

Под редакцией профессора
Н. В. Макаровой

 **ПИТЕР®**

ИНФОРМАТИКА и ИКТ

Практикум

8-9

Рекомендовано Министерством образования и науки РФ

УЧЕБНО-МЕТОДИЧЕСКИЙ КОМПЛЕКТ ДЛЯ СРЕДНЕЙ ШКОЛЫ

ИНФОРМАТИКА и ИКТ ПРАКТИКУМ 8-9 класс

Учебное пособие



**Под редакцией профессора
Н. В. Макаровой**



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

2010

ББК 32.973.2я7
УДК 004.3(075)
И74

УЧЕБНО-МЕТОДИЧЕСКИЙ КОМПЛЕКТ ДЛЯ СРЕДНЕЙ ШКОЛЫ

Под редакцией профессора Н. В. Макаровой

Авторский коллектив: **Макарова Наталья Владимировна** (руководитель авторского коллектива)
Кочурова Елена Геннадиевна
Кузнецова Ирина Николаевна
Николайчук Галина Семеновна
Нилова Юлия Николаевна
Титова Юлияна Францевна

И74 Информатика и ИКТ. Практикум. 8–9 класс / Под ред. проф. Н. В. Макаровой. — СПб.: Питер, 2010. — 384 с.: ил.

ISBN 978-5-469-01622-9

Практикум является частью учебно-методического комплекта для средней школы. Его цель — обучить школьников работе на компьютере в основных программных средах: операционной системе Windows XP, графическом редакторе Paint, в приложениях пакета Microsoft Office 2003 (текстовом процессоре Word, табличном процессоре Excel, системе управления базой данных Access), технологии работы в сети Интернет. Изучаются основы алгоритмизации и основы программирования в среде Logo.

Освоение информационной технологии на компьютере можно проводить независимо от изучения тем теоретической части, представленной в учебнике «Информатика и ИКТ. 8–9 класс», в процессе выполнения разнообразных заданий от простых к сложным. Практикум построен по модульному принципу.

Пособие можно использовать как для работы в классе, так и для самостоятельных занятий.

© ООО «Лидер», 2010

© Макарова Н. В., 2010

Все права защищены. Никакая часть данной книги не может быть воспроизведена в какой бы то ни было форме без письменного разрешения владельцев авторских прав.

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

ISBN 978-5-469-01622-9

ООО «Лидер», 194044, Санкт-Петербург, пр. Б. Сампсониевский, д. 29а.

Налоговая льгота — общероссийский классификатор продукции ОК 005-93, том 2;
953005 литература учебная.

Подписано в печать 25.08.09. Формат 70×90/16. Усл. п. л. 28,08. Доп. тираж 15 000 экз. Заказ № 18086.

Отпечатано с фотоформ в ОАО «Печатный двор» им. А. М. Горького.
197110, Санкт-Петербург, Чкаловский пр., 15.

Введение

Данный практикум является частью учебно-методического комплекта по дисциплине «Информатика и ИКТ», в состав которого также входят учебник, задачник по моделированию, программа обучения и методические пособия для учителей.

Все разделы практикума представляют собой автономные модули, благодаря чему изучение современных информационных технологий можно начинать с любого раздела.

Цель практикума — научить информационной технологии работы на компьютере в офисных программных средах и дать основы алгоритмизации и программирования.

Последовательно выполняя задания, предложенные в *первой части* практикума, вы научитесь работать в операционной системе Windows XP и приложениях пакета Microsoft Office 2003 (текстовом процессоре Word, табличном процессоре Excel, СУБД Access), в графическом редакторе Paint, а также в программах Outlook Express и Internet Explorer, позволяющих использовать возможности сети Интернет. Переходя от более простых заданий к более сложным, а также выполняя задания для самостоятельной работы, вы освоите основную часть технологических приемов работы в программных средах. Помимо этого предлагаются темы и задания повышенной сложности (отмечены звездочкой).

Во *второй части* практикума вы познакомитесь с основами алгоритмизации и программирования на примере языка ЛОГО, которые осваиваются в процессе решения конкретных задач. Каждая задача имеет описание, словесный алгоритм, блок-схему и программы на трех языках: школьном алгоритмическом, Паскале, Visual Basic.

Материал практикума подобран таким образом, чтобы теоретическая информация постоянно подкреплялась практическими навыками в процессе выполнения заданий на компьютере. Это позволит вам контролировать свои достижения, сравнивая результаты, приведенные в практикуме, с полученными самостоятельно.

Практикум может использоваться не только на занятиях в классе, но и в качестве самоучителя. С его помощью вы можете самостоятельно за короткое время освоить базовую компьютерную технологию и поймете, как и где ее можно применять.

Часть 1

Информационные технологии в прикладных программных средах

- Раздел 1** Освоение системной среды Windows XP
- Раздел 2** Освоение среды графического редактора
- Раздел 3** Освоение среды текстового процессора
- Раздел 4** Освоение среды табличного процессора
- Раздел 5** Освоение системы управления базой данных
- Раздел 6** Коммуникации в глобальной сети Интернет

РАЗДЕЛ 1 Освоение системной среды Windows XP

Тема 1.1	Параметры файла и действия над файлом. . . .	6
Тема 1.2	Параметры папки и действия над папкой. . . .	11
Тема 1.3	Работа с окнами графического интерфейса. . . .	16
Тема 1.4*	Настройка параметров Рабочего стола	21
Тема 1.5	Запуск программ (приложений)	27
Тема 1.6	Работа в Windows как в многозадачной среде	31
Тема 1.7*	Работа с папками и файлами с помощью меню и панели инструментов Проводника	36
Тема 1.8	Технология и способы обмена данными	45
Тема 1.9	Антивирусная защита информации.	54
Тема 1.10*	Создание архивных файлов	60

Тема 1.1 **Параметры файла и действия над файлом**

Выполнив задания этой темы, вы научитесь:

- отображать информацию о файлах разными способами;
- выполнять стандартные действия с файлами.

РЕКОМЕНДАЦИИ УЧИТЕЛЮ

- Перед началом занятия создайте на каждом компьютере рабочую папку и сохраните в ней несколько файлов разных типов.
- Выберите в окне рабочей папки команду меню Сервис ▸ Свойства папки. В открывшемся окне на вкладке Общие установите переключатель Использовать обычные папки Windows, на вкладке Вид отключите флажок Скрывать расширения для зарегистрированных типов файлов.
- Выберите команду меню Вид ▸ Плитка.

Задание 1.1

Откройте рабочую папку и посмотрите, где и как в ней отображена информация о файлах. Измените способ отображения информации о файлах в папке и сравните все доступные способы.

Технология работы

Для изменения способа представления информации о файлах в окне папки можно использовать как команды меню Вид, так и команды раскрывающегося списка кнопки Вид, расположенной на панели инструментов.

1. Отобразите информацию о файлах в папке в виде мелких значков, выбрав команду меню Вид ▸ Значки:



- подведите указатель мыши к пункту меню или кнопке Вид (рис. 1.1) и щелкните левой кнопкой мыши, чтобы открыть список команд;

- наведите указатель мыши на команду Значки (при этом строка команды выделится темным прямоугольником) и щелкните левой кнопкой мыши.

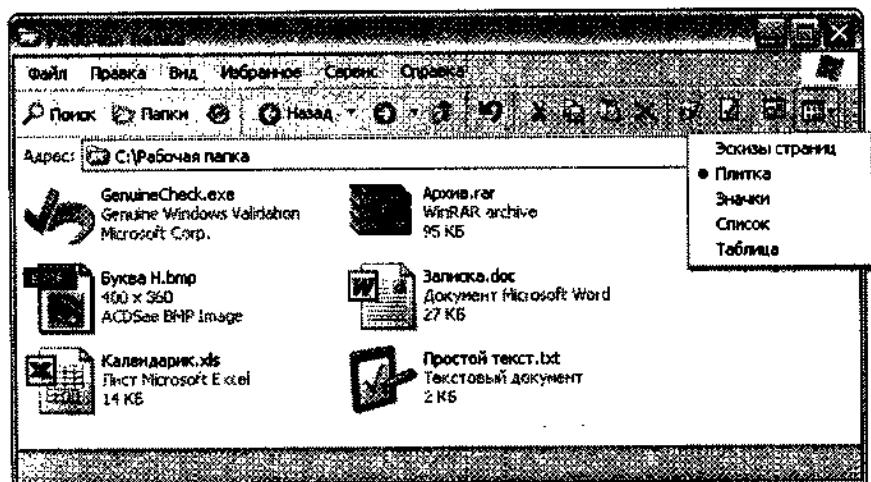


Рис. 1.1. Команды раскрывающегося меню кнопки Вид

2. Отобразите информацию о файлах в папке в виде списка, выбрав команду меню Вид ► Список:
 - подведите указатель мыши к пункту меню или кнопке Вид (см. рис. 1.1) и щелкните левой кнопкой мыши, чтобы открыть список команд;
 - наведите указатель мыши на команду Список (при этом строка команды выделится темным прямоугольником) и щелкните левой кнопкой мыши.
3. Отобразите информацию о файлах в папке в виде крупных значков, выбрав команду меню Вид ► Плитка:
 - подведите указатель мыши к пункту меню или кнопке Вид (см. рис. 1.1) и щелкните левой кнопкой мыши, чтобы открыть список команд;
 - наведите указатель мыши на команду Плитка (при этом строка команды выделится темным прямоугольником) и щелкните левой кнопкой мыши.

4. Отобразите информацию о файлах в папке в виде таблицы, выбрав команду меню Вид ► Таблица:
 - подведите указатель мыши к пункту меню или кнопке Вид (см. рис. 1.1) и щелкните левой кнопкой мыши, чтобы открыть список команд;
 - наведите указатель мыши на команду Таблица (при этом строка команды выделится темным прямоугольником) и щелкните левой кнопкой мыши.
5. Отобразите в папке эскизы файлов, выбрав команду меню Вид ► Эскизы страниц:
 - подведите указатель мыши к пункту меню или кнопке Вид (см. рис. 1.1) и щелкните левой кнопкой мыши, чтобы открыть список команд;
 - наведите указатель мыши на команду Эскизы страниц (при этом строка команды выделится темным прямоугольником) и щелкните левой кнопкой мыши.



При выборе команды Таблица выводится наиболее полная информация о файлах: имя, размер, тип и время последнего сохранения. Команда Плитка отображает избранную информацию о файле, например имя, тип и размер файла. Команды Значки и Список показывают только имена и значки файлов.

Задание 1.2

Посмотрите информацию о файле, открыв окно свойств с помощью контекстного меню.

Технология работы

1. Наведите указатель мыши на значок файла.
2. Вызовите контекстное меню щелчком правой кнопки мыши.
3. Выберите в контекстном меню команду Свойства.
4. Посмотрите информацию о файле в открывшемся окне.
5. Закройте окно свойств щелчком на кнопке Отмена.

Задание 1.3

Выполните стандартные действия с файлами: переместите значок файла, переименуйте файл, создайте копию файла и затем удалите ее.

Технология работы

1. Выберите команду меню Вид ► Значки (см. задание 1.1, п. 1).
2. Переместите значок файла с помощью мыши:
 - наведите указатель мыши на значок файла;
 - нажмите левую кнопку мыши;
 - удерживая нажатой кнопку мыши, переместите значок файла;
 - отпустите кнопку мыши — значок файла останется на новом месте.
3. Создайте с помощью мыши копию файла:
 - наведите указатель мыши на значок файла;
 - нажмите левую кнопку мыши;
 - нажмите на клавиатуре клавишу Ctrl;
 - удерживая нажатыми клавишу Ctrl и левую кнопку мыши, переместите значок файла;
 - отпустите сначала левую кнопку мыши, а затем клавишу Ctrl — в папке появится копия файла;
 - создайте еще несколько копий этого же файла.
4. Переименуйте созданные копии файла:
 - щелкните правой кнопкой мыши на значке файла, чтобы открыть контекстное меню;
 - выберите в меню команду Переименовать — на месте имени файла появится прямоугольная рамка, внутри которой будет выделено старое имя файла;
 - укажите новое имя файла, изменив старое имя файла или заменив его новым;
 - нажмите клавишу Enter;
 - аналогичным образом переименуйте остальные копии файла.
5. Удалите копии файлов:

- щелкните правой кнопкой мыши на значке файла, чтобы открыть контекстное меню;
- выберите в меню команду Удалить;
- в диалоговом окне с запросом о подтверждении удаления файла щелкните на кнопке Да, если вы действительно хотите удалить файл, или на кнопке Нет, если файл удалять не надо;
- удалите все созданные вами копии файла.

Задания для самостоятельной работы

Задание 1.4. Откройте любую папку на вашем компьютере и выберите различные способы представления информации о файлах, хранящихся в папке. Поясните, какая информация отображается в каждом случае.

Задание 1.5. Откройте рабочую папку и выполните следующие действия с одним из хранящихся в ней файлов:

- создайте несколько копий файла;
- переименуйте копии;
- посмотрите свойства новых файлов;
- удалите копии.

Тема 1.2 Параметры папки и действия над папкой

Выполнив задания этой темы, вы научитесь:

- работать с папками;
- получать информацию о папках;
- выполнять стандартные действия с папками.

РЕКОМЕНДАЦИИ УЧИТЕЛЮ

- Перед началом занятия запустите на каждом компьютере программу Проводник, разверните окно Проводника на весь экран и выберите в нем диск С:.
- Выберите команду меню Вид ▸ Плитка.
- Выберите команду меню Вид ▸ Панели инструментов ▸ Настройка и задайте для параметра Текст кнопки значение выборочно текст справа.

Задание 1.6

Научитесь открывать и сворачивать папки в файловой структуре компьютера и просматривать их содержимое.

Технология работы

Чтобы отобразить/скрыть панель Папки с файловой структурой компьютера в левой части окна Проводника, нажмите кнопку-переключатель Папки на панели инструментов.



1. Откройте любую папку на панели Папки. Для этого наведите указатель мыши на значок  слева от имени папки и щелкните на нем левой кнопкой мыши. Папка откроется, а значок слева от имени изменится на .
2. Сверните папку на панели Папки. Для этого наведите указатель мыши на значок  слева от имени папки и щелкните на нем левой кнопкой мыши. Структура папки будет свернута, а значок изменится на .

3. Повторите действия, описанные в пп. 1 и 2, с несколькими папками.
4. Посмотрите содержимое папки, выбрав ее на панели Папки. Для этого щелкните левой кнопкой мыши на ее имени в панели Папки. В правой панели окна Проводника будут отображены все папки и файлы, находящиеся в данной папке.
5. Посмотрите содержимое папки, выбрав ее в правой панели Проводника. Для этого дважды щелкните на ее значке или имени в правой панели Проводника. Все находящиеся в ней папки и файлы будут отображены в правой панели.
6. Отобразите в правой панели Проводника содержимое другой папки, щелчком мыши выбрав ее имя в панели Папки.
7. Повторите действия, описанные в пп. 4–6, с несколькими папками.

Задание 1.7

Получите информацию о параметрах папок, изменяя способ их отображения в окне Проводника.

Технология работы

1. Выберите различные способы отображения информации о папках в окне Проводника:
 - выберите команду меню Вид ► Плитка;
 - выберите команду меню Вид ► Значки;
 - выберите команду меню Вид ► Список;
 - выберите команду меню Вид ► Таблица;
 - выберите команду меню Вид ► Эскизы страниц.

В каком случае информация о папках отображается наиболее полно?
2. Получите информацию о свойствах папки с помощью контекстного меню:
 - вызовите контекстное меню, щелкнув правой кнопкой мыши на имени папки в любой (правой или левой) панели Проводника;

- выберите в контекстном меню команду Свойства;
- закройте окно свойств папки, щелкнув на кнопке Отмена.

Задание 1.8

Выполните стандартные действия с папками: создание папки и копии папки, переименование, удаление. Выполните копирование и перемещение файлов из одной папки в другую.

Технология работы

Перед выполнением задания выберите в меню команду Вид ► Эскизы страниц.

1. Найдите в панели Папки диск C: и выделите его щелчком мыши.
2. Создайте на диске C: папку Children:
 - откройте меню Файл, наведите указатель мыши на строку с командой Создать (при этом строка команды выделится темным прямоугольником) и щелкните на ней левой кнопкой мыши — откроется вложенное меню (подменю);
 - в открывшемся подменю выберите команду Папку — в правой панели Проводника появится значок новой папки, а под ним — прямоугольная рамка для ввода имени с выделенным именем Новая папка;
 - введите имя папки Children и нажмите клавишу Enter.
3. Откройте папку Children и создайте в ней вложенные папки Пример1 и Пример2 (рис. 1.2), воспользовавшись технологией, описанной в п. 2.
4. Сделайте копии папок Пример1 и Пример2:
 - установите указатель мыши на значке папки Пример1;
 - нажмите левую кнопку мыши и клавишу Ctrl на клавиатуре и, не отпуская их, перетащите значок папки на свободное место на правой панели;
 - отпустите сначала кнопку мыши, а затем клавишу Ctrl — в правой панели появится значок копии файла;
 - повторите перечисленные действия для папки Пример2.

