button (Кнопка) на панели Standard (Стандартная). Название этой кнопки (Закрасить) задается в свойстве Caption.

На панели Additional (Дополнительно) имеется компонент Shape (Фигура). Этот компонент помещается в центр формы. Он получит название Shape1 и исходно примет форму квадрата, закрашенного белым цветом. Чтобы превратить его в круг, надо значение свойства Shape изменить на stCircle.

Теперь осталось только добавить кнопку, закрывающую форму. Это действие стандартное, поэтому в *Delphi 7* имеется специальный компонент BitBtn (Кнопка с картинкой) на панели Additional (Дополнительно), позволяющий автоматизировать такие действия, не прибегая к программированию.

После размещения такой кнопки на форме (она получит название BitBtn1) значение ее свойства Kind (Вид выполняемого действия) надо установить в bkClose (Закрыть окно). При этом на кнопке появится изображение стандартной картинки, символизирующей действие закрытия.

В завершение надо изменить заголовок этой кнопки (свойство Caption) с английского слова Close на русское Закрыть, и на этом процесс проектирования приложения можно считать законченным.

Сохранение проекта

Перед тем как приступить к программированию, проект надо сохранить. Это действие выполняется командой File ▸ Save All (Файл ▸ Сохранить все), после чего снача-

ла выбирается каталог и указывается имя файла, в котором хранится программное описание (на Паскале) структуры и работы спроектированной формы. Имя файла будет иметь расширение .PAS по умолчанию. Далее система *Delphi 7* спросит, куда и под каким именем сохранить файл проекта, содержащий всю информацию об используемых формах и модулях (их может быть в одном проекте сколько угодно, но одна форма всегда будет главной) и всевозможные настройки. Название файла проекта не должно совпадать с названием файла с исходным текстом программы.

Обработка нажатия кнопки

В создаваемой программе вручную придется запрограммировать фактически только одно событие — нажатие на кнопку Закрасить. Чтобы создать первоначально пустую подпрограмму, вызываемую при нажатии на эту кнопку, надо просто дважды щелкнуть на ней мышкой. При этом система *Delphi 7* вызовет редактор, автоматически сгенерирует нужный текст и разместит курсор именно в том месте, где можно начать описание нужного алгоритма.

```
procedure TForm1.Button1Click(Sender: TObject);
begin
end;
```

Обработчик события Нажатие на кнопку Button1 — это обычная подпрограмма, метод класса TForm1 (этот класс описывает главную форму Form1). Единственный параметр Sender характеризует источник сообщения о случившемся событии. Его практически всегда можно игнорировать.

Алгоритм работы данного метода будет следующим. Первоначально надо убедиться, что длина введенной в поле Edit1 строки равна 6 символам и каждый из этих символов — шестнадцатеричная цифра. Если это не так, то выполнение обработчика надо сразу завершить (для этого предназначена стандартная процедура Паскаля Exit, мгновенно завершающая работу текущей подпрограммы).

Если же введенные данные корректны, их надо:

1. Преобразовать в промежуточную строку в формате $00xxxxxx, где xxxxxx — шесть введенных цифр.
2. Эту строку преобразовать в число, которое будет рассматриваться как цвет.
3. Установить новый цвет круга на основании полученного значения.

Содержимое поля ввода Edit1 хранится в виде строки в его свойстве Text. Доступ к этому свойству осуществляется с помощью конструкции Edit1.Text.

Длина строки определяется стандартной функцией length() со строкой в качестве параметра.

Стандартная функция Pos(), получая две строки как аргументы, проверяет, не содержится ли первая строка во второй, и если содержится, то возвращает номер начальной позиции. В противном случае Pos() возвращает ноль. Эта функция потребуется для определения, все ли символы во введенной строке допустимы.

Стандартная функция UpperCase() преобразует строку к верхнему регистру. Такое преобразование требуется, чтобы разрешить ввод значений цветов на любых регистрах.

Преобразование строки в число выполняет стандартная функция StrToInt().

Объект Shape1 имеет свойство Brush (Кисть для фона), которое, в свою очередь, имеет вложенное свойство Color (Цвет заливки). Его и надо в конечном счете изменить. Как только это произойдет, цвет круга в окне автоматически изменится на новый.

```
procedure TForm1.Button1Click(Sender: TObject);
var i: integer;
    s: string;
begin
// если длина введенной строки не равна 6,
// то закончить работу
if length(Edit1.Text) <> 6 then exit;
// в локальную переменную s заносится строка,
// содержащая допустимые символы
s := "0123456789ABCDEF";
// проверяется каждый символ во введенной строке
for i := 1 to 6 do
// если очередной символ не найден в строке s, значит,
// он недопустим, и работу требуется прекратить
  if pos(UpperCase(Edit1.Text[i]), s) = 0 then exit;
// все нормально — в переменной s
// готовим промежуточную строку
s := "$00"+Edit1.Text;
// Устанавливаем значение цвета заливки круга равным
// числу, преобразованному из строки в переменной s
Shape1.Brush.Color := StrToInt(s);
end;
```

Запуск программы

Программа запускается нажатием на клавишу F9. Так как *Delphi 7* — это компилирующая система, сначала автоматически выполнится компиляция и только потом программа запустится. Задавая различные строки (FF0FFF, abcdef, 987654 и т. п.), можно наглядно увидеть соответствующие им цвета.

