ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ
ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ
им. Р.Е. Алексеева

С.Н. Митяков, И.В. Лапшин, Е.Ф. Листопад

ИНФОРМАТИКА

КОМПЛЕКС УЧЕБНО-МЕТОДИЧЕСКИХ МАТЕРИАЛОВ

Часть 1

Рекомендовано Ученым советом Нижегородского государственного технического университета в качестве учебно-методического пособия для студентов заочной и дистанционной форм обучения всех специальностей.

Нижний Новгород, 2006

УДК 651.3.06

С.Н. Митяков, И.В. Лапшин, Е.Ф. Листопад. Информатика: комплекс учебно-методических материалов / С.Н. Митяков, И.В. Лапшин, Е.Ф. Листопад; Нижегород. гос. техн. ун-т. Нижний Новгород, 2006. - 70 с.

Изложены основы компьютерной грамотности: архитектура и программное обеспечение ПЭВМ, принципы работы в операционных системах *MS-DOS* и *Windows*, элементы алгоритмизации и алгоритмического языка Фортран. Приведены варианты контрольных заданий, контрольные вопросы для подготовки к зачету. Разработка предназначена для студентов всех специальностей заочной и дистанционной форм обучения.

Рецензент О.Р. Козырев, зав.кафедрой «Информационные системы и технологии» НФ ГУ ВШЭ, профессор

Редактор Э.А. Жирнова

Компьютерная верстка И.В.Лапшин

Подписано в печать 29.03.2006. Формат 60x841/16. Бумага офсетная. Печать офсетная. Печ. л. 4,5. Уч.–изд. л. 4,0. Тираж 300 экз. Заказ

Нижегородский государственный технический университет.
Типография НГТУ
Адрес университета и полиграфического предприятия: 603600, Н. Новгород, ул. Минина, 24.

© Нижегородский государственный
технический университет, 2006

© Митяков С.Н., Лапшин И.В.,
Листопад Е.Ф., 2006

ОГЛАВЛЕНИЕ

1. ПОЯСНИТЕЛЬНАЯ ЗАПИСКА 4
2. РАБОЧАЯ УЧЕБНАЯ ПРОГРАММА ДИСЦИПЛИНЫ «ИНФОРМАТИКА» 5
2.1 Тематический план дисциплины 5
2.2. Описание содержания основных тем 6
3. ОПОРНЫЙ КОНСПЕКТ ЛЕКЦИЙ 8
3.1. Понятия информации, данных и информационных процессов 8
3.2. Устройство и принципы работы персонального компьютера 12
3.3. Программное обеспечение персонального компьютера 23
3.4. Программирование задач на простые переменные 38
3.5.Программирование задач на одномерные и двумерные массивы. Введение 51
4. ОПИСАНИЕ ЛАБОРАТОРНЫХ РАБОТ 63
Порядок выполнения лабораторной работы № 1 63
Порядок выполнения лабораторной работы № 2 63
Порядок выполнения лабораторной работы № 3 64
5. ЗАДАНИЯ И ВАРИАНТЫ ДЛЯ КОНТРОЛЬНЫХ И ЛАБОРАТОРНЫХ РАБОТ 64
Задание №1. 67
Задание №2 68
Задание №3 70
Задание №4 71
6. МЕТОДИЧЕСКИЕ УКАЗАНИЯ И ПРИМЕРЫ ВЫПОЛНЕНИЯ КОНТРОЛЬНЫХ РАБОТ 73
Задание №1 73
Задание №2 73
Задание №3 75
Задание №4 76
7. КОНТРОЛЬ ЗНАНИЙ СТУДЕНТОВ 78
Таблица вариантов заданий 78
Образец таблицы для оформления результатов решения тестов 81
Задание №1. Архитектура ПЭВМ. Программное обеспечение 81
Задание №2. Программа-оболочка Norton Commander 82
Задание №3. Операционная система Windows 83
Контрольные вопросы 84
8. ГЛОССАРИЙ 87
9. СПИСОК ЛИТЕРАТУРЫ 90

1. ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Рабочая программа сформирована на основе Государственных образовательных стандартов высшего профессионального обучения по курсу «Информатика» для соответствующих направлений подготовки дипломированных специалистов. Основной целью базовой дисциплины «Информатика» является освоение студентами современных методов и средств использования компьютеров в учебном процессе и дальнейшей инженерно-технической деятельности.

Основная задача обучения - не только обеспечить приобретение знаний, умений и навыков в соответствии с государственными образовательными стандартами, но и содействовать фундаментализации образования, формированию мировоззрения и развитию системного мышления студентов.

Изучение информатики по заочной форме обучения проводится в течение двух семестров. Данное пособие содержит необходимые материалы для изучения первой части курса (первый семестр).

Эта часть курса посвящена основам информатики и технологий работы с персональным компьютером: понятие информации, архитектура ПЭВМ и программное обеспечение, основы алгоритмизации и программирования, языки высокого уровня. В течение семестра студенты оформляют в тетрадях контрольные работы и тесты. Контрольные работы проводятся по вариантам, приведенным в разделе 5 данного пособия. Они включают в себя 4 задания по основам алгоритмизации. Эти же задания выполняются в конце семестра (перед зачетом) на ЭВМ в ходе соответствующих лабораторных работ. Тесты оформляются по вариантам, приведенным в разделе 7 данного пособия.

Особенностью заочной формы обучения является небольшое количество аудиторной нагрузки. Это компенсируется наличием в каждом семестре аудиторных консультаций, проходящих в компьютерном зале.

По окончании семестра проводится зачет в письменной форме.

2. РАБОЧАЯ УЧЕБНАЯ ПРОГРАММА ДИСЦИПЛИНЫ «ИНФОРМАТИКА»

2.1 Тематический план дисциплины

№ п.п	Наименование	Заочная форма обучения				
		Всего часов	Ауд.	Лекции	Лаб. работы	Самост. работа
1 семестр						
1	Понятия информации, данных и информационных процессов	13	1	1	-	12
2	Устройство и принципы работы персонального компьютера	18	4	3	1	14
3	Программное обеспечение персонального компьютера	17	3	2	1	14
4	Основы алгоритмизации и программирования. Языки программирования высокого уровня	19	5	3	2	14
5	Программирование задач на простые переменные	19	5	3	2	14
6	Программирование задач на одномерные и двумерные массивы	20	6	4	2	14
	Итого за 1 семестр	**106**	**24**	**16**	**8**	**82**
Форма контроля знаний студента: 1-й семестр – зачет.						

2.2. Описание содержания основных тем

1. **Понятия информации, данных и информационных процессов**.
   Общая характеристика процессов сбора, передачи, накопления и обработки информации. Виды информации. История развития информатики.
2. **Устройство и принципы работы персонального компьютера.**
   Принцип работы компьютера. Классификация компьютеров. Основные функциональные части компьютера. Назначение и основные принципы работы устройств, входящих в ПЭВМ. Функциональные блоки компьютера и их назначение. Овладение навыками работы с клавиатурой, мышью, монитором, принтером. Требования к организации студенческого каталога.
3. **Программное обеспечение персонального компьютера.**
   Классификация программного обеспечения. Базовое, системное, служебное и прикладное программное обеспечение. Функции операционной системы. Файлы и их имена. Распределение файлов по диску. Понятие графического интерфейса. Операционная система *Windows*. Организация интерфейса. Рабочий стол, объекты и элементы управления. Панель управления, главное меню, значки. Работа с окнами. Работа с проводником. Создание, открытие, закрытие папок. Работа со стандартными приложениями. Сохранение файлов. Обмен информацией между приложениями через Буфер обмена. Работа в операционной оболочке *Far Manager*. Классификация прикладного ПО. Основные возможности текстового процессора *Word*.
4. **Основы алгоритмизации и программирования. Языки программирования высокого уровня.**
   Понятие алгоритма и алгоритмической системы. Две формы представления алгоритмов: визуальная и текстовая. Блок-схемы. Компьютер как исполнитель алгоритмов. Типовые алгоритмы. Линейные и разветвляющиеся алгоритмы. Программа как изображение алгоритма в терминах команд, управляющих работой компьютера. Основные элементы алгоритмического языка высокого уровня: константы, переменные, знаки арифметических операций,

операции отношения, стандартные функции, типы данных, операторы, структуры. Знакомство с оболочкой базового языка программирования. Примеры решения простейших задач линейной и разветвляющейся структур.

5. **Программирование задач на простые переменные.**

   Многоразветвляющийся, одномерный и многомерный циклические алгоритмы. Структура, параметры и принципы организации циклов. Построение алгоритма из базовых структур. Решение задач на простые переменные со сложной логикой (многомерные циклы, многоразветвляющиеся алгоритмы и т.д.). Отладка программы.

6. **Программирование задач на одномерные и двумерные массивы.**

   Индексированные переменные и массивы. Ввод и вывод массивов. Суммирование, нахождение произведения и количества элементов массива. Перестановка элементов массива. Нахождение максимального (минимального) элемента массива. Форматный вывод данных. Комбинированные задачи на одномерные и двумерные массивы. Формирование новых массивов. Работа с квадратными матрицами.

3. ОПОРНЫЙ КОНСПЕКТ ЛЕКЦИЙ

3.1. Понятия информации, данных и информационных процессов

В широком смысле слова информатика – это область знаний, изучающая как саму информацию, так и вопросы, связанные с ее сбором, обработкой и хранением. Источники передачи информации весьма многочисленны: начиная от обычной почты, газет, телевидения и кончая локальными и глобальными компьютерными сетями. В более узком смысле информатика – это дисциплина, связанная с изучением основных приемов работы на компьютере. Именно в этом смысле мы и будем рассматривать это понятие в нашем курсе.

Информатика как наука наиболее динамично стала развиваться с середины XX-го века с появлением первого алгоритмического языка Фортран. В это время уже существовали первые электронные вычислительные машины (ЭВМ). Компьютеры 40-50-х годов были очень большими устройствами - огромные залы были заставлены шкафами с электронным оборудованием. Все это стоило очень дорого, поэтому компьютеры были доступны только крупным компаниям и учреждениям. Первый шаг к уменьшению размеров компьютеров стал возможен с изобретением в 1948 г. транзисторов - миниатюрных электронных приборов, которые смогли заменить в компьютерах электронные лампы. К середине 60-х годов был подготовлен еще один шаг к миниатюризации компьютеров - были изобретены интегральные схемы. К середине 70-х годов все компьютеры можно было подразделить на три большие класса.

Большие ЭВМ. Представитель – БЭСМ-6. Занимали площадь не менее 200 кв.м. и требовали для обслуживания несколько десятков инженеров, программистов и операторов. Основной режим работы – пакетный. Одновременно (в пакете) обрабатывались десятки и даже сотни задач. Носитель информации – перфокарта. Класс решаемых задач – сложные вычислительные алгоритмы. Языки программирования - Фортран, Алгол.

Малые ЭВМ. Представитель – СМ-4. Занимаемая площадь – 10 – 20 кв.м. Обслуживание – 1-3 чел. Появляется диалоговый режим работы, при котором

пользователь может непосредственно общаться с компьютером, используя дисплей и клавиатуру. Носитель информации – перфолента. Появляются и магнитные носители – цифровые магнитофоны. Кроме вычислительных, появляются новые классы задач и систем: АСНИ (автоматизированные системы научных исследований), АСУТП (автоматизированные системы управления технологическими процессами) и САПР (системы автоматизированного проектирования).

Микро-ЭВМ. Представитель – Электроника-60. Компьютер располагается в небольшом блоке, он легкий, мобильный. Новый импульс в поддержку АСНИ – компьютер можно вывозить в научные экспедиции. Новое качество АСУТП – непосредственная установка микро-ЭВМ или микропроцессора на производстве для управления станками (станки с ЧПУ).

Бурное развитие электронной технологии привело к созданию первых микропроцессоров. Лидирующие позиции в выпуске микропроцессоров занимала фирма *Intel*. Первый микропроцессор *Intel* - 4004 был выпущен в продажу в конце 1970 г. В 1973 г. фирма *Intel* выпустила 8-битный микропроцессор 8008, а в 1974 г. - его усовершенствованную версию *Intel*- 8080, которая до конца 70-х годов стала стандартом для микрокомпьютерной индустрии. В 1974 г. несколько фирм объявили о создании на основе микропроцессора Intel - 8008 компьютера, т.е. устройства, выполняющего те же функции, что и большая ЭВМ. В результате оказалось, что для многих организаций необходимые им расчеты стало возможно выполнять не на больших ЭВМ или мини-ЭВМ, а на персональных компьютерах, что значительно дешевле.

Распространение персональных компьютеров к концу 70-х годов привело к некоторому снижению спроса на большие ЭВМ и мини- ЭВМ. Это стало предметом серьезного беспокойства фирмы *IBM* - ведущей компании по производству больших ЭВМ, и в 1979 г, фирма *IBM* решила попробовать свои силы на рынке персональных компьютеров. Прежде всего в качестве основного микропроцессора компьютера был выбран новейший тогда 16-разрядный микропроцессор *Intel* - 8088. Его использование позволило значительно увеличить потенциальные возможности компьютера, так как новый микропроцессор по-

зволял работать с 1 Мбайтом памяти, а все имевшиеся тогда компьютеры были ограничены 64 Кбайтами. В компьютере были использованы и другие комплектующие различных фирм, а его программное обеспечение было поручено разработать небольшой фирме *Microsoft.*

В августе 1981г. новый компьютер под названием *IBM PC* (читается - Ай-Би-Эм Пи-Си) был официально представлен публике, и вскоре после этого он приобрел большую популярность у пользователей. Через один-два года компьютер *IBM PC* занял ведущее место на рынке, вытеснив модели 8-битных компьютеров. Фактически *IBM PC* стал стандартом персонального компьютера. Сейчас компьютеры, совместимые с *IBM PC*, составляют около 90% всех производимых в мире персональных компьютеров.

Основные особенности персональных ЭВМ:

- небольшие размеры (компьютер умещается на рабочем столе и предназначен, главным образом, для одного пользователя, отсюда название – персональный);
- наличие магнитных носителей, дисплея и клавиатуры;
- высокая надежность и жесткая стандартизация основных узлов;
- высокий уровень сервиса, хороший дизайн.

Развитие компьютеров типа *IBM PC* теперь осуществляется многими конкурирующими фирмами, хотя *IBM* и остается самым крупным производителем этих компьютеров. Компьютеры на основе микропроцессоров *1nte1* - 80386, 80486 и *Pentium*, мониторы типа *SuperVGA* 800x600 и 1024x768 были разработаны уже не *IBM*, а различными другими фирмами. Наибольшее влияние на развитие компьютеров типа *IBM PC* теперь оказывает не *IBM*, а фирма *Intel* - производитель микропроцессоров, являющихся мозгом *IBM PC*, и фирма *Microsoft* - разработчик операционной системы *MS-DOS*, графической операционной оболочки *Windows* и многих других используемых *IBM PC* программ. В чем же причины такого стремительного роста индустрии персональных компьютеров?

Основные причины:

- невысокая стоимость компьютеров (как правило, от нескольких сотен до нескольких тысяч долларов);
- их сравнительная выгодность для многих деловых применений по сравнению с большими ЭВМ и мини-ЭВМ;
- простота использования, обеспеченная с помощью диалогового способа взаимодействия с компьютером, удобных и понятных интерфейсов программ (меню, подсказки, помощь и т.д.);
- возможность индивидуального взаимодействия с компьютером без каких-либо посредников и ограничений;
- относительно высокие возможности по переработке информации (типичная скорость - десятки миллионов операций в секунду, емкость оперативной памяти - десятки и сотни Мбайт, емкость жестких дисков – сотни Гбайт);
- высокая надежность и простота ремонта, основанные на интеграции компонентов компьютера;
- возможность расширения и адаптации к особенностям применения компьютеров,
- один и тот же компьютер может быть оснащен различными периферийными устройствами и разным программным обеспечением;
- наличие программного обеспечения, охватывающего практически все сферы человеческой деятельности, а также мощных систем для разработки нового программного обеспечения.

Персональные компьютеры являются наиболее широко используемым видом компьютеров, их мощность постоянно увеличивается, а область применения расширяется. Персональные компьютеры могут объединяться в сети, что позволяет десяткам и сотням пользователей легко обмениваться информацией и одновременно получать доступ к общим базам данных. Средства электронной почты позволяют пользователям компьютеров с помощью обычной телефонной

сети посылать текстовые и факсимильные сообщения в другие города и страны и получать информацию из крупных банков данных.

3.2. Устройство и принципы работы персонального компьютера

Прежде всего, компьютер должен иметь следующие устройства:

- арифметико-логическое устройство, выполняющее арифметические и логические операции;
- устройство управления, которое организует процесс выполнения программ;
- запоминающее устройство или память для хранения программ и данных; внешние устройства для ввода-вывода информации.

В общих чертах работу компьютера можно описать так. В начале с помощью какого-либо внешнего устройства в память компьютера вводится программа. Устройство управления считывает содержимое ячейки памяти, где находится первая инструкция (команда) программы, и организует ее выполнение. Эта команда может задавать выполнение арифметических или логических операций, чтение из памяти данных для выполнения арифметических или логических операций или запись их результатов в память, ввод данных из внешнего устройства в память или вывод данных из памяти на внешнее устройство.

Как правило, после выполнения одной команды устройство управления начинает выполнять команду из ячейки памяти, которая находится непосредственно за только что выполненной командой. Однако этот порядок может быть изменен с помощью команд передачи управления (перехода). Эти команды указывают устройству управления, что ему следует продолжить выполнение программы, начиная с команды, содержащейся в некоторой другой ячейке памяти. Такой скачок или переход в программе может выполняться не всегда, а только при выполнении некоторых условий, например, если некоторые числа равны, если в результате предыдущей арифметической операции получился нуль и т.д. Это позволяет использовать одни и те же последовательности команд в программе много раз (т.е. организовывать циклы), выполнять различные последо-

вательности команд в зависимости от выполнения определенных условий и т.д., т.е. создавать сложные программы. Таким образом, управляющее устройство выполняет инструкции программы автоматически, т.е. без вмешательства человека. Оно может обмениваться информацией с оперативной памятью и внешними устройствами компьютера. Поскольку внешние устройства, как правило, работают значительно медленнее, чем остальные части компьютера, управляющее устройство может приостанавливать выполнение программы до завершения операции ввода-вывода с внешним устройством. Все результаты выполненной программы должны быть ею выведены на внешние устройства компьютера, после чего компьютер переходит к ожиданию каких-либо сигналов внешних устройств.

Следует заметить, что схема устройства современных компьютеров несколько отличается от приведенной выше. В частности, арифметико-логическое устройство и устройство управления, как правило, объединены в единое устройство - центральный процессор. Кроме того, процесс выполнения программ может прерываться для выполнения неотложных действий, связанных с поступившими сигналами от внешних устройств компьютера - прерываний. Многие быстродействующие компьютеры осуществляют параллельную обработку данных на нескольких процессорах. Тем не менее, большинство современных компьютеров в основных чертах соответствуют принципам, изложенным фон Нейманом.

Компьютер может обрабатывать только информацию, представленную в цифровой форме. Вся другая информация (например, звуки, изображения, показания приборов и т.д.) для обработки на компьютере должна быть преобразована в цифровую форму. Например, чтобы перевести в цифровую форму музыкальный звук, можно через небольшие промежутки времени измерять интенсивность звука на определенных частотах, представляя результаты каждого измерения в числовой форме. С помощью программ для компьютера можно выполнить преобразования полученной информации, например, наложить друг на

друга звуки от разных источников. После этого результат можно преобразовать обратно в звуковую форму.

Аналогичным образом на компьютере можно обрабатывать и текстовую информацию. При вводе в компьютер каждая буква кодируется определенным числом, а при выводе на внешние устройства (экран или печать) для восприятия человеком по этим числам строятся соответствующие изображения букв. Соответствие между набором букв и числами называется кодировкой символов.

Как правило, все числа в компьютере представляются с помощью нулей и единиц (а не десяти цифр, как это привычно для людей). Иными словами, компьютеры обычно работают в двоичной системе счисления, поскольку при этом их устройство получается значительно более простым. Ввод чисел в компьютер и вывод их для чтения человеком может осуществляться в привычной десятичной форме - все необходимые преобразования могут выполнить программы, работающие на компьютере.

Единицей информации в компьютере является бит, т.е. двоичный разряд, который может принимать значение 0 или 1. Как правило, команды компьютеров работают не с отдельными битами, а с восемью битами сразу. Восемь последовательных битов составляют байт. В одном байте можно закодировать значение одного символа из 256 возможных ($256 = 2^{8}$). Более крупными единицами информации являются килобайт (сокращенно обозначаемый - кбайт), равный 1024 байтам ($1024=2^{10}$), мегабайт (сокращенно обозначаемый Мбайт), равный 1024 кбайтам или 2^{20} байтам и гигабайт (сокращенно обозначаемый Гбайт), равный 1024 Мбайтам или 2^{30} байтам.