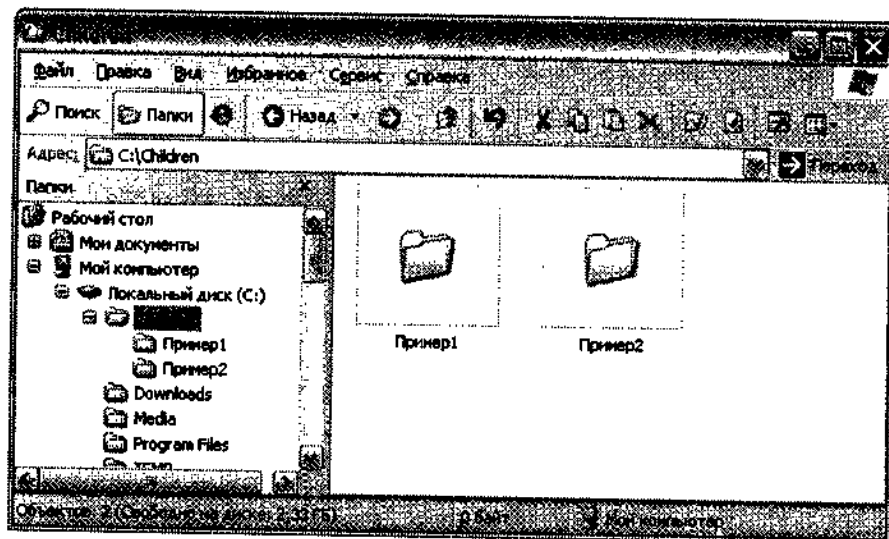


Рис. 1.2. Папка Children со вложенными папками

5. Переименуйте копии папок:

- наведите указатель мыши на значок папки в правой панели Проводника;
- щелчком правой кнопки мыши вызовите контекстное меню;
- в контекстном меню выберите команду Переименовать;
- введите новое имя папки и нажмите клавишу Enter;
- повторите перечисленные действия для второй копии.

6. Удалите копии папок:

- наведите указатель мыши на значок папки в правой панели Проводника;
- щелчком правой кнопки мыши вызовите контекстное меню;
- в контекстном меню выберите команду Удалить;
- в диалоговом окне с запросом о подтверждении удаления папки щелкните на кнопке Да, если вы действительно хотите удалить папку, или на кнопке Нет, если папку удалять не надо;
- повторите перечисленные действия для второй копии.

7. Переместите папку Пример2 в папку Пример1:

- в разделе Папки наведите указатель мыши на значок папки Пример2;
- нажмите левую кнопку мыши и, не отпуская ее, переместите значок папки Пример2 на значок папки Пример1;
- отпустите левую кнопку мыши, когда значок папки Пример1 станет выделенным (имя папки будет выделено серым фоном); если вы все сделали правильно, значок папки Пример2 пропадет из правой панели Проводника;
- откройте папку Пример1 двойным щелчком — вы увидите папку Пример2, которая теперь стала папкой, вложенной в папку Пример1.

8. Скопируйте папку Пример2 в папку Children:

- щелкните на значке  слева от папки Пример1 в панели Папки: в правой панели Проводника вы увидите ее содержимое — папку Пример2;
- наведите указатель мыши на название папки Пример2 в панели Папки;
- нажмите левую кнопку мыши, затем клавишу Ctrl и, не отпуская их, перетащите папку Пример2 на значок папки Children (выше в этой же панели);
- отпустите кнопку мыши и клавишу Ctrl.

Задания для самостоятельной работы

Задание 1.9. Откройте любую папку на вашем компьютере и выберите различные способы представления информации о хранящихся в ней файлах и папках. Поясните, какая информация отображается в каждом случае.

Задание 1.10. Создайте папку Тест, а в ней — папки Тест1, Тест2 и Тест3.

Задание 1.11. Скопируйте в папку Тест1 несколько файлов.

Задание 1.12. Скопируйте папку Тест1 в папку Тест2 и переименуйте копию папки Тест1.

Задание 1.13. Переместите папку Тест3 в папку Тест2.

Тема 1.3 Работа с окнами графического интерфейса

Выполнив задания этой темы, вы научитесь:

- изменять размеры окон;
- перемещать, сворачивать и разворачивать окна.

РЕКОМЕНДАЦИИ УЧИТЕЛЮ

Перед началом занятия откройте на каждом компьютере несколько окон разного типа: окно программы без документов (например, окно Проводника с открытой рабочей папкой), окно программы с документом (например, WordPad), диалоговое окно (например, окно свойств экрана).

Задание 1.14

Выполните стандартные действия с окном:

- ◆ *переместите окно с помощью мыши;*
- ◆ *измените размеры окна;*
- ◆ *измените размеры окон и расположите их на Рабочем столе «мозаикой»;*
- ◆ *посмотрите, какие пункты меню находятся в строке меню каждого из открытых окон и какие команды содержатся в каждом из пунктов меню;*
- ◆ *просмотрите содержимое окна с помощью полос прокрутки;*
- ◆ *сверните окно в значок на панели задач;*
- ◆ *восстановите окно из значка на панели задач;*
- ◆ *закройте все открытые на Рабочем столе окна.*

Технология работы

1. Переместите диалоговое окно с помощью мыши:
 - наведите указатель мыши на заголовок окна;
 - нажмите левую кнопку мыши и, не отпуская ее, передвиньте окно.
2. Измените размеры окна.

Вариант 1

Если окно не развернуто на весь экран, можно произвольно изменять его размеры, передвигая мышью границы (стороны прямоугольной рамки по периметру окна):

- наведите указатель мыши на одну из границ окна — форма указателя изменится на двунаправленную стрелку $\leftrightarrow$, показывающую направление, в котором можно перемещать границу;
- нажмите левую кнопку мыши и, не отпуская ее, переместите указатель мыши в нужном направлении — граница окна переместится вслед за указателем;
- перемещайте указатель, пока окно не примет необходимый размер.



Установив указатель мыши на угол окна, можно изменять одновременно и ширину, и высоту окна.

Вариант 2

Режим, в котором окно занимает весь экран, называется *полноэкранным*. В полноэкранном режиме граница окна не отображается и изменить размеры окна нельзя.

Разверните на весь экран любое окно программы:

- щелкните левой кнопкой мыши на кнопке Развернуть  (рис. 1.3);

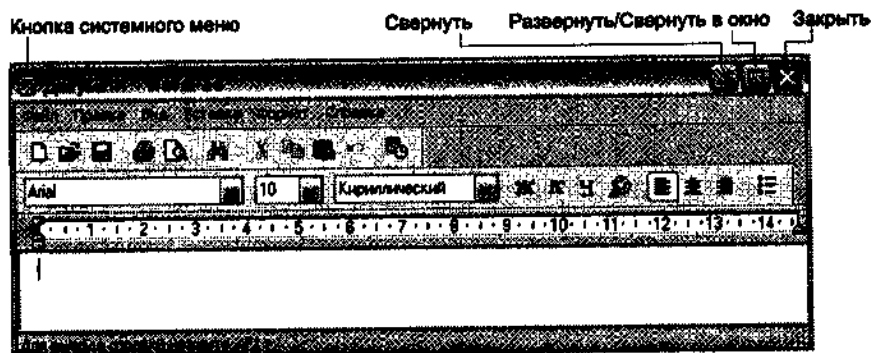


Рис. 1.3. Кнопки управления окном

- щелкните левой кнопкой мыши, чтобы открыть список команд этого пункта меню;
- подведите указатель мыши к нужной команде — она будет выделена темным прямоугольником;
- чтобы отказаться от выполнения выбранной команды, щелкните мышью за пределами открытого меню.



Если команда меню в данный момент недоступна (не может быть выполнена по той или иной причине), ее название в меню будет написано не черным, а серым цветом, и при наведении указателя мыши эта строка не будет выделяться. Такая команда называется **неактивной**.

5. Просмотрите содержимое окна, воспользовавшись полосой прокрутки. Действия, с помощью которых можно управлять полосой прокрутки, приведены в табл. 1.1.

Таблица 1.1. Управление вертикальной полосой прокрутки

Действие	Результат
Щелчок на верхней стрелке полосы прокрутки	Перемещение содержимого окна на одну строку вверх
Щелчок на нижней стрелке полосы прокрутки	Перемещение содержимого окна на одну строку вниз
Щелчок на полосе прокрутки над ползунком	Перемещение содержимого окна на один экран вверх
Щелчок на полосе прокрутки под ползунком	Перемещение содержимого окна на один экран вниз
Перемещение ползунка	Перемещение содержимого окна на расстояние, пропорциональное перемещению ползунка относительно всей полосы

6. Сверните окно в значок на панели задач, используя кнопку управления окном:

- щелкните на кнопке Свернуть в правой части заголовка окна (см. рис. 1.3) — окно исчезнет с Рабочего стола, но на панели задач останется значок этого окна; 

Сверните другое окно, используя кнопку системного меню:

- щелкните мышью на кнопке системного меню (см. рис. 1.3);
- в открывшемся меню выберите команду Свернуть.



Чтобы свернуть сразу все окна, щелкните правой кнопкой мыши на панели задач в нижней части Рабочего стола и выберите в открывшемся контекстном меню команду Показать рабочий стол.

7. Восстановите окно из значка на панели задач:

- щелкните на значке свернутого окна — окно снова появится на Рабочем столе;
- разверните все окна, свернутые в значок на панели задач.

8. Закройте все окна на Рабочем столе любыми из доступных способов:

- с помощью кнопки Закр^ыть  (см. рис. 1.3);
- с помощью команды Закр^ыть системного меню;
- с помощью команды меню Файл > Выход.

Задания для самостоятельной работы

Задание 1.15. Запустите любую программу, например игру. Отобразите окно с этой программой сначала в полноэкранном режиме, а затем в обычном. Измените размеры окна программы, а затем переместите его в другое место. Выполните это задание с несколькими окнами разных программ.

Задание 1.16. Сверните все открытые окна в значки, а затем восстановите их на Рабочем столе.

Задание 1.17. Измените размеры нескольких окон и расположите их на Рабочем столе «мозаикой».

Задание 1.18. Просмотрите содержимое нескольких окон с помощью полос прокрутки.

Тема 1.4* **Настройка параметров Рабочего стола**

Выполнив задания этой темы, вы научитесь:

- ⇒ настраивать различные параметры Рабочего стола: фоновый рисунок, заставку и т. д.;
- ⇒ устанавливать системные дату и время.

Задание 1.19*

Измените следующие параметры Рабочего стола: фоновый рисунок, цветовую схему, заставку.

Технология работы

Параметры Рабочего стола изменяются в диалоговом окне Свойства: Экран (рис. 1.5). Чтобы открыть его, щелкните правой кнопкой мыши на любом свободном месте Рабочего стола и в контекстном меню выберите команду Свойства.

1. Измените фоновый рисунок Рабочего стола:

- выберите вкладку Рабочий стол, щелкнув мышью на ее названии;
- просмотрите с помощью полосы прокрутки список графических файлов для Рабочего стола в поле Фоновый рисунок;
- щелчком мыши на названии графического файла выберите рисунок, например Windows XP (рис. 1.5);
- выберите способ расположения рисунка на Рабочем столе: щелкните на кнопке  справа от поля Расположение и выберите в раскрывающемся списке одно из значений; выберите значение По центру, если вы хотите поместить рисунок в центре экрана; выберите значение Замостить, если вам надо разместить выбранный рисунок мозаикой по всему экрану; выберите значение Растянуть, если вы хотите, чтобы рисунок заполнил весь Рабочий стол;
- посмотрите на рисунок предварительного просмотра, чтобы увидеть, как будет выглядеть Рабочий стол с данным фоновым рисунком при указанном расположении;

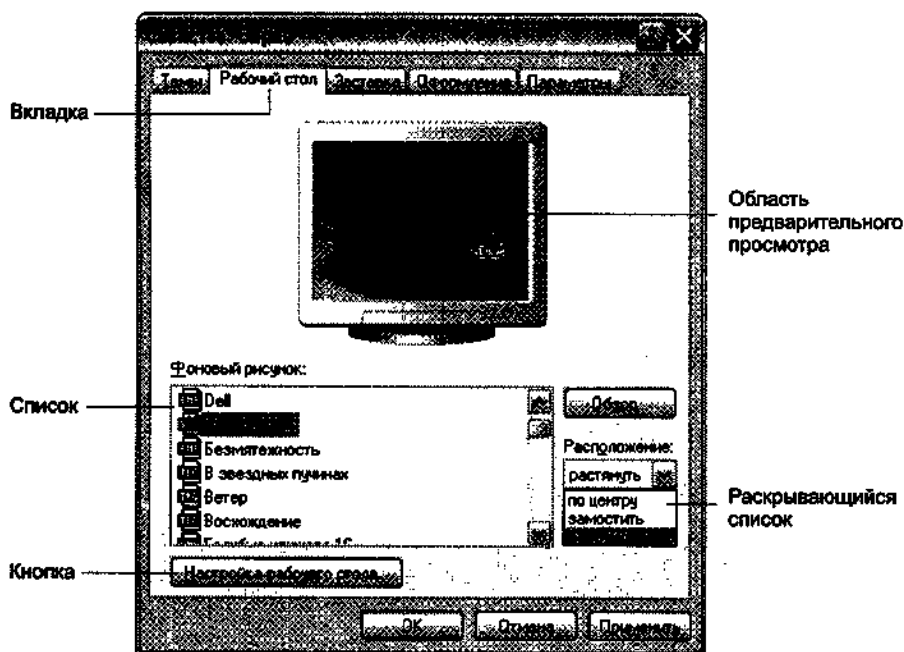


Рис. 1.5. Окно свойств экрана

- выбрав фоновый рисунок, который вы хотите оставить на Рабочем столе, щелкните на кнопке Применить, чтобы сделанные изменения вступили в силу (диалоговое окно Свойства: Экран останется открытым).

2. Измените цветовую схему Рабочего стола:

- выберите вкладку Оформление, щелкнув мышью на ее названии;
- выберите в списке Окна и кнопки один из стилей оформления окон — Классический стиль (как в Windows 2000 и ранее) или Стиль Windows XP;
- выбирайте в списке Цветовая схема различные цветовые схемы и смотрите, как изменяется оформление окон, изображенных на рисунке предварительного просмотра;
- выбрав любую из понравившихся вам цветовых схем, щелкните на кнопке Применить;

- повторите перечисленные действия несколько раз и посмотрите, как изменяется оформление окон на Рабочем столе при выборе разных стилей и цветовых схем;
- выбрав понравившуюся вам цветовую схему, щелкните на кнопке Применить, чтобы сделанные изменения вступили в силу.

3. Измените заставку Рабочего стола.

Заставка — это анимированное или, реже, неподвижное изображение, которое появляется на экране, если в течение определенного времени никто не пользуется ни клавиатурой, ни мышью. При нажатии любой клавиши на клавиатуре и при перемещении мыши восстанавливается прежний вид Рабочего стола.

Для установки или смены заставки Рабочего стола выполните следующие действия:

- выберите вкладку Заставка, щелкнув мышью на ее названии;
- по очереди выбирайте в раскрывающемся списке Заставка различные заставки и смотрите на рисунке предварительного просмотра, как они будут выглядеть на экране;
- чтобы посмотреть заставку в полноэкранном режиме, нажмите кнопку Просмотр; для выхода из полноэкранного режима щелкните мышью в любом месте экрана;
- укажите в поле Интервал время, через которое должна появляться заставка, если компьютер не используется, например, 15 минут;
- оставьте в поле Заставка понравившуюся вам заставку и щелкните на кнопке Применить.

4. Настройте параметры выбранной вами заставки. Рассмотрим настройку параметров на примере заставки Бегущая строка:

- выбрав в поле Заставка значение Бегущая строка, щелкните на кнопке Параметры — откроется окно настройки параметров заставки (рис. 1.6);
- выберите положение текста на экране с помощью переключателя Положение;

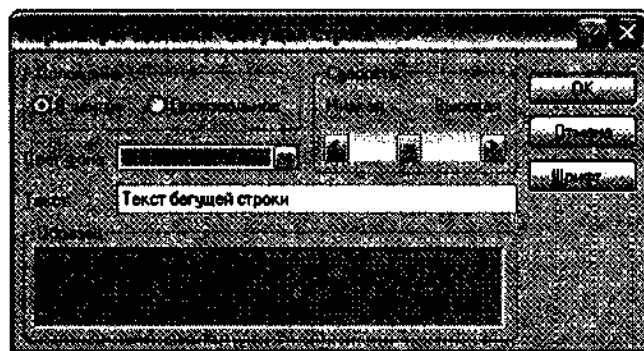


Рис. 1.6. Окно настройки параметров заставки Бегущая строка

- выберите цвет фона в поле Цвет фона;
- выберите скорость перемещения текста, передвинув мышью ползунок Скорость в нужную сторону;
- в текстовом поле Текст напечатайте надпись, которая должна появляться на экране;
- посмотрите в поле Образец, как будет выглядеть заставка, и при необходимости исправьте настройки;
- щелкните на кнопке ОК, чтобы сохранить сделанные настройки и закрыть окно Параметры заставки "Бегущая строка".

