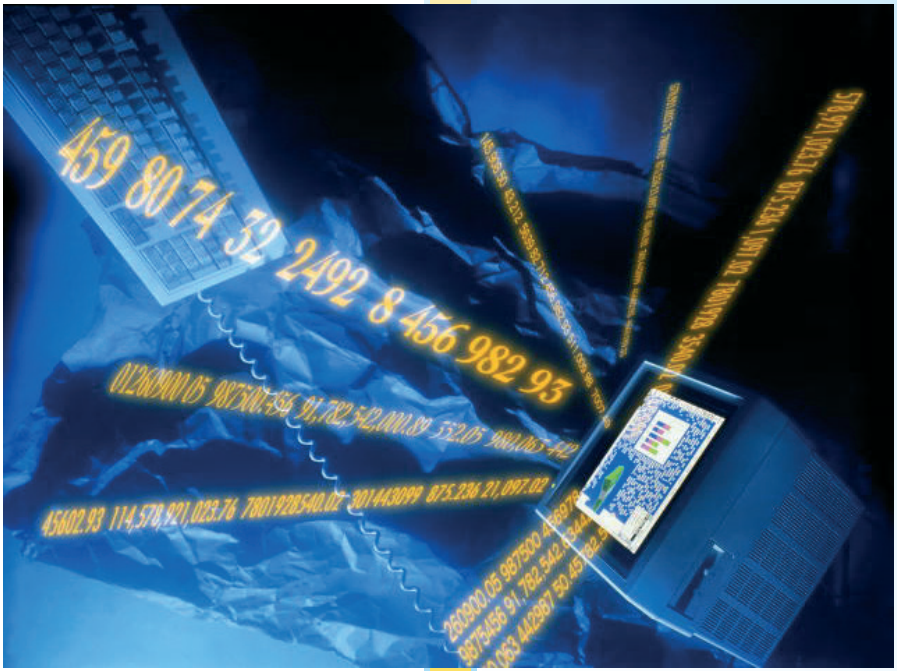
ТОЛЬЯТТИНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

В.Ф. Глазова
А.В. Богданова
Е.В. Панюкова

ИНФОРМАТИКА

Учебно-методическое пособие

Часть 1



Министерство образования и науки Российской Федерации
Тольяттинский государственный университет
Институт математики, физики
и информационных технологий
Кафедра «Информатика и вычислительная техника»

В.Ф. Глазова, А.В. Богданова, Е.В. Панюкова

ИНФОРМАТИКА

Учебно-методическое пособие

В двух частях

Часть 1

Рекомендовано УМО РАЕ по классическому университетскому
и техническому образованию в качестве учебно-методического
пособия для студентов высших учебных заведений,
обучающихся по направлениям подготовки 080200 – «Менеджмент»,
080400 – «Управление персоналом», 221400 – «Управление качеством»
(квалификация (степень) «бакалавр»)

Тольятти
Издательство ТГУ
2013

УДК 004 (075.8)
ББК 32.81я73
Г524

Рецензенты:
канд. техн. наук, доц. Поволжского государственного университета сервиса *Е.Ю. Малышева*;
канд. техн. наук, доц. Тольяттинского государственного университета *Э.В. Егорова*.

Г524 Глазова, В.Ф. Информатика : учеб.-метод. пособие : в 2 ч. / В.Ф. Глазова, А.В. Богданова, Е.В. Панюкова. – Тольятти : Изд-во ТГУ, 2013. – Ч. 1. – 243 с. : обл.

Учебно-методическое пособие (в двух частях) предназначено для обучения дисциплине «Информатика». Первая часть состоит из трех разделов, каждый содержит изложение теоретического материала, вопросы для самоконтроля, задания для выполнения практических работ, тестовые задания. Пособие может быть использовано на аудиторных занятиях и для самостоятельной работы студентов.

Предназначено для студентов системы высшего профессионального образования, обучающихся очно по направлениям подготовки бакалавров 080200 «Менеджмент», 080400 «Управление персоналом», 221400 «Управление качеством».

УДК 004 (075.8)
ББК 32.81я73

Рекомендовано к изданию научно-методическим советом Тольяттинского государственного университета.

Предисловие

Первая из двух частей учебно-методического пособия по дисциплине «Информатика» предназначена для студентов очной формы обучения. Учебные материалы и связанные с ними учебные мероприятия в рамках изучения дисциплины организованы в обучающие модули. Каждый раздел данного пособия соответствует отдельному обучающему модулю дисциплины и имеет следующую структуру: цели изучения; методическая, теоретическая, практическая и контролирующая части.

Цели изучения представлены с позиции обучаемого и являют собой перечень знаний («Вы узнаете ...») и умений («Вы научитесь ...»), которые будут приобретены в результате работы с учебными материалами раздела. Методическая часть содержит краткое содержание и структурную схему работы с учебными материалами модуля, а также ссылки на дополнительные источники информации.

Теоретическая часть представляет собой структурированное по параграфам краткое изложение теоретических учебных материалов. В конце каждого параграфа приводятся вопросы и упражнения для самоконтроля. Практическая часть содержит описание практических работ. Каждая практическая работа включает блок целеполагания, методический блок с рекомендациями по выполнению работы, упражнения или практические задания, пример выполнения задания или описание технологии выполнения задания, варианты заданий.

В конце каждого раздела приводятся тестовые задания по изучаемым темам. Правильность выполнения заданий студент может проконтролировать самостоятельно с помощью ключей к тестам, представленным в приложении 2 данного пособия. Приложение 1 содержит глоссарий, в приложении 3 приведены вопросы для подготовки к экзамену.

Данное пособие содержит учебные материалы по темам, изучаемым **в первом семестре** в рамках дисциплины «Информатика».

Цель изучения дисциплины – формирование системы знаний и умений в области информационных технологий, составляющих основу базовой информационной компетентности выпускника вуза.

В результате изучения дисциплины в первом семестре студент должен **знать:**

- роль и значение информации и информационных технологий в развитии современного общества и экономических знаний;
- арифметические и логические основы организации компьютерных систем;
- основные методы сбора, передачи, обработки и накопления информации с помощью компьютера;
- основы алгоритмизации и программирования;
- современное состояние уровня развития вычислительной техники и программных средств;

уметь:

- использовать системные программные средства для оптимизации вычислительной системы;
- использовать текстовые процессоры для подготовки документов различного назначения;
- обрабатывать данные, используя электронные таблицы;
- применять алгоритмический подход к решению задач обработки информации;

владеть:

- основными методами, способами и средствами получения, хранения, переработки информации;
- навыками работы с компьютером как средством обработки информации;
- навыками работы с программным обеспечением для работы с деловой информацией (текстовые процессоры, электронные таблицы, средства подготовки презентаций).

Содержание разделов дисциплины «Информатика», относящихся к первому семестру обучения, представлено в таблице.

Раздел 1. Основы теории информации и кодирования. Арифметические и логические основы устройства компьютеров	
Тема	**Учебные вопросы темы**
1.1. Информация, её свойства и измерение	1. Понятие и свойства информации. 2. Меры и единицы измерения информации
1.2. Представление чисел в разных системах счисления	1. Арифметические операции с числами в разных системах счисления. 2. Перевод чисел из одной системы счисления в другую
1.3. Кодирование информации и представление данных в памяти компьютера	1. Кодирование и представление в памяти компьютера числовой, текстовой, графической, аудио- и видеоинформации
1.4. Логические основы устройства компьютеров	1. Основы алгебры логики: логические высказывания, логические операции, таблицы истинности логических операций, логические выражения, построение таблиц истинности логических выражений. 2. Логические элементы. Построение логических схем для заданных логических выражений
Раздел 2. Основы алгоритмизации и программирования	
Тема	**Учебные вопросы темы**
2.1. Технология решения задач с использованием компьютера	1. Основные этапы решения задачи на компьютере
2.2. Алгоритмизация вычислений	1. Понятие алгоритма, свойства алгоритмов, средства представления алгоритмов. 2. Базовые структуры алгоритмов. Типовые алгоритмы решения вычислительных задач
2.3. Языки программирования	1. Типы и разновидности языков программирования 2. Технологии программирования на языках высокого уровня

2.4. Основы языка программирования Паскаль	1. Структура программы и основные объекты программы на языке Паскаль. 2. Операторы языка Паскаль. Реализация типовых алгоритмов средствами языка
Раздел 3. Технические и программные средства реализации информационных процессов	
Тема	**Учебные вопросы темы**
3.1. Технические средства реализации информационных процессов	1. История развития компьютерной техники. 2. Основные принципы устройства компьютеров. 3. Устройство персональных компьютеров
3.2. Программные средства реализации информационных процессов	1. Структура программного обеспечения компьютеров. 2. Назначение и состав системного и прикладного программного обеспечения
3.3. Работа с текстовыми документами на компьютере	1. Установка пользовательского интерфейса. 2. Установка параметров страницы, абзаца, шрифта. 3. Работа с таблицами, рисунками, формулами. 4. Выполнение сервисных функций по работе с документом, создание массовых рассылок документов
3.4. Работа с электронными таблицами	1. Установка пользовательского интерфейса. 2. Выполнение расчетов и построение диаграмм. 3. Обработка данных, структурированных в виде списков
3.5. Создание презентаций в Microsoft PowerPoint	1. Режимы работы с программой Microsoft PowerPoint. 2. Способы создания и сохранения презентации. 3. Показ презентации

При работе с пособием предлагается учитывать следующие соглашения:

- впервые используемые в рамках пособия термины и определения выделены ***жирным курсивным*** шрифтом;
- команды меню, названия диалоговых полей, кнопок и прочих элементов интерфейса пользователя программ пакета Microsoft Office выделены **жирным** шрифтом;
- команды меню программы, выполняемые последовательно, разделяются тире, например, строка **Файл – Сохранить** предполагает следующую последовательность операций: открыть пункт меню **Файл**, затем выбрать команду **Сохранить**;
- фрагменты текста пособия, на которые рекомендуется обратить особое внимание, отмечены специальным значком ☑.

Описание работы с текстовым и табличным процессором, а также с программой подготовки презентаций выполнено для соответствующих приложений пакета Microsoft Office 2003.

Над пособием работал авторский коллектив:

Е.В. Панюкова	– раздел «Основы теории информации и кодирования. Арифметические и логические основы устройства компьютеров»;
В.Ф. Глазова	– раздел «Основы алгоритмизации и программирования»;
А.В. Богданова и В.Ф. Глазова	– раздел «Технические и программные средства реализации информационных процессов».

1. ОСНОВЫ ТЕОРИИ ИНФОРМАЦИИ И КОДИРОВАНИЯ. АРИФМЕТИЧЕСКИЕ И ЛОГИЧЕСКИЕ ОСНОВЫ УСТРОЙСТВА КОМПЬЮТЕРОВ

Изучив материалы этого раздела, вы **узнаете**:

- трактовку информатикой понятий «информация» и «данные»;
- способы измерения информации и данных;
- правила выполнения операций с числами в разных системах счисления;
- способы кодирования данных разных типов при их хранении и обработке с помощью компьютера;
- законы алгебры логики, использующиеся в описании функционирования логических схем устройств компьютера.

Выполнив практические работы, вы **научитесь**:

- вычислять количество информации в сообщении на основе вероятностного подхода;
- выполнять перевод чисел из одной системы счисления в другую;
- выполнять арифметические действия с числами в двоичной, восьмеричной, шестнадцатеричной системах счисления;
- записывать машинные коды чисел, заданных в десятичной системе счисления;
- строить таблицы истинности для заданных логических выражений;
- строить функциональные схемы на основе заданных логических выражений.

Методические рекомендации

Схема работы с учебными материалами данного раздела представлена на рис. 1.1. Теоретическая часть предполагает изучение четырех тем. Каждая тема заканчивается перечнем вопросов и упражнений для самоконтроля. Практическая часть предполагает выполнение трех практических работ и индивидуального домашнего задания. Изучение

раздела заканчивается выполнением тестовых заданий, ключ к которым можно найти в прил. 2. Дополнительные учебные материалы по данному разделу можно найти в рекомендованных литературных источниках [1], [2], [4], [7] и интернет-ресурсах [12], [13], список которых представлен в конце пособия.

Рис. 1.1. Схема работы с учебными материалами раздела 1

1.1. Информация, её свойства и измерение

1.1.1. Понятие «информация». Свойства информации

Информатика – наука, изучающая способы создания, хранения, обработки и передачи информации с помощью компьютера, а также принципы функционирования компьютеров и методы управления ими.

Термин «информация» происходит от латинского слова informatio, что в переводе означает «сведение, разъяснение, ознакомление». Информация может существовать в виде текстов, рисунков, фотографий, световых или звуковых сигналов, радиоволн, электрических и нервных импульсов, магнитных записей, запахов и вкусовых ощущений и т. д. ***Сигнал*** представляет собой любой процесс, несущий информацию. Сигнал называется ***непрерывным***, если его параметр в заданных пределах может принимать любые промежуточные значения. Сигнал называется ***дискретным***, если его параметр в заданных пределах может принимать отдельные фиксированные значения.

Различают две формы представления информации – ***непрерывную (аналоговую)*** и ***дискретную***. Поскольку носителями информации являются сигналы, то в качестве сигналов могут использоваться физические процессы различной природы. Например, процесс протекания электрического тока в цепи, процесс механического перемещения тела, процесс распространения света и т. д. Информация представляется (отражается) значением одного или нескольких параметров физического процесса (сигнала) либо комбинацией нескольких параметров.

Качество информации является одним из важнейших параметров для её потребителя. Оно определяется следующими ***свойствами***:

- ***репрезентативность*** – правильность отбора информации в целях адекватного отражения источника информации;
- ***достаточность*** – минимальный, но достаточный объем информации для достижения целей, которые преследует ее потребитель;

- ***доступность*** – мера возможности получить ту или иную информацию;
- ***актуальность*** – степень сохранения ценности информации для управления в момент ее использования, зависит от динамики изменения ее характеристик и от интервала времени, прошедшего с момента возникновения данной информации;
- ***своевременность*** – поступление информации не позже заранее назначенного момента времени, согласованного со временем решения поставленной задачи;
- ***точность*** – степень близости получаемой информации к реальному состоянию объекта, процесса, явления;
- ***адекватность*** – это определенный уровень соответствия создаваемого с помощью полученной информации образа реальному объекту, процессу, явлению;
- ***устойчивость*** – способность информации реагировать на изменения исходных данных без нарушения необходимой точности.

Информация передаётся в виде сообщений от некоторого источника к её приёмнику посредством канала связи между ними. ***Сообщение –*** это информация, представленная в определенной форме и предназначенная для передачи. Источник посылает передаваемое сообщение, которое кодируется в передаваемый сигнал. Этот сигнал посылается по каналу связи. В результате в приёмнике появляется принимаемый сигнал, который декодируется и становится принимаемым сообщением (рис. 1.2).

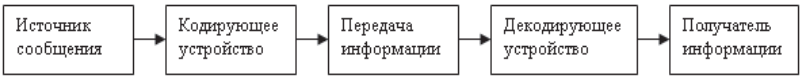


Рис. 1.2. Схема передачи информации

Сообщение, содержащее информацию о прогнозе погоды, передаётся приёмнику (телезрителю) от источника – специалиста-метеоролога посредством канала связи – телевизионной передающей аппаратуры и телевизора. Передача информации по каналам связи часто сопровождается воздействием помех, вызывающих искажение и потерю информации.

Данные – это зарегистрированные сигналы. В зависимости от метода регистрации данные могут храниться на различных носителях: бумага, магнитная лента или магнитный диск, оптический диск и т. д.

1.1.2. Меры и единицы измерения информации

Информация понимается и интерпретируется по-разному в различных предметных областях. Вследствие этого имеются различные подходы к определению измерения информации и различные способы введения меры количества информации. ***Количество информации*** – числовая величина, адекватно характеризующая информацию по разнообразию, сложности, структурированности (упорядоченности), определенности, выбору состояний отображаемой системы. Для измерения количества информации используется понятие меры информации. Существуют различные подходы к определению меры информации.

Синтаксический (вероятностный) подход

Мера, характеризующая количество информации в некотором сообщении, есть в данном случае числовая характеристика, отражающая степень неопределенности (неполноту знаний), которая исчезает после получения сообщения. Эту меру неопределенности в теории информации называют ***энтропией***.

Бит – минимальная единица количества информации, которое содержит сообщение, уменьшающее неопределенность знаний в два раза. Например, каждое бросание монеты дает нам информацию в 1 бит.

Пусть до получения информации потребитель имеет некоторые предварительные (априорные) сведения о системе α. Мерой его неосведомленности о состоянии системы является некоторая функция (энтропия) $H(\alpha)$. После получения некоторого сообщения β получатель приобрел дополнительную информацию, уменьшившую его неосведомленность так, что она стала равной $H_\beta(\alpha)$. Тогда количество информации о системе, полученной в сообщении β, определится как $I_\beta(\alpha) = H(\alpha) - H_\beta(\alpha)$, т. е. количество информации измеряется уменьшением энтропии. В частном случае для системы,

имеющей ***n*** возможных состояний, энтропия может быть вычислена по ***формуле Шеннона***:

$$H(\alpha) = -\sum_{i=1}^{n} p_i \cdot \log p_i = -\left(p_1 \cdot \log_2 p_1 + p_2 \cdot \log_2 p_2 + \ldots + p_n \cdot \log_2 p_n\right), \quad (1.1)$$

где ***n*** – количество возможных состояний системы; p_i – вероятность состояния с номером ***i***.

Наиболее просто определить количество информации, когда все состояния системы равновероятны. В этом случае для вычисления количества информации, содержащейся в сообщении об одном состоянии системы (***I***), используется ***формула Хартли***:

$$I = \log_2 n, \quad (1.2)$$

где ***I*** – количество информации; ***n*** – множество сообщений.

Согласно этой формуле процесс получения информации рассматривается как выбор одного сообщения из конечного заданного множества ***n*** равновероятных сообщений, а количество информации ***I***, содержащееся в выбранном сообщении, определяется как двоичный логарифм от ***n***.

Пример 1.1. Из 8 карточек с номерами 1, 2, ..., 8 случайным образом извлекли одну. Сколько информации будет заключено в сообщении о том, какое число записано на карточке?

Событие, заключающееся в извлечении карточки, имеет 8 возможных исходов, вероятности которых одинаковы. Рассматривается система, для которой справедлива формула Хартли. Следовательно, информация о реализации одного из возможных состояний системы равна $I = \log_2 8 = \log_2 2^3 = 3$ бита. Здесь мы воспользовались одним из свойств логарифмов, согласно которому $\log_2 2^k = k$.

Предположим, что число возможных равновероятных состояний некоторой системы равно ***n1***. Получено некоторое сообщение **β** о системе, в результате чего число возможных состояний системы уменьшилось и стало равным ***n2***. В этом случае количество информации в сообщении I_β можно определить по формуле

$$I_\beta = \log_2 \frac{n1}{n2}. \quad (1.3)$$

Пример 1.2. Колода игральных карт состоит из 16 черных и 16 красных карт (4 туза, 4 короля и т. д.). Случайным образом из колоды извлекается одна карта. Получено сообщение о том, что извлечена карта черной масти. Сколько информации содержит это сообщение?

Вероятность извлечения любой карты из колоды одинакова для всех карт. Мы имеем дело с системой, для которой справедлива формула Хартли. Число возможных состояний системы до получения сообщения равно количеству карт, т. е. ***n1*** = *32*. После получения сообщения число возможных состояний уменьшилось и стало равно числу карт черной масти в колоде, т. е. ***n2*** = *16*. Количество информации в сообщении определим по формуле (1.3):

$$I = \log_2 \frac{32}{16} = \log_2 2 = 1 \text{ бит}.$$

Пример 1.3. В условиях примера 1.2 определить, какое из двух сообщений «извлечен туз» или «извлечена дама пик» содержит большее количество информации?

В данном случае, как и в примере 1.2, ***n1*** = *32*. Для сообщения «извлечен туз» число возможных состояний системы после получения сообщения равно ***n2*** = *4* (в колоде 4 туза). Соответственно, для сообщения «извлечена дама пик» ***n2*** = *1* (в колоде одна дама пик). Вычислим количество информации в сообщении «извлечен туз»: $I = \log_2 \frac{32}{4} = \log_2 8 = \log_2 2^3 = 3 \text{ бит}$. Количество информации в сообщении «извлечена дама пик»: $I = \log_2 \frac{32}{1} = \log_2 32 = \log_2 2^5 = 5 \text{ бит}$. Расчеты показывают, что больше информации содержит второе сообщение.

☑ В теории информации доказано, что чем менее вероятным является событие, информация о котором содержится в сообщении, тем больше информации содержит это сообщение. Рассмотренный пример иллюстрирует данный факт.

Семантическая мера информации

Семантическая мера используется для измерения смыслового содержания информации. Наибольшее признание

получила тезаурусная мера, которая связывает семантические свойства информации со способностью пользователя принимать поступившее сообщение, а также новизной сведений, содержащихся в сообщении.

Тезаурус – совокупность сведений, которыми располагает пользователь или система (потребитель информации). Максимальное количество семантической информации потребитель получает при согласовании ее смыслового содержания с тезаурусом, когда поступающая информация понятна пользователю и несет ему ранее неизвестные сведения. Если тезаурус мал, то пользователь не понимает поступающую информацию и количество получаемой им информации близко к нулю. Если поступающая информация известна пользователю, то он не получает новых сведений, в этом случае количество информации с точки зрения семантического подхода также близко к нулю.

Прагматическая мера информации

Прагматическая мера определяет полезность (ценность) информации для достижения пользователем поставленной цели. Эта мера также величина относительная, обусловленная особенностями использования этой информации в той или иной системе. Ценность информации целесообразно измерять в тех же самых единицах (или близких к ним), в которых измеряется целевая функция.

Измерение объема данных

Объем данных в сообщении измеряется количеством символов (разрядов) в этом сообщении. В зависимости от используемой системы кодирования один разряд имеет различный вес и соответственно меняется единица измерения данных. В двоичной системе счисления единица измерения – ***бит*** (binary digit – двоичный разряд), в десятичной системе счисления единица измерения – ***дит*** (десятичный разряд). Например, сообщение в двоичной системе в виде восьмиразрядного двоичного кода 10111011 имеет объем данных 8 бит.

Наряду с минимальной единицей измерения данных (бит) широко используется укрупненная единица измерения ***байт***, равная 8 битам. Производными единицами являются:

1 Кбайт (килобайт) = 2^{10} байт = 1024 байт;
1 Мбайт (мегабайт) = 2^{10} Кбайт = 1024 Кбайт;
1 Гбайт (гигабайт) = 2^{10} Мбайт = 1024 Мбайт;
1 Тбайт (терабайт) = 2^{10} Гбайт = 1024 Гбайт.

? Вопросы и упражнения для самоконтроля

1. Опишите содержание понятия «информация». Назовите формы представления информации.
2. Перечислите свойства информации.
3. Дайте определение понятию «количество информации».
4. Как связаны между собой понятия «энтропия» и «информация»?
5. Опишите назначение и условия применения формулы Шеннона.
6. Каким свойствам должна удовлетворять система, чтобы для нее была справедлива формула Хартли?
7. Что измеряют семантическая мера информации, прагматическая мера информации?

1.2. Представление чисел в разных системах счисления

Системой счисления называется способ записи чисел с помощью заданного набора специальных знаков (цифр). Различают системы счисления позиционные и непозиционные.

В ***позиционных*** системах счисления вес каждой цифры изменяется в зависимости от ее положения (позиции) в последовательности цифр, изображающих число. Например, в числе 555 первая пятерка означает пять сотен, вторая – 5 десятков, а третья 5 единиц. В ***непозиционных*** системах вес цифры (т. е. тот вклад, который она вносит в значение числа) не зависит от ее позиции в записи числа. Например, в римской системе счисления в числе XXI (двадцать один) вес цифры X в любой позиции равен десяти.

Любая позиционная система счисления характеризуется ***основанием*** – количеством различных знаков или символов, используемых для изображения чисел в данной системе. За основание системы можно принять любое натуральное

число. Следовательно, возможно бесчисленное множество позиционных систем: двоичная, троичная, четверичная и т. д.

Наибольшее распространение для представления чисел при их обработке на компьютере получили двоичная, восьмеричная и шестнадцатеричная системы счисления. В ***двоичной системе счисления*** для представления числа применяются две цифры: 0 и 1. В ***восьмеричной системе счисления*** применяются цифры от 0 до 7. ***Шестнадцатеричная система счисления*** использует для представления чисел цифры от 0 до 9 и буквы латинского алфавита – A (10), B (11), C (12), D (13), E (14), F (15). Запись первых 16 положительных чисел в этих системах счисления представлена в табл. 1.1.

Таблица 1.1

Представление чисел в различных системах счисления

Восьмеричная	Двоичная	Шестнадцатеричная	Двоичная	Шестнадцатеричная	Двоичная
0	000	0	0000	8	1000
1	001	1	0001	9	1001
2	010	2	0010	A (10)	1010
3	011	3	0011	B (11)	1011
4	100	4	0100	C (12)	1100
5	101	5	0101	D (13)	1101
6	110	6	0110	E (14)	1110
7	111	7	0111	F (15)	1111

1.2.1. Арифметические операции над числами в разных системах счисления

Выполнение операций с двоичными числами рассмотрим на примерах.

Пример 1.4. Даны числа 101_2 и 11_2. Найти сумму этих чисел.

При сложении двух чисел, равных 1, в данном разряде получается 0, а единица переносится в старший разряд.

$$\begin{array}{r} 1\ 0\ 1 \\ +\quad 1\ 1 \\ \hline 1\ 0\ 0\ 0 \end{array}$$

Выполним сложение чисел: $1\ 0\ 0\ 0$. Для проверки переведем слагаемые в десятичную систему счисления: $101_2 = 5_{10}$, $11_2 = 3_{10}$, $1000_2 = 8_{10}$. Получаем $5 + 3 = 8$.

Пример 1.5. Даны числа 101_2 и 11_2. Найти разность этих чисел.

При вычитании из нуля единицы занимается единица из старшего ближайшего разряда, отличного от нуля. При этом единица, занятая в старшем разряде, даёт 2 единицы в младшем разряде и по единице во всех разрядах между старшим и младшим.

$$\begin{array}{r} 1\ 0\ 1 \\ -\quad 1\ 1 \\ \hline 1\ 0 \end{array}$$

Выполним вычитание: $1\ 0$. Для проверки переведем слагаемые в десятичную систему счисления: $101_2 = 5_{10}$, $11_2 = 3_{10}$, $10_2 = 2_{10}$. Получаем $5 - 3 = 2$.

Пример 1.6. Даны числа 11_2 и 10_2. Найти произведение этих чисел.

Операция умножения сводится к многократному сдвигу и сложению.

$$\begin{array}{r} 1\ 1 \\ \times\quad 1\ 0 \\ \hline 0\ 0 \\ 1\ 1\ \ \\ \hline 1\ 1\ 0 \end{array}$$

Выполним умножение: . Здесь $11_2 = 3_{10}$, $10_2 = 2_{10}$, $110_2 = 6_{10}$. Проверка: $3 \cdot 2 = 6$.

Арифметические операции в восьмеричной системе счисления

При сложении двух чисел, в сумме равных 8, в данном разряде получается 0, а единица переносится в старший разряд.

Пример 1.7. Даны числа 165_8 и 13_8. Найти сумму этих чисел.

$$\begin{array}{r} 165 \\ +\ 13 \\ \hline 200 \end{array}$$

Выполним сложение чисел: . Здесь $165_8 = 117_{10}$, $13_8 = 11_{10}$, $200_8 = 128_{10}$. Проверка: $117 + 11 = 128$.

При вычитании из меньшего числа большего, занимается единица из старшего ближайшего разряда, отличного от 0. При этом единица, занятая в старшем разряде, дает 8 в младшем разряде.

Пример 1.8. Даны числа 114_8 и 15_8. Найти разность этих чисел.

$$\begin{array}{r} 114 \\ -\ 15 \\ \hline 77 \end{array}$$

Выполним вычитание: , где $114_8 = 76_{10}$, $15_8 = 13_{10}$, $77_8 = 63_{10}$. Проверка: 76 – 13 = 63.

Арифметические операции в шестнадцатеричной системе счисления

При сложении двух чисел, в сумме равных 16, в данном разряде записывают 0, а единицу переносят в старший разряд.

Пример 1.9. Даны числа $1B5_{16}$ и 53_{16}. Найти сумму этих чисел.

$$\begin{array}{r} 1B5 \\ +\ 53 \\ \hline 208 \end{array}$$

Выполним сложение чисел: , где $1B5_{16} = 437_{10}$, $53_{16} = 83_{10}$, $208_{16} = 520_{10}$. Проверка: 437 + 83 = 520.

При вычитании из меньшего числа большего занимается единица из старшего ближайшего разряда, отличного от 0. При этом единица, занятая в старшем разряде, даёт 16 в младшем разряде.

Пример 1.10. Даны числа $11A_{16}$ и $2C_{16}$. Найти разность этих чисел.

$$\begin{array}{r} 11A \\ -\ 2C \\ \hline EE \end{array}$$

Выполним вычитание: , где $11A_{16} = 282_{10}$, $2C_{16} = 44_{10}$, $EE_{16} = 238_{10}$. Проверка: 282 – 44 = 238.

1.2.2. Перевод чисел из одной системы счисления в другую

Перевод в десятичную систему счисления целых чисел, представленных в другой системе счисления

Пример 1.11. Перевести число 1101_2 из двоичной системы счисления в десятичную.

Для перевода числа из любой системы счисления в десятичную необходимо разложить это число по степеням основания этой системы, затем каждую цифру числа умножить на основание системы счисления, возведенное в соответствующую степень.

$1101_2 = 1^3 1^2 0^1 1^0{}_2 = 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 8 + 4 + 0 + 1 = 13_{10}$.

Получаем: $1101_2 = 13_{10}$.

Примечание. При переводе важно помнить, что любое число в нулевой степени равно 1.

Пример 1.12. Перевести число 13_4 из четверичной системы счисления в десятичную.

Решение: $13_4 = 1^1 3^0{}_4 = 1 \cdot 4^1 + 3 \cdot 4^0 = 4 + 3 = 7_{10}$. Получаем: $13_4 = 7_{10}$.

Перевод в десятичную систему счисления действительных чисел, представленных в другой системе счисления

Пример 1.13. Дано число 101.11_2. Необходимо перевести его из двоичной системы счисления в десятичную.

Перевод действительного числа из любой системы счисления в десятичную осуществляется по тем же правилам, что и перевод целого числа.

$$101.11_2 = 1^2 0^1 1^0 . 1^{-1} 1^{-2}{}_{(2)} = 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 + 1 \cdot 2^{-1} + $$
$$+ 1 \cdot 2^{-2} = 4 + 0 + 1 + 1/2 + 1/4 = 5.75_{10}.$$

Перевод целых чисел из десятичной системы счисления в любую другую

Пример 1.14. Перевести число 13_{10} из десятичной системы счисления в двоичную.

Для перевода чисел из десятичной системы счисления в любую другую необходимо делить десятичное число на основание новой системы счисления, сохраняя при этом остатки от каждого деления. Деление продолжается до тех пор, пока результат деления не станет меньше делителя. Число формируется из полученных остатков (рис. 1.3), записанных в обратном порядке (от последнего к первому).

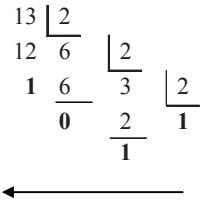


Рис. 1.3. Перевод числа из десятичной системы счисления в двоичную

13/2 = 6 (остаток 1); так как частное 6 больше делителя 2, то продолжаем выполнять деление. 6/2 = 3 (остаток 0); так как частное 3 больше делителя 2, то продолжаем выполнять деление. 3/2 = 1 (остаток 1); так как частное 1 меньше делителя 2, то записываем полученное число. Результат записываем справа налево. Получаем $13_{10} = 1101_2$.

Пример 1.15. Перевести число 7_{10} из десятичной системы счисления в четверичную.

$$\begin{array}{r|l} 7 & 4 \\ \hline 4 & 1 \\ \hline \mathbf{3} & \end{array}$$

Выполним деление: ⟵ . Получаем $7_{10} = 13_4$.

Перевод действительных чисел из десятичной системы счисления в любую другую

Пример 1.16. Перевести число 5.75_{10} из десятичной системы счисления в двоичную.

Перевод действительного числа из десятичной системы счисления в любую другую осуществляется по следующему алгоритму.

1. Целая часть действительного числа преобразуется де-

$$\begin{array}{r|l} 5 & 2 \\ \hline 4 & 2 \quad \vert \quad 2 \\ \hline \mathbf{1} & 2 \quad \vert \quad 1 \\ \hline & \mathbf{0} \end{array}$$

лением на основание новой системы счисления: ⟵. Получаем $5_{10} = 101_2$.

2. Теперь переводим дробную часть числа. Для этого умножаем дробную часть числа на 2. Если целая часть полученного числа четная или равна 0, то записываем 0 в дробной части действительного числа; если нечетная, то записываем 1. Умножения повторяются до тех пор, пока число не станет целым.

$0.75 \cdot 2 = 1.5$ – записываем 1 в дробной части числа 101.1_2;

$1.5 \cdot 2 = 3$ – записываем 1 в дробной части числа 101.11_2. Завершаем выполнение умножения, так как получено целое число.

Получаем окончательно: $5.75_{10} = 101.11_2$.

Перевод целых чисел из двоичной системы счисления в восьмеричную (шестнадцатеричную) и обратно

Так как основания восьмеричной и шестнадцатеричной систем счисления являются степенями двойки, то перевод чисел из этих систем счисления в двоичную и наоборот основан на методах триад и тетрад.

При переводе из двоичной системы счисления в восьмеричную необходимо число разбить на ***триады*** по три цифры справа налево.

Пример 1.17. Перевести число 11011_2 в восьмеричную систему счисления.

1. Разбиваем число 11011_2 на триады: $11\underline{011}_2$.
2. При необходимости следует добавить слева нули, чтобы получилась длина строки, кратная трем: $\underline{011}011_2$.
3. Из табл. 1.1 выпишем для каждой триады соответствующую цифру в восьмеричной системе счисления. В результате получим 33_8.

При переводе из двоичной системы счисления в шестнадцатеричную необходимо двоичное число разбить на ***тетрады*** по четыре цифры справа налево.

Пример 1.18. Перевести число 11011_2 в шестнадцатеричную систему счисления.

1. Разбиваем число 11011_2 на тетрады: $1\underline{1011}_2$.
2. При необходимости следует добавить слева нули, чтобы получилась длина строки, кратная 4: $0001\underline{1011}_2$.
3. Из табл. 1.1 выписать для каждой тетрады соответствующую цифру в шестнадцатеричной системе счисления: $1B_{16}$.

При переводе из восьмеричной системы счисления в двоичную необходимо для каждой цифры числа из табл. 1.1 выписать соответствующую триаду (слева направо).

Пример 1.19. Перевести число 147_8 в двоичную систему счисления.

Для цифры 1 запишем 001, для 4 – 100, для 7 – 111, получим $\underline{001}100\underline{111}_2$. Нули слева можно отбросить, получим окончательно $147_8 = \underline{1}100\underline{111}_2$.

При переводе из шестнадцатеричной системы счисления в двоичную необходимо для каждой цифры числа из табл. 1.1 выписать соответствующую тетраду (слева направо).

Пример 1.20. Перевести число $A11_{16}$ в двоичную систему счисления.

Для цифры А запишем 1010, для 1 – 0001. Соединяем тетрады, получим $A11_{16} = \underline{1010}0001\underline{0001}_2$.

При переводе восьмеричного числа в шестнадцатеричное (или обратно) необходимо перевести число из восьмеричной (шестнадцатеричной) системы счисления в двоичную, а затем в шестнадцатеричную (восьмеричную) систему счисления.

? Вопросы и упражнения для самоконтроля

1. Чем отличается позиционная система счисления от непозиционной? Приведите примеры позиционной и непозиционной систем счисления.
2. Сколько цифр используется для записи чисел в двоичной, восьмеричной, шестнадцатеричной системах счисления?
3. Опишите алгоритм перевода числа из десятичной системы счисления в любую другую систему.
4. Как перевести число из двоичной, восьмеричной, шестнадцатеричной системы счисления в десятичную?
5. Как выполняется перевод числа из двоичной системы счисления в восьмеричную или шестнадцатеричную и обратно?
6. Выполните перевод чисел 12_{10} и 5_{10} в двоичную, восьмеричную и шестнадцатеричную системы счисления. Выполните сложение и вычитание чисел в каждой системе счисления.

1.3. Кодирование информации и представление данных в памяти компьютера

Код – набор условных обозначений для представления информации. ***Кодирование*** – процесс представления информации в виде кода. Процедура кодирования применяется к дискретной информации. Аналоговая (непрерывная) информация, к которой, в частности, относятся графическая, аудио- и видеоинформация, должна быть сначала каким-либо образом преобразована в дискретную. Этот процесс

называется ***оцифровкой информации***. Данные в памяти компьютера представляются в виде кода, который состоит из единиц и нулей в разной последовательности. Для разных видов информации используются различные способы кодирования.

1.3.1. Кодирование числовой информации

Есть два основных формата представления чисел в памяти компьютера. Один из них используется для кодирования целых чисел, второй (так называемое представление числа в формате с плавающей точкой) используется для задания некоторого подмножества действительных чисел.

Целые числа могут представляться в компьютере со знаком или без знака. ***Целые числа без знака*** обычно занимают в памяти компьютера один или два байта. В однобайтовом формате принимают значения от 00000000_2 до 11111111_2. В двухбайтовом формате – от 0000000000000000_2 до 1111111111111111_2. Диапазоны представления чисел без знака приведены в табл. 1.2.

Таблица 1.2

Диапазоны значений целых чисел без знака

Формат числа	Диапазон	
	Запись с порядком	Обычная запись
1 байт	$0 \ldots 2^8-1$	0 ... 255
2 байта	$0 \ldots 2^{16}-1$	0 ... 65535

Например, число $72_{10} = 1001000_2$ в **однобайтовом** формате будет храниться в памяти в следующем виде:

Номер разряда	7	6	5	4	3	2	1	0
Биты числа	0	1	0	0	1	0	0	0

Целые числа со знаком обычно занимают в памяти компьютера один, два или четыре байта, при этом *самый левый (старший) разряд содержит информацию о знаке числа*. Диапазоны представления целых чисел со знаком приведены в табл. 1.3.

Таблица 1.3

Диапазоны значений целых чисел со знаком

Формат числа	Диапазон	
	Запись с порядком	Обычная запись
1 байт	$-2^7 \dots 2^7-1$	−128 ... 127
2 байта	$-2^{15} \dots 2^{15}-1$	−32768 ... 32767
4 байта	$-2^{31} \dots 2^{31}-1$	−2147483648 ... 2147483647

Рассмотрим особенности записи целых чисел со знаком на примере **однобайтового формата**, при котором для знака отводится один разряд, а для цифр абсолютной величины числа – семь разрядов. Применяются три формы записи (кодирования) целых чисел со знаком: прямой код, обратный код, дополнительный код. Последние две формы применяются особенно широко, так как позволяют упростить конструкцию арифметико-логического устройства компьютера в результате сведения всех арифметических операций к операциям сложения и сдвига.

Положительные целые числа в прямом, обратном и дополнительном кодах изображаются одинаково – двоичными кодами с цифрой 0 в знаковом разряде. Например, число 13_{10} (в двоичной системе счисления 1101) будет представлено в следующем виде:

0	0	0	0	1	1	0	1

Знаковый разряд здесь содержит 0, поскольку кодируется положительное число.

☑ В дальнейшем изложении будем представлять коды чисел как восьмиразрядные последовательности нулей и единиц, в которых знаковый разряд отделяется символом «запятая»: 0,0001101.

Отрицательные целые числа в прямом, обратном и дополнительном кодах имеют разное изображение.

Прямой код. В знаковый разряд помещается цифра 1, а в разряды цифровой части числа – двоичный код его абсолютной величины. Например, для числа -13_{10} прямой код числа будет иметь вид: 1,0001101.

Обратный код отрицательного числа получается из прямого кода заменой всех двоичных цифр числа на противоположные (1 на 0, 0 на 1). В знаковый разряд заносится единица. Так, для числа -13_{10} обратный код будет равен 1,1110010.

Дополнительный код отрицательного числа образуется из обратного кода прибавлением к младшему разряду единицы. Для числа -13_{10} дополнительный код равен 1,1110011.

Пример 1.21. Найти прямой, обратный и дополнительный коды в однобайтовом представлении для чисел 25_{10} и -25_{10}.

Выполнив перевод чисел в двоичную систему счисления, получим: $25_{10} = 11001_2$; $-25_{10} = -11001_2$. Запишем прямые коды чисел: 0,0011001 – для числа 25, 1,0011001 – для числа -25. Прямой код положительного числа 25 равен обратному и дополнительному кодам.

Для отрицательного числа -25 обратный код получается из прямого инверсией всех разрядов (кроме знакового). Дополнительный код получается из обратного кода прибавлением к двоичному числу единицы:

	25_{10}	-25_{10}
Прямой код	0,0011001	1,0011001
Обратный код	0,0011001	1,1100110
Дополнительный код	0,0011001	1,1100111

Пример 1.22. Задан дополнительный код числа в однобайтовом представлении: 1,1011100. Найти число в десятичной системе счисления.

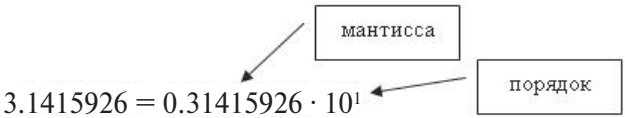
Поскольку первый разряд содержит число 1, делаем вывод, что число отрицательное. Определяем обратный код числа: из числа 1011100 (без первого разряда) вычитаем 1, получаем 1011011. Обратный код числа равен 1,1011011. Определяем прямой код числа. Для этого выполняем инверсию всех разрядов обратного кода. Получаем 1,0100100.

Искомое число в двоичной системе счисления равно – 100100. Переводим число из двоичной системы счисления в десятичную, получаем –36 (два в пятой степени плюс два в квадрате).

Пример 1.23. Задан дополнительный код числа в однобайтовом представлении: 0,1011100. Найти число в десятичной системе счисления.

Поскольку первый разряд содержит цифру 0, делаем вывод, что число положительное. Это означает, что дополнительный код равен обратному коду и равен прямому коду. Число в двоичной системе счисления получаем из записи прямого кода. Это число 1011100. Переводим число из двоичной системы счисления в десятичную, получаем +92 (два в 6 степени плюс два в 4 степени плюс два в 3 степени плюс два в квадрате).

Для кодирования ***действительных чисел*** используют 80-разрядное кодирование. При этом число предварительно преобразуется в нормализованную форму записи, состоящую из мантиссы и порядка.



Мантисса – это число, меньшее 1, не содержащее цифры 0 в первом разряде после десятичной точки, а ***порядок*** – степень, в которую нужно возвести число 10, чтобы в результате умножения мантиссы на 10 в этой степени было получено исходное число. Большую часть из 80 бит отводят для хранения мантиссы (вместе со знаком) и некоторое фиксированное количество разрядов отводят для хранения порядка со знаком. При этом и мантисса, и порядок хранятся как целые числа.

1.3.2. Кодирование текстовой и графической информации

Поскольку текст изначально дискретен (он состоит из отдельных символов), для компьютерного представления текстовой информации используется способ, при котором текст представляется в виде набора чисел – кодов символов, его составляющих. Для этого используются так называемые кодовые таблицы символов, в которых каждому коду символа ставится в соответствие изображение символа.

Во всем мире в качестве стандарта принята таблица ASCII (American Standard Code for Information Interchange – Американский стандартный код для обмена информацией). Для хранения двоичного кода одного символа выделен 1 байт = 8 бит. Учитывая, что каждый бит принимает значение 0 или 1, количество их возможных сочетаний в байте равно $2^8 = 256$. Значит, с помощью 1 байта можно получить 256 разных двоичных кодовых комбинаций и отобразить с их помощью 256 различных символов. Эти комбинации и составляют таблицу ASCII.

Первые 128 символов таблицы с кодами от 0 до 127 – цифры, буквы латинского алфавита, управляющие символы. Коды от 128 до 255 обычно содержат буквы национальных алфавитов, графические знаки. Первая половина таблицы является неизменной, а вторая – переменной. Восьмибитовая кодировка является недостаточной для кодировки всех символов расширенных алфавитов. Все препятствия могут быть сняты при переходе на 16-битовую кодировку Unicode, допускающую 65536 кодовых комбинаций.

Форму представления на экране дисплея графического изображения, состоящего из отдельных точек (пикселей), называют растровой. Растровые изображения представляют собой однослойную сетку точек, называемых пикселями (*pixel* – от англ. *picture element*). Код пикселя содержит информацию о его цвете.

Для ***черно-белого изображения*** (без полутонов) пиксель может принимать только два значения: белый и черный (светится – не светится), а для его кодирования достаточно одного бита памяти: 1 – белый, 0 – черный.

Пиксель на цветном дисплее может иметь различную окраску, поэтому одного бита на пиксель недостаточно. Для ***кодирования 4-цветного изображения*** требуются два бита на пиксель, поскольку два бита могут принимать 4 различных состояния. Может использоваться, например, такой вариант кодировки цветов: 00 – черный, 10 – зеленый, 01 – красный, 11 – коричневый.

Если говорить о кодировании цветных графических изображений, то нужно рассмотреть принцип декомпозиции

произвольного цвета на основные составляющие. При выводе изображений на экран монитора используется принцип декомпозиции на основе цветовой модели RGB. Суть его заключается в следующем: известно, что любой цвет можно представить в виде комбинации трех цветов: красного (Red, R), зеленого (Green, G), синего (Blue, B). Другие цвета и их оттенки получаются за счет наличия или отсутствия этих составляющих. По первым буквам основных цветов система и получила свое название – RGB.

Данная цветовая модель является аддитивной, то есть любой цвет можно получить сочетанием основных цветов в различных пропорциях. При наложении одного компонента основного цвета на другой яркость суммарного излучения увеличивается. Если совместить все три компонента, то получим ахроматический серый цвет, при увеличении яркости которого происходит приближение к белому цвету.

При ***256 градациях тона*** (каждая точка кодируется 3 байтами) минимальные значения RGB (0, 0, 0) соответствуют черному цвету, а белому – максимальные с координатами (255, 255, 255). Чем больше значение байта цветовой составляющей, тем этот цвет ярче. Например, темно-синий цвет кодируется тремя байтами (0, 0, 128), а ярко-синий – (0, 0, 255).

Различают несколько режимов представления цветной графики: полноцветный (True Color), High Color, индексный.

При ***полноцветном режиме*** для кодирования яркости каждой из составляющих используют по 256 значений (восемь двоичных разрядов), то есть на кодирование цвета одного пикселя (в системе RGB) надо затратить $8 \times 3 = 24$ разряда. Это позволяет однозначно определять 16,5 млн цветов. Это довольно близко к чувствительности человеческого глаза.

Режим High Color – это кодирование при помощи 16-разрядных двоичных чисел. Уменьшается количество двоичных разрядов при кодировании каждой точки, но при этом значительно уменьшается диапазон кодируемых цветов (65536 цветов).

При ***индексном кодировании*** цвета можно передать всего лишь 256 цветовых оттенков. Каждый цвет кодируется при помощи восьми бит данных. Но так как 256 значений

не передают весь диапазон цветов, доступный человеческому глазу, то подразумевается, что к графическим данным прилагается палитра (справочная таблица), без которой воспроизведение будет неадекватным: море может получиться красным, а листья – синими. Сам код точки растра в данном случае означает не сам по себе цвет, а только его номер (индекс) в палитре. Отсюда и название режима – индексный.

Количество различных отображаемых цветов K и битовая глубина b (число разрядов, используемых для кодировки цвета) связаны формулой:

$$K = 2^b, \tag{1.4}$$

где K – количество цветов; b – битовая глубина.

☑ Объем памяти, необходимой для хранения графического изображения, занимающего весь экран, равен произведению количества пикселей (разрешающей способности) на число бит, кодирующих одну точку. Объем графического файла в битах определяется как произведение количества пикселей $N \times M$ на разрядность цвета C (битовую глубину): $V = N \times M \times C$.

Например, при разрешении 640×480 и количестве цветов 16 (4 бита) объем памяти равен: 640 × 480 × 4 = 1 228 800 (бит) = = 153 600 (байт) = 150 (Кбайт).

1.3.3. Кодирование аудио- и видеоинформации

Из физики известно, что звук – это колебания воздуха. Если преобразовать звук в электрический сигнал (например, с помощью микрофона), то видно плавно изменяющееся с течением времени напряжение. Для компьютерной обработки такой (непрерывный) сигнал нужно каким-то образом преобразовать в последовательность двоичных чисел. Это можно сделать, например, так: измеряется напряжение через равные промежутки времени, полученные значения записываются в память компьютера. Этот процесс называется дискретизацией (или оцифровкой), а устройство, выполняющее его, – аналого-цифровым преобразователем (АЦП). Чтобы воспроизвести закодированный таким образом звук, нужно сделать обратное

преобразование (для этого служит цифро-аналоговый преобразователь – ЦАП), а затем сгладить получившийся ступенчатый сигнал.