Если бы *IBM PC* был сделан так же, как другие существовавшие во время его появления компьютеры, он бы устарел через два-три года, и мы давно бы уже о нем забыли. К счастью, в *IBM PC* была заложена возможность усовершенствования его отдельных частей и использования новых устройств. Фирма *IBM* сделала компьютер не единым неразъемным устройством, а обеспечила возможность его сборки из независимо изготовленных частей аналогично детскому конструктору. При этом методы сопряжения устройств с компью-

тером *IBM PC* не только не держались в секрете, но и были доступны всем желающим. Этот принцип, называемый принципом открытой архитектуры, наряду с другими достоинствами, обеспечил потрясающий успех компьютеру *IBM PC*.

Как же устроен этот конструктор? На основной электронной плате компьютера *IBM PC* (системной, или материнской, плате) размещены только те блоки, которые осуществляют обработку информации (вычисления). Схемы, управляющие всеми остальными устройствами компьютера - монитором, дисками, принтером и т.д., реализованы на отдельных платах, которые вставляются в стандартные разъемы на системной плате - слоты. К этим электронным схемам подводится электропитание из единого блока питания, а для удобства и надежности все это заключается в общий металлический или пластмассовый корпус - системный блок.

По-видимому, фирма *IBM* рассчитывала, что открытость архитектуры *IBM PC* позволит независимым производителям разрабатывать различные дополнительные устройства, что увеличит популярность компьютера. Так оно и произошло, и через один-два года на рынке предлагались сотни разных устройств и комплектующих для *IBM PC*.

Наибольшую выгоду от открытости архитектуры *IBM PC* получили, естественно, пользователи. Они могли самостоятельно расширять возможности своих компьютеров, покупая соответствующие устройства и подсоединяя их в свободные разъемы на системной плате. При этом они не были связаны ассортиментом моделей, предлагаемых фирмой *IBM*, так как могли покупать дополнительные устройства, производимые независимыми фирмами. Они могли сэкономить деньги, ориентируясь при покупке компьютеров на свои сегодняшние, а не будущие потребности - ведь при необходимости компьютер можно модернизировать.

Развитие микрокомпьютеров и увеличение их вычислительной мощности сопровождалось одновременным уменьшением их стоимости. Появляется по-

требность в связанных друг с другом компьютерных системах. В производственной сфере зарождается концепция локальных вычислительных сетей (ЛВС).

В 90-х годах мини-компьютерные системы не вытеснили большие ЭВМ, а дополнили их. ЛВС также функционально дополняют большие ЭВМ и многопользовательские системы на основе мини-ЭВМ. Имеющиеся ЭВМ взяли на себя роль главного компьютера в локальных вычислительных сетях - сервера. Мини- и микрокомпьютеры предоставили в распоряжение пользователей мощные накопители, периферийное оборудование и возможность взаимодействия друг с другом. Стало возможным использование преимущества централизованного хранения данных и обеспечения их защиты.

Рассмотрим структурную схему персонального компьютера (рис. 1).

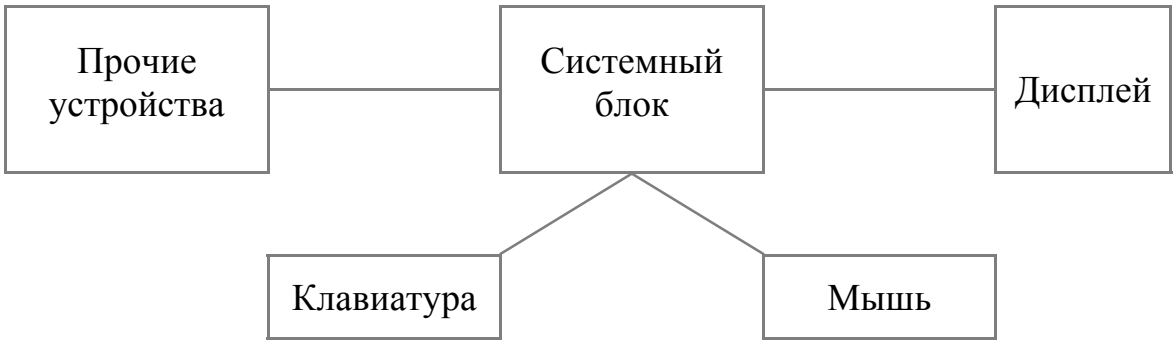


Рис. 1. Структурная схема ПЭВМ

Рассмотрим основные составные части ПК.

1. ***Системный блок***, включающий:

- материнскую плату (основной микропроцессор, оперативная память, порты ввода-вывода);
- жесткий диск (винчестер);
- гибкий диск;
- контроллеры внешних устройств;
- лазерный диск (*CD-ROM*) – не обязателен.

Самым главным элементом в компьютере, его мозгом, является *микропроцессор* - небольшая (в несколько сантиметров) электронная схема, выпол-

няющая все вычисления и обработку информации. Микропроцессор умеет производить сотни различных операций и делает это со скоростью в несколько десятков или даже сотен миллионов операций в секунду. В компьютерах типа *IBM PC* используются микропроцессоры фирмы *Intel*, а также совместимые с ними микропроцессоры других фирм.

Следующим очень важным элементом компьютера является *оперативная память*. Именно из нее процессор берет программы и исходные данные для обработки, в нее они записывают полученные результаты. Название «оперативная» эта память получила потому, что она работает очень быстро, так что процессору не приходится ждать при чтении данных из памяти или записи в память. Однако содержащиеся в ней данные сохраняются, только пока компьютер включен, при выключении компьютера содержимое оперативной памяти стирается. Объем оперативной памяти в среднем составляет 256 Мбайт- 1 Гбайт.

Порты ввода-вывода служат для связи ПК с внешними устройствами. Чтобы компьютер мог работать, необходимо, чтобы в его оперативной памяти находились программы и данные. А попадают они туда из различных устройств компьютера: клавиатуры, дисководов для магнитных дисков и т.д. Результаты выполнения программ также выводятся на внешние устройства - монитор, диски, принтер и т.д. Таким образом, для работы компьютера необходим обмен информацией между оперативной памятью и внешними устройствами. Такой обмен называется вводом-выводом. Простейшая организация такого обмена происходит с помощью портов. Порты бывают двух типов – последовательный и параллельный.

Накопители на жестком диске (винчестеры) предназначены для постоянного хранения информации, используемой при работе с компьютером: программ операционной системы, часто используемых пакетов программ, редакторов документов, трансляторов с языков программирования и т.д. Емкость жесткого диска 40 – 200 Гбайт.

Гибкие диски (дискеты) позволяют переносить документы и программы с одного компьютера на другой, хранить информацию, не используемую постоянно на компьютере, делать архивные копии информации, содержащейся на жестком диске. Емкость гибкого диска 1,4Мбайт.

Контроллеры внешних устройств предназначены для управления работой различных устройств, подключаемых к компьютеру. Для каждого внешнего устройства в компьютере имеется электронная схема, которая им управляет. Эта схема называется контроллером или адаптером. Некоторые контроллеры (например, контроллер дисков) могут управлять сразу несколькими устройствами. Все контроллеры и адаптеры взаимодействуют с микропроцессором и оперативной памятью через системную магистраль передачи данных, которую в просторечии обычно называют шиной. Контроллеры могут конструктивно располагаться на отдельных электронных платах, либо входить в состав материнской платы. Основные из них:

- *контроллер дисплея* (видео-карта) – обязательное устройство, используемое для управления работой дисплея;
- *звуковая карта* – используется для подключения к компьютеру устройств мультимедиа;
- *сетевая карта* – устройство для подключения компьютера в локальную вычислительную сеть.

Модем – устройство для подключения компьютера в глобальную вычислительную сеть обычно через телефонный канал; модемы бывают встроенными (вставляемыми в системный блок компьютера) и внешними (подключаемыми через коммуникационный порт). Функции модема следуют из его названия - МОдулятор-ДЕМодулятор: при передаче он переводит цифровой сигнал компьютера в аналоговый, а при приеме, наоборот, - аналоговый телефонный сигнал в воспринимаемую компьютером последовательность цифр.

Факс-модем - устройство, сочетающее возможности модема и средства для обмена факсимильными изображениями с другими факс-модемами и обычными телефаксными аппаратами.

Лазерный диск (CD-ROM) используется главным образом для установки программного обеспечения и имеет емкость около 700Мбайт. Кроме считывающих, существуют записываемые и перезаписывыемые (рассчитанные на много записей) лазерные диски. Основная их функция – хранение архивов программ и данных.

2. Дисплей (монитор) предназначен для отображения информации в процессе работы ПК в символьном и графическом виде. Принцип действия мониторов на базе электронно-лучевой трубки (ЭЛТ) мало чем отличается от принципа действия обычного телевизора и заключается в том, что испускаемый катодом (электронной пушкой) пучок электронов, попадая на экран, покрытый люминофором, вызывает его свечение. На пути пучка электронов обычно находятся дополнительные электроды: модулятор, регулирующий интенсивность пучка электронов и связанную с ним яркость изображения, и отклоняющая система, позволяющая изменять направление пучка.

При работе в символьном режиме экран разделен на отдельные прямоугольные поля (знакоместа), в которые с помощью набора сегмента отображаются в соответствии с кодом определенные символы. Число символов в строке – 80, число строк на экране – 25. При работе в графическом режиме изображение формируется с помощью маленьких точек – пикселей. Каждая такая точка может иметь свой цвет и свою яркость. Палитра современных мониторов содержит не менее 256 цветов. Чем больше точек на экране (разрешение определяется размером экрана и качеством видео-карты), тем больше качество изображения и меньше вреда для здоровья человека. Типичное разрешение современных дисплеев составляет 1024 точки по горизонтали и 768 по вертикали. Немаловажное значение имеет и размер экрана. Самый маленький дисплей имеет размер экрана по диагонали 14 дюймов. Дальнейший ряд – 15, 17, 19, 21 дюйм и т.д.

3. Клавиатура - устройство для ввода данных, представленных символами (цифры, буквы, некоторые специальные знаки) в ПК. Главное свойство клавиатуры - ее стандартная форма. Это означает, что независимо от выпускающей

фирмы клавиатура содержит стандартный набор символов (клавиш), каждая из которых находится на строго определенном месте. Стандартная клавиатура содержит 4 группы клавиш. Первая – основная группа занимает большую часть клавиатуры. Она содержит буквы (для России, кроме латинского алфавита, представлена кириллица), цифры, специальные символы, а также некоторые управляющие клавиши. Вторая группа – функциональные клавиши (*F1-F12*) расположена в верхней части клавиатуры. Эти клавиши используются для обозначения определенных команд. Третья и четвертая группы частично дублируют друг друга и расположены в правой части клавиатуры.

4. Мышь – манипулятор, который предназначен для управления работой с графической информацией. Перемещение мыши вызывает поворот резинового шарика и соответствующее смещение курсора на экране дисплея. Мышь содержит две клавиши. Левая обычно является наиболее активной. При необходимости – для левши – можно перенастроить мышь, сделав основной правую кнопку.

5. Прочие устройства – устройства, не входящие в обязательную конфигурацию персонального компьютера. Рассмотрим наиболее важные из них.

Принтер – устройство для вывода на бумажный носитель текстовой и графической информации. В настоящее время в основном применяются принтеры следующих типов: матричные игольчатые принтеры; струйные принтеры; лазерные и светодиодные принтеры.

Современные матричные принтеры используют печатающую головку с 9 или 24 иглами, управляемыми при помощи электромагнитов. Быстродействие последних и количество печатающих игл в основном определяют скорость печати. Печать осуществляется при горизонтальном движении головки (каретки) ее иглами через красящую ленту, заправленную в специальную кассету (картридж). Переход к следующей строке достигается синхронизированным движением бумаги. Особенности матричных игольчатых принтеров такие, как высокая скорость печати, неприхотливость к качеству бумаги и низкая цена, при сравнительно невысоком качестве изображения предопределяют область их

применения. Домашнему и офисному применению этих принтеров часто препятствует традиционно высокий уровень их шума. В целом группа матричных игольчатых принтеров неуклонно сдает свои позиции конкурентам.

Струйные принтеры использует метод «выбрасывания» капель красителя на бумагу, соответствующая матрица печати представляет собой набор сопел (до 256), с которыми соединены емкости для чернил и управляющие механизмы (как правило - пьезоэлектрического типа). Фактически, имея цену на 150-200 долларов ниже, чем у лазерных аппаратов, и качество, приближающееся к ним, семейство струйных принтеров устойчиво увеличивает свою долю на рынке, чему способствует и их активная реклама. Струйные принтеры практически бесшумны и весьма универсальны (особенно аппараты с опцией цветной печати), цена их постоянно снижается, а качество печати улучшается. Вместе с тем, эти принтеры требуют весьма аккуратного обращения, а расходные материалы обходятся дороже, чем для лазерных.

В лазерных и светодиодных печатающих устройствах, как и в копировальных аппаратах, используется свойство фоточувствительности ряда материалов (например, селена), которые изменяют свой поверхностный электростатический заряд под воздействием света. Для реализации этого процесса, помимо тракта протяжки бумаги, данные принтеры содержат светочувствительный барабан, зеркальную систему развертки, устройства фокусировки и лазерный диод (или матрицу светодиодов). После зарядки и поточечной засветки светочувствительного барабана, соответствующей формируемому изображению, на него подается и закрепляется в соответствии с распределением электрического заряда специальный красящий порошок - тонер. Далее по барабану прокатывается бумага и снимает с него тонер. Окончательное закрепление изображения на бумаге достигается ее разогревом до температуры расплавления тонера. Лазерная печать (как и сами принтеры) дороже, чем у других групп печатающих устройств, однако цены на лазерные принтеры непрерывно и заметно снижаются, а расходы оправдываются весьма высоким качеством продукции, приближающейся к уровню полиграфии.

Сканер - устройство для считывания графической и текстовой информации в компьютер. Сканеры могут вводить в компьютер рисунки. С помощью специального программного обеспечения компьютер может распознавать символы во введенной через сканер картинке, это позволяет быстро вводить напечатанный (а иногда и рукописный) текст в компьютер.

Самые простые и дешевые сканеры - ручные. В них роль привода считывающего механизма выполняет рука. К основным достоинствам этого типа сканеров относятся небольшие габаритные размеры и сравнительно низкая цена, однако добиться высокого качества изображения с их помощью очень трудно, поэтому ручные сканеры можно использовать для ограниченного круга задач. Наиболее распространенный тип сканеров - планшетный. Почти все модели имеют съемную крышку, что позволяет сканировать толстые оригиналы (журналы, книги). Основное отличие барабанных сканеров состоит в том, что оригинал закрепляется на прозрачном барабане, который вращается с большой скоростью. Считывающий элемент располагается максимально близко от оригинала. Данная конструкция обеспечивает наибольшее качество сканирования.

Графопостроитель (плоттер) - устройство для вывода чертежей на бумагу. Плоттеры несколько дешевле, чем лазерные принтеры, но скорость печати у них ниже. Плоттеры бывают барабанного типа (работают с рулоном бумаги) и планшетного типа (в них лист бумаги лежит на плоском столе).

Стример - устройство для быстрого сохранения информации, находящейся на жестком диске. Стример записывает информацию на кассеты с магнитной лентой, похожие на кассеты для бытовых магнитофонов.

Стабилизация напряжения. Напряжение в сети может сильно колебаться. Для компьютера такие изменения напряжения являются нежелательными (особенно вредны резкие понижения напряжения), поэтому лучше подключать компьютеры через стабилизаторы. Наиболее надежную защиту от неприятностей, связанных с нестабильностью электропитания, осуществляют специальные устройства непрерывного питания *(UPS)*, которые не только обеспечивают строго постоянное напряжение питания, но и дают возможность работы ком-

пьютеров при полном отключении электропитания в течение от 5 мин до нескольких часов (в зависимости от мощности устройства).

Мультимедиа. С увеличением мощности компьютеров появилось новое направление в их использовании. Это так называемые мультимедиа-приложения, которые сочетают в себе не только текстовое представление данных, но и использование в программах графики, звука и видеоизображения. Использование этих видов данных требует наличие в компьютере соответствующих ресурсов и звуковой карты.

3.3. Программное обеспечение персонального компьютера

Под программным обеспечением ПК в общем случае понимается совокупность программ, установленных (записанных) на данный компьютер, обеспечивающих его работу, диалог с пользователем или предназначенных для решения каких-либо целевых пользовательских задач (вычисления по заданному алгоритму, подготовка текстового документа, создание рисунков и т.д.). Все множество программных продуктов может быть разделено на 2 основных класса:

- Системное (базовое) программное обеспечение является необходимым дополнением к техническим средствам ПК, обеспечивает предоставление пользователю определенных услуг в процессе работы.
- Прикладное программное обеспечение - набор средств для решения тех или иных проблем пользователя или создания программ, осуществляющих такое решение.

БАЗОВОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ

Рассмотрим более подробно основные элементы базового программного обеспечения. К системному обеспечению относятся:

- операционные системы (*MS-DOS, Windows, UNIX* и др.). Операционная система - программа, начинающая работу сразу после включения ПК, которая организует управление техническими устройствами, выполнение других программ и взаимодействие пользователя с компьютером.

- сервисные системы, облегчающие работу пользователя с операционными системами - т.н. программы-оболочки (*Far* и др.).

Первая операционная система, созданная фирмой *Microsoft* для персональных компьютеров, была операционная система *MS-DOS* (дисковая операционная система). Из самого названия следует, что основное назначение данной программы – управление информацией, расположенной на магнитных дисках. Любая информация на магнитных дисках хранится в виде файлов. Файл - это совокупность информации, имеющая имя и записанная на магнитный диск. Имя файла может содержать от 1 до 8 символов и обычно отражает либо содержание файла, либо имя его создателя. Кроме имени, файл имеет расширение. Расширение, как правило, указывает на тип файла (текстовые файлы, программы на различных алгоритмических языках и т.д.), начинается с точки, за которой следует от 1 до 3 символов. Например, *document.txt*. Здесь *document* – имя файла, *.txt* – расширение, указывающее на то, что файл текстовый. Примеры расширений: *.doc* файл документа; *.xls* – файл электронных таблиц *Excel*; *.bas* – программа на языке Бейсик; *.for* – программа на языке Фортран; *.pas* – программа на языке Паскаль; *.c* – программа на языке СИ; *.exe* – готовая к исполнению программа и т.д.

На одном магнитном диске может содержаться множество различных файлов. Поиск нужных файлов значительно облегчается за счет их объединения в так называемые каталоги. В каталогах хранятся имена файлов, данные об их размере, времени последнего обновления и другие сведения. Каталоги имеют имена, требования к которым те же, что и к именам файлов, за исключением одного - расширение не используется.

Жесткий диск (винчестер) – физический диск - при установке системы разбивается на несколько логических дисков. На каждом логическом диске имеется один главный каталог, называемый корневым. Его имя совпадает с именем логического диска и состоит из латинской буквы и следующего за ней двоеточия (рис. 2). Так, на гибком диске он может называться *A:* или *B:* (в зави-

симости от имени дисковода, в котором находится дискета), а на винчестере - *C:, D:, E:* и т.д.

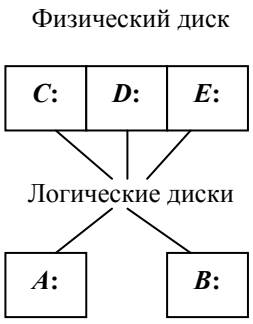


Рис. 2. Логические диски

Корневой каталог может содержать имена файлов, находящихся непосредственно в нем самом и имена каталогов, являющихся его подкаталогами. Последние также могут включать в себя имена содержащихся в них файлов и подкаталогов и т.д. Эта последовательность каталогов, вложенных друг в друга, называется деревом каталогов данного диска. Например, дерево каталогов может выглядеть так, как показано на рис. 3.

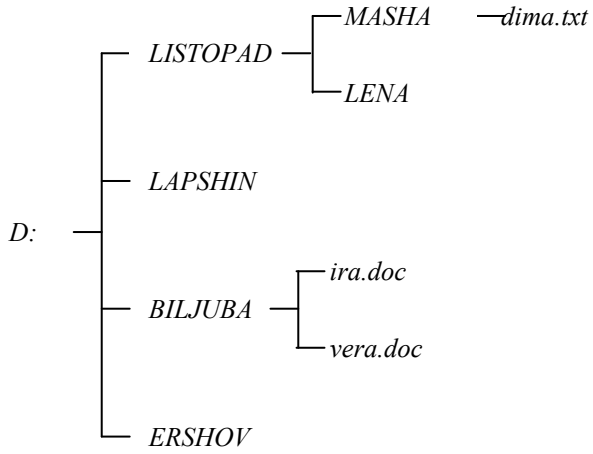


Рис. 3. Пример дерева каталогов

Каталог, в котором в данный момент работает пользователь, называется текущим. Для каких-либо действий с файлами из этого каталога соответствующая команда должна содержать только имя нужного файла. Если же интересующий пользователя файл находится в другом каталоге, необходимо указать не только его имя, но и последовательность каталогов от корневого до того, в

котором непосредственно находится нужный файл - путь к файлу. Имена каталогов, входящих в путь, отделяются друг от друга знаком "\".