После завершения всех настроек щелкните на кнопке ОК, чтобы закрыть диалоговое окно свойств экрана и сохранить сделанные изменения.

Задание 1.20*

Установите системное время и текущую дату. Выполните переключение раскладки клавиатуры и языка ввода.

Технология работы

1. Установите системное время и текущую дату.

Многие программы используют во время работы информацию о текущем времени и дате. Например, текстовый редактор может автоматически вставить дату в письмо. Необходимо периодически сверять показания системных часов с реальным значением точного времени.

Откорректируйте системные время и дату:

- дважды щелкните мышью на часах в правом нижнем углу панели задач — откроется диалоговое окно Свойства: Дата и время (рис. 1.7);
- укажите месяц, выбрав нужное значение в раскрывающемся списке;
- укажите год, изменяя значение с помощью стрелок вверх и вниз справа от поля со значением года;
- щелчком мыши укажите дату, выбрав ее в календарике;
- щелкните мышью в поле с системным временем;
- переместите текстовый курсор на ту позицию, где следует произвести изменения (часы, минуты, секунды);
- удалите старое значение, используя клавиши Backspace или Delete на клавиатуре, и введите новое;
- сохраните изменения, нажав кнопку ОК.

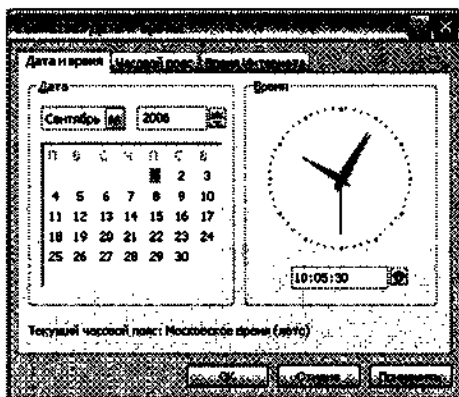


Рис. 1.7. Окно настройки даты и времени

2. Переключите раскладку клавиатуры и язык ввода текста.

Клавиатуру можно переключать в режимы ввода букв разных алфавитов, например кириллицы (русский язык) или латиницы (английский язык). Для переключения клавиатуры из одного режима в другой выполните следующие действия:

- щелкните левой кнопкой мыши на индикаторе клавиатуры Ru или En на панели задач;
- в появившемся меню выберите нужный язык ввода.



Удобнее всего переключать раскладку клавиатуры с помощью комбинации клавиш, например Ctrl+Shift.

Задания для самостоятельной работы

Задание 1.21. Измените оформление Рабочего стола, выбирая для него разные фоновые рисунки.

Задание 1.22. Измените заставку Рабочего стола, выбирая разные цветовые схемы и задавая для них разные параметры.

Задание 1.23. Познакомьтесь с настройками на других вкладках окна Свойства: Экран, не применяя их для окончательного оформления экрана.

Тема 1.5 **Запуск программ (приложений)**

Выполнив задания этой темы, вы научитесь:

- ➡ запускать приложения из меню Пуск;
- ➡ запускать приложения из программы Проводник;
- ➡ запускать приложения из системной папки Мой компьютер.

Задание 1.24

1. *Запустите из меню Пуск графический редактор Paint.*
2. *Запустите из меню Пуск программу Калькулятор.*
3. *Создайте на диске C: папку Игры и скопируйте в нее несколько программ, например игру Сапер и программу Блокнот.*
4. *Из программы Проводник запустите игру Сапер и программу Блокнот, находящиеся в папке Игры.*
5. *Из папки Мой компьютер запустите любую программу, находящуюся в папке Игры.*
6. *Завершите работу в среде Windows и выключите компьютер.*

Технология работы

1. Запустите графический редактор Paint из меню Пуск:
 - щелкните на кнопке Пуск на панели задач;
 - выберите щелчком мыши пункт меню Все программы — откроется вложенное меню (подменю) со списком установленных на компьютере программ;
 - щелчком мыши выберите в подменю пункт Стандартные — откроется вложенное меню со списком стандартных программ Windows;
 - щелкните левой кнопкой мыши на строке с названием графического редактора Paint (рис. 1.8);
 - закройте программу Paint, используя команду Файл ► Выход.

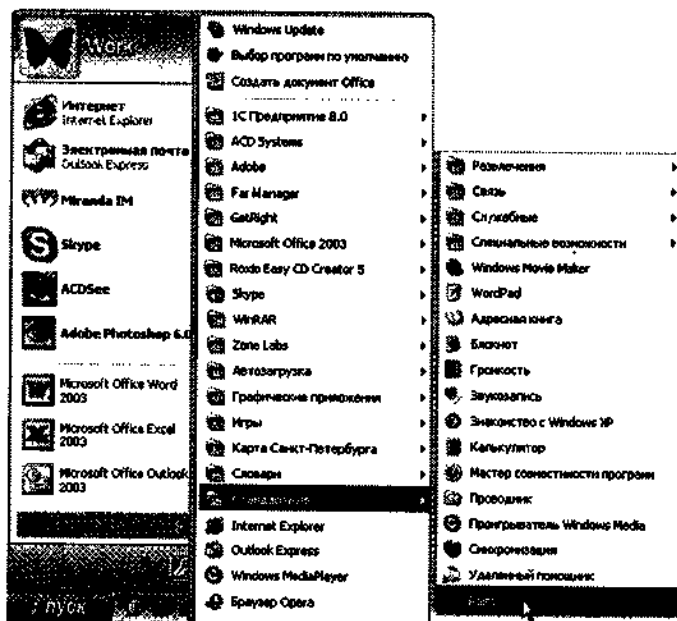


Рис. 1.8. Запуск графического редактора Paint из меню Пуск

2. Запустите Калькулятор из меню Пуск:

- щелкните на кнопке Пуск на панели задач;
- выберите щелчком мыши пункт меню Все программы;
- выберите в подменю пункт Стандартные;
- щелкните левой кнопкой мыши на строке с названием программы Калькулятор;
- закройте программу Калькулятор с помощью кнопки  управления окном Закройте.



Последовательность команд, выбираемых одна за другой во вложенных меню, принято обозначать стрелками, например: Все программы ▶ Стандартные ▶ Калькулятор.

- ## 3. Создайте на диске C: папку Игры, воспользовавшись технологией, описанной в теме 1.2 (задание 1.8), и скопируйте в нее несколько программ, например игру Сапер (файл winmine.exe)

и программу Блокнот (файл notepad.exe) из папки C:\Windows\System32.

4. С помощью программы Проводник запустите игру Сапер из папки Игры (рис. 1.9):

- щелкните на кнопке Пуск на панели задач;
- запустите Проводник, выбрав в меню команду Все программы ► Стандартные ► Проводник;
- найдите на диске C: папку Игры и откройте ее щелчком мыши;
- запустите игру Сапер двойным щелчком на значке исполняемого файла winmine.exe;
- закройте программу командой меню Игра ► Закрывать.

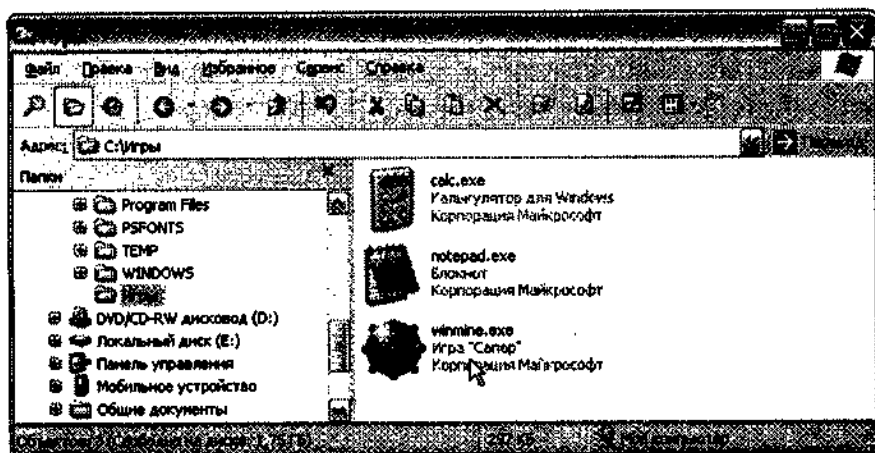


Рис. 1.9. Запуск приложения из программы Проводник

5. Запустите программу Блокнот, находящуюся в папке Игры, используя системную папку Мой компьютер:

- дважды щелкните на значке Мой компьютер на Рабочем столе;
- выберите в списке папок и устройств локальный диск C::
- найдите и откройте на диске C: папку Игры;
- запустите приложение Блокнот двойным щелчком на значке исполняемого файла notepad.exe;

- закройте программу Блокнот, щелкнув на кнопке управления окном Закреть. 
- 6. Завершите работу в среде Windows и выключите компьютер:
 - щелкните на кнопке Пуск на панели задач;
 - выберите в нижней части меню кнопку Выключение; 
 - в открывшемся окне выберите команду Выключение.

Задания для самостоятельной работы

Задание 1.25. Скопируйте программу Калькулятор (calc.exe) из папки C:\Windows\System32 в папку Игры и запустите ее разными способами:

- из меню Пуск;
- из программы Проводник;
- из системной папки Мой компьютер.

Задание 1.26. Перезагрузите компьютер, используя  команду меню Пуск ► Выключение и кнопку Перезагрузка.

Задание 1.27. Завершите работу в среде Windows и выключите компьютер.

Тема 1.6 Работа в Windows как в многозадачной среде

Выполнив задания этой темы, вы научитесь:

- ⇒ работать в программе Калькулятор;
- ⇒ работать в текстовом редакторе WordPad;
- ⇒ переключаться между приложениями.

Задание 1.28

Запустите программу Калькулятор и вычислите количество минут в году.

Программа Калькулятор позволяет выполнять как простые арифметические вычисления, так и более сложные инженерные и статистические расчеты. В первом случае используется обычный интерфейс (вид) программы, во втором — инженерный.

Рассмотрите обычный интерфейс программы Калькулятор (рис. 1.10). Работа с программой происходит так же, как и при использовании обычного карманного калькулятора. Ввод чисел, операций и команд осуществляется с помощью мыши или с клавиатуры.

- ◆ При вводе чисел и операций с клавиатуры нажимайте соответствующие цифровые клавиши или знаки операций.
- ◆ При вводе информации с помощью мыши указывайте цифру или действие щелчком на соответствующей кнопке в окне программы.
- ◆ Для получения результата нажмите кнопку со знаком равенства или клавишу Enter.

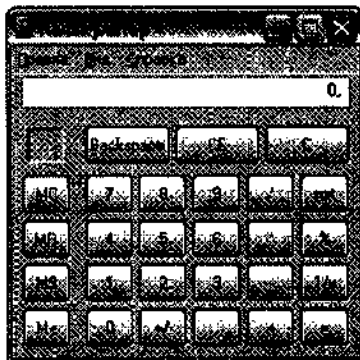


Рис. 1.10. Окно программы Калькулятор (обычный вид)

Технология работы

1. Запустите программу Калькулятор, выбрав в меню Пуск команду Все программы ► Стандартные ► Калькулятор.

2. Выберите команду меню Вид ► Обычный.
3. Вычислите с помощью Калькулятора количество минут в году, перемножив количество суток в году, количество часов в сутках и количество минут в часе:
 - введите с помощью мыши число 365, щелкнув в окне Калькулятора на кнопках с цифрами 3, 6 и 5, — число 365 появится в строке ввода;
 - введите с помощью мыши знак умножения, щелкнув на кнопке с символом ×;
 - введите с клавиатуры число 24;
 - введите знак умножения, нажав на клавиатуре сочетание клавиш Shift+8 (сначала клавишу Shift и затем, не отпуская ее, клавишу 8);
 - введите число 60 с клавиатуры или при помощи мыши;
 - введите знак равенства, щелкнув мышью на кнопке со знаком =, и прочитайте результат вычислений в строке ввода.
4. Закройте программу Калькулятор щелчком на кнопке управления окном Закрыть.

Задание 1.29

Запустите текстовый редактор WordPad, создайте и сохраните документ с именем Набор.txt.

Программа WordPad предназначена для создания, редактирования и печати текстов. Она не обеспечивает поддержки таких функций, как проверка орфографии, автоматическая расстановка переносов в словах, использование словаря синонимов, вставка таблиц, но ее возможностей достаточно для создания текстового документа с простым форматированием.

Окно текстового редактора WordPad представлено на рис. 1.11, назначение пунктов меню приведено в табл. 1.2.

При подготовке текстового документа выполняются следующие шаги:

- ◆ ввод текста;
- ◆ редактирование текста (исправление ошибок);

- ◆ форматирование текста;
- ◆ сохранение документа.

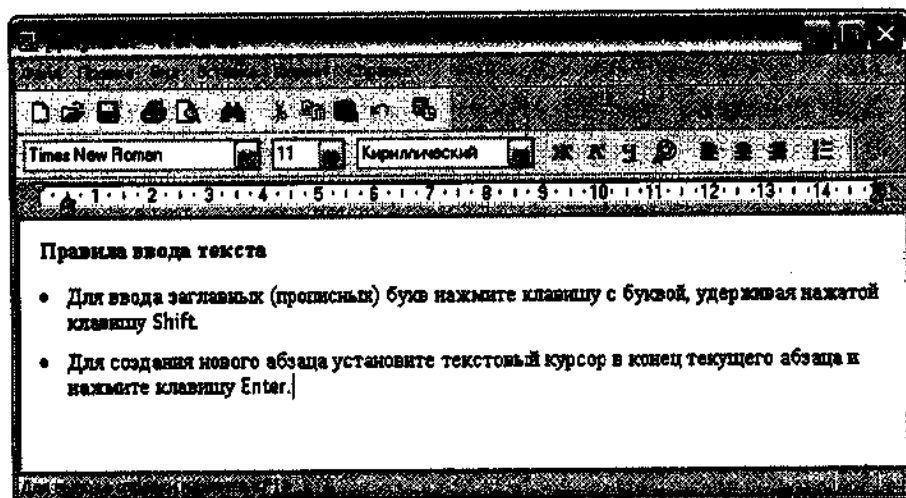


Рис. 1.11. Окно текстового редактора WordPad

Таблица 1.2. Назначение меню текстового редактора WordPad

Меню	Назначение команд меню
Файл	Действия с файлом (Создать, Открыть, Сохранить, Печать, Предварительный просмотр), настройка параметров файла, завершение работы программы
Правка	Операции с текстом (Вырезать, Копировать, Вставить, Очистить, Выделить все), поиск и замена фрагментов текста, отмена последнего действия
Вид	Выбор отображаемых элементов интерфейса окна программы (таких как панель инструментов, панель форматирования, линейка, строка состояния), настройка параметров отображения текста в окне программы
Вставка	Вставка в текст даты и времени, внедрение объектов из других приложений
Формат	Изменение форматирования символов и абзацев
Справка	Справочные сведения о программе WordPad

Правила ввода текста

- ◆ Для ввода заглавных (прописных) букв нажмите клавишу с буквой, удерживая нажатой клавишу Shift.
- ◆ Для создания нового абзаца установите текстовый курсор в конец текущего абзаца и нажмите клавишу Enter.
- ◆ Для создания пустой строки установите текстовый курсор в конец текущего абзаца и нажмите клавишу Enter два раза.
- ◆ Чтобы удалить символ *слева* от курсора (*перед* курсором), нажмите клавишу Backspace.
- ◆ Чтобы удалить символ *справа* от курсора (*после* курсора), нажмите клавишу Delete.

Технология работы

1. Создайте на диске C: папку WordPad.
2. Запустите программу WordPad, выбрав в меню Пуск команду Все программы ► Стандартные ► WordPad.
3. Создайте текстовый документ по образцу на рис. 1.12.
4. Сохраните документ:

- выберите в меню команду Файл ► Сохранить как;
- в открывшемся окне выберите папку WordPad;
- в поле Имя файла введите *Набор*;
- в поле Тип файла выберите значение Текстовый документ;
- нажмите кнопку ОК;
- закройте программу WordPad.

Набор для переключения		
1. Теряем	4 р.	5 шт.
2. Алкоголь для восстановления	32 р.	1 шт.
3. Шерстяная ручка	2 р. 50 к.	2 шт.
4. Карандаш простой	1 р. 50 к.	3 шт.
5. Карандаш цветной	30 р.	1 шт.
6. Линейка	4 р.	1 шт.
7. Кран	20 р.	1 шт.
Итого:	117 р.	15 предм.

Рис. 1.12. Образец текстового документа (к заданию 1.29)

Задание 1.30

Выполните переключение между несколькими запущенными приложениями.