Чем выше частота дискретизации и чем больше разрядов отводится для каждого отсчета, тем точнее будет представлен звук, но при этом увеличивается и размер звукового файла. В настоящее время при записи звука в мультимедийных технологиях применяются частоты 8, 11, 22 и 44 кГц. Так, частота дискретизации 44 килогерца означает, что одна секунда непрерывного звучания заменяется набором из сорока четырех тысяч отдельных отсчетов сигнала. Чем выше частота дискретизации, тем лучше качество оцифрованного звука. Поэтому в зависимости от характера звука, требований, предъявляемых к его качеству и объему занимаемой памяти, выбирают некоторые компромиссные значения.

Качество преобразования звука в цифровую форму определяется не только частотой дискретизации, но и количеством битов памяти, отводимых на запись кода одного отсчета. Этот параметр принято называть разрядностью преобразования. В настоящее время обычно используется разрядность 8, 16 и 24 бит. На описанных выше принципах основывается формат WAV (от WAVeform-audio – волновая форма аудио) кодирования звука. Получить запись звука в этом формате можно от подключаемых к компьютеру микрофона, проигрывателя, магнитофона, телевизора и других стандартно используемых устройств работы со звуком. Однако формат WAV требует очень много памяти. Так, при записи стереофонического звука с частотой дискретизации 44 килогерца и разрядностью 16 бит на одну минуту записи требуется около десяти миллионов байтов памяти.

Издавна для музыки используется нотная запись. В ней специальными символами указывается, какой высоты звук, на каком инструменте и как сыграть. Фактически ее можно считать алгоритмом для музыканта, записанным на особом формальном языке. В 1983 году ведущие производители компьютеров и музыкальных синтезаторов разработали стандарт, определивший такую систему кодов. Он получил название MIDI. Такая система кодирования позволяет записать далеко не всякий звук, она подходит только для инструментальной

музыки. Но есть у нее и преимущества: чрезвычайно компактная запись, естественность для музыканта (практически любой MIDI-редактор позволяет работать с музыкой в виде обычных нот), легкость замены инструментов, изменения темпа и тональности мелодии.

Есть и другие, чисто компьютерные, форматы записи музыки. Среди них – формат MP3, позволяющий с очень большим качеством и степенью сжатия кодировать музыку, при этом вместо 18–20 музыкальных композиций на стандартном компакт-диске (CD-ROM) помещается около 200. Одна песня занимает примерно 3,5 Mb, что позволяет пользователям сети Интернет легко обмениваться музыкальными композициями.

Видеоинформация включает последовательность кадров и звуковое сопровождение, поэтому кодирование видеоинформации – еще более сложная проблема, чем кодирование звуковой информации, так как нужно позаботиться не только о дискретизации непрерывных движений, но и о синхронизации изображения со звуковым сопровождением. В настоящее время для этого используется формат, который называется AVI (Audio-Video Interleaved – чередующиеся аудио и видео).

? Вопросы и упражнения для самоконтроля

1. Как представляются в памяти компьютера целые числа без знака?
2. Как связаны между собой формат представления чисел в памяти компьютера и диапазон возможных (представимых) значений?
3. Опишите способы представления в памяти компьютера целых чисел со знаком.
4. Как представляются в памяти компьютера числа с дробной частью?
5. Как кодируются текстовые данные в памяти компьютера? Что собой представляет таблица ASCII кодов?
6. Опишите способы кодирования графических данных. От чего зависит количество возможных цветов точки?
7. Какие принципы используются при кодировании звуковых данных?

1.4. Логические основы устройства компьютеров

1.4.1. Основы алгебры логики

Алгебра логики (булева алгебра) изучает высказывания, рассматриваемые со стороны их логических значений (истинности или ложности), и логические операции над ними. Основным предметом алгебры логики являются ***высказывания***. Под ***высказыванием*** понимается имеющее смысл языковое выражение, относительно которого можно утверждать, что оно либо истинно, либо ложно.

Примеры высказываний:

- «5 – простое число». Это высказывание является истинным.
- «роза – цветок». Это высказывание является истинным.
- «3 + 5 = 9». Это высказывание является ложным.

Построение из данных высказываний (или из данного высказывания) нового высказывания называется ***логической операцией***. Знаки логических операций называются ***логическими связками***. Например, из высказываний «$x > 2$», «$x < 3$» при помощи связки ***и*** можно получить высказывание «$x > 2$ и $x < 3$».

Истинность или ложность получаемых таким образом высказываний зависит от истинности и ложности исходных высказываний и соответствующей трактовки связок как операций над высказываниями.

Одной из основных операций алгебры логики является операция ***отрицания***. Отрицание высказывания A (т. е. не A) обозначается $\overline{A}$ и читается: «отрицание A», «не A» или «A с чертой».В табл. 1.4 приведены основные бинарные логические операции и связки.

Рассмотрим примеры использования логических операций.

Инверсия. Дано высказывание **А** = «Киев – столица Франции». Тогда **не А** = «**неверно, что** Киев – столица Франции».

Конъюнкция. Результатом конъюнкции для высказываний **А** и **В** будет истина только тогда, когда истинны одновременно оба высказывания. Например, даны высказывания **А** = «Москва – столица России» и **В** = «Рим – столица Италии». Сложное высказывание **А** $\wedge$ **В** = «Москва – столица

России **и** Рим – столица Италии» истинно, так как истинны оба высказывания.

Таблица 1.4

Основные бинарные логические операции и связки

Обозначение операции	Другие обозначения операции	Название логической операции	Логические связки
$A \wedge B$	A & B A · B A B	**Конъюнкция**, логическое умножение, логическое «и»	A **и** B
$A \vee B$	A + B	**Дизъюнкция**, логическое сложение, логическое «или»	A **или** B
$A \to B$	$A \supseteq B$ $A \Rightarrow B$	**Импликация**, логическое следование	**Если** A, **то** B
$A \oplus B$	$A \triangle B$	**Разделительная дизъюнкция**, разделительное «или»	**Либо** A, **либо** B
A ~ B	A ≡ B $A \leftrightarrow B$ $A \Leftrightarrow B$	**Эквиваленция**, тождественность, равнозначность	A **тогда и только тогда, когда** B
A | B	$\overline{A \wedge B}$	Штрих Шеффера, **антиконъюнкция**	**Неверно, что** A **и** B
$A \downarrow B$	$\overline{A \vee B}$	Стрелка Пирса, **антидизъюнкция**	**Неверно, что** A **или** B

Примечание. ***A*** и ***B*** являются высказываниями.

Дизъюнкция. Результатом дизъюнкции для высказываний **A** и **B** будет истина, когда истинно хотя бы одно из высказываний. Например, даны высказывания **A** = «2 + 3 = 5» и **B** = «3 + 3 = 5». Сложное высказывание $\mathbf{A} \vee \mathbf{B}$ = «2 + 3 = 5 **или** 3 + 3 = 5» истинно, так как истинно высказывание **A**.

Эквиваленция. Результатом эквиваленции для высказываний **A** и **B** будет истина только тогда, когда истинны или ложны одновременно оба высказывания. Например, даны высказывания **A** = «2 +2 = 7» и **B** = «1 – 8 = 5». Сложное высказывание **A~B** = «2 + 2 = 7 **тогда и только тогда, когда** 1 – 8 = 5» истинно, так как оба высказывания ложны.

Импликация. Результатом импликации для высказывания **A→B** будет ложь только тогда, когда первое высказывание (**A**) истинно, а второе (**B**) ложно. При этом **A** – предпосылка, а **B** – следствие. В остальных случаях результатом операции всегда будет истина. Например, даны высказывания **A** = «2 + 2 = 4» и **B** = «1 – 8 = 5». Сложное высказывание **A→B** = «**если** 2 + 2 = 4, **то** 1 – 8 = 5» ложно, так как высказывание **A** истинно, а **B** – ложно.

Антиконъюнкция. Результатом антиконъюнкции для высказываний **A** и **B** будет ложь только тогда, когда оба высказывания истинны. В остальных случаях результатом операции будет истина. Например, даны высказывания **A** = «Москва – столица России» и **B** = «Рим – столица Италии». Сложное высказывание **A | B** = «**неверно, что** Москва – столица России **и** Рим – столица Италии» ложно, так как истинны оба высказывания.

Антидизъюнкция. Результатом антидизъюнкции для высказываний **A** и **B** будет истина только тогда, когда оба высказывания ложны. В остальных случаях результатом операции будет ложь. Например, даны высказывания **A** = «Рим – столица России» и **B** = «Москва – столица Италии». Сложное высказывание **A ↓ B** = «**неверно, что** Рим – столица России **или** Москва – столица Италии» истинно, так как ложны оба высказывания.

Связки и частица «не» рассматриваются в алгебре логики как операции над логическими переменными, принимающими значения 0 (ложь / false) и 1 (истина / true), и результатом применения этих операций также являются логические значения 0 или 1.

☑ В алгебре логики логические операции описываются при помощи ***таблиц истинности***. Таблицы истинности описывают результаты логических операций на всех

возможных наборах значений логических переменных. Табл. 1.5 представляет таблицу истинности для операции ***отрицания*** (***инверсия***). Можно видеть, что если переменная **A = 1** (истина), то после применения операции инверсии ее значение станет равным **0 (**ложь).

Таблица 1.5

Таблица истинности для операции отрицания

A	**не A**
0	1
1	0

В табл. 1.6 представлены все возможные наборы значений переменных **A** и **B** и значения бинарных (связывающих 2 переменные) логических операций на этих наборах.

Таблица 1.6

Таблица истинности для бинарных логических операций

A	**B**	$\wedge$	$\vee$	$\rightarrow$	$\oplus$	~	$\mid$	$\downarrow$
0	**0**	0	0	1	0	1	1	1
0	**1**	0	1	1	1	0	1	0
1	**0**	0	1	0	1	0	1	0
1	**1**	1	1	1	0	1	0	0

☑ С помощью таблиц истинности логических операций можно вычислять значения сложных логических выражений. При выполнении вычислений необходимо учитывать порядок выполнения логических операций: 1 – отрицание, 2 – конъюнкция, 3 – дизъюнкция, 4 – импликация, 5 – эквиваленция. Для изменения установленного порядка используются круглые скобки.

Например, в выражении $A \rightarrow B \vee A \sim \overline{B}$ сначала выполнится операция $\overline{B}$, затем – $B \vee A$, затем – импликация, в последнюю очередь – эквиваленция. В выражении $A \rightarrow B \vee (A \sim \overline{B})$ сначала выполнится отрицание, затем – эквиваленция, затем – дизъюнкция, в последнюю очередь – импликация.

Любое логическое выражение можно записать, используя только операции отрицания, конъюнкции и дизъюнкции. При этом используются следующие доказанные факты:

- логическую операцию «импликация» можно выразить через операции отрицания и дизъюнкции: $A \rightarrow B = \overline{A} \vee B$;
- логическую операцию «эквиваленция» можно выразить через операции отрицания, дизъюнкции и конъюнкции: $A \sim B = (\overline{A} \vee B) \wedge (\overline{B} \vee A)$.

Примеры вычисления значений логических выражений с помощью таблиц истинности будут рассмотрены в описании практических работ.

1.4.2. Логические элементы. Построение логических схем

На этапе конструирования микросхем компьютера можно описать функционирование элементов микросхем с помощью логических функций и, используя законы алгебры логики, значительно упростить эти функции, оптимизируя тем самым конструкцию микросхем.

Логический элемент компьютера – это часть электронной логической схемы, которая реализует элементарную логическую формулу. Логическими элементами компьютеров являются электронные схемы «И», «ИЛИ», «НЕ», «И-НЕ», «ИЛИ-НЕ». С помощью этих схем можно реализовать любую логическую формулу, описывающую работу устройств компьютера. Каждый логический элемент имеет свое условное обозначение, которое выражает его логическую формулу, но не указывает на то, какая именно электронная схема в нем реализована. Это упрощает запись и понимание сложных логических схем.

Схема «И» реализует конъюнкцию двух или более логических значений. Условное обозначение структурной схемы «И» представлено на рис. 1.4 слева. На выходе схемы «И» значение «1» будет тогда и только тогда, когда на всех входах будут «1». Если хотя бы на одном входе будет «0», на выходе также будет «0». Операция конъюнкции на функциональных схемах обозначается знаком «&» (читается «амперсанд»).

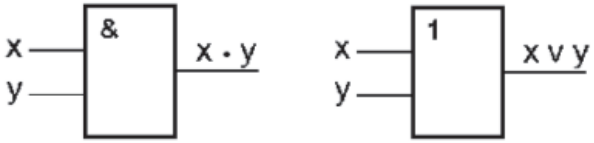


Рис. 1.4. Схема «И» (слева) и схема «ИЛИ» (справа)

Схема «ИЛИ» реализует дизъюнкцию двух логических значений. Условное обозначение схемы «ИЛИ» представлено на рис. 1.4 справа. На выходе схемы «ИЛИ» значение «0» будет тогда и только тогда, когда на всех входах будут «0». Когда хотя бы на одном входе будет «1», на выходе также будет «1». Операция дизъюнкции на функциональных схемах обозначается знаком «1».

Схема «НЕ» (инвертор) реализует операцию отрицания. Условное обозначение схемы «НЕ» представлено на рис. 1.5. Если на входе схемы – «0», то на выходе будет «1». Когда на входе – «1», на выходе будет «0».

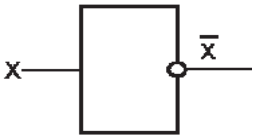


Рис. 1.5. Схема «НЕ»

Схема «И-НЕ» состоит из элемента «И» и инвертора и осуществляет отрицание результата схемы «И». Условное обозначение схемы «И-НЕ» представлено на рис. 1.6 слева. На выходе схемы «И-НЕ» значение «0» будет тогда и только тогда, когда на всех входах будут «1».

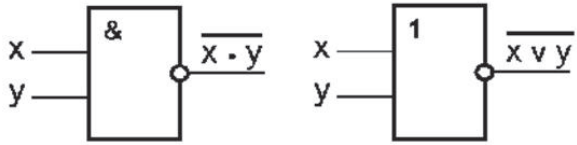


Рис. 1.6. Схема «И-НЕ» (слева) и схема «ИЛИ-НЕ» (справа)

Схема «ИЛИ-НЕ» состоит из элемента «ИЛИ» и инвертора и осуществляет отрицание результата схемы «ИЛИ». Условное обозначение схемы «ИЛИ-НЕ» представлено на рис. 1.6 справа. На выходе схемы «ИЛИ-НЕ» значение «1» будет тогда и только тогда, когда на всех входах будут «0».

Рассмотрим описание функционирования простейшего элемента микросхемы на примере триггера. ***Триггер*** – это электронная схема, широко применяемая в регистрах компьютера для надежного запоминания одного разряда двоичного кода. Триггер имеет два устойчивых состояния, одно из которых соответствует двоичной единице, а другое – двоичному нулю.

Рассмотрим принцип функционирования триггера на примере RS-триггера. RS-триггер имеет два входа (R – установка 0 (от слова reset), S – установка 1 (от слова *set*)) и два выхода (Q – прямой, $\overline{Q}$ – обратный (инверсный)). *Состояние триггера определяется состоянием прямого выхода*.

На рис. 1.7 показана реализация RS-триггера с помощью логических элементов «ИЛИ-НЕ». Пусть R = 0, S = 1, т. е. на вход S подан сигнал, а на входе R сигнал отсутствует. Нижний логический элемент выполняет логическую функцию «ИЛИ-НЕ», т. е. «1» на любом его входе приводит к тому, что на его выходе будет логический ноль ($\overline{Q} = 0$). На выходе Q будет «1» (Q = 1), так как на оба входа верхнего элемента поданы нули (один ноль – с входа R, другой – с выхода $\overline{Q}$). Триггер находится в единичном состоянии. Если теперь убрать сигнал установки, т. е. установить комбинацию R = 0, S = 0, то на выходе Q ситуация не изменится. Несмотря на то что на нижний вход нижнего логического элемента будет поступать «0», на его верхний вход поступает «1» с выхода верхнего логического элемента. Триггер будет находиться в устойчивом состоянии хранения «1» до тех пор, пока на вход R не поступит сигнал сброса (R = 1).

Пусть теперь R=1, S=0. Тогда Q = 0, а $\overline{Q} = 1$, т. е. триггер переключился в состояние «0». Если после этого убрать сигнал сброса, т. е. установить R = 0, S = 0, то триггер не изменит своего состояния и будет хранить «0» до тех пор, пока на вход S не поступит сигнал установки S = 1. Таким образом, триггер

всегда хранит один двоичный разряд (0 или 1). Для запоминания байта нужно 8 триггеров, для запоминания 1 килобайта, соответственно, $8 \cdot 2^{10} = 8192$ триггера. Современные микросхемы памяти содержат миллионы триггеров.

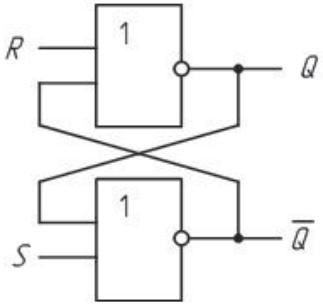


Рис. 1.7. Схема простейшего RS-триггера

? Вопросы и упражнения для самоконтроля

1. Что является предметом изучения алгебры логики?
2. Перечислите основные логические операции.
3. Для чего используются таблицы истинности?
4. Через какие логические операции можно реализовать импликацию и эквиваленцию? Запишите соответствующие формулы.
5. Как аппарат алгебры логики можно использовать для описания схем функционирования элементов компьютера?
6. Как изображаются логические схемы «НЕ», «И», «ИЛИ», «И-НЕ», «ИЛИ-НЕ»?
7. Опишите назначение и схему функционирования RS-триггера.

1.5. Практические работы

Практическая работа 1.1. Представление числовых данных в разных системах счисления

Цель работы – научиться:

- выполнять перевод чисел из одной системы счисления в другую;

• выполнять арифметические действия с числами в двоичной, восьмеричной, шестнадцатеричной системах счисления.

Порядок выполнения работы

1. Изучить теоретический материал из раздела 1.2 данного пособия.
2. Выполнить задание, выбрав вариант из табл. 1.7.
3. Оформить отчет о работе.

Требования к содержанию и оформлению отчета о работе

☑ Отчет выполняется письменно в тетради, он должен содержать тему работы, условие и ход решения каждой задачи.

Таблица 1.7

Варианты заданий

Номер варианта	Формулировка задания
1	**1.** Выполнить действия в двоичной системе счисления: a) 1100 · 0111; b) 11001 + 1011; c) 111101 – 1011. **2.** Перевести числа из одной системы счисления в другую: a) число 123 из десятичной системы счисления в двоичную; b) число 1100101.001 из двоичной системы счисления в десятичную; c) число 110100100010 из двоичной системы счисления в шестнадцатеричную
2	**1.** Выполнить действия в двоичной системе счисления: a) 11100101 – 100011; b) 11001101 + 1111; c) 1001 · 1101. **2.** Перевести числа из одной системы счисления в другую: a) число 141 из десятичной системы счисления в двоичную; b) число 101111.001 из двоичной системы счисления в десятичную; c) число 511 из восьмеричной системы счисления в шестнадцатеричную

Продолжение табл. 1.7

Номер варианта	Формулировка задания
3	**1.** Выполнить действия в двоичной системе счисления: a) 11010101 + 110011; b) 11001101 – 100111; c) 1111 · 1001. **2.** Перевести числа из одной системы счисления в другую: a) число 153 из десятичной системы счисления в двоичную; b) число 11011001.01 из двоичной системы счисления в десятичную; c) число 404 из шестнадцатеричной системы счисления в восьмеричную
4	**1.** Выполнить действия в двоичной системе счисления: a) 10000001 + 111011; b) 11011101 – 111111; c) 1110 · 1101. **2.** Перевести числа из одной системы счисления в другую: a) число 131 из десятичной системы счисления в двоичную; b) число 1101001.11 из двоичной системы счисления в десятичную; c) число 351 из восьмеричной системы счисления в двоичную
5	**1.** Выполнить действия в двоичной системе счисления: a) 10101101 + 100011; b) 110101 – 1111; c) 1011 · 1101. **2.** Перевести числа из одной системы счисления в другую: a) число 33 из десятичной системы счисления в двоичную; b) число 1101001.11 из двоичной системы счисления в десятичную; c) число 1101000101 из двоичной системы счисления в восьмеричную

Продолжение табл. 1.7

Номер варианта	Формулировка задания
6	**1.** Выполнить действия в двоичной системе счисления: a) 11100101 + 10011; b) 1100110 – 1111; c) 1011 · 101. **2.** Перевести числа из одной системы счисления в другую: a) число 77 из десятичной системы счисления в двоичную; b) число 1001011.1 из двоичной системы счисления в шестнадцатеричную; c) число 142 из восьмеричной системы счисления в шестнадцатеричную
7	**1.** Выполнить действия в двоичной системе счисления: a) 11100101 + 100011; b) 11001101 – 1111; c) 1011 · 1001. **2.** Перевести числа из одной системы счисления в другую: a) число 59 из десятичной системы счисления в двоичную; b) число 111101.001 из двоичной системы счисления в десятичную; c) число 440 из шестнадцатеричной системы счисления в восьмеричную
8	**1.** Выполнить действия в двоичной системе счисления: a) 11100101 + 100011; b) 11001101 – 1111; c) 1011 · 1001. **2.** Перевести числа из одной системы счисления в другую: a) число 106 из десятичной системы счисления в двоичную; b) число 1101001.01 из двоичной системы счисления в десятичную; c) число 11100110101 из двоичной системы счисления в шестнадцатеричную

Продолжение табл. 1.7

Номер варианта	Формулировка задания
9	**1.** Выполнить действия в двоичной системе счисления: a) 11100101 + 100011; b) 11001101 – 1111; c) 1011 · 1001. **2.** Перевести числа из одной системы счисления в другую: a) число 115 из десятичной системы счисления в двоичную; b) число 1010101.11 из двоичной системы счисления в десятичную; c) число 306 из восьмеричной системы счисления в шестнадцатеричную
10	**1.** Выполнить действия в двоичной системе счисления: a) 11100101 + 100011; b) 11001101 – 1111; c) 1011 · 1001. **2.** Перевести числа из одной системы счисления в другую: a) число 88 из десятичной системы счисления в двоичную; b) число 1011101.11 из двоичной системы счисления в десятичную; c) число 1035 из восьмеричной системы счисления в двоичную
11	**1.** Выполнить действия в двоичной системе счисления: a) 11100101 + 100011; b) 11001101 – 1111; c) 1011 · 1001. **2.** Перевести числа из одной системы счисления в другую: a) число 91 из десятичной системы счисления в двоичную; b) число 1101101.101 из двоичной системы счисления в десятичную; c) число 648 из шестнадцатеричной системы счисления в восьмеричную

Продолжение табл. 1.7

Номер варианта	Формулировка задания
12	**1.** Выполнить действия в двоичной системе счисления: a) 11100101 + 100011; b) 10001001 – 1001; c) 1001 · 1011. **2.** Перевести числа из одной системы счисления в другую: a) число 55 из десятичной системы счисления в двоичную; b) число 100110.01 из двоичной системы счисления в десятичную; c) число 442 из восьмеричной системы счисления в шестнадцатеричную
13	**1.** Выполнить действия в двоичной системе счисления: a) 11100101 + 100011; b) 11001101 – 1111; c) 1011 · 1001. **2.** Перевести числа из одной системы счисления в другую: a) число 111 из десятичной системы счисления в двоичную; b) число 110111 из двоичной системы счисления в десятичную; c) число 201 из восьмеричной системы счисления в шестнадцатеричную
14	**1.** Выполнить действия в двоичной системе счисления: a) 11100101 + 100011; b) 11001101 – 1111; c) 1011 · 1001. **2.** Перевести числа из одной системы счисления в другую: a) число 107 из десятичной системы счисления в двоичную; b) число 111010111 из двоичной системы счисления в десятичную; c) число 1100110101 из двоичной системы счисления в шестнадцатеричную

Окончание табл. 1.7

Номер варианта	Формулировка задания
15	**1.** Выполнить действия в двоичной системе счисления: a) 11100101 + 100011; b) 11001101 – 1111; c) 1011 · 1001. **2.** Перевести числа из одной системы счисления в другую: a) число 89 из десятичной системы счисления в двоичную; b) число 1101101.01 из двоичной системы счисления в десятичную; c) число 332 из восьмеричной системы счисления в шестнадцатеричную

Практическая работа 1.2. Кодирование данных

Цель работы – научиться:

- записывать прямой, обратный и дополнительный коды в однобайтовом представлении для целых чисел со знаком;
- вычислять объем памяти, требуемый для хранения графического изображения.

Порядок выполнения работы

1. Изучить теоретический материал из раздела 1.3 данного пособия.
2. Выполнить задания 1, 2, 3.
3. Оформить отчет о работе.

Требования к содержанию и оформлению отчета о работе

☑ Отчет выполняется письменно в тетради, он должен содержать тему работы, условие и ход решения для каждого задания.

Задание 1

Для заданного отрицательного целого десятичного числа записать дополнительный код числа в 1-байтовом представ-

лении. Выбрать число в соответствии с номером варианта из табл. 1.8.

Таблица 1.8

Варианты для задания 1

Номер варианта	Число	Номер варианта	Число	Номер варианта	Число
1	-48	6	-57	11	-46
2	-54	7	-72	12	-58
3	-62	8	-65	13	-75
4	-39	9	-55	14	-39
5	-43	10	-39	15	-51

Задание 2

Задан дополнительный код числа в 1-байтовом представлении. Найти значение числа в десятичной системе счисления. Выбрать запись кода числа в соответствии с номером варианта из табл. 1.9.

Таблица 1.9

Варианты для задания 2

Номер варианта	Дополнительный код числа	Номер варианта	Дополнительный код числа	Номер варианта	Дополнительный код числа
1	1,0011001	6	1,1001001	11	1,1001011
2	1,0111011	7	1,0111000	12	1,1010110
3	1,1111001	8	1,1011000	13	1,1010101
4	1,0111011	9	1,0011111	14	1,0001110
5	1,0011000	10	1,1111101	15	1,1011110

Задание 3

Определить объем видеопамяти в байтах, требуемый для хранения изображения на экране монитора, который может отображать **N** точек по горизонтали и **M** точек по вертикали при заданном цветовом режиме. Выбрать значения **N**, **M**, а также цветовой режим в соответствии с номером варианта из табл. 1.10.

Таблица 1.10

Варианты для задания 3

Номер варианта	N	M	Цветовой режим
1	1600	900	Черно-белое изображение
2	1280	768	256 оттенков серого цвета
3	1152	864	High Color
4	1024	768	True Color
5	800	600	Индексное кодирование
6	1600	900	256 оттенков серого цвета
7	1280	768	High Color
8	1152	864	True Color
9	1024	768	Индексное кодирование
10	800	600	Черно-белое изображение
11	1600	900	High Color
12	1280	768	True Color
13	1152	864	Индексное кодирование
14	1024	768	Черно-белое изображение
15	800	600	256 оттенков серого цвета

Практическая работа 1.3. Построение таблиц истинности и логических схем

Цель работы – научиться:

- записывать таблицы истинности для заданных логических выражений;
- по заданному логическому выражению строить соответствующую логическую схему.

Порядок выполнения работы

1. Изучить теоретический материал из раздела 1.4 данного пособия.
2. Выполнить задания 1, 2.
3. Оформить отчет о работе.

Требования к содержанию и оформлению отчета о работе

☑ Отчет выполняется письменно в тетради, он должен содержать тему работы, условие и ход решения для каждого задания.

Задание 1

Составить таблицу истинности для заданного логического выражения. Вид выражения выбрать из табл. 1.11 в соответствии с номером варианта.

Таблица 1.11

Варианты для задания 1

Вариант 1	Вариант 6	Вариант 11
$(x+\bar{y}) \mid (x \sim yz)$	$((x \vee \bar{y})z) \to ((x \sim z)+y)$	$((x \sim z)+y)\cdot(x \mid yz)$
Вариант 2	Вариант 7	Вариант 12
$\bar{x} \to (z \sim (y+x\bar{z}))$	$(x \vee \bar{y})z \to ((x \downarrow y) \mid z)$	$\bar{x} \to (\bar{z} \sim (y+xz))$
Вариант 3	Вариант 8	Вариант 13
$(xz \to y) \mid (xy+xz)$	$(x+(yz)) \mid (xy)$	$(x \sim (\bar{y}+z)) \mid (xy)$
Вариант 4	Вариант 9	Вариант 14
$(\bar{x} \vee y) \downarrow \bar{z} \to ((x+y) \mid z)$	$(x+y\bar{z}) \to (z \sim (y \downarrow (x \vee \bar{z})))$	$(x\bar{y}) \mid (x \sim yz)$
Вариант 5	Вариант 10	Вариант 15
$(x \sim \bar{y}) \mid (x \downarrow (\bar{y}z+\bar{x}))$	$((x \vee \bar{y})z) \to (x \sim y)$	$((x \downarrow \bar{y})z) \to ((x \sim z)+y)$

Пример выполнения задания

Составим таблицу истинности для выражения

(x ~ z) | ((x · y) ~ (y · z)).

Заданное выражение содержит три логические переменные. Столбцы 1, 2, 3 таблицы истинности (табл. 1.12) содержат возможные комбинации значений переменных **x**, **y**, **z**. Таблица истинности будет содержать 8 строк по количеству возможных различных комбинаций значений переменных.

Таблица 1.12

Таблица истинности с промежуточными результатами вычислений

1	2	3	4	5	6	7	8
x	**y**	**z**	**x ~ z**	**x · y**	**y · z**	**(x · y) ~ (y · z)**	**(x ~ z) | ((x · y) ~ (y· z)**
0	**0**	**0**	1	0	0	1	**0**
0	**0**	**1**	0	0	0	1	**1**
0	**1**	**0**	1	0	0	1	**0**
0	**1**	**1**	0	0	1	0	**1**
1	**0**	**0**	0	0	0	1	**1**
1	**0**	**1**	1	0	0	1	**0**
1	**1**	**0**	0	1	0	0	**1**
1	**1**	**1**	1	1	1	1	**0**

Столбцы 4, 5, 6, 7 содержат результаты промежуточных вычислений, в столбце 8 находятся значения логического выражения для каждой комбинации значений переменных.

Задание 2

Построить логическую схему для заданной логической функции ***F***. Вид выражения выбрать из табл. 1.13 в соответствии с номером варианта.

Таблица 1.13

Варианты для задания 2

Вариант 1	Вариант 6	Вариант 11
$F = \overline{\overline{A} + B} + A \cdot \overline{B}$	$F = A \vee (\overline{B} + C) + \overline{\overline{A}B}$	$F = C + A\overline{B} + (\overline{A\overline{B} + A})$
Вариант 2	Вариант 7	Вариант 12
$F = (\overline{A} \vee C)\overline{A} + (B \vee \overline{C})$	$F = (A \vee B)(\overline{A} + B)$	$F = (B \vee \overline{C}) + A \vee \overline{\overline{A}B}$
Вариант 3	Вариант 8	Вариант 13
$F = \overline{A\overline{B}} + C + \overline{A}$	$F = A\overline{B} \vee \overline{A}C + \overline{C}$	$F = \overline{A \vee \overline{B}} + C\overline{A}$
Вариант 4	Вариант 9	Вариант 14
$F = \overline{A \vee \overline{B}} + A \cdot B$	$F = (\overline{AB} + \overline{C}) \vee B\overline{C}$	$F = A + B(\overline{A} \vee \overline{C})$
Вариант 5	Вариант 10	Вариант 15
$F = C\overline{A} \vee (A + \overline{B})$	$F = \overline{\overline{A}\,\overline{C}} + \overline{C \vee B}$	$F = A(\overline{B} + C) + \overline{A} \vee B$

Пример выполнения задания

Допустим, задана логическая функция: $F = \bar{B}A + B\bar{A} + C\bar{B}$. Построение схемы удобнее выполнять, начиная с ее выхода.

Первый этап. Функция ***F*** рассматривается как результат операции «ИЛИ» (логическое сложение), примененной к функциям $\bar{B}A$, $B\bar{A}$ и $C\bar{B}$. Используем логический элемент «ИЛИ» и получаем первый фрагмент схемы (рис. 1.8,*а*).

Второй этап. К входам элемента «ИЛИ» подключаются логические элементы «И», входными переменными которых являются ***A, B, C*** и их инверсии (рис. 1.8,*б*).

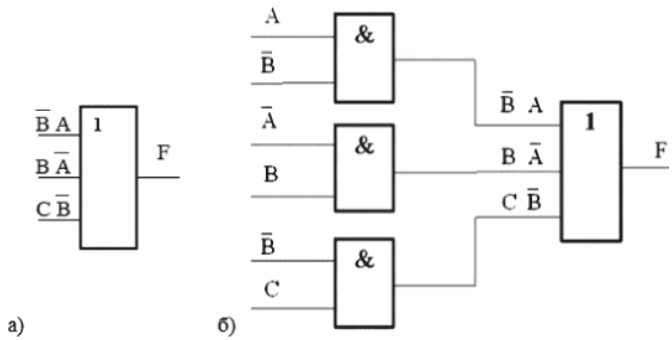


Рис. 1.8. Первый *(а)* и второй *(б)* этапы построения схемы

Третий этап. Для получения инверсий для ***A*** и ***B*** на соответствующих входах ставим инверторы и соединяем одноименные входные переменные в одну линию (рис. 1.9). Построение схемы закончено.

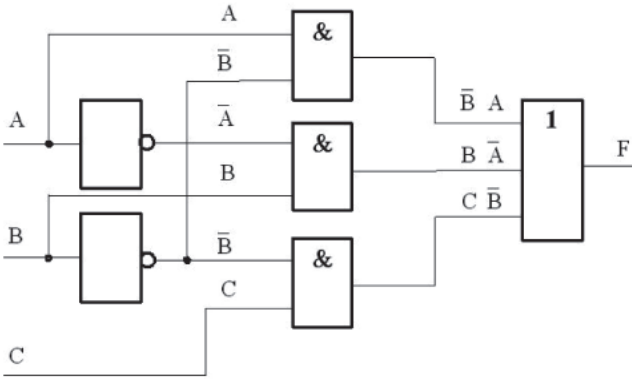


Рис. 1.9. Результат построения схемы

1.6. Тестовые задания

1. Зафиксированные сигналы или наблюдения – это...

a) информация;
b) данные;
c) базы данных;
d) знания.

2. Сведения об объектах или явлениях, уменьшающие степень неопределенности знаний об этих объектах или явлениях, – это...

a) знания;
b) данные;
c) информация;
d) базы данных.

3. Сигнал называют аналоговым, если...

a) он может принимать конечное или бесконечное число конкретных, отделенных друг от друга по времени, значений;
b) он непрерывно изменяется во времени;
c) он несет текстовую информацию;
d) он несет звуковую информацию.

4. Преобразование непрерывных изображений и звука в набор дискретных значений называют...

a) дискретизацией (оцифровкой);
b) кодированием;
c) декодированием;
d) информатизацией.

5. Информацию, не зависящую от личного мнения или суждения, называют...

a) достоверной;
b) актуальной;
c) объективной;
d) полной.

6. Двоичное число 10001_2 соответствует десятичному числу...

a) 11_{10};
b) 17_{10};
c) 256_{10};
d) 1001_{10}.

7. Максимальным из чисел 132_4, 132_5, 132_6, 132_8 является число...

a) 132_4;
b) 132_5;
c) 132_8;
d) 132_6.

8. Расположите числа, заданные в различных системах счисления $X = 46_7$, $Y = 65_{10}$, $Z = 34_6$, в порядке возрастания:

a) Z, X, Y;
b) Y, X, Z;
c) Z, Y, X;
d) X, Z, Y.

9. Пять килобайт равны...

a) 5000 байтам;
b) 5000 битам;
c) 500 байтам;
d) 5120 байтам.

10. Рассматриваются системы счисления с основаниями 10, 8, 16. Запись числа 81...

a) отсутствует в десятичной системе счисления;
b) существует во всех перечисленных системах счисления;
c) отсутствует в восьмеричной системе счисления;
d) отсутствует в шестнадцатеричной системе счисления.

11. Количество различных символов, закодированных байтами в сообщении 100010011000000010001001, равно...

a) 2;
b) 3;
c) 6;
d) 12.

12. Для кодирования цвета точки в черно-белом изображении с 256 оттенками серого цвета достаточно... двоичных разрядов:

a) 16;
b) 8;
c) 256;
d) 24.

13. Для кодирования цвета точки в цветном изображении (система RGB допускает 16,5 миллиона цветов) достаточно... двоичных разрядов:

a) 16;
b) 8;
c) 256;
d) 24.

14. Система кодирования текстовых символов UNICODE использует для кодирования одного текстового символа... двоичных разрядов:

a) 16;
b) 8;
c) 24;
d) 256.

15. Понятие «тезаурус», используемое в семантическом подходе к измерению информации, означает...

a) множество зафиксированных сигналов;
b) объем информации, содержащейся в сообщении;
c) совокупность сведений, которыми располагает пользователь или система;
d) количество принятых системой сообщений.

16. Мера «полезность информации» используется при подходе к измерению информации...

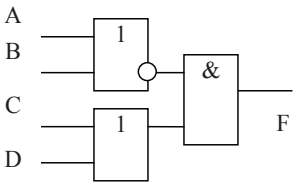
a) прагматическом;
b) синтаксическом;
c) семантическом;
d) экономическом.

17. Для заданной таблицы истинности определите результат логической функции $F = A \wedge B$:

A	0	0	1	1
B	0	1	0	1
F	?	?	?	?

a) 0 1 1 1;
b) 0 0 0 1;
c) 0 1 1 0;
d) 1 0 0 1.

18. Показанная ниже схема соответствует функции...



a) $F = A \vee B \wedge C \vee D$;
b) $F = \overline{A \wedge B} \vee C \wedge D$;
c) $F = (\overline{A \vee B}) \wedge (C \vee D)$;
d) $F = A \wedge B \vee C \wedge D$.

2. ОСНОВЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ

Изучив материалы этого раздела, вы **узнаете**:

- последовательность и содержание этапов решения задачи на компьютере;
- средства представления алгоритмов;
- назначение и свойства базовых алгоритмических структур;
- различия между языками программирования разных типов;
- этапы работы с программой на языке программирования высокого уровня;
- особенности существующих методологий программирования;
- состав и назначение основных структур языка программирования Паскаль.

Выполнив практические работы, вы **научитесь**:

- применять систему программирования Turbo Pascal для работы с программой;
- решать задачи, сформулированные в терминах некоторой предметной области (экономика, управление и пр.), с использованием программ на языке Паскаль, реализуя полный цикл от постановки задачи до анализа результатов.

Методические рекомендации

Схема работы с учебными материалами данного раздела представлена на рис. 2.1. Теоретическая часть предполагает изучение четырех тем. Каждая тема заканчивается перечнем вопросов и упражнений для самоконтроля. Ответив на вопросы, вы сможете проверить, насколько хорошо освоили учебный материал. Практическая часть предполагает выполнение четырех практических работ по теме «Основы языка программирования Паскаль». Изучение раздела заканчивается выполнением тестовых заданий, ключ к которым можно найти в прил. 2. Дополнительные учебные материалы по данному разделу можно найти

в рекомендованных литературных источниках [3], [5], [6], [7] и интернет-ресурсах [14], [17], список которых представлен в конце пособия.

Рис. 2.1. Схема работы с учебными материалами раздела 2

2.1. Технология решения задач с использованием компьютера

Человек использует компьютер для решения самых разных информационных задач. Если в составе доступного программного обеспечения имеется программа, подходящая для решения задачи, то пользователь выбирает ее в качестве инструмента (текстовый или табличный процессор, специализированный пакет программ и др.). В том случае, когда готовой программой воспользоваться нельзя, приходится использовать языки программирования.

В решении прикладной задачи на компьютере с использованием программирования можно выделить следующие этапы:

- постановка задачи;
- формализация (математическое моделирование);
- построение алгоритма;
- составление программы на языке программирования;
- отладка и тестирование программы;
- проведение расчетов;
- анализ полученных результатов.

Рассмотрим каждый из перечисленных этапов на примере решения задачи расчета заработной платы.

Постановка задачи. На этапе постановки задачи должно быть четко определено, какие исходные данные известны и что требуется определить. Необходимо также определить области допустимых значений для исходных данных. Важным моментом постановки задачи является определение формы выдачи результатов решения задачи. Постановка задачи включает следующие необходимые моменты: сбор информации о задаче, формулировка условия задачи, определение конечных целей решения задачи, определение формы выдачи результатов, описание данных.

Обратившись к задаче расчета заработной платы, на этапе постановки задачи следует прежде всего выяснить систему оплаты труда для разных категорий работающих: фиксированный оклад или почасовая система, наличие или отсутствие премиальных выплат, полагающиеся льготы и пр. Проведенный анализ позволит определить набор исходных данных, а также набор результатов решения задачи. Например, можно определить в качестве исходных данных следующие: количество часов, отработанных каждым работником за рассматриваемый период, размер оплаты за один час, процент премии. В качестве результата можно принять начисленную заработную плату за период для каждого работающего рассматриваемого подразделения (фирмы, отдела и т. д.).

Определив в качестве цели решения задачи получение определенных бухгалтерских документов, формируемых

при расчете и выплате заработной платы, мы тем самым определим форму выдачи результатов. При описании данных важно учесть, что обрабатываются числовые значения, представляющие собой денежные суммы. Это обязывает использовать в расчетах данные вещественного типа (с дробной частью) с обеспечиваемой точностью не менее двух знаков после запятой.

Формализация. На этом этапе строится математическая модель, т. е. система математических соотношений (формул, уравнений, неравенств и т. д.). Создавая математическую модель, нужно выделить существенные стороны рассматриваемого процесса или явления, определить математические соотношения, связывающие результаты решения задачи с исходными данными.

Возвращаясь к примеру решения задачи расчета заработной платы, мы могли бы записать формулы для расчета начисленной заработной платы. Например, начисленная заработная плата может быть рассчитана по формуле $S = n \cdot T \cdot (1+P)$, где S – размер начисленной заработной платы, n – количество отработанных работником часов за расчетный период, T – тариф (размер платы за один час), P – процент премии.

Построение алгоритма. Последовательность шагов решения задачи от ввода исходных данных до вывода результатов записывается в виде алгоритма. Разработка алгоритма включает: выбор метода проектирования и формы записи алгоритма; выбор метода тестирования; формирование тестов; проектирование алгоритма.

В случае решения задачи расчета заработной платы алгоритм должен представлять собой описание последовательности шагов (ввод исходных данных, порядок расчета по формулам, проверка тех или иных условий, вывод результатов и т. д.), позволяющих на основе заданных значений исходных данных получить требуемые результаты, описанные при постановке задачи.

Программирование. Этап программирования предполагает выполнение следующих видов работ: выбор языка программирования, уточнение способов представления данных, запись алгоритма на выбранном языке программирования.

В задаче расчета заработной платы для реализации расчетов можно было бы использовать, например, любой процедурный язык программирования (Pascal, C, Basic). Следует заметить, что в рассматриваемой задаче можно было бы обойтись без этапа программирования, заменив его использованием современных программных приложений общего назначения (например, электронными таблицами) или специализированных бухгалтерских систем (например, Бухгалтерия 1С).

Отладка и тестирование программы. Под отладкой программы понимается процесс испытания работы программы и исправления обнаруженных при этом ошибок. Ошибки, вызванные нарушением правил записи языка программирования (синтаксические ошибки), обнаруживаются средствами системы программирования. Если ошибки носят логический характер (например, последовательность действий в программе не соответствует алгоритму), то для их обнаружения потребуется выполнить тестовые расчеты по программе.

Тест – это конкретный набор значений исходных данных, для которого известен результат решения задачи, используемый для проверки правильности работы программы. Виды тестирования:

- ***альфа-тестирование*** – тестирование готового программного продукта разработчиками этого продукта на специально созданных задачах;
- ***бета-тестирование*** – использование программы потенциальными пользователями (о найденных ошибках и замечаниях пользователи сообщают разработчику);
- ***тестирование программы «как белого ящика»*** – тестирование, подразумевающее знание исходного кода программы и полный доступ к нему;
- ***тестирование программы «как черного ящика»*** – тестирование, при котором программа рассматривается как объект, внутренняя структура которого неизвестна.

Для задачи расчета заработной платы также необходимо было бы подготовить несколько наборов исходных данных, отражающих различные возможные ситуации, для которых расчеты следовало провести, например, с помощью

калькулятора. Эти наборы можно было бы использовать при тестировании правильности работы программы.

Проведение расчетов. Анализ результатов. Последний этап – это использование разработанной программы для получения результатов (эксплуатация программы). Анализ получаемых результатов иногда может потребовать вернуться к одному из предшествующих этапов, чтобы внести изменения в математическую модель, алгоритм или программу.

? Вопросы и упражнения для самоконтроля

1. Перечислите основные этапы решения прикладной задачи пользователя на компьютере с использованием программирования.
2. На каком этапе технологической цепочки решения задачи на компьютере выделяются исходные данные и результаты решения задачи, области определения исходных данных и форма выдачи результатов?
3. Каково назначение этапа формализации задачи?
4. В чем состоит процесс разработки алгоритма решения задачи?
5. Какие виды работ выполняются на этапе программирования при решении задачи на компьютере?
6. Каково назначение этапа отладки и тестирования программы? Опишите разные виды тестирования программных продуктов.
7. Как называется этап, на котором выполняется использование разработанной программы для решения задачи?

2.2. Алгоритмизация вычислений

2.2.1. Основные понятия и определения

Алгоритм – заранее заданное понятное и точное предписание возможному исполнителю совершить определенную последовательность действий для получения решения задачи за конечное число шагов. Под ***исполнителем алгоритма*** понимается некоторая абстрактная или реальная (техническая,

биологическая или биотехническая) система, способная выполнить действия, предписываемые алгоритмом. В информатике универсальным исполнителем алгоритмов является компьютер.

Алгоритм должен обладать рядом необходимых свойств:

- ***понятность*** – алгоритм должен включать только такие инструкции (команды), которые понятны исполнителю;
- ***дискретность*** (прерывность, раздельность) – алгоритм должен представлять процесс решения задачи как последовательное выполнение некоторых простых шагов (этапов);
- ***результативность*** – за конечное число шагов алгоритм должен приводить либо к решению поставленной задачи, либо к завершению выполнения с выдачей сообщения о невозможности получить решение;
- ***массовость*** – алгоритм решения задачи должен быть применим для некоторого класса задач, различающихся лишь исходными данными, которые могут выбираться из некоторой области, называемой областью применимости алгоритма;
- ***определенность (детерминированность)*** – при выполнении алгоритма с одним и тем же набором исходных данных всякий раз должен получаться один и тот же результат.

Существуют различные ***формы представления алгоритмов***:

- *словесная* – запись алгоритма на естественном разговорном языке;
- *графическая* – представление алгоритма с помощью графических символов;
- *псевдокод* – полуформализованное описание алгоритма на условном алгоритмическом языке, включающее элементы языка программирования и элементы разговорного языка;
- *программная* – запись алгоритма на языке программирования.

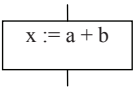
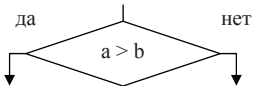
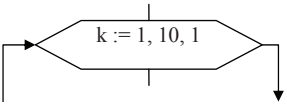
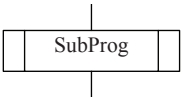
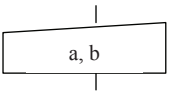
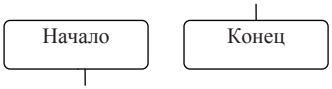
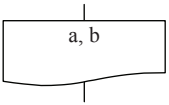
Словесный способ широкого распространения не получил, так как имеет ряд существенных недостатков: отсутствие строгости, неоднозначность толкования отдельных предписаний. При графическом представлении алгоритм изображается в виде последовательности связанных между собой функциональных

блоков, каждый из которых соответствует выполнению одного или нескольких действий. Такое графическое представление называется ***блок-схемой***.

☑ В блок-схеме каждому типу действий (вводу исходных данных, вычислению значений выражений, проверке условий и т. п.) соответствует геометрическая фигура строго определенного вида, представленная в виде блочного символа. Блочные символы соединяются линиями переходов, определяющими очередность выполнения действий. В табл. 2.1 приведены наиболее часто употребляемые блочные символы.

Таблица 2.1

Основные блочные символы для изображения блок-схем

Название символа	Обозначение и пример заполнения	Пояснение
Процесс		Вычислительное действие, результат присваивается переменной
Решение		Проверка условий и выбор одного из двух альтернативных решений
Модификация		Начало цикла с параметром (в данном примере ***k*** – параметр цикла, изменяющийся от 1 до 10 с шагом 1)
Предопределенный процесс		Вызов подпрограммы (в данном примере SubProg – имя подпрограммы)
Ввод данных		Ввод значений переменных с клавиатуры
Пуск и останов		Начало и конец алгоритма
Документ		Вывод результатов на печать

☑ Логическая структура любого алгоритма может быть представлена комбинацией трех базовых алгоритмических структур: ***следование, ветвление, цикл***. Характерной особенностью базовых структур является наличие в них одного входа и одного выхода.

Базовая алгоритмическая структура «следование» (рис. 2.2) образуется последовательностью действий, следующих одно за другим.