Таким образом, для того, чтобы точно определить не только название, но и местоположение файла, используется понятие «полное имя файла». Полное имя файла включает в себя 4 элемента: название логического диска, путь, имя файла и расширение. Например, полное имя файла *dima.txt* имеет вид:

D:\LISTOPAD\ MASHA\dima.txt.

Интерфейс – совокупность программно-аппаратных средств, обеспечивающих пользователю необходимый уровень общения с компьютером. Дружественный интерфейс обеспечивает быстрое и комфортное взаимодействие пользователя с компьютером. Для управления файлами и каталогами на диске операционная система *MS-DOS* использует множество команд, которые пользователь должен набрать в командной строке, расположенной в нижней части экрана. Например, для того, чтобы скопировать с тем же именем файл *dima.txt* в каталог *ERSHOV*, необходимо набрать следующую команду:

copy D:\LISTOPAD\ MASHA\dima.txt D:\ ERSHOV

Как видно из примера, пользователь должен потрудиться, чтобы работать с файлами в рамках операционной системы *MS-DOS*. Следовательно, данная система не обеспечивает дружественного интерфейса.

Для улучшения интерфейса стали создаваться специальные программы – оболочки, работающие под управлением операционной системы. Наиболее распространенной такой программой стала программа *Far*, названная в честь ее создателя Питера Нортона.

Следующий крупный шаг в деле улучшения интерфейса был сделан с появлением *Windows*-технологий. Первые версии Windows представляли собой, как и *Far*, операционные оболочки и работали под управлением операционной системы *MS-DOS*. Начиная с версии *Windows-95*, эта программа становится полноценной операционной системой, сочетая в себе свойства операционной оболочки и графической интерфейсной среды.

Основные особенности *Windows*-технологий:

- Графический интерфейс. В Windows существуют различные графические символы (значки), меню, диалоговые окна, что избавляет пользователя от необходимости запоминать команды *DOS*. Основной принцип – «делай то, что видишь», то есть подключается ассоциативное мышление. Например, глядя на кнопку с изображением ножниц, пользователь может понять, что эта кнопка означает команду «вырезать».
- Многооконный интерфейс. Все окна имеют стандартную структуру. Для всех программ в *Windows* используются одни и те же команды управления, что существенно облегчает работу в этой среде.
- В *Windows* можно одновременно работать с несколькими программами и переключаться от одной к другой. Находясь в *Windows*, пользователь имеет в своем распоряжении буфер обмена, с помощью которого можно копировать или переносить информацию как в пределах одной программы, так и между отдельными программами.

Для копирования файла в программе Проводник достаточно переместить соответствующий значок приложения (файла) в нужную папку (каталог).

ПРИКЛАДНОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ

Системы и языки программирования

Системы программирования - инструменты для продуцирования пользователем собственных программных продуктов при использовании алгоритмических языков высокого уровня (Бейсик, Фортран, Паскаль, Си). Выбор языка программирования зависит от характера решаемой задачи и подготовки программиста. Например, Бейсик используется в мире как один из основных языков, предназначенных для обучения школьников и студентов основам алгоритмизации и программирования. Он имеет простой синтаксис и хорошие графические возможности. Фортран используется для проведения сложных математических расчетов. Он имеет богатые библиотеки – специальные программы, используемые для сложных вычислений. Паскаль был разработан как язык структурного программирования и получил широкое распространение в нашей

стране. Одно из его достоинств – возможность эффективной работы с различными типами данных. Язык Си имеет более сложный синтаксис и, в основном, используется профессиональными программистами.

В последнее время широкое распространение получили языки программирования, поддерживающие графический интерфейс и технологии *Windows*. Пример – *Visual Basic*.

После написания программы на языке программирования высокого уровня необходимо перевести ее на язык машинных кодов. В зависимости от способа этого перевода различают два класса языков программирования: трансляторы и интерпретаторы.

Рассмотрим работу языка-транслятора на примере Фортрана. В этом случае обработка (прохождение задачи на ЭВМ) проводится в три этапа.

1. Написание программы. В соответствии с алгоритмом составляется программа на языке Фортран. В результате на диске появляется файл с расширением *.for* – исходный модуль.
2. Трансляция. При этом с помощью специальной программы – транслятора – происходит перевод программы с языка Фортран на язык машинных кодов. В результате на диске появляется новый файл с расширением .obj – объектный модуль.
3. Компоновка. На этом заключительном этапе устанавливаются необходимые связи и подключаются нужные библиотеки. В результате на диске появляется файл с расширением *.exe* – рабочий модуль. Это – готовая к выполнению программа, которая может быть запущена на счет.

Рассмотрим работу интерпретатора на примере языка Бейсик. В этом случае после написания программы образуется файл с расширением *.bas*. Этот файл сразу запускается на исполнение. При этом каждый оператор последовательно «интерпретируется», то есть переводится на машинный язык и тут же исполняется. Поскольку в этом случае отсутствует предварительная подготовка, работа интерпретатора обычно проходит более медленно, чем транслятора.

Система программирования включает в себя следующие элементы:

- язык программирования, транслятор и другие вспомогательные программы;
- библиотеки языка программирования;
- программы-оболочки, обеспечивающие дружественный интерфейс при работе пользователя с языком программирования.

Пакеты прикладных программ

Пользовательские пакеты прикладных программ (ППП) можно разбить на три группы:

- функциональные ППП;
- интегрированные ППП;
- специализированные (проблемно-ориентированные) ППП.

Рассмотрим отдельно каждую группу.

Функциональные пакеты это - группы программ, имеющих определенное назначение.

Текстовые редакторы - программы, создающие и выводящие на печать документы разной сложности. Эти программы могут использоваться в задачах делопроизводства и издательской деятельности. В настоящее время наиболее распространенным пакетом данного типа является текстовый процессор *Word*, который входит в программную группу *Microsoft Office*. Основные особенности пакета:

- Возможность эффективной работы с абзацем (редактирование и форматирование). Редактирование предполагает изменение текста, а форматирование – изменение его внешнего вида. При наборе текста внутри абзаца нельзя пользоваться пробелом, кроме разделителя между слов; абзацный отступ устанавливается специальными инструментами; клавиша Enter может быть использована только в конце абзаца.
- Возможность эффективной работы с иллюстративными таблицами.
- Возможность набора математических формул.

- Возможность импорта объектов, например рисунков, а также создания собственных рисунков, графиков и чертежей.
- Возможность качественного форматирования страниц (нумерация, поля, колонтитулы и т.д.).
- Возможность использования стилей.

Электронные таблицы – программы, применяемые для решения плановых, бухгалтерских и любых связанных с числовыми, вычисляемыми или текстовыми таблицами задач. Окно таблицы состоит из множества поименованных ячеек, каждая из которых может содержать данные. Данные могут быть трех типов: число, текст и формула. Наиболее распространенным пакетом данного типа является табличный процессор *Excel*, который входит в программную группу *Microsoft Office*. Основные особенности пакета:

- Возможность эффективной работы с рядами данных. Ряд может располагаться в строке или столбце таблицы. Использование абсолютной, относительной и смешенной адресации позволяет осуществлять различные действия с рядами и копирование действий.
- Использование Мастера диаграмм для графической иллюстрации исходных данных и результатов расчетов.
- Использование Мастера функций для вычислений. Пакет содержит несколько классов функций (математические, статистические, финансовые и т.д.).
- Возможность использования Пакета анализа для решения сложных задач математики и статистики, а также инструмента Поиск решения для задач оптимизации и линейного программирования.

Системы управления базами данных - СУБД, обеспечивающих поиск, выборку, дополнение и другую обработку информации, сохраняемой в определенной форме (базы данных). Представителем таких программ является *Access*, также входящий в программную группу *Microsoft Office*.

Графические пакеты, позволяющие представить полученные результаты (численные данные) в виде графиков и диаграмм, создающие и редактирующие

различные рисунки (*CorelDraw* – пакет прецизионной графики, использующийся в рекламном бизнесе, *PowerPoint* – пакет, использующийся для презентаций и т.д.).

Пакеты формульных преобразований и математических вычислений (*MathCad, , MatLab*), дающие наглядное решение математических задач в аналитическом и численном виде без использования языков программирования.

Интегрированные пакеты

Это - программы, совмещающие возможности функциональных пакетов и обеспечивающие возможность решения комплексных проблем в рамках одного программного продукта. Например, пакет *FrameWork* одновременно сочетает в себе функции текстового редактора, электронной таблицы и СУБД. Недостаток такого подхода – ограниченность ресурсов компьютера, что делает невозможным высококачественную работу каждой из функций пакета. Однако с развитием *Windows*-технологий необходимость в интегрированных пакетах стала отпадать, поскольку появилась возможность одновременной работы сразу с несколькими приложениями. Например, пользователь может эффективно работать с пакетами *Word и Excel,* осуществляя обмен данных с помощью Буфера обмена (рис. 4.).

Рис. 4. Обмен данными

Проблемно-ориентированные (специализированные) пакеты

Это – программы, предназначенные для использования в конкретной области деятельности: обучение, бухгалтерский учет и делопроизводство, инженерные разработки и научные исследования, автоматизация рабочих мест (АРМ) руководителя, конструктора, врача и т.д. Примеры: бухгалтерский пакет «1С-бухгалтерия», правовой пакет «Консультант-плюс», пакет бизнес-планирования «*Project Expert*» и т.д.

Основы алгоритмизации и программирования. Языки программирования высокого уровня.

Алгоритм - это последовательность действий, приводящих к намеченному результату. Алгоритм может быть представлен в обычной словесной форме в виде последовательности пронумерованных предложений.

Пример. Вычислить значение функции $y=x^2+bx+c$ при любых значениях x, b, c.

Алгоритм в словесной форме выглядит следующим образом:

1. Ввести в память ЭВМ числовые значения параметров x, b, c.
2. Вычислить значение функции y в соответствии с заданной формулой.
3. Напечатать на экране дисплея числовые значения параметров x, b, c и функции y.
4. Закончить алгоритм.

Однако наиболее наглядным является графический способ описания алгоритмов (в виде блок-схем). В этом случае каждому действию соответствует определенная алгоритмическая фигура (блок). Блоки соединяются между собой стрелками, указывающими порядок выполнения действий.

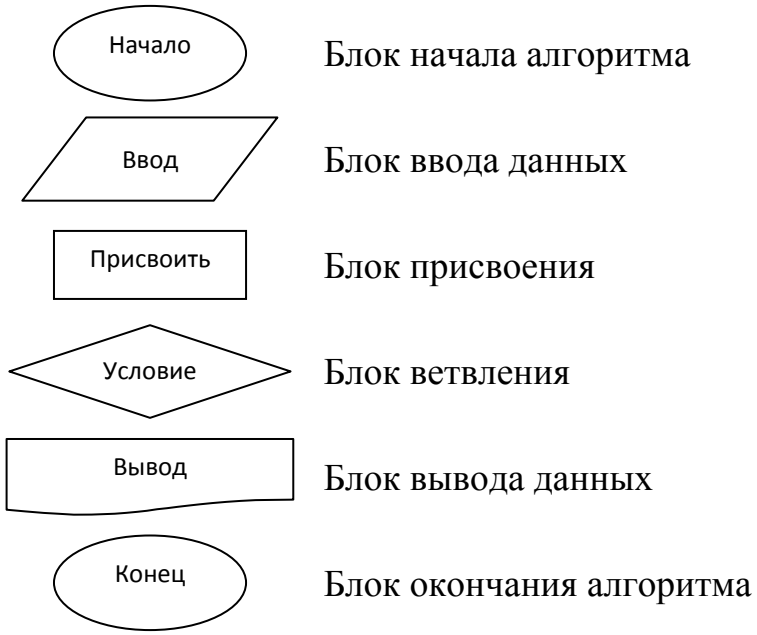


Рис. 5. Типы блоков

Запишем алгоритм нашего примера в графической форме (рис. 6)

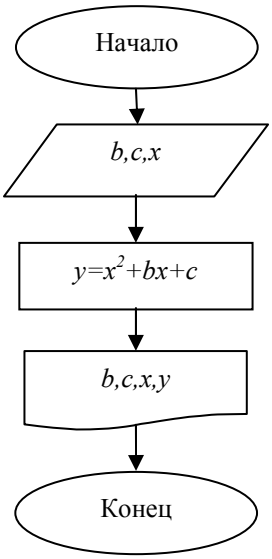


Рис 6.

Данный алгоритм представляет собой пример линейного алгоритма. Линейный алгоритм - это такой, в котором действия выполняются последовательно, в порядке расположения блоков.

Элементы алгоритмического языка Фортран

Алгоритмический язык Фортран является одним из наиболее распространенных в мире языков программирования, особенно для решения задач, требующих сложных вычислений. Программа на Фортране представляет собой последовательность строк-операторов. Рассмотрим основные элементы языка.

Структура строки. Строка программы имеет следующую структуру. Первая позиция (крайняя слева) используется для комментариев. Для этого в ней ставится латинская буква С. В этом случае вся информация в строке воспринимается как комментарий к программе. Позиции 2-5 используются для метки. Метка ставится только в случае необходимости и представляет собой целое число от 1 до 9999. Позиция 6 используется для переноса строки в том случае, если оператор слишком длинный. Позиции 7 и 8 не используются. Позиции 9-72 отведены под оператор.

Основные символы. Знаки арифметических операций. Операции отношения. Основные символы языка Фортран - это латинские буквы, цифры, знаки арифметических операций (+ - сложить; – - вычесть; * - умножить; / - разделить; ** - возвести в степень), а также некоторые специальные символы (. ; , ; ' ; (;) ; пробел ; /). Знаки операций отношения обозначаются сочетаниями из двух латинских букв, заключенных в точки (.LT. - меньше; .GT. - больше; .LE. - меньше или равно; .GE. - больше или равно; .EQ. - равно; .NE. - не равно).

Константы. В Фортране различают два типа числовых констант - вещественные и целые. Они отличаются по внешнему виду, способу внутреннего хранения и обработки их в ЭВМ. Вещественные числа содержат целую и дробную часть, могут содержать порядок с символом E. Например, число 235,82 в Фортране запишется 235.82 или 0.23582E03. Вместо запятой целая и дробная части разделяются десятичной точкой. Целые числа не содержат десятичной точки, дробной части или порядка с символом E.

Переменные. Переменные Фортрана - это величины, принимающие в процессе вычислений различные значения. Имя переменной - сочетание 1-6 латинских букв или цифр (первая - буква). Например, *A, B2, SIGMA*. Так же, как и константы, переменные бывают вещественными и целыми. Целые переменные имеют имена, которые начинаются с одной из букв: *I, J, K, L, M*, *N* (например, *IRA, K12*). Остальные переменные являются вещественными. Существуют следующие правила операций с переменными и константами:

- Операции с двумя целыми переменными (константам) дают целый результат.
- Операции с двумя вещественными переменными (константам) дают вещественный результат.
- Если один операнд целый, а другой вещественный, то результат – вещественный.

Наиболее распространенная ошибка возникает при делении целого числа (переменной) на целое. Например, 2/3 даст в результате ноль, так как дробная часть будет отброшена. Поэтому в таких задачах необходимо перевести хотя бы

один из операндов в вещественную форму, поставив после числа десятичную точку: 2./3 даст правильный результат 0,6666667.

Стандартные функции. Некоторые наиболее употребительные функции вычисляются в Фортране автоматически (приближенно, в виде ряда). Такие функции называются стандартными и приведены в табл. 1.

Таблица 1

Обычная запись	**Запись в Фортране**
$\sin x$	SIN(X)
$\cos x$	COS(X)
$\mathrm{tg}\, x$	TAN(X)
$\mathrm{arctg}\, x$	ATAN(X)
$\arcsin x$	ASIN(X)
$\arccos x$	ACOS(X)
$\sqrt{x}$	SQRT(X)
e^x	EXP(X)
$\ln x$	ALOG(X)
$\lg x$	ALOG10(X)
$\lvert x \rvert$	ABS(X)

Аргументом стандартной функции может быть число, переменная или выражение вещественного типа. Результат вычислений стандартной функции - вещественный. После имени стандартной функции обязательно стоит скобка.

Арифметические выражения. Арифметическим выражением называется набор констант, переменных и функций, соединенных знаками арифметических операций. Последовательность выполнения операций в выражении определяется скобками, а если их нет, то операции выполняются слева направо в соответствии с приоритетом:

- вычисление функций (высший);
- возведение в степень;
- умножение, деление;
- сложение, вычитание (низший).

При необходимости смены приоритета можно пользоваться круглыми скобками.

Например, выражение $\sqrt{\frac{ay^2+x}{bz^2+u}} * \sin^2 t$ в Фортране имеет вид:

```
SQRT((a*y**2+x)/(b*z**2+u))*SIN(t)**2
```

Если вместо квадратного корня стоит, например, кубический, то такая функция отсутствует, и необходимо выражение возвести в степень 1/3. При этом необходимо помнить о правилах расчетов между целыми и вещественными константами:

```
(((a*y**2+x)/(b*z**2+u))*SIN(t)**2)**(1./3.)
```

Простейшие операторы Фортрана

Оператор присваивания

Структура оператора: A=B

Здесь A - переменная; B - арифметическое выражение, значение которого присваивается переменной A. Например, последовательность операторов: x=1; y=3; z=2x+5y выполняется следующим образом. Сначала в условные ячейки памяти, соответствующие переменным x и y, будут записаны соответствующие константы. Затем процессор производит вычисление выражения 2*1+5*2, а результат (12) будет направлен в ячейку, соответствующую переменной z (рис. 7).

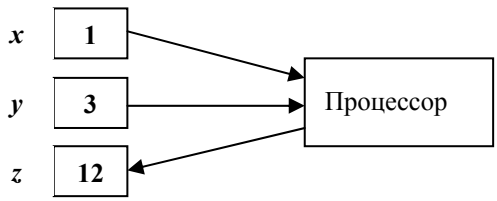


Рис. 7. Иллюстрация работы оператора присваивания

Оператор ввода данных

Структура оператора: READ(*,*) A,B

Здесь READ(*,*) - имя оператора; *A,B* - список переменных, значения которых вводятся с клавиатуры. Первая звездочка в имени оператора указывает

на то, что ввод осуществляется с клавиатуры, а вторая – на то, что ввод бесформатный.

Например, рассмотрим работу оператора READ(*,*) x,y.

Компьютер, выполняя этот оператор, находится в ожидании ввода двух чисел. Ввод осуществляется с клавиатуры через запятую, по окончании ввода нужно нажать клавишу *Enter*. После этого в условные ячейки памяти *x* и *y* будут записаны соответствующие данные. В отличие от предыдущего оператора, эти данные могут меняться при повторном запуске программы.

Оператор вывода данных

Структура оператора: WRITE(*,*) *'текст',*A,B

Здесь WRITE(*,*) - имя оператора, 'текст' - необходимые пояснения (могут отсутствовать), A,B - список переменных, значения которых выводятся на экран дисплея.

Например, пусть в процессе работы программы возникла ситуация, показанная на рис. 7, *(x=1; y=3; z=12)*. Теперь запишем оператор WRITE(*,*) x,y,z. В этом случае на экране дисплея появится следующая информация:

1 3 12

Более наглядным является вывод с использованием текстовых фрагментов. В результате работы оператора

WRITE(*,*) ‘x=’,x,’y=’,y,’z=’,z

на экране дисплея появится следующая информация:

x= 1 y= 3 z= 12

Оператор окончания программы

Структура оператора: END

Ставится в конце программы и не может иметь метки.

3.4. Программирование задач на простые переменные

Запишем программу примера линейного алгоритма, блок-схема которого приведена на рис. 6.

```
WRITE(*,*)’Введите b,c,x’
READ(*,*)b,c,x
y=x**2+b*x+c
WRITE(*,*)’b=’,b,’c=’,c,’x=’,x,’y=’,y
END
```

Разветвляющийся алгоритм

Часто требуется в зависимости от конкретного набора данных или промежуточных результатов выбрать один из двух или более различных вариантов продолжения вычислительного процесса. Если в зависимости от выполнения условия выбирается один из двух (или более) различных вариантов вычислительного процесса, то такой алгоритм называется разветвляющимся. В зависимости от того, удовлетворяется ли условие, выполняется «действие 1» или «действие 2», после чего вычислительный процесс вновь сводится в единое русло.

Для программирования разветвлений используются операторы условного и безусловного перехода. Кроме того, могут использоваться структурный оператор IF...ELSEIF...ELSE ...ENDIF, что делает программу более компактной. Рассмотрим вначале структуру операторов условного и безусловного перехода, поддерживающих классический стиль программирования. В этом случае алгоритм можно представить в виде блок-схемы, каждому элементу которой соответствует определенный оператор.