Технология работы

1. Запустите программы Калькулятор, WordPad, Paint; каждая из программ будет запущена в своем окне.

2. Выполните переключение между задачами, щелкая мышью на любом видимом участке окна программы.
3. Выполните переключение между задачами, щелкая мышью на значке программы на панели задач.
4. Выполните переключение между задачами, используя комбинацию клавиш Alt+Tab:
 - нажмите на клавиатуре клавишу Alt и, не отпуская ее, нажмите клавишу Tab — появится окно, в котором значками будут обозначены все запущенные программы, причем один из значков будет выделен рамкой;
 - нажимайте клавишу Tab, пока значок программы, в которую вам надо переключиться, не окажется обведенным рамкой;
 - отпустите клавишу Alt.
5. Закройте все запущенные вами программы.

Задания для самостоятельной работы

Задание 1.31. Запустите программу Калькулятор и вычислите количество часов и минут в одном столетии.

Задание 1.32. Запустите программу Калькулятор и вычислите площадь вашей квартиры.

Задание 1.33. Запустите программу WordPad и напишите в ней письмо своему другу; сохраните созданный файл.

Задание 1.34. Осуществите переключение между запущенными приложениями всеми известными вам способами.

Тема 1.7* **Работа с папками и файлами с помощью меню и панели инструментов Проводника**

Выполнив задания этой темы, вы научитесь:

- ➡ совершать стандартные действия с папками и файлами при помощи команд меню;
- ➡ использовать кнопки панели инструментов.

Задание 1.35*

Определите тип и размер свободного пространства накопителей, установленных на вашем компьютере. Посмотрите, какие папки и файлы хранятся на дисках.

Технология работы

1. Определите, какие накопители установлены на вашем компьютере.

Вариант 1

- Откройте папку Мой компьютер, дважды щелкнув мышью на его значке; по виду значка и букве, сопровождающей значок, можно судить о типе накопителя.

Вариант 2

- Запустите программу Проводник, выбрав в меню Пуск команду Все программы ► Стандартные ► Проводник;
- в левой панели окна программы Проводник под значком папки Мой компьютер перечислены все накопители; по виду значка и букве рядом с ним можно судить о типе накопителя.

2. Определите количество свободного места на диске.

Вариант 1

- Откройте папку Мой компьютер, дважды щелкнув мышью на его значке;

- щелкните левой кнопкой мыши на имени диска или съемного накопителя (если вы выделяете съемный накопитель, например карту памяти, проверьте, чтобы он был установлен в устройство для чтения);
- в строке состояния в нижней части окна выводится информация об общей емкости накопителя и имеющемся свободном пространстве.

Вариант 2

- Запустите программу Проводник, выбрав в меню Пуск команду Все программы ► Стандартные ► Проводник;
 - вызовите контекстное меню, щелкнув правой кнопкой мыши на значке накопителя;
 - выберите пункт меню Свойства;
 - в открывшемся окне на вкладке Общие посмотрите информацию о размере диска и имеющемся свободном пространстве;
 - закройте окно, щелкнув на кнопке Отмена.
3. Посмотрите, какие папки и файлы хранятся на дисках.

Вариант 1

- Откройте папку Мой компьютер;
- откройте папку накопителя двойным щелчком и посмотрите, какие в нем хранятся папки и файлы;
- закройте папку накопителя.

Вариант 2

- Запустите программу Проводник, выбрав в меню Пуск команду Все программы ► Стандартные ► Проводник;
- откройте папку накопителя двойным щелчком и посмотрите, какие в нем хранятся папки и файлы;
- закройте окно программы Проводник.

Задание 1.36*

Создайте на диске C: папку Новая, а в ней — папки Моя папка и Общая папка (рис. 1.13).

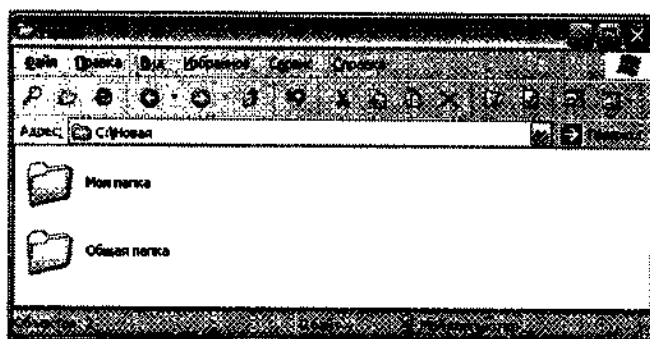


Рис. 1.13. Окно папки Новая

Технология работы

1. Создайте на диске C: папку Новая:

- откройте папку Мой компьютер и выберите диск C:;
- выберите в меню команду **Файл** ▶ **Создать** ▶ **Папку** (рис. 1.14) — на диске появится значок новой папки с именем Новая папка;
- измените имя созданной папки, оставив название Новая;
- нажмите клавишу Enter.

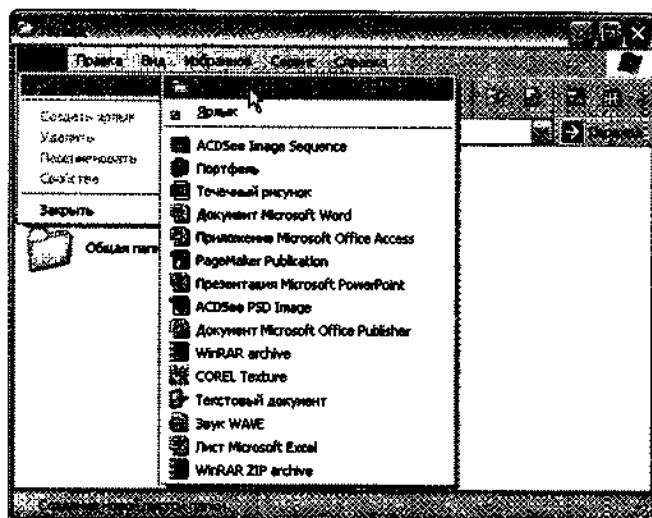


Рис. 1.14. Создание папки с помощью команды меню Файл

2. Создайте папки **Моя папка** и **Общая папка** в папке **Новая**:

- откройте папку **Новая**, дважды щелкнув на значке папки;
- создайте папку **Моя папка** и папку **Общая папка**, используя технологию работы, описанную в п. 1.

Задание 1.37*

Выполните в программе **Проводник** стандартные действия с папками и файлами, воспользовавшись командами **Вырезать**, **Копировать**, **Вставить** из меню **Правка** (рис. 1.15).

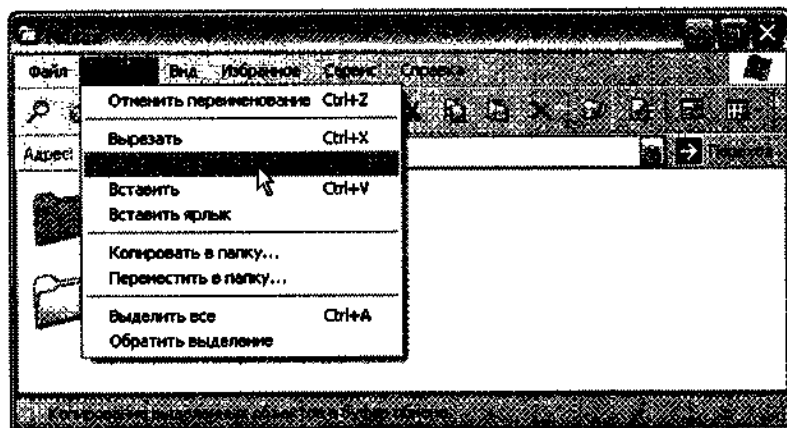


Рис. 1.15. Команды меню **Правка**

Технология работы

1. Запустите программу **Проводник** и откройте в нем папку **Новая**.
2. Переместите папку **Моя папка** в папку **Общая**:
 - откройте папку **Новая**;
 - выделите щелчком мыши папку **Моя папка**;
 - выберите в меню команду **Правка** ► **Вырезать** — папка **Моя папка** будет помещена в буфер обмена;
 - перейдите в папку **Общая папка**;

- выберите в меню команду Правка ▸ Вставить — папка Моя папка будет вставлена из буфера обмена в папку Общая папка и удалена из папки Новая.
3. Скопируйте папку Моя папка из папки Общая папка в папку Новая:
- откройте папку Общая папка;
 - выделите папку Моя папка;
 - выберите в меню команду Правка ▸ Копировать;
 - перейдите в папку Новая;
 - выберите в меню команду Правка ▸ Вставить.
4. Скопируйте любой файл с диска C: в папку Новая:
- выберите в окне Проводника диск C: и выделите любой файл;
 - выберите команду меню Правка ▸ Копировать;
 - перейдите в папку Новая;
 - выберите команду меню Правка ▸ Вставить.
5. Скопируйте папку Новая на съемный носитель, например на дискету или карту памяти:
- вставьте дискету в дисковод;
 - выберите в окне Проводника диск C:;
 - выделите папку Новая;
 - выберите команду меню Правка ▸ Копировать;
 - выберите в окне Проводника диск A: (дисковод);
 - выберите команду меню Правка ▸ Вставить.
6. Удалите файл из папки Новая, папку Моя папка из папки Общая папка, папку Новая с диска A::
- откройте папку Новая на диске C:;
 - выделите файл;
 - выберите команду меню Файл ▸ Удалить;
 - подтвердите удаление;
 - откройте папку Общая папка;
 - выделите папку Моя папка;
 - выберите команду меню Файл ▸ Удалить;

- подтвердите удаление;
- выделите папку Новая на диске A;
- выберите команду меню Файл ► Удалить;
- подтвердите удаление.

7. Переименуйте папки Моя папка и Общая папка:

- откройте папку Новая;
- выделите папку Моя папка;
- выберите команду меню Файл ► Переименовать;
- в поле ввода введите новое имя — Моя папка1;
- выделите папку Общая папка;
- выберите команду меню Файл ► Переименовать;
- в поле ввода введите новое имя — Общая папка1.

Задание 1.38*

Выполните стандартные действия с папками и файлами, используя кнопки панели инструментов.

Панель инструментов находится в верхней части окна под строкой меню (рис. 1.16). Каждая кнопка на панели инструментов соответствует определенной команде, которая выполняется при щелчке на данной кнопке. Кнопки, соответствующие командам, которые не могут быть выполнены в данный момент, отображаются серым цветом.

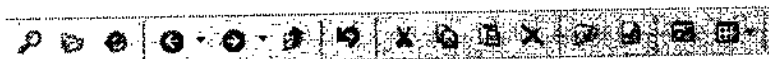


Рис. 1.16. Панель инструментов

Действие любой кнопки можно узнать из подсказки, появляющейся, если на кнопку навести указатель мыши. Назначение кнопок панели инструментов Проводника приведено в табл. 1.3.

Если в окне Проводника не отображается панель инструментов, выберите команду меню Вид ► Панели инструментов ► Обычные кнопки.

Таблица 1.3. Назначение кнопок панели инструментов Проводника

Кнопка	Название	Действие
	Поиск	Отобразить/скрыть панель поиска
	Папки	Отобразить/скрыть панель со структурой папок
	Журнал	Отобразить/скрыть Журнал (список открывавшихся за последнее время папок, файлов и страниц Интернета)
	Назад	Перейти назад к предыдущей просмотренной папке
	Вперед	Перейти вперед к следующей просмотренной папке
	Вверх	Перейти к вышестоящей папке, непосредственно в которой находится текущая папка
	Отменить	Отменить последнее действие с файлом или папкой
	Вырезать	Копировать выделенный объект в буфер обмена и удалить его с текущего места после выполнения команды вставки
	Копировать	Копировать выделенный объект в буфер обмена
	Вставить	Вставить объект из буфера обмена
	Удалить	Удалить выделенный объект
	Свойства папки	Настроить вид папки
	Свойства	Показать окно свойств выделенного объекта
	Во весь экран	Развернуть панели Проводника на весь экран/Вернуться в обычный режим
	Вид	Выбрать способ отображения информации о файлах и папках в правой панели Проводника

Технология работы

1. Выберите различные способы отображения информации о файлах и папках в правой панели Проводника:

- нажмите кнопку Вид;
- выберите в раскрывшемся списке один из способов отображения информации о файлах и папках;
- повторите указанные действия, но выберите другие способы отображения информации.



2. Получите информацию о свойствах файла:

- выберите файл;
- нажмите кнопку Свойства — откроется окно свойств файла;
- посмотрите информацию о файле;
- закройте окно свойств щелчком на кнопке Закрыть.



3. Используя кнопки панели инструментов, выполните действия, аналогичные указанным в задании 1.37 (пп. 2–6), используя вместо папки Моя папка папку Моя папка1, вместо папки Общая папка — папку Общая папка1.

Задание 1.39*

Восстановите удаленные папки.

Технология работы

1. Восстановите удаленные папки Моя папка и Общая папка:

- откройте системную папку Корзина, дважды щелкнув на ее значке на Рабочем столе;
- выделите папку Моя папка;
- выберите в меню команду Файл ► Восстановить;
- выделите папку Общая папка;
- выберите команду меню Файл ► Восстановить;
- закройте Корзину, щелкнув на кнопке управления окном Закрыть.

2. Удалите окончательно файлы, находящиеся в Корзине:

- откройте Корзину, дважды щелкнув на ее значке на Рабочем столе;
- выберите в меню команду Файл ► Очистить корзину;
- закройте Корзину, щелкнув на кнопке управления окном Закрыть.

Задания для самостоятельной работы

Задание 1.40. Определите размер свободного пространства на накопителях, установленных на вашем компьютере.

Задание 1.41. Посмотрите, какие папки и файлы хранятся на дисках.

Задание 1.42. Откройте папку Мой компьютер и создайте с помощью команд меню несколько вложенных папок.

Задание 1.43. Откройте Проводник и выполните копирование, перемещение и удаление созданных папок при помощи команд Вырезать, Копировать, Вставить и Удалить из меню Правка. Скопируйте в них файлы.

Задание 1.44. Выполните копирование, удаление и перемещение папок и файлов, используя кнопки панели инструментов.

Задание 1.45. Очистите Корзину.

Тема 1.8 **Технология и способы обмена данными**

Выполнив задания этой темы, вы научитесь:

- ⇒ обмениваться данными через буфер обмена;
- ⇒ обмениваться данными по технологии OLE;
- ⇒ создавать составные документы.

Задание 1.46

Откройте текстовый документ Набор.txt, созданный при выполнении задания 1.29. Рассчитайте с помощью программы Калькулятор стоимость товаров, перечисленных в тексте. Результат вычислений вставьте в текст через буфер обмена. Сохраните документ в формате RTF.

Технология работы

1. Запустите программу WordPad, выбрав в меню Пуск команду Все программы ► Стандартные ► WordPad, и разверните его на весь экран.
2. Откройте в программе WordPad файл Набор.txt:
 - выберите в меню команду Файл ► Открыть;
 - в открывшемся окне в поле Папка выберите папку WordPad;
 - в поле, где показано содержимое папки, выделите имя файла Набор.txt (его имя автоматически будет помещено в поле Имя файла);
 - щелкните на кнопке ОК.
3. Запустите программу Калькулятор; расположите окно Калькулятора так, чтобы оно не закрывало данные для расчетов в текстовом документе.
4. Рассчитайте стоимость набора канцелярских товаров для первоклассника по данным из текстового файла Набор.txt с помощью программы Калькулятор:
 - используя описанную в задании 1.28 технологию работы с Калькулятором, вычислите общую стоимость тетрадей, перемножив их цену (4 р.) на количество (5 шт.);

- занесите результат вычислений в память Калькулятора, щелкнув на кнопке M+;
 - наберите цену альбома для рисования (32 р.) и добавьте ее к числу, хранящемуся в памяти Калькулятора, щелкнув на кнопке M+ (операцию умножения на количество (1) можно пропустить);
 - аналогичным образом вычислите стоимость остальных принадлежностей и каждый раз добавляйте результат к сумме, хранящейся в памяти, щелкая на кнопке M+;
 - посмотрите общую стоимость принадлежностей, щелкнув на кнопке MR; результат должен быть равен 114 р.
5. Вставьте результат вычислений в документ Набор.txt через буфер обмена:
- скопируйте результат вычислений в буфер обмена, выбрав в меню программы Калькулятор команду Правка ► Копировать (рис. 1.17);
 - перейдите в окно программы WordPad;
 - установите курсор на место вставки текста, щелкнув мышью перед вопросительными знаками напротив слова *Итого*;;
 - выберите в меню команду Правка ► Вставить — данные из буфера обмена будут вставлены в текст (рис. 1.17);
 - удалите вопросительные знаки.