Рис. 2.2. Базовая алгоритмическая структура «следование»

Базовая алгоритмическая структура «ветвление» обеспечивает в зависимости от результата проверки некоторого условия выбор одного из альтернативных путей работы алгоритма. Каждый из путей ведет к общему выходу. Структура «ветвление» существует в двух основных вариантах: «неполная развилка», «полная развилка» (рис. 2.3).

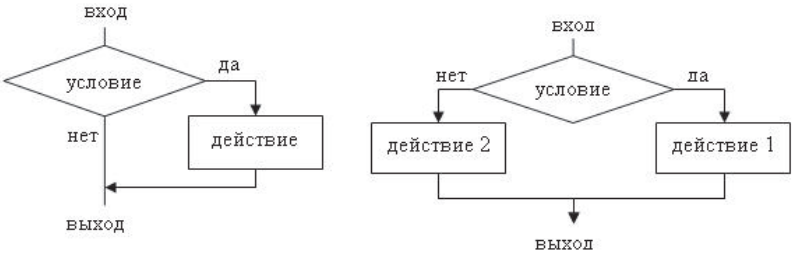


Рис. 2.3. Базовые алгоритмические структуры «неполная развилка» (слева) и «полная развилка» (справа)

Базовая алгоритмическая структура «цикл» обеспечивает многократное выполнение некоторой совокупности

действий, которая называется ***телом цикла***. Основные разновидности циклов: «цикл с параметром» (рис. 2.4), «цикл с предусловием», «цикл с постусловием» (рис. 2.5).

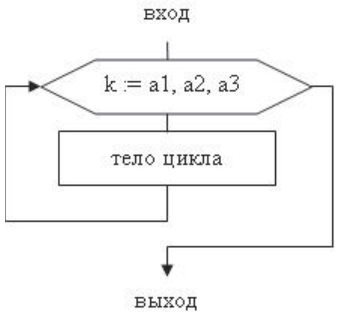


Рис. 2.4. Базовая алгоритмическая структура «цикл с параметром»

Цикл с параметром предписывает выполнять тело цикла для всех значений некоторой переменной (параметра цикла), изменяющейся в заданном диапазоне. На рис. 2.4 параметром цикла является переменная ***k***, значение которой изменяется от ***a1*** до ***a2*** с шагом ***a3***. При каждом значении параметра цикла выполняются действия, входящие в тело цикла. Для применения цикла с параметром необходимо, чтобы количество повторений цикла было известно заранее (в частности, это количество всегда можно вычислить при заданных значениях ***a1***, ***a2*** и ***a3*** по формуле $n = \left[\frac{a2 - a1}{a3}\right] + 1$, где квадратные скобки обозначают операцию вычисления целой части).

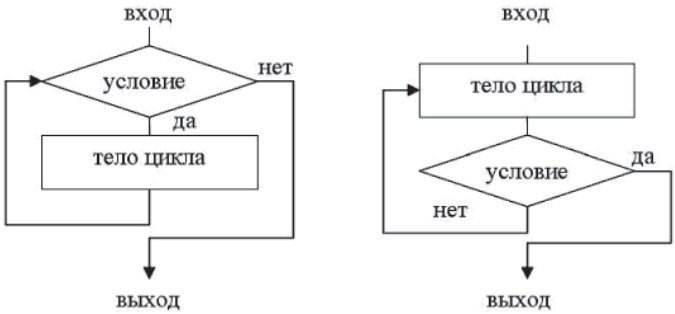


Рис. 2.5. Базовые алгоритмические структуры «цикл с предусловием» (слева) и «цикл с постусловием» (справа)

Алгоритмические структуры «цикл с предусловием» и «цикл с постусловием» можно использовать как в случае известного числа повторений цикла, так и для организации ***итерационного цикла***, для которого число повторений тела цикла заранее определить нельзя. Выход из итерационного цикла осуществляется при выполнении некоторого условия. ***Цикл с предусловием*** начинается с проверки условия выхода из цикла. Если оно истинно, то выполняется тело цикла, в противном случае происходит выход из цикла. В ***цикле с постусловием*** сначала выполняется один раз тело цикла, затем проверяется логическое выражение, определяющее условие выхода из цикла. Если условие истинно, то цикл прекращает свою работу, в противном случае происходит повторение действий, указанных в цикле.

В зависимости от того, какие базовые алгоритмические структуры используются в алгоритме, различают три основных типа алгоритмов: линейные, разветвленные, циклические. Если алгоритм не содержит базовых структур ветвления и цикла, он считается ***линейным***. Если в алгоритме присутствует хотя бы одна структура ветвления, но нет ни одного цикла, алгоритм называется ***разветвленным***. Если алгоритм содержит хотя бы одну циклическую структуру, он называется ***циклическим***. При этом алгоритмические структуры могут встраиваться друг в друга. Например, внутри цикла может присутствовать структура ветвления или другая циклическая структура.

2.2.2. Примеры типовых алгоритмов

Линейный алгоритм

Задача 2.1. При изготовлении некоторых изделий используется сталь определенной марки. **Норма расхода материала (NR)** при производстве одного изделия определяется как **чистый вес изделия (V)** плюс **вес отходов от изготовления изделия (VO)**. **Коэффициент использования материала (K)** рассчитывается как отношение чистого веса изделия к норме расхода. Определить, сколько потребуется закупить стали для производства заданного количества изделий, если известен

чистый вес изделия и коэффициент использования стали. Определить суммарный вес отходов от производства.

Исходные данные: V – чистый вес одного изделия; **K** – коэффициент использования стали в процентах; **N** – число производимых изделий.

Требуется определить: S – общий вес стали для производства **N** изделий; **VO** – общий вес отходов от производства **N** изделий.

Расчетные формулы

Запишем соотношения, следующие из определения нормы расхода материала (**NR**) и коэффициента использования материала (**K**) для производства одного изделия:

$$NR = V + VO; \quad (2.1)$$

$$K = V / NR \cdot 100 \text{ (в процентах)}. \quad (2.2)$$

Если заданы **K** и **V**, из (2.2) получаем формулу для расчета нормы расхода стали для производства одного изделия:

$$NR = V \cdot 100 / K. \quad (2.3)$$

Зная **NR**, из (2.1) определим вес отходов при производстве одного изделия: **VO = NR – V**. Определим расход стали (**S**) и вес отходов при производстве **N** изделий (**SO**):

$$S = N \cdot NR; \; SO = N \cdot VO.$$

Алгоритм решения задачи

Представим словесный алгоритм решения задачи.

1. Начать вычисления.
2. Ввести исходные данные: **V** – чистый вес одного изделия; **K** – коэффициент использования материала в процентах; **N** – число производимых изделий.
3. Вычислить **NR** (норма расхода стали для одного изделия) по формуле **NR = V · 100 / K**.
4. Вычислить **VO** (вес отходов при производстве одного изделия) по формуле **VO = NR – V**.
5. Вычислить общий расход стали по формуле **S = N · NR**.
6. Вычислить общий вес отходов по формуле **SO = N · VO**.
7. Вывести на экран вычисленные значения **S**, **SO**.
8. Закончить вычисления.

☑ Анализ алгоритма показывает, что он представляет собой последовательность однозначно определенных действий, выполняемых однократно. Это означает, что алгоритм решения задачи является линейным.

Разветвленный алгоритм

Задача 2.2. Поставщики предлагают сталь для производства некоторых изделий по разным ценам: поставщик **A** – 20000 руб. за тонну, поставщик **B** – 23000 руб., поставщик **C** – 22000 руб. При использовании стали поставщика **A** чистый вес изделия составляет 20 кг, поставщика **B** – 18 кг, поставщика **C** – 19 кг. Во всех трех случаях коэффициент использования материала равен 98%. Определить, какому из поставщиков следует отдать предпочтение при закупке стали, если критерием выбора является экономия средств при производстве изделий.

Исходные данные:

ZA – цена закупки 1 кг стали поставщика **A**; **ZB –** цена закупки 1 кг стали поставщика **B**; **ZC –** цена закупки 1 кг стали поставщика **C**;

VA – чистый вес одного изделия из стали поставщика **A**; **VB** – из стали поставщика **B**; **VC** – из стали поставщика **C**;

K – коэффициент использования материала в процентах.

Для простоты значения **VA, VB, VC** можно считать неизменными (константами), предполагая, что физические характеристики стали не изменяются в течение длительного периода времени. Значение **K** также будем считать неизменным. Значения **ZA, ZB, ZC** могут изменяться в зависимости от конъюнктуры рынка, поэтому будем считать их переменными.

Требуется определить:

RA – денежные расходы на закупку материалов для производства одного изделия из стали поставщика **A**, **RB** – из стали поставщика **B, RC** – из стали поставщика **C**.

Расчетные формулы

Если известны коэффициент использования материала (**K**) и чистый вес изделия из стали поставщика **A** (**VA**), можно определить из соотношения (2.3) норму расхода стали поставщика **A** на производство одного изделия: **NRA = VA · 100 / K**.

Аналогично рассчитываются нормы расхода материала для поставщиков **B** и **C**: **NRB = VB · 100 / K**; **NRC = VC · 100 / K**.

Зная нормы расхода материалов, можно определить денежные расходы на закупку стали для производства одного изделия по каждому из поставщиков:

RA = NRA · ZA; **RB = NRB · ZB**; **RC = NRC · ZC**.

Таким образом, получаем расчетные формулы:

RA = VA · 100 / K · ZA;
RB = VB · 100 / K · ZB;
RC = VC · 100 / K · ZC.

Алгоритм решения задачи

Алгоритм решения задачи представлен на рис. 2.6. Для определения наилучшего поставщика выполняется выбор наименьшего из трех значений (**RA, RB, RC**). Значения попарно сравниваются друг с другом. Всякий раз из двух значений выбирается то, которое меньше либо равно сравниваемому с ним.

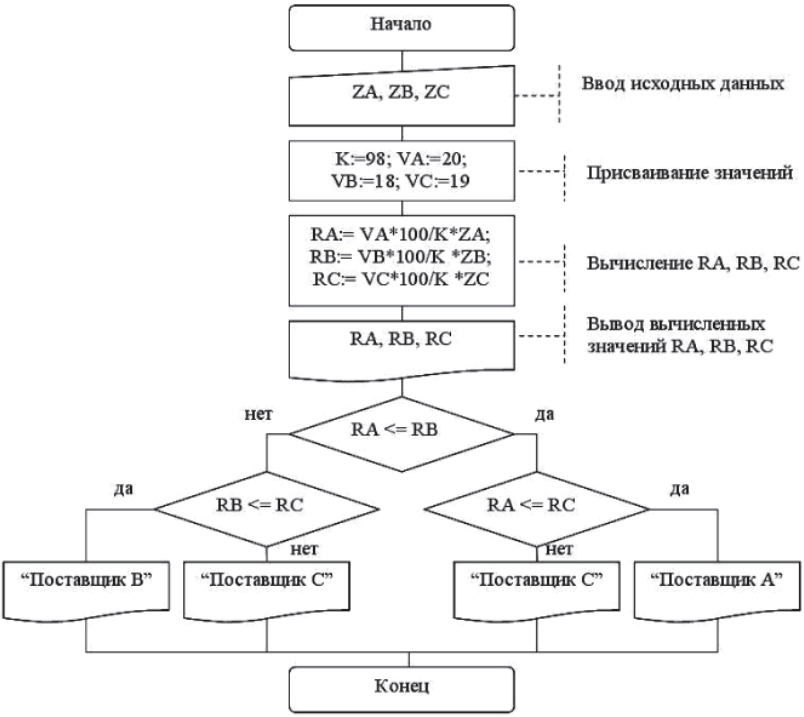


Рис. 2.6. Блок-схема алгоритма решения задачи 2.2

☑ Анализ алгоритма показывает, что он содержит структуры «ветвление», но не содержит циклических структур. Этот факт позволяет отнести алгоритм решения задачи к классу разветвленных алгоритмов.

Циклический алгоритм

Задача 2.3. Предприятие рассматривает предложения **N** поставщиков стали для производства некоторого изделия. Каждый поставщик определил цену за 1 тонну стали: **Z [i], i = 1, 2, ..., N**. Технологи предприятия, исходя из характеристик стали, просчитали вес изделия для каждой марки стали: **V [i], i = 1, 2, ..., N**. Во всех случаях коэффициент использования материала равен 98%. Определить, кому из поставщиков следует отдать предпочтение при закупке стали, если критерием выбора является экономия средств при производстве изделий.

Исходные данные

N – количество поставщиков;

Набор значений (массив) Z [i], i = 1, 2, ..., N – цена закупки 1 кг стали поставщика с номером **i**;

Набор значений (массив) V [i], i = 1, 2, ..., N – чистый вес одного изделия из стали поставщика с номером **i**;

K – коэффициент использования материала в процентах.

Требуется определить:

Набор значений (массив) R [i], i = 1, 2, ..., N – денежные расходы на закупку материалов для производства одного изделия из стали поставщика с номером **i**;

MINR – наименьшее из значений в массиве **R**.

Расчетные формулы

Если известны коэффициент использования материала (**K**) и чистый вес изделия из стали поставщика с номером **i**, можно определить денежные расходы на закупку стали для производства одного изделия:

$$R[i] = V[i] \cdot 100 / K \cdot Z[i].$$

Затем в полученном массиве нужно определить наименьшее значение **MINR**, используя попарное сравнение значений между собой. Результатом решения задачи будут номера поставщиков, для которых значение **R [i]** равно **MINR.**

Алгоритм решения задачи

Блок-схема алгоритма представлена на рис. 2.7.

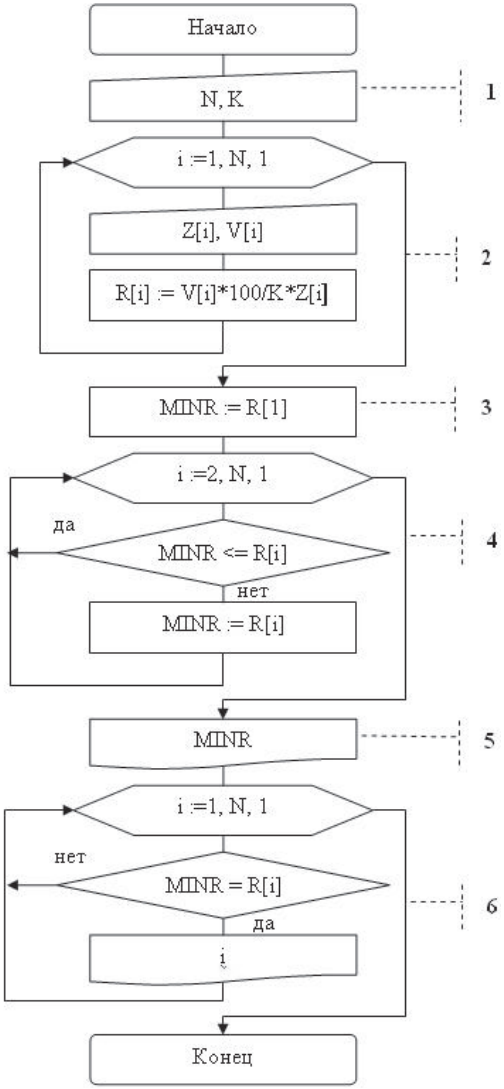


Рис. 2.7. Блок-схема алгоритма решения задачи 2.3

Блок 1 – ввод исходных данных: **N** – количество поставщиков; **K** – коэффициент использования материала.

Блок 2 – ввод исходных данных и вычисление значений элементов массива **R [i]** (денежные расходы на закупку материалов для производства одного изделия из стали поставщика с номером **i**).

Блок 3 – за начальное значение минимума среди значений элементов массива **R** принимается элемент с номером 1.

Блок 4 – поиск минимального значения элементов в массиве **R**. Начиная с элемента с номером 2, значение каждого элемента сравнивается со значением переменной **MINR**. Если сравниваемый элемент меньше, чем значение **MINR**, то переменной **MINR** присваивается значение этого элемента. Если **MINR** меньше либо равно значению элемента, то **MINR** не изменяется.

Блок 5 – вывод на экран найденного значения **MINR**.

Блок 6 – определение и вывод номеров поставщиков, для которых денежные расходы на закупку материалов для производства одного изделия являются минимальными.

☑ Анализ алгоритма решения задачи показывает, что наряду со структурами следования и ветвления он содержит алгоритмические структуры «цикл с параметром». Это позволяет классифицировать рассматриваемый алгоритм как циклический.

? Вопросы и упражнения для самоконтроля

1. Что такое алгоритм? Назовите способы записи алгоритмов.
2. Какие средства используются при описании алгоритмов в виде блок-схем?
3. Объясните смысл понятий «результативность алгоритма», «определенность алгоритма» и «массовость алгоритма».
4. Перечислите основные алгоритмические структуры.
5. Назовите варианты представления базовой алгоритмической структуры «ветвление».
6. Опишите виды алгоритмических структур организации цикла. Что такое тело цикла?

7. Какая базовая алгоритмическая структура реализует цикл с известным числом повторений?

8. Какие алгоритмические структуры используются для представления итерационных циклов?

2.3. Языки программирования

2.3.1. Типы и разновидности языков программирования

Язык программирования – это формальная знаковая система, используемая для связи человека с компьютером, предназначенная для описания данных и алгоритмов их обработки на компьютере.

Существуют такие языки программирования, которые отражают структуру данного класса компьютеров, и поэтому их называют ***машинно-ориентированными языками*** или ***языками низкого уровня***. Особенностью этих языков является жесткая ориентация на определенную архитектуру микропроцессора. В стремлении приспособить эти языки к человеку был разработан язык символического кодирования – язык Ассемблер, который также относят к языкам низкого уровня.

Для облегчения труда программистов были созданы ***языки программирования высокого уровня***. Отличительной особенностью этих языков является их ориентация не на систему команд микропроцессора, а на систему операторов, характерных для записи определенного класса алгоритмов. К ним относятся, например, Basic (Бейсик), Pascal (Паскаль), C (Си).

Так как текст программы, записанной на языке высокого уровня, не понятен микропроцессору, необходим перевод этого текста на внутренний язык компьютера (язык машинных кодов). Процедура перевода текста программы с языка программирования высокого уровня на язык машинных кодов называется ***трансляцией***. Она выполняется специальными программами, которые называются ***трансляторами***. Существует два основных вида трансляторов**:** интерпретаторы и компиляторы.

Интерпретаторы анализируют и выполняют (интерпретируют) программу последовательно по одной инструкции

(оператору). Синтаксическая ошибка обнаруживается интерпретатором только тогда, когда он приступает к выполнению инструкции, содержащей эту ошибку. ***Компиляторы*** преобразуют всю программу в модуль на машинном языке после выполнения процедуры проверки текста программы на наличие синтаксических ошибок и их исправления программистом. Только потом программа может быть выполнена.

Работа с программой, написанной на языке программирования высокого уровня, выполняется с помощью набора специальных инструментальных программных средств, образующих ***систему программирования***. Обычно система программирования включает в себя следующие основные компоненты:

- *текстовый редактор*, предназначенный для ввода и редактирования текста программы на языке программирования;
- *транслятор;*
- *отладчик*, предназначенный для поиска и исправления ошибок в программе в диалоговом режиме;
- *загрузчик*, запускающий программу на выполнение, и др.

Для многих языков программирования созданы ***интегрированные среды***, включающие в себя все компоненты системы программирования, работающие под общим управлением. Примером такой интегрированной среды разработки программного обеспечения является система Turbo Pascal.

Можно выделить две большие группы языков программирования высокого уровня: ***директивные*** языки, называемые также *процедурными* или *императивными*, и ***декларативные*** языки.

Директивное программирование – это наиболее естественный для человека подход к написанию программ. Директивное программирование стали называть ***процедурным***, когда в процессе увеличения сложности и размера программ возникла концепция ***подпрограмм***. Подпрограмма позволяет локализовать в ней процесс выполнения определенных действий, который может быть повторен многократно с помощью механизма вызова.

Двумя основными классами декларативных языков программирования являются ***функциональные*** языки и ***логические*** языки. ***Функциональная программа*** состоит из совокупности определений функций, каждая из которых, в свою очередь, представляет собой вызовы других функций, а также инструкций, управляющих последовательностью вызовов. Каждая функция возвращает некоторое значение в вызвавшую ее функцию. Наиболее известным представителем функциональных языков программирования является язык Lisp (Лисп).

В ***логическом программировании*** программа рассматривается как набор логических фактов и правил вывода, а выполнение программы состоит в вычислении истинности некоторого утверждения. Prolog (Пролог) – наиболее известный язык логического программирования.

Языки Prolog и Lisp часто называют *языками искусственного интеллекта*, с их помощью решаются задачи создания экспертных систем и систем распознавания образов.

Существует несколько сотен языков программирования. Назовем и кратко охарактеризуем некоторые из них.

АССЕМБЛЕР и **МАКРОАССЕМБЛЕР** – языки низкого уровня, используемые для представления в символической форме программ, записанных на машинном языке.

ALGOL (Алгол) – название семейства языков программирования, которые оказали сильное влияние на все разработанные позднее директивные языки программирования.

FORTRAN (Фортран) – один из первых языков программирования высокого уровня директивного типа. Главное назначение – числовые расчеты с использованием сложных математических формул.

BASIC (Бейсик) – директивный язык программирования (1964 г.). Создавался как очень простой в изучении язык, которому отводилась роль первого шага на пути к изучению других языков программирования.

VISUAL BASIC – объектно-ориентированная среда визуального программирования, внутренний язык приложений Microsoft Windows. Использует базовые конструкции языка Basic.

PASCAL (Паскаль) – язык директивного типа, разработан Никлаусом Виртом в конце 60-х годов прошлого века для целей обучения программированию. PASCAL включает элементы, общие для большинства языков программирования, поддерживает объектно-ориентированное программирование.

DELPHI – высокопроизводительный инструмент построения приложений к базам данных, предоставляющий средства визуального программирования. В основе лежит объектно-ориентированный язык Object Pascal, который является расширением языка Pascal.

C и **C++** – языки программирования директивного типа. Язык C разработан в начале 70-х годов как язык для разработки операционной системы UNIX. C++ – расширение языка C. В частности, C++ поддерживает объектно-ориентированное программирование.

COBOL (Кобол) – один из первых языков программирования. Ориентирован на обработку больших массивов данных.

PROLOG (Пролог) – реализует концепцию логического программирования. На Прологе написаны многие экспертные системы, позволяющие делать выводы на основе имеющихся фактов и обширной базы правил, представленных в виде «если – тогда...».

LISP (Лисп) – язык обработки списков, является одним из основных средств моделирования различных аспектов искусственного интеллекта.

ADA – объектно-ориентированный язык, содержащий высокоуровневые средства программирования. Язык назван в честь Ады Лавлейс (1815–1852), дочери английского поэта лорда Байрона, которую принято считать *основоположницей программирования*. Она занималась разработкой программ для аналитической машины Бэббиджа.

JAVA – машинно-независимый объектно-ориентированный язык. Известно, что каждой аппаратной платформе, определяемой архитектурой центрального процессора и используемыми шинами, соответствуют совместимые с ней операционные системы и прикладные программы, которые могут на ней запускаться. Основной целью создания Java

была разработка языка программирования, независимого от платформы. Компилятор языка Java создает промежуточный код для некой абстрактной **виртуальной машины Java** (Java Virtual Machine – JVM). Чтобы обеспечить работу Java-программ на желаемой платформе, достаточно реализовать JVM для данной платформы. Виртуальная машина Java является интерпретатором для промежуточного кода.

PERL – язык программирования общего назначения, который был первоначально создан для манипуляций с текстом, но на данный момент используется для выполнения широкого спектра задач. Первые версии языка появились в конце 80-х годов XX века, в настоящее время язык продолжает развиваться, разрабатываются новые версии языка.

PYTHON – работает почти на всех компьютерных платформах. Реализует функциональную и объектно-ориентированную методологии программирования. Первая версия языка появилась в 1991 году. В настоящее время активно развивается, новые версии выходят примерно раз в два с половиной года.

2.3.2. Технологии программирования на языках высокого уровня

Технологией программирования называют совокупность методов и средств, используемых в процессе разработки программ. Основой для определения технологии программирования служит совокупность методов разработки программ, образующая ***методологию программирования***. В развитии программирования можно выделить несколько принципиально отличающихся методологий.

В начале 70-х годов XX века возникла ***структурная методология программирования*** (структурный подход к программированию). В основе структурного подхода лежит *декомпозиция сложных программных систем на части* с целью последующей реализации в виде отдельных небольших подпрограмм. Другим базовым принципом структурного программирования является *использование при составлении программ только базовых алгоритмических структур* (следование, ветвление, цикл), как следствие, *запрет на использование оператора безусловного перехода* (GOTO).

Проектирование программы осуществляется «сверху – вниз» с использованием метода пошаговой детализации. Сначала пишется текст основной программы, в котором вместо каждого связного логического фрагмента текста вставляется вызов подпрограммы, которая будет выполнять этот фрагмент. Вместо настоящих работающих подпрограмм в программу вставляются «заглушки», которые не выполняют никаких действий. Полученная программа проверяется и отлаживается. После того как программист убедится, что подпрограммы вызываются в правильной последовательности, то есть общая структура программы верна, подпрограммы-заглушки последовательно заменяются на реально работающие, причём разработка каждой подпрограммы ведётся тем же методом, что и разработка основной программы. Такая последовательность гарантирует, что на каждом этапе разработки программист может быть уверен, что общая структура всех более высоких уровней программы верна.

Дальнейший рост сложности и размеров разрабатываемого программного обеспечения потребовал развития структурирования данных. Появилась и стала развиваться технология модульного программирования. ***Модульное программирование*** предполагает выделение групп подпрограмм, использующих одни и те же данные, в библиотеки подпрограмм. Эту технологию поддерживают современные версии языков Pascal и C (C++), язык Ada и др.

Объектно-ориентированное программирование (ООП) определяется как технология создания программного обеспечения, основанная на представлении программы в виде совокупности объектов, каждый из которых является ***экземпляром*** определенного типа (***класса***), а классы образуют иерархию с наследованием свойств. ***Объект ООП*** – это совокупность данных (свойств объекта) и связанных с ними методов. Под ***методами объекта*** понимают процедуры и функции, объявление которых включено в описание объекта и которые выполняют те или иные действия.

Объектный подход предлагает новые способы организации программ, основанные на механизмах инкапсуляции, наследования, полиморфизма. ***Инкапсуляция*** – это механизм,

который объединяет в одно целое данные и методы, применяющиеся к этим данным, и защищает то и другое от неправильного использования. Когда методы и данные объединяются таким способом, создается объект.

Наследование – это процесс, посредством которого один объект может наследовать свойства другого объекта и добавлять к ним новые свойства, характерные только для него. В итоге создаётся иерархия объектных типов, где данные и методы «предков» автоматически являются данными и методами «потомков». Смысл и универсальность наследования заключается в том, что не надо каждый раз заново описывать новый объект, а можно указать «родителя» (базовый класс) и описать отличительные особенности нового класса. В результате новый объект будет обладать всеми свойствами родительского класса плюс своими собственными отличительными особенностями.

Полиморфизм – это свойство, которое позволяет одно и то же имя использовать для решения нескольких технически разных задач. Полиморфизм подразумевает такое определение методов, при котором метод с одним именем может применяться к различным родственным объектам.

Современное программирование использует ***компонентный подход***, который предполагает построение программ из отдельных компонентов. Компоненты можно распространять в двоичном виде (без исходных текстов) и использовать в любом языке программирования, поддерживающем соответствующую технологию.

? Вопросы и упражнения для самоконтроля

1. Что такое язык программирования?
2. По какому признаку языки программирования делят на языки программирования низкого уровня и высокого уровня?
3. К какой группе языков программирования (низкого уровня или высокого уровня) относятся языки АССЕМБЛЕР и МАКРОАССЕМБЛЕР?
4. Каково назначение процедуры трансляции при работе с программой на языке программирования высокого уровня?

5. Опишите механизм работы интерпретатора и компилятора при переводе текста программы с языка программирования высокого уровня на язык машинных кодов.

6. Какие элементы включает система программирования на языке высокого уровня?

7. Какие разновидности языков программирования используются в декларативном программировании?

8. В чем суть структурной методологии программирования?

9. На чем основана модульная технология программирования?

10. Опишите основные понятия объектно-ориентированного программирования: объект, метод, класс, экземпляр, инкапсуляция, наследование, полиморфизм.

11. На чем основан компонентный подход к программированию?

2.4. Основы языка программирования Паскаль

2.4.1. Структура программы на языке Паскаль. Основные объекты программы

Программа на Паскале состоит из раздела описаний и раздела операторов. Каждый элемент ***раздела описаний*** начинается соответствующим служебным словом (*uses, var, const, label, type, procedure, function*), после которого идет последовательность имен, разделенных запятой. Например, описание используемых в программе переменных можно было бы записать в следующем виде:

var a1, a2, a3: integer;

Раздел операторов начинается служебным словом *begin*. Далее следуют операторы программы, которые разделяются между собой символом «точка с запятой». Заканчивается раздел служебным словом *end*, после которого ставится символ «точка».

Тексты программ на языке программирования записываются с помощью его алфавита. ***Алфавит языка Паскаль*** включает следующие группы символов:

• прописные и строчные буквы латинского алфавита: *A, B, ..., Y, Z, a, b, ..., y, z ;*
• цифровые символы: *0, 1, 2, ..., 9*;
• специальные символы: *+, –, *, /, >, <, =, ;, #, ‘, :, { , }, [,], (,)* и др.

☑ Буквы русского алфавита можно использовать только в комментариях внутри фигурных скобок (например, *{это комментарий}*) или в строковых константах внутри апострофов (например, *‘это строковая константа’*).

Из символов алфавита составляются ***лексемы***, то есть минимальные единицы языка, имеющие самостоятельный смысл. Лексемы языка программирования аналогичны словам естественного языка. Среди лексем выделяют зарезервированные слова, стандартные идентификаторы и идентификаторы пользователя.

Зарезервированные слова имеют фиксированное написание и строго определенный смысл (*program, begin, if, for, function, while, repeat* и др.). Они не могут изменяться программистом, и их нельзя использовать в качестве имен для обозначения других объектов программы. Для обозначения определенных разработчиками языка стандартных объектов служат ***стандартные идентификаторы***. Например, стандартный идентификатор *Sqr* используется для вызова функции, которая возводит в квадрат некоторое заданное число.

Идентификаторы пользователя – это имена, которые присваивает объектам программы сам программист. При выборе идентификаторов пользователя необходимо соблюдать правила их записи:

• идентификатор может состоять из букв, цифр и символа подчеркивания;
• идентификатор всегда начинается с буквы или символа подчеркивания.

Задача любой программы состоит в обработке данных. В Паскале данные могут быть значениями констант и переменных. ***Константами*** называются такие данные, которые не изменяются в процессе выполнения программы в отличие от ***переменных***, которые могут менять свои значения. Переменная обозначает область оперативной памяти компьютера, в которую можно записать какое-либо значение.

Тип данных – это множество значений, объединенных определенной совокупностью допустимых для них операций и способом представления в памяти компьютера. Различные типы отличаются диапазоном значений и требуемым для их хранения объемом памяти. Каждому стандартному типу соответствует специальное зарезервированное слово для его обозначения (*integer, real, char* и др.). Рассмотрим основные стандартные типы данных, используемые в языке Паскаль.

Целые типы данных (табл. 2.2) используются для представления целых чисел.

Вещественные типы данных используются для представления значений, которые могут иметь дробную часть. Основной вещественный тип ***real*** используется для представления чисел с дробной частью, содержащей 11–12 знаков после запятой, требует для размещения 6 байт. При записи в программе числа с дробной частью в качестве разделителя целой и дробной части используется символ «точка».

Таблица 2.2

Характеристики целых типов данных

Тип	Название	Диапазон возможных значений	Требуемая память в байтах
Integer	Целое	-32768 ... 32767 (-2^{15}... (2^{15}-1))	2
Longint	Длинное целое	-2^{31} ... (2^{31}-1)	4
Byte	Байт	0 ... 255 (0 ... (2^{8}-1))	1
Shortint	Короткое целое	-128 ... 127 (-2^{7} ... (2^{7}-1))	1
Word	Слово	0 ... 65535 (0 ... (2^{16}-1))	2

Символьный (литерный) тип char определяется множеством значений кодовой таблицы компьютера. Каждому символу приписывается целое число в диапазоне от 0 до 255. В памяти хранится код символа, представленный в двоичной системе счисления. ***Логический (булевский) тип boolean*** используется для представления логических данных, которые могут принимать только два значения: *true* (истина) и *false*

(ложь). Для размещения в памяти данных литерного и логического типа требуется 1 байт.

К данным целых и вещественных типов могут применяться ***арифметические операции*** (табл. 2.3). В зависимости от операции и типа операндов (данных, к которым применяется операция) результат операции может иметь различный тип. Например, результат применения операции сложения к целым данным будет также целым, а результат применения той же операции к вещественным данным приведет к получению значения вещественного типа. Операция деления в любом случае дает результат вещественного типа.

Таблица 2.3

Арифметические операции языка Паскаль

Операция	Действие	Операция	Действие
+	Сложение	*	Умножение
–	Вычитание	/	Деление
DIV, MOD	Целая часть результата деления одного целого числа на другое и остаток от деления целых чисел		

☑ Операции DIV и MOD могут применяться только к данным целого типа, результат также является целым. Так, для нахождения целой части результата деления числа **17** на число **5** следует записать 17 DIV 5. Результат операции в данном случае будет равен 3. Для нахождения целого остатка от деления числа **17** на число **5** следует записать 17 MOD 5. Результат в этом случае будет равен 2.

☑ В Паскале отсутствует операция возведения числа в степень. Возведение числа ***x*** в целую степень заменяется операцией умножения: $y = x^n = \underbrace{x \cdot x \cdot \ldots \cdot x}_{n \quad раз}$. Для вычисления значения функции x^n, где ***n*** – вещественное (*real*), используется формула $y = x^n = e^{n \cdot \ln(x)}$.

Стандартные функции. В языке Паскаль существует ряд заранее разработанных функций, которые можно использовать как готовые объекты. Чтобы воспользоваться такой

функцией, нужно указать ее имя и аргумент функции, который обязательно должен быть заключен в круглые скобки. Например, для вычисления значения e^{x+1} нужно записать *Exp(x+1)*. Здесь *Exp* – имя стандартной функции, *x+1* – аргумент функции. В табл. 2.4 описаны стандартные математические функции.

Таблица 2.4

Стандартные математические функции языка Паскаль

Вызов функции	Функция
Abs (x)	Абсолютное значение (модуль)
Arctan (x)	Арктангенс
Cos (x)	Косинус
Exp (x)	e^x – экспонента
Frac (x)	Дробная часть
Int (x)	Целая часть (возвращает результат вещественного типа)
Trunc (x)	Целая часть (возвращает результат целого типа)
Ln (x)	Натуральный логарифм
Sin (x)	Синус
Sqr (x)	Возведение в квадрат
Sqrt (x)	Корень квадратный

Выражение – это синтаксическая единица языка, определяющая способ вычисления некоторого значения. ***Арифметическое выражение*** задает способ вычисления числового значения. Оно строится из числовых констант, переменных, функций с использованием арифметических операций и круглых скобок. Пример арифметического выражения: ***ln(x) * ln(x) – cos(a * a)***.

Логические выражения строятся посредством операций отношения, логических операций и круглых скобок из логических переменных и констант, а также выражений других типов. Результатом вычисления логического выражения является одно из двух логических значений (*true* или *false*).

Операции отношения (= (равно), <> (не равно), > (больше), < (меньше), >= (больше либо равно), <= (меньше либо равно)) выполняют сравнение двух операндов и определяют результат сравнения как значение логического типа (*true* или *false*). Например, логическое выражение $x > y$ имеет значение *true* при $x = 10$, $y = 2$, но имеет значение *false* при $x = 1$, $y = 2$. В Паскале используются следующие ***логические операции***: ***not*** – логическое отрицание, ***and*** – логическое умножение, ***or*** – логическое сложение, ***xor*** – исключающее «или». В табл. 2.5 приведены результаты применения логических операций к разным значениям операндов. Использованы обозначения: 1 – *true*, 0 – *false*. Пример логического выражения: ***(x > 0) and (y > 0)***.

Таблица 2.5

Таблицы истинности для логических операций

A	*B*	*not A*	*A and B*	*A or B*	*A xor B*
1	1	0	1	1	0
1	0	0	0	1	1
0	0	1	0	0	0
0	1	1	0	1	1

Ввод данных – это передача данных в оперативную память для обработки, ***вывод данных*** – передача после обработки из оперативной памяти на внешний носитель информации (экран монитора, принтер и другие устройства). ***Ввод данных с клавиатуры*** выполняется с помощью стандартной процедуры *Read*. Общий вид оператора вызова процедуры:

Read (список переменных);

В списке переменных перечисляются имена переменных. Значения этих переменных при выполнении программы вводятся пользователем через пробел с клавиатуры. После набора данных для одной процедуры *Read* нажимается клавиша **<Enter>**. Процедура чтения *Readln* аналогична процедуре *Read* с той разницей, что после считывания последнего в списке значения курсор переходит на начало новой строки экрана.

Пример 2.1

```
program Prim_2_1;
var m, k : integer ;
    c, d : real ;
begin
  Readln ( c, d ) ;
  Read ( m, k ) ;
...
end.
```

При выполнении программы следует ввести сначала два числа (числа могут иметь дробную часть), разделив их при вводе символом «пробел». Переменной ***c*** присваивается значение, равное первому введенному числу, переменной ***d*** – значение, равное второму введенному числу. После ввода этих значений курсор переходит на начало новой строки экрана. Далее требуется ввести еще два целых числа, значения которых будут присвоены переменным ***m*** и ***k***.

Вывод данных на экран выполняется с помощью стандартной процедуры *Write*, оператор вызова которой имеет следующий вид:

Write (список вывода);

В списке вывода могут быть представлены константы, переменные, выражения и произвольный текст, заключенный в апострофы. Процедура производит вывод значений констант, переменных, выражений на экран. Процедура *Writeln* аналогична процедуре *Write*. Отличие состоит в том, что после вывода последнего в списке выражения курсор переходит на начало новой строки экрана.

☑ В процедурах *Write* и *Writeln* можно указать формат, определяющий ширину поля вывода и количество выводимых разрядов дробной части числа, если число вещественного типа. Для переменной вещественного типа ***x*** и переменной целого типа ***y*** запись ***Write (x : 6 : 2, y : 6)*** предполагает выполнение следующих действий:

1) вывести на экран значение переменной ***x*** с 2 разрядами в дробной части числа, используя для вывода 6 знакомест строки вывода экрана;

2) вывести на экран значение целой переменной ***y***, используя для вывода 6 знакомест строки вывода.

Если заданного количества знакомест недостаточно для вывода числа, поле вывода автоматически расширяется

до нужного размера. В табл. 2.6 приведены примеры записи операторов вывода данных.

Таблица 2.6

Примеры организации вывода данных

Пример оператора	Комментарий
Write(‘ Привет ‘);	на экран выводится текст **Привет**;
Write(‘ b= ‘, b, ‘ d= ‘, d);	на экран выводятся значения переменных ***b*** и ***d*** с сопроводительными комментариями;
Write (k : 6);	для вывода значения переменной целого типа ***k*** отводится 6 знакомест строки экрана;
Write (m : 8 : 2);	для вывода значения переменной вещественного типа ***m*** отводится 8 знакомест строки экрана, при этом будут выводиться 2 разряда дробной части числа

? Вопросы и упражнения для самоконтроля

1. Опишите структуру программы на языке Паскаль.
2. Какие группы символов образуют алфавит языка Паскаль? В каких случаях допустимо использование в программе символов русского алфавита?
3. Какие объекты языка программирования называются зарезервированными словами? Каковы правила их использования в программе?
4. Какие правила необходимо соблюдать при выборе имен (идентификаторов) для обозначения в программе объектов, вводимых программистом?
5. Объясните смысл понятий «константа» и «переменная» в языке Паскаль.
6. Что такое «тип данных»? Укажите основные стандартные типы данных языка Паскаль.
7. Перечислите арифметические операции языка Паскаль. Какими средствами реализуется операция возведения числа в степень?

8. Что такое «стандартная функция» в Паскале? Перечислите основные арифметические стандартные функции.

9. Из каких элементов строится арифметическое выражение в Паскаль-программе? Что является результатом вычисления арифметического выражения?

10. Из каких элементов строится логическое выражение в Паскаль-программе? Что является результатом вычисления логического выражения?

11. Какие стандартные процедуры языка Паскаль используются для ввода с клавиатуры значений переменных и для вывода на экран данных при выполнении программы?

2.4.2. Операторы языка Паскаль

Оператором называется предложение языка программирования, задающее полное описание некоторого действия, которое необходимо выполнить.

Оператор присваивания и составной оператор

Для присваивания значений переменным используется ***оператор присваивания***. В его левой части указывается имя переменной, в правой части – выражение. Символ «**:=**» означает «присвоить значение». Оператор присваивания выполняется в два этапа. Первый этап – вычисление значения выражения, указанного справа от символа присваивания, второй этап – присваивание вычисленного значения переменной, указанной слева от символа присваивания. Например, при выполнении оператора ***b := 32 * a + Sin(x);*** сначала будет вычислено значение в правой части оператора, затем будет выполнено присваивание полученного значения переменной ***b***.

Составной оператор представляет собой совокупность произвольного числа операторов, отделенных друг от друга точкой с запятой, и ограниченную операторными скобками *begin* и *end*:

```
begin
  оператор 1;
  оператор 2;
  ...
end;
```

Эта конструкция рассматривается в программе как один оператор.

Условные операторы

Эти операторы предназначены для выбора к исполнению одного из возможных действий в зависимости от некоторого условия. Одним из таких операторов является оператор *if*, существующий в двух формах:

1) полный оператор: ***if <условие> then <оператор 1> else <оператор 2>;***

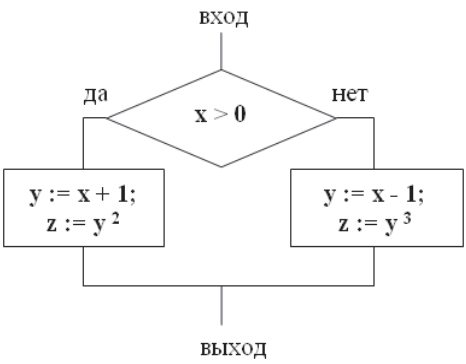
2) короткий оператор: ***if <условие> then <оператор>;***

Операторы реализуют базовые алгоритмические структуры «ветвление» (см. рис. 2.3). Примеры записи оператора *if* иллюстрирует табл. 2.7.

Таблица 2.7

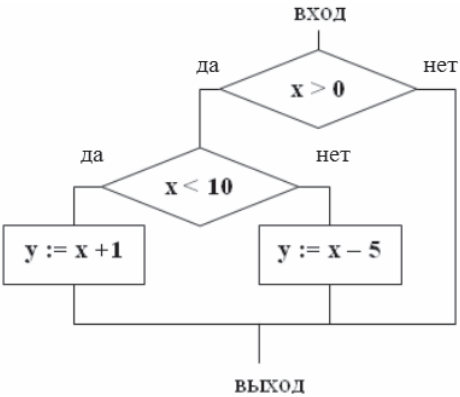
Примеры использования оператора ***if***

Пример записи оператора	Действие оператора

if x > 0 then
begin
*y := x + 1; z := y * y*
end
else
begin
*y := x – 1; z := y * y * y*
end;
	Действие оператора иллюстрирует следующая блок-схема:
	 

Окончание табл. 2.7

Пример записи оператора	Действие оператора

if x > 0 then
if x < 10 then
y := x + 1
else y := x – 5 ;
 	
В данном фрагменте программы может возникнуть вопрос об установлении соответствия между *if* и *else*. По правилам языка Паскаль *else* всегда относится к самому близкому *if*, в данном случае – ко второму *if*
	Действие оператора иллюстрирует следующая блок-схема:
	 

if x > 0 then
y := x + 1
else
y := x – 10;
	Если $x > 0$, то переменной y присваивается значение выражения $x + 1$, иначе переменной y присваивается значение выражения $x - 10$

☑ **Пример 2.2**. Составить программу вычисления значения функции y для произвольного x, если

$$y = \begin{cases} \dfrac{\sin^2 x}{2}, & \text{если } x \le 0 \\ x, & \text{если } 0 < x \le 1 \\ x^3, & \text{если } 1 < x \le 3 \\ 27 + (x-3)^3, & \text{если } x > 3 \end{cases}.$$

Исходные данные: x – аргумент функции, любое число.
Требуется найти: значение функции y при заданном x.

Алгоритм

Следует проанализировать введенное значение x на принадлежность его одному из четырех возможных интервалов числовой оси (рис. 2.8).

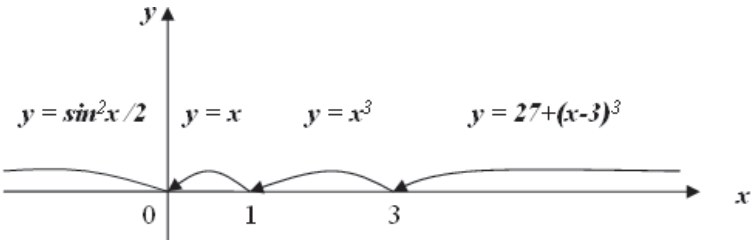


Рис. 2.8. Точки ветвления функции ***y***

Проверку точек ветвления целесообразно начать с одной из крайних точек. Следует учитывать, что каждая из точек ветвления задается двумя отношениями. Из этих отношений в операторе *if* всякий раз следует проверять то, для которого по направлению *then* можно выделить расчетную формулу. В нашем примере начать проверку можно либо с точки $x = 3$, либо с точки $x = 0$. Пусть в качестве первой точки для проверки выбрана точка $x = 3$. Для проверки в операторе *if* следует выбрать отношение $x > 3$. При выборе этого отношения в ветви *then* можно выполнить вычисления по формуле $y = 27 + (x - 3)^3$. Общее количество операторов *if* не превосходит количества точек ветвления. В данном примере таких точек три ($x = 3$, $x = 1$ и $x = 0$). Блок-схема алгоритма представлена на рис. 2.9.

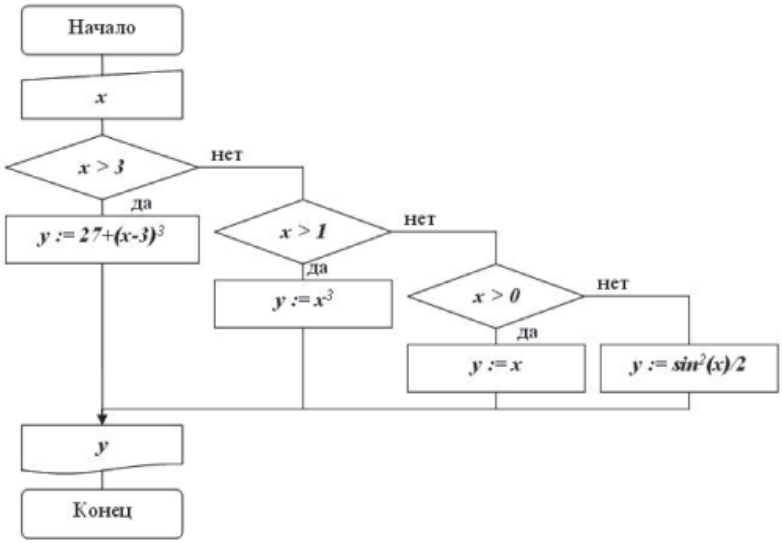


Рис. 2.9. Блок-схема алгоритма для примера 2.2

Программа

```
program Prim_2_2 ;
var x, y : real ;
begin
  Writeln ( ' Введите значение x ' ) ;
  Readln ( x ) ;
  if x > 3 then y := 27 * Sqr ( x – 3 ) * ( x – 3 )
  else
    if x > 1 then y := Sqr ( x ) * x
    else
      if x > 0 then y := x else y := Sqr ( Sin ( x ) ) / 2 ;
    Writeln ( ' x = ', x : 6 : 2, ' y = ', y : 8 :2);
end.
```

Операторы организации цикла

Если в программе возникает необходимость неоднократного выполнения некоторых операторов, то для этого используются специальные операторы. Если число требуемых повторений цикла заранее известно, то используется ***оператор цикла с параметром***, который имеет два варианта записи:

for **<имя переменной> := <начальное значение>** ***to*** **<конечное значение>** ***do*** **<тело цикла>;**

или

for **<имя переменной> := <начальное значение>** ***downto*** **<конечное значение>** ***do*** **<тело цикла>;**

Здесь <имя переменной> – параметр цикла, переменная порядкового, чаще всего – целого типа; <начальное значение> – начальное значение параметра цикла; <конечное значение> – конечное значение параметра цикла; <тело цикла> – простой или составной оператор, для выполнения которого организован цикл. Цикл повторяется до тех пор, пока значение параметра лежит в интервале между начальным и конечным значениями.

☑ В первом варианте при каждом повторении цикла значение параметра цикла увеличивается на 1, во втором – уменьшается на 1. Оператор *for* реализует базовую алгоритмическую структуру «цикл с параметром» (см. рис. 2.4).