Оператор условного перехода

Структура оператора: IF (A¤B) GOTO M

Здесь *IF* - имя оператора; *A* и *B* - арифметические выражения; ¤ - операция отношения; *M* - метка оператора, к которому осуществляется переход при выполнении условия. Если условие не выполнено, управление передается на

следующий после *IF* оператор. На блок-схеме этот оператор соответствует блоку разветвлений.

Оператор безусловного перехода

Структура оператора: GOTO M

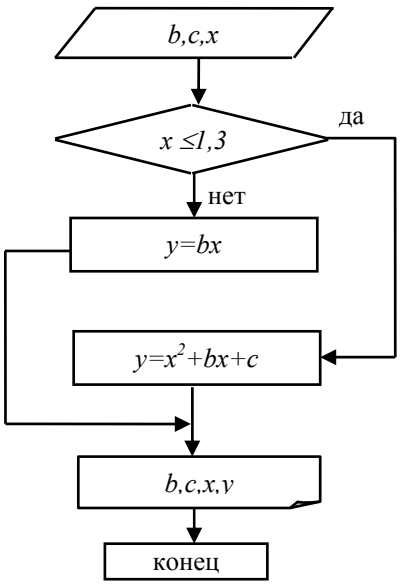
Здесь GOTO - имя оператора, M - метка оператора, к которому осуществляется переход. В отличие от оператора условного перехода, данный оператор осуществляет переход к оператору с меткой *M* всегда, то есть без проверки каких-либо условий. На блок-схеме оператор безусловного перехода соответствует обводящей стрелке, указывающей соответствующий переход.

Пример 1. Вычислить значение функции

$$y = \begin{cases} x^2 + bx + c, & \text{если} \quad x \le 1{,}3; \\ bx, & \text{если} \quad x > 1{,}3 \end{cases}$$

при любых заданных значениях параметров *b, c, x*. В зависимости от значения *x* вычисление *y* происходит либо по первой, либо по второй формуле.

Блок-схема



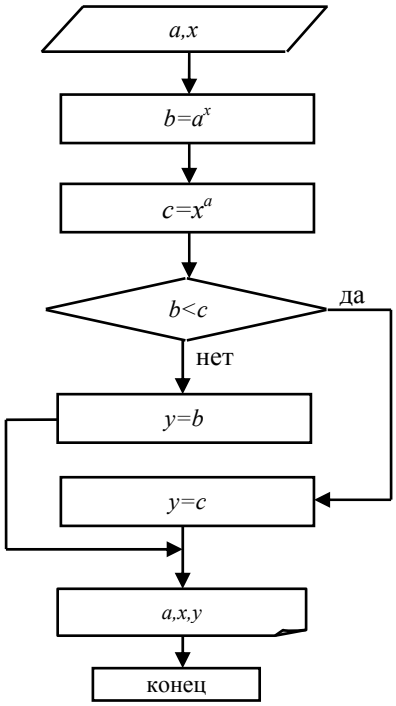
Программа на Фортране

```
WRITE(*,*) ’Введите b,c,x’
READ(*,*) b,c,x
IF (x.LT.1.3) GOTO 5
y=b*x+c
GOTO 6
5  y=x**2+b*x+c
        6  WRITE(*,*)
’b=’,b,’c=’,c,’x=’,x,’y=’,y
END
```

Пример 2. Из двух возможных значений, просчитываемых по формулам *ax* и *xa*, выбрать наибольшее и присвоить его переменной *y*. Таким образом, требуется найти $y = \max\{ax, xa\}$, если заданы *a* и *x*.

В отличие от предыдущей задачи, где, в зависимости от проверяемого условия, производилось вычисление *y* по одной из заданных формул, в данном примере производятся вычисления по обеим формулам, а лишь затем производится отбор максимального значения.

Блок-схема



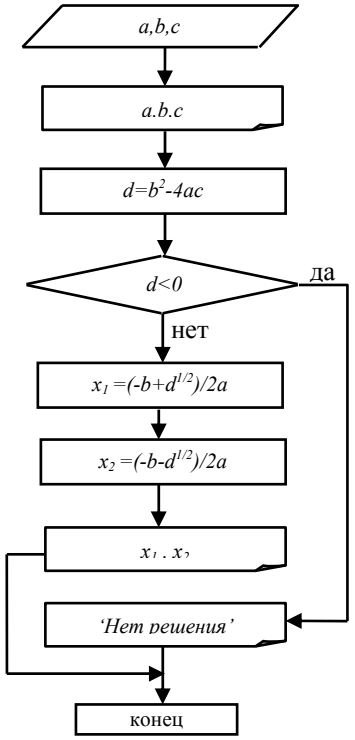
Программа на Фортране

```
WRITE(*,*) ’Введите a,x’
READ(*,*) a,x
b=a**x
c=x**a
IF (b.LT.c) GOTO 7
y=b
GOTO 8
7     y=c
8     WRITE(*,*) ’a=’,a,’b=’,b
WRITE(*,*)’y=’,y
END
```

Пример 3. Составить программу для решения квадратного уравнения $ax^2 + bx + c = 0$. Данное уравнение имеет решение, если дискриминант $D = b^2 - 4ac$ не является отрицательным.

После ввода коэффициентов *a, b, c* вычисляется *D* и проверяется условие $D < 0$. Если условие выполняется, то выводится сообщение «нет решения». Если условие $D < 0$ не выполняется, вычисляются корни по соответствующим формулам.

Блок-схема



Программа на Фортране

```
      WRITE(*,*) 'Введите a,b,c'
      READ(*,*) a,b.c
      WRITE(*,*) 'a=',a,'b=',b',c=',c
      d=b**2-4*a*c
      IF (d.LT.0) GOTO 2
      x1=(-b+SQRT(d))/(2*a)
      x2=(-b-SQRT(d))/(2*a)
      WRITE(*,*) 'x1=',x1,'x2=',x2
      GOTO 3
2     WRITE(*,*) 'Нет решения'
3     WRITE(*,*)
      END
```

Примечание. Поскольку оператор END не может иметь метки, перед ним ставится пустой оператор WRITE(*,*), который приводит к простому пропуску строки.

В данном случае, в отличие от предыдущих примеров, сначала вычисляется значение переменной *d*, зависящее от входных параметров. Затем, в зависимости от знака *d*, алгоритм разветвляется на две ветки: одна – линейный алгоритм с печатью на конце, вторая – один блок печати.

Алгоритм, в котором в зависимости от условия возможны два пути продолжения вычислительного процесса, называется простым разветвляющимся

алгоритмом. Алгоритм, в котором в зависимости от выполнения условий возможны несколько путей продолжения вычислительного процесса, называется многоразветвляющимся алгоритмом.

Для программирования сложных разветвлений целесообразно использовать составной оператор IF. В этом случае применяется стиль структурного программирования, программа становится более компактной, хотя и не соответствует классической блок-схеме.

Составной оператор IF-ELSEIF-ELSE-END IF

Структура оператора:

```
IF(A¤B) THEN
 оператор 1
ELSEIF(C¤D) THEN
 оператор 2
ELSEIF(E¤F) THEN
 оператор 3
ELSE
 оператор k
END IF
```

Если условие *(A¤B)* выполняется, то выполняется оператор 1, и управление передается на конец конструкции *(END IF)*. Иначе, если *(ELSEIF)* выполняется условие *(C¤D),* то выполняется оператор 2, управление передается на *END IF*. Если нет, то проверяется следующее условие и т.д. *(ELSEIF* может быть несколько). Если ни одно из условий не выполнено *(ELSE)*, то выполняется оператор *k.*

Структура ELSEIF может вообще отсутствовать, тогда оператор принимает более простой вид:

```
IF(A¤B) THEN
 оператор 1
ELSE
 оператор 2
END IF
```

Пример. Вычислить значение функции

$$y = \begin{cases} x^2 + bx + c, & \text{если} \quad x < 0; \\ bx + c, & \text{если} \quad 0 \le x \le 1; \\ \sqrt{x}, & \text{если} \quad x > 1 \end{cases}$$

при любых заданных значениях x ,b ,c.

Перед решением задачи целесообразно нарисовать ось x и отложить на ней точки ветвления (рис. 8.). Блоки условного перехода расставляем в порядке расположения точек ветвления и в том же количестве. Проверяемые условия обычно имеют знаки «меньше» или «меньше или равно». Первое проверяемое условие в данном случае $x < 0$. Если оно не выполнено, вычисление проводится по первой формуле, не имеет смысла проверять обратное условие. Второе проверяемое условие - $x \le 1$. Если оно выполнено, то вычисление осуществляется по второй формуле. Если оба условия не выполнены, то нет смысла проверять последнее условие, а вычисление проводится по третьей формуле.

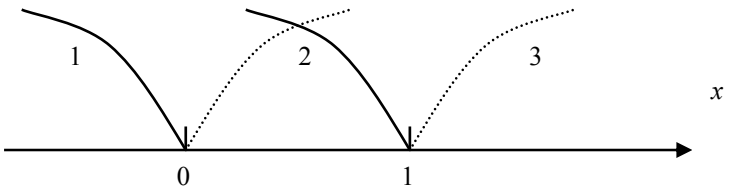


Рис. 8. Точки ветвления

После выполнения действия по одному из разветвлений вычислительный процесс опять сводится в одно русло.

При построении многоразветвляющегося алгоритма следует пользоваться правилом:

вначале записываются блоки разветвления, число которых на один меньше, чем количество разветвлений; эти блоки содержат лишь односторонние условия (обычно со знаками «меньше» и «меньше или равно»); двойные условия записывать в блок-схеме и программе нельзя;

затем записываются блоки вычисления в порядке, обратном исходным формулам;

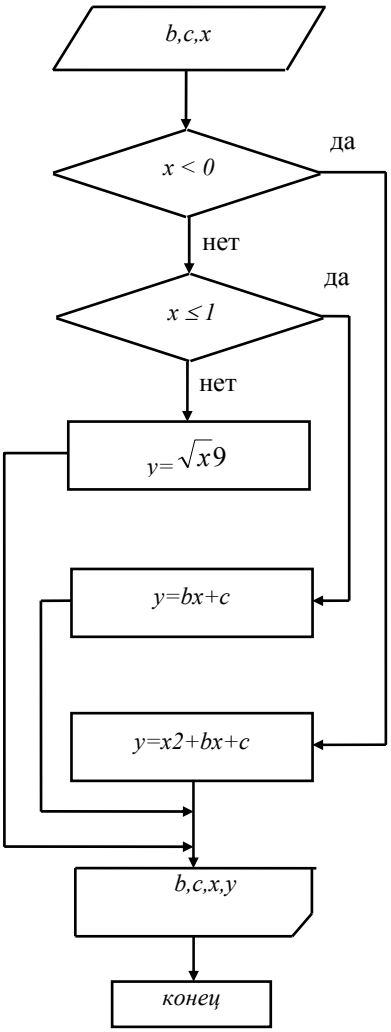
от каждого из блоков вычисления процесс переходит к одному блоку (обычно блоку печати), для этого после каждого блока вычисления (кроме по-

следнего) следует использовать безусловный переход (стрелка на блок-схеме или оператор GOTO в программе);

число проверяемых условий всегда на единицу меньше числа разветвлений.

Приведем блок-схему и два варианта программы. Первый вариант полностью соответствует блок-схеме. При этом блоки вычислений записываются в обратном порядке, а возвращение вычислительного процесса в единое русло (к блоку печати результатов) осуществляется с помощью оператора безусловного перехода GOTO. Более рациональной является запись программы с использованием структурного оператора IF (второй вариант).

Блок-схема



Программа на Фортране

Первый вариант

```
      WRITE(*,*) ’Введите b,c,x’
      READ(*,*) b,c,x
      IF (x.LT.0) GOTO 1
      IF (x.LE.1) GOTO 2
      y=SQRT(x)
      GOTO 3
      2  y=b*x+c
        GOTO 3
   1  y=x**2+b*x+c
      3  WRITE(*,*)
   ’b=’,b,’c=’,c,’x=’,x,’y=’,y
        END
```

Второй вариант

```
      WRITE(*,*) ’Введите b,c,x’
      READ(*,*) b,c,x
      IF (x.LT.0) THEN
      y=x**2+b*x+c
      ELSE IF (x.LE.1) THEN
      y=b*x+c
      ELSE
      y=SQRT(x)
        END IF
   WRITE(*,*)
   ’b=’,b,’c=’,c,’x=’,x,’y=’,y
        END
```

Циклический алгоритм

Циклическим алгоритмом называется алгоритм, часть которого выполняется многократно с различными значениями изменяющейся по определенному закону переменной (переменной цикла). На рис. 9 приведена обобщенная схема циклического алгоритма.

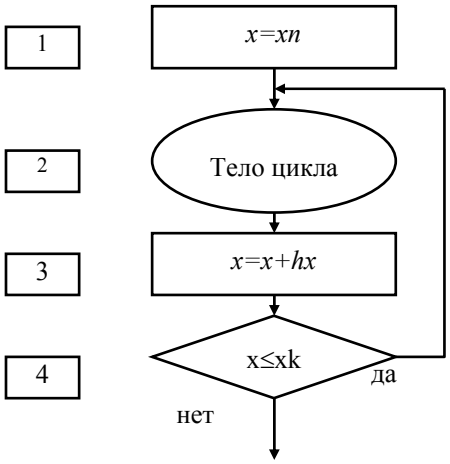


Рис. 9. Обобщенная схема циклического алгоритма

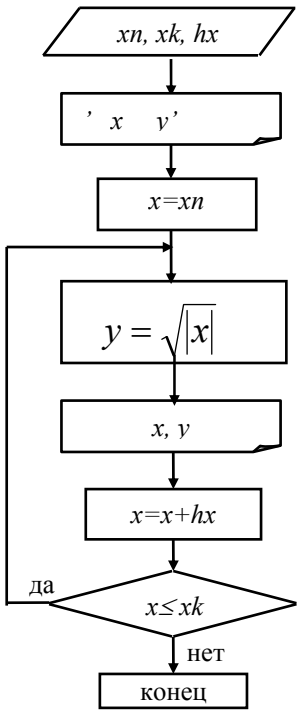
Для организации цикла определяются переменная цикла (*x*) и его параметры - начальное и конечное значение переменной цикла и шаг ее изменения (*xn, xk, hx*). Блоки 1 – 4 называются блоками организации цикла:

1. Блок начального присвоения (переменной цикла присваивается начальное значение).
2. Тело цикла - многократно повторяющаяся часть алгоритма, внутри которой переменная цикла не изменяется. Телом цикла может быть любой алгоритм: линейный, разветвляющийся или циклический.
3. Блок приращения значения переменной цикла (переменная цикла изменяет свое значение на величину шага).
4. Блок проверки условия продолжения цикла (цикл выполняется до тех пор, пока значение переменной цикла не превысит *xk*).

Пример 1. Вычислить таблицу значений $y=\sqrt{|x|}$ при x, изменяющемся в интервале $[xn;xk]$ с шагом hx. Параметры цикла xn, xk, hx задаются произвольно.

Телом цикла в данном примере является линейный алгоритм, состоящий из двух блоков: блока вычислений и блока печати.

Блок-схема



Программа на Фортране

```
      WRITE(*,*) 'Введите xn,xk,hx'
      READ(*,*) xn,xk,hx
      WRITE(*,*) '  x      y'
      x=xn
10    y=SQRT(ABS(x))
      WRITE(*,*) x,y
      x=x+hx
      IF (x.LE.xk) GOTO 10
      END
```

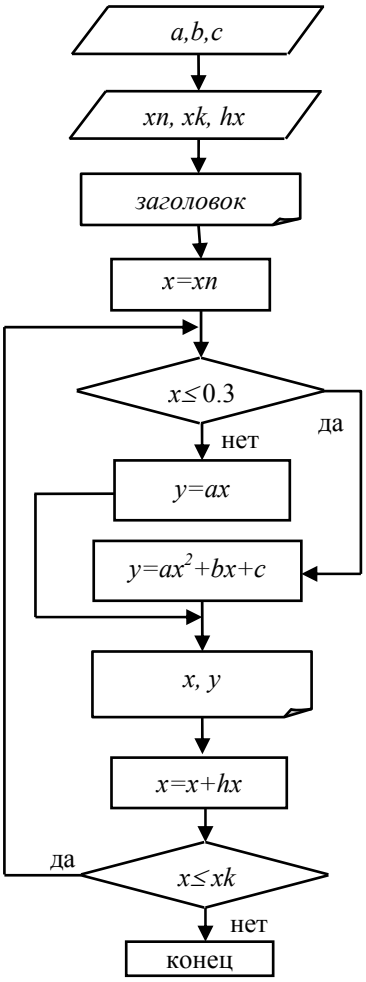
Одномерный циклический алгоритм реализуется, если имеется лишь одна переменная цикла. Рассмотрим случай, когда телом цикла является разветвляющийся алгоритм.

Пример 2. Вычислить таблицу значений функции

$$y=\begin{cases} ax^2+bx+c, & \text{если} \quad x \le 1{,}3; \\ ax, & \text{если} \quad x > 1{,}3 \end{cases}$$

при различных значениях переменной x, изменяющейся в интервале от $xn = 0{,}1$ до $xk = 0{,}4$. по закону арифметической прогрессии с шагом $hx = 0{,}1$. Здесь a, b, c – заданные константы ($a = 0{,}5$; $b = 1$; $c = 2$).

Блок-схема



Программа на Фортране

```
      WRITE(*,*) 'Введите a,b,c'
      READ(*,*) a,b,c
      WRITE(*,*) 'Введите xn,xk,hx'
      READ(*,*) xn,xk,hx
      WRITE(*,*) '   x      y'
      x=xn
4     IF (x.LE.1.3) GOTO 5
      y=ax
      GOTO 6
5     y=a*x**2+b*x+c
6     WRITE(*,*) x,y
      x=x+hx
      IF (x.LE.xk) GOTO 4
      END
```

Часто бывает полезным перед тем, как запустить программу на счет, прокрутить алгоритм вручную. Приведем ручную прокрутку данного алгоритма.

1. Ввод исходных данных и параметров цикла.
2. Печать заголовка таблицы.
3. Присвоение начального значения переменной цикла x = 0,1.
4. x≤ 0,3 – да; y = ax = 0,5*0,1 = 0,05; печать 0,1 0,05; x = 0,2; x≤ 0,4 – да.
5. x≤ 0,3 – да; y = ax = 0,5*0,2 = 0,1; печать 0,2 0,1; x = 0,3; x≤ 0,4 – да.
6. x≤ 0,3 – да; y = ax = 0,5*0,3 = 0,15; печать 0,3 0,15; x = 0,4; x≤ 0,4 – да.
7. x≤ 0,3 – нет; y = ax2+bx+c = 0,5*0,4*0,4+1*0,4+2 = 2,48; печать 0,4 2,48; x = 0,5; x≤ 0,4 – нет.
8. Конец.

Циклический алгоритм, содержащий в теле цикла циклы по другим переменным, называется многомерным циклом. Правило построения многомерных циклов:

1. Вначале записываются блоки начального присвоения, начиная с самого внешнего и кончая самым внутренним циклом.

2. Далее следует тело самого внутреннего цикла.

3. В заключение записываются парами блоки №3 и №4 (приращения и проверки условия продолжения цикла), начиная с самого внутреннего и кончая самым внешним циклом.

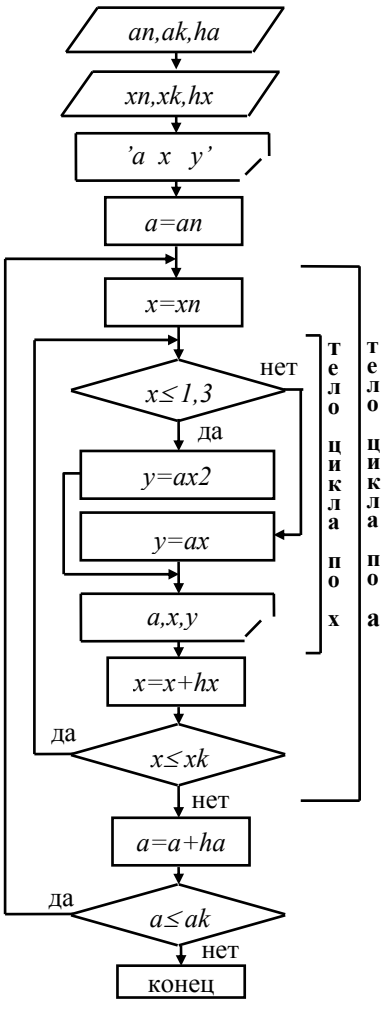
Рассмотрим пример двумерного циклического алгоритма, в котором телом внутреннего цикла является разветвляющийся алгоритм.

Пример 3. Вычислить таблицу значений функции

$$y = \begin{cases} ax^2, & \text{если } x \le 1{,}3; \\ ax, & \text{если } x > 1{,}3 \end{cases}$$

так, что для каждого значения a, изменяющегося в интервале $[an;ak]$ с шагом ha, переменная x принимает все значения в интервале $[xn;xk]$ с шагом hx.