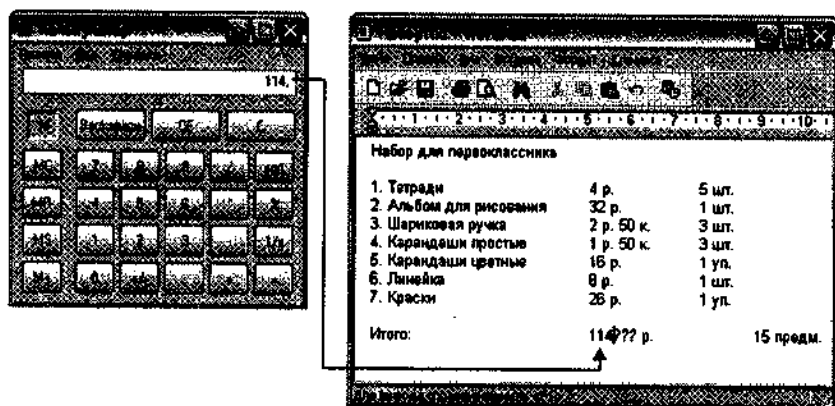


Рис. 1.17. Вставка результата вычислений в текстовый документ через буфер обмена

6. Сохраните сделанные изменения, выбрав команду меню Файл ► Сохранить.

Задание 1.47

Запустите графический редактор Paint, создайте и сохраните любой рисунок, например букву (рис. 1.18).

Графический редактор Paint также является стандартным приложением Windows. Эта программа предназначена для создания и редактирования на экране графических объектов.

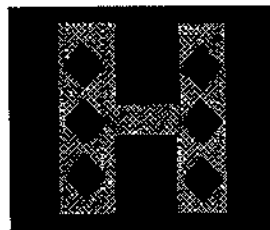


Рис. 1.18.
Графический объект
«Буква Н»



Выполнение этого задания предполагает умение работать в среде графического редактора. Если вы никогда не работали в графическом редакторе, пропустите это задание или сперва освоите эту среду, воспользовавшись разделом 2.

Технология работы

1. Создайте графический объект, например букву Н, используя инструменты Кисть, Распылитель и палитру цветов.
2. Сохраните рисунок:
 - выберите в меню команду Файл ► Сохранить как;
 - выберите в списке папок папку Children;
 - в поле Имя файла введите название *Буква*;
 - в поле Тип файла оставьте предложенное автоматически значение;
 - нажмите кнопку ОК;
 - закройте программу Paint.

Задание 1.48

Создайте составной (интегрированный) документ, используя технологию OLE для внедрения объекта. Для этого выполните внедрение фрагмента рисунка или всего рисунка из файла Буква.bmp в текстовый файл Набор.txt и сохраните его в формате RTF.

Технология работы

1. Запустите приложение-источник — программу Paint, выбрав в меню Пуск команду Все программы ► Стандартные ► Paint.
2. Откройте файл *Буква.bmp*, созданный при выполнении задания 1.47:
 - выберите команду меню Файл ► Открыть;
 - в открывшемся диалоговом окне в поле Папка выберите папку Children;
 - в поле, где показано содержимое папки, выделите файл *Буква.bmp* (его имя автоматически будет помещено в поле Имя файла);
 - щелкните на кнопке ОК.
3. Скопируйте фрагмент рисунка, который надо поместить в текстовый файл:
 - инструментом Выделение выделите фрагмент рисунка, который будет использован как объект для внедрения в текстовый файл;
 - выберите команду меню Правка ► Копировать.
4. Запустите приложение-приемник — программу WordPad.
5. Откройте файл *Набор.txt*, созданный при выполнении задания 1.29 и сохраненный в папке WordPad.
6. Вставьте скопированный фрагмент рисунка в текстовый документ через буфер обмена:
 - установите курсор в текстовый документ в то место, куда должен быть помещен рисунок;
 - выберите команду меню Правка ► Специальная вставка;
 - в открывшемся диалоговом окне (рис. 1.19) выберите значение Вставить — Точечный рисунок;
 - щелкните на кнопке ОК — рисунок будет внедрен в текстовый документ (рис. 1.20).



Если объект внедряется в документ в виде значка, то его можно будет посмотреть, дважды щелкнув на значке. Такой вариант вставки часто применяют при внедрении звуковых объектов.

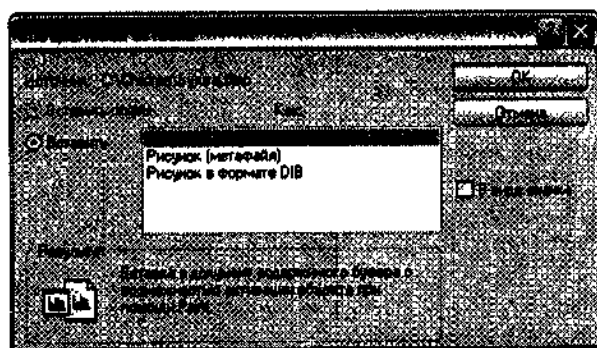


Рис. 1.19. Диалоговое окно Специальная вставка

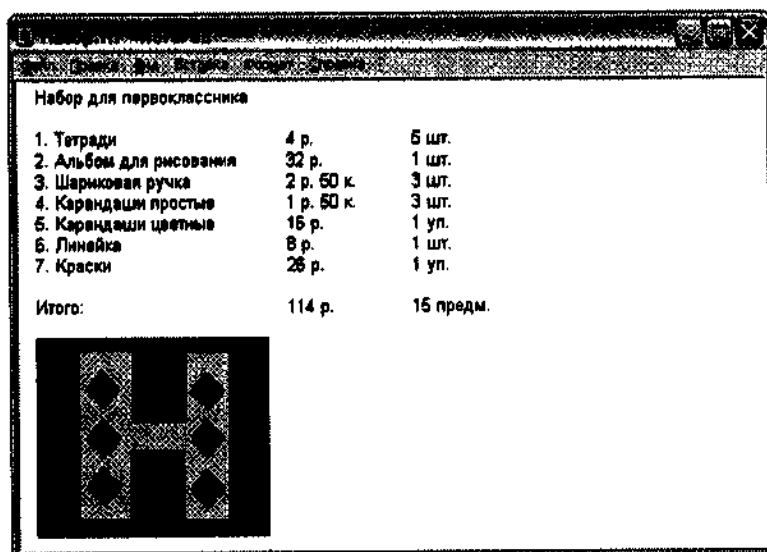


Рис. 1.20. Составной (интегрированный) документ

7. Сохраните документ в формате RTF, выбрав команду меню Файл ► Сохранить как и указав в поле Тип файла значение Файл RTF.

8. Измените внедренный рисунок:

- дважды щелкните на рисунке в тексте: интерфейс программы WordPad изменится на интерфейс приложения-источника — программы Paint (рис. 1.21);

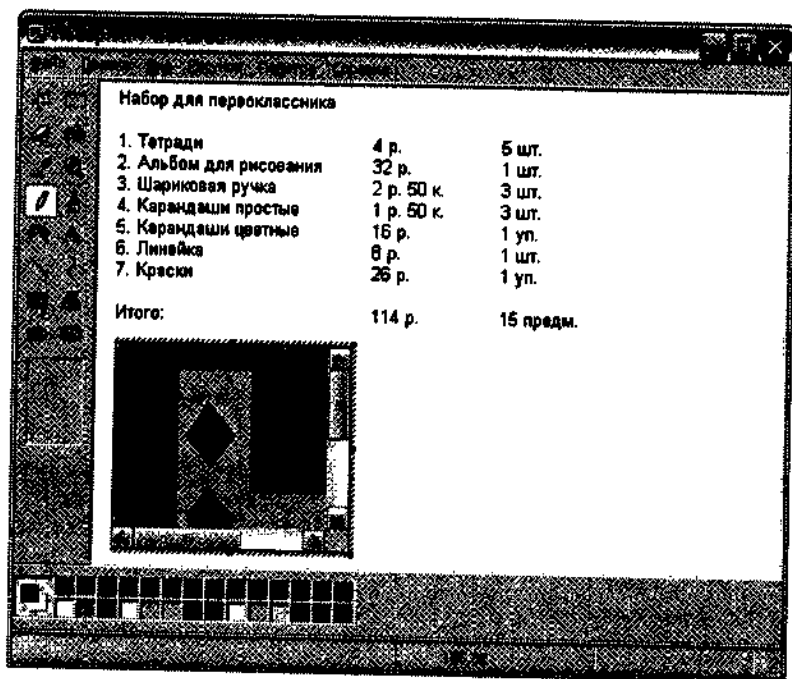


Рис. 1.21. Редактирование точечного рисунка, внедренного в текстовый документ WordPad

- увеличьте окно редактирования, в котором отображается рисунок, потянув за маркеры размера на его границах;
 - используя инструменты программы Paint, внесите изменения в рисунок;
 - по окончании редактирования щелкните левой кнопкой мыши в любом месте за пределами окна редактирования рисунка — интерфейс программы WordPad примет обычный вид, причем все сделанные изменения рисунка будут сохранены в документе.
9. Переключитесь в программу Paint и обратите внимание на то, что изменения, сделанные в текстовом документе, не нашли отражения в исходном файле Буква.bmp.



В некоторых случаях команда Правка > Вставка выполняет внедрение объекта.

Задание 1.49*

Создайте в программе WordPad составной (интегрированный) документ, используя технологию OLE для внедрения и связывания объектов. Для этого выполните внедрение и связывание файла Буква.bmp с файлом, созданным в WordPad.

Технология работы

1. Запустите программу WordPad, выбрав в меню Пуск команду Все программы ► Стандартные ► WordPad.
2. Откройте документ Набор.rtf:
 - выберите команду меню Файл ► Открыть;
 - в списке папок выберите папку WordPad;
 - в поле, где показано содержимое папки, выделите файл Набор.rtf;
 - щелкните на кнопке ОК.
3. Вставьте в документ Набор.rtf связь с файлом Буква.bmp:
 - выберите команду меню Вставка ► Объект — откроется диалоговое окно, позволяющее либо создать новый объект OLE при помощи любого из приложений-источников, установленных на вашем компьютере и перечисленных в поле Тип объекта (рис. 1.22), либо вставить объект OLE, уже сохраненный на компьютере;

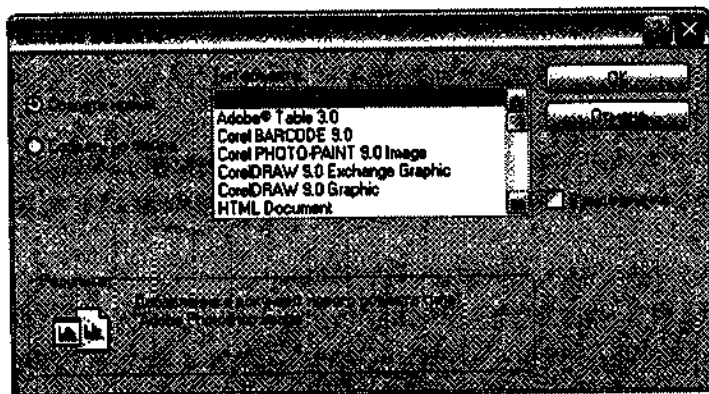


Рис. 1.22. Создание и вставка нового объекта OLE

- в открывшемся окне выберите переключатель Создать из файла и щелкните на кнопке Обзор;
- в диалоговом окне выбора файла откройте папку Children, щелкните на файле Буква.bmp и нажмите кнопку Открыть — полный путь к выбранному файлу C:\Children\Буква.bmp будет отображен в поле Файл (рис. 1.23);
- установите флажок Связь и щелкните на кнопке ОК.

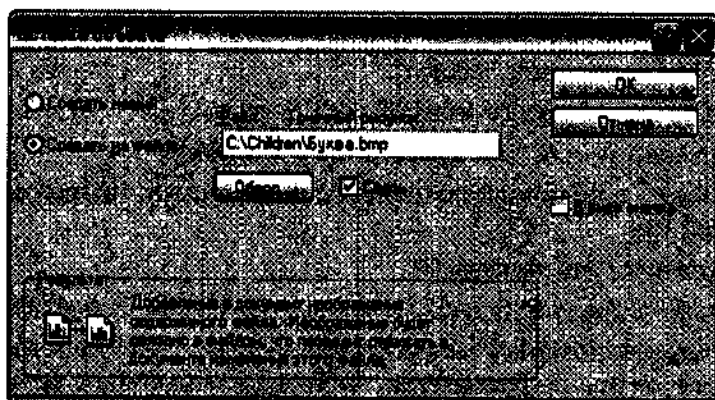


Рис. 1.23. Вставка объекта OLE (файла), уже сохраненного на компьютере

4. Закройте приложение WordPad, сохранив при выходе изменения, сделанные в файле Набор.rtf.
5. Отредактируйте файл Буква.bmp средствами приложения-источника:
 - запустите программу Paint;
 - откройте файл Буква.bmp;
 - внесите изменения, используя инструменты программы Paint.
6. Посмотрите, отразились ли сделанные изменения в документе Набор.rtf:
 - запустите приложение WordPad;
 - откройте документ Набор.rtf;
 - убедитесь, что изменения, сделанные в файле Буква.bmp, отражены в документе Набор.rtf.
7. Закройте приложения WordPad и Paint.

Задания для самостоятельной работы

Задание 1.50. Оформите в текстовом редакторе меню на день с указанием стоимости каждого блюда. Подсчитайте отдельно стоимость завтрака, обеда и ужина в программе Калькулятор. Скопируйте полученные значения в текстовый редактор.

Задание 1.51. Создайте эмблему в графическом редакторе. Наберите в текстовом редакторе WordPad текст пригласительного билета и поместите в него эмблему по технологии OLE:

- вставив одно изображение, без связи с исходным файлом;
- вставив изображение в виде значка;
- вставив изображение и связав его с графическим файлом.

Тема 1.9 **Антивирусная защита информации**

Выполнив задания этой темы, вы научитесь:

- ⇒ обнаруживать вирусы по типовым признакам;
- ⇒ работать с антивирусной программой Dr.Web.

Компьютерные вирусы создаются для того, чтобы нанести ущерб информации, хранящейся на компьютерах. Их появление связано с идеей создания самовоспроизводящихся программ, возникшей еще в 50-е годы прошлого столетия.

Распространение вирусов началось в конце 80-х годов XX века. Свое название эти программы получили в силу того, что очень похожи на биологические вирусы. Компьютерные вирусы, так же как и биологические, представляют опасность для той системы, в которую они внедряются; они способны размножаться, передаваться от одного носителя к другому. Часто вирус обнаруживает себя не сразу, а по прошествии некоторого времени.

Компьютерные вирусы очень разнообразны. Есть вирусы, которые заставляют компьютер выполнять действия, не предусмотренные запущенными на нем программами, например выводить на экран рекламные тексты, визуальные эффекты, видеоролики или делать бессмысленные расчеты, в результате чего компьютер начинает работать в замедленном темпе. Другие программы-паразиты многократно копируют себя в различные участки памяти, что приводит к уменьшению объема памяти и сбоям в работе компьютера. Есть компьютерные вирусы, которые стирают всю записанную в памяти компьютера информацию или блокируют системы программного обеспечения, управляющие работой компьютера, и он полностью выходит из строя.

Как и прочие объекты, компьютерные вирусы можно классифицировать. Например, их можно разделить по степени сложности. Простые вирусы представляют собой отрывок программного кода. Обнаружив его, вирус можно удалить. Более сложные вирусы состоят из нескольких фрагментов кода, которые, на первый взгляд, никак не связаны между собой. При определенных условиях эти фрагменты могут соединиться, и вирус начнет функ-

ционировать. С такими вирусами бороться сложнее. Еще более опасны вирусы, которые, отдав команды изменить или уничтожить информацию на компьютере, полностью стирают себя из памяти, из-за чего впоследствии невозможно установить причину сбоев. Существуют и другие способы классификации, например по среде обитания, по особенностям алгоритма и т. д.

Воздействие вирусов на программное обеспечение происходит в два этапа: заражение и проявление. На этапе заражения вирус включает в другие программы свои копии, которые также способны размножаться. На этапе проявления вирус выполняет те или иные несанкционированные действия.

Важно своевременно обнаруживать вирус. К числу признаков, указывающих на поражение программ вирусом, относятся:

- ◆ неправильная работа программ;
- ◆ замедленная работа компьютера;
- ◆ невозможность загрузки операционной системы;
- ◆ исчезновение файлов;
- ◆ изменение даты, времени создания или размера файла;
- ◆ существенное уменьшение объема свободной памяти;
- ◆ вывод на экран непредусмотренных сообщений или изображений;
- ◆ подача непредусмотренных звуковых сигналов;
- ◆ частые «зависания» компьютера.

Для защиты от компьютерных вирусов, их обнаружения и удаления разработаны *антивирусные программы*. Желательно обнаружить вирус до того, как он активизируется и начнет действовать. Однако так можно выявить только уже известные на данный момент вирусы. Многие популярные антивирусные программы предназначены для выявления известных вирусов, однако поскольку постоянно появляются новые вирусы, такие антивирусные программы быстро устаревают и их необходимо регулярно обновлять.