Оператор while реализует базовую алгоритмическую структуру «цикл с предусловием» (см. рис. 2.5). Общий вид оператора:

***while* <условие> *do* <тело цикла>;**

Здесь <тело цикла> – простой или составной оператор. Перед каждым выполнением тела цикла вычисляется значение логического выражения <условие>. Если результат равен «истина», тело цикла выполняется и снова вычисляется выражение условия. Если результат равен «ложь», происходят выход из цикла и переход к следующему оператору программы.

Оператор цикла с постусловием repeat реализует алгоритмическую структуру «цикл с постусловием» (см. рис. 2.5). Общий вид оператора:

```
Repeat
 оператор 1;
 оператор 2;
 ...
until <условие>;
```

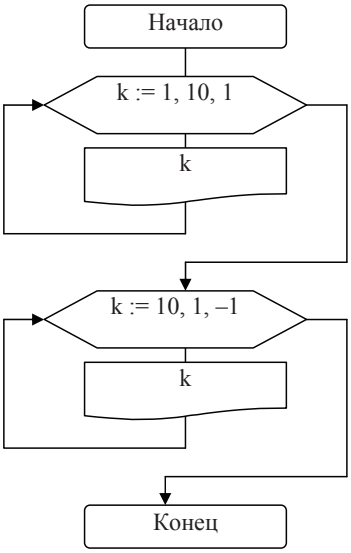
Здесь *оператор 1; оператор 2; ... оператор n* образуют тело цикла. Цикл будет выполняться, пока логическое условие после слова *until* ложно. Как только условие станет истинным, произойдет выход из цикла. Проверка условия выполняется в конце каждой итерации цикла. Этот вид цикла применяется в тех случаях, когда тело цикла необходимо обязательно выполнить хотя бы один раз.

Пример 2.3. Вывести на экран числа от 1 до 10 сначала в прямом порядке, затем – в обратном. Блок-схема и программа решения задачи представлены в табл. 2.8.

Таблица 2.8

Блок-схема и программа для примера 2.3

Блок-схема	Программа

	*program Prim_2_3 ;*
	var k : byte;
	begin
	Writeln (‘ В прямом порядке ‘);
	for k := 1 to 10 do
	Writeln (k);
	Writeln(‘ В обратном порядке ‘);
	for k := 10 downto 1 do
	Writeln (k);
	end.

Пример 2.4. Найти сумму 10 произвольных целых чисел, значения которых вводятся с клавиатуры.

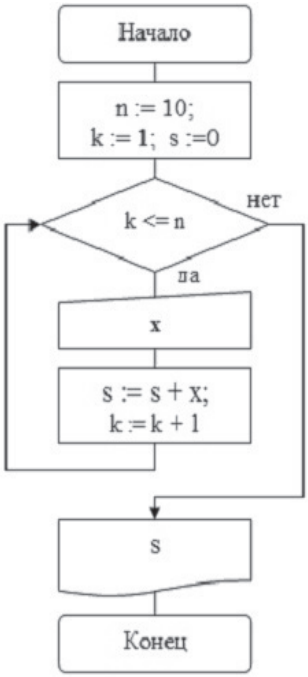
Блок-схема и программа решения задачи представлены в табл. 2.9.

Комментарий к программе. В строке 1 присваиваются начальные значения переменной ***k*** (номер вводимого с клавиатуры числа) и ***s*** (сумма чисел, введенных с клавиатуры). В строке 2 записан оператор цикла с предусловием, при выполнении которого проверяется условие ***k <= n*** (номер вводимого с клавиатуры числа меньше либо равен 10). Если условие равно «истина» (при ***k*** = 1, 2, ..., 10), то выполняются операторы тела цикла, входящие в составной оператор *begin ... end* (операторы в строках 4, 5, 6). При ***k*** = 11 происходит выход из цикла и на печать выводится вычисленная сумма ***s***.

Таблица 2.9

Блок-схема и программа для примера 2.4

Блок-схема	Программа

	*program Prim_2_4 ;*
	const n = 10 ;
	var k, x, s : integer ;
	{k – номер числа, x – вводимое с клавиатуры число; s – сумма введенных чисел }
	begin
	k := 1; s := 0 ; {1}
	while k <= n do {2}
	begin
	Write (‘ Введите число ‘) ; {3}
	Readln (x) ; {4}
	s := s + x ; {5}
	k := k + 1 ; {6}
	end;
	Writeln(‘ Сумма чисел равна ‘, s) ;
	end.

☑ **Пример 2.5.** С клавиатуры вводится целое число ***x***. По условию задачи требуется, чтобы число было в диапазоне от 10 до 50. Организовать проверку вводимого значения на соответствие условию.

Программа

```
program Prim_2_5;
var x : integer;
begin
  repeat
    Write( ‘ Введите число из диапазона 10 – 50 ‘ ); Readln ( x ) ;
  until ( x >= 10 ) and ( x <= 50 ) ;
...
end.
```

Комментарий к программе. При входе в цикл выполняется оператор *Write*, который выводит на экран подсказку «Введите число из диапазона 10–50». Следующий оператор *Readln (x)* вводит в переменную ***x*** число. Затем проверяется выполнение условия *(x >= 10) and (x <= 50)*. Если введено число из диапазона значений 10–50, условие выполняется и происходит выход из цикла. Если введено число, не удовлетворяющее заданному условию, управление передается на начало цикла. Вновь выводится подсказка о вводе числа, вводится новое число, и так будет продолжаться до тех пор, пока не будет введено число из диапазона 10–50.

? Вопросы и упражнения для самоконтроля

1. Что называется оператором языка программирования? Каким символом разделяются между собой операторы в программе на языке Паскаль?
2. Какова форма записи оператора присваивания? Какие действия реализуются при выполнении оператора присваивания?
3. Опишите структуру и назначение составного оператора языка Паскаль.
4. Какую базовую алгоритмическую структуру реализует оператор *if*? Опишите действие полного оператора *if*. В чем заключается отличие краткого оператора *if* от полного?
5. Каково назначение операторов цикла? Опишите структуру оператора цикла с параметром.
6. Чем оператор *for ... to ... do* отличается от оператора *for ... downto ... do*?
7. Какого типа переменная может быть использована в качестве параметра цикла оператора *for*?
8. Какую базовую алгоритмическую структуру реализует оператор *while ... do*? Возможна ли ситуация, когда при выполнении оператора *while ... do* тело цикла не будет выполнено ни разу?
9. Какую базовую алгоритмическую структуру реализует оператор *repeat ... until?* Возможна ли ситуация, когда при выполнении оператора *repeat ... until* тело цикла не будет выполнено ни разу?

2.4.3. Обработка массивов

Массив – это упорядоченная последовательность данных, состоящая из фиксированного числа элементов, имеющих один и тот же тип, и обозначаемая одним общим именем. Каждый элемент массива имеет уникальный номер (индекс или совокупность индексов). Перед использованием в программе массив должен быть объявлен в её описательной части.

Одномерный массив – это массив, в описании которого указан только один индекс. При описании указывается диапазон значений индекса и тип данных в массиве. Например:

var a : array [1.. 20] of real ;

Объявляется одномерный массив ***a***, содержащий значения типа *real*, состоящий из 20 элементов.

Доступ к каждому конкретному элементу массива осуществляется по имени с указанием значения индекса (номера) элемента. Например: ***a [1]*** – элемент массива с номером 1; ***a [i]*** – элемент массива с номером ***i***.

Двумерный массив – это массив, в описании которого указаны два индекса. Двумерный массив можно рассматривать как матрицу или таблицу, в которой каждый элемент однозначно определяется номером строки и номером столбца, на пересечении которых он находится. При этом первый индекс определяет номер строки, второй – номер столбца. Например, если в массиве количество строк равно количеству столбцов и равно 3, то массив можно представить в следующем виде:

a [1, 1]	***a [1, 2]***	***a [1, 3]***
a [2, 1]	***a [2, 2]***	***a [2, 3]***
a [3, 1]	***a [3, 2]***	***a [3, 3]***

Доступ к каждому конкретному элементу массива осуществляется по имени с указанием значений индексов элемента: ***a [1, 2]*** – элемент массива, расположенный на пересечении строки с номером 1 и столбца с номером 2. Двумерный массив можно объявить, например, следующим образом:

var a : array [1.. 10, 1.. 10] of integer;

Здесь объявляется двумерный массив ***a***, содержащий данные типа *integer*, состоящий из 10 строк и 10 столбцов.

Ввод и вывод значений элементов одномерного массива

Для ввода с клавиатуры значений элементов массива или вывода значений на экран используются циклы с параметром, в которых в качестве параметра используется текущий номер элемента в массиве.

Пример 2.6. Задана последовательность чисел вещественного типа. Используя для представления чисел одномерный массив, составить программу, которая позволит выполнить следующие действия:

- ввести с клавиатуры количество элементов массива и значения элементов массива;
- изменить все элементы массива, увеличив их значения на 1;
- вывести полученный массив на экран.

Программа

```
program Prim_2_6;
var
a : array [ 1..10 ] of real ;
i, n : byte ;
begin
{ ввод количества элементов одномерного массива }
Writeln( ' Введите кол-во элементов массива 0<n<=10 ' );
Readln ( n );
{ ввод значений элементов одномерного массива }
for i :=1 to n do
  begin
    Writeln ( ' Введите ', i, ' элемент массива ' );
    Readln ( a [ i ] );
  end;
{ изменение значений элементов массива и вывод значений
на экран }
for i := 1 to n do
  begin
    a [ i ] := a [ i ] + 1; Writeln ( a [ i ] : 8 : 2 );
  end;
end.
```

Ввод и вывод значений элементов двумерного массива

☑ При работе с двумерными массивами используются циклические структуры «цикл в цикле» следующего вида:

```
for i := 1 to n do
 for k := 1 to m do
   begin
    <тело цикла>
   end;
```

Такие циклы работают по принципу «часовой и минутной стрелок», т. е. при каждом значении параметра внешнего цикла (переменная ***i***) параметр внутреннего цикла (переменная ***k***) пробегает все множество допустимых значений ***1, 2, …, m***. Тело цикла будет выполнено ***n*** · ***m*** раз. Таким образом, будут последовательно перечислены следующие пары значений параметров внешнего и внутреннего циклов:

i	1	1	...	1	2	2	...	2	...	n	n	...	n
k	1	2	...	m	1	2	...	m	...	1	2	...	m

Пример 2.7. Задана матрица, состоящая из ***n*** строк и ***m*** столбцов. Используя для представления чисел двумерный массив, составить программу, которая позволит выполнить следующие действия:

- ввести с клавиатуры количество строк и столбцов в массиве, а также значения элементов массива;
- вывести элементы массива на экран в виде матрицы;
- вычислить сумму значений элементов массива и вывести на экран полученное значение суммы.

Программа

```
program Prim_2_7;
b : array [ 1.. 10, 1.. 10 ] of real ;
var i, k, n, m : byte ; { n – количество строк, m – количество столбцов в массиве b; i, k – текущие значения индексов элементов массивов}
sum : real ; { sum – сумма значений элементов массива }
begin
Writeln ( ' Введите количество строк 0<n<=10 ' );
Readln ( n );
```

```
Writeln ( ' Введите количество столбцов 0<m<=10 ' );
Readln ( m );
{ ввод элементов двумерного массива построчно }
for i := 1 to n do
  begin
    Writeln( ' Вводите элементы', i, ' строки ' );
    for k := 1 to m do Readln( b [ i, k ] );
  end;
{ вывод на экран элементов двумерного массива }
Writeln( ' Массив b ' );
for i := 1 to n do
  begin
    for k := 1 to m do
    Write ( b [ i, k ] : 6 : 2); {вывод на экран элементов одной
    строки}
    Writeln; {перевод курсора на новую строку}
  end;
{вычисление суммы значений элементов двумерного
массива}
sum := 0;
for i := 1 to n do
  for k := 1 to m do
    sum := sum + b [ i, k ];
Writeln( ' Сумма элементов массива = ', sum : 8 : 2 );
end.
```

Обработка одномерных массивов

Обработка одномерных массивов предполагает выполнение типовых операций над элементами массива, таких как вычисление суммы и произведения значений элементов массива, определение количества элементов, удовлетворяющих некоторому условию, определение минимального и максимального значений элементов. В табл. 2.10 приводятся примеры выполнения типовых операций с одномерными массивами. Рассматривается одномерный массив ***a***, состоящий из ***n*** элементов.

Таблица 2.10

Типовые процедуры обработки одномерных массивов

<table>
<tr><th colspan="2">Вычисление суммы значений элементов одномерного массива</th></tr>
<tr><td>Задача</td><td>Фрагмент программы</td></tr>
<tr><td>Вычисление суммы и среднего арифметического значения элементов одномерного массива</td><td>sum := 0;
for i := 1 to n do
  sum := sum + a [i];
sred := sum / n;
Writeln (‘ Сумма = ‘, sum);
Writeln (‘ Среднее значение = ‘, sred);</td></tr>
<tr><td colspan="2">Комментарий. Перед входом в цикл значение переменной для вычисления суммы (sum) обнуляется. В цикле, добавив к сумме значение элемента с номером i, записываем полученное значение в ту же самую переменную sum. Выполнив эту операцию n раз (при i = 1, 2, …, n), после выхода из цикла получаем в переменной sum сумму всех значений элементов массива.
Среднее арифметическое значение получается делением найденной суммы на количество элементов в массиве (n)</td></tr>
<tr><td>Задача</td><td>Фрагмент программы</td></tr>
<tr><td>Вычисление суммы элементов массива, значения которых больше 1 и меньше 3</td><td>sum := 0;
for i := 1 to n do
  if (a [i] > 1) and (a [i] < 3) then
  sum := sum + a [i] ;
Writeln (‘ Сумма = ‘, sum);</td></tr>
<tr><th colspan="2">Вычисление произведения значений элементов одномерного массива</th></tr>
<tr><td>Задача</td><td>Фрагмент программы</td></tr>
<tr><td>Вычисление произведения значений элементов массива</td><td>p := 1;
for i := 1 to n do
  p := p * a [i];
Writeln (‘ Произведение = ‘, p);</td></tr>
</table>

Комментарий. Перед входом в цикл переменной ***p*** присваивается значение 1. В цикле, умножив значение переменной ***p*** на значение элемента с номером ***i***, записываем полученное значение в ту же самую переменную ***p***. Выполнив эту операцию ***n*** раз (при ***i = 1, 2, …, n***), после выхода из цикла получаем в переменной ***p*** произведение всех значений элементов массива	
Вычисление количества значений элементов одномерного массива, удовлетворяющих некоторому условию	
Задача	Фрагмент программы
Вычисление количества положительных по значению элементов массива	*kol := 0;* *for i := 1 to n do* *if a [i] > 0 then kol := kol + 1;* *Writeln (‘ Количество = ‘, kol);*
Задача	Фрагмент программы

Вычисление количества элементов массива, имеющих значения в диапазоне от –2 до +5	*kol :=0;*
	for i := 1 to n do
	if (a [i] >= –2) and (a [i] <= 5) then
	kol:= kol+1;
	Writeln (‘ Количество = ‘, kol);

Определение наибольшего и наименьшего значений элементов в одномерном массиве	
Задача	Фрагмент программы

Вычисление наибольшего значения среди всех элементов массива и вывод на экран номеров элементов, равных по значению наибольшему элементу	*max := a [1];*
	for i := 2 to n do
	if max < a [i] then max := a [i];
	Writeln (‘ Наибольшее значение = ‘, max);
	for i := 1 to n do
	if max = a [i] then Writeln (‘ Номер = ‘, i);

Комментарий. Сначала за наибольшее значение принимается значение элемента массива с номером 1. Затем в цикле по переменной ***i*** наибольшее значение (переменная ***max***) сравнивается с каждым из элементов массива ***a [i]***. Если при сравнении оказывается, что сравниваемый элемент массива имеет большее значение, чем ***max***, то значение этого элемента принимается за наибольшее. Для вычисления **наименьшего значения** потребуется лишь изменить знак сравнения в операторе *if* на противоположный.
Для вывода на экран номеров элементов, равных по значению наибольшему (их может быть несколько), выполняется сравнение значения каждого элемента с найденным в предыдущем цикле наибольшим значением на совпадение. В случае совпадения значений на экран выводится номер элемента

Преобразование элементов одномерного массива	
Задача	Фрагмент программы

Увеличение значений всех элементов массива на величину заданного числа ***r*** и вывод на экран элементов получившегося массива	*for i := 1 to n do*
	begin
	a [i] := a [i] + r;
	Write (‘ a [‘, i, ‘]= ‘, a [i]);
	end;

Задача	Фрагмент программы
Замена нулевыми значениями тех элементов массива, значения которых меньше 5, и вывод на экран элементов получившегося массива	*for i := 1 to n do* *if a [i] < 5 then a [i] := 0;* *Writeln (‘ Новый массив a ‘);* *for i := 1 to n do* *Writeln (‘ a [‘, i, ‘]= ‘, a [i]);*

Обработка двумерных массивов

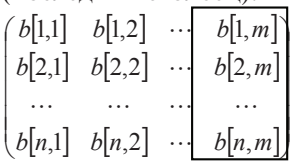
При обработке двумерных массивов операции могут выполняться с элементами массива в целом либо с отдельными строками или столбцами массива. При обработке всего массива важно правильно выбрать последовательность обработки – по строкам или по столбцам. Приводимые в табл. 2.11 примеры обработки двумерного массива иллюстрируют

указанные особенности. Далее будем рассматривать двумерный массив ***b***, состоящий из ***n*** строк и ***m*** столбцов, содержащий элементы вещественного типа.

Таблица 2.11

Типовые процедуры обработки двумерных массивов

<table>
<tr><th colspan="2">Операции с отдельными строками и столбцами двумерного массива</th></tr>
<tr><td>Задача</td><td>Фрагмент программы</td></tr>
<tr><td>Вычисление суммы значений элементов строки с номером 1 двумерного массива:
$\begin{pmatrix} b[1,1] & b[1,2] & \cdots & b[1,m] \\ b[2,1] & b[2,2] & \cdots & b[2,m] \\ \cdots & \cdots & \cdots & \cdots \\ b[n,1] & b[n,2] & \cdots & b[n,m] \end{pmatrix}$</td><td>sum := 0;
for k := 1 to m do
sum := sum + b [1, k] ;
Writeln (‘ Сумма = ‘, sum);
Комментарий. Суммируются элементы строки с номером 1, поэтому первый индекс полагается равным 1, второй индекс ***k*** изменяется в цикле от 1 до ***m*** с шагом 1</td></tr>
<tr><td>Задача</td><td>Фрагмент программы</td></tr>
<tr><td>Вычисление суммы элементов, имеющих значения меньше 1, в последней строке (строка с номером ***n***) двумерного массива:
$\begin{pmatrix} b[1,1] & b[1,2] & \cdots & b[1,m] \\ b[2,1] & b[2,2] & \cdots & b[2,m] \\ \cdots & \cdots & \cdots & \cdots \\ b[n,1] & b[n,2] & \cdots & b[n,m] \end{pmatrix}$</td><td>sum := 0;
for k := 1 to m do
if b[n, k] < 1 then
sum := sum + b [n, k];
Writeln (‘ Сумма = ‘, sum);
Комментарий. Суммируются элементы строки с номером ***n***, поэтому первый индекс полагается равным ***n***, второй индекс ***k*** изменяется в цикле от 1 до ***m*** с шагом 1</td></tr>
<tr><td>Задача</td><td>Фрагмент программы</td></tr>
<tr><td>Вычисление количества положительных значений элементов в столбце с номером 2 двумерного массива</td><td>kol := 0;
for i := 1 to n do
if b [i, 2] > 0 then kol := kol + 1;
Writeln (‘ Количество = ‘, kol);</td></tr>
<tr><td colspan="2">Комментарий. Поскольку обрабатываются элементы столбца с номером 2, то второй индекс полагается равным 2, первый индекс ***i*** изменяется в цикле от 1 до ***n*** с шагом 1</td></tr>
</table>

Задача	Фрагмент программы
Определение минимального значения в столбце с номером ***m*** (последний столбец): 	*min := b [1, m];* *for i := 2 to n do* *if b [i, m] < min then min := b [i, m];* *Writeln (‘ Минимум = ‘, min);* **Комментарий**. За начальное значение переменной ***min*** принимается значение, находящееся на пересечении столбца с номером ***m*** и строки с номером 1
Операции со всеми элементами двумерного массива	
Задача	Фрагмент программы

Вычисление суммы элементов двумерного массива, значения которых отрицательны	*sum := 0;*
	for i := 1 to n do
	for k := 1 to m do
	if b [i, k] < 0 then sum:= sum + b [i, k];
	Writeln (‘ Сумма = ‘, s);

Задача	Фрагмент программы

Формирование одномерного массива ***a***, состоящего из ***m*** элементов, содержащего суммы значений элементов соответствующих *столбцов* массива ***b*** (***a [1]*** – сумма значений элементов первого столбца, ***a [2]*** – второго столбца и т. д.)	*for k := 1 to m do*
	begin
	sum := 0;
	for i := 1 to n do
	sum := sum + b [i, k];
	a [k] := sum;
	end;
	{вывод на экран массива a}
	Writeln (‘ Массив a ‘);
	for k := 1 to m do Writeln (a [k]);

Комментарий. Массив ***b*** обрабатывается по столбцам, поэтому цикл по номеру столбца – внешний (переменная ***k***), цикл по номеру строки – внутренний (переменная ***i***). При обработке двумерного массива по строкам внешний цикл должен быть образован по номеру строки (первый индекс), внутренний – по номеру столбца (второй индекс)	

? Вопросы и упражнения для самоконтроля

1. Что такое массив? В чем отличие одномерного массива от двумерного?
2. Какой индекс элемента двумерного массива (первый или второй) обозначает номер строки, а какой – номер столбца, на пересечении которых расположен данный элемент?
3. По какому закону изменяются параметры внешнего и внутреннего циклов, образующих структуру «цикл в цикле»?
4. В какой последовательности нужно вводить с клавиатуры значения элементов двумерного массива (по строкам слева направо или по столбцам сверху вниз), если цикл ввода организован таким образом, что внешним является цикл по первому индексу массива?
5. Перечислите типовые операции обработки одномерных массивов.
6. Перечислите типовые операции обработки двумерных массивов.

2.4.4. Организация подпрограмм

В языке Паскаль предусмотрены средства, позволяющие оформлять некоторые выполняемые действия как подпрограмму. ***Подпрограмма*** – именованная часть программы, содержащая описание определённого набора действий. Подпрограмма может быть многократно вызвана из разных частей программы. Это бывает необходимо тогда, когда какие-либо действия неоднократно повторяются в программе.

Описание подпрограммы помещается в раздел описаний основной программы. Вызов подпрограммы осуществляется указанием её имени и, возможно, входных и выходных параметров. При вызове происходит выполнение входящих в подпрограмму операторов с использованием указанных при вызове параметров. После выполнения подпрограммы работа программы продолжается с того оператора, который непосредственно следует за оператором вызова подпрограммы.

Подпрограмма имеет доступ к объектам данных, описанным в основной программе, поэтому, для того чтобы передать в подпрограмму обрабатываемые данные, их достаточно

присвоить, например, переменным, объявленным в вызывающей программе. Эти переменные называются ***глобальными переменными***. Но такой путь часто приводит к ошибкам. Для обеспечения контролируемой передачи данных в подпрограмму и возврата результатов из неё используется механизм параметров. ***Формальные параметры*** указываются при описании подпрограммы (в её заголовке). При вызове подпрограммы формальным параметрам присваиваются конкретные значения. Используемые при этом данные называются ***фактическими параметрами***. Для описания действий, выполняемых в подпрограмме, наряду с формальными параметрами и глобальными переменными могут использоваться переменные, объявленные в подпрограмме. Они называются ***локальными переменными*** и сохраняют свои значения только внутри подпрограммы.

Формальные параметры подпрограммы можно условно разделить на входные и выходные. ***Входные параметры*** используются для передачи данных из вызывающего блока программы в подпрограмму, а ***выходные параметры*** – для возврата из подпрограммы в вызывающий блок результатов работы подпрограммы. В языке Паскаль имеется два вида подпрограмм: функции и процедуры.

Подпрограмма-функция имеет следующую структуру:

function *имя_функции (формальные параметры: тип): тип результата;*
раздел описаний функции
begin
исполняемая часть функции
имя_функции := результат;
end;

Исполняемая часть функции содержит операторы вычисления значения функции. Результат вычислений в подпрограмме-функции возвращается в вызывающий блок посредством имени функции. Поэтому в исполняемой части функции *обязательно должен присутствовать оператор, присваивающий имени функции какое-либо значение*. Подпрограмма-функция всегда возвращает *один результат*.

Пример 2.8. Составить программу определения наибольшего из четырех целых чисел с использованием подпрограммы-функции.

Исходные данные: ***a, b, c, d*** **–** произвольные целые числа.

Требуется определить: ***m*** **–** наибольшее из заданных чисел.

Алгоритм

Определим наибольшее значение, выполнив трижды процедуру определения наибольшего из двух чисел: для ***a*** и ***b*** (результат в ***p***), для ***c*** и ***d*** (результат в ***q***), для ***p*** и ***q*** (результат в ***m***). Для определения наибольшего из двух чисел используем подпрограмму-функцию ***Maxnum***.

Программа

```
program Prim_2_8;
var a, b, c, d, m, p, q : integer;
function Maxnum ( x, y : integer ) : integer; {заголовок функции}
var max : integer ;          {раздел описаний функции}
  begin
    if x > y then max := x else max := y;
    Maxnum := max ;
  end;                       {конец описания функции}
begin
  Write ( ' Введите 4 числа ' );
  Readln ( a, b, c, d );
  p := Maxnum ( a, b );      {определение наибольшего из a и b}
  q := Maxnum ( c, d );      {определение наибольшего из c и d}
  m := Maxnum ( p, q );      {определение наибольшего из p и q}
  Write ( ' Наибольшее число ', m : 6 );
end.
```

Комментарий к программе. Здесь ***x, y*** **–** формальные параметры функции. Переменная ***max*** является локальной переменной подпрограммы. При первом вызове функции ***Maxnum*** вместо формального параметра ***x*** передается значение переменной ***a***, вместо формального параметра ***y*** – значение переменной ***b***. Затем выполняются действия, указанные в исполнительной части функции. Результат возвращается в вызывающую программу и присваивается переменной ***p***. При втором вызове вместо ***x*** передается значение переменной ***c***,

вместо ***y*** – значение переменной ***d***. Результат присваивается переменной ***q***. При третьем вызове вместо ***x*** передается значение переменной ***p***, вместо ***y*** – значение переменной ***q***. Результат присваивается переменной ***m***.

Подпрограмма-процедура имеет следующую структуру:

procedure *имя_процедуры (формальные параметры: тип параметров);*

раздел описаний процедуры

begin

исполняемая часть процедуры

end;

Вызов процедуры производится оператором следующего вида:

имя_процедуры (список фактических параметров);

При вызове в скобках указываются фактические параметры, одни из которых используются как входные, другие – как выходные. Последовательность и типы фактических параметров при вызове должны совпадать с последовательностью и типами формальных параметров в заголовке процедуры при ее объявлении.

Возврат из процедуры после ее выполнения происходит на оператор, следующий после оператора вызова процедуры. Процедура, в отличие от функции, может возвращать любое количество значений выходных параметров (один, несколько, ни одного).

Передача параметров в процедуру может производиться двумя способами – по значению или по ссылке. В первом случае значения фактических параметров копируются в соответствующие формальные параметры. *По значению передаются входные параметры.* При этом фактические параметры могут быть константами, переменными, выражениями. Во втором случае (передача по ссылке) в процедуру передаются не значения переменных, а их адреса. *По ссылке передаются выходные параметры.* При вызове соответствующие им фактические параметры могут быть только переменными. В заголовке процедуры такие параметры указываются после слова *var*.

Пример 2.9. Составить программу определения наибольшего из 4 целых чисел ***a, b, c, d*** с использованием подпрограммы-процедуры.

Программа

```
program Prim_2_9;
var a, b, c, d, m, p, q : integer;
procedure Maxnum ( x, y : integer; var max : integer );  {заголовок процедуры}
  begin
    if x > y then max := x else max := y ;
  end;                                  {конец описания процедуры}
begin
  Write ( ' Введите 4 числа ' ); Readln ( a, b, c, d );
  Maxnum ( a, b, p);                    {вычисление наиб. из a и b, результат – в p }
  Maxnum ( c, d, q );                   {вычисление наиб. из c и d, результат – в q}
  Maxnum ( p, q, m );                   {вычисление наиб. из p и q, результат – в m}
  Write ( ' Наибольшее число ', m : 6 );
end.
```

? Вопросы и упражнения для самоконтроля

1. Что такое подпрограмма? В каких случаях является целесообразным использование подпрограмм?
2. В какой части программы должны быть описаны подпрограммы?
3. Какова структура подпрограммы-функции?
4. Какова структура подпрограммы-процедуры?
5. Каково назначение формальных и фактических параметров подпрограммы?
6. Какие переменные называются глобальными переменными, а какие – локальными?
7. Сколько значений возвращает в вызывающую программу подпрограмма-функция, сколько – подпрограмма-процедура?
8. В чем заключается различие между входными и выходными параметрами подпрограммы?

2.5. Практические работы

Практическая работа 2.1. Работа с программой в среде программирования Turbo Pascal 7.0

Цель работы – научиться:

- выполнять основные операции в среде программирования Turbo Pascal 7.0 по вводу, редактированию, отладке, сохранению и запуску программы на языке программирования Паскаль;
- вводить с клавиатуры исходные данные для работы программы;
- идентифицировать различные этапы работы с программой (ввод текста программы, компиляция, выполнение программы, редактирование текста программы, отладка программы).

Порядок выполнения работы

1. Познакомьтесь с постановкой задачи, алгоритмом и программой для решения **задачи 2.4**.
2. Активируйте среду программирования Turbo Pascal 7.0.
3. Выполните в меню **File – New** для входа в окно встроенного ***текстового редактора системы программирования***.
4. Введите с клавиатуры текст программы, приведенной в описании решения **задачи 2.4** (рис. 2.10, табл. 2.12).

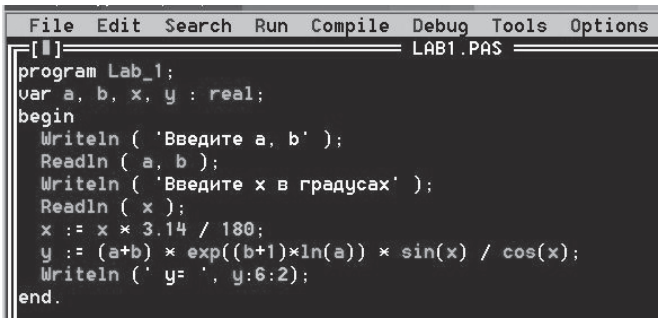


Рис. 2.10. Текст программы в окне редактора Turbo Pascal

При вводе текста для *переключения клавиатуры на кириллицу* одновременно нажимаются клавиши **<Ctrl>** + **<Shift>**

справа, *на латиницу* – **<Ctrl> + <Shift> слева**. Для удаления символов используются клавиши **<Backspase>** (удаляет символ слева от курсора) и **<Delete>** (удаляет символ справа от курсора). Клавиша **<Insert>** используется для перехода из режима вставки символов в режим замены символов при вводе.

Команды меню **Edit** позволяют вырезать, копировать и вставлять текст в окне редактора. Команда **Cut** вырезает выделенный блок текста и помещает его в буфер обмена. Команда **Copy** помещает копию выделенного блока текста в буфер. Команда **Paste** вставляет текст из буфера в позицию, указанную курсором. Команда **Clear** удаляет отмеченный блок текста. Текст, удаленный с помощью этой команды, нельзя восстановить.

5. Завершив ввод текста программы, выполните ***компиляцию программы*** (клавиша **<F9>** на клавиатуре). Если будет обнаружена ***синтаксическая ошибка***, поверх текста программы появится красная строка с сообщением об ошибке. После нажатия клавиши **<Esc>** сообщение исчезает, курсор устанавливается в строку с ошибкой. Внесите исправления в текст программы и вновь выполните компиляцию.

6. После исправления всех ошибок сохраните текст программы на диске, выполнив в меню **File – Save as ...** и введя имя файла, например, **z:\lab1.pas**.

7. Запустите программу на выполнение, выбрав в меню **Run – Run** или одновременно нажав клавиши <**Ctrl**> + <**F9**>. При выполнении программы окно редактирования текста программы (синего цвета) сменяется окном вывода результатов работы программы (черного цвета).

8. В ответ на подсказку о вводе исходных данных введите числовые значения переменных ***a*** и ***b***, разделив их при вводе символом пробел. Завершите ввод нажатием клавиши **<Enter>**. После следующей подсказки введите значение ***x*** в градусах. Фрагмент окна вывода результатов, в котором показан ввод исходных данных и результат работы программы, представлен на рис. 2.11.

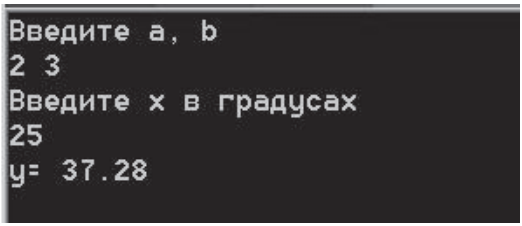


Рис. 2.11. Ввод исходных данных и вывод результатов работы программы

9. После выполнения программы вновь откроется окно редактирования (синего цвета). Чтобы посмотреть результат работы программы, нажмите одновременно клавиши **<Alt>** + **<F5>**.

10. Сравните результат расчетов с правильным значением (рис. 2.11). Если результат совпал, этот этап работы можно считать законченным. Если результат отличается, проверьте запись в программе расчетных формул, найдите несовпадение с текстом программы, приведенным в пособии, исправьте ошибку и вновь выполните компиляцию и запуск программы на выполнение. Сохраните исправленный текст программы на диске (**File – Save**).

11. Завершите работу с программой и с системой программирования, выполнив в меню **File – Exit**.

12. С помощью инженерного калькулятора (**Пуск – Все программы – Стандартные – Калькулятор – Инженерный**) выполните расчет значения функции ***y*** в соответствии с номером вашего варианта (табл. 2.13).

13. Вновь активируйте среду программирования Turbo Pascal и откройте файл с сохраненной программой, выполнив в меню **File – Open**. В строке ввода имени файла укажите имя сохраненного файла (например, **z:\lab1.pas**).

14. Внесите исправления в текст программы, записав вычисление функции в соответствии с номером вашего варианта.

15. Выполните компиляцию программы и расчеты с указанными в задании (табл. 2.13) исходными данными.

16. Сравните полученный результат с результатом расчета на калькуляторе. В случае совпадения этот этап программы

можно считать законченным. В случае несовпадения следует еще раз вернуться к пункту 13 данного задания.

17. Сохраните измененный текст программы на диске. Предъявите результаты работы преподавателю.

18. Завершите работу с системой программирования Turbo Pascal 7.0 (**File – Exit**).

19. Оформите отчет о работе.

Требования к содержанию и оформлению отчета о работе

☑ Отчет выполняется письменно в тетради. В нем должны быть представлены ответы на следующие вопросы.

1. Как выполнить запуск интегрированной среды программирования Turbo Pascal?
2. Какая команда управляющего меню открывает окно редактора для ввода текста программы?
3. Какими клавишами производится переключение регистров клавиатуры с кириллицы на латиницу и обратно?
4. С помощью каких команд меню **Edit** можно выполнить удаление, копирование, перемещение выделенного фрагмента текста программы?
5. Как запустить процедуру компиляции текста программы?
6. Как сохранить текст программы на диске?
7. Как запустить процедуру выполнения программы?
8. Какая комбинация клавиш используется для перехода из окна редактора текстов в окно просмотра результатов работы программы?
9. Как открыть в окне редактирования текст программы, сохраненной ранее на диске?
10. Как завершить работу в среде программирования Turbo Pascal?

Задача 2.4. Составить алгоритм и программу вычисления значения функции $y = (a+b)\cdot a^{b+1}\cdot tgx$ для произвольных $\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{x}$, значения которых вводятся с клавиатуры. Значение $\boldsymbol{x}$ задавать в градусах. Выполнить расчеты при следующих значениях исходных данных: $\boldsymbol{a} = 2$; $\boldsymbol{b} = 3$; $\boldsymbol{x} = 25°$.

Алгоритм и программа решения задачи приведены в табл. 2.12.

Комментарий к программе. Для хранения исходных данных и результатов требуется выделить место в памяти. Это сделано в операторе 1. Для переменных ***a, b, x, y*** выбран вещественный тип *real.* Операторы 2 и 4 представляют собой приглашения к вводу данных. Ввод выполняется операторами 3 и 5 с помощью процедуры *Readln*. Оператор 6 нужен для перевода значения переменной ***x*** из градусов в радианы, так как стандартные функции *sin* и *cos* требуют, чтобы угол был задан в радианах. В операторе 7 вычисляется выражение, записанное справа от операции присваивания, результат присваивается переменной ***y*** . Оператор 8 выводит значение ***y*** с соответствующими пояснениями.

Таблица 2.12

Алгоритм и программа для решения задачи 2.4

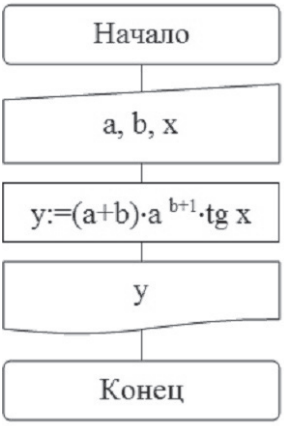
Алгоритм	Программа
	*program Lab_1;* *var a, b, x, y : real;* { 1 } *begin* *Writeln (‘ Введите a, b ‘);* { 2 } *Readln (a, b);* { 3 } *Writeln (‘ Введите x в градусах ‘);* { 4 } *Readln (x);* { 5 } *x := x * 3.14 / 180;* { 6 } *y := (a + b)*exp ((b + 1)*ln (a))** { 7 } *sin (x)/cos(x);* *Writeln (‘ y = ‘, y : 6 : 2);* { 8 } *end.*

Таблица 2.13

Варианты заданий

Номер варианта	Вид функции	Исходные данные для расчетов
1	$y = \left(\frac{1}{\sin x} + a\right) + a^b \cdot \sqrt{b}$	a = 2; b = 3; x = 45°
2	$y = (a + tg\ x) \cdot (b + \ln a) + e^{-b}$	a = 2; b = 1; x = 25°
3	$y = e^{\lvert a\rvert} \cdot \left(a + \frac{1}{\sin x}\right) + \sqrt{a + b}$	a = 1; b = 2; x = 55°
4	$y = \ln b \cdot \left(a + \frac{1}{\sin x}\right) \cdot \frac{1}{e^{b+1}}$	a = 2; b = 3; x = 75°
5	$y = e^{b-1} \cdot \left(b - \frac{1}{\sin^2 x}\right) + \sqrt{a+1}$	a = 5; b = 2; x = 30°
6	$y = tgx \cdot \frac{(a+b)}{\sqrt{b}} + \lvert 2a\rvert$	a = 10; b = 25; x = 60°
7	$y = \left(\frac{1}{\cos x} + \lvert a\rvert\right) \cdot a^b \cdot (a - b)$	a = 2; b = 3; x = 15°
8	$y = \left(\frac{1}{\sin^2 x} + b\right) + e^b \cdot \ln(a + b)$	a = 2; b = 0; x = 45°
9	$y = tgx \cdot a^{(b+1)} + \ln(a + b)$	a = 2; b = 3; x = 45°
10	$y = \left(\frac{a}{\cos x} + b\right) \cdot e^b \cdot \sqrt{(a + b)}$	a = 2; b = 1; x = 25°
11	$y = \left(\frac{1}{\cos 2x} + 5\right) \cdot \ln(a + 1) \cdot \sqrt{(b - 1)}$	a = 1; b = 2; x = 45°
12	$y = (a + 2b) \cdot \sqrt{(b + 2a)} \cdot \frac{1}{\cos x}$	a = 2; b = 1; x = 35°
13	$y = tgx \cdot \sqrt{b} \cdot \frac{\ln(a+b)}{\cos x}$	a = 2; b = 5; x = 30°
14	$y = (a + 1) \cdot (b + 2a) \cdot tgx \cdot a^b$	a = 2; b = 2; x = 55°
15	$y = (\lvert a\rvert + \cos x) \cdot \sqrt{(b^a + 2)} \cdot e^{a+b}$	a = 2; b = 5; x = 30°

Практическая работа 2.2. Организация ветвлений в программе на языке Паскаль

Цель работы – научиться:

- выполнять формализацию задачи, сформулированной в терминах описываемой предметной области, приводя ее к решению типовой задачи вычисления значения функции, заданной несколькими выражениями;
- выбирать тип данных для адекватного представления в программе исходных данных и результатов решаемой задачи;
- организовывать в программе ввод исходных данных и вывод результатов решения задачи;
- использовать оператор присваивания для выполнения вычислений и сохранения результатов вычислений;
- использовать оператор *if* для организации ветвлений в программе;
- формировать тестовые варианты расчетов для проверки правильности работы программы.

Порядок выполнения работы

1. Рассмотрите пример выполнения задания.
2. Выберите из табл. 2.15 задание в соответствии с номером варианта.
3. Определите исходные данные и результаты, которые должны быть получены, обратив внимание на тип данных, который должен быть использован для их представления.
4. Выведите расчетные формулы. Для этого определите и запишите вид функции, к вычислению значения которой сводится решение задачи.
5. Подготовьте тестовые варианты расчетов. Количество таких вариантов должно быть равно количеству областей определения функции. Выполните расчеты с помощью калькулятора для каждого тестового варианта.
6. Составьте блок-схему и программу решения задачи.
7. Выполните ввод и отладку программы в среде программирования Turbo Pascal. Для проверки правильности работы программы используйте подготовленные тестовые варианты расчетов.

8. Предъявите результаты расчетов преподавателю.

9. Оформите отчет о работе в соответствии с заданными требованиями.

Требования к содержанию и оформлению отчета о работе

☑ Отчет выполняется письменно в тетради. Он должен содержать:

- тему работы и условие задачи;
- описание исходных данных и результатов;
- расчетные формулы;
- алгоритм (блок-схема) решения задачи;
- текст программы на языке Паскаль;
- тестовые варианты расчетов и результаты расчетов по программе.

Пример выполнения задания

Задание. Телекомпания принимает заказы от предприятий и организаций на размещение рекламы в телеэфире. Существует базовая цена на показ в эфире рекламного ролика длительностью 1 минута. Стоимость показа зависит от длительности демонстрации ролика в эфире, а также от времени выхода в эфир телепередачи, в рамках которой демонстрируется рекламный ролик. Если время выхода ролика в эфир попадает в интервал от 17 до 24 часов, к базовой цене применяется коэффициент 2. Если время выхода в эфир от 7 до 17 часов, коэффициент равен 1, в ночное время от 24 до 7 утра цена снижается на 40% от базовой. Составить программу определения стоимости одного показа рекламного ролика, если известна длительность показа ролика, базовая цена на показ 1 минуты рекламы и время выхода рекламы в эфир.

Исходные данные:

C – базовая цена 1 мин рекламы в телеэфире (число с дробной частью);

D – длительность показа ролика в минутах (число с дробной частью);

T – время выхода рекламы в эфир (число с дробной частью).

Требуется определить:

ST – стоимость показа рекламного ролика (число с дробной частью).

Расчетные формулы

Стоимость показа рекламного ролика определяется как произведение длительности показа в мин. на цену 1 мин. показа ролика. Цена 1 минуты показа (**C1**) определяется как значение следующей функции от переменной **T**:

$$C1 = \begin{cases} C, \text{ если } 7 \le T \le 17 \\ C*2, \text{ если } 17 < T \le 24 \\ C*0.6, \text{ если } 0 \le T < 7 \\ 0, \text{ если } T < 0 \text{ или } T > 24 \end{cases}.$$

Значение **C1**, равное нулю, означает, что время показа рекламы введено неверно. В этом случае результатом вычислений будет вывод на экран соответствующего сообщения.

Тестовые варианты расчетов

Правильность работы программы должна быть проверена для значений переменной **T** из пяти интервалов, на которых функция **C1** изменяет свой вид. Возможные наборы тестовых данных приведены в табл. 2.14.

Таблица 2.14

Тестовые данные
для проверки правильности работы программы

№ варианта	Исходные данные	Результат
1	C = 1000, D = 2, T = 25	Сообщение «Время введено неверно»
2	C = 1000, D = 2, T = -1	Сообщение «Время введено неверно»
3	C = 1000, D = 2, T = 2.30	1200.00
4	C = 1000, D = 2, T = 8.20	2000.00
5	C = 1000, D = 2, T = 19.10	4000.00

Алгоритм решения задачи

Блок-схема алгоритма решения задачи представлена на рис. 2.12.

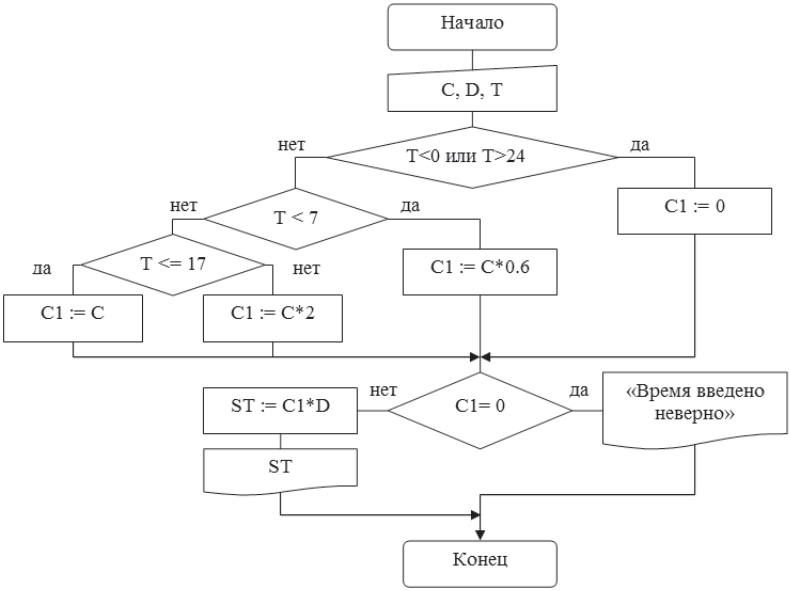


Рис. 2.12. Блок-схема алгоритма решения задачи

☑ Поставленная задача представляет собой задачу вычисления значения функции **C1**, представленной на разных интервалах числовой оси разными выражениями, решение которой рассмотрено в примере 2.2 данного пособия. В нашем случае точками ветвления функции являются: **T = 0**, **T = 7**, **T = 17**, **T = 24**. Начать проверку можно с условия «**T < 0 или T > 24**» (время рекламы введено неверно). В этом случае **C1** (цена 1 мин. показа рекламы) присваивается значение 0.

Если введенное время показа рекламы попало в диапазон от 0 до 24 часов, нужно определить, в каком из ценовых интервалов оно находится, чтобы вычислить соответствующую цену 1 минуты показа рекламы в эфире (**C1**). Если выполнилось условие **T < 7**, это значит, что время показа находится в диапазоне от 0 часов до 7 часов, соответствующее значение **C1** вычисляется по формуле **C1 := C*0.6**. Если условие не выполнилось, необходимо проверить условие

T < 17. Проверка этого условия приводит к окончательному выбору расчетной формулы для **C1**. Далее следует проверить, не оказалось ли **C1** равным нулю, что соответствует неправильному вводу исходных данных. Если **C1** не равно нулю, стоимость демонстрации рекламного ролика определяется умножением цены 1 минуты демонстрации (**C1**) на длительность ролика в минутах (**D**).