Блок-схема



Программа на Фортране

```
      WRITE(*,*)'Введите an,ak,ha'
      READ(*,*) an,ak,ha
      WRITE(*,*)'Введите xn,xk,hx'
      READ(*,*) xn,xk,hx
      WRITE(*,*)' a    x    y'
      a=an
20    x=xn
15    IF (x.LE.1.3) THEN
      y=a*x**2
      ELSE
      y=a*x
      END IF
      WRITE(*,*) a,x,y
      x=x+hx
      IF (x.LE.xk) GOTO 15
      a=a+ha
      IF (a.LE.ak) GOTO 20
      END
```

Пример 4. Поиск экстремального значения функции методом простого перебора значений аргументов.

Найти значения *x, y, z* с наименьшим *z = f(x,y).* Каждый из аргументов *x, y* изменяется с заданным шагом в заданной области (см. рис. 10.).

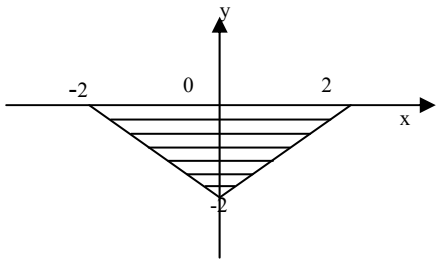
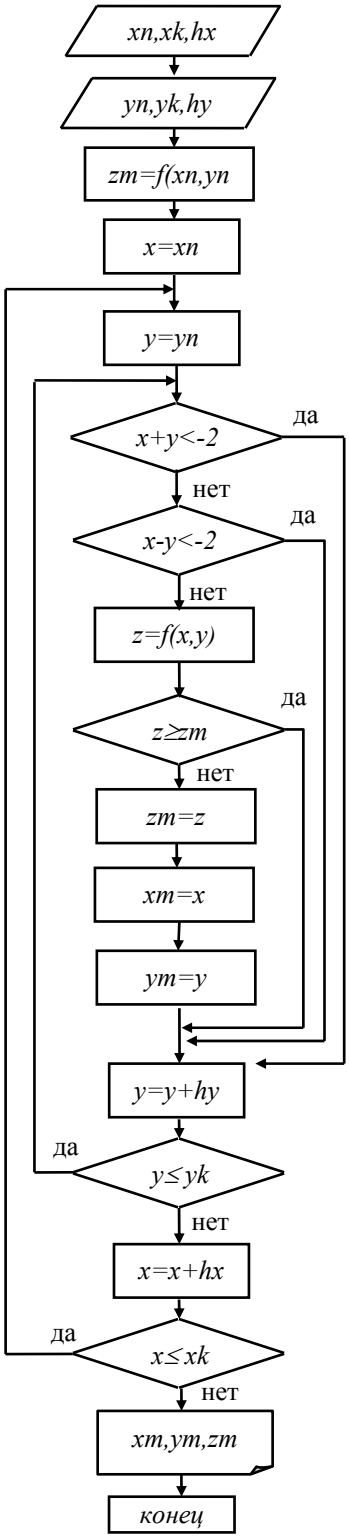


Рис. 10. Иллюстрация к примеру 4

В данном случае область ограничена сверху осью абсцисс (*y*=0), а снизу – двумя прямыми *y=x-2* и *y=-x-2*. Пусть *z=2x+3y-2xy; hx =0,2*; *hy = 0,1*. Условие ограничения снизу имеет вид: $y \geq x-2$ и $y \geq -x-2$. Минимальное значение функции *z* и соответствующие значения аргументов будут расположены в ячейках *zm, xm, ym*, причем в начале программы идет присвоение *zm = f(xn,yn),* где *xn = -2, yn = 0*. Далее организуется двумерный цикл, в котором *x* изменяется от *xn = -2 до xk = 2*, а *y* изменяется от *yn = 0* до *yk = -2,* причем *hx =0,2; hy = -0,1* (отрицательный шаг). Внутри тела цикла организован алгоритм поиска минимального значения функции при условии попадания аргументов в заданную олбласть.

Блок-схема



Программа на Фортране

```
      WRITE(*,*)'Введите xn,xk,hx'
      READ(*,*) xn,xk,hx
      WRITE(*,*)'Введите yn,yk,hy'
      READ(*,*) yn,yk,hy
      zm=f(xn,yn)
      x=xn
10    y=yn
5     IF (x+y.LE.-2) GOTO 1
      IF (x-y.LE.-2) GOTO 1
      z=f(x,y)
      IF (z.GE.zm) GOTO 1
      zm=z
      xm=x
      ym=y
1     y=y+hy
      IF (y.LE.yk) GOTO 5
      x=x+hx
      IF (x.LE.xk) GOTO 10
      WRITE(*,*) xm,ym,zm
      END
```

3.5.Программирование задач на одномерные и двумерные массивы. Введение

В предыдущем разделе были рассмотрены основные принципы работы с простыми переменными, характеризующимися тем, что имеют определенное имя и занимают в памяти ЭВМ одну условную ячейку.

В действительности число элементарных ячеек - бит, занимаемых переменной, зависит от ее типа (целая, вещественная, расширенная целая, расширенная вещественная) и от разрядности ЭВМ. Но для простоты будем считать, что простая переменная занимает в памяти одну условную поименованную ячейку.

В отличие от простых индексированные переменные занимают в памяти ЭВМ несколько ячеек, имеющих одно и то же имя, но разные номера (индексы). Индексом может быть целое положительное число либо переменная целого типа (имя которой начинается с букв *i, j, k, l, m, n*). При изображении индексированной переменной на блок-схеме индексы записываются подстрочным шрифтом, например, x_i, a_{12} и т.д. В программе индексы записываются после имени переменной в скобках, если имеется два индекса, то они разделяются запятой, например, *x(i), a*(1,2) и т.д.

Совокупность индексированных переменных, имеющих одно имя, называется массивом. В зависимости от наличия у индексированной переменной одного или двух индексов, различают одномерный или двумерный массивы. На рисунке приведен пример организации в памяти ЭВМ простых переменных и массивов.

Память компьютера не безгранична, поэтому в отличие от простых переменных при работе с массивами необходимо прежде всего зарезервировать нужное число ячеек памяти. Для каждого массива это число определяется его размерностью - общим количеством элементов. Описание оператора, осуществляющего резервирование места в памяти, а также других необходимых для работы с массивами операторов, приводится далее.

Простые переменные

ira	v1	y

Одномерный массив x(5)

x1	x2	x3	x4	x5

Двумерный массив a(4,4)

a11	a12	a13	a14
a21	a22	a23	a24
a31	a32	a33	a34
a41	a42	a43	a44

Операторы Фортрана при работе с массивами

Отметим, что все рассмотренные ранее операторы, применяемые при работе с простыми переменными, годятся и для массивов. В то же время необходимо указать на наличие специальных операторов, без применения которых работа с индексированными переменными либо невозможна, либо затруднительна.

Оператор описания массива

Структура оператора:

DIMENSION a(4,4), x(5)

Служит для отведения необходимого места в памяти ЭВМ для работы с массивами. В данном случае отводится 16 ячеек в памяти ЭВМ для двумерного массива *a*(4,4), содержащего 4 строки и 4 столбца, а также 5 ячеек для одномерного массива *x*(5). Оператор всегда ставится в начале программы.

Оператор организации цикла

Ранее рассматривалась организация циклического алгоритма с помощью операторов присваивания и условного перехода. При работе с массивами постоянно приходится сталкиваться с циклами. Особенностью является наличие целой переменной цикла (обычно приходится перебирать индексы), целых параметров цикла и во многих случаях - единичного шага. В этом случае целесообразно применять специальный оператор DO - END DO. Его структура:

DO i=N1,N2,N3
тело цикла по i
END DO

Здесь *i* - переменная цикла, *N1* - начальное значение переменной цикла (обычно единица), *N2* - конечное значение переменной цикла (наибольшее зна-

чение *i*), *N3* - шаг (поскольку шаг во многих случаях равен единице, его можно опускать).

При работе с двумерным массивом, как правило, используется двойной цикл. При этом будем использовать переменную *i* для перебора строк и переменную *j* для перебора столбцов:

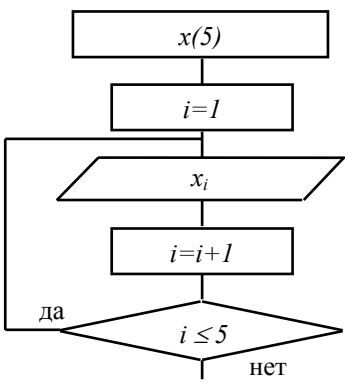
```
DO i=N1,N2,N3
DO j=M1,M2,M3
тело цикла по j        ] тело цикла по i
END DO
END DO
```

В качестве тела цикла могут выступать различные алгоритмы: ввод и вывод массивов, суммирование, нахождение количества и произведения элементов массива, максимального и минимального элементов и др. Базовые алгоритмы, необходимые для решения задач на одномерные и двумерные массивы, рассмотрены далее.

Ввод и вывод массивов

После того, как отработает оператор DIMENSION, в памяти ЭВМ будет отведено необходимое место для массива. Это - как строительство нового дома: уже есть комнаты, но еще нет жильцов. Для «заселения» жильцов - заполнения пустых ячеек исходными данными - необходимо использование циклического алгоритма. Телом цикла в данном случае является оператор ввода числовых данных с клавиатуры. Пусть требуется ввести в память ЭВМ числовые данные массива *x(5)*. В этом случае фрагмент блок-схемы и программы имеет следующий вид:

Блок-схема

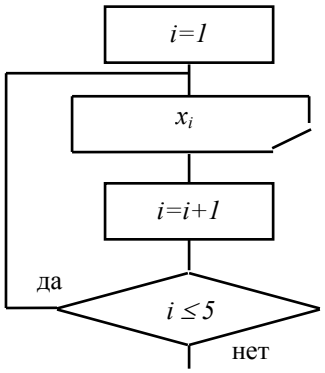


Программа на Фортране

```
WRITE(*,*)'Массив x(5)'
DO i=1,5
READ(*,*) x(i)
END DO
```

Аналогичную структуру имеет блок-схема и программа вывода массива, только в этом случае телом цикла служит блок (оператор) вывода:

Блок-схема

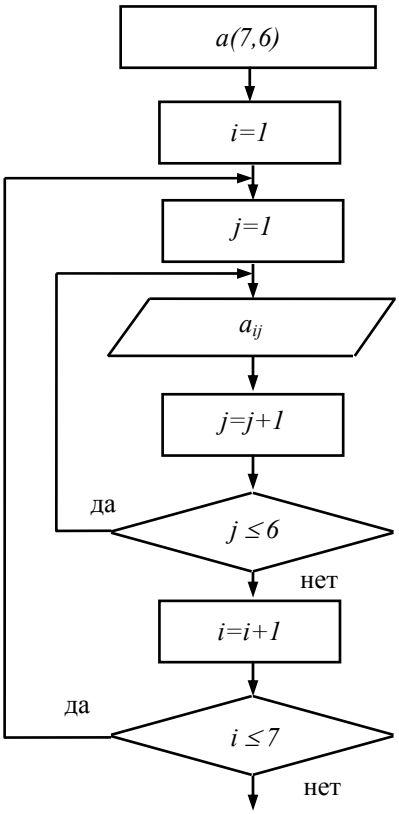


Программа на Фортране

```
WRITE(*,*)’Массив x(5)’
DO i=1,5
WRITE(*,*) x(i)
END DO
```

Для ввода и вывода двумерных массивов используется двумерный цикл. Пусть требуется ввести в память ЭВМ матрицу $a(7,6)$. Фрагмент блок-схемы и программы имеет вид:

Блок-схема



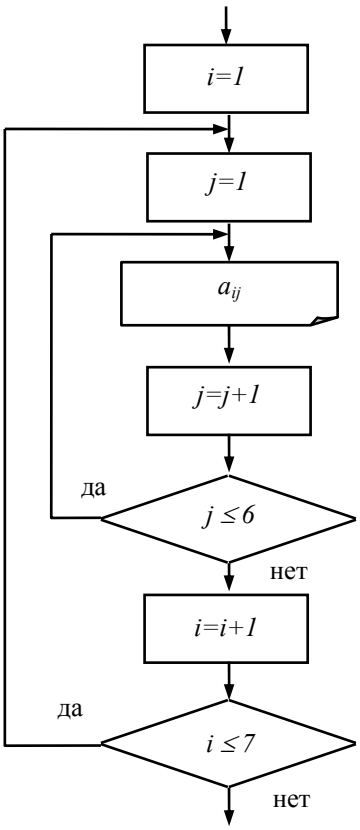
Программа на Фортране

```
DIMENSION a(7,6)
WRITE(*,*)’Введите матрицу a(7,6)’
DO i=1,7
READ(*,*) (a(i,j),j=1,6)
END DO
```

Примечание. В данном случае используется конструкция, называемая «неявным циклом». Неявный цикл по переменной *j* позволяет осуществлять ввод матрицы по строкам (числа внутри строки вводятся через запятую, по окончании ввода строки следует нажать *«Enter»)*.

Аналогично выглядит алгоритм вывода матрицы *a*(7,6) на печать. В этом случае также используется двумерный цикл, а телом внутреннего цикла служит блок (оператор) вывода. В программе также следует использовать «неявный цикл» по столбцам. В этом случае матрица выводится в привычной для нас форме, а фрагмент блок-схемы и программы имеет следующий вид:

Блок-схема



Программа на Фортране

```
WRITE(*,*)’   Матрица a(7,6)’
DO i=1,7
WRITE (*,*) (a(i,j),j=1,6)
END DO
```

Суммирование, нахождение произведения и количества элементов массива

Рассмотрим задачу суммирования элементов массива. Пусть дан одномерный массив *x(5)* и требуется найти сумму его элементов. Предварительно мы зарезервировали место в памяти ЭВМ и осуществили ввод данных в ячейки массива, например, *x*{1,3,-2,0,5}. Для суммирования нам понадобится еще одна ячейка - простая переменная *s*, которую мы должны положить равной нулю.

Таким образом, перед началом цикла суммирования в памяти ЭВМ будет следующая картина:

0

1	3	-2	0	5

Внутри тела цикла необходимо поставить оператор присваивания *s=s+x(i)*. В этом случае работа алгоритма в динамике имеет вид:

i	1	2	3	4	5
s	1	4	2	2	7

Таким образом, по окончании цикла в ячейке *s* содержится сумма элементов массива *x*.

Усложним задачу. Пусть теперь требуется найти сумму положительных элементов массива. В этом случае в тело цикла необходимо дополнительно вставить блок разветвления, определяющий условие положительности элемента, а работа алгоритма имеет вид:

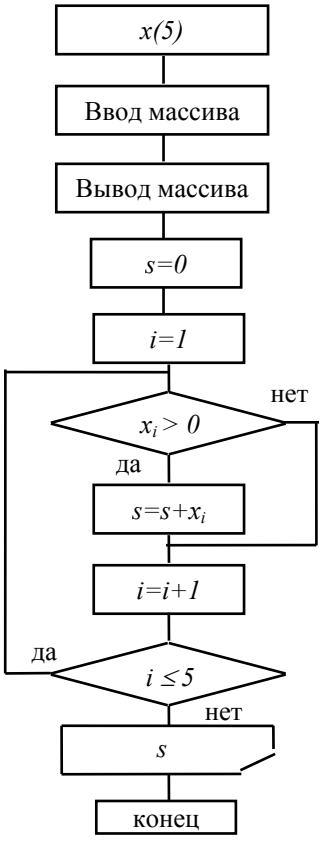
i	1	2	3	4	5
s	1	4	4	4	9

Аналогично устроена работа алгоритмов по нахождению произведения и количества элементов массива.

Для нахождения произведения необходимо ввести дополнительную ячейку *p*, в которую до начала цикла записать с помощью оператора присваивания единицу: *p*=1. Внутри тела цикла должен быть блок нахождения произведения *p=p*x(i)*. По окончании цикла в ячейке *p* содержится произведение элементов массива. Для нахождения количества элементов (например, положительных) необходимо до цикла записать в ячейку *k* ноль: *k*=0. В теле цикла, наряду с условием положительности, содержится блок *k=k*+1, подсчитывающий число положительных элементов.

Рассмотрим полностью решение задачи о нахождении суммы положительных элементов массива *x*(5) (фрагменты блок-схем ввода и вывода массива даны в сокращении).

Блок-схема



Программа на Фортране

```
DIMENSION x(5)
WRITE(*,*)'Введите массив x(5)'
DO i=1,5
    READ(*,*) x(i)
END DO
WRITE(*,*)'Массив x(5)'
DO i=1,5
    WRITE(*,*) x(i)
END DO
s=0
DO i=1,5
    IF(x(i).GT.0) THEN
          s=s+x(i)
    END IF
END DO
WRITE(*,*) 's=',s
END
```

Перестановка элементов массива

Для перестановки местами элементов массива необходимо ввести промежуточную ячейку *«c»* и выполнить действия, аналогичные замене старого телевизора на новый. Если Вы приобрели новый телевизор (в коробке) и Вам необходимо установить его на тумбочку, а старый - в коробку, то вам понадобится стол (ячейка *«c»*).

То же самое нужно сделать, если нужно поменять местами, например, элементы x_1 и x_5 массива. Последовательность действий представлена на рис. 10.

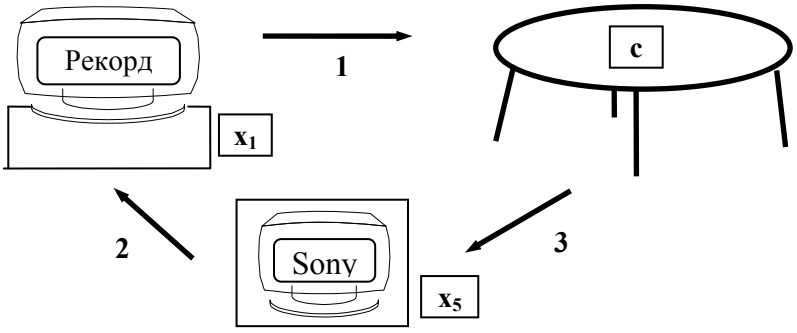


Рис. 11. Иллюстрация алгоритма перестановки элементов

Для перестановки местами элементов массива необходимо ввести промежуточную ячейку *c*. Если нужно поменять местами, например, элементы x_1 и x_5 массива, то выполняется следующая последовательность действий:

```
c = x(1)
x(1) = x(5)
x(5) = c
```

Во многих задачах требуется поменять местами максимальный (минимальный) элемент массива с другим элементом. В этом случае при нахождении максимума (минимума) запоминают его индекс, а затем используют его в алгоритме замены.

Нахождение максимального (минимального) элемента массива

Пусть требуется найти максимальный элемент массива *x*(5). Для этого необходимо использовать цикл и алгоритм «взвешивания». Понадобится также дополнительная ячейка *xmax*, фиксирующая максимальный элемент и ячейка *imax*, фиксирующая его индекс. Последняя ячейка нужна только в том случае, если в дальнейшем потребуется менять местами максимальный элемент массива с другим.

Алгоритм нахождения максимального элемента сводится к следующему (рис. 12). Представьте, что у Вас есть мешок с яблоками (массив *x*(5)), из которого нужно выбрать яблоко максимального веса. В вашем распоряжении есть двуплечные весы без гирь. Левая чаша весов - ячейка *xmax*, где должен оказаться максимальный элемент. Правая чаша весов предназначена для текущего элемента массива *xi* .

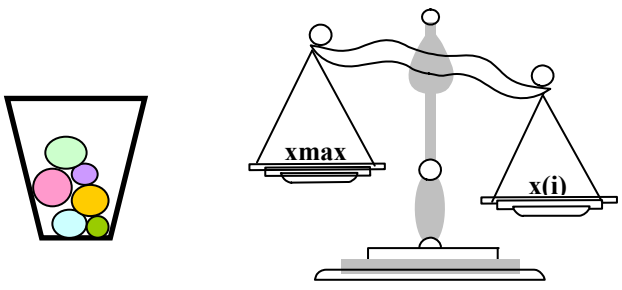


Рис. 12. Иллюстрация алгоритма нахождения максимального элемента массива

До начала цикла взвешивания нужно в ячейку *xmax* записать очень маленькое число, заведомо меньшее всех элементов массива (например, -1000). Затем внутри тела цикла для каждого элемента массива осуществляется «взвешивание» - сравнение с содержимым ячейки *xmax*. Если перевешивает левая чаша весов, то нужно перейти к следующему элементу массива.

Если же перевесила правая чаша весов, то значит, что текущий элемент больше содержимого ячейки *xmax*, и его нужно переложить на левую чашу *(xmax=x(i))*. Кроме того, в случае необходимости здесь нужно зафиксировать индекс найденного «более тяжелого» элемента *(imax=i)*. Таким образом, по окончании цикла ячейка xmax будет содержать максимальный элемент массива, а ячейка *imax* - его номер (индекс).