Наиболее известными на сегодня являются антивирусные пакеты *Symantec Norton Antivirus*, *Антивирус Касперского*, *Dr. Web*, *McAfee VirusScan*, *Panda Titanium Antivirus*, которые находят зараженные вирусами файлы и могут удалить из файла тело программы вируса.

Чтобы уберечь свой компьютер от проникновения вирусов и защитить хранящуюся на нем информацию от их вредоносного воздействия, необходимо соблюдать следующие простые правила:

- ◆ Регулярно проверяйте компьютер на предмет наличия на нем зараженных вирусами файлов с помощью антивирусной программы.
- ◆ Перед копированием файлов с дискет и других съемных носителей всегда проверяйте их на наличие вирусов.
- ◆ Всегда защищайте свои дискеты от записи при работе на других компьютерах.
- ◆ Регулярно делайте архивные копии наиболее важной для вас информации.
- ◆ Не оставляйте без необходимости дискету в дисковом диске.
- ◆ Не используйте программы, поведение которых вам непонятно.
- ◆ Регулярно обновляйте антивирусную программу, которую вы используете.

Задание 1.52

Протестируйте на наличие вирусов папку на диске C: и дискету с помощью антивирусной программы Dr.Web.

Технология работы

1. Запустите программу Dr.Web, выбрав в меню Пуск команду Все программы ► Dr.Web ► Сканер Dr.Web (рис. 1.24).
2. Дождитесь завершения тестирования оперативной памяти компьютера, которая выполняется автоматически после запуска программы Dr.Web.
3. Выберите на диске C: папку, которую надо проверить, например папку Children. Для этого откройте список папок на диске C: и щелчком мыши выделите папку Children (см. рис. 1.24).
4. Настройте режим работы антивирусной программы:
 - выберите команду меню Настройки ► Изменить настройки или щелкните на кнопке Настройки на панели инструментов — появится диалоговое окно Настройки Dr.Web (рис. 1.25);



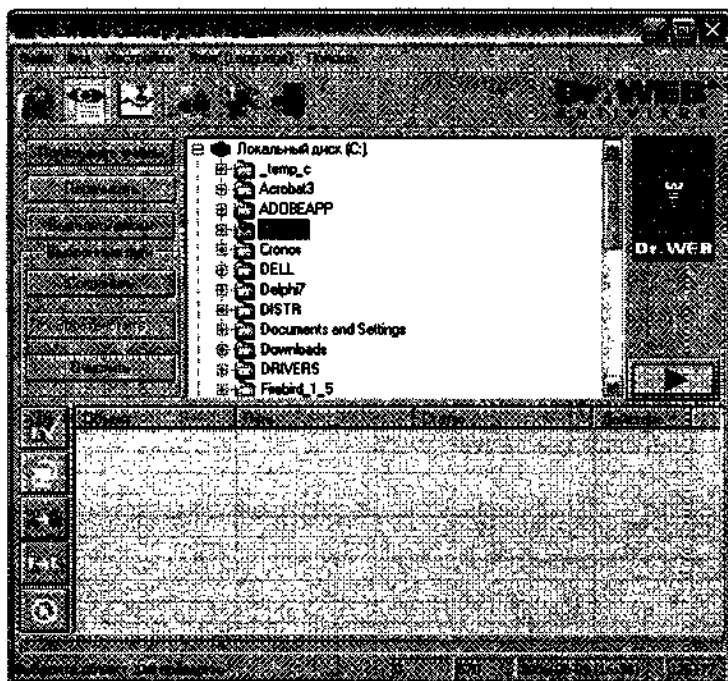


Рис. 1.24. Окно антивирусной программы Dr.Web

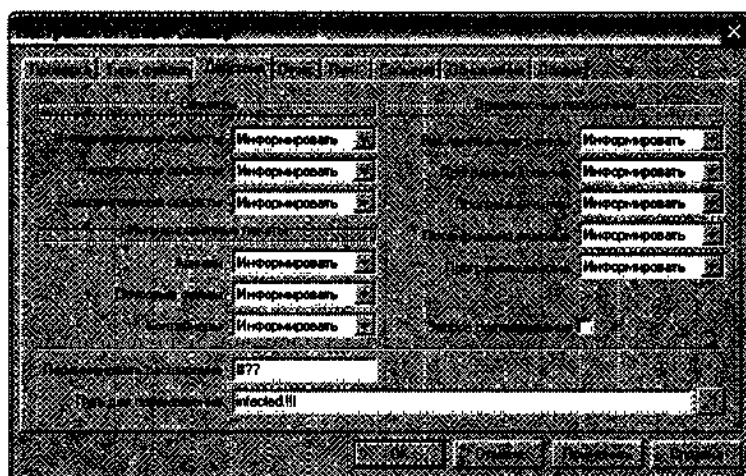


Рис. 1.25. Окно Настройки Dr.Web

- выберите вкладку Действия и установите параметр Вылечить для инфицированных файлов;
- щелкните на кнопке ОК.



Вкладка Действия позволяет выбрать вариант действий программы для инфицированных объектов:

Информировать — сообщить пользователю о зараженных или подозрительных объектах;

Вылечить — удалить вирусы из зараженных объектов;

Удалить — удалить зараженные или подозрительные файлы;

Переименовать — переименовать зараженные или подозрительные файлы;

Переместить — переместить зараженные или подозрительные файлы в указанную папку.

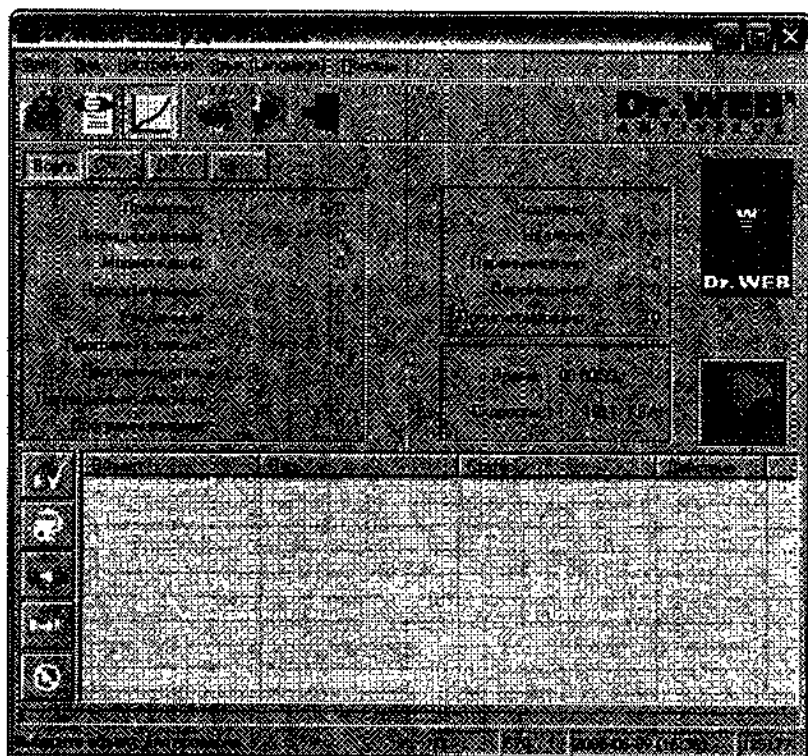


Рис. 1.26. Окно Статистика программы Dr.Web

5. Щелкните на кнопке Пуск в окне программы Dr.Web. 
6. Протестируйте диск A: (дисковод):
 - выберите диск A: щелчком левой кнопки мыши на значке диска A:;
 - щелкните на кнопке Пуск.
7. Проанализируйте результаты тестирования:
 - выберите команду меню Вид ► Статистика или щелкните на кнопке Статистика на панели инструментов (рис. 1.24); 
 - в окне Статистика (рис. 1.26) просмотрите результаты тестирования для всех дисков.
8. Установите вид отображения информации Дерево дисков. Для этого выберите команду меню Вид ► Дерево дисков или щелкните на кнопке Дерево дисков на панели инструментов.

Задания для самостоятельной работы

Задание 1.53. Проверьте на наличие вируса весь жесткий диск.

Задание 1.54. Проверьте на наличие вируса все свои дискеты.

Тема 1.10* **Создание архивных файлов**

Выполнив задания этой темы, вы научитесь:

- ➡ упаковывать файлы в архив с помощью программы WinRAR;
- ➡ распаковывать файлы из архива.

Работая на компьютере, вы можете столкнуться с такой проблемой, как недостаток свободного места на дисках, а при копировании файлов на дискету может оказаться, что размер файла больше ее емкости. В этом случае прибегают к *архивации* файла, при которой происходит уменьшение размера файла за счет сжатия хранящейся в нем информации.

Уменьшить объем файла можно, исключив избыточность представления информации в файле. Например, обычный текст содержит повторяющиеся слова и фразы. Можно исключить повторения, заменив все дублирующиеся фрагменты ссылкой на первый подобный фрагмент. Сжимать можно как один файл, так и несколько файлов, которые в сжатом виде помещаются в *архивный файл*.



Архивный файл — файл, содержащий в сжатом виде один или несколько файлов и служебную информацию.

В результате сжатия информация будет представлена в новом виде. Степень сжатия будет зависеть от метода сжатия и от типа данных в файле. Эффективнее всего сжимаются текстовые и графические файлы. Архивный файл удобен при хранении, копировании и перемещении файлов. Однако в сжатом виде информация представлена так, что без предварительной обработки текст невозможно прочитать, рисунок невозможно посмотреть и т. д., поэтому для работы с информацией требуется ее обратное преобразование — *разархивация*.



Архивация (упаковка в архив) — размещение исходных файлов в архивный файл.



Разархивация (распаковка из архива) — восстановление файлов из архивного файла в том виде, какой они имели до помещения в архив.

Программы, осуществляющие упаковку и распаковку файлов, называются *программами-архиваторами*. Программы-архиваторы отличаются друг от друга методами сжатия, возможными функциями и интерфейсом. Из числа наиболее популярных можно выделить WinRAR, WinZIP, ARJ.

Задание 1.55*

Используя программу-архиватор WinRAR, упакуйте в архив файлы различных типов. Проанализируйте результаты архивации. Удалите исходные файлы. Восстановите файлы из архивных.

При выполнении задания вы должны освоить следующие технологические приемы:

- помещение файлов в архив;
- восстановление файлов из архива;
- помещение нескольких файлов в архив.

Технология работы

1. Создайте на диске C: папку Архивы и скопируйте в нее несколько файлов различных типов, например BMP, DOC, EXE.
2. Запустите программу-архиватор WinRAR, выбрав в меню Пуск команду Все программы ► WinRAR ► WinRAR.
3. Откройте в окне программы WinRAR папку Архивы (рис. 1.27).
4. Запакуйте в архив графический файл формата BMP:
 - щелчком мыши выделите файл формата BMP в окне программы WinRAR;
 - щелкните на кнопке Добавить (Добавить файлы в архив) на панели инструментов — откроется окно, в котором можно уточнить настройки архивации (рис. 1.28);
 - по умолчанию файлу архива назначается имя исходного файла; при необходимости укажите другое имя для архива в поле Имя архива (рис. 1.28);

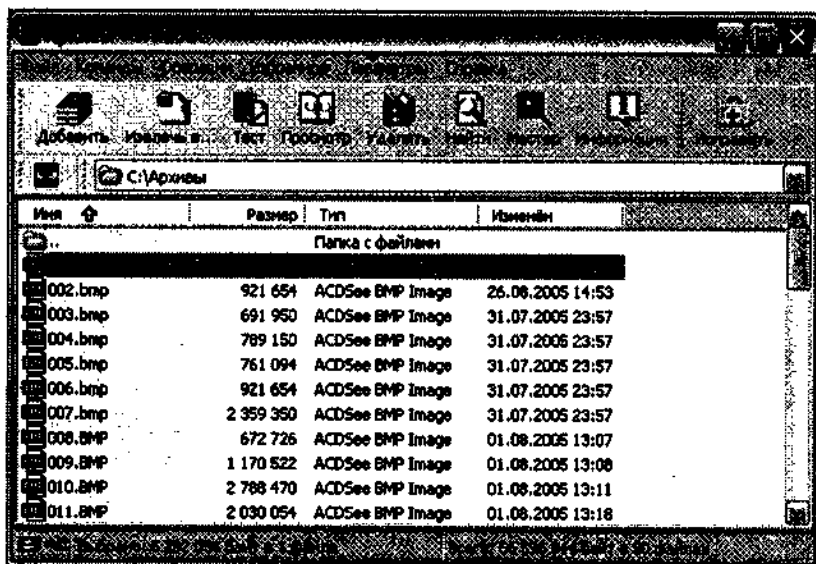


Рис. 1.27. Окно программы WinRAR

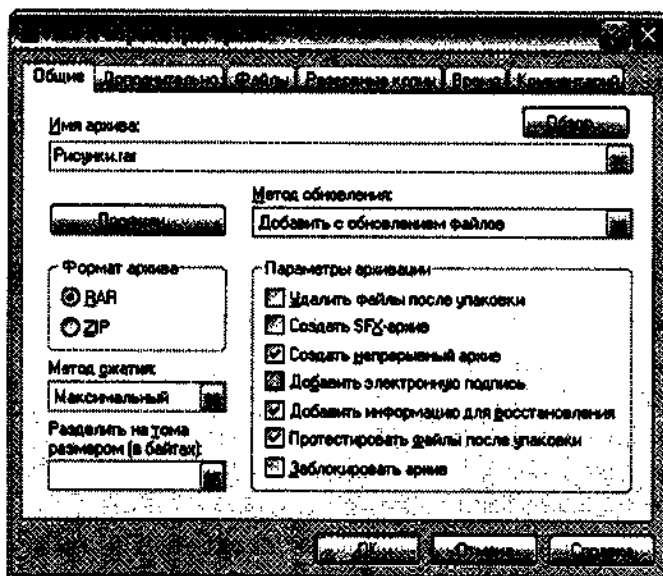


Рис. 1.28. Диалоговое окно архивации

- выберите формат архивного файла — RAR или ZIP; остальные параметры можно оставить без изменений;
 - щелкните на кнопке ОК.
5. Сравните размеры исходного файла и архива; во сколько раз архив меньше исходного файла?
6. Заархивируйте текстовый файл (DOC), повторив действия, указанные в п. 4. Сравните размеры исходного файла и архива.
7. Заархивируйте исполняемый файл (EXE) и сравните размеры исходного файла и архива.
8. Удалите исходные файлы:
- выделите исходные файлы, щелкая на них мышью при нажатой кнопке Ctrl;
 - щелкните на кнопке Удалить на панели инструментов.
9. Распакуйте из архива графический файл:
- выделите архивный файл, содержащий файл формата BMP;
 - щелкните на кнопке Извлечь (Извлечь файлы из архива в указанную папку) на панели инструментов;
 - в открывшемся диалоговом окне в поле Путь для извлечения укажите папку Архивы в качестве папки, в которую следует поместить распакованный файл (рис. 1.29);
 - нажмите кнопку ОК.
10. Распакуйте из архива текстовый файл, повторив действия, указанные в п. 9.
11. Распакуйте из архива исполняемый файл.
12. Поместите в архив несколько файлов:
- выделите несколько файлов, по очереди щелкая на них мышью при нажатой кнопке Ctrl;
 - щелкните на кнопке Добавить на панели инструментов;
 - в открывшемся окне в поле Имя архива укажите имя, например *Группа файлов*;
 - выберите формат архивного файла, например ZIP;
 - щелкните на кнопке ОК.

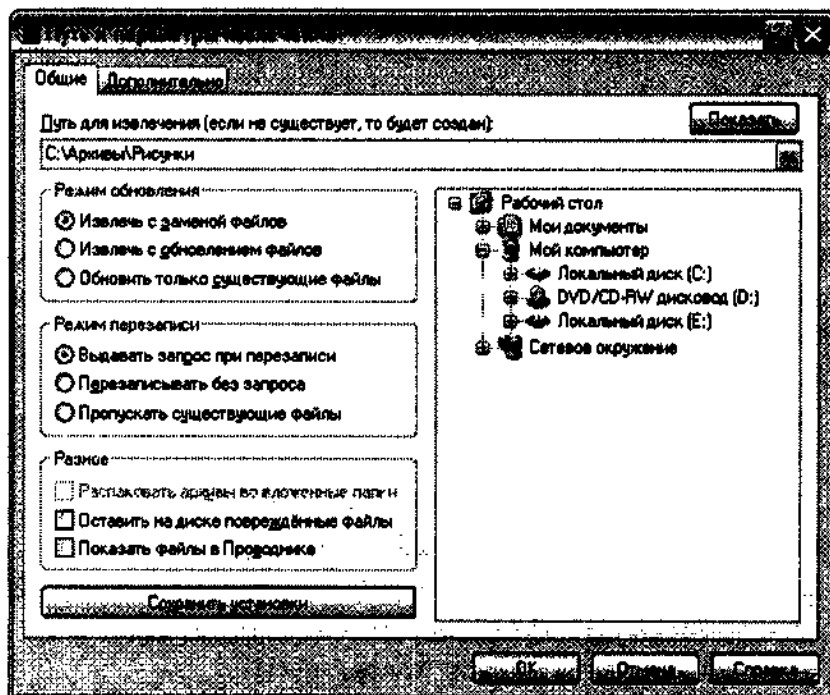


Рис. 1.29. Диалоговое окно параметров извлечения файлов из архива

Задания для самостоятельной работы

Задание 1.56. Посмотрите информацию о размере исходных файлов и архивных файлов. Файлы какого типа сжимаются более эффективно, менее эффективно?