Программа

```
program Lab_2;
var c, d, t, c1, st : real ;
begin
  Writeln ( ' Введите базовую цену 1 мин рекламы ' );
  Readln ( c );
  Writeln ( ' Введите длительность рекламы в минутах ' );
  Readln ( d );
  Writeln ( ' Введите время начала демонстрации рекламы ' );
  Readln ( t );
  {выбор формулы для расчета значения c1}
  if (t < 0) or (t > 24) then c1 := 0
  else
    if t < 7 then c1 := c * 0.6
    else
      if t <= 17 then c1 := c
      else c1 := c * 2;
  if c1 = 0 then Writeln ( ' Время введено неверно ' )
  else
    begin
    st := c1 * d; Writeln ( ' Стоимость рекламы = ', st : 8 : 2)
  end;
end.
```

Ввод исходных данных и результат расчета для тестового варианта номер 5 показан на рис. 2.13.

```
Введите базовую цену 1 мин рекламы
1000
Введите длительность рекламы в минутах
2
Введите время начала демонстрации рекламы
19.10
Стоимость рекламы равна = 4000.00
```

Рис. 2.13. Результат работы программы

Таблица 2.15

Варианты заданий

Номер варианта	Условие задачи
1	Оплата за пользование электроэнергией производится по разным тарифам в зависимости от оборудования дома электрическими или газовыми плитами. В первом случае (электрические плиты) стоимость 1 кВт/час составляет 2 руб., во втором – 2,5 руб. Составить программу расчета суммы оплаты за пользование электроэнергией за месяц, если известен расход электроэнергии (количество потребленных в течение месяца кВт/час)
2	Оплата за пользование услугами сети Интернет осуществляется по входящему трафику согласно следующей схеме: от 0 до 100 Мб – фиксированная плата 120 руб., сверх 100 Мб – 2 руб. за каждый Мб входящего трафика. Составить программу расчета оплаты за пользование услугами сети Интернет, если известен объем входящего трафика
3	Кредитное общество выдает денежные кредиты сроком на 1 год. Через год кредит должен быть возвращен вместе с процентами. Величина процентов зависит от суммы кредита: если сумма меньше или равна 100000 руб., начисляется 12%, если больше 100000 руб. – 10%. Составить программу вычисления размера возвращаемой клиентом суммы, если известен размер кредита

Продолжение табл. 2.15

Номер варианта	Условие задачи
4	Кредитное общество выдает клиентам денежные кредиты на различные сроки. Процент, под который выдается кредит, зависит от срока, на который выдается кредит. Если срок меньше или равен 12 месяцам, начисляется 12%, если больше 12 месяцев – 15%. Составить программу вычисления размера возвращаемой клиентом суммы, если известен размер выданного кредита и срок, на который предоставляется кредит
5	Фирма предоставляет автомобили напрокат. Стоимость проката в течение суток определяется как произведение оценочной стоимости автомобиля на некоторый коэффициент. Величина коэффициента зависит от длительности проката: если длительность меньше или равна 3 суткам, коэффициент равен 0,001, если больше 3 суток – 0,0005. Составить программу вычисления размера оплаты за прокат автомобиля, если известны оценочная стоимость автомобиля и длительность проката, измеренная в сутках
6	Магазин реализует товары, предоставляя покупателям скидки в зависимости от суммы, на которую приобретен товар. Если сумма покупки составляет меньше 10000 руб., скидка не предоставляется. При покупке на сумму от 10000 руб. до 30000 руб. скидка составляет 5%. При покупке на сумму свыше 30000 руб. размер скидки – 10% от суммы покупки. Составить программу вычисления оплачиваемой покупателем суммы, если известна сумма покупки без скидки
7	Размер оплаты за посещение ребенком детского сада зависит от среднего месячного дохода на 1 человека в семье, который определяется делением совокупного дохода семьи на количество человек в семье. Если при этом доход на 1 человека меньше 5000 руб. в месяц, предоставляется скидка, равная 50% от базовой оплаты. Если доход на 1 человека больше либо равен 5000 руб. и меньше либо равен 7000 руб., предоставляется скидка 30%, если доход больше 7000 руб., скидка не предоставляется. Составить программу определения оплаты при заданных значениях совокупного месячного дохода семьи, количественного состава семьи и базовой ставки оплаты за пользование детским садом

Продолжение табл. 2.15

Номер варианта	Условие задачи
8	Стоимость туристической путевки определяется в зависимости от времени года. Базовая стоимость путевки умножается на сезонный коэффициент, который определяется следующим образом: октябрь, май, январь – коэффициент 1; июнь, июль август, сентябрь – коэффициент 1,2; ноябрь, декабрь, февраль, март, апрель – коэффициент 0,8. Составить программу определения стоимости путевки, если известны ее базовая стоимость и номер месяца, на который приобретается путевка
9	Транспортное предприятие доставляет заказчикам грузы. Стоимость услуги определяется дальностью поездки: если дальность меньше 10 км, то тариф составляет 20 руб./км, от 10 до 100 км – 15 руб./км, свыше 100 км – 10 руб./км. Составить программу расчета стоимости доставки груза, если известна дальность поездки
10	Транспортное предприятие перевозит грузы из пункта А в пункт В, расстояние между которыми составляет 100 км. Стоимость доставки определяется весом перевозимого груза. Базовый тариф равен 20 руб./км и применяется для грузов весом от 500 до 1000 кг. Если вес груза меньше 500 кг, базовый тариф умножается на коэффициент 0,8, если груз больше 1000 кг, применяется коэффициент 1,5. Составить программу определения стоимости доставки груза, если известен его вес
11	Страховая компания занимается страхованием домашнего имущества. При заключении договора клиент оплачивает страховой взнос, размер которого определяется как произведение объявленной стоимости страхуемого имущества (страховая сумма) на тариф. Величина тарифа определяется величиной страховой суммы: если страховая сумма меньше либо равна 100000 руб., тариф равен 0,01, если страховая сумма больше 100000 руб., тариф равен 0,02. Составить программу определения размера страхового взноса, если известна страховая сумма

Окончание табл. 2.15

Номер варианта	Условие задачи
12	Медицинская клиника проводит платный прием пациентов. За каждый прием пациент платит фиксированную сумму, определенный процент от которой составляет заработная плата врача, осуществляющего прием. Указанный процент зависит от категории врача: для врача высшей категории – 40%, первой категории – 30%, второй категории – 20%. Составить программу расчета размера отчисления на заработную плату врача, если известны стоимость приема и категория врача
13	Авиакомпания предоставляет скидки на билеты в определенные периоды года (январь, февраль, июнь, июль, август). Если полет выполняется в январе или феврале, скидка составляет 40% от базового тарифа, в летнее время скидка равна 20%. Составить программу определения стоимости билета, если известны базовый тариф и время выполнения полета (номер месяца)
14	Сельхозпредприятие реализует продукцию оптовым покупателям. Существует базовая цена реализации на каждый вид продукции. В зависимости от количества реализуемой продукции покупателям могут предоставляться скидки: при весе продукции от 100 до 300 кг предоставляется скидка 5%, если вес больше 300 кг, скидка составляет 10% от базовой цены. Составить программу определения стоимости реализуемой продукции, если известны базовая цена и количество продукции
15	Фирма сдает в аренду клиентам офисные помещения. Существует базовая цена 1 кв. м сдаваемых помещений. Если площадь арендуемых клиентом помещений от 20 до 100 кв. м, применяется базовая цена. Если общая площадь меньше 20 кв. м, базовая цена увеличивается на 10%, если же площадь больше 100 кв. м, базовая цена уменьшается на 10%. Составить программу вычисления стоимости аренды офисных помещений, если известны базовая цена 1 кв. м и общая площадь арендуемых клиентом помещений

Практическая работа 2.3. Использование циклов для обработки одномерных массивов

Цель работы – научиться:

- выполнять формализацию задачи, сформулированной в терминах описываемой предметной области, приводя ее к типовой задаче обработки одномерного массива;
- использовать оператор цикла *for* для организации ввода и вывода элементов одномерного массива;
- организовывать процедуры стандартной обработки одномерных массивов с использованием циклов.

Порядок выполнения работы

1. Выберите из табл. 2.16 задание в соответствии с номером варианта.
2. Рассмотрите пример выполнения задания.
3. Укажите исходные данные и результаты, которые должны быть получены.
4. Выполните формализацию задачи, сформулировав ее в терминах требований по обработке одномерного массива.
5. Подготовьте тестовый вариант расчетов. Выполните расчеты с помощью калькулятора для тестового варианта.
6. Составьте алгоритм и программу решения задачи.
7. Выполните ввод и отладку программы в среде программирования Turbo Pascal. Для проверки правильности работы программы используйте подготовленный тестовый вариант расчетов.
8. Предъявите результаты расчетов преподавателю.
9. Оформите отчет о работе в соответствии с заданными требованиями.

Требования к содержанию и оформлению отчета о работе

☑ Отчет выполняется письменно в тетради. Он должен содержать:

- тему работы и условие задачи; описание исходных данных и результатов;
- формализацию задачи;
- текст программы на языке Паскаль;
- тестовый вариант расчетов и результаты расчета по программе.

Пример выполнения задания

Задание. Известны значения объема выпуска продукции предприятием в денежном выражении за **N** месяцев. Определить суммарный выпуск продукции за рассматриваемый период, а также среднее и наименьшее значения объема выпуска продукции. Вывести номера месяцев, для которых объем выпуска продукции оказался в диапазоне от 300000 до 500000 руб.

Исходные данные:

N – количество месяцев (целое число);

A [i], i = 1, 2, ..., N – нумерованная последовательность значений выпуска продукции в денежном выражении (массив значений вещественного типа).

Требуется определить:

SUM – суммарный объем выпуска продукции в денежном выражении за рассматриваемый период (число с дробной частью);

SRED – среднее значение ежемесячного выпуска продукции (число с дробной частью);

MIN – наименьшее значение ежемесячного выпуска продукции (число с дробной частью).

Формализация задачи

Значения выпуска продукции предприятием представляют собой конечный набор (**N** значений) однотипных данных. Если перенумеровать эти значения, то для их хранения можно использовать одномерный массив **A**, состоящий из **N** элементов: **A [1]** – объем выпуска продукции за первый месяц рассматриваемого периода, **A [2]** – за второй месяц, ..., **A [N]** – за месяц с номером **N**.

Абстрагируясь от физического смысла значений элементов массива, сформулируем задачу обработки абстрактного одномерного массива **A**.

Задача. Для заданного одномерного массива **A**, состоящего из **N** элементов, выполнить следующее:

1) найти сумму значений элементов массива (**SUM**);
2) найти среднее арифметическое значение элементов массива (**SRED**);
3) найти наименьшее из значений элементов массива (**MIN**);

4) вывести на экран номера элементов массива, значения которых находятся в диапазоне от 300000 до 500000.

Тестовый вариант расчетов

Исходные данные	**Результаты**
N = 4	SUM = 1300000;
Массив A:	SRED = 325000;
A[1] = 250000; A[2] =300000;	MIN = 250000;
A[3] = 350000; A[4] = 400000.	Номера месяцев: 2, 3, 4.

Алгоритм решения задачи

Алгоритм решения задачи представлен блок-схемой на рис. 2.14. Ввод исходных данных и результат расчета для тестового варианта показан на рис. 2.15.

Программа

```
program Lab_3;
var
a : array [ 1.. 100 ] of real ; {объявляется одномерный массив}
i, n : byte ; {n – количество элементов в массиве, i – текущий номер элемента в массиве}
sum, sred, min : real ;
begin
  Writeln ( ' Введите количество месяцев 0<n<=100 ' );
  Readln ( n );
  {ввод значений выпуска продукции по месяцам}
 for i := 1 to n do
    begin
      Writeln( ' Введите объем выпуска продукции за месяц с номером ', i );
       Readln ( a [ i ] );
    end;
{вычисление суммарного и среднемесячного выпуска продукции}
sum := 0;
for i :=1 to n do
    sum := sum + a [ i ];
```

```
sred := sum / n;
Writeln ( ' Суммарный выпуск продукции = ', sum );
Writeln ( ' Среднемесячный выпуск продукции = ', sred );
{вычисление наименьшего выпуска продукции}
min := a [ 1 ];
for i := 2 to n do
    if min > a [ i ] then min := a [ i ];
  Writeln ( ' Наименьший выпуск продукции = ', min );
{вывод на экран номеров месяцев с заданным объемом выпуска }
 for i := 1 to n do
 if (a [ i ]>= 300000 ) and (a [ i ]<= 500000) then Writeln ('Номер ', i);
end.
```

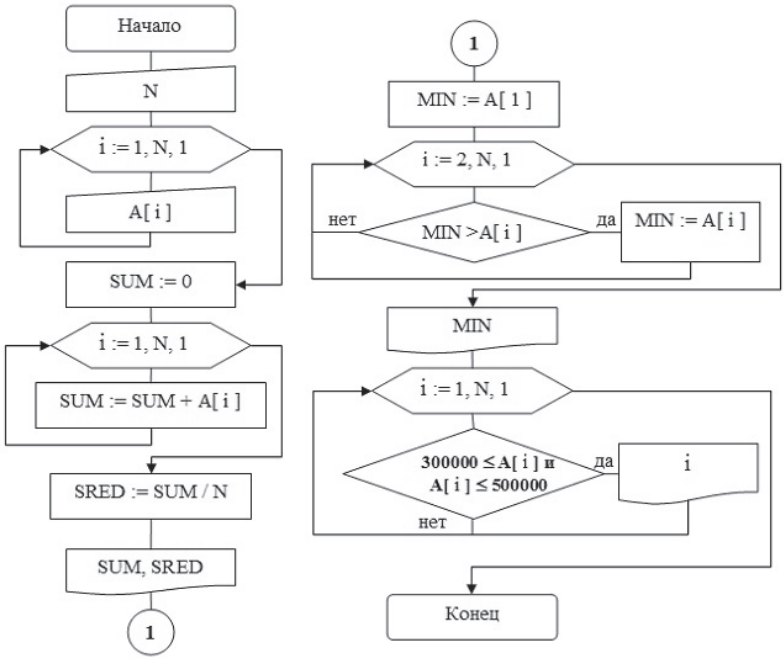


Рис. 2.14. Блок-схема алгоритма решения задачи

```
Введите количество месяцев 0<n<=100
4
Введите объем выпуска продукции за месяц с номером 1
250000
Введите объем выпуска продукции за месяц с номером 2
300000
Введите объем выпуска продукции за месяц с номером 3
350000
Введите объем выпуска продукции за месяц с номером 4
400000
 Суммарный выпуск продукции равен 1300000.00
 Среднемесячный выпуск продукции равен 325000.00
 Наименьший выпуск продукции равен 250000.00
 Номер = 2
 Номер = 3
 Номер = 4
```

Рис. 2.15. Результат работы программы

Таблица 2.16

Варианты заданий

Номер варианта	Условие задачи
1	Заданы цены за одну и ту же услугу в N различных фирмах города. Определить среднее значение цены по всем фирмам и количество фирм, в которых цена за услугу не превосходит 100 руб.
2	Известны размеры ежемесячного дохода по итогам работы магазина за N месяцев года. Определить суммарное, среднее и наибольшее значения дохода за рассматриваемый период
3	Известны размеры экономии фонда заработной платы на предприятии за каждый из N месяцев года. Определить наибольшее и наименьшее значения месячной экономии фонда заработной платы. Вывести на экран номера месяцев, когда экономия фонда заработной платы была минимальной
4	Известно количество ежемесячно обслуживаемых фирмой клиентов за каждый из N месяцев года. Определить среднее и наибольшее значения ежемесячно обслуживаемых фирмой клиентов

Продолжение табл. 2.16

Номер варианта	Условие задачи
5	Заданы цены за одну и ту же услугу в N различных фирмах города. Определить наибольшее и наименьшее значения цены. Вывести на экран номера фирм, в которых зафиксирована наименьшая цена
6	Известны размеры экономии фонда заработной платы на предприятии за каждый из N месяцев года. Определить количество месяцев, для которых размер экономии фонда заработной платы оказался меньше 100000 руб. Вывести на экран номера этих месяцев
7	Заданы цены за одну и ту же услугу в N различных фирмах города. Определить среднее значение цены по всем фирмам и вывести на экран номера фирм, в которых цена не превосходит среднего значения
8	Известны размеры экономии фонда заработной платы на предприятии за каждый из N месяцев года. Определить среднее значение размера экономии фонда заработной платы за рассматриваемый период. Найти количество месяцев, для которых экономия фонда заработной платы оказалась выше среднего значения
9	Заданы цены за одну и ту же услугу в N различных фирмах города. Определить количество фирм, в которых цена на услугу находится в диапазоне от 100 до 120 руб., вывести на экран номера этих фирм
10	Известны размеры ежемесячного дохода по итогам работы магазина за N месяцев года. Определить среднее значение дохода за рассматриваемый период. Вывести на экран номера месяцев, когда доход оказался ниже среднего
11	Известны размеры экономии фонда заработной платы на предприятии за каждый из N месяцев года. Вывести на экран номера месяцев, когда размер экономии фонда заработной платы оказался в диапазоне от 100000 до 200000 руб. Определить количество таких месяцев за рассматриваемый период

Окончание табл. 2.16

Номер варианта	Условие задачи
12	Заданы цены за одну и ту же услугу в N различных фирмах города. Определить среднее значение цены по всем фирмам, а также количество фирм, в которых цена меньше или равна среднему значению
13	Известны размеры ежемесячного дохода по итогам работы магазина за N месяцев года. Определить наименьшее значение ежемесячного дохода за рассматриваемый период. Вывести на экран номера месяцев, когда был получен наименьший доход
14	Известны размеры экономии фонда заработной платы на предприятии за каждый из N месяцев года. Определить среднее значение месячной экономии фонда заработной платы за рассматриваемый период. Вывести на экран номера месяцев, когда размер экономии фонда заработной платы оказался выше среднего значения
15	Известно количество ежемесячно обслуживаемых фирмой клиентов за каждый из N месяцев года. Определить количество месяцев, когда количество клиентов оказалось меньше 100. Вывести на экран номера этих месяцев

Практическая работа 2.4. Обработка двумерных массивов

Цель работы – научиться:

- выполнять формализацию задачи, сформулированной в терминах описываемой предметной области, приводя ее к типовой задаче обработки двумерного массива;
- объявлять в программе двумерный массив;
- использовать оператор цикла *for* для организации циклов структуры «цикл в цикле»;
- организовывать процедуры типовой обработки двумерных массивов с использованием циклов.

Порядок выполнения работы

1. Выберите из табл. 2.17 задание в соответствии с номером варианта.

2. Рассмотрите пример выполнения задания.

3. Укажите исходные данные и результаты, которые должны быть получены.

4. Выполните формализацию задачи, сформулировав ее в терминах требований по обработке двумерного массива.

5. Подготовьте тестовый вариант расчетов. Выполните расчеты с помощью калькулятора для тестового варианта.

6. Составьте алгоритм и программу решения задачи.

7. Выполните ввод и отладку программы в среде программирования Turbo Pascal. Для проверки правильности работы программы используйте подготовленный тестовый вариант расчетов.

8. Предъявите результаты расчетов преподавателю.

9. Оформите отчет о работе в соответствии с заданными требованиями.

Требования к содержанию и оформлению отчета о работе

☑ Отчет выполняется письменно в тетради. Он должен содержать тему работы и условие задачи, описание исходных данных и результатов, формализацию задачи, текст программы на языке Паскаль, тестовый вариант расчетов, результаты расчета по программе.

Пример выполнения задания

Задание. В двумерном массиве **B**, состоящем из **N** строк и **M** столбцов, находятся размеры начисленной заработной платы **N** работников предприятия за **M** месяцев, т. е. **B [I, K]** – это заработная плата работника предприятия с номером **I** за месяц с номером **K**.

Составить программу, позволяющую:

- ввести с клавиатуры количество работников и количество месяцев, а также размеры заработной платы, для ввода исходных размеров заработной платы использовать подпрограмму-процедуру;
- определить наименьшую заработную плату за месяц с номером 1;

Исходные данные:

N – количество работников предприятия (целое число);

M – количество месяцев (целое число);

B [I, K] (I = 1, 2, 3, …, N; K = 1, 2, 3, …, M) – массив значений заработной платы (значения вещественного типа).

Требуется определить:

MIN – наименьшую заработную плату за месяц с номером 1 (число с дробной частью).

Формализация задачи

Значения заработной платы представляют собой конечный набор (**N** × **M** значений) однотипных данных. Для их хранения можно использовать двумерный массив **B**, состоящий из **N** строк и **M** столбцов (**B [1, 1]** – заработная плата работника номер 1 за первый месяц, **B [1, 2] –** заработная плата работника номер 1 за второй месяц, **B [2, 1]** – заработная плата работника номер 2 за первый месяц и т. д.

Абстрагируясь от физического смысла значений элементов массива, сформулируем задачу обработки абстрактного двумерного массива **B**.

Задача. Для заданного двумерного массива **B**, состоящего из **N** строк и **M** столбцов, выполнить:

- ввод с клавиатуры количества строк и количества столбцов массива, а также значений элементов массива;
- определение наименьшего значения среди элементов столбца с номером 1 (**MIN**).

Тестовый вариант расчетов

Исходные данные:

N = 4; M = 3. Массив B:

5000	7000	11000
8000	12000	13000
11000	4000	12000
4000	5000	5000

Результаты:

MIN = 4000

Алгоритм решения задачи

Определение наименьшего значения элементов первого столбца (**MIN**) выполняется попарным сравнением элементов, при этом второй индекс элементов массива будет неизменным и равным 1. Блок-схема алгоритма решения задачи показана на рис. 2.16. Ввод исходных данных и результат расчета для тестового варианта показаны на рис. 2.17.

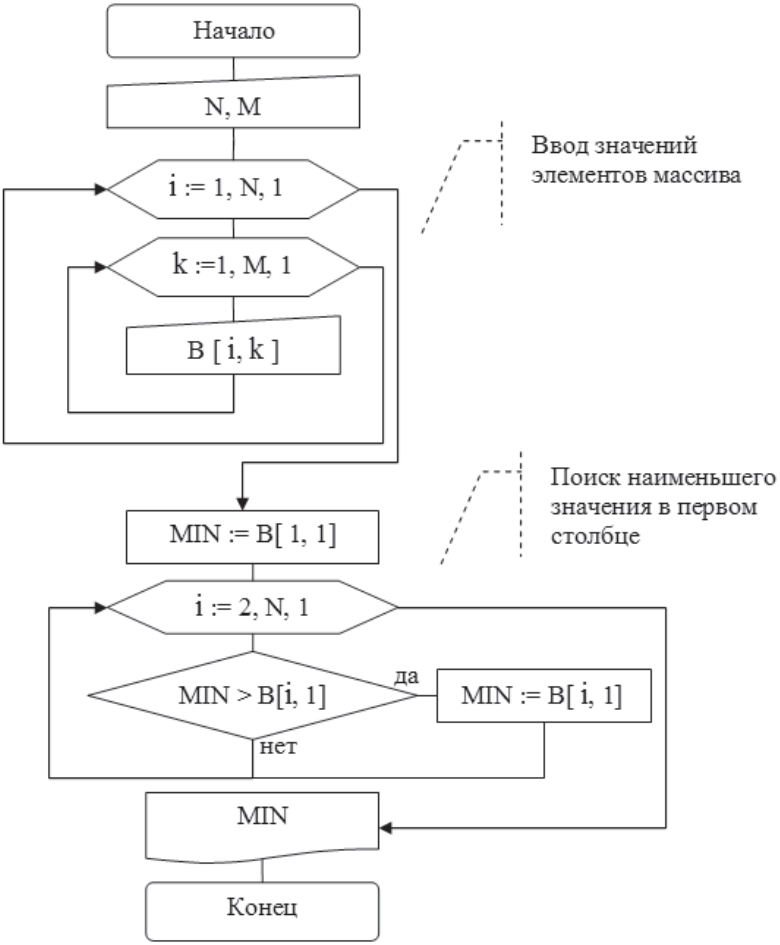


Рис. 2.16. Блок-схема алгоритма решения задачи

Программа

```
program Lab_4;
var n, m, i, k : byte;
min : real;
b : array [ 1.. 10, 1.. 10 ] of real;
begin
  Writeln ( ' Введите количество строк и столбцов в массиве B ' );
```

```
    Readln ( n, m );
    Writeln ( ' Вводите значения зарплат из массива B построчно ' );
    for i := 1 to n do
    for k := 1 to m do
      Readln ( b[ i, k ] );
    {поиск минимального значения в первом столбце массива B}
    min := b [ 1, 1 ];
    for i := 1 to n do
      if min > b [ i, 1 ] then min := b [ i, 1 ];
    Writeln ( ' Минимальная зарплата за первый месяц= ', min : 8 : 2 );
  end.
```

```
Введите количество строк и столбцов в массиве B
4 3
Вводите значения зарплат из массива B построчно. Ввод каждого значения
е нажатием Enter
5000
7000
11000
8000
12000
13000
11000
4000
12000
4000
5000
5000
Минимальная зарплата за первый месяц = 4000.00
```

Рис. 2.17. Результат работы программы

Задание. В двумерном массиве **B**, состоящем из **N** строк и **M** столбцов, находятся размеры начисленной заработной платы **N** работников предприятия за **M** месяцев, т. е. **B [I, K]** – это заработная плата работника предприятия с номером **I** за месяц с номером **K**.

Составить программу, позволяющую выполнить следующее:

- ввести с клавиатуры количество работников и количество месяцев, а также размеры заработной платы;

• выполнить задание в соответствии с номером варианта;
• провести расчёты для **N = 4, M = 3** и следующих значений заработной платы:

	1	2	3
1	5000	7000	11000
2	8000	12000	13000
3	11000	4000	12000
4	4000	5000	5000

⇐ номер месяца

⇑
номер работника

Таблица 2.17

Варианты заданий

Номер варианта	Задание
1	Определить наибольшее значение заработной платы работника с номером 2
2	Определить среднюю заработную плату работника с номером N
3	Определить наибольшее значение заработной платы за месяц с номером 1
4	Вывести на экран номера работников, заработная плата которых за месяц с номером 1 составила больше 6000 руб.
5	Вывести на экран номера месяцев, заработная плата за которые у работника с номером 1 составила меньше10000 руб.
6	Определить суммарную заработную плату за месяц с номером 1
7	Определить суммарную заработную плату за месяц с номером M
8	Определить количество месяцев, в каждом из которых заработная плата работника с номером 1 составила больше 6000 руб.
9	Определить наименьшее значение заработной платы работника с номером 1

Продолжение табл. 2.17

Номер варианта	Задание
10	Определить количество работников, заработная плата которых за месяц с номером 1 составила больше 6000 руб.
11	Определить среднее значение заработной платы за весь период среди всех работников
12	Определить наименьшее значение заработной платы работника с номером N
13	Определить наибольшее значение заработной платы за месяц с номером M
14	Определить наибольшее значение заработной платы работника с номером 1
15	Определить суммарную заработную плату за месяц с номером 1

2.6. Тестовые задания

1. Выберите правильную последовательность указанных этапов решения задачи с использованием компьютера:

1) формализация (математическое моделирование);
2) отладка и тестирование программы;
3) анализ полученных результатов;
4) построение алгоритма;
5) постановка задачи;
6) проведение расчетов;
7) составление программы на языке программирования.

Варианты ответов:

a) 5 – 4 – 1 – 7 – 6 – 2 – 3;
b) 5 – 1 – 4 – 7 – 2 – 6 – 3;
c) 4 – 3 – 1 – 2 – 5 – 7 – 6;
d) 1 – 2 – 3 – 5 – 7 – 6 – 4.

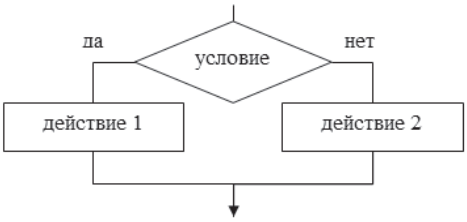
2. Свойство алгоритма, заключающееся в том, что алгоритм должен быть применим для некоторого класса задач, различающихся лишь исходными данными, носит название...

a) детерминированность (определенность);
b) конечность;
c) массовость;
d) результативность.

3. Свойство алгоритма, заключающееся в том, что алгоритм должен представлять процесс решения задачи как последовательное выполнение простых шагов (этапов), носит название...

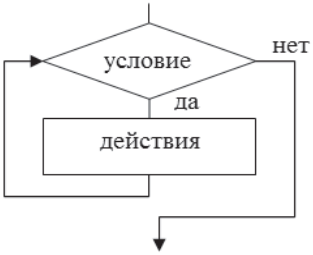
a) детерминированность (определенность);
b) конечность;
c) массовость;
d) дискретность.

4. Показанная ниже блок-схема представляет алгоритмическую структуру...



a) цикл с предусловием;
b) цикл с параметром;
c) ветвление;
d) цикл с постусловием.

5. Показанная ниже блок-схема представляет алгоритмическую структуру…



a) цикл с параметром;
b) цикл с предусловием;
c) ветвление;
d) цикл с постусловием.

6. Дан фрагмент алгоритма, записанный с помощью псевдокода (*нц* – начало цикла с предусловием, *кц* – конец цикла):

c := 0; a := 27; b := 4; d := a;
нц пока d >= b
*c := c + 1; d := d – 2 * b;*
кц

После выполнения алгоритма значения переменных *c* и *d* будут равны...

a) c = 3, d = 2;
b) c = 2, d = 3;
c) c = 3, d = 3;
d) c = 2, d = 2.

7. Дан фрагмент алгоритма обработки одномерного массива *x[1.. N],* записанный на псевдокоде (*нц* – начало цикла с параметром, *кц* – конец цикла):

s:=0;
нц для k от 1 до N
если x [k] < 0 то s := s + x [k]
кц

Данный алгоритм определяет вычисление...

a) минимального элемента массива;
b) суммы отрицательных элементов массива;
c) количества отрицательных элементов массива;
d) максимального элемента массива.

8. К машинно-ориентированным языкам программирования относится язык...

a) АССЕМБЛЕР;
b) FORTRAN;
c) BASIC;
d) PASCAL.

9. Процесс преобразования программы с языка программирования высокого уровня в форму внутреннего представления программы в памяти компьютера называется...

a) отладкой программы;
b) редактированием;
c) трассировкой программы;
d) трансляцией программы.

10. К декларативным языкам программирования относятся языки...

a) BASIC и PASCAL;
b) HTML и XML;

c) LISP и PROLOG;
d) FORTRAN и DELPHI.

11. В объектно-ориентированном программировании объединение в одном объекте данных с процедурами и функциями, которые применяются к этим данным, называется...

a) инкапсуляцией;
b) наследованием;
c) полиморфизмом;
d) группировкой.

12. Тип данных *integer* в языке программирования Паскаль определяет...

a) целочисленные данные со значениями в диапазоне от 0 до 255;
b) целочисленные данные со значениями в диапазоне от -32768 до 32767;
c) логические значения;
d) символы, определяемые множеством значений кодовой таблицы компьютера.

13. Для переменных ***x*** и ***y*** укажите правильный результат выполнения фрагмента программы:

```
x := – 5; y := 4;
if abs ( x ) > abs ( y ) then x := x * 5; write ( x, y );
```

Варианты ответов:

a) -5, 4;
b) 4, 0;
c) -25, 4;
d) -5, 0.

14. Укажите значение логического выражения *(x > 2) and (y < 0)* при x = 10, y = 5:

a) 5;
b) 10;
c) true;
d) false.

15. При выполнении фрагмента программы

```
for k := 5 downto 1 do
  begin
    y := k * k; writeln( y );
  end;
```

на экран будут выведены числа...

a) 1, 2, 3, 4, 5;
b) 25, 16, 9, 4, 1;
c) 1, 4, 9, 16, 25;
d) 5, 4, 3, 2, 1.

16. При выполнении фрагмента программы

```
k := 5;
 while k > 0 do
   begin
    y := k * k; writeln( y ); k := k – 2;
   end;
```

на экран будут выведены числа...

a) 25, 9, 1;
b) 25, 16, 9, 4, 1;
c) 5, 3, 1;
d) 5, 4, 3, 2, 1.

3. ТЕХНИЧЕСКИЕ И ПРОГРАММНЫЕ СРЕДСТВА РЕАЛИЗАЦИИ ИНФОРМАЦИОННЫХ ПРОЦЕССОВ

Изучив материалы этого раздела, вы **узнаете**:

- этапы развития компьютерной техники;
- структуру программного обеспечения компьютеров;
- базовые технологические операции подготовки текстовых документов на компьютере;
- порядок работы с электронными таблицами в среде табличного процессора;
- способы подготовки и показа электронной презентации.

Выполнив практические работы, вы **научитесь**:

- создавать, редактировать, форматировать текстовые документы в среде текстового процессора Microsoft Word;
- включать в текстовый документ графические объекты и формулы;
- создавать, редактировать, форматировать электронные таблицы в среде табличного процессора Microsoft Excel;
- иллюстрировать числовые данные с помощью диаграмм.

Методические рекомендации

Схема работы с учебными материалами данного раздела представлена на рис. 3.1. Теоретическая часть предполагает изучение четырех тем. Практическая часть предполагает выполнение квалификационных заданий. Результаты выполнения определяют индивидуальную траекторию обучения каждого студента в рамках данного раздела. Квалификационные задания приводятся в прил. 4 данного пособия. В результате каждый студент должен выполнить четыре практические работы из шести, представленных в этом разделе. Изучение раздела заканчивается выполнением тестовых заданий, ключ к которым можно найти в прил. 2. Дополнительные учебные материалы по данному разделу можно найти в литературных источниках [1], [2], [4], [8], [11] и интернет-ресурсах [12], [13], [15], [16], [18], список которых представлен в конце пособия.

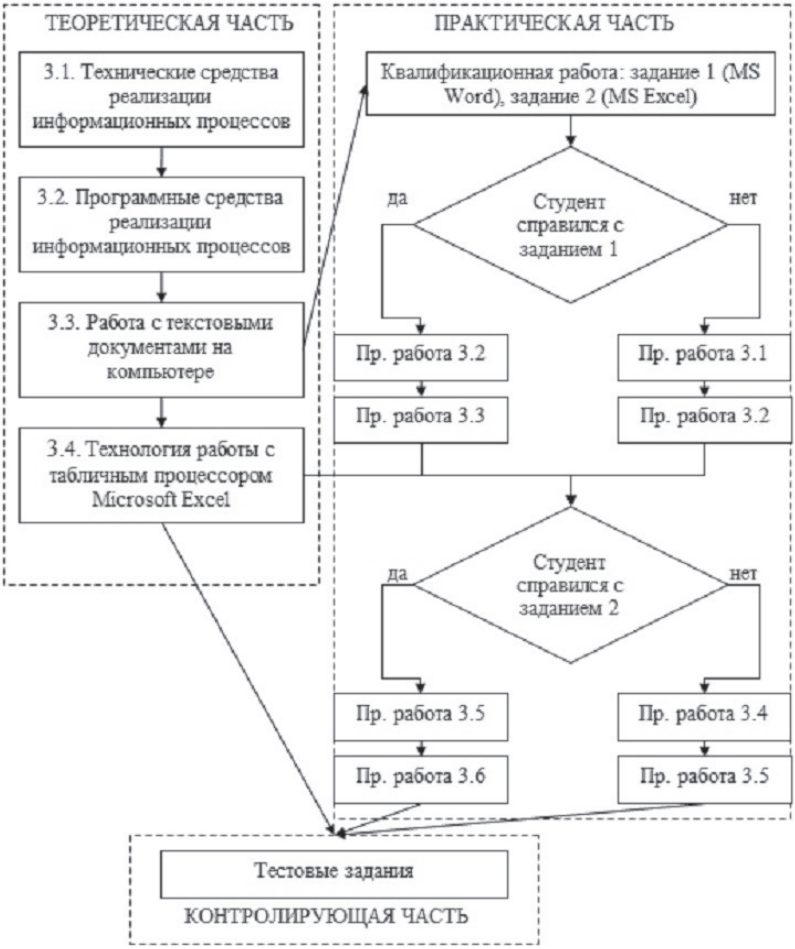


Рис. 3.1. Схема работы с учебными материалами раздела 3

3.1. Технические средства реализации информационных процессов

История развития компьютерной техники

Началом развития компьютерной техники принято считать середину двадцатого века, когда появились первые электронные вычислительные машины (ЭВМ). Тогда же были заложены теоретические основы информатики.

К компьютерам первого поколения относят появившиеся в 50-х годах прошлого века ЭВМ, в которых основой элементной базы являлись электронные лампы. Со ***вторым поколением компьютеров*** связывают две значимые вехи в развитии информатики – появление первых операционных систем и разработку первых языков программирования. Это были компьютеры с элементной базой на полупроводниках и долговременными запоминающими устройствами на магнитных лентах.

Если компьютеры первого и второго поколений были доступны только профессионалам-программистам, то компьютеры ***третьего поколения*** стали доступны широкому кругу пользователей. В середине 60-х годов прошлого века были выпущены компьютеры серии IBM-360, на которых вместо разрозненных транзисторов были использованы малые интегральные схемы. Появились магнитные диски. ***Четвертое поколение*** компьютеров создавалось на базе серийных микропроцессоров. Появление компьютеров этого поколения связывают с разработкой в начале 70-х годов микропроцессора на базе больших интегральных схем (БИС). В настоящее время наибольшее распространение получили компьютеры, предназначенные для индивидуальной работы – так называемые ***персональные компьютеры***.

Основные принципы построения компьютеров

☑ Основные принципы построения компьютеров были сформулированы американским учёным Джоном фон Нейманом в 40-х годах XX века. Назовем эти принципы.

1. Информация, с которой работает компьютер, делится на два типа: набор команд по обработке данных (программы) и данные, подлежащие обработке.

2. И команды, и данные вводятся в память – ***принцип хранимой программы***.

3. Все операции по обработке данных компьютер выполняет под управлением программы – ***принцип программного управления***.

4. Любой компьютер образуют три основных компонента: центральный процессор, память и устройства ввода-вывода (УВВ) (рис. 3.2). Обработку информации по

заданной программе выполняет центральный процессор, включающий ***устройство управления*** и ***арифметико-логическое устройство***. Устройство управления выбирает команды программы из памяти и организует их выполнение, а арифметико-логическое устройство производит арифметические и логические операции над данными. С процессором и памятью взаимодействуют устройства ввода-вывода.

5. Основным источником информации для компьютера является наличие или отсутствие электрического сигнала. При помощи комбинаций из двух состояний (0 – есть сигнал, 1 – нет сигнала или наоборот) компьютер распознает всю введенную в него информацию. При этом для кодирования информации ***используется двоичная система счисления***.

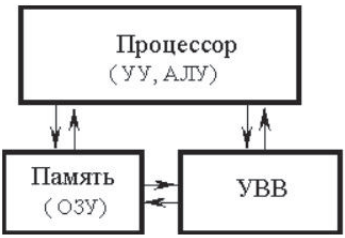


Рис. 3.2. Основные компоненты компьютера

Устройство персональных компьютеров

Обязательный аппаратный состав персонального компьютера (ПК) включает системный блок, дисплей (монитор), клавиатуру. **Базовый комплект** обычно включает еще манипулятор «мышь» и принтер.

Системный блок является центральной частью ПК. Внутри корпуса системного блока размещены электронные схемы, смонтированные на нескольких печатных платах. Кроме того, в системном блоке находится ***блок питания***, преобразующий поступающий из сети переменный ток напряжением 220 В в постоянный ток низкого напряжения, а также вентилятор, жесткий магнитный диск, устройства для чтения/записи CD (DVD) дисков.

Основные электронные схемы в системном блоке размещены на ***системной плате***. Здесь находятся микропроцессор, микросхемы оперативной и постоянной памяти. ***Микропро-***

цессор – это интегральная микросхема, на которой целиком помещается центральный процессор. Быстродействие ПК зависит от характеристик микропроцессора. Скорость работы микропроцессора зависит от его разрядности и тактовой частоты. ***Тактовая частота*** – это частота периодического электрического сигнала, вырабатываемого специальной микросхемой, называемой ***генератором тактовых импульсов***. Указанный периодический сигнал используется для синхронизации работы всех устройств компьютера. Тактовая частота указывает, сколько элементарных операций (тактов) микропроцессор выполняет в одну секунду. Тактовая частота измеряется в ***мегагерцах*** (миллионах тактов в секунду). Разные модели микропроцессоров выполняют одни и те же операции (например, сложение или умножение) за разное число тактов. Чем совершеннее модель микропроцессора, тем меньше тактов требуется для выполнения одних и тех же операций.

Информация внутри компьютера передается не сплошным потоком, а порциями – машинными словами. ***Машинным словом*** называется передаваемая за один такт работы компьютера группа двоичных кодов. ***Разрядность микропроцессора*** – это длина используемых им машинных слов. 8-разрядный процессор за одну операцию обрабатывает 8 бит данных, 32-разрядный – 32 бита данных.

С другими устройствами процессор связан группами проводников, которые называются шинами: адресная шина, командная шина, шина данных. Данные, которые передаются по ***адресной шине***, трактуются как адреса ячеек оперативной памяти. По ***шине данных*** происходит копирование данных из оперативной памяти в регистры процессора и наоборот. По ***командной шине*** из оперативной памяти поступают команды, выполняемые процессором.

Устройство памяти компьютера

Память компьютера – это совокупность устройств для хранения программ и данных. Во ***внутренней памяти*** хранятся сравнительно небольшие объемы информации, участвующей в процессах обработки при включенном компьютере. ***Внешняя память*** нужна для длительного хранения больших объемов информации независимо от того, включен или вы-

ключен компьютер. ***Энергозависимой*** называется память, данные в которой стираются при выключении компьютера. ***Энергонезависимая*** память не стирается при выключении компьютера.

Содержимое ***постоянного запоминающего устройства*** (ПЗУ) устанавливается на заводе-изготовителе и в дальнейшем не меняется. Обычно в ПЗУ записываются программы, обеспечивающие минимальный базовый набор функций управления устройствами компьютера. При включении компьютера первоначально управление передается программе BIOS из ПЗУ, которая тестирует компоненты компьютера и запускает программу-загрузчик операционной системы.

К энергозависимой внутренней памяти относятся ***оперативное запоминающее устройство*** (ОЗУ), видеопамять и кэш-память. В ОЗУ в двоичном коде хранятся информация, программа ее обработки, промежуточные данные и результаты работы. ОЗУ обеспечивает режимы записи, считывания и хранения информации, причём в любой момент времени возможен доступ к любой произвольно выбранной ячейке памяти (RAM – Random Access Memory, память с произвольным доступом). Часть оперативной памяти (***видеопамять***) отводится для хранения изображений, получаемых на экране монитора. Чем больше видеопамять, тем более сложные и качественные изображения можно получать на дисплее. Высокоскоростная ***кэш-память*** служит для увеличения скорости выполнения операций компьютером и используется при обмене данными между микропроцессором и RAM.

Устройства внешней памяти с произвольным доступом позволяют получить доступ к произвольному блоку данных примерно за одно и то же время доступа. Используют следующие основные типы устройств памяти с произвольным доступом.

1. Накопители на жёстких магнитных дисках (винчестеры, НЖМД) – несъемные жесткие магнитные диски. Это основной вид внешней памяти.

2. Накопители на гибких магнитных дисках (флоппи-дисководы, НГМД) – устройства для записи и считывания информации с небольших съемных магнитных дисков (дискет).

Максимальная ёмкость 3,5-дюймовой дискеты – 1,44 Мбайт (в настоящее время используются крайне редко).

3. Оптические диски (CD-ROM, DVD-ROM) – устройства хранения информации, для считывания которой используется луч лазера.

4. Флэш-память (Flash) – съемные накопители данных, отличающиеся малым размером и постоянно растущим от одной модели к другой объемом памяти.

5. Съемные жесткие диски (СЖД) – съемные устройства памяти. Емкость современных СЖД достигает нескольких сотен гигабайт.

Устройства памяти с последовательным доступом представлены накопителями на магнитных лентах (НМЛ, стримеры).

Устройства ввода информации

Устройства ввода информации используются для ввода в память компьютера программ, данных для работы программ и команд управления компьютером. Наиболее распространенным устройством ввода информации является ***клавиатура –*** стандартное устройство, предназначенное для ручного ввода информации. Работой клавиатуры управляет контроллер клавиатуры, расположенный на материнской плате. При нажатии пользователем клавиши на клавиатуре контроллер преобразует код нажатой клавиши в соответствующую последовательность битов. Существуют беспроводные модели клавиатуры, в них связь клавиатуры с компьютером осуществляется посредством инфракрасных лучей.

К ***манипуляторам*** относят устройства, преобразующие движения руки пользователя в управляющую информацию для компьютера. Среди манипуляторов выделяют *мыши, трекболы, джойстики*. ***Мышь*** повышает комфорт работы пользователя и предназначена для выбора и перемещения графических объектов на дисплее. ***Трекбол*** – это «мышка наоборот». Само устройство, в отличие от мышки, всегда остается неподвижным, а управление перемещением курсора осуществляется вращением шарика, который находится в верхней части трекбола. ***Джойстик*** представляет собой основание с подвижной рукояткой, которая может наклоняться в про-

дольном и поперечном направлениях. Рукоятка и основание снабжаются кнопками.

Дигитайзер – это устройство для ввода графических данных (чертежи, схемы, планы и т. п.). Он состоит из планшета и соединенного с ним визира или специального карандаша. Перемещая карандаш по планшету, пользователь рисует изображение, которое выводится на экран.

Сканер – устройство ввода графических изображений в компьютер. Устройство считывает изображения с бумажных носителей и пересылает компьютеру в цифровом виде. Во время сканирования вдоль листа с изображением плавно перемещается мощная лампа и линейка с множеством расположенных на ней в ряд светочувствительных элементов. Главные характеристики сканеров – это скорость считывания, которая выражается количеством сканируемых страниц в минуту *(pages per minute – ppm)*, и разрешающая способность, выражаемая числом точек получаемого изображения на дюйм оригинала *(dots per inch – dpi)*.

Устройства вывода информации

Монитор (дисплей) является наиболее распространенным устройством вывода информации. Любое изображение на экране монитора образуется из светящихся разными цветами точек, называемых пикселями. ***Пиксель*** – это самый мелкий элемент, который может быть отображен на экране. По принципу действия мониторы подразделяются на мониторы с электронно-лучевой трубкой *(Catode Ray Tube – CRT)* и жидкокристаллические *(Liquid Crystal Display – LCD)*.

В ***мониторах с электронно-лучевой трубкой*** изображение формируется с помощью зерен люминофора – вещества, которое светится под воздействием электронного луча. ***Жидкокристаллические мониторы*** имеют меньшие размеры, потребляют меньше электроэнергии. Принцип отображения на жидкокристаллических мониторах основан на поляризации света. Источником излучения здесь служат лампы подсветки, расположенные по краям жидкокристаллической матрицы.

Для получения копий изображения на бумаге применяют *принтеры*. ***Матричные принтеры*** схожи по принципу действия с печатной машинкой. Печатающая головка перемещается

в поперечном направлении и формирует изображение из множества точек, ударяя иголками по красящей ленте. ***Струйный принтер*** относится к безударным принтерам. Изображение в нем формируется с помощью чернил, которые распыляются через капилляры печатающей головки. ***Лазерный принтер*** также относится к безударным принтерам. Первоначально изображение создается на фотобарабане, который предварительно электризуется статическим электричеством. Луч лазера в соответствии с изображением снимает статический заряд на белых участках рисунка. Затем на барабан наносится специальное красящее вещество – тонер, который прилипает к фотобарабану на участках с неснятым статическим зарядом. Тонер переносится на бумагу и нагревается, частицы тонера плавятся и прилипают к бумаге.

? Вопросы и упражнения для самоконтроля

1. В чем заключаются основные отличия компьютеров второго поколения от своих предшественников?
2. Какими особенностями характеризуется третье поколение компьютеров? Какие значимые события в области информационных технологий связывают с появлением третьего поколения компьютерной техники?
3. Назовите отличительные особенности компьютеров четвертого поколения.
4. Кем и когда были сформулированы основные принципы построения компьютеров? Назовите эти принципы.
5. Что такое память компьютера? В чем принципиальная разница между энергозависимой и энергонезависимой памятью?
6. Какие устройства находятся внутри системного блока персонального компьютера?
7. Каково назначение блока питания персонального компьютера?
8. Какие элементы расположены на системной плате персонального компьютера?
9. Назовите основные характеристики микропроцессора.

3.2. Программные средства реализации информационных процессов

Программа – это запись алгоритма решения задачи в виде последовательности команд на языке, который понимает компьютер. ***Программное обеспечение (ПО)*** – совокупность программ, процедур и правил, а также документации, обеспечивающих функционирование системы обработки данных. По способу взаимодействия с аппаратными средствами компьютеров все программное обеспечение можно распределить по уровням, где каждый высший уровень базируется на программном обеспечении предшествующих уровней (рис. 3.3).

Базовое ПО отвечает за взаимодействие с базовыми аппаратными средствами, содержится в составе базового аппаратного обеспечения и сохраняется в специальных микросхемах постоянного запоминающего устройства (ПЗУ), образуя базовую систему ввода-вывода (BIOS).

Системное ПО обеспечивает взаимодействие других программ компьютера с программами базового уровня и непосредственно с аппаратным обеспечением. Совокупность программного обеспечения системного уровня образует ядро операционной системы компьютера. Ядро операционной системы выполняет функции управления памятью, процессами ввода-вывода и файловой системой, учета использования ресурсов, обработки команд и т. д.

Рис. 3.3. Классификация ПО компьютеров по способу взаимодействия с аппаратными средствами

Программы ***служебного уровня*** взаимодействуют как с программами базового уровня, так и с программами системного уровня. Они предназначены для автоматизации работ по про-

верке и настройке компьютерной системы, а также для улучшения функций системных программ. Некоторые служебные программы входят в состав операционной системы, дополняя ее ядро, другие программы являются внешними и расширяют функции операционной системы.

Прикладное ПО используется для решения конкретных задач обработки информации. Программы прикладного уровня реализуют свои функции в непосредственном взаимодействии с программами системного уровня.

3.2.1. Системное программное обеспечение компьютеров

Операционная система (ОС) – это комплекс программ, управляющих аппаратными и программными средствами компьютера с целью выполнения задач пользователя.

Операционная система обычно хранится во внешней памяти компьютера. При включении компьютера она считывается с дисковой памяти и размещается в ОЗУ. Этот процесс называется ***загрузкой операционной системы***. После загрузки операционной системы вся работа с процессором и другими устройствами осуществляется посредством специальных программ, входящих в операционную систему. Если по каким-то причинам загрузка операционной системы с жесткого диска не состоится, то работа с компьютером невозможна.