Рассмотрим, как изменяются ячейки памяти ЭВМ при нахождении максимального элемента массива *x*(5). Перед началом цикла «взвешивания» в памяти ЭВМ будет следующая картина:

xmax	-1000	*imax*			1	3	-2	0	5

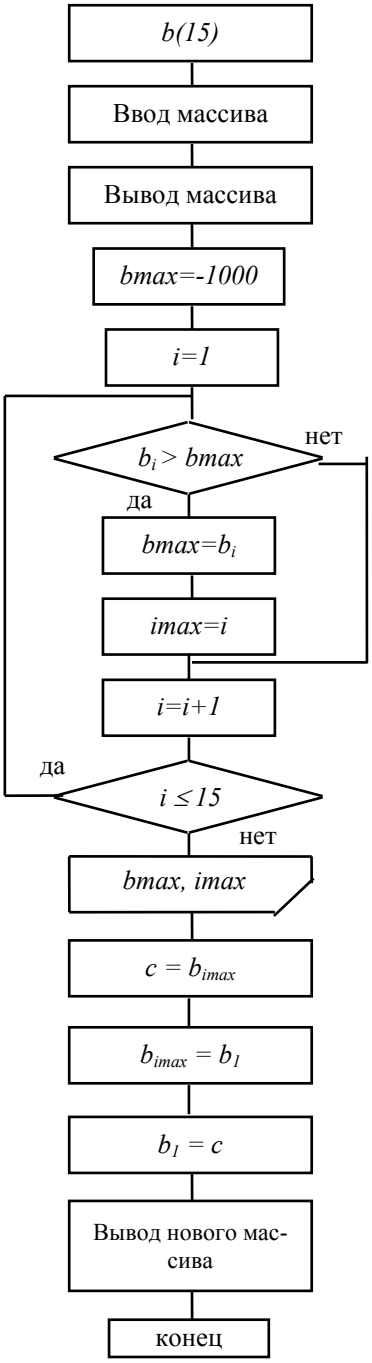
Работа алгоритма в динамике имеет вид:

i	1	2	3	4	5
xmax	1	3	3	3	5
imax	1	2	2	2	5

Аналогично работает алгоритм нахождения минимального элемента массива. В этом случае вводим ячейки *xmin* и *imin*. До начала цикла взвешивания в ячейку *xmin* записывается очень большое число (например, 1000). При взвешивании проверяется условие *x(i)<xmin*. Если оно выполняется, то нужно «переложить» элемент *(xmin= x(i))* и, если необходимо, зафиксировать его индекс (imin=i). Если условие не выполняется, то нужно перейти к следующему элементу. По окончании цикла ячейка *xmin* будет содержать минимальный элемент массива, а ячейка *imin* - его индекс.

Пример. В одномерном массиве *b(15)* найти максимальный элемент и поменять его местами с первым.

Блок-схема

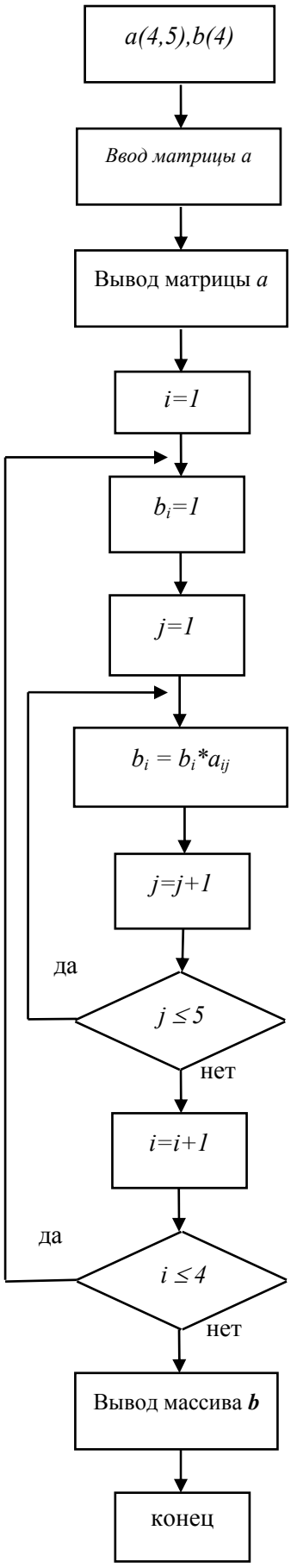


Программа на Фортране

```
DIMENSION b(15)
WRITE(*,*)’Введите массив b(15)’
DO i=1,15
      READ(*,*) b(i)
END DO
WRITE(*,*)’Исходный массив b(15)’
DO i=1,15
      WRITE(*,*) b(i)
END DO
bmax=-1000
DO i=1,15
      IF(b(i).GT.bmax) THEN
            bmax=b(i)
            imax=i
       END IF
END DO
WRITE(*,*) ’bm=’,bmax,’im=’,imax
c=b(imax)
b(imax)=b(1)
b(1)=c
WRITE(*,*)’Новый массив b(15)’
DO i=1,15
      WRITE(*,*) b(i)
END DO
END
```

Пример 1. Дана матрица *a* из четырех строк и пяти столбцов. Для каждой строки вычислить произведение ее элементов. Из полученных произведений образовать новый массив *b* и вывести его на печать. Результирующий массив *b* будет содержать четыре элемента, причем его элементы вычисляются по формуле: $b_i = \prod_{j=1}^{5} a_{ij}, i = 1,...,4,$ где a_{ij} – элемент матрицы *a*.

Блок-схема

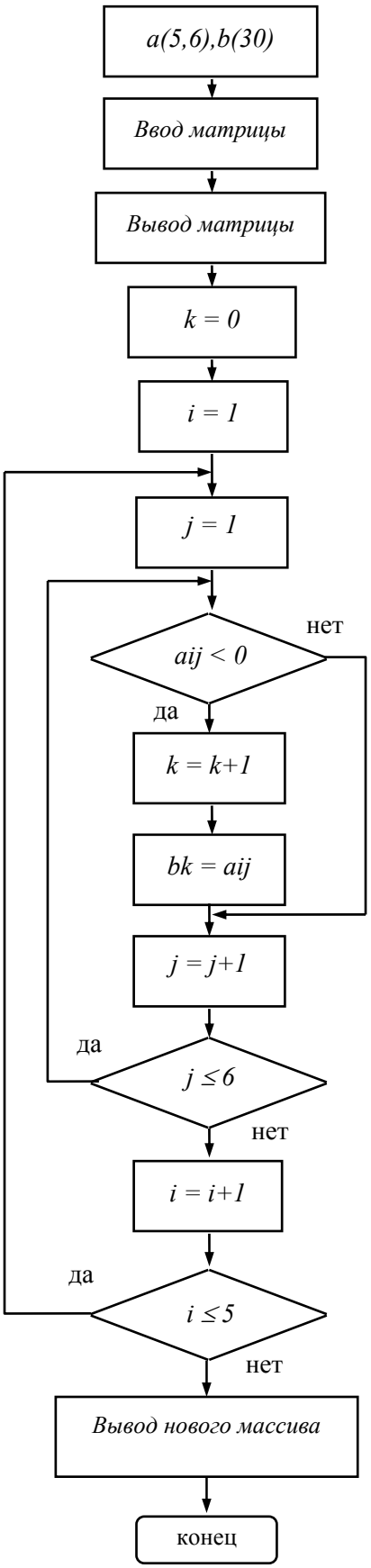


Программа на Фортране

```
DIMENSION a(4,5),b(4)
WRITE(*,*)’Введите матрицу a(4,5)’
DO i=1,4
READ(*,*) (a(i,j),j=1,5)
END DO
WRITE(*,*)((a(i,j),j=1,5),i=1,4)
DO i=1,4
b(i)=1
DO j=1,5
b(i)=b(i)*a(i,j)
END DO
END DO
WRITE(*,*)(b(i),i=1,4)
END
```

Пример 2. Сформировать одномерный массив *b* из отрицательных элементов матрицы *a*(5,6).

Блок-схема



Программа на Фортране

```
DIMENSION a(5,6),b(30)
WRITE(*,*)'Введите м-цу a(5,6)'
DO i=1,5
READ(*,*) (a(i,j),j=1,6)
END DO
WRITE(*,*)((a(i,j),j=1,6),i=1,5)
k=0
DO i=1,5
DO j=1,6
IF(a(i,j).LT.0) THEN
k=k+1
b(k)=a(i,j)
END IF
END DO
END DO
WRITE(*,*)k,(b(i),i=1,k)
END
```

4. ОПИСАНИЕ ЛАБОРАТОРНЫХ РАБОТ

Лабораторные работы выполняются в конце семестра перед сдачей зачета (экзамена). При выполнении лабораторных работ студенты закрепляют навыки, полученные в течение семестра (лекции, консультации, самостоятельная работа).

Состав лабораторных работ

№ п.п.	Семестр	Название лабораторной работы	Кол-во часов
1	1	Работа в среде Windows	2
2	1	Программирование задач на простые переменные	3
3	1	Программирование задач на индексированные переменные	3

Порядок выполнения лабораторной работы № 1

1. С помощью графического редактора *Paint* нарисовать рисунок (по вариантам) и сохранить информацию в отдельном файле студенческого каталога.
2. С помощью текстового редактора *Word* набрать и отформатировать текстовый документ (по вариантам). Сохранить документ в отдельном файле студенческого каталога.
3. С помощью Буфера обмена скопировать рисунок в текстовый файл.
4. Сохранить документ в отдельном файле студенческого каталога.

Порядок выполнения лабораторной работы № 2

Студенты выполняют по вариантам задания №1, №2 на ЭВМ в соответствии со следующим порядком:

1. постановка задачи;
2. составление алгоритма (блок-схемы);
3. написание программы на языке Фортран;
4. набор программы, используя редактор Far, сохранение на диске в студенческом каталоге исходного модуля (расширение .for);
5. трансляция и компоновка программы. При этом в текущем каталоге будут созданы объектный модуль (расширение .obj) и готовая к выполне-

нию программа (расширение .exe). Если при трансляции будут обнаружены ошибки, необходимо их исправить в исходном модуле, сохранить его на диске и вновь приступить к трансляции;

6. запуск готовой программы, ввод исходных данных;
7. получение и анализ результатов.

Порядок выполнения лабораторной работы № 3

Аналогичен порядку выполнения лабораторной работы №2 (задания №3, №4).

5. ЗАДАНИЯ И ВАРИАНТЫ ДЛЯ КОНТРОЛЬНЫХ И ЛАБОРАТОРНЫХ РАБОТ

Контрольная работа оформляется в тетради и содержит четыре задания. При выполнении каждого задания необходимо наличие следующих элементов:

- титульный лист;
- постановка задачи;
- блок-схема;
- программа на языке Фортран;
- результаты расчетов (для задания №2).

Задания выполняются по вариантам, номер которого соответствует двум последним цифрам зачетной книжки студента. Каждое задание содержит 15 задач, из которых выбирается одна в соответствии с номером варианта.

№ варианта	Задание 1	Задание 2	Задание 3	Задание 4
00	7	9	11	13
01	1	1	1	1
02	2	2	2	2
03	3	3	3	3
04	4	4	4	4
05	5	5	5	5
06	6	6	6	6
07	7	7	7	7
08	8	8	8	8
09	9	9	9	9
10	10	10	10	10

№ варианта	Задание 1	Задание 2	Задание 3	Задание 4
11	11	11	11	11
12	12	12	12	12
13	13	13	13	13
14	14	14	14	14
15	15	15	15	15
16	1	3	5	15
17	2	4	6	8
18	10	12	14	1
19	3	6	9	12
20	15	12	9	6
21	1	4	7	9
22	9	7	4	1
23	12	15	11	14
24	14	11	15	12
25	2	5	8	11
26	14	1	4	7
27	11	8	5	2
28	7	4	1	14
29	3	7	11	15
30	2	6	10	14
31	1	5	9	13
32	4	8	12	1
33	15	11	7	3
34	14	10	6	2
35	13	9	5	1
36	1	12	8	4
37	2	10	3	11
38	4	12	5	13
39	6	14	7	15
40	8	1	9	2
41	11	3	10	2
42	13	5	12	4
43	15	7	14	6
44	2	9	1	8
45	3	11	4	12
46	5	13	6	14
47	7	15	8	1
48	9	2	10	3
49	12	4	11	3

№ варианта	Задание 1	Задание 2	Задание 3	Задание 4
50	14	6	13	5
51	1	8	15	7
52	3	10	2	9
53	10	3	11	4
54	12	5	13	6
55	14	7	15	8
56	1	9	2	10
57	4	11	3	10
58	6	13	5	12
59	8	15	7	14
60	10	2	9	1
61	11	4	12	5
62	13	6	14	7
63	15	8	1	9
64	2	10	4	11
65	5	12	4	11
66	7	14	6	13
67	9	1	8	15
68	11	4	10	2
69	1	2	3	4
70	5	6	7	8
71	9	10	11	12
72	13	14	15	1
73	2	3	4	5
74	6	7	8	9
75	10	11	12	13
76	14	15	1	2
77	3	4	5	6
78	7	8	9	10
79	11	12	13	14
80	15	1	2	3
81	4	5	6	7
82	8	9	10	11
83	12	13	14	15
84	15	14	13	12
85	11	10	9	8
86	7	6	5	4
87	3	2	1	15
88	14	13	12	11
89	10	9	8	7

№ варианта	Задание 1	Задание 2	Задание 3	Задание 4
90	6	5	4	3
91	2	1	15	14
92	13	12	11	10
93	9	8	7	6
94	5	4	3	2
95	2	15	14	13
96	12	11	10	9
97	8	7	6	5
98	4	3	2	1
99	14	12	10	8

Задание №1.

Составить блок-схему и программу для вычисления Y и Z по заданным формулам

1. $Y = 67\text{rcos}\, x + a^2$, $Z = \ln(x^3 + \cos a)$ при a=0,75, x=0,14.

2. $Y = \lg(\frac{x}{2}) + a^3$, $Z = \text{tg}(e^x + \cos a)$ при a=0,34, x=0,02.

3. $Y = \sqrt{x^2 + a}$, $Z = \arcsin(x^3 - a)$ при a=0,01, x=0,12.

4. $Y = \ln(1{,}5x) + a^4$, $Z = \text{arctg}(\cos\frac{a}{x^2})$ при a=2,5, x=3,11.

5. $Y = \frac{\sqrt{x^{1.5}}}{a^2}$, Z=cos(3,56(x+a)) при a=-5,1, x=4,78.

6. $Y = \cos(x^3 + a^3)$, $Z = \text{arctg}\sqrt{2x + a}$ при a=2,48, x=0,21.

7. $Y = 2e^{4x} + \text{arctg}(\frac{x}{a})$, $Z = \cos x^3 + \sin^2 x$ при a=2,8, x=3,29.

8. $Y = |\sin(x - a^2)|^4$, $Z = e^{2x} + 67\text{rcos}(2x + a)$ при a=0,35, x=0,21.

9. $Y = \ln|\sin(x+a)|$, $Z = \text{tg}(xe^x)^2$ при a=-3,4, x=2,75.

10. $Y = \ln 2x^3 + a^{3/2}$, $Z = 3{,}7\ \text{tg}^2 2x$ при a=2,53, x=0,7.

11. $Y = \frac{\sqrt{xa^{1,2}}}{a^3}$, Z= ctg(2,6(x + a))

12. Y= ctg(x^3+ a^3), $Z = \arcsin\sqrt[6]{x^2 + a^2}$

при a=-5,1, x=4,78. при a=2,48, x=0,11.

13. $Y = 5e^{2x} + \mathrm{tg}(\frac{x}{a})$, $Z = \cos a^3 + \sin^2 x$ 14. $Y = |\sin(x-a^2)|^4$, Z=$\arccos^2|(2x+ a)|$

при a=6,8, x=3,2. при a=0,3, x=0,2.

15. $Y = \lg|\cos(x-a)|$, $Z = \mathrm{ctg}(ae^x)^2$

при a=-3,6, x=0,75.

Задание № 2

Составить блок-схему и программу для вычисления таблицы значений функции $U(x,y)$ при изменении значений аргументов x и y в заданных пределах и с заданным шагом.

1

$$U = \begin{cases} xe^{-yx}, & \text{если} \quad x + y < 0; \\ x\sin x, & \text{если} \quad 0 \le x + y < 3; \\ y^x + \cos^3 x, & \text{если} \quad x + y \ge 3. \end{cases}$$

xn = 0,5; xk = 1,6; hx= 0,5;
yn = -1,5; yk= 1,6; hy = 1,5.

2

$$U = \begin{cases} ye^x, & \text{если} \quad x - y < 0; \\ \ln(x^4 + 1), & \text{если} \quad 0 \le x - y < 3; \\ x^3 + y^3, & \text{если} \quad x - y \ge 3. \end{cases}$$

xn = 0,5; xk = 1,6; hx= 0.5;
yn = 0,5; yk = 1,6; hy = 0.5.

3

$$U = \begin{cases} x\sin^2 y, & \text{если} \quad xy \le 1; \\ \mathrm{ctg}x, & \text{если} \quad 1 < xy < 3; \\ \ln^2(x + y), & \text{если} \quad xy \ge 3. \end{cases}$$

xn = 0,5; xk = 1,6; hx = 0.5;
yn = 1; yk = 2,1; hy = 0.5.

4

$$U = \begin{cases} \mathrm{tg}x, & \text{если} \quad x^2 y < 0; \\ x\sin^2 y, & \text{если} \quad 0 \le x^2 y < 5; \\ |x - y|, & \text{если} \quad x^2 y \ge 5. \end{cases}$$

xn = 0,5; xk = 1,6; hx = 0,5;
yn = -0,5; yk = 2,6; hy = 1,5.

5

$$U = \begin{cases} \cos e, & \text{если} \quad |xy| < 1; \\ x^2 + y^2, & \text{если} \quad 1 \le |xy| < 3; \\ \sin x, & \text{если} \quad |xy| \ge 3. \end{cases}$$

xn = -0,5; xk = 1,6; hx = 1;
yn = 1; yk = 2,1; hy = 0,5.

6

$$U = \begin{cases} 2^{x+y}, & \text{если} \quad xy^3 < 0; \\ x\cos y, & \text{если} \quad 0 \le xy^3 < 3; \\ 2y - x, & \text{если} \quad xy^3 \ge 3. \end{cases}$$

xn = 0,5; xk = 1,6; hx = 0,5;
yn = -0,5; yk = 1,6; hy = 1.

7

$$U=\begin{cases} x+\sin y, & \text{если} \quad x-y<0; \\ \operatorname{tg}^2 xy, & \text{если} \quad 0\le x-y<3; \\ \ln|x+y|, & \text{если} \quad x-y\ge 3. \end{cases}$$

xn = -0,5; xk = 3,6; hx = 2;
yn = 0; yk = 2,1; hy = 1.

8

$$U=\begin{cases} \operatorname{ctg}(x+y), & \text{если} \quad |x^2+y^2|<1; \\ x^2+y^2, & \text{если} \quad 1\le |x^2+y^2|<4; \\ x-y, & \text{если} \quad |x^2+y^2|\ge 4 \end{cases}$$

xn = 0,5; xk = 1,6; hx = 0,5;
yn = -0,5; yk = 1,6; hy = 1.

9

$$U=\begin{cases} x^2 y, & \text{если} \quad x+y<0; \\ x^{-y}, & \text{если} \quad 0\le x+y<4; \\ \sin(xy), & \text{если} \quad x+y\ge 4. \end{cases}$$

xn = -0,5; xk = 3,6; hx = 2;
yn = 0, yk = 2,1; hy = 1.

10

$$U=\begin{cases} \sin(x+y)+x, & \text{если} \quad x^7 y<0; \\ x^4+\ln(x^7+y^3), & \text{если} \quad 0\le x^7 y<5; \\ \sin y, & \text{если} \quad x^7 y\ge 5. \end{cases}$$

xn = 0,5; xk = 1,6; hx = 0,5;
yn = -0,5; yk = 1,6; hy = 1.

11

$$U=\begin{cases} e^{-x}, & \text{если} \quad x+y<0; \\ \cos(\sin x), & \text{если} \quad 0\le x+y<3; \\ y^x+\operatorname{ctg}^3 x, & \text{если} \quad x+y\ge 3. \end{cases}$$

xn = 0,5; xk = 1,6; hx = 0.5;
yn = -1,5; yk = 1,6; hy = 1.5.

12

$$U=\begin{cases} y\lg|x|, & \text{если} \quad x-y<0; \\ \ln(x^4+3), & \text{если} \quad 0\le x-y<3; \\ x^3+y\operatorname{tg}x, & \text{если} \quad x-y\ge 3. \end{cases}$$

xn = 0,5; xk = 1,6; hx = 0,5;
yn = 0,5; yk = 1,6; hy = 0.5.

13

$$U=\begin{cases} \sin^2 xy, & \text{если} \quad xy\le 1; \\ \operatorname{ctg}x\cdot e^y, & \text{если} \quad 1<xy<3; \\ \lg^2(x+y), & \text{если} \quad xy\ge 3. \end{cases}$$

xn = 0,5; xk = 1,6; hx = 0,5;
yn = 1; yk = 2,1; hy = 0.5.