Задание 1.57. Объясните назначение кнопок на панели инструментов программы-архиватора WinRAR.

Задание 1.58. Объясните, для чего нужна архивация файлов.

Задание 1.59. Научитесь самостоятельно упаковывать и распаковывать файлы с помощью программы-архиватора WinZIP.

РАЗДЕЛ 2 Освоение среды графического редактора

Тема 2.1	Общая характеристика графического редактора	66
Тема 2.2	Создание и редактирование рисунка	74
Тема 2.3	Создание и редактирование рисунка с текстом	80

Тема 2.1 **Общая характеристика графического редактора**

Изучив эту тему, вы узнаете:

- ➡ для чего используются графические редакторы;
- ➡ что такое растровая графика;
- ➡ что такое векторная графика;
- ➡ из каких элементарных объектов строится рисунок и какие инструменты используются для их создания.

Назначение графических редакторов

Рисовать можно кистями и красками или использовать для этого карандаши. Можно рисовать на бумаге, холсте, ткани, а можно и на экране компьютера.

На заре своего развития компьютеры не предоставляли пользователю практически никаких средств для создания графических объектов. В то время на экран можно было выводить только символы: буквы, цифры, специальные знаки. Но уже тогда люди пытались использовать компьютер в качестве художественного инструмента. Например, известен случай, когда с помощью компьютерной программы из символов было составлено изображение известной картины Леонардо да Винчи «Мона Лиза».

Шло время, и люди научили компьютер строить простые изображения: точку, прямую линию, окружность. Мониторы в то время были только черно-белые, и поэтому все построенные компьютером изображения напоминали работы художников-графиков.

В отличие от других жанров, например живописи, произведения графиков характеризуются прежде всего четкой прорисовкой линий. Именно поэтому изобразительные возможности компьютеров того времени и стали называть компьютерной графикой. И хотя сегодня с помощью компьютера художники создают произведения в различных жанрах живописи, название «компьютерная графика», прочно закрепившись в сознании людей, употребляется и поныне.

Создание рисунков на компьютере осуществляется с помощью специально предназначенной для этого программы — *графического редактора*.



Графический редактор — это прикладная среда, предназначенная для создания и редактирования графических изображений.

Документом графического редактора является рисунок. По принципу построения графические изображения могут быть *векторными* и *растровыми*.

Векторное графическое изображение

Векторный графический редактор можно сравнить с конструктором, в котором изображение строится из готовых элементов — *графических примитивов* (рис. 2.1).

Процесс редактирования в векторных графических редакторах заключается в добавлении новых объектов и изменении их параметров. Вы создаете один из объектов, изменяете его параметры (размер, положение, угол поворота), а программа сама чертит его в соответствии с заданными характеристиками (рис. 2.2). Такие действия напоминают черчение, когда из простейших геометрических фигур можно создавать разнообразные объекты.

Средства редактирования векторных изображений могут встраиваться в прикладные программы. Например, они есть в текстовом редакторе Word и в табличном редакторе Excel.

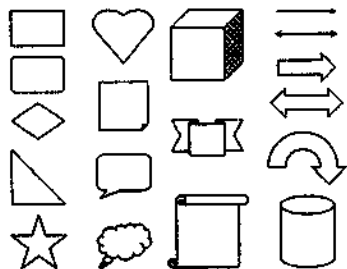


Рис. 2.1. Векторные графические примитивы

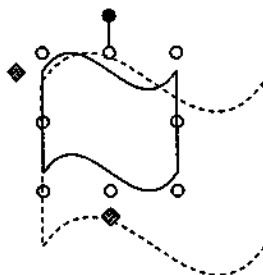


Рис. 2.2. Изменение размера графического примитива в векторном редакторе

Растровое графическое изображение

Растровое изображение состоит из равных по величине очень мелких точек. Каждой из этих точек может быть назначен свой цвет. На экране точки сливаются в целостную картинку, однако



Рис. 2.3. Пиксельная структура растрового изображения

массива точек, упорядоченных в строки и столбцы, называется *растром* (от англ. *raster*).

Вы часто могли слышать слова: «Рисунок имеет разрешение 800 на 600». Это означает, что по горизонтали он состоит из 800 пикселей, а по вертикали — из 600.



Рис. 2.4. Наложение двух объектов

при увеличении растрового изображения они становятся хорошо видны как квадратики и «ступеньки» (рис. 2.3).

Элементы (точки), из которых состоит растровое изображение, называются *пикселями* (от англ. *pixel* — picture element, элемент картинки), а способ представления изображения в виде сплошного

Полученное при помощи растрового редактора изображение состоит как бы из «мазков» кистью по холсту. Представьте себе, что вы нарисовали солнце, а затем изобразили закрывшее его облако (рис. 2.4). При этом часть пикселей, которые составляли часть изображения солнца и были закрашены желтым цветом, теперь «принадлежат» облаку и окрашены в серый цвет.

Изображение в растровом редакторе можно строить либо путем последовательного закрашивания пикселей, либо так же, как и в редакторе векторной графики, из простейших объектов (примитивов). К ним относятся следующие геометрические формы (рис. 2.5):

- ◆ линия;
- ◆ кривая;
- ◆ прямоугольник;
- ◆ эллипс (овал);
- ◆ многоугольник;
- ◆ скругленный прямоугольник.

- ◆ *Линия* — это отрезок прямой произвольной длины. Настраиваемыми параметрами линии являются толщина и длина. При необходимости угол наклона линии можно ограничить строго горизонтальным, строго вертикальным или диагональным (45°) направлением.
- ◆ *Кривая* — это изогнутый в двух местах (без образования углов) отрезок прямой. Наследует все свойства объекта *линия*.
- ◆ *Прямоугольник, эллипс, скругленный прямоугольник*. Эти объекты различаются между собой только формой и характеризуются следующими параметрами: размер, цвет контура (границы), цвет заливки.
- ◆ *Многоугольник* — замкнутая ломаная линия. Наследует все свойства объекта *прямоугольник*. Если это необходимо, можно построить многоугольник, сторонами которого являются только горизонтальные, вертикальные и диагональные отрезки.



Рис. 2.5. Графические примитивы растрового рисунка

Построение любого графического примитива осуществляется перетаскиванием мыши по диагонали воображаемого прямоугольника, в который должна быть вписана фигура.

Кривая строится в три этапа. Первый этап — построение прямой линии. Второй и третий — это формирование двух изгибов. Для создания изгиба нужно установить указатель мыши рядом с точкой предполагаемого изгиба и перетащить мышь при нажатой левой кнопке в сторону от линии до получения желаемого эффекта.

При формировании *многоугольника* сначала создается первый отрезок прямой, затем последовательными щелчками мыши указываются остальные вершины многоугольника. Линии между вершинами будут нарисованы автоматически. Для соедине-

ния последней и первой вершин многоугольника надо дважды щелкнуть левой кнопкой мыши. Обратите внимание, что закрашивание многоугольника происходит после завершения его формирования.

В простейших графических редакторах, таких как Paint, любой объект изображения теряет свою «самостоятельность» сразу после создания. Он становится неотъемлемой частью общего рисунка и в дальнейшем уже воспринимается только как совокупность пикселей. Поэтому говорить о графических примитивах как об объектах с собственными параметрами можно только на стадии их создания.

В более сложных графических редакторах имеется возможность редактировать отдельные объекты и после создания рисунка.

Типовые действия над фрагментом растрового изображения

Основным объектом, над которым пользователь может производить действия, является *фрагмент изображения*. Под фрагментом изображения понимается произвольно выделенная часть рисунка.

Единственным параметром фрагмента является прозрачность фона. Существует режим, в котором все пиксеты, закрашенные цветом фона, при наложении на другие области рисунка воспринимаются как прозрачные.

Для получения возможности работы с фрагментом его необходимо выделить. Так, графический редактор Paint предоставляет две возможности выделения фрагмента: в виде прямоугольной области и в виде области произвольной формы.



Выделение прямоугольной области выполняется движением мыши по диагонали выделяемой области. Для выделения произвольной области ее надо «обвести» инструментом выделения, как карандашом.

К типовым действиям над фрагментом изображения относятся масштабирование, поворот, инверсия цвета, а также выполнение операций редактирования: удаление (стирание), изменение цвета, вставка, замена надписей и т. д.

- ◆ **Масштабирование** в растровом редакторе осуществляется за счет добавления новых или удаления уже имеющихся закрашенных точек (пикселей). При этом не всегда удается сохранить первоначальную прорисовку линий. Некоторые линии при увеличении масштаба теряют точность границ, а при уменьшении исчезают (рис. 2.6 и 2.7).



Рис. 2.6. Исходное изображение

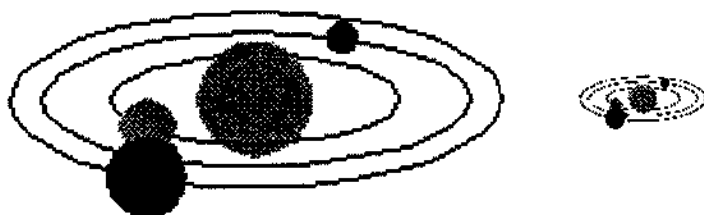


Рис. 2.7. Увеличенное и уменьшенное изображения

- ◆ **Инверсия цвета** — это замена выбранного цвета на противоположный (рис. 2.8).
- ◆ **Поворот.** Выделенный фрагмент можно повернуть по часовой стрелке на заданный угол (90° , 180° , 270°), а также отразить сверху вниз или слева направо, получив его зеркальное отражение относительно вертикальной или горизонтальной оси.

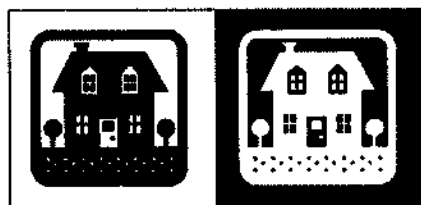


Рис. 2.8. Инверсия изображения

Инструменты графического редактора

Для выполнения типовых действий в графическом редакторе предусмотрена специальная панель инструментов (рис. 2.9). Ниже приводится описание инструментов графического редактора Paint для создания и редактирования фрагментов рисунка. Все действия с инструментами выполняются при нажатой левой кнопке мыши.



Рис. 2.9. Инструменты графического редактора

- ◆ Инструменты **Выделение** и **Выделение произвольной области** позволяют выделить фрагмент изображения для последующего выполнения операции с ним (копирования, удаления и т. д.).
- ◆ Инструмент **Ластик** закрашивает фоновым цветом пиксели, попадающие под курсор мыши при его перемещении.
- ◆ Инструмент **Заливка** позволяет заменить один цвет на рисунке на другой. Закрашивание осуществляется во всех направлениях от пиксела, на котором установлен указатель мыши, до границы с любым другим цветом.
- ◆ Используя инструмент **Выбор цветов**, можно выбрать для рисования любой цвет из уже имеющихся на рисунке. Для этого надо щелкнуть мышью на пикселе соответствующего цвета.
- ◆ Инструмент **Масштаб** позволяет выбрать масштаб отображения рисунка на экране. Изменения размеров самого рисунка при этом не происходит. Рисунок можно рассматривать с увеличением в 2, 6 и 8 раз. Дополнительные значения масштаба можно выбрать с помощью команды меню Вид ► Масштаб.
- ◆ Инструмент **Карандаш** предназначен для рисования объектов произвольной формы. Для рисования надо перемещать мышь при нажатой левой кнопке.
- ◆ Инструмент **Кисть** имитирует движение кисти с краской или пера по бумаге. Можно выбрать несколько различных конфигураций кистей.

- ◆ Инструмент **Распылитель** позволяет наносить краску на поверхность не сплошным слоем, а «рассеивая» отдельные точки. Ширину следа распылителя можно настраивать.
- ◆ Инструмент **Надпись** позволяет вставлять в рисунок текст. При создании текста можно выбирать гарнитуру шрифта, изменять его размер и начертание. После создания надписи, как и другие графические примитивы, перестает быть самостоятельным объектом.
- ◆ Инструменты **Линия**, **Кривая**, **Прямоугольник**, **Многоугольник**, **Эллипс**, **Скругленный прямоугольник** позволяют создавать соответствующие графические примитивы. Их создание было рассмотрено ранее.

Помимо инструментов рисования в графическом редакторе имеется палитра (рис. 2.10). С ее помощью можно выбирать основной и фоновый цвета. Информация о том, какие цвета заданы в данный момент в качестве основного и фоновых, представлена на панели текущих цветов.



Рис. 2.10. Палитра цветов

Контрольные вопросы

1. В чем заключаются особенности графического редактора?
2. Что такое растр? Что такое пиксел?
3. Как вы понимаете слова «изображение выполнено с разрешением 640×480»?
4. Что такое графический примитив?
5. В чем состоят особенности векторной графики?
6. Какие вы знаете объекты растровой графики?
7. Перечислите основные действия над объектами растрового графического редактора.

Тема 2.2 Создание и редактирование рисунка

Выполнив задания этой темы, вы научитесь:

- ⇒ использовать инструменты графического редактора для создания рисунка;
- ⇒ редактировать рисунок.

Задание 2.1

Создайте рисунок по мотивам абстрактных композиций известного русского художника-абстракциониста Василия Кандинского (рис. 2.11).

В процессе выполнения задания вы научитесь работать с основными инструментами графического редактора *Paint*: Линия, Заливка, Эллипс, Прямоугольник, Масштаб, Ластик, Кривая. При создании группы однотипных объектов строго соблюдайте технологию.

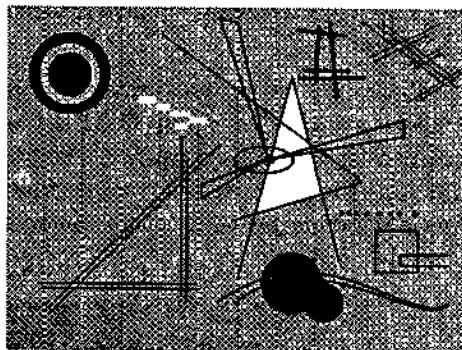


Рис. 2.11. Образец графической композиции (к заданию 2.1)

Технология работы

1. Запустите графический редактор *Paint*.
2. Создайте фон рисунка:
 - выберите цвет фона щелчком *правой* кнопки мыши на нужном цвете в палитре цветов;
 - выберите команду меню Рисунок ▸ Очистить.



Если в палитре цветов нет подходящего цвета, выберите команду меню Палитра ▸ Изменить палитру. В диалоговом окне Изменение палитры щелкните на кнопке Определить цвет; на цветовом круге выберите цвет, уточните оттенок на вертикальной палитре и щелкните на кнопке Добавить в набор. Закройте окно выбора цвета, щелкнув на кнопке ОК.

3. Нарисуйте прямые линии разной толщины, которые на рис. 2.12 объединены в группы 1, 2, 3, 4:

- на панели инструментов выберите инструмент Линия;
- на панели настройки инструмента Линия выберите толщину линии;
- на палитре цветов выберите цвет линии щелчком *левой* кнопки мыши.



Рисую линии группы 1, держите нажатой клавишу Shift для получения строго горизонтальных, строго вертикальных и диагональных линий с углом наклона 45°.

Для линий группы 2 установите самую большую толщину линии.

Для линий группы 3 установите самую маленькую толщину линии.

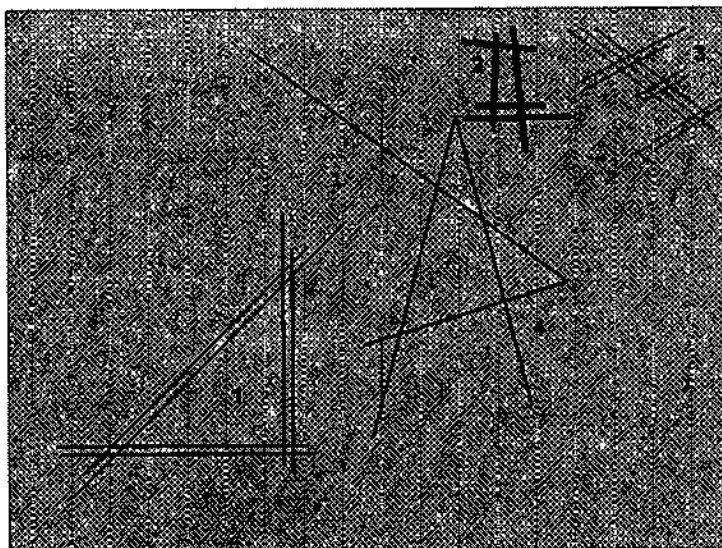


Рис. 2.12. Порядок создания фрагментов композиции

4. Закрасьте некоторые замкнутые области, получившиеся в результате пересечения линий:

- выберите инструмент Заливка на панели инструментов;
- выберите цвет на палитре цветов;

- установите указатель мыши внутри замкнутой области, которую необходимо закрасить;
- щелкните левой кнопкой мыши.