По числу одновременно выполняемых задач операционные системы могут быть разделены на два класса: ***однозадачные*** (например, MS DOS) и ***многозадачные*** (OS/2, UNIX, Windows). Многозадачные ОС управляют разделением совместно используемых ресурсов (процессор, оперативная память, файлы и внешние устройства) между выполняемыми параллельно задачами.

По числу одновременно работающих пользователей ОС делятся на ***однопользовательские*** и ***многопользовательские***. Главным отличием многопользовательских систем от однопользовательских является наличие средств защиты информации каждого пользователя от несанкционированного доступа других пользователей.

Многозадачные ОС подразделяются на три типа в соответствии с использованными при их разработке критериями

эффективности: системы пакетной обработки, системы разделения времени, системы реального времени. В ***системах пакетной обработки*** формируется пакет заданий. В процессе выполнения пакета пользователь не может вмешиваться в процесс выполнения заданий.

В системах разделения времени каждому пользователю предоставляется терминал, с которого он может вести диалог со своей программой. Критерием эффективности систем разделения времени является удобство и эффективность работы пользователя.

Системы реального времени применяются для управления различными техническими объектами (станок, спутник, научная экспериментальная установка) или технологическими процессами (гальваническая линия, доменный процесс и т. п.). Основным критерием эффективности для систем реального времени является их способность выдерживать заранее заданные интервалы времени между запуском программы и получением результата.

Операционная система Windows

☑ В дальнейшем термином «операционная система Windows» будем обозначать семейство операционных систем, начало которому было положено появлением в 1995 году операционной системы Windows 95. Рассмотрим основные характеристики ОС Windows.

Многозадачность. ОС Windows способна «одновременно» выполнять несколько программ. Микропроцессор в каждый момент времени выполняет команды только одной программы, но операционная система настолько быстро реагирует на потребности той или иной программы, что создается впечатление одновременности их работы.

Поддержка технологии «Plug and Play» («включай и работай»). ОС самостоятельно распознает подключаемое к ПК периферийное устройство (принтер, сканер и т. д.) и производит его установку.

Использование виртуальной памяти. Проблема нехватки оперативной памяти решается в среде Windows с помощью виртуальной оперативной памяти. ***Виртуальная память*** – это расширение ОЗУ, полученное за счет использования

части внешней дисковой памяти. Используемая для этой цели часть дисковой памяти называется ***файлом подкачки***.

Наличие коммуникационных программных средств. Важнейшим свойством Windows является включение в ее состав специальных программных средств для поддержки коммуникаций и компьютерных сетей.

Наличие средств мультимедиа. ОС Windows обеспечивает работу со звуком, изображениями и видео при помощи специальных аппаратных и программных средств.

Графический пользовательский интерфейс. Пользователь управляет работой компьютера, работая с системой графических изображений на экране монитора. ***Рабочий стол*** – основной экран Windows. На рабочем столе располагается ***Панель задач***, на которой отображаются кнопки активных в данный момент программ и кнопка ***Пуск***, при нажатии на которую появляется ***Главное меню*** Windows. ***Окна*** – основные объекты графического пользовательского интерфейса Windows. Различают окна следующих типов: окно папки, диалоговое окно, окно справочной системы, окно приложения, окно документа.

Windows позволяет рассматривать диски, папки и файлы как объекты. Все эти объекты имеют определенные свойства, и над ними могут проводиться определенные операции. Ознакомиться со свойствами объекта, а также выполнить над ним разрешенные операции можно с помощью ***контекстного меню***. Для вызова контекстного меню необходимо осуществить правый щелчок мышью на значке объекта.

Файлы и файловая система Windows

Файл – это определенное количество информации, имеющее имя и хранящееся в долговременной (внешней) памяти. ***Имя файла*** состоит из двух частей, разделенных точкой: собственно имя файла и расширение, определяющее его тип (программа, данные и т. д.).

Файловая система – это система хранения файлов и организации каталогов. Каталог содержит для каждого файла его имя и указание на начало его размещения на диске. Начальный, корневой каталог содержит вложенные каталоги

1-го уровня, в свою очередь, каждый из них может содержать вложенные каталоги 2-го уровня и т. д. В каталогах всех уровней могут храниться файлы.

Чтобы найти конкретный файл, необходимо указать ***путь к файлу***. В путь к файлу входят записываемые через разделитель «\» логическое имя диска и последовательность имен вложенных друг в друга каталогов, в последнем из которых содержится нужный файл. Путь к файлу вместе с именем файла называют ***полным именем файла***. Пример полного имени файла: **C\ДОКУМЕНТЫ\CHE\che.exe**

Иерархическая файловая система представлена в операционной системе Windows в форме иерархической системы папок и документов. На вершине иерархии папок находится папка **Рабочий стол**. Следующий уровень представлен папками **Мой компьютер, Корзина, Мои документы** и др.

В окне папки **Мой компьютер** находятся значки имеющихся в компьютере дисков. Открыв любой диск, можно увидеть список папок, находящихся на этом диске. Затем можно открыть любую папку и увидеть её содержимое и т. д. Папка может содержать значки других папок, значки файлов и ярлыки. ***Ярлык –*** это ссылка на какой-либо объект, вторичное изображение этого объекта, указывающая на его местоположение. Для одного объекта можно создать любое количество ярлыков. Папка **Корзина** временно содержит все удаленные папки и файлы. При необходимости удаленные и хранящиеся в **Корзине** папки и документы можно восстановить.

Сервисные системные программы

Важным классом системных программ являются программы вспомогательного назначения – ***утилиты***. Они либо расширяют и дополняют соответствующие возможности операционной системы, либо решают самостоятельные важные задачи.

Программы для контроля, тестирования и диагностики компьютера используются для проверки правильности функционирования устройств компьютера и для обнаружения неисправностей в процессе эксплуатации, указывают причину и место неисправности. Для тестирования компьютеров используются встроенные средства операционных

систем и специальные пакеты программ (**Norton Utilities**, **SiSoft SANDRA** и др.).

Программы для обслуживания дисков позволяют выполнять различные сервисные функции по отношению к жестким магнитным дискам: разбиение диска на разделы, форматирование, проверка диска на наличие логических и физических ошибок, дефрагментация, очистка диска. Рассмотрим эти функции подробнее.

Жесткий диск может быть разбит на разделы (логические диски), в каждом из которых может быть создана своя файловая система. Последние версии ОС Windows имеют встроенную программу разбиения жестких дисков на разделы. Кроме того, существуют специализированные программы, например, **FDisk** или **Partition Magic**.

Различают низкоуровневое (физическое) форматирование, которое делит поверхности магнитных пластин на дорожки и сектора, и высокоуровневое (логическое), которое заключается в разбиении на кластеры и размещении на диске файловой системы. Логическое форматирование выполняется средствами ОС Windows. Необходимо помнить, что при форматировании диска все хранящиеся на нем данные будут потеряны.

Вследствие некорректного завершения работы компьютера, при зависании системы могут возникать ошибки на дисках. Обнаружить их и предотвратить связанные с этим проблемы поможет стандартная программа Windows **Проверка диска**.

Каждый файл на диске занимает определенное пространство. Это пространство разбито на кластеры. Каждый кластер принадлежит определенному файлу. Операционная система не всегда может выделить файлу место таким образом, чтобы его кластеры шли друг за другом. В этом случае говорят, что файл фрагментирован. Если на диске образуется много таких файлов, то скорость работы системы заметно падает. Для решения этой проблемы можно использовать программу ОС Windows **Дефрагментация диска**. При регулярной работе на компьютере иногда накапливается некоторый пользовательский и системный «мусор», который

полезно периодически ликвидировать. Для этого существует много различных программ, а в ОС Windows существует утилита **Очистка диска**.

Программы-архиваторы позволяют за счет применения специальных алгоритмов сжимать информацию на дисках, т. е. создавать копии файлов меньшего размера. Представители этого класса программ – программы **WinRar** и **WinZip**.

Антивирусные программы предназначены для предотвращения заражения компьютерными вирусами и ликвидации последствий заражения вирусом. Представители семейства антивирусных программ – **Kaspersky Antivirus**, **DrWeb**, **Norton Antivirus**.

Коммуникационные программы предназначены для организации обмена информацией между компьютерами.

3.2.2. Прикладное программное обеспечение компьютеров

Прикладные программы предназначены для того, чтобы обеспечить применение вычислительной техники в различных сферах деятельности человека.

Редакторы текстовых документов используются для создания и редактирования текстовых документов с помощью компьютера. Представители редакторов документов – программы **Microsoft Word**, **Wordpad, Блокнот**.

Браузеры – средства просмотра Web-документов. Способны воспроизводить текст, графику, звук, видеоконференции, разрешают работать с электронной почтой. Широкое распространение получили браузеры **Internet Explorer, Opera, Mozilla Firefox**.

Графические редакторы – широкий класс программ, предназначенных для создания и обработки графических изображений. В ***растровых редакторах*** (**Adobe Photoshop, Paint**) графический объект представлен в виде комбинации точек, каждая из которых имеет свои яркость и цвет. В ***векторных редакторах*** (**Corel Draw, Adobe Illustrator, OpenOffice.org Draw**) изображения строятся на основе линий. Каждая линия рассматривается как математическая кривая 3-го порядка и представлена формулой. Особый класс программ для работы с любыми видами изображений представляют

программы-просмотрщики графических изображений. Они позволяют просматривать графические файлы различных форматов, создавать фотоальбомы на жестком диске, перемещать, переименовывать, изменять размеры, а также конвертировать изображения из одного формата в другой. На сегодняшний день лидером в данной области является программа **ACDSee**.

Электронные таблицы предоставляют комплексные средства для хранения и обработки разных типов данных, структурированных в виде таблиц. Основная особенность электронных таблиц состоит в автоматическом изменении содержимого всех ячеек при изменении отношений, заданных математическими или логическими формулами. Представители семейства табличных процессоров – **Microsoft Excel**, **Quatro Pro, OpenOffice.org Calc**.

Системы управления базами данных (СУБД) используются для хранения и обработки больших массивов структурированных данных. Представители данного класса программ – **Microsoft Access, Clipper, Paradox, OpenOffice.org Base**.

Средства разработки презентаций и публикаций. Компьютерная презентация представляет собой набор слайдов (электронных страниц), последовательность показа которых может меняться в процессе демонстрации презентации. Наиболее популярной программой для создания презентаций является **Microsoft PowerPoint**. ***Компьютерная публикация*** представляет собой набор электронных документов, который может использоваться как на компьютере, так и в сети Интернет. Они могут быть представлены в виде Web-сайтов, буклетов. Характерным примером программы для создания компьютерных публикаций является программа **Microsoft Publisher**.

Автоматизация ввода информации в компьютер. Основным методом перевода бумажных документов в электронную форму является сканирование, в результате которого создается графический образ бумажного документа. Для перевода графического образа в текстовый формат используются специальные программные средства, называемые ***средствами распознавания образов***. Из программ, способных распоз-

навать текст на русском языке, наиболее известной является программа **Fine Reader**.

Системы автоматизированного проектирования (САПР) позволяют осуществлять черчение и конструирование различных предметов и механизмов с помощью компьютера. В настоящее время широкое распространение получили программы **AutoCad, ArchiCad**, **Компас**.

Настольные издательские системы позволяют автоматизировать процесс верстки полиграфических изданий. Существуют различные настольные издательские системы, среди которых можно выделить следующие: **Adobe PageMaker**, **Corel Ventura**, **QuarkXPress**.

Системы автоматизированного перевода используются для перевода текстов с одного языка на другой. ***Электронные словари*** – это средства для перевода отдельных слов в документе (**ABBYY Lingvo**). ***Программы перевода*** переводят весь заданный текст с одного языка на другой. Наиболее популярной является программа **Promt XT**.

Бухгалтерские системы выполняют функции текстовых, табличных редакторов и СУБД. Предназначены для автоматизации подготовки начальных бухгалтерских документов предприятия и их учета, регулярных отчетов по итогам производственной, хозяйственной и финансовой деятельности (**1С: Предприятие**, **Инфо-бухгалтер** и др.).

Правовые базы данных содержат тексты нормативных документов и предоставляют возможности справки, контекстного поиска, распечатки и т. д. Представители правовых баз данных – пакеты **Гарант** и **Консультант+**.

Экспертные системы предназначены для анализа данных, содержащихся в базах знаний и выдачи результатов по запросу пользователя. Такие системы используются в медицине, фармакологии, юриспруденции и др. областях.

Геоинформационные системы (ГИС) предназначены для автоматизации картографических и геодезических работ на основе информации, полученной топографическим или аэрографическими методами.

Интегрированные системы сочетают в себе возможность системы управления базами данных, табличного процессора,

текстового редактора, системы деловой графики, а иногда и другие возможности. Как правило, все компоненты интегрированной системы имеют схожий интерфейс, что облегчает обучение работе с ними. Представители интегрированных систем – пакеты **Microsoft Office, Open Office.org**.

? Вопросы и упражнения для самоконтроля

1. Какие уровни программного обеспечения можно выделить по способу взаимодействия с аппаратными средствами?
2. К какому типу программного обеспечения относятся антивирусные программы, графические редакторы?
3. Для чего необходима операционная система?
4. По какому признаку операционные системы разделяют на многозадачные и однозадачные?
5. В чем заключается свойство «поддержка технологии Plug and Play» ОС Windows?
6. Как решается в ОС Windows проблема нехватки оперативной памяти?
7. Какие объекты образуют графический пользовательский интерфейс ОС Windows?
8. Для чего предназначено и как открывается контекстное меню объекта Windows? Как можно увидеть свойства любого объекта Windows?
9. Что такое файл? Каковы правила образования имен файлов?
10. Что такое каталог? Как организовано хранение каталогов и файлов на диске?
11. Какова структура системного программного обеспечения компьютеров?
12. Назовите основные классы прикладных программ, выполнив классификацию по функциональному назначению программ.

3.3. Работа с текстовыми документами на компьютере

Наиболее широкое распространение в качестве средства подготовки комплексных текстовых документов получил в настоящее время текстовый процессор Microsoft

Word (MS Word). В данном пособии рассматривается версия программы Microsoft Word 2003.

3.3.1. Пользовательский интерфейс программы MS Word. Ввод, редактирование и форматирование текста

После запуска программы на экране монитора появляется её рабочее окно. Большую часть окна занимает ***рабочая область документа***. Кроме того, в окне находятся строка управляющего меню программы, панели инструментов, строка состояния документа. В правой части окна может находиться ***область задач***, в которой дублируются некоторые команды управляющего меню.

Режимы и масштаб просмотра документа

Любой документ может быть просмотрен на экране в нескольких режимах: обычный, разметка страницы, структура, Web-документ. В режиме ***Разметка страницы*** документ показывается на экране так, как он будет напечатан. Режим устанавливается командами меню **Вид – Разметка страницы** или щелчком по кнопке **Режим разметки**  . Управлять размером изображения текста документа на экране можно с помощью команды **Вид – Масштаб** или инструмента **Масштаб** на панели **Стандартная**.

☑ В тексте документа присутствуют ***непечатаемые знаки***, которые важны для определения структуры документа, но на печать не выводятся. Сделать эти знаки видимыми на экране можно, если нажать кнопку **Непечатаемые знаки**  на панели **Стандартная**. Видеть непечатаемые знаки полезно при разрешении возникающих проблем в процессе работы с документом.

Ввод и редактирование текста

Пользователь вводит текст с клавиатуры, при этом осуществляется автоматический переход с одной строки на другую. Если при вводе текста нажать клавишу **<Enter>**, это приведет к образованию нового абзаца. Одновременное нажатие клавиш **<Shift>** + **<Enter>** обеспечивает переход на новую строку в пределах текущего абзаца. Если требуемый в тексте символ отсутствует на клавиатуре (например, сим-

вол **§**), для его ввода можно выполнить **Вставка – Символ**, найти недостающий символ и нажать кнопку **Вставить**.

Параметры страницы

Установка параметров страницы выполняется в диалоговом окне **Параметры страницы**, которое открывается командами меню **Файл – Параметры страницы**. В окне можно установить размеры полей, ориентацию страницы (***книжная*** или ***альбомная***), размер листа бумаги и др.

Колонтитулами называют области, расположенные в верхнем и нижнем поле каждой страницы документа (рис. 3.4). В колонтитул можно вставить текст, рисунок, дату печати документа и т. п. Колонтитул формируется на одной странице раздела документа и затем воспроизводится на каждой странице данного раздела (новый раздел создается в документе при вставке разрыва раздела командами меню **Вставка – Разрыв – Новый раздел...**).

Номера страниц являются частью верхнего или нижнего колонтитула и могут быть вставлены в процессе редактирования колонтитулов. Можно также вставить номера страниц командами меню **Вставка – Номера страниц**. Удаление номеров страниц выполняется в процессе редактирования колонтитула, содержащего номера страниц.

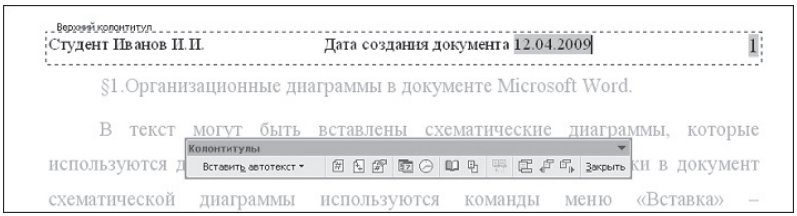


Рис. 3.4. Редактирование верхнего колонтитула

Параметры абзаца

Абзац – это часть текста, которая заканчивается символом конца абзаца. Этот символ вставляется в текст при нажатии на клавишу **<Enter>**. В режиме просмотра непечатаемых знаков конец абзаца в тексте представлен символом **¶**. Установка параметров абзаца выполняется в диалоговом окне **Абзац**, которое открывается командами меню **Формат – Абзац**.

На вкладке **Отступы и интервалы** можно установить ***отступы абзаца***: **слева** – отступ левой границы от левого поля; **справа** – отступ правой границы от правого поля; **первая строка** – отступ первой строки от левой границы абзаца. Отступы абзаца можно также установить на горизонтальной масштабной линейке с помощью ***бегунков*** (рис. 3.5). Левый нижний бегунок определяет положение левой границы абзаца, правый нижний – положение правой границы абзаца, верхний бегунок – положение первой строки.

Рис. 3.5. Установка отступов абзаца на линейке

Из списка **Выравнивание** в диалоговом окне **Абзац** можно выбрать способ ***выравнивания абзаца***, в группе опций **Интервал** – установить интервал между строками внутри абзаца, а также дополнительные интервалы перед абзацем и после абзаца.

Параметры шрифта

Параметры шрифта устанавливаются в диалоговом окне **Шрифт**, которое открывается командами меню **Формат – Шрифт**. Основные параметры шрифта: ***шрифтовой набор*** или ***гарнитура*** (общий стиль оформления символов), ***начертание*** (обычный, *курсив*, **полужирный**, ***полужирный курсив***), ***размер***, ***цвет***, ***подчеркивание***.

Основные параметры шрифта и абзаца можно определить с помощью инструментов панели **Форматирование** (рис. 3.6).

Стилем форматирования называется набор параметров форматирования, который применяется к тексту, таблицам и спискам. Стили позволяют одним действием применить сразу всю группу атрибутов форматирования. С помощью области задач **Стили и форматирование** можно создавать, просматривать и применять стили.

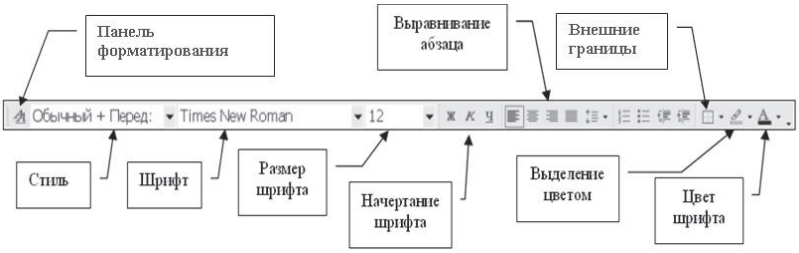


Рис. 3.6. Панель инструментов **Форматирование**

3.3.2. Представление текста в формате списков и таблиц. Вставка в документ графических объектов и формул

Списки и таблицы

Список – это несколько расположенных подряд абзацев, отмеченных маркерами или номерами. Форматировать списки можно несколькими способами: задав команду **Формат – Список,** с помощью команды **Список** из контекстного меню или с помощью кнопок **Нумерация** и **Маркеры** на панели **Форматирование**.

Существуют разные способы создания таблицы в документе:

- командой **Таблица – Вставить – Таблица**;
- кнопкой **Добавить таблицу** на панели инструментов **Стандартная**;
- командами **Таблица – Преобразовать – Текст в таблицу**.

Создав таблицу, можно изменять ее структуру (добавляя, удаляя или перемещая ячейки, строки, столбцы; объединяя и разъединяя части таблицы); сортировать данные в таблице; снабжать таблицу заголовком и т. д. Команды редактирования таблицы содержатся в меню **Таблица**.

Содержимое ячеек таблицы можно отформатировать как обычный текст, используя команды меню **Формат**. К операциям форматирования таблицы относятся также: выбор цвета заливки ячеек, выбор типа, толщины и цвета линий границ ячеек, выбор направления текста в ячейках. Командой **Таблица – Автоформат** можно автоматически форматировать таблицу, выбирая один из стандартных форматов.

Операции по редактированию и форматированию таблицы удобно выполнять с помощью инструментов панели **Таблицы и границы**, которая выводится на экран командами **Вид – Панели инструментов – Таблицы и границы**.

Операции с рисунками

Доступ к коллекции клипов осуществляется приложением **Microsoft Clipart** (**Clip Gallery**), входящим в состав MS Office и поддерживающим рисунки, звуки, видео. Для входа в Microsoft Clipart следует выполнить в меню **Вставка – Рисунок – Картинки**. Откроется панель задач **Вставка картинки**, на которой нужно выбрать пункт **Коллекция картинок**. В появляющемся диалоговом окне следует выбрать необходимый раздел и рисунок.

Для вставки графических объектов из файлов используется команда **Вставка – Рисунок – Из файла**. Далее в диалоговом окне **Добавление рисунка** нужно найти файл рисунка и щелкнуть кнопку **Вставить**.

Выделив рисунок, его можно *переместить в другое место, скопировать, изменить размер, удалить*. Перемещение и копирование удобно выполнять с помощью мыши. Можно выполнить форматирование вставленного в документ рисунка, т. е. изменить яркость и заливку, размер, способ обтекания рисунка текстом, положение, сделать обрезку и т. д. Для этого нужно выделить рисунок (щелчком левой кнопки) и, открыв щелчком правой кнопки мыши контекстное меню, выбрать в нем (щелчком левой кнопки мыши) команду **Формат рисунка**.

Графические объекты в MS Word можно создавать и редактировать, используя инструменты панели **Рисование**. При этом изображение строится из элементарных графических фигур (квадратов, окружностей, линий и др.).

Вставка в документ формул

В программе MS Word средством создания, редактирования и форматирования сложных математических формул является редактор формул **Microsoft Equation 3.0**. Для запуска редактора служит команда **Вставка – Объект**. В открывшемся диалоговом окне **Вставка объекта** следует выбрать пункт **Microsoft Equation 3.0**. При этом строка меню окна текстового процессора замещается строкой меню окна ре-

дактора формул и появляется панель инструментов **Формула**. Ввод формулы завершается закрытием панели редактора формул или щелчком левой кнопкой мыши в поле документа вне области ввода формулы. Для редактирования формулы непосредственно в документе достаточно выполнить на ней двойной щелчок.

3.3.3. Структурирование документа. Выполнение сервисных функций

Поиск и замена текста в документе

Для поиска текста в документе следует выполнить в меню **Правка – Найти**, затем в поле **Найти** ввести искомый текст. Можно производить замену одних текстовых фрагментов на другие, используя команду **Правка – Заменить**. Для этого следует в поле **Найти** ввести искомый текст, а в поле **Заменить** – заменяющий текст.

Автозамена и автотекст

С помощью ***автозамены*** MS Word автоматически заменяет элементы при наборе текста на клавиатуре. Например, можно создать сокращения для часто встречающихся в тексте слов и словосочетаний. Для создания элемента автозамены в меню **Сервис** нужно выбрать команду **Параметры автозамены**, выбрать вкладку **Автозамена**, а затем ввести в списке **Заменять при вводе** аббревиатуру или слово, которое следует автоматически заменять в процессе ввода с клавиатуры, и текст, который будет появляться взамен указанного.

Автотекст можно использовать для сохранения иллюстрации, графики, часто используемого фрагмента текста и т. д. Чтобы создать элемент автотекста, нужно выделить фрагмент документа, выбрать в меню **Сервис – Параметры автозамены**, перейти на вкладку **Автотекст**, ввести имя для фрагмента и нажать кнопку **Добавить**. Чтобы вставить элемент автотекста в документ, нужно выбрать команду **Автотекст** в меню **Правка** и выбрать элемент автотекста из списка всех существующих элементов.

Вставка разрывов

В документ могут быть вставлены разрывы, позволяющие в любом месте документа начать новую страницу (встав-

ляется ***разрыв страницы***), новый раздел (вставляется ***разрыв раздела***), новую колонку (вставляется ***разрыв колонки***). Для вставки разрыва нужно выполнить в меню **Вставка – Разрыв**. Откроется диалоговое окно для выбора типа разрыва.

Новый раздел документа организуется в том случае, если нужно, например, изменить ориентацию страниц в части документа с книжной на альбомную или наоборот. При изменении числа колонок текста в какой-либо части документа автоматически образуется новый раздел.

Вставка закладок и гиперссылок

Гиперссылка – это цветной подчеркнутый текст или графический объект в документе, по щелчку на котором выполняется переход к определенному месту в текущем документе, к другому файлу или Web-странице в сети Интернет. Для создания гиперссылок на элементы текущего документа используются либо стили заголовков, либо закладки.

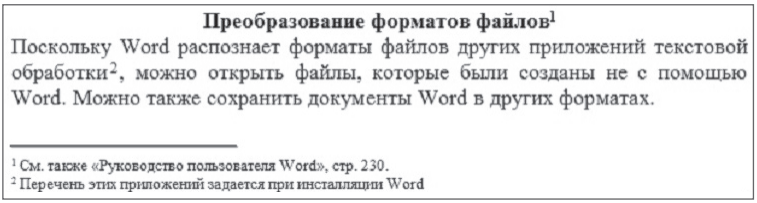
Закладка – отметка или место в тексте, которому присвоено определенное имя, что позволяет быстро переходить к нему в дальнейшем. Для создания закладки нужно установить курсор в позицию вставки закладки и выполнить в меню **Вставка – Закладка**, ввести имя закладки и нажать кнопку **Добавить**. Закладка будет добавлена в документ, это не приведет к каким-то изменениям во внешнем представлении текста. Однако в дальнейшем можно будет создать гиперссылку, которая указывает на эту закладку.

Вставка оглавления и сноски

Оглавление представляет собой список заголовков документа. Чтобы можно было создать оглавление автоматически, нужно все заголовки в документе отформатировать стилями заголовков (**Заголовок 1, Заголовок 2** и т. д.). Для создания оглавления нужно выполнить в меню **Вставка – Ссылка – Оглавление и указатели**. При сборке оглавления в него будут включены все абзацы текста, отформатированные стилями заголовков с указанием страниц, на которых расположены эти заголовки.

Сноски используются в документе для пояснений, комментариев и ссылок на другие документы (рис. 3.7). Сноска состоит из двух связанных частей: **знака сноски** и **текста сноски**.

Знак сноски ставится справа от последнего слова текстового фрагмента, к которому делается пояснение. Текст сноски, отмеченный соответствующим знаком, печатается внизу страницы (обычная сноска) или в конце документа (концевая сноска).


Преобразование форматов файлов[1]

Поскольку Word распознает форматы файлов других приложений текстовой обработки[2], можно открыть файлы, которые были созданы не с помощью Word. Можно также сохранить документы Word в других форматах.

[1] См. также «Руководство пользователя Word», стр. 230.
[2] Перечень этих приложений задается при инсталляции Word

Рис. 3.7. Текст со сносками

Для вставки сноски курсор помещается в позицию вставки сноски и в меню выполняются команды **Вставка – Ссылка – Сноска**.

Режим переноса слов и проверка правописания

Для включения режима разделения слов на части при переходе с одной строки на другую следует выполнить в меню **Сервис – Язык – Расстановка переносов** и включить параметр **Автоматическая расстановка переносов**.

Во время ввода текста программа производит автоматическую проверку орфографии и синтаксиса, если соответствующие опции включены на вкладке **Правописание** (открывается командами меню **Сервис – Параметры**). Слова, отсутствующие в словаре или введенные с ошибкой, выделяются на экране красной волнистой линией. При неправильно расставленных знаках пунктуации или неправильно построенной фразе фрагменты текста выделяются волнистой линией зеленого цвета. Проверка введенного ранее текста начинается нажатием на кнопку **Правописание**  или по команде **Сервис – Правописание**.

3.3.4. Шаблоны и формы. Создание рассылок

Использование шаблонов и форм

☑ Каждый документ MS Word базируется на определенном шаблоне. ***Шаблон*** – это образец для подготовки типовых документов. В шаблоне хранятся стили оформления,

элементы автотекста, индивидуальные меню и панели инструментов пользователя. Шаблоны сохраняются в файлах с расширением **.dot**.

При выполнении команды **Создать** из меню **Файл** открывается область задач, в которой нужно выбрать шаблон для создания документа. При выборе опции **Новый документ** открывается универсальный шаблон **Normal.dot**. Однако пользователь может выбрать другой шаблон, указав опцию **На моем компьютере** в области задач.

Если мы хотим создать не документ, а новый шаблон, нужно в окне выбора шаблона в группе **Создать** установить переключатель **шаблон**. ***Форма*** – это шаблон, содержащий специальные поля для ввода данных. Форма состоит из постоянной (неизменной) и переменной (изменяемой при вводе данных) частей. Изменяемая часть представлена ***полями***. Существуют следующие виды полей формы: текстовое поле, флажок и поле со списком.

Поле ***Флажок*** используется в тех случаях, когда требуется ввести ответ типа «да/нет» или выделить нужный элемент в заданном перечне.

Поле со списком содержит список возможных ответов для выбора.

Текстовые поля служат для ввода значений с клавиатуры и делятся на несколько типов: ***обычный текст*** – может включать любые символы и используется для ввода коротких текстовых ответов; ***число*** – предназначено для ввода чисел (например, цены или количества в форме заказа); ***дата*** – позволяет ввести дату в одном из стандартных форматов; ***текущая дата*** – используется для ввода в форму текущей даты. Для вставки в документ полей нужно вывести на экран панель инструментов **Формы** (**Вид – Панели инструментов – Формы**) (рис. 3.8).

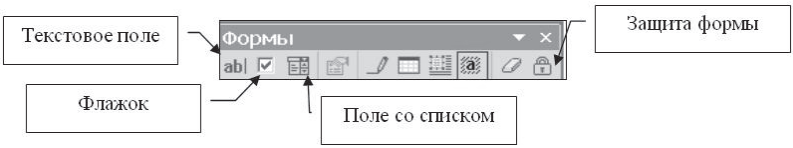


Рис. 3.8. Панель инструментов **Формы**

После того как все поля в шаблон добавлены, его можно защитить от изменений. Для этой цели используется инструмент **Защита формы**. После установки защиты вводить новый текст можно будет только в имеющиеся в документе поля формы.

Использование инструмента слияния для создания рассылок

В жизни часто возникает необходимость отправки стандартных писем нескольким адресатам. Текстовый процессор MS Word дает возможность автоматизированной подготовки подобных писем для рассылки. Эта процедура называется ***слиянием*** и включает несколько шагов.

На первом шаге определяется тип ***основного документа*** (письмо, почтовая наклейка, конверт и т. д.) и создается его текст. Текст, помещенный в основной документ, будет повторен в каждом экземпляре рассылки. На втором шаге определяется ***источник данных***. Источником данных является таблица, каждый столбец которой соответствует одному полю слияния формируемого документа. ***Поля слияния*** – это поля ввода данных в основной документ из документа – источника данных, заполняемые на последнем шаге процедуры слияния. Источником данных может быть любая структурированная в виде таблицы информация, например лист MS Excel.

Разметка документа с помощью вставки полей слияния производится на следующем шаге. На последнем шаге выполняется заполнение полей слияния данными из документа-источника таким образом, что каждому экземпляру данных будет соответствовать отдельный документ рассылки. Каждый из полученных в результате составных документов размещается на отдельной странице нового документа. Полученный документ можно сохранить или вывести на печать.

? Вопросы и упражнения для самоконтроля

1. Как вывести на экран (убрать с экрана) требуемую для работы панель инструментов программы MS Word?
2. Какая информация выводится в строке состояния рабочего окна программы MS Word?
3. Как изменить масштаб просмотра документа?

4. Как вставить в текст символ, отсутствующий на клавиатуре?

5. Назовите параметры форматирования шрифта и абзаца. Как изменить параметры форматирования шрифта, абзаца?

6. Что такое «стиль форматирования»?

7. Назовите параметры форматирования страницы. Как задать параметры страницы?

8. Как создать маркированный или нумерованный список?

9. Какая панель инструментов используется для редактирования и форматирования таблиц?

10. Какой тип объекта используется для создания сложных математических формул в документе?

11. В каких случаях возникает необходимость создания в документе нового раздела?

12. Для чего используются закладки в документе MS Word?

13. Каково назначение шаблонов и форм MS Word?

14. Опишите последовательность создания массовых рассылок при помощи процедуры слияния.

3.4. Работа с электронными таблицами

3.4.1. Пользовательский интерфейс программы Microsoft Excel. Создание и редактирование таблиц

Документ в программе Microsoft Excel (MS Excel) называется ***рабочей книгой***, которая состоит из ***рабочих листов***. Файлы, созданные в программе MS Excel, имеют расширение.**xls**. ***Ячейка*** – минимальная структурная единица на рабочем листе. Каждая ячейка имеет адрес, состоящий из имени столбца и имени строки, например **B25, C100**. Ячейка, в которой в данный момент находится курсор, называется ***активной***. В операциях обработки данных с помощью программы MS Excel часто используется не отдельная ячейка, а ***блок (диапазон) ячеек***, т. е. прямоугольная область смежных ячеек. Например, запись **C3:E6** обозначает блок ячеек, для которого ячейка **C3** является левой верхней ячейкой, а ячейка **E6** – правой нижней.

Ввод, редактирование и форматирование данных

☑ Ячейка может содержать текст, число или формулу. Ввод формулы в ячейку начинается с символа «=». После нажатия клавиши **<Enter>** в ячейке с формулой показывается результат вычисления, сама формула появляется в ***строке ввода*** при выделении ячейки (рис. 3.9).

В качестве операндов в формулах используются числа, текст, ссылки на ячейки (например, **D34** или **R7:U15**), логические значения и условия (например, **S12 >= R22**), стандартные функции. Операнды в формулах соединяются с помощью символов операций: + (сложение), - (вычитание), / (деление), * (умножение), ^ (возведение в степень), > (больше), < (меньше), <= (не больше), = (равно), <> (не равно).

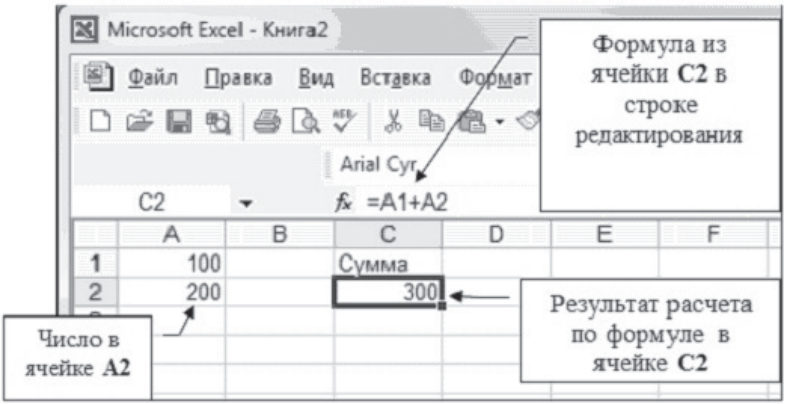


Рис. 3.9. Ввод данных в ячейки

Формат ячеек устанавливается командой меню **Формат – Ячейки**, которая открывает диалоговое окно **Формат ячеек** с несколькими вкладками: **Число, Выравнивание, Шрифт, Граница, Вид, Защита**.

Автозаполнение ячеек

Существует возможность автоматического заполнения смежных ячеек таблицы данными с помощью ***маркера заполнения*** (рис. 3.10). Часто при подготовке таблиц требуется заполнить некоторый диапазон ячеек арифметической или геометрической прогрессией. Автоматически создать такую последовательность можно одним из следующих способов:

• ввести данные в первые две ячейки ряда и выделить их, затем протянуть маркер заполнения по всему ряду (величина приращения числовых значений будет задана разностью значений, находящихся в первых двух ячейках);
• ввести данные в первую ячейку ряда, выделить все ячейки, которые должны быть заполнены данными, выполнить в меню **Правка** – **Заполнить** и указать параметр **Прогрессия**, задать тип заполняемого ряда и указать приращение.

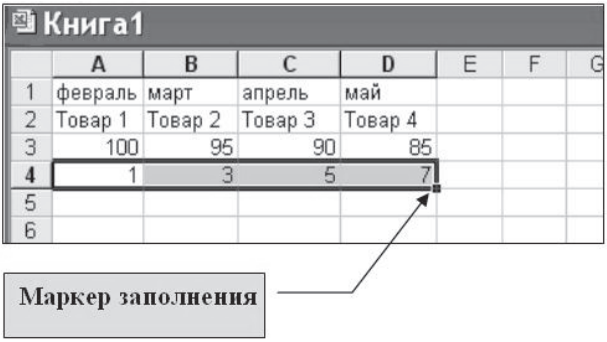


Рис. 3.10. Автозаполнение смежных ячеек

Существует несколько готовых списков (например, названия дней недели, месяцев), которые можно использовать для автозаполнения таблиц. Введя какой-либо элемент такого списка в ячейку и протянув мышью за маркер заполнения, можно автоматически заполнять любой диапазон значениями из этого списка. Для просмотра существующих списков нужно выполнить в меню **Сервис – Параметры** и открыть вкладку **Списки**. Для создания собственного списка следует на вкладке **Списки** выбрать пункт **Новый список** и ввести элементы списка.

3.4.2. Выполнение расчетов и построение диаграмм

Адресация ячеек в формулах

Ссылки на ячейки (адреса ячеек) в формулах могут быть абсолютными, относительными и смешанными. ***Абсолютный адрес*** определяет конкретную ячейку таблицы, перед но-

мером столбца и строки в этом случае указывается символ **$** (например, **$F$7**). ***Относительный адрес*** определяет относительное местоположение адресуемой ячейки от ячейки с формулой (например, **F7**). ***Смешанный адрес*** – комбинация абсолютного и относительного типов (**F$7** или **$F7**).

☑ При копировании формулы адреса, используемые в формуле, по-разному ведут себя в зависимости от их типа. Абсолютные адреса при копировании не изменяются. Относительные адреса изменяются так, что при новом местоположении формулы адрес указывает новое местоположение адресуемой ячейки. В случае смешанных адресов, если символ **$** стоит перед номером строки (**F$7**), то при копировании не изменяется номер строки, если перед номером столбца (**$F7**) – не изменяется номер столбца.

Правила изменения адресов ячеек при копировании формулы иллюстрирует пример на рис. 3.11. В ячейке **E1** таблицы находится формула **=A1+$B$1+C$1+$D1.** Поскольку в исходной формуле ячейка **A1** является четвертой слева ячейкой относительно ячейки с формулой, при копировании формулы в другие ячейки адрес **A1** изменяется, всякий раз указывая на четвертую слева ячейку от ячейки, содержащей формулу. Адрес **$B$1** не изменяется. В адресах **C$1** и **$D1** изменяется та часть адреса, которая не отмечена символом **$**.

E1 = =A1+$B$1+C$1+$D1

	A	B	C	D	E	F	G
1	1	2	3	1	=A1+$B$1+C$1+$D1	=B1+$B$1+D$1+$D1	=C1+$B$1+E$1+$D1
2	2	3	4	1	=A2+$B$1+C$1+$D2		
3	3	4	5	1	=A3+$B$1+C$1+$D3		

Рис. 3.11. Изменение адресов при копировании формулы

Внешняя ссылка может ссылаться на ячейку с другого рабочего листа той же рабочей книги или любой другой рабочей книги (например: **Лист2!B3**).

☑ Инструмент **Автозаполнение** можно использовать не только для ввода числовых и текстовых значений, но и для ввода формул. В исходной ячейке, которая используется для автозаполнения и в которую формула вводится вруч-

ную, важно правильно использовать абсолютные и относительные адреса ячеек. Если адрес не должен изменяться при копировании формулы, то его следует сделать абсолютным; если адрес должен изменяться, то его следует сделать относительным.

Автосуммирование

Операция ***Автосуммирование*** позволяет получить в текущей ячейке сумму чисел из заданного диапазона ячеек. Для выполнения автосуммирования следует выделить ячейку, в которой должна быть получена сумма, и выбрать инструмент **Автосумма**  на панели инструментов **Стандартная**. Затем указать мышью диапазон суммируемых ячеек и нажать клавишу **<Enter>**.

Стандартные функции

Стандартная функция в MS Excel – это разработанная для типовых операций программа с уникальным именем, для которой пользователь должен задать конкретные значения аргументов. В MS Excel есть специальное средство для работы с функциями – это **Мастер функций,** который запускается с помощью инструмента  , расположенного в строке редактирования, или командами меню **Вставка** – **Функция**. При работе с **Мастером функций** сначала следует выбрать категорию и функцию из списка. На следующем шаге нужно ввести аргументы функции.

Статистические и логические функции

Некоторые наиболее часто используемые функции из категории **Статистические** приведены в табл. 3.1.

Таблица 3.1

Статистические функции MS Excel

Название функции	Результат применения функции	Пример использования
СРЗНАЧ	Среднее арифметическое значение для чисел из указанного диапазона	**=СРЗНАЧ(B7:B10)**
МИН	Наименьшее значение из указанного диапазона	**=МИН(A3:A7)**

Окончание табл. 3.1

Название функции	Результат применения функции	Пример использования
МАКС	Наибольшее значение из указанного диапазона	**=МАКС(А3:А7)**
СЧЁТЕСЛИ	Количество ячеек в указанном диапазоне, удовлетворяющих заданному условию	**=СЧЁТЕСЛИ(А1:А10; “январь”)**

Результатом вычисления логической функции является одно из двух возможных значений: **Истина (True)** или **Ложь (False)**. К категории логических функций в MS Excel относятся функции **И**, **ИЛИ**, **ЕСЛИ**. Аргументами этих функций обычно являются ***условные выражения***. Примеры таких выражений приведены в табл. 3.2.

Таблица 3.2

Примеры вычисления значений логических выражений

Условное выражение	Значение выражения
A2 > 1	Значение логического выражения равно **Истина**, если в ячейке **А2** находится число, **большее 1**. В противном случае значение равно **Ложь**
A1 = A2	Значение логического выражения равно **Истина**, если в ячейке **А1** находится то же значение, что и в ячейке **А2**

Функция **И (логическое выражение1; логическое выражение2; …)** принимает значение **Истина**, если все выражения в скобках имеют значение **Истина**. Например, функция **И (А2>50; В2>20)** будет иметь значение **Истина**, если оба условия в скобках выполняются. Если хотя бы одно условие не выполняется, то функция имеет значение **Ложь**.

Функция **ИЛИ (логическое выражение1; логическое выражение2; …)** принимает значение **Истина**, если хотя бы одно выражение в скобках имеет значение **Истина**. Например, функция **ИЛИ (А2>50; В2>20)** будет иметь значение **Истина**,

если выполнено хотя бы одно из условий: **A2>50** или **B2>50**. Функция имеет значение **Ложь** только в одном случае, когда оба условия не выполняются.

Функция **ЕСЛИ (логическое выражение; значение_1; значение_2)** принимает значение **значение_1**, если **логическое выражение** равно **Истина**, принимает значение **значение_2**, если **логическое выражение** равно **Ложь.** Примеры использования функции **ЕСЛИ** приведены в табл. 3.3.

Таблица 3.3

Примеры использования функции ЕСЛИ

Число в ячейке **A1**	Число в ячейке **A2**	Формула в ячейке **A3**	Результат в ячейке **A3**
20	10	**=ЕСЛИ (A1>A2; A1+A2; A1)**	30
Комментарий. Число в ячейке **A1** больше числа в ячейке **A2**, поэтому логическое выражение **A1>A2** равно **Истина**. В этом случае результат функции **ЕСЛИ** равен **A1+A2**, т. е. равен сумме чисел в ячейках **A1** и **A2**, равен **30**			
10	20	**=ЕСЛИ (A1>A2; A1+A2; A1)**	10
Комментарий. Число в ячейке **A1** меньше числа в ячейке **A2**, поэтому логическое выражение **A1>A2** равно **Ложь**. В этом случае результат функции **ЕСЛИ** равен **A1**, т. е. равен числу в ячейке **A1**, равен **10**			
20	10	**=ЕСЛИ (ИЛИ (A1>A2; A2>0); "красный"; "синий")**	Слово «красный»
Комментарий. Результат функции **ЕСЛИ** зависит от результата вычисления функции **ИЛИ**. Функция или возвращает значение **ИСТИНА**, если хотя бы одно из выражений в скобках **(A1>A2; A2>0)** равно **ИСТИНА**. В нашем случае выполняются оба условия, следовательно, результат функции **ИЛИ** равен **ИСТИНА**. Тогда результат функции **ЕСЛИ** равен **"красный" (**в ячейке **A3** мы увидим текст **"красный"**)			
10	20	**=ЕСЛИ (И (A1>A2; A2>0); "красный"; "синий")**	Слово «синий»
Комментарий. Функция **И** возвращает значение **ИСТИНА** в случае, когда оба условия в скобках **(A1>A2; A2>0)** равны **ИСТИНА**. В нашем случае условие **A1>A2** равно **ЛОЖЬ**, следовательно, значение функции **И** равно **ЛОЖЬ**. Тогда результат функции **ЕСЛИ** равен **"синий" (**в ячейке **A3** мы увидим текст **"синий"**)			

Окончание табл. 3.3

Число в ячейке **A1**	Число в ячейке **A2**	Формула в ячейке **A3**	Результат в ячейке **A3**
20	10	**=ЕСЛИ (И (A1>A2; A2>0); A1+A2; ”красный”)**	30
10	20	**=ЕСЛИ (И (A1>A2; A2>0); A1+A2; ”красный”)**	Слово «красный»
20	10	**=ЕСЛИ (ИЛИ (A1>A2; A2>0); A1+A2; A1)**	30
10	20	**=ЕСЛИ (ИЛИ(A1>A2; A2>0); A1+A2; A1)**	30
-2	-1	**=ЕСЛИ (ИЛИ(A1>A2; A2>0); A1+A2; A1)**	-2

Построение диаграмм

Диаграмма – это форма графического представления данных. При создании диаграммы используются выделенные заранее ячейки с данными, которые затем отображаются в виде полос, линий, столбиков, секторов, точек и т. д. Их называют ***маркерами данных***. Группы элементов данных или их маркеров, отображающие содержимое одной строки или одного столбца таблицы, составляют ***ряд данных***. Каждый ряд на диаграмме выделяется уникальным цветом или узором. Расшифровка цветов диаграммы называется ***легендой*** (рис. 3.12). В табл. 3.4 показаны основные типы диаграмм и указано их назначение.

Для создания диаграмм используется специальная программа **Мастер диаграмм,** которая вызывается с помощью инструмента **Мастер диаграмм**  на панели инструментов **Стандартная** или командами меню **Вставка** – **Диаграмма**.

☑ Построенная диаграмма может быть изменена путем добавления или удаления данных, изменения цвета, узора, способа заливки и т. д. Чтобы провести форматирование отдельного элемента диаграммы, надо выполнить двойной щелчок по нему и в появившемся окне произвести необходимые изменения. Изменить тип диаграммы можно, если щелкнуть

правой кнопкой мыши в области диаграммы и в появившемся контекстном меню выбрать пункт **Тип диаграммы**.