14

$$U=\begin{cases} \cos x, & \text{если} \quad x^2 y<0; \\ y\sin^3 y, & \text{если} \quad 0\le x^2 y<5; \\ \ln|x-y|, & \text{если} \quad x^2 y\ge 5. \end{cases}$$

xn = 0,5; xk = 1,6; hx = 0,5;
yn = -0,5; yk = 2,6; hy = 1,5.

15

$$U=\begin{cases} \operatorname{tg}e, & \text{если} \quad |xy|<1; \\ x^2+y^2, & \text{если} \quad 1\le |xy|<3; \\ \operatorname{ctg}xy, & \text{если} \quad |xy|\ge 3. \end{cases}$$

xn = -0,5; xk = 1,6; hx = 1;
yn = 1; yk = 2,1; hy = 0,5.

Задание № 3

В одномерном массиве $X(15)$ найти:

1. Максимальный из отрицательных элементов и поменять его местами с последним.
2. Сумму отрицательных, количество положительных и произведение ненулевых элементов.
3. Среднее арифметическое элементов, удовлетворяющих условию $\cos x_i < 0$.
4. Минимальный из положительных элементов и количество нулевых.
5. Произведение элементов, удовлетворяющих условию $0 < \operatorname{tg} x_i < 1$, а также сумму положительных.
6. Минимальный из элементов, больших двух, и поменять его местами с первым.
7. Сумму неположительных, произведение неотрицательных элементов, поменять местами первый и последний элементы.
8. Количество элементов, удовлетворяющих условию $-0{,}5 < \sin x_i \le 0$, минимальный элемент.
9. Максимальный и минимальный элементы и поменять их местами.
10. Сумму и произведение элементов и выбрать из них наибольшее.
11. Максимальный из элементов, меньших трех, и поменять его местами с предпоследним.
12. Количество нулевых элементов, сумму положительных, поменять местами второй и десятый элементы.
13. Среднее арифметическое неотрицательных элементов.
14. Максимальный из отрицательных элементов и сумму неположительных.
15. Сумму элементов, удовлетворяющих условию $0{,}5 < \cos x_i \le 1$, максимальный элемент.

Задание №4

1. В матрице $A(7,7)$ найти наибольший из элементов, удовлетворяющих условию $\cos^2 a_{ij} > 0{,}5$, и поменять его местами с первым элементом шестого столбца.
2. Создать новый одномерный массив, состоящий из произведений элементов матрицы $A(6,4)$ по столбцам.
3. Найти строку, содержащую минимальный элемент матрицы $A(5,7)$, и поменять ее местами с четвертой строкой.
4. В строке, содержащей максимальный элемент матрицы $A(6,8)$, подсчитать сумму положительных элементов.
5. Найти сумму отрицательных элементов на побочной диагонали матрицы $A(8,8)$, а также произведение ненулевых элементов в области выше главной диагонали.
6. В квадратной матрице $A(7,7)$ найти число отрицательных элементов в области выше главной и ниже побочной диагонали. Поменять местами минимальный элемент первой строки и последний элемент матрицы.
7. Сформировать одномерный массив B из отрицательных элементов области исходной матрицы $A(9,7)$, лежащей левее шестого столбца. В полученном массиве поменять местами первый и максимальный элементы.
8. В квадратной матрице $A(7,7)$ поменять местами минимальные элементы первой и третьей строк, максимальные элементы главной и побочной диагоналей.
9. Сформировать одномерный массив B из элементов исходной матрицы $A(7,7)$, удовлетворяющих условию $0{,}8 < \mathrm{t}\,g\,(a_{ij}) < 1{,}8$. В полученном массиве найти минимальный из положительных элементов.
10. В квадратной матрице $A(7,7)$ найти среднее арифметическое положительных элементов, произведение ненулевых элементов, а также количество нулей на побочной диагонали.

11. В матрице $A(8,8)$ поменять местами первую и шестую строку. В полученной матрице найти сумму элементов, расположенных ниже побочной диагонали.
12. В матрице $A(7,8)$ найти наибольший из элементов, удовлетворяющих условию $\cos^2 a_{ij} > 0{,}5$, и поменять его местами с первым элементом шестого столбца.
13. Создать новый одномерный массив B из элементов исходной матрицы $A(6,8)$, удовлетворяющих условию $0 \le \cos a_{ij} < 0{,}5$. В полученном массиве поменять местами максимальный и минимальный элементы.
14. В матрице $A(7,7)$ найти произведение ненулевых диагональных элементов, максимальный элемент третьего столбца и минимальный элемент шестой строки.
15. В матрице $A(7,5)$ найти количество нулевых элементов. Поменять местами минимальные элементы второй и пятой строк.

6. МЕТОДИЧЕСКИЕ УКАЗАНИЯ И ПРИМЕРЫ ВЫПОЛНЕНИЯ КОНТРОЛЬНЫХ РАБОТ

Ниже приведены образцы оформления контрольных работ

Задание №1

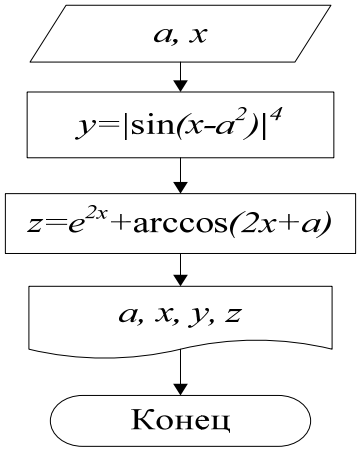
Составить блок-схему и программу для вычисления Y и Z по заданным формулам

$$Y = \left|\sin(x - a^2)\right|^4$$

$$Z = e^{2x} + \arccos(2x + a)$$

при a=0,35, x=0,21

Блок-схема



Программа на Фортране

```
WRITE(*,*) ' введение a и x'
READ (*,*) a ,x
Y=ABS (sin(x-a**2))**4
Z=EXP (2*x)+ACOS (2*x+a)
WRITE (*,*)' a=',a, 'x=',x, 'Y=',Y,
'Z=',Z
END
```

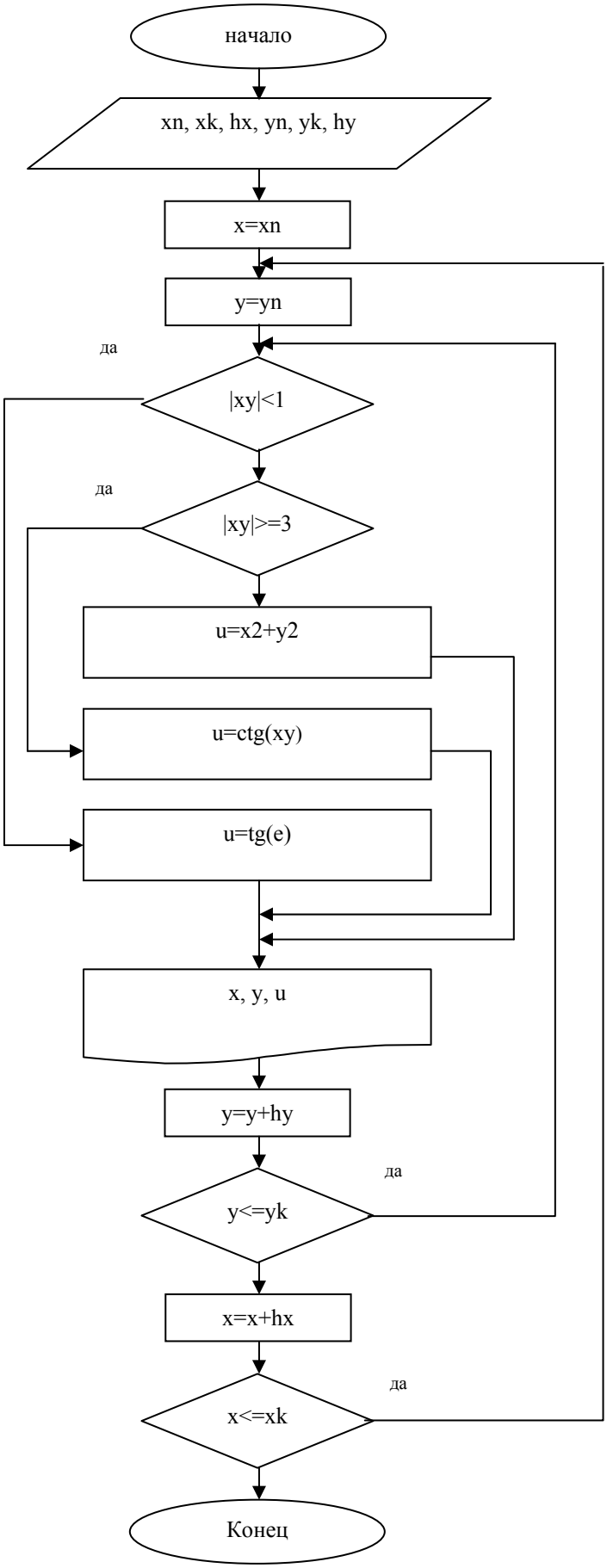
Задание №2

Составить блок-схему и программу для вычисления таблицы значений функции $U(x,y)$ при изменении значений аргументов x и y в заданных пределах и с заданным шагом

$$U = \begin{cases} \operatorname{tg} e, \text{ если } |xy| < 1 \\ x^2 + y^2, \text{ если } 1 \le |xy| < 3 \\ \operatorname{ctg} xy, \text{ если } |xy| \ge 3 \end{cases}$$

xn=-0.5; xk=1.6; hx=1;
yn=1; yk=2.1; hy=0.5

Блок-схема

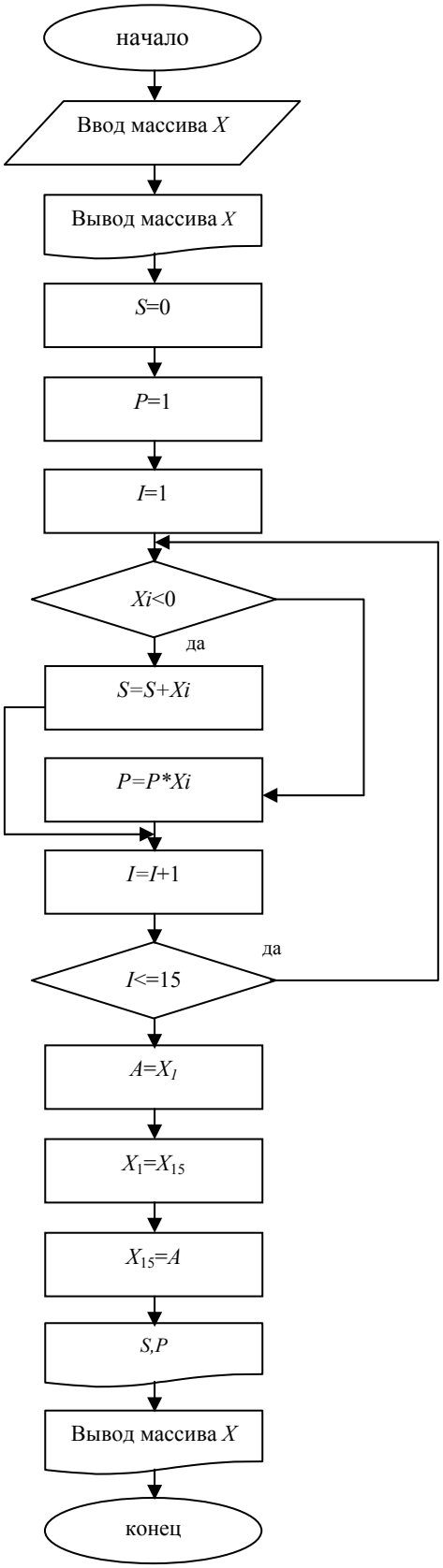


Программа на Фортране

```
WRITE(*,*)’введите
xn,xk,hx,yn,yk,hy’
READ (*,*) xn,xk,hx,yn,yk,hy
DO x=xn,xk,hx
      DO y=yn,yk,hy
            If (ABS(x*y).LT.1)
THEN
        U=TAN (EXP(1))
ELSEIF (ABS(x*y).GE.3) THEN
      U=1/TAN(x*y)
      ELSE
      U=x**2+y**2
      END IF
      WRITE (*,*)’ x=’,x,
‘y=’,y,U=’,U
        END DO
END DO
  END
```

Задание №3

В одномерном массиве X (15) найти сумму неположительных, произведение неотрицательных элементов, поменять местами первый и последний элементы

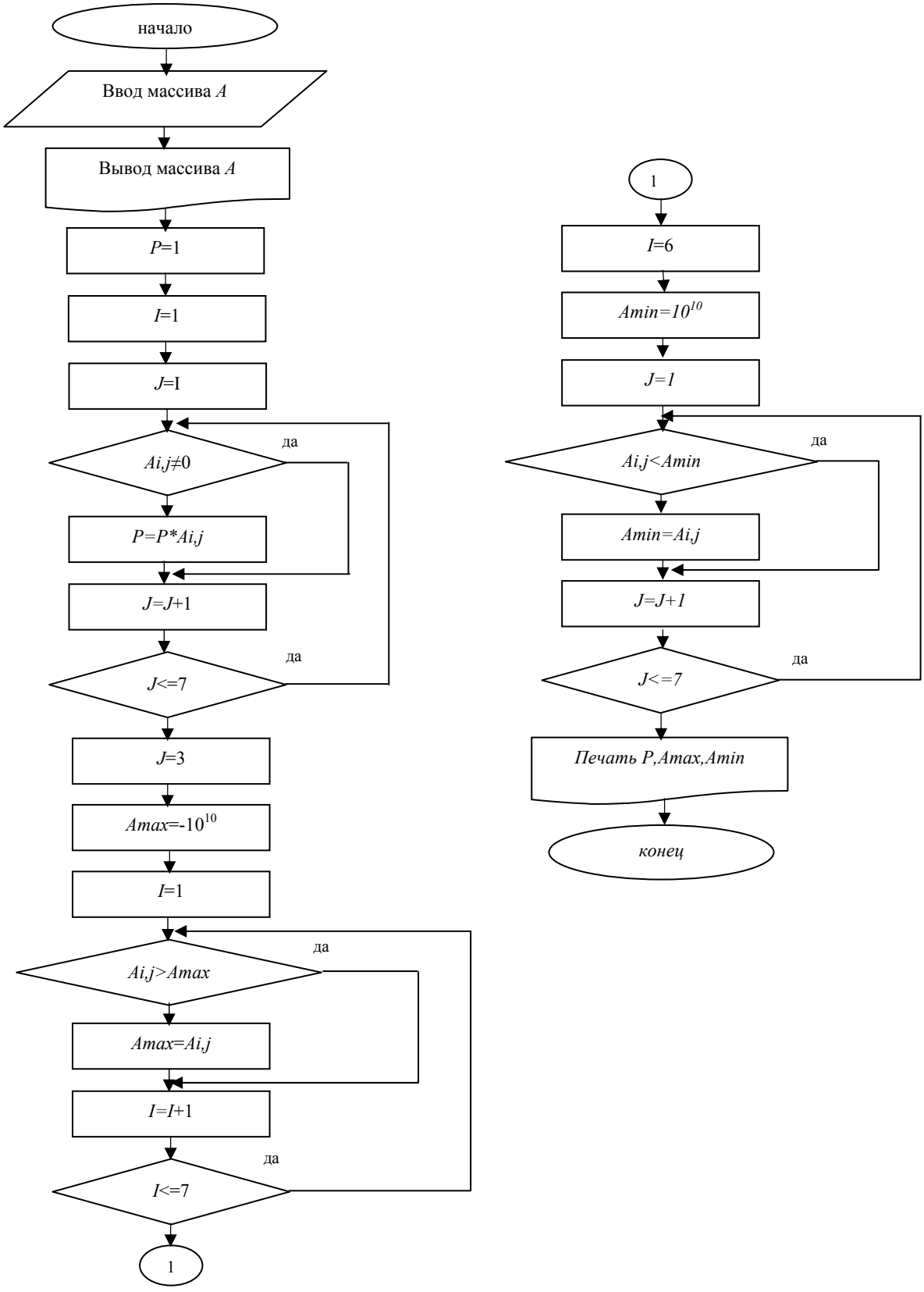



```
DIMENSION x(15)
WRITE(*,*)'введите массив x(15)'
DO i=1,15
        READ(*,*) x(i)
END DO
WRITE(*,*)'массив x(15)'
        DO i=1,15
                WRITE(*,*) x(i)
 END DO
S=0
P=1
DO i=1,15
        IF (x(i).LE.0) THEN
S=S+x(i)
ELSE
        P=P*x(i)
END IF
END DO
a=x(1)
x(1)=x(15)
x(15)=a
WRITE (*,*)' S=',S, 'P=',P, 'новый
массив'
        DO i=1,15
                WRITE(*,*) x(i)
        END DO
END
```

Задание №4

В матрице $A(7,7)$ найти произведение ненулевых диагональных элементов, максимальный элемент третьего столбца и минимальный элемент шестой строки.

БЛОК-СХЕМА



ПРОГРАММА

```
DIMENSION a(7,7)
WRITE(*,*)’введите массив a(7,7)’
DO i=1,7
        READ(*,*) (a(i,j) ,j=1,7)
END DO
WRITE(*,*)’матрица a(7,7)’
        DO i=1,7
                WRITE(*,*) (a(i,j) ,j=1,7)
        END DO
P=1
DO i=1,7
        j=i
        IF (a(i,j).NE.0) THEN
        P=P+a(i,j)
END DO
j=3
amax=-10**10
DO i=1,7
        IF (a(i,j).GT.amax) THEN
        amax=a(i,j)
END DO
i=6
amin=10**10
DO j=1,7
        IF (a(i,j).LT.amin) THEN
        amin=a(i,j)
END DO
WRITE (*,*)’ P=’,P, ‘amax=’, amax, ‘amin=’, amin
END
```

7. КОНТРОЛЬ ЗНАНИЙ СТУДЕНТОВ

Контрольная работа включает в себя три задания. Каждое задание, в свою очередь, включает 15 контрольных тестов, из которых для ответа нужно выбрать 7 в соответствии с номером варианта. Номер варианта соответствует двум последним цифрам номера зачетной книжки. Номера тестов выбираются в соответствии со следующей таблицей.