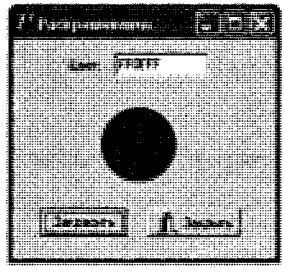


Рис 20.3. Программа закраски в работе

20.9. Пример на Си++. Рисование графиков

Система *Microsoft Visual Studio .NET* во многом схожа с рассмотренной средой *Delphi*, но визуальные возможности построения пользовательского интерфейса реализованы в ней только для Бейсика и C#. Поддержка Си++ в этой системе унаследована от старых версий и не предоставляет разработчику столь удобного проектирования. Тем не менее мы рассмотрим простой пример ее использования для создания программы, рисующей график функции.

Постановка задачи

Некоторая подпрограмма задает зависимость значения функции от аргумента. Надо нарисовать в окне график, показывающий эту зависимость.

Принципы рисования в Visual Studio

Перерисовывать экран в *Windows* приходится по самым разным причинам. Например, окно было закрыто другими приложениями, свернуто или оказалось временно заслоненным своими вспомогательными окнами. При этом перерисовывать приходится или все содержимое, или только часть. Программа, созданная с помощью *Visual Studio*, сама определяет, что и когда ей надо перерисовать, и все элементы управления тоже это «понимают». Особое требование к организации перерисовки возникает, только когда программист напрямую использует функции рисования. Все эти функции в таком случае надо размещать в обработчике события OnDraw, которое вызывается автоматически.

Технология рисования

Система *Microsoft Visual Studio .NET* не содержит визуальных средств создания программ на Си++, аналогичных возможностям *Delphi*. Она предлагает дизайнеры форм только для Бейсика и C#. Поэтому мы изучим несложный вариант использования графических примитивов *Windows* для демонстрации техники рисования графиков в пределах клиентского окна шаблонного приложения. Любое окно *Windows* характеризуется так называемым контекстом устройства, своеобразным объектом, содержащим различные методы графического вывода в пределах этого окна. Доступ к контексту нужного нам окна будет автоматически предоставлен *Microsoft Visual Studio .NET* при формировании обработчика OnDraw.

Для создания графика потребуются два метода этого объекта: метод MoveTo(x,y), устанавливающий новое начальное положение — точку (x, y) для следующих операций рисования, и метод LineTo(x,y), проводящий линию из предыдущей точки в новую.

Метод отрисовки

Так как система *Microsoft Visual Studio .NET* не дает возможности работать с формой напрямую, подготовка «пустого» приложения будет немного сложнее, чем в предыдущих примерах. После запуска системы надо дать команду File ▸ New ▸ Project (Файл ▸ Создать ▸ Проект), на панели Project Types (Типы проектов) выбрать раздел

Visual C++ Projects (Проекты Visual C++), а на панели Templates (Шаблоны) — значок MFC Application (Оконное приложение).

Название проекта (например, Grafiki) и его местонахождение задается в полях Name (Имя) и Location (Расположение). После нажатия на кнопку OK запускается Мастер настройки вида будущего приложения. Изменять в нем ничего не надо, достаточно нажать кнопку Finish (Готово). Система сгенерирует заготовку пустого, но работоспособного проекта. Посмотреть его структуру можно с помощью средства Просмотра решения (*Solution Explorer*), вызываемого командой View ▸ Solution Explorer (Вид ▸ Просмотр решения).

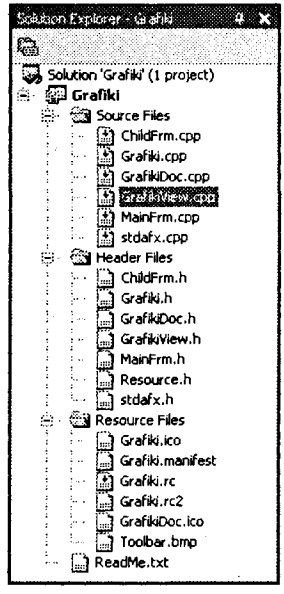


Рис. 20.4. Структура проекта Visual Studio .NET, сформированная автоматически

В этом множестве автоматически сгенерированных файлов нас интересует файл GrafikiView.cpp, непосредственно ответственный за отображение содержимого клиентских окон на экране. Чтобы открыть его в редакторе, надо дважды щелкнуть мышкой на соответствующей строке.