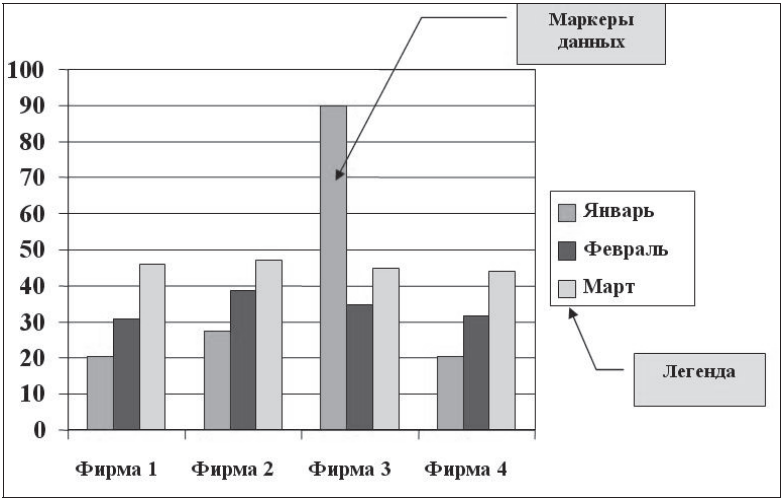
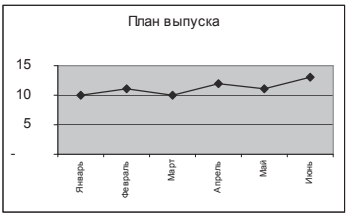
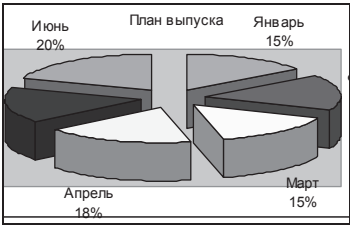
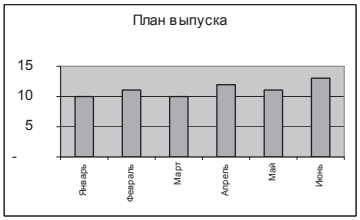


Рис. 3.12. Пример диаграммы

Таблица 3.4

Основные типы диаграмм

Вид диаграммы	Тип и назначение диаграммы
	***График*** – отражает тенденции изменения данных за равные промежутки времени
	***Круговая диаграмма*** – показывает долю каждого элемента ряда данных в общей сумме

Вид диаграммы	Тип и назначение диаграммы
	***Гистограмма*** **–** иллюстрирует соотношение отдельных значений данных

3.4.3. Представление и обработка данных с помощью списков

В программе Microsoft Excel ***списком*** называется набор из последовательно расположенных строк рабочего листа электронной таблицы, содержащих однотипные данные. Примером списка может служить телефонный справочник, прайс-лист, перечень клиентов фирмы и т. д. На рис. 3.13 показан пример списка.

	A	B	C	D	E	F
1	***Дата продажи***	***Покупатель (фирма)***	***Товар***	***Единица измерения***	***Цена за единицу***	***Количество***
2	02.08.09	Веста	Кофе молотый	кг	300,00р.	10
3	02.08.09	Арго	Кофе молотый	кг	400,00р.	10
4	02.08.09	Малахит	Перец чёрный	уп (200 г)	35,00р.	30
5	02.08.09	Эльбрус	Перец чёрный	уп (200 г)	35,00р.	20
6	02.08.09	Рондо	Шоколад	кг	300,00р.	60

Рис. 3.13. Пример списка на рабочем листе Microsoft Excel

Сортировка списка

Изменение порядка строк в списке в соответствии с некоторыми критериями называется ***сортировкой списка***. Данные в списке можно упорядочить в алфавитном, числовом или хронологическом порядке по одному, двум или трем полям списка. Для выполнения операции сортировки необходимо выделить любую ячейку в списке и выбрать в меню команду **Данные**, затем – **Сортировка**. Для выполнения сортировки по одному столбцу можно использовать инструменты **Сортировка по возрастанию** и **Сортировка по убыванию**  , расположенные на панели инструментов **Форматирование**.

Фильтрация списка

Для отображения в списке части данных, удовлетворяющих некоторым условиям, используется операция ***фильтрации списка***. Для установки фильтра нужно в меню программы выполнить команду **Данные – Фильтр**. В подменю команды можно выбрать отбор записей при помощи ***Автофильтра*** или ***Расширенного фильтра***.

При выборе команды **Автофильтр** в заголовке каждого столбца появится кнопка раскрывающегося списка (рис. 3.14), в котором перечислены все возможные значения, содержащиеся в данном столбце. После выбора элемента из данного списка MS Excel оставит на рабочем листе записи, содержащие указанное значение в данном столбце. Остальные записи будут временно скрыты. Критерий **Условие** позволяет определить условия отбора записей.

	A	B	C	D	E
1	*Дата продажи*	*Покупатель (фирма)*	*Товар*	*Единица измерени*	*Цена за единицу*
2	10.08.09	Арго	(Все)	кг	50,00р.
3	13.08.09	Арго	(Первые 10...)	кг	50,00р.
4	03.08.09	Арго	(Условие...)	кг	60,00р.
5	02.08.09	Арго	Корица Кофе молотый Перец чёрный Сахар	кг	400,00р.

Рис. 3.14. Вид таблицы с активированной опцией **Автофильтр**

Подведение промежуточных итогов в списке

Для подведения итогов могут быть использованы различные операции, например вычисление суммы, наибольшего или наименьшего значения в группе. Для расчета промежуточных итогов нужно выполнить в меню команды **Данные – Итоги**. В диалоговом окне **Промежуточные итоги** следует установить параметры, определяющие процедуру подведения итогов:

- в раскрывающемся списке **При каждом изменении в** выбирается столбец, по которому проводится группировка строк списка;
- в раскрывающемся списке **Операция** выбирается функция, которая будет использоваться для вычисления итогов (сумма, среднее, количество, максимум, минимум, произведение);

• в списке **Добавить итоги по** отмечаются поля, для которых будут вычисляться итоги по каждой группе.

В результате подведения итогов структура таблицы изменяется. В конце каждой группы строк, имеющих одно и то же значение в поле, по которому была выполнена сортировка списка, добавляется новая строка, содержащая итоговые данные.

Создание сводных таблиц

Сводная таблица представляет собой интерактивную таблицу, с помощью которой можно быстро объединять и сравнивать большие объемы данных, структурированных в виде списков. Запуск мастера сводных таблиц осуществляется командами меню **Данные – Сводная таблица**. На первом шаге определяется источник данных, по которому создается итоговая таблица. Если в качестве источника данных выбран список MS Excel, то на втором шаге определяется диапазон ячеек, содержащий данные.

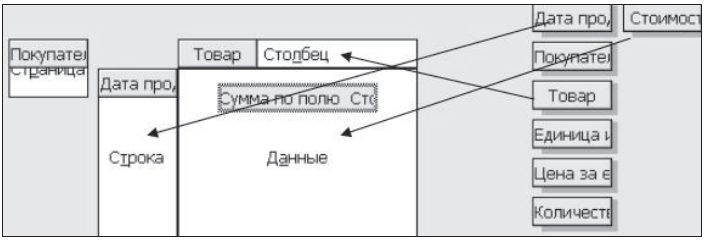


Рис. 3.15. Формирование макета сводной таблицы

На третьем шаге следует указать, где будет размещена сводная таблица (на новом или на существующем листе рабочей книги). Щелчком мыши по кнопке **Макет** открывается диалоговое окно, в котором создается макет сводной таблицы (рис. 3.15). Таблица формируется перетаскиванием с помощью мыши кнопок из правой части окна в нужную область макета таблицы.

? Вопросы и упражнения для самоконтроля

1. Перечислите типы данных, которые можно использовать в электронной таблице.
2. Как осуществить заполнение диапазона ячеек последовательностью чисел?

3. Назовите и охарактеризуйте типы адресов ячеек, которые могут быть использованы в формулах.

4. Перечислите основные элементы диаграммы. Как можно изменить тип созданной диаграммы?

5. Какие операции можно выполнять с листами рабочей книги Excel?

6. Какие ссылки на ячейки в формулах называются внешними ссылками? Опишите последовательность вставки в формулу внешней ссылки.

7. Опишите действие логических функций **И**, **ИЛИ**, **ЕСЛИ**.

8. Чему равно значение функции **И(A1>1;A1<10)** при **A1=5**?

9. Каким должно быть содержимое ячейки **A1,** для того чтобы значение функции **И(A1>0;A1<5)** было равно **TRUE**?

10. Определите результат вычисления по формуле **=ЕСЛИ(ИЛИ(A1<=1;A1>=5);100;200)** при **A1=6**.

11. Какая операция с записями списка называется сортировкой списка?

12. Какая операция с записями списка называется фильтрацией списка?

13. Объясните смысл операции подведения промежуточных итогов в списке.

14. Какие области содержит макет сводной таблицы?

3.5. Создание презентаций в Microsoft PowerPoint

Документ Microsoft PowerPoint – это ***презентация***. Он состоит из отдельных ***слайдов***. На экран их можно выводить в разных режимах, которые имеют различное назначение:

- **обычный** – создание структуры презентации, отдельных слайдов и их наполнения;
- **сортировщик слайдов –** обозначение порядка следования слайдов, настройка свойств презентации для демонстрации;
- **показ слайдов –** демонстрация слайдов в установленном порядке;
- **страницы заметок –** дополнение слайдов необходимыми докладчику заметками (тезисы выступления или

сведения, не вошедшие в слайд, но необходимые для представления зрителям).

Способы создания презентации в MS PowerPoint

MS PowerPoint предлагает несколько способов создания презентаций. Выбор способа осуществляется при помощи панели задач **Создание презентации**. На основе шаблона **Новая презентация** создается пустая презентация, формат которой полностью зависит от желаний пользователя. Автоматически пустая презентация создается при нажатии кнопки **Создать** на панели инструментов **Стандартная**.

При выборе опции **Из шаблона оформления** открывается область задач **Дизайн слайда**, где можно выбрать один из стандартных вариантов оформления слайдов презентации. Способ **Из мастера автосодержания** запускает **Мастер автосодержания**, посредством которого пользователь может выбрать и задать некоторые параметры самой презентации и способ ее вывода на экран. В качестве шаблона можно использовать созданную ранее презентацию. Для этого выбирайте вариант **Из имеющейся презентации**.

MS PowerPoint предоставляет пользователю комплект стандартных шаблонов оформления презентации. Шаблон задает для всех слайдов презентации общее решение, единый дизайн. Его можно выбрать при создании новой презентации или на любом этапе ее разработки.

Для ***выбора шаблона оформления*** выполните следующие действия:

- вызовите на экран область задач **Дизайн слайда**, где можно увидеть все доступные шаблоны;
- если шаблон будет применяться к отдельным слайдам презентации, их нужно выделить на структуре презентации, затем навести курсор на иконку шаблона и, щёлкнув по ней, выбрать в открывающемся меню вариант применения шаблона.

Редактирование слайдов в MS PowerPoint

С объектами слайда можно выполнять различные действия, общие для всех программ пакета MS Office (выделение, перемещение, копирование, изменение параметров форматирования и т. д.). В большинстве диалоговых окон

MS PowerPoint присутствует кнопка **Просмотр**, позволяющая сразу же оценить результаты произведенных изменений. Для перемещения, копирования и других действий по изменению порядка следования слайдов в презентации лучше всего использовать режим **Сортировщик слайдов**. По умолчанию в данном режиме отражаются миниатюры слайдов презентации MS PowerPoint.

Для перемещения или копирования слайдов выберите и выделите слайд или группу слайдов, наведите мышь на выделенную группу, нажмите левую клавишу мыши и перетяните слайды в желаемую позицию.

Настройка и показ презентации

MS PowerPoint позволяет создавать ***управляемые*** и ***автоматические*** презентации. Для автоматической демонстрации при создании файла следует установить время показа отдельных слайдов. Для большей наглядности в презентациях MS PowerPoint используются многочисленные эффекты и схемы анимации, а также звуковые эффекты. Однако излишние эффекты могут ухудшить восприятие презентации слушателями, поэтому следует ограничиться минимальным их набором. Возможна настройка анимации и для отдельных объектов слайда. Для этого используется меню **Показ слайдов – Настройка анимации**.

Используя меню **Показ слайдов – Настройка презентации**, можно ***задать параметры демонстрации слайдов***. Здесь нужно указать, как происходит управление показом; отметить слайды, которые будут выводиться на экран; выбрать способ смены слайдов.

Для ***начала показа*** следует воспользоваться клавишей **F5** или меню **Показ слайдов – Начать показ.** Управлять показом в управляемом режиме можно при помощи мыши или клавиш **<Пробел>, <Enter>, <↑>, <↓>, <Home>, <End>, <Page Up>, <Page Down>**. Завершить показ слайдов можно в любой момент, открыв контекстное меню и выбрав команду **Завершить показ слайдов**.

Сохранение презентации

Существуют различные форматы сохранения презентации в зависимости от цели, с которой делается сохранение. Самые распространенные варианты:

• сохранение с возможностью модификации (формат **.ppt**);

• сохранение для показа в формате демонстрации (формат **.pps**);

• сохранение презентации для публикации в Интернет (формат **.html** или **.mht**).

При запуске просмотра презентации в формате **.pps** рабочее окно программы MS PowerPoint не открывается, а сразу начинается показ слайдов.

? Вопросы и упражнения для самоконтроля

1. Каково назначение программы MS PowerPoint?
2. Что представляет собой документ MS PowerPoint?
3. В каких режимах возможна работа с программой MS PowerPoint?
4. Назовите способы создания новой презентации. В каких случаях удобнее использовать вариант **Из имеющейся презентации** для создания новой презентации?
5. Как применить к выделенным слайдам выбранный шаблон оформления слайдов?
6. Какие варианты анимации можно настроить для отдельных объектов на слайде?
7. Каково назначение режима просмотра **Сортировщик слайдов**?
8. Назовите способы запуска показа презентации. Какие клавиши используются для управления показом презентации?
9. Опишите форматы сохранения презентации.

3.6. Практические работы

Практическая работа 3.1. Форматирование документа MS Word. Представление текста в документе MS Word в формате списков и таблиц

Цель работы – научиться:

• вводить с клавиатуры, удалять, копировать, перемещать фрагменты текста;

• вставлять в текст специальные символы, отсутствующие на клавиатуре;

• устанавливать параметры страницы в документе, формировать колонтитулы;
• устанавливать параметры абзаца и шрифта.
• формировать маркированные и нумерованные списки;
• создавать в документе таблицы, выполнять редактирование и форматирование таблиц;
• использовать инструменты панели **Таблицы и границы** для редактирования и форматирования таблиц.

Порядок выполнения работы

1. Выполните упражнения 1, 2, 3.

Упражнение 1

1. Создайте новый документ. Сохраните документ с именем **Текст1.doc**.

2. Установите параметры страницы (**Файл – Параметры страницы**): все поля – **2 см**; ориентация страницы – **книжная**.

3. Выполните в меню **Вид – Колонтитулы**. Откроется верхний колонтитул, и появится панель инструментов **Колонтитулы** (рис. 3.16). Введите в верхний колонтитул свою фамилию и инициалы.

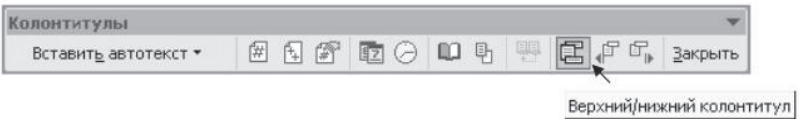


Рис. 3.16. Панель инструментов **Колонтитулы**

4. Используя инструмент **Верхний/нижний колонтитул** на панели инструментов **Колонтитулы**, переключитесь в нижний колонтитул.

5. Введите в нижний колонтитул текст «**Дата создания документа**» и щелкните кнопку **Дата** на панели инструментов **Колонтитулы**. В колонтитул будет вставлена текущая дата. Закройте колонтитулы с помощью инструмента **Закрыть** на панели **Колонтитулы**.

6. Введите с клавиатуры следующий текст как один абзац без рамки, клавишу **<Enter>** нажмите один раз в конце текста.

Во время ввода текста программа производит автоматическую проверку орфографии и синтаксиса, если соответствующие опции включены на вкладке «Правописание» (открывается командами меню «Сервис» – «Параметры»). Слова, отсутствующие в словаре (с которым осуществляется сверка вводимых слов) или введенные с ошибкой, выделяются на экране красной волнистой линией. При неправильно расставленных знаках пунктуации или неправильно построенной фразе фрагменты текста выделяются волнистой линией зеленого цвета.

7. Разбейте текст на 3 абзаца, нажав клавишу **<Enter>** в конце первого и в конце второго предложения.

8. Установите параметры шрифта и абзаца для каждого из получившихся абзацев так, как указано в табл. 3.5.

Таблица 3.5

Параметры форматирования текста

Номер абзаца	Параметры шрифта	Параметры абзаца
1	Стиль – **Обычный** Шрифт – **Times New Roman** Размер шрифта (кегль) – **14 пт** Начертание – *курсивный* Цвет шрифта – **темно-красный**	Отступы: **слева – 0, справа – 0, первая строка – 2 см.** Выравнивание – **по центру** Междустрочный интервал – **полуторный**
2	Стиль – **Обычный** Шрифт – **Verdana** Размер шрифта (кегль) – **12 пт** Начертание – **полужирный** Цвет шрифта – **черный**	Отступы: **слева – 2 см, справа – 0, первая строка – 0.** Выравнивание – **по ширине** Междустрочный интервал – **одинарный** Внешние границы – **все границы**
3	Стиль – **Заголовок 2** Шрифт – ***Courier New*** Размер шрифта (кегль) – **12 пт** Начертание – ***полужирный курсив*** Цвет шрифта – **синий**	Отступы: **слева – 0, справа – 2 см, первая строка – 0.** Выравнивание – **по правому краю** Междустрочный интервал – **двойной**

Установку параметров шрифта можно выполнить в диалоговом окне **Шрифт** (**Формат – Шрифт**), установку параметров абзаца – в диалоговом окне **Абзац** (**Формат – Абзац**). Можно также использовать инструменты панели **Форматирование**. Отформатированный текст представлен на рис. 3.17.

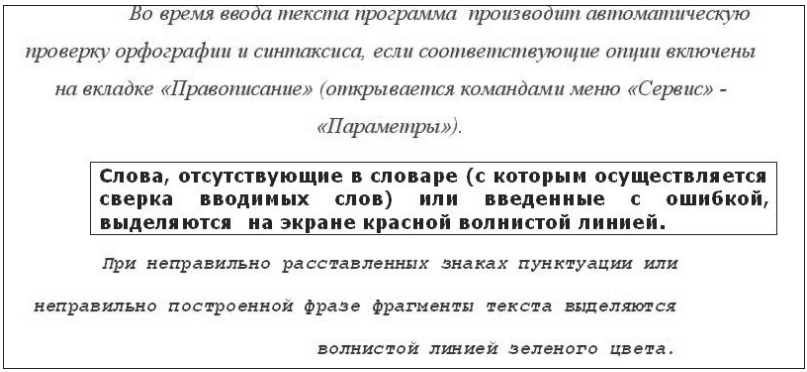


Рис. 3.17. Результат выполнения упражнения 1

9. Сохраните документ.

Упражнение 2

1. Создайте новый документ. Сохраните документ с именем **Текст2.doc**.

2. Установите *параметры страницы* документа: все поля – **2,5 см**, ориентация страницы – **книжная**.

3. Введите с клавиатуры следующий текст как один абзац без рамки, клавишу **<Enter>** нажмите **1 раз в конце текста**.

> На практике обычно используются следующие единицы измерения информации: байт, килобайт, мегабайт, гигабайт.

4. Установите следующие *параметры абзаца*: отступы: слева – **0 см**, справа **– 0 см**, первая строка **– нет**; выравнивание – **по ширине**; междустрочный интервал – **одинарный**.

5. Установите следующие *параметры шрифта*: шрифт – **Times New Roman**; размер – **14 пт**; начертание – **обычный**; цвет – **черный**.

6. Установив курсор ввода текста перед словом «**байт**», нажмите клавишу **<Enter>**. Аналогичную процедуру выполните для слов «килобайт», «мегабайт», «гигабайт». Теперь текст состоит из 5 абзацев (рис. 3.18).

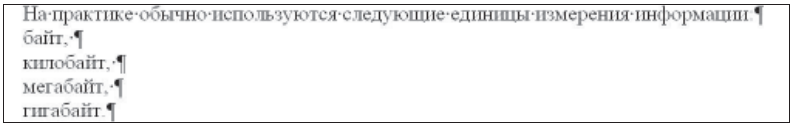


Рис. 3.18. Вид введенного текста
в режиме просмотра непечатаемых знаков

7. Выделите абзацы с перечислением единиц измерения информации и нажмите на панели инструментов **Форматирование** кнопку **Маркеры**. В результате абзацы будут преобразованы в *маркированный список*, вид которого показан на рис. 3.19 (маркеры могут отличаться).

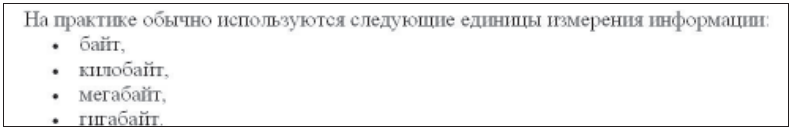


Рис. 3.19. Текст с маркированным списком

8. Скопируйте этот текст и вставьте в документ 5 раз. Должно получиться 6 одинаковых текстов с маркированными списками.

9. Выполните изменение формата маркированных списков в текстах 1–3, выполняя в меню **Формат** команды **Список – Маркированный – Изменить – Знак**. Для выбора маркера используйте шрифтовой набор **Wingdings**. В результате должен получиться текст, показанный на рис. 3.20.

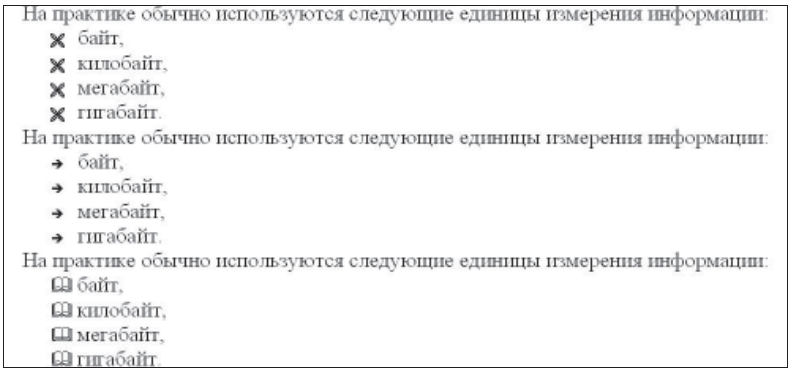


Рис. 3.20. Результат выполнения пункта 9 упражнения 2

10. Измените маркированные списки в текстах 4 – 6 на *нумерованные списки*. Для этого выделите каждый список и нажмите на панели **Форматирование** кнопку **Нумерация**.

11. Выполняя в меню **Формат** команды **Список – Нумерованный – Изменить**, измените формат нумерованных списков (рис. 3.21).

На практике обычно используются следующие единицы измерения информации:
a) байт,
b) килобайт,
c) мегабайт,
d) гигабайт.
На практике обычно используются следующие единицы измерения информации:
*1*байт,
*2*килобайт,
*3*мегабайт,
*4*гигабайт.
На практике обычно используются следующие единицы измерения информации:
I. байт,
II. килобайт,
III. мегабайт,
IV. гигабайт.

Рис. 3.21. Результат выполнения пункта 11 упражнения 2

12. Для изменения формата первого списка выделите его и выполните в меню **Формат – Список – Нумерованный – Изменить**. В открывшемся окне выберите нумерацию строчными латинскими буквами, в поле **Формат номера** введите с клавиатуры символ «**закрывающая скобка**» после номера (рис. 3.22). Аналогично измените формат двух других списков.

13. Сохраните документ.

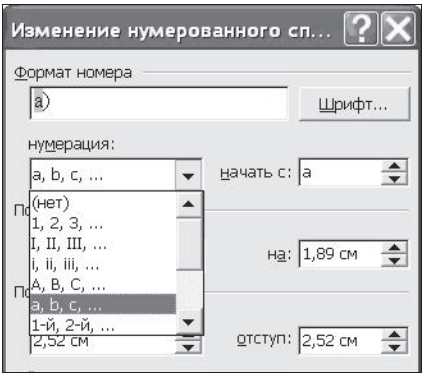


Рис. 3.22. Изменение формата нумерованного списка

Упражнение 3

1. Создайте новый документ. Сохраните документ с именем **Текст3.doc**.

2. Установите параметры страницы документа: все поля – **2 см**, ориентация страницы – **книжная**.

3. Вставьте в документ таблицу, состоящую из 4 столбцов и 5 строк. Для этого выполните в меню **Таблица – Вставить – Таблица**, задайте в диалоговом окне число столбцов и число строк.

4. Измените ширину столбцов таблицы, переместив границы столбцов с помощью мыши в нужном направлении.

5. Введите в ячейки таблицы текст, показанный на рис. 3.23.

Клавиша	Назначение	Клавиша	Назначение
F1	Помощь	**F5**	Копирование файла или каталога
F2	Вызов меню пользователя	**F6**	Переименование файла или каталога
F3	Просмотр файла	**F7**	Создание каталога
F4	Редактирование файла	**F8**	Удаление файла или каталога

Рис. 3.23. Исходная таблица

6. Скопируйте таблицу и вставьте копию в документ дважды, не забывая нажимать перед вставкой таблицы клавишу **<Enter>** для вставки пустой строки между таблицами.

7. В *первой копии* таблицы *удалите третий и четвертый столбцы*. Для этого выделите эти столбцы и выполните в меню **Таблица – Удалить – Столбцы**. Вставьте перед первым столбцом таблицы *новый столбец*. Для этого выделите первый столбец и выполните в меню **Таблица – Вставить – Столбцы слева**. В ячейки нового столбца введите текст (рис. 3.24).

8. Во второй копии таблицы *вставьте новую строку перед первой строкой*: выделите первую строку, выполните в меню **Таблица – Вставить – Строки выше**. Объедините ячейки в первой строке: выделите первую строку, выполните в меню **Таблица – Объединить ячейки**

9. В третьей копии таблицы *вставьте новую строку в конец таблицы*, для этого установите курсор в крайнюю правую ячейку последней строки и нажмите на клавиатуре клавишу **<Tab>**. Объедините в последней строке первую ячейку со второй, третью ячейку с четвертой.

№	Клавиша	Назначение
1	F1	Помощь
2	F2	Вызов меню пользователя
3	F3	Просмотр файла
4	F4	Редактирование файла

Функциональные клавиши			
Клавиша	Назначение	Клавиша	Назначение
F1	Помощь	F5	Копирование файла или каталога
F2	Вызов меню пользователя	F6	Переименование файла или каталога
F3	Просмотр файла	F7	Создание каталога
F4	Редактирование файла	F8	Удаление файла или каталога

Клавиша	Назначение	Клавиша	Назначение
F1	Помощь	F5	Копирование файла или каталога
F2	Вызов меню пользователя	F6	Переименование файла или каталога
F3	Просмотр файла	F7	Создание каталога
F4	Редактирование файла	F8	Удаление файла или каталога

Рис. 3.24. Результат выполнения действий, заданных пунктами 7–9 упражнения 3

10. Выполните форматирование таблиц (рис. 3.25, 3.26), используя инструменты панели **Таблицы и границы**. Панель выводится на экран командами меню **Вид – Панели инструментов – Таблицы и границы**.

№	Клавиша	Назначение
1	F1	Помощь
2	F2	Вызов меню пользователя
3	F3	Просмотр файла
4	F4	Редактирование файла

Рис. 3.25. Таблица 1

Клавиша	Назначение
F1	Помощь
F2	Вызов меню пользователя
F3	Просмотр файла
F4	Редактирование файла

Рис. 3.26. Таблица 2

Таблица 1 (рис. 3.25): выберите инструмент **Нарисовать таблицу**, установите тип линии **двойная линия**, обведите мышкой внешние границы таблицы. В **таблице 2** (рис. 3.26)

последовательно выполните заливку ячеек цветом: выделите ячейку, выберите инструмент **Цвет заливки**, выберите цвет.

11. Сохраните документ.

Практическая работа 3.2. Создание и редактирование графических объектов в документе MS Word. Вставка в документ формул. Структурирование документа

Цель работы – научиться:

- вставлять в текст рисунки из коллекции рисунков **Microsoft Clipart**, выполнять настройку параметров рисунка;
- вставлять в текст формулы, выполненные средствами редактора формул **Microsoft Equation 3.0**;
- выполнять поиск и замену текста в документе;
- вставлять в документ сноски, оглавление, закладки, гиперссылки.

Порядок выполнения работы

1. Выполните упражнения 4, 5, 6.

Упражнение 4

1. Создайте новый документ. Сохраните документ с именем **Текст4.doc**.

2. Введите с клавиатуры следующий текст как один абзац без рамки.

> Можно провести редактирование вставленного в документ рисунка, т. е. изменить цвета, яркость и заливку, размер, способ обтекания рисунка текстом, положение, сделать обрезку и т. д. Для этого нужно выделить рисунок (щелчком левой кнопки) и, открыв щелчком правой кнопки мыши контекстное меню, выбрать в нем (щелчком левой кнопки мыши) команду **Формат рисунка**.

3. Установите выравнивание абзаца по ширине, остальные параметры абзаца и шрифта выберите на свое усмотрение.

4. Вставьте внутрь абзаца произвольный рисунок из подборки **Microsoft Clipart**. Для этого выполните в меню **Вставка – Рисунок – Картинки**. Откроется панель задач **Вставка картинки**. В нижней части области задач щелкните ссылку **Коллекция**

картинок. В открывшемся окне выберите **Коллекция Microsoft Office**, затем выберите категорию, выберите рисунок и вставьте его в текст (рис. 3.27).

Можно провести редактирование вставленного в документ рисунка, т.е. изменить цвета, яркость и заливку, размер, способ обтекания рисунка текстом, положение, сделать обрезку и

т.д. Для этого нужно выделить рисунок (щелчком левой кнопки) и, открыв щелчком правой кнопки мыши контекстное меню, выбрать в нем (щелчком левой кнопки мыши) команду **Формат рисунка**.

Рис. 3.27. Абзац текста с рисунком из коллекции **Microsoft Clipart**

5. Выделите рисунок одиночным щелчком левой кнопки мыши, установите курсор на любой из угловых маркеров (черный квадратик на контуре рисунка) и переместите маркер так, чтобы *размер рисунка уменьшился*.

6. Выделите текст с рисунком, скопируйте и вставьте копию текста в документ дважды. В результате у вас получится 3 одинаковых абзаца с рисунками.

7. Выделите рисунок в первом абзаце. На экране появится панель инструментов **Настройка изображения** (рис. 3.28). Если панель не появилась, выполните в меню **Вид – Панели инструментов** и включите панель.

Рис. 3.28. Панель инструментов **Настройка изображения**

8. С помощью инструмента **Обтекание текстом** откройте диалоговое окно и на вкладке **Положение** установите тип обтекания **Вокруг рамки**. С помощью инструмента **Тип линии** сделайте рамку вокруг рисунка.

9. Выделите рисунок во втором абзаце. Выполните в меню **Формат – Рисунок**. В диалоговом окне **Формат рисунка** на вкладке **Размер** установите размер рисунка по высоте 2 см. На вкладке **Положение** выберите обтекание **по контуру**.

10. Выделите рисунок в третьем абзаце. Выполните обрезку рисунка, скрыв часть рисунка. Используйте инстру-

мент **Обрезка** на панели **Настройка изображения**. Сделайте рамку вокруг рисунка двойной линией. Результат выполнения операций форматирования показан на рис. 3.29.

11. Сохраните документ.

Рис. 3.29. Результат выполнения упражнения 4

Упражнение 5

1. Создайте новый документ. Сохраните документ с именем **Текст5.doc**.

2. Используя *редактор формул*, введите формулу $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$.

☑ Перед началом ввода формулы рекомендуется установить масштаб просмотра документа в пределах 150–200 процентов.

Установив курсор в документе на место вставки формулы, выполните в меню **Вставка – Объект… – Microsoft Equation 3.0**. Откроется окно ввода формулы и появится панель инструментов **Формула**.

Введите с клавиатуры символ «***x***», щелчком по инструменту **Надстрочные знаки** откройте набор знаков и выберите первый слева знак в четвертом сверху горизонтальном ряду (рис. 3.30).

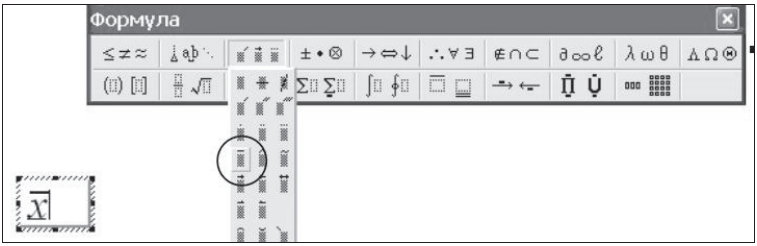


Рис. 3.30. Начало ввода формулы

Нажмите на клавиатуре клавишу с символом «=». Щелчком по инструменту **Шаблоны дробей и радикалов** откройте набор шаблонов и выберите первый слева шаблон в верхнем ряду. Установите курсор в числитель дроби и введите с клавиатуры символ «***1***», в знаменатель – символ «***n***». Переместите курсор в крайнюю правую позицию и с помощью инструмента **Шаблоны сумм** вставьте шаблон суммы (рис. 3.31).

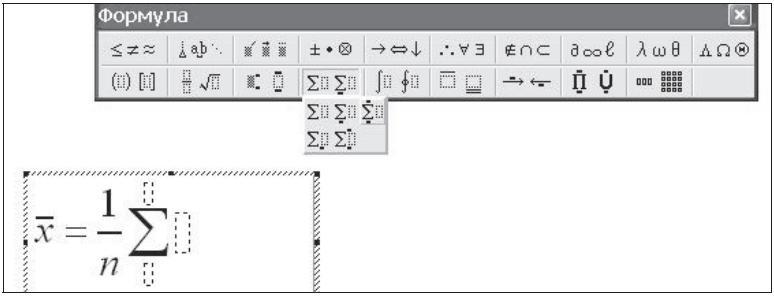


Рис. 3.31. Вставка в формулу шаблона для ввода суммы

В поле нижнего предела суммы введите с клавиатуры «***i = 1***». В поле верхнего предела суммы – символ «***n***». В поле слагаемых введите символ «***x***». Щелчком по инструменту **Шаблоны верхних и нижних индексов** выберите второй слева шаблон в верхнем ряду (рис. 3.32). Введите с клавиатуры символ «***i***». Завершите ввод формулы и перейдите в режим редактирования документа щелчком вне окна редактирования формулы.

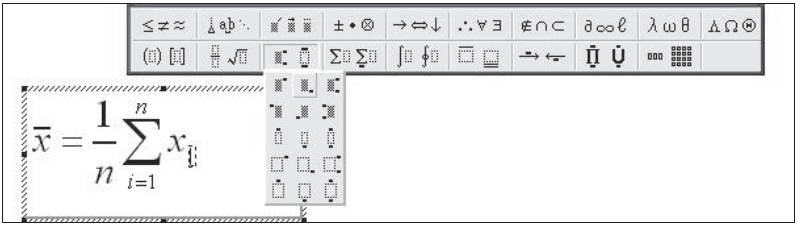


Рис. 3.32. Ввод в формулу нижнего индекса

3. Используя редактор формул, введите формулу

$$F(x) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{1}{2}\left(\frac{x-a}{\sigma}\right)^2} dx.$$

При вводе используются инструменты: **Шаблоны дробей и радикалов, Шаблоны интегралов, Шаблоны верхних и нижних индексов, Шаблоны скобок, Греческие буквы**.

4. Используя редактор формул, введите формулу

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix} = \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{pmatrix}.$$

При вводе используются инструменты: **Шаблоны матриц, Шаблоны скобок, Шаблоны верхних и нижних индексов, Пробелы и многоточия**.

5. Сохраните документ.

Упражнение 6

1. Для выполнения упражнения вам потребуется любой текстовый документ, состоящий из нескольких страниц. Если такого документа нет, то создайте его. Можно ввести 1 абзац текста, а затем выполнить копирование и вставку текста в конец документа несколько раз так, чтобы образовались 3–4 полностью заполненных текстом страницы. Сохраните документ в файле с именем **Текст6.doc**.

2. Выполните проверку правописания (**Сервис – Правописание...**).

3. Выполните поиск в документе какого-либо слова (**Правка – Найти**). Выполните замену одного произвольного слова, которое встречается в документе много раз, на другое.

4. Создайте в документе 2 сноски произвольного содержания (**Вставка – Ссылка – Сноска…**).

5. Создайте элемент автотекста (**Вставка – Автотекст – Создать…**) с именем **Элемент 1**. Вставьте элемент несколько раз в документ.

6. Вставьте в документ номера страниц (**Вставка – Номера страниц**).

7. Вставьте в документ разрыв раздела дважды – в конце первой страницы и в начале третьей страницы (**Вставка** – **Разрыв** – **Новый раздел** – **на текущей странице**). Включив режим показа непечатных знаков, убедитесь, что символы разрыва раздела появились в документе. Теперь документ состоит из трех разделов.

8. Установите курсор внутрь второго раздела. Измените ориентацию страниц второго раздела, выполнив в меню **Файл – Параметры страницы** – вкладка **Поля** – ориентация **альбомная**.

9. Отформатируйте текст во втором разделе в виде колонок газетного стиля, выполнив в меню **Формат** – **Колонки** – **Число колонок** равно 3.

10. Отформатируйте некоторые абзацы текста стилем **Заголовок 1**, некоторые – стилем **Заголовок 2** (выделить текст, затем в меню **Формат** – **Стили и форматирование** – выбрать стиль).

11. Вставьте оглавление в конец документа. Для этого установите курсор в конец документа и выполните в меню **Вставка – Ссылка – Оглавление и указатели**. Выберите формат оглавления.

12. Создайте закладку **«Начало»** в начале первой страницы документа. Для этого поместите курсор в начало первого абзаца текста документа и выполните в меню **Вставка – Закладка –** ввести имя закладки **«Начало» – Добавить.**

13. В конце первой страницы создайте гиперссылку **«В начало документа»**. Для этого поместите курсор в место вставки гиперссылки и выполните в меню **Вставка – Ги-**

перссылка – Местом в этом документе – выбрать закладку «Начало». Закрыв диалоговое окно создания гиперссылки, проверьте ее действие. Щелчок по гиперссылке при нажатой клавише **<Ctrl>** на клавиатуре должен перемещать вас в начало документа.

14. Скопируйте гиперссылку и вставьте ее в конец каждой страницы документа. Операция копирования гиперссылки выполняется точно так же, как операция копирования обычного фрагмента текста.

15. Сохраните документ.

Практическая работа 3.3. Работа с программой MS Word. Использование инструмента слияния для создания массовых рассылок

Цель работы – научиться создавать массовые рассылки документов.

Порядок выполнения работы

1. Выполните упражнение 7 и практическое задание.

Упражнение 7

Создадим серию документов-предписаний должникам по оплате коммунальных услуг (рис. 3.33) на основе имеющегося списка должников.

Предписание

Адрес: улица Свердлова дом 4 квартира 10
Получатель: КАСАТКИН А.А.

Правление ТСЖ-44 доводит до Вашего сведения, что долг по оплате коммунальных услуг по Вашей квартире достиг суммы 110 рублей.

Просим Вас оплатить долг. В противном случае Правление ТСЖ-44 будет вынуждено принять адекватные меры воздействия.

Рис. 3.33. Экземпляр документа **Предписание**

Технология выполнения задания

1. Запустите программу MS Excel. Создайте список рассылки (рис. 3.34).

	A	B	C	D	E
1	**ФИО**	**Улица**	**Дом**	**Кв**	**Долг**
2	КАСАТКИН А.А.	Свердлова	4	10	110,00р.
3	КОРОЛЕВА М.Я.	Ленинградская	22	33	300,00р.
4	КРОТОВ П.С.	Родины	123	87	450,00р.
5	НИКИТИН И.Р.	Чапаева	50	22	100,00р.
6	ИГОНИНА Л.П.	Свердлова	22	12	320,00р.
7	АГЕЕВА.А.	Родины	19	15	220,00р.

Рис. 3.34. Таблица MS Excel, содержащая список адресатов

2. Сохраните документ в файле **Список.xls**. Завершите работу программы MS Excel. Список будет использован при создании документов рассылки.

3. Создайте новый документ MS Word. Введите с клавиатуры основной документ, содержащий общий для всех документов текст (рис. 3.35):

Предписание

Адрес: улица дом квартира
Получатель:

Правление ТСЖ-44 доводит до Вашего сведения, что долг по оплате коммунальных услуг по Вашей квартире достиг суммы рублей.

Просим Вас оплатить долг. В противном случае Правление ТСЖ-44 будет вынуждено принять адекватные меры воздействия.

Рис. 3.35. Основной документ рассылки

4. Выполните в меню **Сервис** – **Письма и рассылки** – **Слияние**. В области задач выберите тип документа **Письма**. Перейдите к следующему этапу.

5. На втором этапе выберите тип документа **Текущий документ**.

6. Перейдите к третьему этапу **Мастера слияния** и выберите в списке **Выбор получателей** опцию **Использование списка**. Затем щелчком по кнопке **Обзор** откройте диалоговое окно для выбора файла со списком. Выберите файл **Список.xls**. Откроется диалоговое окно **Выделить таблицу** для выбора листа рабочей книги. Выберите **Лист1** и щелчком по кнопке **ОК** закройте окно. Затем в новом окне **Получатели слияния** можно выбрать отдельные записи из списка либо оставить весь список.

7. Перейдите к следующему этапу (этап 4) **Создание письма**. Установите курсор в документе после слова «**улица**», выберите опцию **Другие элементы**. Откроется окно для выбора поля слияния (рис. 3.36), в котором нужно выбрать поле **Улица**, нажать кнопку **Вставить** и закрыть окно.

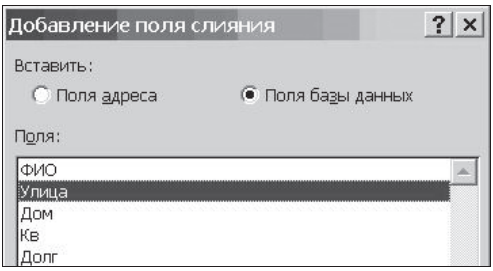


Рис. 3.36. Выбор поля слияния для вставки в документ

8. Повторите эту процедуру для остальных полей. В результате документ примет вид, показанный на рис. 3.37.

Предписание

Адрес: улица «Улица» дом «Дом» квартира «Кв»
Получатель: «ФИО»

Правление ТСЖ-44 доводит до Вашего сведения, что долг по оплате коммунальных услуг по Вашей квартире достиг суммы «Долг» рублей.

Просим Вас оплатить долг. В противном случае Правление ТСЖ-44 будет вынуждено принять адекватные меры воздействия.

Рис. 3.37. Вид документа с вставленными в него полями слияния

9. Перейдите на следующий этап (5 из 6) **Просмотр писем**. Произойдет слияние основного документа с источником данных. Пользователю будет представлена для просмотра первая страница.

10. Перейдите к этапу 6 **Завершение слияния** и выберите опцию **Изменить часть писем**. Откроется диалоговое окно, в котором нужно выбрать опцию **все**. Процедура слияния завершится. Каждой строке из документа MS Excel будет поставлена в соответствие страница документа, содержащая текст предписания, адресованный конкретному человеку.

11. Сохраните документ в файле **Предписание.doc**.

Практическое задание

В соответствии с номером варианта выберите тип документа из табл. 3.6. Создайте основной документ и файл источника данных для процедуры слияния. Выполните процедуру слияния.

Таблица 3.6

Варианты заданий

Номер варианта	Тип документа	Вид итогового документа	Содержание источника данных
1, 6, 11	Приглашение	Квартира: 56 ФИО: Иван Петрович Громов Правление ТСЖ-23 приглашает Вас принять участие в собрании жильцов дома, которое состоится 12.05.2011 в 18 часов во дворе дома	Фамилия, имя, отчество, номер квартиры
2, 7, 12	Предложение услуги	Адрес: Мира, 43, кв. 78 Уважаемые жильцы! Сообщаем, что в 2011 году истекает срок эксплуатации прибора учета потребления электроэнергии, установленного по Вашему адресу. Наша фирма может в кратчайший срок и с высоким качеством выполнить услуги по замене и регистрации приборов учета электроэнергии. ООО «Электрика»	Улица, номер дома, номер квартиры

Окончание табл. 3.6

Номер варианта	Тип документа	Вид итогового документа	Содержание источника данных
3, 8, 13	Приглашение	ФИО: Ильин Игорь Иванович Адрес: Тополиная, 12, кв. 45 Число голосующих акций: 300 Уважаемый акционер! Совет директоров ОАО «Этна» извещает о проведении 25.06.11 годового общего собрания акционеров. Начало собрания в 9.00. Место проведения: Южное шоссе, 12, актовый зал	Фамилия, имя, отчество, улица, номер дома, номер квартиры, число голосующих акций
4, 9, 14	Приглашение	Адрес: ул. Снежная, 25, кв. 77 ФИО: Зыкова Анна Николаевна Уважаемый ветеран! Приглашаем Вас на торжественное собрание и концерт по случаю празднования Дня машиностроителя. Мероприятие состоится 30 июня в 17-00 во Дворце культуры «Машиностроитель»	Улица, номер дома, номер квартиры, фамилия, имя, отчество
5, 10, 15	Поздравительное письмо	ФИО: Крылова Мария Сергеевна Подразделение: Бухгалтерия Мария Сергеевна! Руководство и профсоюзный комитет ОАО «Крокус» поздравляет Вас с 20-летием нашего предприятия и сообщает о награждении Вас Почетной грамотой и премией в размере 5000 рублей.	Фамилия, имя, отчество, подразделение, размер премии

Практическая работа 3.4. Использование мастера функций MS Excel. Построение диаграмм

Цель работы – научиться:

- вводить формулы в ячейки таблицы;
- использовать **Мастер функций** MS Excel для вставки в формулы стандартных функций;
- использовать в формулах абсолютные и относительные ссылки на ячейки;
- выполнять построение диаграмм разных типов.

Порядок выполнения работы

1. Выполните практическое задание.

Практическое задание

Рассматривается некоторое предприятие, выпускающее продукцию, объем выпуска которой оценивается в денежном выражении. Создать таблицу **«Показатели производства»** (рис. 3.38). Построить диаграммы:

- ***диаграмму-график,*** отражающую ***динамику изменения фактического выпуска продукции в течение года***;
- ***круговую диаграмму,*** показывающую ***долю каждого месяца в общей сумме*** по столбцу ***Фактически выпущено***.

	A	B	C	D	E
1	ПОКАЗАТЕЛИ ПРОИЗВОДСТВА				
2	*Месяцы*	*План выпуска*	*Фактически выпущено*	*Процент выполнения плана*	*Выполнено в процентах к фактически выпущенному за год*
3	Январь	100 000,00р.	110 000,00р.	110,00%	8,01%
4	Февраль	110 000,00р.	100 000,00р.	90,91%	7,28%
5	Март	100 000,00р.	102 000,00р.	102,00%	7,43%
6	Апрель	110 000,00р.	105 000,00р.	95,45%	7,65%
7	Май	120 000,00р.	130 000,00р.	108,33%	9,47%
8	Июнь	150 000,00р.	140 000,00р.	93,33%	10,20%
9	Июль	120 000,00р.	120 000,00р.	100,00%	8,74%
10	Август	130 000,00р.	130 000,00р.	100,00%	9,47%
11	Сентябрь	100 000,00р.	120 000,00р.	120,00%	8,74%
12	Октябрь	110 000,00р.	105 000,00р.	95,45%	7,65%
13	Ноябрь	120 000,00р.	110 000,00р.	91,67%	8,01%
14	Декабрь	100 000,00р.	101 000,00р.	101,00%	7,36%
15	*Итого за год:*	1 370 000,00р.	1 373 000,00р.	100,22%	
16		*Максимально за месяц*	140 000,00р.	120,00%	
17		*Минимально за месяц*	100 000,00р.	90,91%	
18		*В среднем за месяц*	114 416,67р.	100,68%	

Рис. 3.38. Вид таблицы для практической работы 3.4

Исходные данные:

- планируемые показатели производства продукции за 12 месяцев (столбец **B**);
- фактические показатели по результатам работы (столбец **C**).

Вычисляются:

- процент выполнения плана в каждом месяце (столбец **D**) и процент выполнения плана в целом за год (ячейка **D15**);
- доля фактического выпуска продукции в каждом месяце в общем объеме продукции, выпущенной за год (столбец **E**);
- суммарные показатели планового (ячейка **B15**) и фактического (ячейка **C15**) выпуска продукции;
- максимальное (**C16**), минимальное (**C17**) и среднее (**C18**) значения фактического выпуска за 12 месяцев; максимальное (**D16**), минимальное (**D17**) и среднее (**D18**) значения процента выполнения плана за 12 месяцев.

Расчетные формулы:

- $Процент\ выполнения\ плана = \frac{Фактически\ выпущено}{План\ выпуска};$
- $Вып.в\ проц.к\ факт.вып.\ за\ год = \frac{Фактически\ выпущено за\ данный\ месяц}{Фактически\ выпущено за\ год}.$

Технология выполнения задания

1. Запустите программу **Microsoft Excel**. Для работы используйте **Лист1** чистой рабочей книги.

2. Объедините ячейки диапазона **A1:E1** в одну, щелкнув мышкой инструмент **Объединить и поместить в центре**  на панели инструментов **Форматирование,** и введите в нее текст **«Показатели производства»**.

3. Перед вводом текста в ячейки диапазона **A2:E2** выделите их и установите выравнивание так, чтобы текст в ячейке не выходил за её границы:

- выполните в меню **Формат – Ячейки – Выравнивание**;
- установите флажок **переносить по словам** в группе **Отображение**.

4. Введите текст заголовков столбцов.

5. Заполните **столбец А** названиями месяцев, используя **Автозаполнение**. Для этого в ячейку **А3** введите текст **«Январь»**, протяните вниз маркер заполнения.