Таблица вариантов заданий

№ варианта	**Номера тестов**						
	1	***2***	***3***	***4***	***5***	***6***	***7***
00	2	3	6	8	11	13	15
01	4	5	9	10	11	12	14
02	1	2	6	7	9	10	15
03	3	5	7	8	10	13	14
04	1	3	4	6	7	9	11
05	1	4	6	8	10	12	15
06	4	6	7	9	11	13	14
07	1	3	5	7	8	12	14
08	1	2	3	4	5	8	9
09	2	3	5	10	11	12	15
10	2	4	5	8	11	12	15
11	3	6	9	10	11	13	14
12	1	5	7	8	9	10	15
13	3	2	6	7	10	13	14
14	1	3	5	6	7	10	11
15	1	4	6	7	11	12	15
16	3	5	7	8	10	11	14
17	1	2	4	7	9	12	14
18	1	2	3	5	7	8	10
19	2	4	6	10	11	13	15
20	3	5	6	8	10	13	14

№ варианта	Номера тестов						
	1	*2*	*3*	*4*	*5*	*6*	*7*
21	4	6	8	10	11	13	14
22	1	2	5	6	9	12	15
23	3	5	7	9	11	13	14
24	1	3	5	6	8	9	12
25	1	4	5	7	11	13	15
26	3	5	8	9	10	12	14
27	1	3	5	8	9	11	12
28	1	2	3	5	6	7	10
29	2	3	7	12	13	14	15
30	1	3	6	7	10	11	12
31	3	4	7	9	11	13	14
32	1	2	5	7	8	9	11
33	2	3	4	7	10	12	14
34	1	2	5	6	8	9	12
35	1	4	6	7	11	14	15
36	3	5	8	9	10	13	14
37	1	2	5	8	9	11	12
38	1	2	5	7	8	9	13
39	2	3	6	10	11	14	15
40	2	4	5	8	10	13	15
41	4	6	9	10	12	13	14
42	1	2	4	6	9	11	15
43	3	5	7	9	12	13	14
44	1	2	3	6	7	8	11
45	1	4	6	7	10	13	15
46	4	5	7	8	10	13	14
47	1	3	4	7	9	11	14
48	1	2	4	8	9	10	13
49	2	4	5	9	10	12	15
50	2	3	7	8	12	14	15
51	4	5	8	9	10	12	14
52	1	2	4	8	9	11	15
53	3	5	6	8	11	12	14
54	1	3	4	7	8	9	13
55	1	4	5	6	10	12	15
56	3	5	7	9	11	13	14
57	1	3	5	8	10	12	14
58	1	2	3	6	9	14	15
59	2	4	5	9	10	11	15
60	2	3	7	9	10	12	13

№ варианта	Номера тестов						
	1	*2*	*3*	*4*	*5*	*6*	*7*
61	4	5	6	8	9	10	11
62	1	2	6	9	13	14	15
63	3	5	7	8	11	12	13
64	1	3	4	6	7	8	9
65	1	4	6	9	10	13	14
66	3	4	5	6	8	11	12
67	1	3	4	7	9	12	14
68	1	2	3	4	5	6	7
69	9	10	11	12	13	14	15
70	2	5	6	7	10	12	15
71	4	5	7	8	9	12	14
72	1	2	3	7	8	9	10
73	3	5	7	9	11	13	14
74	1	3	4	5	6	9	10
75	1	4	6	9	11	12	15
76	3	5	6	7	8	10	14
77	1	3	6	7	9	12	13
78	1	2	6	7	9	10	15
79	2	3	4	7	10	12	15
80	2	4	6	7	10	12	15
81	3	5	7	9	11	12	14
82	1	2	4	5	9	11	15
83	3	6	8	9	10	12	14
84	1	2	4	5	7	8	11
85	1	3	5	9	10	11	15
86	4	5	6	8	10	12	14
87	1	2	5	8	9	11	14
88	1	2	3	4	6	7	13
89	1	3	7	11	12	13	14
90	2	3	4	5	10	11	15
91	4	5	6	8	10	12	14
92	1	2	7	8	9	11	15
93	3	4	6	8	11	12	14
94	1	2	4	6	8	9	13
95	1	4	6	12	13	14	15
96	4	6	8	10	11	13	14
97	1	3	5	6	7	10	11
98	1	2	3	4	6	8	12
99	2	3	6	7	10	11	13

Образец таблицы для оформления результатов решения тестов

Задание № ___ Вариант № ___

№№	1	2	3	4	5	6	7
№ теста							
№ правильного ответа							

Задание №1. Архитектура ПЭВМ. Программное обеспечение

№	***Вопрос***
1.	В создание первой персональной ЭВМ типа *IBM-PC-XT* не участвовала фирма: а) *Microsoft;* б) *Makintosh;* в) *Intel;* г) Все участвовали
2.	Особенностью персональной ЭВМ является: а) высокая надежность; б) хороший дизайн; в) наличие магнитных дисков; г) все ответы верны
3.	Особенностью персональной ЭВМ не является: а) высокая надежность; б) небольшие размеры; в) пакетный режим; г) все является
4.	В базовую конфигурацию персональной ЭВМ не обязательно входит: а) принтер; б) мышь; в) клавиатура; г) все перечисленное входит
5.	К дополнительным устройствам ПК не относится: а) стример; б) сканер; в) винчестер; г) принтер
6.	1Мбайт - это: а) 210 байт; б) 220 байт; в) 230 байт; г) нет правильного ответа
7.	1Кбайт - это: а) 210 байт; б) 220 байт; в) 230 байт; г) нет правильного ответа
8.	Емкость современного жесткого диска лежит в пределах: а) 40-200 Гбайт; б) 3-20 Мбайт; в) 1-1,4 Мбайт; г) нет правильного ответа
9.	Емкость гибкого диска составляет: а) 1,4 Гбайт; б) 3,2 Мбайт; в) 700 Мбайт; г) нет правильного ответа
10.	В состав системного блока ПК не входит: а) сканер; б) винчестер; в) звуковая карта; г) все вышеперечисленное входит
11.	Для долговременного хранения информации не используется: а) НГМД; б) винчестер; в) оперативная память; г) стример
12.	Программа Norton Commander - это: а) операционная система; б) операционная оболочка; в) графическая интерфейсная среда; г) ничего из перечисленного
13.	В состав системного программного обеспечения не входят: а) системы программирования; б) операционные системы; в) операционные оболочки; г) все вышеперечисленное входит
14.	В состав функциональных пакетов не входят: а) системы управления базами данных; б) бухгалтерский пакеты; в) табличные процессоры; г) все из вышеперечисленных входит
15.	В состав поблемно-ориентированных пакетов входят: а) системы управления базами данных; б) бухгалтерский пакеты; в) табличные процессоры; г) ничего из вышеперечисленного не входит

Задание №2. Программа-оболочка Norton Commander

№	Вопрос
1.	Что включает в себя полное имя файла? а) имя файла и каталога, в котором находится файл; б) имя и расширение файла; в) название диска, последовательность вложенных каталогов до того, в котором находится файл, имя и расширение файла; г) имя и расширение файла, название диска и каталога, в котором находится файл
2.	Укажите неправильный вариант записи полного имени файла *result.dat*: а) *C:\stud\group\result.da;* б) *C:\STUD\GROUP\RESULT.DAT;* в) *STUD\GROUP\result.dat;* г) все варианты правильны
3.	Могут ли в одном каталоге существовать файлы с одинаковыми именами? а) могут; б) могут, если имеют разные расширения; в) могут, если первоначально созданы в разных каталогах; г) не могут
4.	Могут ли в разных каталогах существовать файлы с одинаковыми именами? а) могут; б) могут, если имеют разные расширения; в) могут, если каталоги на разных дисках; г) не могут
5.	Может ли имя файла содержать пробелы? а) может, между именем и расширением; б) не может; в) может, но не более одного; г) не может только в расширении
6.	Как вызвать меню, задающее установки режима работы Norton Commander? а) <*F7*>; б) <*F8*>; в) <*F9*>; г) <*F10*>
7.	Как переключить активность панелей Norton Commander? а) <*Alt*>; б) <*Esc*>; в) <*Tab*>; г) другая клавиша
8.	Как вызвать редактор для создания нового файла, работая в Norton Commander? а) <*F4*>; б) <*Alt-F2*>; в) <*Shift-F4*>; г) <*Shift-F6*>
9.	Как вызвать редактор для редактирования имеющегося на диске файла, работая в Norton Commander? а) <*F3*>; б) <*Shift-F4*>; в) <*Ctrl-F4*>; г) <*F4*>
10.	Как переключить режимы редактирования (вставка-замена)? а) <*F2*>; б) <*Del*>; в) <*F9*>; г) другая клавиша
11.	Как выбрать новый диск на правой панели Norton Commander? а) <*Alt-F1*>; б) <*Alt-F2*>; в) <*Shift-F1*>; г) <*Shift-F2*>
12.	Как переименовать файл, работая в Norton Commander? а) <*F6*>; б) <*F3*>; в) <*F9*>; г) <*F2*>
13.	Как создать каталог, работая в Norton Commander? а) - <*F2*>; б) <*F7*>; в) <*F6*>; г) другая клавиша
14.	Как войти в каталог, работая в Norton Commander? а) нажать клавишу <*F7*>; б) установить курсор на имя каталога, нажать <*Enter*>; в) набрать имя каталога в командной строке, нажать <*Enter*>; г) установить курсор на имя каталога
15.	Как скопировать каталог вместе со всеми содержащимися в нем файлами, работая в Norton Commander? а) <*Shift-F5*>; б) <*F7*>, затем <*F5*>; в) <*F6*>; г) <*F5*>

Задание №3. Операционная система Windows

№	Вопрос
1.	Сколько программ одновременно может быть запущено при работе в *Windows*? а) не более 3; б) сколько угодно, при условии что открыто только окно активной в данной момент программы, а все остальные свернуты до значка; в) сколько угодно (количество ограничено размером оперативной памяти компьютера); г) одна
2.	Сколько ярлыков для одного и того же приложения может содержать *Windows*? а) только один в личной папке; б) только один на рабочем столе в) один в личной папке и один на рабочем столе; г) сколько угодно
3.	Панель задач *Windows* не содержит: а) кнопку Пуск; б) закрытые, но не удаленные приложения; в) открытые, но свернутые приложения; г) может содержать все из вышеперечисленного
4.	Движок вертикальной полосы прокрутки: а) является индикатором размера приложения; б) является индикатором расположения окна внутри приложения; в) служит для перемещения окна внутри приложения; г) все утверждения верны
5.	Для уменьшения размеров *Windows*-окна необходимо: а) переместить угол окна; б) переместить границу окна; в) выбрать команду **Свернуть**; г) выполнить любое из этих действий
6.	Для записи *Windows*-приложения с прежним именем на магнитный диск необходимо выбрать: а) команду **Сохранить**; б) команду **Копировать**; в) команду **Сохранить как**…; г) другую команду
7.	Меню Правка не содержит команды: а) **Копировать**; б) **Вставить**; в) **Изменить**; г) содержит все вышеперечисленные команды
8.	Для копирования приложения в другую папку необходимо: а) воспользоваться командой **Копировать** меню **Правка**; б) перетащить значок приложения с помощью мыши; в) перетащить значок приложения с помощью мыши с нажатой клавишей <Ctrl>; г) возможен любой из перечисленных способов
9.	Работая в графическом редакторе *Paint*, пользователь нарисовал дом, изобразив в нем окна двух типов: сначала круглые, затем – квадратные. Обнаружив ошибку, пользователь желает все окна заменить на круглые. Какое из перечисленных действий не приведет к желаемому результату? а) использование **Ластика**; б) использование команды **Правка/Отменить**; в) использование кнопки **Масштаб**; г) выделение ненужных фрагментов, удаление их и рисование новых
10.	Работая в редакторе *Word*, пользователь выделил некоторый фрагмент текста и удалил его с экрана клавишей <*DELETE*>. Как восстановить фрагмент на экране? а) выбрав в пункте меню **Правка** команду **Вставить**; б) выбрав в пункте меню **Правка** команду **Отменить**; в) выбрав в пункте меню **Правка** команду **Отменить удаление**; г) только набрать фрагмент заново
11.	Может ли фрагмент выполненного в *Paint* рисунка быть помещен с помощью **Буфера Обмена** в документ, созданный в редакторе *Word*? а) да, может; б) нет, не может; в) может, но только весь рисунок целиком; г) нет, т.к. **Буфер** Обмена доступен только для символьной информации

12.	Как при работе в редакторе *Word* наиболее быстро переместиться в конец документа? а) щелкнув мышью на стрелке в нижней части полосы прокрутки; б) нажав клавишу <*PgDn*>; в) щелкнув мышью под движком на полосе прокрутки; г) протащив вниз движок вертикальной полосы прокрутки
13.	Как при работе в редакторе *Word* переместить некоторую часть текста в другое место документа? а) выделить нужный фрагмент и переместить его, пользуясь клавишами-стрелками; б) выделить нужный фрагмент, дважды щелкнуть на нем мышью, затем дважды щелкнуть мышью в том месте, где должен появиться текст; в) выделить нужный фрагмент, поместить его в **Буфер Обмена** командой **Копировать**, переместиться в то место документа, где должен появиться текст и выполнить команду **Вставить**; г) выделить нужный фрагмент, поместить его в **Буфер Обмена** командой **Вырезать**, переместиться в то место документа, где должен появиться текст и выполнить команду **Вставить**
14.	Работая в редакторе *Word,* пользователь набрал нужный ему текст и хочет перейти к набору нового текста в новом файле. Какие действия он должен выполнить? а) выбрать команду **Файл/Сохранить**, затем команду **Закрыть** в системном меню, после чего загрузить *Word* двойным щелчком мыши на его значке; б) выбрать команду **Файл/Сохранить**, а затем команду **Файл/ Открыть**; в) выбрать команду **Файл/Сохранить**, а затем командой **Файл/ Сохранить Как**... создать на диске файл с новым именем, при этом окно редактора *Word* очистится и можно будет набирать новый текст; г) выбрать команду **Файл/Сохранить**, затем команду **Файл/Создать**
15.	Набрав текст в редакторе *WordPad*, пользователь решил выделить все символы в одной строке курсивом. Для этого он должен: а) удалить строку, затем выбрать команду **Формат/Шрифт**…, выбрать вид шрифта Курсив и набрать строку заново; б) перевести указатель мыши на первый символ в строке, выбрать команду **Формат/Шрифт**…, выбрать вид шрифта **Курсив**; в) выделить строку, выбрать команду **Формат/Шрифт**…, выбрать вид шрифта **Курсив**; г)выделить строку, занести ее в **Буфер Обмена**, где выбрать команду **Формат/Шрифт**

Контрольные вопросы

1. Архитектура ПК, ее основные элементы.
2. Классификация программного обеспечения.
3. Что такое файл? Обозначение файла на диске. Приведите примеры использования расширений.
4. Что такое каталог? Сколько файлов может содержать каталог? Сколько каталогов может содержаться на диске?
5. Сколько файлов с одинаковым обозначением может содержаться на диске? В одном и том же каталоге?
6. Логические диски, их обозначения.
7. Дерево каталогов. Понятия корневого каталога, подкаталогов.
8. Путь к файлу. Полное имя файла.

9. Назначение программы *Far*.
10. Структура панелей *Far*.
11. Функциональные клавиши *Far*.
12. Редактирование файла средствами *Far*.
13. Копирование, перенос и другие действия с файлами и каталогами в *Far*.
14. В чем особенности операционной системы *Windows* ?
15. Основные элементы рабочего стола *Windows*.
16. Папки и ярлыки.
17. Состав Панели задач.
18. Для чего используется указатель мыши? Основные виды указателя.
19. Какие существуют приемы работы с мышью?
20. Структура *Windows*-окна.
21. Работа с окнами. Перемещение, изменение размеров, реорганизация.
22. Использование меню *Windows*-приложения.
23. Использование панелей инструментов *Windows*-приложения.
24. Назначение и основные принципы работы с программой Проводник.
25. Графический редактор *Paint.* Окно программы. Палитра. Панель инструментов. Корректировка изображения. Создание текста. Работа с фрагментами.
26. Текстовый редактор *Word*. Окно программы. Набор и редактирование текста. Действия с выделенными фрагментами. Форматирование текста. Сохранение и печать документа.
27. Назначение Буфера обмена и технология работы с ним.
28. Элементы алгоритмического языка Фортран. Структура программы.
29. Основные символы языка. Знаки арифметических операций. Операции отношений.
30. Константы и переменные. Стандартные функции.
31. Арифметическое выражение. Последовательность выполнения операций.
32. Оператор присваивания. Что записывается слева и справа от знака присваивания?
33. Операторы ввода-вывода данных.
34. Операторы условного и безусловного перехода.
35. Структура составного оператора условия.
36. Понятие алгоритма. Формы представления алгоритма.
37. Блок-схема. Основные типы блоков.
38. Линейный алгоритм, его блок-схема и реализация на алгоритмическом языке. Привести примеры.
39. Разветвляющийся алгоритм. Блок-схемы алгоритмов при наличии двух и более условий.
40. Программная реализация разветвляющегося алгоритма с использованием операторов безусловного и условного перехода.
41. Программная реализация разветвляющегося алгоритма с использованием структурного оператора условия.
42. Циклический алгоритм и его блок-схема.
43. Что такое переменная цикла, тело цикла, параметры цикла?

44. Какой алгоритм может быть телом цикла?
45. Правила организации многомерных (вложенных) циклов.
46. Отличия простых и индексированных переменных.
47. Понятие массива. Одномерные и двумерные массивы. Размерность массива. Правила записи индексов.
48. Оператор описания массива. Назначение и структура.
49. Оператор для работы с циклами *DO - END DO*.
50. Базовые алгоритмы и программы ввода и вывода одномерных и двумерных массивов.
51. Алгоритмы суммирования, нахождения произведения и количества элементов массива.
52. Перестановка местами элементов массива.
53. Алгоритмы нахождения максимального и минимального элементов массива. Когда необходима фиксация индексов?
54. Формирование новых массивов.
55. Работа с ограничениями и квадратными матрицами.

8. ГЛОССАРИЙ

CD-ROM – устройство для чтения компакт-дисков.
Алгоритм – это последовательность действий, приводящих к намеченному результату.
Бит – единица информации в компьютере, двоичный разряд, принимающий значения 0 или 1.
Байт – восемь последовательных бит.
Блок-схема – графический способ описания алгоритмов.
Буфер обмена – *(Windows)* – специальная область памяти, которая позволяет разным приложениям Операционной системы (см.) обмениваться данными.
Гигабайт – единица измерения информации, равная 1024 Мбайт.
Двумерный массив – имеет два индекса.
Дисплей – устройство вывода (отображения) информации.
Жесткий диск – внутреннее устройство долговременного хранения информации. Одно физическое устройство может быть разделено на несколько логических.
Индексированная переменная – ячейки в памяти ЭВМ, которые имеют одно и тоже имя, но отличаются номером (индекса).
Интерфейс – совокупность программно-аппаратных средств, обеспечивающих пользователю необходимый уровень общения с компьютером.
Информатика – это область знаний, изучающая как саму информацию, так и вопросы, связанные с ее сбором, обработкой и хранением.
Килобайт – единица измерения информации, равная 1024 байт.
Клавиатура – устройство ввода информации.
Компакт-диск – долговременное хранилище информации. Запись и чтение осуществляется посредством лазерного луча. (см. *CD-ROM*).
Линейный алгоритм – это такой алгоритм, в котором действия выполняются последовательно, в порядке расположения блоков.

Массив – совокупность индексированных переменных.

Мегабайт – единица измерения информации, равная 1024 кбайт.

Микропроцессор – электронная схема, выполняющая все вычисления и обработку информации.

Модем – устройство для подключения компьютера в глобальную вычислительную сеть через телефонный канал.

Монитор – см. Дисплей.

Мышь – устройство управления графическим курсором. Состоит из самого манипулятора и нескольких кнопок на нем.

Одномерный массив – имеет один индекс.

Окно – *(Windows)* – область экрана, в которой выполняется программа. В Windows каждая программа выполняется в своем окне.

Папка – (см. каталог) – термин операционной системы *Windows.*

Перетаскивание - *(Windows)* – распространенная операция. Заключается в следующем: мышь (см.) наводится на объект и нажимается левая кнопка. Потом не отпуская левой кнопки производится перемещение мыши в новое положение. После отпускания левой кнопки объект оказывается в том месте, куда была перемещена мышь.

Персональный компьютер (ПК) – совокупность системного блока (см.) и периферийных устройств (см.)

Полное имя файла – состоит из имени логического диска, имени каталога и имени файла.

Принтер – устройство вывода информации на бумагу

Программное обеспечение – совокупность программ, обеспечивающих его функционирование. Подразделяется на:

- ***Операционные системы*** – организуют управление внутренними устройствами ПК и периферией.
- ***Прикладные программы*** – набор средств для решения тех или иных задач пользователя, либо создания программ, осуществляющих такое ре-

шение.

- ***Интегрированные системы*** – предлагают комплексный подход для решения проблем в рамках одного программного продукта.
- ***Проблемно-ориентированные пакеты*** – узкоспециализированное программное обеспечение для применения в конкретной области (бухучет, инженерская деятельность, научные исследования и т.п.)

Разветвляющийся алгоритм – содержит блок разветвления.

Размерность массива – общее количество элементов.

Сервер – главный компьютер в локальных вычислительных сетях.

Системный блок – совокупность внутренних устройств ПК, собранных в едином корпусе.

Сканер – устройство для считывания графической и текстовой информации в компьютер.

Стример – устройство для работы с одноименными носителями информации. Запись и чтение осуществляется посредством электромагнитного поля. Носитель информации – пленка с напылением ферромагнитного слоя.

Тело цикла – многократно повторяющаяся часть алгоритма, внутри которой переменная цикла не изменяется.

Транслятор – специальная программа перевода программ с языка высокого уровня на язык машинных кодов.

Файл – совокупность информации, имеющая имя и записанная на логический носитель.

Ярлык – (см. файл) специальный файл операционной системы *Windows,* хранящий в себе путь к объекту, на который этот ярлык ссылается.

Циклический алгоритм – часть которого выполняется многократно с различными значениями изменяющийся по определенному закону переменной.

9. СПИСОК ЛИТЕРАТУРЫ

1. Симонович, С.В. Информатика. Базовый курс: учебник для вузов / С.В. Симонович [и др.]- СПб.: Питер, 2000.
2. Фигурнов, В.Э IBM PC для пользователя / В. Э. Фигурнов, Изд. 7-е, перераб. и доп.- М.: ИНФРА-М., 1999.
3. Меткалф, М. Описание языка программирования ФОРТРАН-90 / М. Меткалф, Дж.М. Рид - Мир.: 1995.
4. Богумирский, Б.В. Windows 98: справочник / Б.В. Богумирский - СПб.: Питер, 1999
5. Кетков, Ю.Л. Командно-файловый процессор Norton Commander, версия 3.0: спецпрактикум / Ю.Л. Кетков, - Н. Новгород: Изд-во ННГУ, 1991.
6. Основы алгоритмизации. Алгоритмический язык Фортран**:** метод. указания по курсу «Информатика» для студентов заочного отделения / НГТУ; Сост.: С.Н.Митяков, Е.Ф.Листопад, С.П.Никитенкова, Н.Я.Николаев. Н.Новгород, 2000.
7. Начальные сведения о персональной ЭВМ типа IBM PC. Программа – оболочка Norton Commander.Операционная система Windows: методические указания по курсу "Информатика" для студентов заочного отделения. Часть 1 / НГТУ, Н.Новгород, 2000