Алгоритм отображения графика несложен. Он умещается в нескольких операторах.

Координату по оси Y нельзя взять непосредственно из переменной y, а надо вычислять по формуле Height–y, потому что в системе *Windows* считается, что точка с координатами (0,0) расположена в верхнем левом углу окна, а ось Y направлена вниз. Для удобства восприятия эту ось надо перевернуть.

Необходимо сформировать обработчик события OnDraw. Для этого перейдем в раскрывающийся список в верхнем правом углу окна редактора исходных текстов, и найдем в нем строку OnDraw. После ее выбора в редакторе появится следующий новый текст:

```
// CGrafikiView drawing

void CGrafikiView::OnDraw(CDC* /*pDC*/)
{
  CGrafikiDoc* pDoc = GetDocument();
  ASSERT_VALID(pDoc);

  // TODO: add draw code for native data here
}
```

Параметр pDC — это нужный нам указатель на контекст устройства. Его мы и будем использовать для вывода графиков. Обратите внимание, что по умолчанию он взят в скобки-комментарии и недоступен внутри функции. Поэтому его надо раскомментировать.

```
void CGrafikiView::OnDraw(CDC* pDC)
{
  CGrafikiDoc* pDoc = GetDocument();
  ASSERT_VALID(pDoc);

  // TODO: add draw code for native data here

// начальные координаты
  int x,y;
  x = 0;
  y = 0;

// переменная-прямоугольник
  CRect rect;

// определяем размеры клиентского окна
// вызовом стандартной функции Windows
    GetClientRect(&rect);

// фиксируем начальную точку
  pDC->MoveTo(0,rect.Height());

// цикл, пока каждая координата очередной точки
// укладывается в клиентскую область
  while( x < rect.Width() && y < rect.Height() )
  {
  x = x + 1;
// соответствующее значение по оси Y
  y = f(x);

// в новую точку (x,rect.Height()-y)
// рисуется линия
  pDC->LineTo(x,rect.Height()-y);
  }
```

Чуть выше метода OnDraw надо определить функцию f(), не привязанную ни к какому классу. В ней происходит вычисление значения анализируемой математической функции по заданному аргументу. Для примера, она может выглядеть так:

```
int f(int x)
{
int y;
y = 50*log(x);
return y;
}
```

Стандартная функция log() вычисляет значение логарифма. Коэффициент 50 нужен, чтобы кривая пропорционально размещалась в окне. Хотя функция log() рассчитывает действительное значение, компилятор автоматически настроит программный код так, чтобы оно было преобразовано в целый тип, соответствующий типу переменной y.

Исходно функция log() и ряд других не подключены к текущему проекту. Чтобы они стали доступными, библиотеку, в которой они хранятся, необходимо явно указать компилятору. Делается это с помощью командной строки

```
#include "Math.h"
```

которую можно поместить в самое начало текущего файла.

Далее проект надо сохранить, выполнить компиляцию и запустить, нажав клавишу F5. В дальнейшем, изменив один оператор присваивания в функции f() и подобрав подходящие коэффициенты, с помощью этой программы можно строить самые разные графики.

Практические задания по программированию

Задание 1

Дано натуральное число. Составить программу, которая представляет данное число в виде суммы квадратов натуральных чисел, содержащей минимальное число слагаемых. Например:

$9=3^2$

$12=2^2+2^2+2^2$

$23=3^2+3^2+2^2+1^2$

Задание 2

Дан массив, содержащий N элементов.

Написать подпрограммы, выполняющие следующие действия:

- перестановку элементов массива в обратном порядке;
- вычисление суммы A[1] + A[2]*A[2] + A[3]*A[3]*A[3] ...;

- определение элементов массива, разность модулей которых имеет наибольшее значение;
- определение значения, которое встречается среди элементов массива максимальное число раз, и вычисление количества таких вхождений;
- упорядочение элементов массива по возрастанию.

Задание 3

Дан двумерный массив, содержащий $N \times N$ элементов.

Написать подпрограммы, выполняющие следующие действия:

- вычисление среднего арифметического для элементов каждой строки массива;
- замену нулями всех элементов, расположенных на главной диагонали матрицы;
- определение наибольшего элемента и его положения в массиве.

Задание 4

Дана текстовая строка.

Написать подпрограммы, выполняющие следующие действия:

- подсчет количества слов в строке (в качестве границ слов рассматриваются пробелы);
- подсчет количества цифр в строке;
- определение десятичного числа, которому соответствует строка, если она представляет запись этого числа в шестнадцатеричной системе;
- проверку соответствия содержимого строки правилам записи идентификаторов языков программирования.