6. Введите числовые значения в столбцы **В** и **С**. Перед вводом выделите ячейки диапазона **В3:С18** и установите денежный формат. Для этого выполните в меню **Формат – Ячейки – Число**, откроется диалоговое окно, в котором нужно выбрать формат. Аналогично выделите ячейки диапазона **D3:E18** и установите процентный формат. Можно также использовать инструменты **Денежный формат** и **Процентный формат** на панели инструментов **Форматирование**.

☑ При вводе чисел не следует вводить пробелы между цифрами или указывать денежные единицы. В этом случае вводимые данные будут интерпретированы программой как текст, а не как число.

7. В ячейки **В15** и **С15** введите формулы для вычисления сумм значений в соответствующих столбцах, для ввода используйте инструмент **Автосумма**. Для заполнения ячейки **В15** выделите ее, щелкните мышкой инструмент **Автосумма**, выделите мышкой диапазон **В2:В14**, при этом выделяемый диапазон отмечается «бегущей» пунктирной рамкой, нажмите на клавиатуре **<Enter>**. Аналогично заполните ячейку **С15**.

8. Заполните диапазон ячеек **D3:D15** формулами для расчета **процента выполнения плана в каждом месяце и в целом за год**. В ячейку **D3** введите формулу **=C3/B3**, затем протяните ячейку вниз до **D15**, используя маркер заполнения. Обратите внимание на то, что при протягивании адреса в формулах автоматически изменяются, так как в исходной формуле были использованы относительные адреса.

9. Заполните диапазон ячеек **E3:E14** формулами для расчета доли фактического выпуска продукции в каждом месяце в общем объеме продукции, выпущенной за год. В ячейку **E3** введите формулу **=C3/C15**. Чтобы в данном случае можно было применить механизм автозаполнения смежных ячеек формулами, нужно ***«зафиксировать»*** адрес **С15** в формуле ячейки **E3**. Для этого двойным щелчком выделите вновь эту ячейку, чтобы войти в режим редактирования формулы, выделите адрес **С15** в формуле и нажмите на

клавиатуре клавишу **<F4>**. Адрес **C15** изменится на **$C$15**, т. е. превратится в абсолютный адрес, который не изменяется при автозаполнении. Нажмите клавишу **<Enter>**, чтобы завершить ввод формулы в ячейку. Протяните формулу на ячейки диапазона **E4:E14**.

10. Ячейки диапазона **A16:B16** объедините в одну и введите текст «**Максимально за месяц**». Аналогично заполните расположенные ниже ячейки.

11. Введите формулу в ячейку **C16** для вычисления максимального значения в диапазоне **C3:C14**. Для вычисления используйте стандартную функцию **МАКС**. Формула **=МАКС(C3:C14)** позволит вычислить наибольшее числовое значение в диапазоне ячеек **C3:C14**.

Вставка стандартной функции **МАКС** в формулу выполняется с помощью **Мастера функций** по следующему алгоритму:

- выполняем в меню **Вставка – Функция**, в открывшемся диалоговом окне выбираем категорию **Статистические** и функцию **МАКС;**
- в следующем окне **Аргументы функции** указываем диапазон **C3:C14** в поле **Число1** (рис. 3.39), выделив мышкой этот диапазон в таблице, нажимаем кнопку **ОК**.

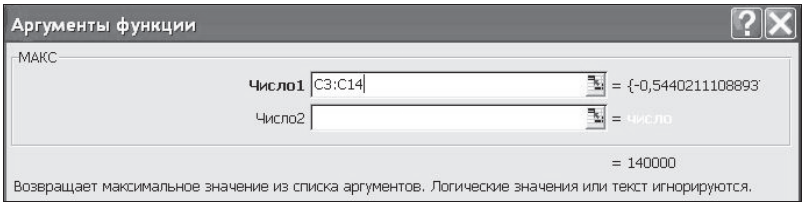


Рис. 3.39. Ввод аргументов функции **МАКС**

12. Аналогично введите формулу в ячейку **D16** для вычисления максимального значения в диапазоне **D3:D14** по формуле **=МАКС(D3:D14)**. Для вычисления минимального значения в ячейках **C17** и **D17** используйте аналогичную процедуру вставки функции **МИН**, для вычисления среднего значения в ячейках **C18** и **D18** – процедуру вставки функции **СРЗНАЧ**.

☑ **Примечание.** Вставку функций **МАКС, МИН, СРЗНАЧ** можно выполнить за меньшее число шагов с помощью инструмента **Автосумма**, открыв список и выбрав соответствующее действие (рис. 3.40).

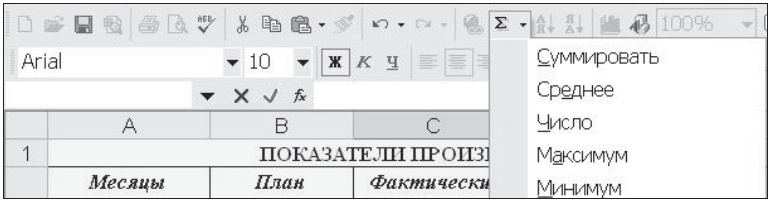


Рис. 3.40. Использование инструмента **Автосумма** для выбора функции

13. Выполните обрамление и заливку цветом ячеек таблицы.

14. Постройте ***диаграмму-график,*** отражающую ***динамику изменения фактического выпуска продукции*** (рис. 3.41):

- выделите в таблице диапазон ячеек **A2:A14**, затем нажмите на клавиатуре клавишу <**Ctrl**> и выделите диапазон ячеек **C2:C14** ;
- щелкните мышкой инструмент **Мастер диаграмм** на панели инструментов **Стандартная**; в окне **Мастер диаграмм (шаг 1)** выберите тип диаграммы **График** и вид диаграммы **График с маркерами,** нажмите кнопку **Далее**;
- в следующем окне согласитесь с предложенными программой параметрами, нажав кнопку **Далее**; на третьем шаге построения откажитесь от легенды, сбросив флажок **Добавить легенду** на вкладке **Легенда,** нажмите кнопку **Далее**;
- на последнем шаге укажите местоположение диаграммы (на отдельном листе или на листе с расчетной таблицей) и нажмите кнопку **Готово**. Диаграмма построена.

15. Постройте аналогично ***круговую диаграмму,*** показывающую ***долю каждого месяца в общей сумме*** по столбцу ***Фактически выпущено*** (рис. 3.42):

- выделите диапазон ячеек **A2:A14**, затем нажмите на клавиатуре <**Ctrl**> и выделите диапазон ячеек **C2:C14**;
- запустите программу **Мастер диаграмм** и выполните необходимые действия. При определении параметров диаграммы (шаг 3) включите в подписи данных **имена категорий** и **доли**.

Рис. 3.41. Диаграмма-график

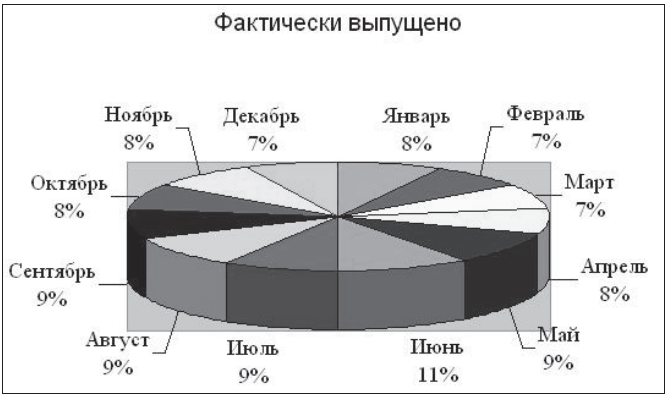


Рис. 3.42. Круговая диаграмма

16. Сохраните документ.

Практическая работа 3.5. Использование логических функций. Операции с листами рабочей книги MS Excel

Цель работы – научиться:

- выполнять вычисления с использованием логических функций;
- выполнять операции с листами рабочей книги;
- использовать в формулах адреса ячеек с других листов рабочей книги.

Порядок выполнения работы

1. Выполните практическое задание.

Практическое задание

Рассматривается некоторое подразделение (отдел или цех). Начисление заработной платы сотрудникам подразделения выполняется в конце каждого месяца на условиях почасовой оплаты труда. Нужно создать таблицу расчета заработной платы за 3 месяца (январь, февраль, март). Форма таблицы для расчета заработной платы за январь показана на рис. 3.43.

	A	B	C	D	E	F	G	H
1	*Минимальная заработная плата*			**800,00р.**				
2	*Ставка подоходного налога*			**13%**				
3			*Тариф*	**100,00р.**				
4								
5	*Ф.И.О.*	*Кол-во иждивенцев*	*Кол-во отработанных часов*	*Начислено*	*Начислено в сумме с начала года*	*Налоговый вычет*	*Подоходный налог*	*Сумма к выдаче*
6	Бекетова Л.Я.	**2**	**100**	10 000,00р.	10 000,00р.	2 400,00р.	988,00р.	9 012,00р.
7	Иванов Н.В.	**1**	**150**	15 000,00р.	15 000,00р.	1 600,00р.	1 742,00р.	13 258,00р.
8	Кротов В.И.	**0**	**150**	15 000,00р.	15 000,00р.	800,00р.	1 846,00р.	13 154,00р.
9	Репина Е.А.	**3**	**100**	10 000,00р.	10 000,00р.	3 200,00р.	884,00р.	9 116,00р.
10	Розов С.П.	**2**	**150**	15 000,00р.	15 000,00р.	2 400,00р.	1 638,00р.	13 362,00р.
11	*Всего*	8	650	65 000,00р.	65 000,00р.	10 400,00р.	7 098,00р.	57 902,00р.
12	*Среднее*	1,6	130,0	13 000,00р.	13 000,00р.	2 080,00р.	1 419,60р.	11 580,40р.
13	*Наибольшее*	3	150	15 000,00р.	15 000,00р.	3 200,00р.	1 846,00р.	13 362,00р.
14	*Наименьшее*	0	100	10 000,00р.	10 000,00р.	800,00р.	884,00р.	9 012,00р.

Рис. 3.43. Форма таблицы для практической работы 3.5

Исходные данные: минимальная заработная плата, ставка подоходного налога, тариф, количество отработанных часов и количество иждивенцев[1] для каждого сотрудника.

Вычисляются:

- начисленная заработная плата за месяц (столбец **Начислено**);
- суммарная заработная плата с начала года (столбец **Начислено в сумме с начала года**);

[1] Иждивенцы – лица, находящиеся на длительном или постоянном материальном или денежном обеспечении со стороны других лиц (Райзберг Б.А., Лозовский Л.Ш., Стародубцева Е.Б. Современный экономический словарь. 2-е изд., испр. М., 1999). К числу иждивенцев относятся, например, несовершеннолетние дети, находящиеся на материальном обеспечении у своих родителей. При наличии иждивенцев работнику предоставляется налоговый вычет (льгота).

- размер налогового вычета (столбец **Налоговый вычет**);
- подоходный налог (столбец **Подоходный налог**);
- итоговая сумма после удержания подоходного налога (столбец **Сумма к выдаче**).

Технология выполнения задания

1. Запустите программу **MS Excel**. Переименуйте листы рабочей книги, используя контекстное меню для ярлычков листов: **Лист1 → Январь**, **Лист2 → Февраль**, **Лист3 → Март**.

2. Заполните ячейки, содержащие исходные данные.

3. В ячейку **D6** введите формулу **=$D$3*C6** (***Начислено = Тариф * Кол-во отработанных часов***). Используя маркер автозаполнения, скопируйте формулу на все ячейки столбца **D**.

4. В ячейку **E6** введите формулу **=D6** (в январе суммарное начисление с начала года равно тому, что начислено в январе). Используя маркер автозаполнения, скопируйте формулу на все ячейки столбца **E**.

5. В ячейку **F6** введите формулу **=ЕСЛИ(E6 <= 20000; $D$1*(B6+1); 0).**

(*Если* **Начислено в сумме с начала года <= 20000,** *тогда* ***Налоговый вычет = Минимальная заработная плата * (Кол-во иждивенцев + 1),*** *иначе* ***Налоговый вычет = 0***). Используя маркер автозаполнения, скопируйте формулу на все ячейки столбца **F**.

6. В ячейку **G6** введите формулу **=(D6-F6)*$D$2** (***Подоходный налог = (Начислено – Налоговый вычет) * Ставка подоходного налога***). Используя маркер автозаполнения, скопируйте формулу на все ячейки столбца **G**.

7. В ячейку **H6** введите формулу **=D6-G6** (***Сумма к выдаче = Начислено – Подоходный налог***). Используя маркер автозаполнения, скопируйте формулу на все ячейки столбца **H**.

8. В ячейку **B11** введите формулу **=СУММ(B6:B10)**, используя инструмент **Автосумма**. Используя маркер автозаполнения, скопируйте формулу на все ячейки строки **11**.

9. Заполните строку **12**, используя функцию **СРЗНАЧ**. Заполните строку **13**, используя функцию **МАКС**. Заполните строку **14**, используя функцию **МИН**.

10. Скопируйте таблицу с листа **Январь** на листы **Февраль** и **Март**. Измените данные в столбце **Кол-во отработанных часов** на листах **Февраль** и **Март**.

11. Измените формулу в столбце **Начислено в сумме с начала года** на листе **Февраль**: дважды щелкните ячейку **E6**, нажмите на клавиатуре клавишу «+», щелчком по ярлычку листа переключитесь на лист **Январь**, выделите ячейку **E6** на листе **Январь**, нажмите клавишу **<Enter>**. В ячейку будет введена формула **=D6+Январь!E6.** Используя маркер автозаполнения, скопируйте формулу на все ячейки столбца **E**. Аналогично измените формулу в столбце **Начислено в сумме с начала года** на листе **Март**.

12. Сохраните документ.

Практическая работа 3.6. Обработка информации, структурированной в виде списков MS Excel

Цель работы – научиться:

- выполнять контроль ввода данных в таблицу;
- использовать инструмент создания сводных таблиц для обработки данных, структурированных в виде списков MS Excel.

Порядок выполнения работы

1. Выполните практическое задание.

Практическое задание

Организация ведет учет поступающих материалов с помощью электронной книги, содержащей три таблицы: **Справочник поставщиков**, **Справочник материалов** и **Приходная накладная**. В таблице **Справочник поставщиков** хранится информация о поставщиках материалов. В таблице **Справочник материалов –** о поставляемых материалах. В таблице **Приходная накладная** фиксируются факты поступления материалов от поставщиков.

Требуется:

- создать рабочую книгу, содержащую таблицы, указанные в задании;
- заполнить таблицы данными;
- сформировать ведомость «**Фактическое выполнение поставок**», для чего необходимо рассчитать два показа-

теля: 1) сумма поставок, выполненная каждым поставщиком; 2) общая сумма поставок, выполненная всеми поставщиками.

Технология выполнения задания

1. Запустите программу MS Excel. Переименуйте лист рабочей книги **Лист1** в **Справочник поставщиков**. Введите заголовок таблицы и заголовки столбцов таблицы (рис. 3.44).

	A	B	C	D
1	Поставщики			
2	Код поставщика	Наименование поставщика	Адрес поставщика	Расчетный счет
3				

Рис. 3.44. Форма таблицы **Справочник поставщиков**

2. Организуйте контроль ввода данных в столбец **Код поставщика**:

• выделите ячейки **A3:A7**, выполните команды меню **Данные – Проверка...**, в диалоговом окне на вкладке **Параметры** выберите в раскрывающемся списке значение **Целое число** и укажите минимальное и максимальное значения вводимых кодов поставщиков (будем считать, что коды могут быть в диапазоне от 100 до 105);

• сформируйте подсказку об ограничениях ввода, выбрав вкладку **Сообщение для ввода** и заполнив соответствующие поля диалога (рис. 3.45). При активации ячеек из диапазона **A3:A7** на экран будет выводиться подсказка «Код поставщика может принимать значение в диапазоне 100–105»;

• сформируйте сообщение об ошибке, которое будет выводиться на экран при вводе в ячейки диапазона **A3:A7** неверных данных, выбрав вкладку **Сообщение об ошибке** и заполнив соответствующие поля диалога (рис. 3.46).

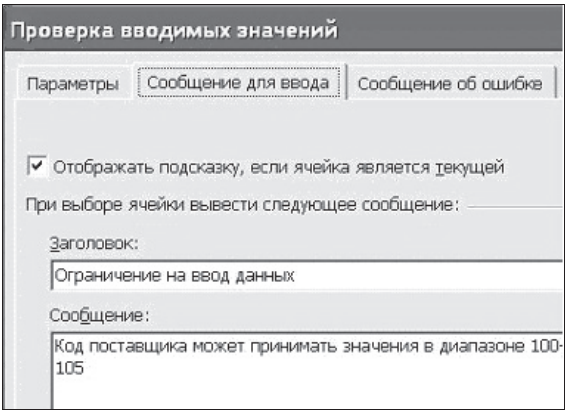


Рис. 3.45. Формирование подсказки

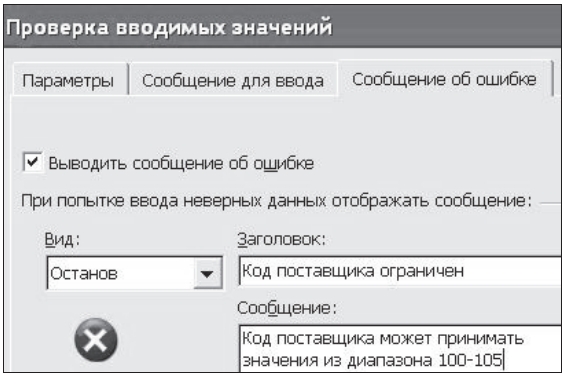


Рис. 3.46. Формирование сообщения об ошибке при неправильном вводе данных

3. Отформатируйте ячейки **D3:D7** для ввода текстовых символов, выполнив в меню **Формат – Ячейки – Число – Текстовый**. Заполните таблицу значениями (рис. 3.47).

4. Присвойте блоку ячеек **A3:D7** имя **Код_Поставщика**: выделите ячейки **A3:D7** и выполните в меню **Вставка – Имя – Присвоить**, в диалоговом окне **Присвоение имени** введите имя, нажмите кнопку **Добавить** и закройте окно кнопкой **OK**.

☑ Теперь к диапазону ячеек **A3:D7** на листе **Справочник поставщиков** можно обращаться по имени **Код_поставщика**.

	A	B	C	D
1	Поставщики			
2	Код поставщика	Наименование поставщика	Адрес поставщика	Расчетный счет
3	100	Заря	Москва	11111111111111
4	101	Аврора	Казань	22222222222222
5	102	Восход	Пермь	33333333333333
6	103	Космос	Тверь	44444444444444
7	104	Азов	Тула	55555555555555

Рис. 3.47. Данные о поставщиках

5. Переименуйте лист рабочей книги **Лист2** в **Справочник материалов**. Введите заголовок таблицы и заголовки столбцов (рис. 3.48). Организуйте контроль ввода данных в столбец **Код материала**, исходя из того, что коды могут иметь значения в диапазоне от 1000 до 1200. Заполните таблицу значениями.

	A	B	C
1	Материалы		
2	Код материала	Наименование материала	Единица измерения
3	1001	Краска	кг
4	1002	Лак	кг
5	1003	Цемент	т
6	1004	Кирпич	шт
7	1005	Стекло	кв_м

Рис. 3.48. Таблица **Справочник материалов**

6. Присвойте блоку ячеек **A3:D7** имя **Код_материала**.

7. Переименуйте **Лист3** в **Приходная накладная**. Заполните таблицу **Приходная накладная** (рис. 3.49). Организуйте проверку ввода данных в столбцы **Код поставщика** и **Код материала**.

8. Заполните столбец **Наименование поставщика** в соответствии с кодом поставщика, используя стандартную функцию **ВПР** для выборки по коду поставщика соответствующего наименования из таблицы **Справочник поставщиков**. Для этого в ячейку **B3** введите формулу **=ВПР(A3; Код_поставщика; 2)**.

Приходная накладная					
Код поставщика	Наименование поставщика	Код материала	Наименование материала	Дата поставки	Сумма поставки фактическая
100		1001		05,09,05	5,00
100		1001		05,09,05	7,00
101		1003		06,09,05	3,00
101		1005		07,09,05	4,00
102		1001		07,09,05	2,00
102		1002		07,09,05	3,00
102		1003		07,09,05	5,00
103		1004		08,09,05	1,00
103		1005		08,09,05	2,00
103		1005		09,09,05	5,00
103		1006		09.09.05	5.00

Рис. 3.49. Таблица **Приходная накладная**

9. Скопируйте формулу из ячейки **B3** в ячейки диапазона **B4:B13**. В столбец **Наименование поставщика** будут введены названия поставщиков.

10. Заполните столбец **Наименование материала** в соответствии с кодом материала, используя стандартную функцию **ВПР** для выборки по коду материала соответствующего наименования из таблицы **Справочник материалов**. В ячейку **D3** следует ввести формулу **=ВПР(C3; Код_материала; 2)**, затем выделить ячейку **D3** и протянуть выделение вниз за правый нижний угол ячейки.

В результате выполненных действий таблица **Приходная накладная** будет заполнена значениями (рис. 3.50).

	A	B	C	D	E	F
1	Приходная накладная					
2	Код поставщика	Наименование поставщика	Код материала	Наименование материала	Дата поставки	Сумма поставки фактическая
3	100	Заря	1001	Краска	05,09,05	5,00
4	100	Заря	1001	Краска	05,09,05	7,00
5	101	Аврора	1003	Цемент	06,09,05	3,00
6	101	Аврора	1005	Стекло	07,09,05	4,00
7	102	Восход	1001	Краска	07,09,05	2,00
8	102	Восход	1002	Лак	07,09,05	3,00
9	102	Восход	1003	Цемент	07,09,05	5,00

Рис. 3.50. Таблица **Приходная накладная** с введенными данными

11. Создайте ведомость «**Фактическое выполнение поставок**»:

• установите курсор в любую ячейку с данными таблицы **Приходная накладная**;

• выполните в меню **Данные – Сводная таблица**, откроется окно **Мастера сводных таблиц и диаграмм**;

• с помощью кнопки **Далее** переместитесь на **шаг 2** и, ничего не изменяя, переместитесь на **шаг 3**; нажмите кнопку **Макет**, откроется диалоговое окно формирования сводной таблицы;

• переместите с помощью мыши поле **Код материала** в область **Страница**; переместите поле **Наименование поставщика** в область **Строка**; переместите поле **Сумма поставки фактическая** в область **Данные** (рис. 3.51), нажмите **ОК**, выберите опцию **новый лист**, затем нажмите кнопку **Готово**.

Рис. 3.51. Формирование макета сводной таблицы

Вы увидите сводную таблицу, в которой представлены суммарные значения поставок по каждому поставщику (рис. 3.52).

	A	B
1	Код материала	(Все)
2		
3	Сумма по полю Сумма поставки фактическая	
4	Наименование поставщика	Итог
5	Аврора	7
6	Восход	10
7	Заря	12
8	Космос	13
9	Общий итог	42

Рис. 3.52. Сформированная ведомость
Фактическое выполнение поставок

12. Переименуйте лист со сводной таблицей в **Фактическое выполнение поставок**.

13. Сохраните документ.

3.7. Тестовые задания

1. Согласно принципам фон Неймана для хранения программы и обрабатываемых программой данных используется...

a) устройство ввода данных;
b) арифметико-логическое устройство;
c) устройство управления;
d) запоминающее устройство.

2. Основу элементной базы компьютеров первого поколения составляли...

a) электронные лампы;
b) БИС (большие интегральные схемы);
c) интегральные схемы;
d) транзисторы.

3. Основными характеристиками микропроцессора являются...

a) тактовая частота;
b) разрядность;
c) надежность;
d) вес.

4. Комплекс программ, обеспечивающих управление работой всех аппаратных устройств и доступ пользователя к ним, называется...

a) утилитой;
b) операционной системой;
c) пакетом прикладных программ;
d) интерфейсом.

5. Прикладное программное обеспечение – это...

a) совокупность программ для обеспечения работы компьютера;
b) совокупность программ для решения задач конкретной предметной области;
c) совокупность инструментальных средств для разработки программ;

d) совокупность системных программ вспомогательного назначения.

6. Антивирусные программы относят к группе программ...

a) базовые системные;
b) прикладные;
c) инструментальные;
d) сервисные системные.

7. Основным форматом сохранения документов Microsoft Word является формат...

a) .TXT
b) .RTF
c) .DOC
d) .XLS

8. Документ Microsoft Word показывается в окне редактора так, как он будет напечатан, в режиме...

a) разметка страницы;
b) обычный;
c) WEB-документ;
d) структура.

9. Выберите из списка параметры форматирования шрифта:

a) гарнитура (шрифтовой набор);
b) отступы;
c) начертание;
d) размер;
e) междустрочный интервал.

10. Выберите из списка параметры форматирования абзаца:

a) размер;
b) междустрочный интервал;
c) отступы;
d) гарнитура (шрифтовой набор);
e) выравнивание.

11. Для печати в документе MS Word сложных математических формул используется инструмент вставки объекта...

a) Microsoft Clip Gallery;
b) Microsoft Equation 3.0;
c) QuickTimeMovie;
d) QuickTimePicture.

12. Файлы документов программы Microsoft Excel имеют расширение...

a) .XLS
b) .DOC
c) .MDB
d) .BMP

13. Ввод формулы в ячейку электронной таблицы начинается с...

a) символа «+»;
b) любого символа;
c) символа “ (кавычки);
d) символа «=» (равно).

14. Абсолютный адрес ячейки в формуле при копировании этой формулы в другие ячейки...

a) изменяется в соответствии с новым местоположением формулы;
b) остается неизменным;
c) изменяется в той его части, которая не отмечена символом $;
d) изменяется в той его части, которая отмечена символом $.

15. В ячейке **B1** находится формула **=A1+$A$2**. При копировании формулы в ячейку **C1** она примет вид...

a) **=B1+$A$2**;
b) **=A1+$A$2**;
c) **=A2+$A$2**;
d) **=A3+$A$2**.

16. В ячейке **B3** находится формула **=МИН(A1:A5) * 5 – 1**. В ячейках **A1:A5** находятся числа 1, 2, 3, 4, 5. Результат расчета в ячейке **B3** равен...

a) 5;
b) 3;
c) 7;
d) 4.

17. Для иллюстрации доли каждого значения в сумме всех значений некоторого ряда данных следует использовать диаграмму типа...

a) график;

b) круговая;
c) точечная;
d) гистограмма.

18. Документ Microsoft PowerPoint – это...
a) электронная таблица;
b) база данных;
c) электронная презентация;
d) текстовый документ.

Библиографический список

Рекомендуемая литература

1. Информатика : учебник / Б.В. Соболь [и др.]. – Ростов н/Д : Феникс, 2010. – 446 с.
2. Информатика для юристов и экономистов / под ред. С.В. Симоновича. – СПб. : Питер, 2008. – 687 с.
3. Климова, Л.М. Pascal 7.0: практическое программирование. Решение типовых задач : учебн. пособие / Л.М. Климова. – М. : КУДИЦ-образ, 2000. – 516 с.
4. Могилев, А.В. Информатика : учебн. пособие для вузов / А.В. Могилев, Н.И. Пак, Е.К. Хеннер ; под ред. Е.К. Хеннера. – М. : Академия, 2004. – 842 с.
5. Немнюгин, С.А. Изучаем Turbo Pascal : учебн. пособие / С.А. Немнюгин, Л.В. Перколаб. – СПб. : Питер, 2005. – 312 с.
6. Павловская, Т.А. Паскаль: программирование на языке высокого уровня : учебн. для вузов / Т.А. Павловская. – СПб. : Питер, 2007. – 392 с.
7. Стариченко, Б.Е. Теоретические основы информатики : учебн. пособие для вузов / Б.Е. Стариченко. – М. : Горячая линия-Телеком, 2004. – 311 с.
8. Сырецкий, Г.А. Информатика: фундаментальный курс. В 2 т. Т. 1. Основы информационной и вычислительной техники : учебн. для вузов / Г.А. Сырецкий. – СПб. : БХВ-Петербург, 2005. – 822 с.
9. Сырецкий, Г.А. Информатика: фундаментальный курс. В 2 т. Т. 2. Информационные технологии и системы : учебн. для вузов / Г.А. Сырецкий. – СПб. : БХВ-Петербург, 2007. – 846 с.
10. Фаронов, В.В. Turbo Pascal : учебн. пособие для вузов / В.В. Фаронов. – СПб. : Питер, 2009. – 366 с.
11. Экономическая информатика : учебник / В.П. Косарев [и др.] ; под ред. В.П. Косарева, Л.В. Еремина. – М. : Финансы и статистика, 2002. – 590 с.

12. Алексеев, Е.Г. Информатика : учебник [Электронный ресурс] / Е.Г. Алексеев, С.Д. Богатырев. – URL : http://inf.e-alekseev.ru/, свободный.
13. Афанасьева, Т.В. Информатика для вас : учебник [Электронный ресурс] / Т.В. Алексеева. – URL : http://pmi.ulstu.ru/new_project/, свободный.
14. Грибанов, В.П. Основы программирования : учебн. пособие [Электронный ресурс] / В.П. Грибанов, О.В. Калмыкова, Р.И. Сорока. – URL : http://declic.narod.ru/ossio/index.html, свободный.
15. Путеводитель по обучению работе с Word 2003 на сайте корпорации Microsoft [Электронный ресурс]. – URL : http://office.microsoft.com/ru-ru/word-help/HA001118952.aspx?CTT=1, свободный.
16. Путеводитель по обучению работе с Excel 2003 на сайте корпорации Microsoft [Электронный ресурс]. – URL : http://office.microsoft.com/ru-ru/excel-help/HA001116128.aspx?CTT=1, свободный.
17. Сергеев, А.Н. Изучаем Паскаль : учебн. пособие [Электронный ресурс] / А.Н. Сергеев. – URL : http://mif.vspu.ru/books/pascal/index.html, свободный.
18. Энциклопедия персонального компьютера и Интернета Кирилла и Мефодия [Электронный ресурс]. – URL : http://mega.km.ru/pc_2001/, свободный.

Приложение 1

Глоссарий

Алгоритм – заранее заданное понятное и точное предписание возможному исполнителю совершить определенную последовательность действий для получения решения задачи за конечное число шагов.

Аппаратное обеспечение (Hardware) – комплекс электронных, электрических и механических устройств, входящих в состав вычислительной системы или сети.

Арифметико-логическое устройство (АЛУ) – блок процессора, который служит для выполнения арифметических и логических операций над данными.

Бит – двоичная единица измерения информации.

Блок-схема – графический способ представления алгоритмов, в котором форма геометрической фигуры (блока) однозначно определяет вид выполняемого действия.

Второе поколение компьютерной техники – электронные вычислительные машины, сконструированные примерно в 1955–1965 гг. Характеризуются использованием в них как электронных ламп, так и дискретных транзисторных логических элементов. Их оперативная память была построена на магнитных сердечниках.

Графический интерфейс – система средств, предназначенная для взаимодействия пользователя с компьютером, основанная на представлении всех доступных пользователю системных объектов и функций в виде графических компонентов экрана (окон, значков, меню, кнопок, списков и т. п.).

Диаграмма – средство наглядного представления данных, облегчающее выполнение сравнений, выявление закономерностей и тенденций данных.

Директивное (императивное) программирование – описывает процесс вычисления в виде инструкций, изменяющих состояние программы. Императивная программа представляет собой последовательность команд, которые должен выполнить компьютер.

Дискретность (прерывность) алгоритма – требование к алгоритму, заключающееся в том, что он должен представ-

лять процесс решения задачи как последовательное выполнение некоторых простых шагов (этапов).

Драйвер – специальная вспомогательная программа, управляющая внешними устройствами ПК или выполнением программ.

Загрузчик операционной системы – специальная программа, предназначенная для инициирования процесса загрузки системы.

Запоминающее устройство с последовательным доступом – может передавать данные только в определённой последовательности. Память на магнитных лентах имеет такой тип доступа.

Запоминающее устройство с произвольным доступом – отличается возможностью передать любые данные в любом порядке. Оперативное запоминающее устройство (ОЗУ) и винчестер – примеры такой памяти.

Инкапсуляция *(в объектно-ориентированном программировании)* – механизм, который объединяет в одно целое данные и методы, применяющиеся к этим данным, и защищает и то и другое от неправильного использования.

Интегрированная среда программирования – система программных средств, используемая программистами для разработки программного обеспечения. Примером интегрированной среды программирования является система программирования Turbo Pascal 7.0.

Интерпретатор – тип транслятора, принцип работы которого основан на построчном переводе каждого оператора программы в машинные коды и выполнении этого оператора.

Интерфейс пользователя – порядок, определяющий процедуры взаимодействия пользователя с компьютерной программой.

Исполнитель алгоритма – некоторая абстрактная или реальная (техническая, биологическая или биотехническая) система, способная выполнить действия, предписываемые алгоритмом. В информатике универсальным исполнителем алгоритмов является компьютер.

Класс объектов *(в объектно-ориентированном программировании)* – множество объектов, имеющих общую структуру. Каждый конкретный объект является ***экземпляром*** класса.

Кодирование – процесс представления данных последовательностью символов иной формы или значения.

Компилятор – тип транслятора, принцип работы которого предполагает преобразование всей программы в модуль на машинном языке после выполнения процедуры проверки текста программы на наличие синтаксических ошибок и их исправления программистом.

Компьютерный блок питания – блок питания, предназначенный для снабжения узлов компьютера электрической энергией. В его задачу входит преобразование сетевого напряжения до заданных значений, их стабилизация и защита от незначительных помех питающего напряжения.

Контекстное меню *(в графическом интерфейсе пользователя)* – меню, открываемое, как правило, при нажатии правой кнопки мыши. В этом меню отображаются команды, которые могут быть применены к выделенному объекту.

Кэш-память – память с большей скоростью доступа, предназначенная для ускорения обращения к данным, содержащимся в основной памяти с меньшей скоростью доступа.

Логическое программирование – концепция и средства программирования, основанные на теории и аппарате математической логики.

Маркер заполнения – элемент интерфейса MS Excel, маленький черный квадрат в правом нижнем углу выделенного диапазона электронной таблицы. Указатель мыши при наведении на маркер заполнения принимает вид черного крестика.

Массив (*в языке программирования*) – однородная структура однотипных данных, хранящихся в последовательных ячейках оперативной памяти. Эта структура должна иметь имя и определять заданное количество данных (элементов).

Массовость алгоритма – требование к алгоритму, заключающееся в том, что он должен быть применим для некоторого класса задач, различающихся лишь исходными данными, которые могут выбираться из некоторой области, называемой областью применимости алгоритма.

Мастер функций MS Excel – система диалоговых окон, облегчающая ввод функций при создании формул. При вводе

функции в формулу диалоговое окно ***Мастер функций*** отображает имя функции, все ее аргументы, описание функции и каждого аргумента, текущий результат функции и всей формулы.

Материнская плата (англ. *motherboard*) – сложная многослойная печатная плата, на которой устанавливаются основные компоненты персонального компьютера (центральный процессор, контроллер ОЗУ и собственно ОЗУ, ПЗУ, контроллеры базовых интерфейсов ввода-вывода).

Меню – список команд или функций, предлагаемых пользователю на выбор.

Микропроцессор – интегральная микросхема, на которой целиком помещается центральный процессор.

Наследование *(в объектно-ориентированном программировании)* – процесс, посредством которого один объект может наследовать свойства другого и добавлять к ним новые свойства, характерные только для него.

Объект *(в объектно-ориентированном программировании)* – совокупность данных (свойств объекта) и связанных с ними методов. Под ***методами объекта*** понимают процедуры и функции, объявление которых включено в описание объекта и которые выполняют те или иные действия.

Объектно-ориентированное программирование – парадигма программирования, в которой основными концепциями являются понятия **объекта** и **класса.**

Окно – основной объект графического пользовательского интерфейса Windows.

Оперативная память (***оперативное запоминающее устройство, ОЗУ***) – память, часть системы памяти ЭВМ, к которой процессор может обратиться за одну операцию.

Оператор программы – предложение языка программирования, задающее полное описание некоторого действия, которое необходимо выполнить.

Операционная система (ОС) – базовый комплекс компьютерных программ, обеспечивающий интерфейс с пользователем, управление аппаратными средствами компьютера, работу с файлами, ввод и вывод данных, а также выполнение прикладных программ и утилит.

Определенность (детерминированность) алгоритма – требование к алгоритму, заключающееся в том, что при выполнении алгоритма с одним и тем же набором исходных данных всякий раз должен получаться один и тот же результат.

Память компьютера – совокупность устройств для хранения программ, вводимой информации, промежуточных результатов и выходных данных, получаемых в результате работы компьютера.

Папка – средство организации и представления системных ресурсов компьютера в операционных системах Windows.

Первое поколение компьютерной техники – машины, созданные на рубеже 50-х годов XX века. В их схемах использовались электронные лампы.

Переменная *(в программировании)* – имя, обозначающее область оперативной памяти компьютера, которая хранит данные. Обращение к этой области выполняется по имени переменной. Значения хранимых данных могут изменяться в процессе выполнения программы.

Персональный компьютер (ПК) – компьютер универсального назначения, рассчитанный на одного пользователя и управляемый одним человеком.

Пиктограмма – небольшое графическое изображение объекта или действия в виде условного значка.

Подпрограмма – часть программы, локализующая процесс выполнения определенных действий, который может быть повторен многократно с помощью механизма вызова.

Полиморфизм *(в объектно-ориентированном программировании)* – свойство, которое позволяет одно и то же имя использовать для решения нескольких технически разных задач. Полиморфизм подразумевает такое определение методов, при котором метод с одним именем может применяться к различным родственным объектам.

Понятность алгоритма – требование к алгоритму, заключающееся в том, что он должен содержать только такие инструкции (команды), которые понятны исполнителю.

Постоянное запоминающее устройство (ПЗУ) – энергонезависимая память, используемая для хранения неизменяемых данных.

Презентация – документ программы Microsoft PowerPoint, сочетание компьютерной анимации, графики, видео, музыки и звукового ряда, которые организованы в единую среду.

Программирование – процесс создания программы для компьютера.

Рабочий стол – основной экран Windows.

Программное обеспечение (Software) – совокупность программ, процедур и правил, а также документации, обеспечивающих функционирование компьютерной системы обработки данных.

Разрядность микропроцессора – длина обрабатываемых микропроцессором машинных слов.

Результативность алгоритма – требование к алгоритму, заключающееся в том, что он должен за конечное число шагов приводить либо к решению поставленной задачи, либо к завершению выполнения с выдачей сообщения о невозможности получить решение.

Структурное программирование – методология разработки программного обеспечения, в основе которой лежит представление программы в виде иерархической структуры блоков.

Тактовая частота – частота периодического электрического сигнала, вырабатываемого специальной микросхемой, называемой генератором тактовых импульсов, входящей в состав микропроцессора.

Текстовый редактор – компьютерная программа, предназначенная для создания, изменения, просмотра на экране и вывода на печать текстовых документов. Все существующие текстовые редакторы условно разделяют на две группы. К первой группе относят текстовые редакторы, ориентированные на операции по вводу и редактированию текста. Операции по форматированию документов в этих редакторах представлены в минимальном объеме. К числу таких программ относится редактор **Блокнот** (**Notepad**), входящий в состав операционной системы Microsoft Windows. Второй тип текстовых редакторов реализует функции форматирования текста, внедрения в него графики, формул, диаграмм и других объектов. Такие редакторы называют ***текстовыми процессорами***. К их числу относится программа **Microsoft Word**.

***Тип данных** (в программировании)* – множество значений, объединенных определенной совокупностью допустимых для них операций и способом представления в памяти компьютера.

Транслятор – программное средство, выполняющее функции по переводу программы, написанной на языке программирования, во внутреннее представление программы в памяти компьютера.

Третье поколение компьютерной техники – электронные вычислительные машины, созданные после 60-х годов XX века. Это семейства машин с единой архитектурой, т. е. программно совместимых. В качестве элементной базы в них используются интегральные схемы, которые также называются микросхемами.

Утилита – программа вспомогательного или служебного назначения.

Файл – поименованная совокупность данных на машиночитаемом носителе информации. ***Файловая система*** – система хранения файлов и организации каталогов.

Формулы MS Excel – выражения, по которым выполняются вычисления в электронной таблице. Формула начинается со знака равенства «=» и может включать функции, ссылки, операторы и константы.

Функциональное программирование – основано на вычислении результатов функций от исходных данных и результатов других функций и не предполагает явного хранения состояния программы.

Центральный процессор – основной рабочий компонент компьютера, который выполняет арифметические и логические операции, заданные программой, управляет вычислительным процессом и координирует работу всех устройств компьютера.

Четвёртое поколение компьютерной техники – поколение компьютерной техники, разработанное после 1970 года.

Шина – аппаратное средство, к которому одинаковым образом подключается группа взаимодействующих друг с другом устройств компьютера.

Электронные таблицы (***табличные процессоры***) – прикладные программы, предназначенные для автоматизации

табличных расчетов. Основное свойство электронных таблиц – автоматический пересчет формул при изменении значений входящих в них операндов.

Язык программирования – формальная знаковая система, используемая для связи человека с компьютером, предназначенная для описания данных (информации) и алгоритмов (программ) их обработки на компьютере.

Ярлык – в операционной системе Windows определяется как файл, содержащий путь к объекту.

Ячейка MS Excel – минимальный адресуемый элемент электронной таблицы.

Приложение 2

Ответы к тестовым заданиям

Ответы к тестовым заданиям раздела «Основы теории информации и кодирования. Арифметические и логические основы устройства компьютеров»

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
b	c	b	a	c	b	c	b	d	c	a	b	d	a	c	a	b	c

Ответы к тестовым заданиям раздела «Основы алгоритмизации и программирования»

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
b	c	d	c	b	c	b	a	d	c	a	b	c	d	b	a

Ответы к тестовым заданиям раздела «Технические и программные средства реализации информационных процессов»

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
d	a	a, b	b	b	d	c	a	a, c, d	b, c, e	b	a	d	b	a	d	b	c

Приложение 3

Вопросы для подготовки к экзамену

1. Понятие информации. Свойства информации. Меры информации.
2. Позиционные системы счисления. Использование двоичной, восьмеричной и шестнадцатеричной систем счисления для представления данных.
3. Представление числовых данных в памяти компьютера.
4. Представление в памяти компьютера данных текстового типа.
5. Представление в памяти компьютера графической информации.
6. Кодирование аудио- и видеоинформации.
7. Логические операции. Построение таблиц истинности.
8. Применение средств алгебры логики для описания функционирования устройств компьютера. Построение логических схем.
9. Этапы решения задачи на компьютере.
10. Понятие алгоритма, свойства алгоритма. Средства представления алгоритмов.
11. Базовые алгоритмические структуры: следование, ветвление, цикл.
12. Алгоритмический язык Паскаль. Структура программы. Типы данных.
13. Алгоритмический язык Паскаль. Ввод и вывод данных. Оператор присваивания. Составной оператор.
14. Алгоритмический язык Паскаль. Операторы организации ветвлений в программе.
15. Алгоритмический язык Паскаль. Операторы цикла.
16. Алгоритмический язык Паскаль. Организация обработки массивов.
17. Алгоритмический язык Паскаль. Организация подпрограмм.
18. Языки программирования высокого уровня: назначение, классификация, области использования.
19. Понятие технологии программирования. Структурное программирование.

20. Объектно-ориентированное программирование: основные понятия, принципы.

21. Виды обеспечения информационных процессов. Понятие и состав вычислительной системы.

22. Основные и дополнительные устройства персонального компьютера.

23. Классификация программного обеспечения компьютеров.

24. Системное программное обеспечение: назначение, структура.

25. Прикладное программное обеспечение: структура, применение в экономике и управлении.

Приложение 4

Квалификационные задания для определения траектории обучения в рамках темы «Технические и программные средства реализации информационных процессов»

Задание 1

(работа с текстовым процессором Microsoft Word)

Ввести с клавиатуры **Текст 1**. Используя копирование и форматирование текста, получить **Текст 2** – **Текст 5**.

Текст 1

К системному блоку компьютера подключаются различные устройства ввода и вывода информации. Кроме монитора, клавиатуры и мыши такими устройствами являются: **принтер** – для вывода на печать текстовой и графической информации; **сканер** – для считывания графической информации; **плоттер** – для вывода графической информации; **стример** – для хранения данных на магнитной ленте.

Текст 2

К системному блоку компьютера подключаются различные устройства ввода и вывода информации. Кроме монитора, клавиатуры и мыши такими устройствами являются:

- ***принтер*** *– для вывода на печать текстовой и графической информации;*
- ***сканер*** *– для считывания графической информации;*
- ***плоттер*** *– для вывода графической информации;*
- ***стример*** *– для хранения данных на магнитной ленте.*

Текст 3

К системному блоку компьютера подключаются различные устройства ввода и вывода информации. Кроме монитора, клавиатуры и мыши такими устройствами являются:

a) принтер – для вывода на печать текстовой и графической информации;

b) сканер – для считывания графической информации;

c) плоттер – для вывода графической информации;

d) стример – для хранения данных на магнитной ленте.

Текст 4

К системному блоку компьютера подключаются различные устройства ввода и вывода информации.

Название устройства	Назначение устройства
принтер	для вывода на печать текстовой и графической информации
сканер	для считывания графической информации
плоттер	для вывода графической информации
стример	для хранения данных на магнитной ленте

Текст 5

К системному блоку компьютера подключаются различные устройства ввода и вывода информации.

Название и назначение устройства	
принтер	*для вывода на печать текстовой и графической информации*
сканер	*для считывания графической информации*
плоттер	*для вывода графической информации*
стример	*для хранения данных на магнитной ленте*

Задание 2

(работа с табличным процессором Microsoft Excel)

1. Создать в Microsoft Excel таблицу расчета статистических показателей по результатам успеваемости учащихся школы за период с 2001 по 2010 год.

В ячейки, содержащие знак вопроса, ввести расчётные формулы:

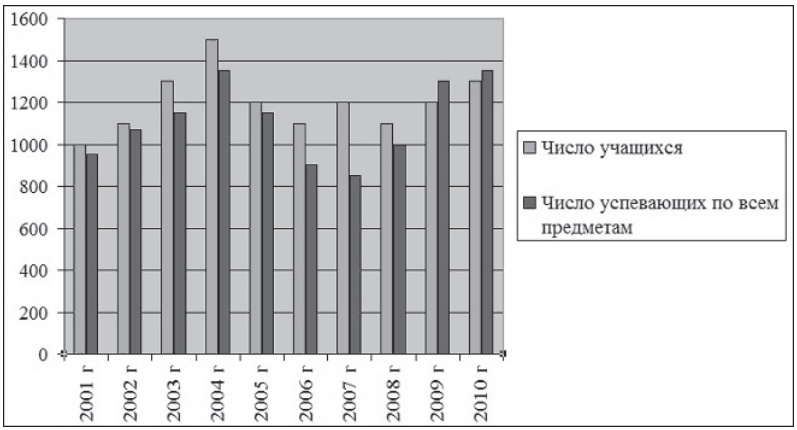
- Процент успеваемости = Число успевающих по всем предметам / Число учащихся;
- Процент качества = Число успевающих на 4 и 5 по всем предметам / Число учащихся;
- Отклонение числа учащихся от среднего числа учащихся = Число учащихся – Среднее значение числа учащихся за весь период.

Для вычисления ***Среднего, наименьшего и наибольшего числа учащихся*** использовать стандартные функции **СРЗНАЧ, МИН, МАКС**.

В столбцах ***Процент успеваемости*** и ***Процент качества*** установить ***процентный формат***.

	A	B	C	D	E	F	G
1	**Статистика успеваемости учащихся школы**						
2	*Год*	*Число учащихся*	*Отклонение числа учащихся от среднего значения*	*Число успевающих по всем предметам*	*Процент успеваемости*	*Число успевающих на 4 и 5 по всем предметам*	*Процент качества*
3	2001 г	1000	?	950	?	500	?
4	2002 г	1100	?	1070	?	600	?
5	2003 г	1300	?	1150	?	800	?
6	2004 г	1500	?	1350	?	850	?
7	2005 г	1200	?	1150	?	700	?
8	2006 г	1100	?	900	?	400	?
9	2007 г	1200	?	850	?	400	?
10	2008 г	1100	?	1000	?	700	?
11	2009 г	1200	?	1300	?	900	?
12	2010 г	1300	?	1350	?	850	?
13	*Итого*	?		?		?	
14	*Среднее значение числа учащихся за весь*				?		
15	*Наименьшее число учащихся*				?		
16	*Наибольшее число учащихся*				?		

2. Построить ***гистограмму***, отражающую изменение ***числа учащихся и числа успевающих по всем предметам за весь период с 2001 по 2010 год***.



Содержание

Учебное издание

Глазова Вера Федоровна
Богданова Анна Владимировна
Панюкова Екатерина Владимировна

ИНФОРМАТИКА

Учебно-методическое пособие

В двух частях

Часть 1

Редактор *Т.Д. Савенкова*
Технический редактор *З.М. Малявина*
Компьютерная верстка: *И.И. Шишкина*
Дизайн обложки: *Г.В. Карасева*

Подписано в печать 05.03.2013. Формат 60×84/16.
Печать оперативная. Усл. п. л. 14,12.
Тираж 100 экз. Заказ № 1-66-11.

Издательство Тольяттинского государственного университета
445667, г. Тольятти, ул. Белорусская, 14