Закрашиваемая область должна быть ограничена непрерывной линией.

5. Нарисуйте группы окружностей 5, 6 и эллипс 7 (рис. 2.13):

- выберите инструмент Эллипс на панели инструментов;
- выберите на палитре цветов два цвета: основной цвет щелчком левой кнопки мыши и цвет фона щелчком правой кнопки;
- выберите режим работы инструмента Эллипс на палитре настройки инструментов.

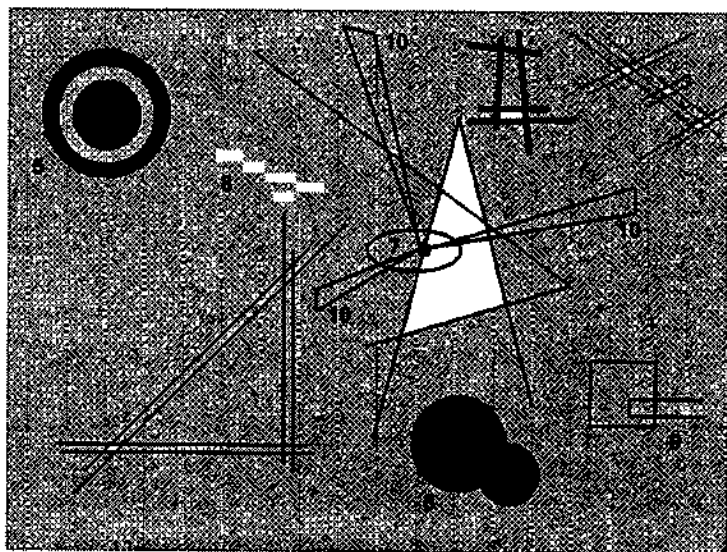


Рис. 2.13. Порядок создания фрагментов композиции



Для окружностей группы 5 выберите настройку инструмента Эллипс так, чтобы была нарисована закрашенная окружность без контура. При рисовании используйте левую кнопку мыши и держите нажатой клавишу Shift для получения кругов.



Для получения группы окружностей, выровненных по центру, начиная рисовать очередную внутреннюю окружность, установите указатель мыши на границу предыдущей окружности.

Для получения окружностей группы 6 выберите настройку инструмента Эллипс так, чтобы была нарисована окружность цвета фона с контуром. Нарисуйте одну окружность при нажатой левой кнопке мыши, вторую — при нажатой правой кнопке мыши. Это позволит изменить соотношение основного цвета и цвета фона.

Для эллипса 7 выберите настройку инструмента Эллипс так, чтобы был нарисован эллипс без заливки с контуром основного цвета.

6. Нарисуйте группы прямоугольников 8, 9 (см. рис. 2.13):

- выберите инструмент Прямоугольник;
- выберите в палитре цветов два цвета: основной и цвет фона;
- выберите режим работы инструмента Прямоугольник на палитре настройки.



Для прямоугольников группы 8 выберите настройку инструмента Прямоугольник так, чтобы был нарисован прямоугольник основного цвета без контура.

Для прямоугольников группы 9 выберите настройку инструмента Прямоугольник так, чтобы был нарисован прямоугольник без заливки с контуром основного цвета. Для того чтобы получить квадрат, держите нажатой клавишу Shift.

7. Удалите одну вертикальную линию в прямоугольнике группы 9:

- для увеличения фрагмента изображения выберите инструмент Масштаб — появится прямоугольник для выделения области увеличения;
- подведите прямоугольник к группе 9;
- щелкните левой кнопкой мыши;
- выберите инструмент Ластик;
- выберите размер ластика на панели настройки инструментов;
- подведите указатель к правой границе прямоугольника и сотрите ее при нажатой левой кнопке мыши;
- для возврата к обычному масштабу изображения выберите команду меню Вид ▸ Масштаб ▸ Обычный или выберите инструмент

Масштаб и на панели настройки инструмента установите масштаб 1х.

8. Нарисуйте треугольники группы 10 (см. рис. 2.13):

- выберите инструмент **Линия** на панели инструментов;
- выберите вторую толщину линии из предложенных вариантов на панели настройки инструмента **Линия**;
- выберите в качестве основного цвета черный;
- установите указатель мыши в центр эллипса;
- нарисуйте отрезок, который будет первой стороной треугольника;
- из конца первого отрезка проведите второй отрезок — вторую сторону треугольника;
- аналогичным образом нарисуйте третий отрезок так, чтобы получился треугольник;
- повторите указанные действия для двух других треугольников с вершиной в центре эллипса;
- инструментом **Заливка** закрасьте несколько замкнутых областей внутри треугольников;
- инструментом **Эллипс** нарисуйте закрашенную окружность в центре эллипса.

9. Нарисуйте кривые линии (см. рис. 2.11):

- выберите инструмент **Кривая** на панели инструментов;
- проведите линию при нажатой левой кнопке мыши, чтобы получился отрезок прямой;
- установите указатель над левой половиной отрезка;
- нажмите левую кнопку мыши и, не отпуская ее, передвиньте мышь вверх, «выгнув» отрезок, отпустите кнопку мыши;
- установите указатель под правой половиной отрезка и передвиньте мышь вниз, «выгнув» вторую половину отрезка вниз;
- повторите указанные действия для второй кривой.

10. Нарисуйте точки (см. рис. 2.11):

- выберите инструмент Кисть;
- на панели настройки инструментов выберите самый большой кружок, который соответствует круглой кисти самого большого размера;
- щелчком левой кнопки мыши нарисуйте несколько точек.

11. Сохраните рисунок в файле с именем Абстракция.bmp:

- выберите команду меню Файл ► Сохранить как;
- в открывшемся диалоговом окне выберите папку, в которой должен быть сохранен файл;
- в поле Имя файла укажите имя файла — Абстракция;
- щелкните на кнопке Сохранить.

12. Закройте программу Paint, выбрав в меню команду Файл ► Выход.**Задания для самостоятельной работы**

Задание 2.2. Нарисуйте собственную абстрактную композицию.

Задание 2.3. Нарисуйте план вашей квартиры.

Задание 2.4. Нарисуйте план расстановки мебели в вашей комнате.

Задание 2.5. Нарисуйте план парка или сквера с использованием выбранных вами условных обозначений для скамеек, деревьев, клумб, кустарников, пруда, детской площадки и т. д.

Тема 2.3 Создание и редактирование рисунка с текстом

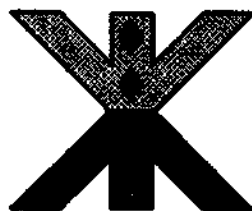
Выполнив задания этой темы, вы научитесь:

- ⇒ добавлять в рисунок текстовые надписи;
- ⇒ создавать буквицу.

Задание 2.6*

Создайте рисунок страницы, которой будет начинаться сказка. Рисунок должен содержать изображение буквицы и текст (рис. 2.14).

В процессе выполнения задания вы познакомитесь с инструментами Многоугольник и Текст.



или-были
старик со старухой...

Рис. 2.14. Текст с буквицей

Технология работы

1. Запустите графический редактор Paint.
2. Создайте фон рисунка:
 - выберите цвет фона щелчком правой кнопки мыши на нужном цвете в палитре цветов;
 - выберите в меню команду Рисунок ► Очистить.
3. Нарисуйте буквицу:
 - выберите инструмент Линия;
 - на панели настройки инструментов выберите среднюю толщину линии, чтобы задать толщину линии для инструмента Многоугольник;
 - выберите инструмент Многоугольник;
 - нажмите клавишу Shift и, не отпуская ее, проведите отрезок горизонтальной линии, который будет основанием буквицы (рис. 2.15);

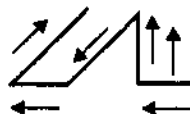


Рис. 2.15. Фрагмент контура буквицы

- установите указатель мыши в следующей вершине контура и щелкните левой кнопкой;
- опишите таким образом весь контур буквы Ж;
- замкните контур буквы двойным щелчком мыши.

4. Раскрасьте буквицу:

- выберите инструмент Линия;
- выберите красный в качестве основного цвета и проведите горизонтальную линию, делящую буквицу пополам;
- выберите инструмент Заливка;
- закрасьте верхнюю часть буквицы зеленым цветом, а нижнюю — синим.

5. Украсьте буквицу узором:

- выберите красный в качестве основного цвета;
- выберите инструмент Многоугольник;
- на панели настройки инструментов выберите рисование основным цветом без контура;
- нажмите клавишу Shift для рисования линий под углом 45°;
- нарисуйте ромб рядом с буквицей;
- выберите инструмент Выделение для выделения прямоугольной области;
- выделите фрагмент рисунка, содержащий ромб;
- выберите в меню команду Правка ► Копировать для помещения выделенного фрагмента в буфер обмена;
- в меню Рисунок снимите флажок рядом с командой Непрозрачный фон, чтобы вставляемые из буфера фрагменты рисунка имели прозрачный фон;
- выберите в меню команду Правка ► Вставить;
- установите указатель мыши внутри выделенного фрагмента в левом верхнем углу и, удерживая нажатой левую кнопку мыши, переместите фрагмент рисунка на изображение буквицы;
- повторите действия для вставки остальных ромбов;
- удалите ромб, нарисованный рядом с буквицей, выделив его инструментом Выделение и нажав на клавишу Delete;

- аналогичным образом можно нарисовать на буквице другие узоры, например треугольники или круги.
6. Введите текст:
- выберите инструмент Текст;
 - нарисуйте мышью текстовую рамку;



Если после создания текстовой рамки не появилась панель атрибутов текста, выберите команду меню Вид ▸ Панель атрибутов текста.

- установите следующие атрибуты текста: шрифт Arial (кириллица), размер 36;
 - щелкните левой кнопкой мыши внутри текстовой рамки;
 - введите текст: или-были (буква Ж пропущена, так как она сделана буквицей);
 - щелкните вне текстовой рамки;
 - выделите текст инструментом Выделение;
 - поместите указатель мыши внутрь выделенного фрагмента, нажмите левую кнопку мыши и, не отпуская ее, переместите текст к буквице;
 - выполните аналогичные действия для текста старик со старухой.
7. Сохраните файл с именем Буквица.bmp:
- выберите в меню команду Файл ▸ Сохранить как;
 - выберите папку, в которую надо сохранить файл;
 - введите имя файла — Буквица;
 - щелкните на кнопке Сохранить.
8. Закройте программу Paint, выбрав команду меню Файл ▸ Выход.

Задания для самостоятельной работы

Задание 2.7. Нарисуйте буквицу для текста: «В некотором царстве, в некотором государстве...».

Задание 2.8. На плане, где указана расстановка мебели в комнате, сделайте подписи.

Задание 2.9. Нарисуйте цветок и укажите его основные части, сделав к ним подписи.

РАЗДЕЛ 3 Освоение среды текстового процессора

Тема 3.1	Общая характеристика текстового процессора	84
Тема 3.2	Создание и редактирование текстового документа	97
Тема 3.3	Форматирование текста	105
Тема 3.4	Оформление текста в виде таблицы и печать документа	113
Тема 3.5*	Включение в текстовый документ графических объектов	117

Тема 3.1 **Общая характеристика текстового процессора**

Изучив эту тему, вы узнаете:

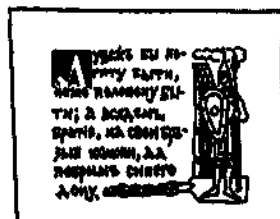
- ➡ назначение и возможности текстового процессора;
- ➡ что такое макет текстового документа;
- ➡ из каких элементарных объектов состоит текстовый документ;
- ➡ способы выделения объектов текстового процессора.

История обработки текстовых документов

Очень давно люди изобрели специальную систему знаков для кодирования звуков человеческой речи. Так зародилась письменность. В древности для письма использовались разнообразные инструменты. Сначала это были стило и вошенная дощечка, глиняные таблички, выделанная кожа (пергамент), берестяные свитки. Им на смену пришли бумага и перо. Работая этими инструментами, человек должен быть очень внимательным, чтобы не допустить ошибки. Одно неосторожное движение пером — и весь текст придется переписывать заново.



Люди стремились не только точно записать смысл сказанного, но и сделать текст красочным, художественно выразительным. Например, заглавные буквы в начале повествования рисовывались киноварью — красной краской. Поля страницы и текст украшались завитушками и орнаментами. Появился даже особый вид искусства — каллиграфия.



Однако создание текстов и книг выполнялось человеком вручную от начала до конца. Чтобы получить новый экземпляр рукописи, требовалось полностью переписать ее заново. Поскольку создать точную копию вручную просто невозможно, каждая книга была произведением искусства,

существующим в единственном экземпляре, а трудоемкость переписывания текстов делала книги очень дорогими и недоступными простым людям. Тиражирование рукописей стало возможным лишь с изобретением книгопечатания.

В России первые печатные книги появились в Москве в эпоху Ивана Грозного. Русский первопечатник Иван Федоров со своим помощником Петром Мстиславцем напечатал первую книгу в 1564 году.

Любая рукопись теперь могла быть издана в большом количестве экземпляров. Подготавливая книгу к изданию, создавали прототип каждой страницы в виде формы из дерева или металла. Наборщики буква за буквой составляли текст будущей печатной страницы. Художники рисовали иллюстрации, которые затем также переводили на твердую основу. Когда все было готово, делали нужное количество оттисков.

Весь процесс подготовки рукописи к печати занимал много времени, хотя и позволял получить в результате большое количество копий книги. Для книгопечатания требовалось множество квалифицированных специалистов и сложное громоздкое оборудование. Осуществить издание рукописи одному человеку было практически не под силу. Этим занимались специальные предприятия — типографии.

Изобретение печатной машинки упростило процесс тиражирования рукописей для обычного человека, но тоже не решило всех проблем. Стало возможно выпустить сразу несколько копий одного текста. Правда, печатная машинка помогала только в обработке текста. Иллюстрации, как раньше, приходилось рисовать от руки и переснимать через копировальную бумагу.

Только после изобретения компьютера практически все типографские возможности стали доступны рядовому пользователю.



Макет текстового документа

Представьте себе, что весь учебный материал, помещенный в этой книге, состоял бы только из текста. Читать такую книгу было

бы очень утомительно. Оживить однообразный текст можно, дополнив его иллюстрациями, рисунками, схемами, таблицами и т. д. Современная компьютерная технология позволяет подготовить к изданию любую книгу.

Пока книга хранится в электронном виде, она называется *текстовым документом*, как это принято в среде прикладных программ Windows.



Текстовый документ — это документ, созданный в прикладной среде и состоящий из текстов, рисунков и таблиц.



Прежде чем издать любой текстовый документ, будь то книга или реферат, письмо или справка, необходимо изготовить *макет* этого документа, чтобы каждый объект документа (иллюстрация, схема, таблица) занимал свое место в соответствии со следующими правилами:

- ◆ Каждый объект документа должен находиться в том месте, где это необходимо по смыслу.
- ◆ Каждый объект должен иметь размер, соответствующий своей смысловой нагрузке. Например, если текст о рыцарской

битве сопровождается картой сражения и изображением рыцаря, то главной по смысловому значению в данном случае является карта. Поэтому она должна выделяться размером по сравнению с второстепенной иллюстрацией, изображающей рыцаря.

- ◆ На всем протяжении документа должен соблюдаться определенный ритм чередования различных объектов (рис. 3.1).

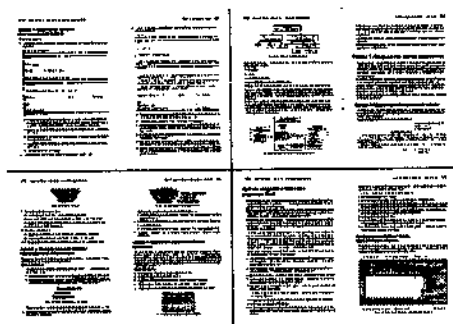


Рис. 3.1. Чередование объектов в тексте

Тексту также необходимо придать выразительность. Для этого текст *структурируют*. Под структурированием текста понимают разделение его на более мелкие смысловые фрагменты и оформление этих фрагментов в соответствии с их значимостью. Так, художественный текст делится на главы. Текст учебника разбивают на разделы, темы, параграфы. Заголовки и наиболее важные места в тексте выделяют с помощью отличающихся от остального текста шрифтов, а также линий или рамок.

Соблюдение вышеперечисленных правил позволяет создать из начального набора разнородных объектов макет документа.

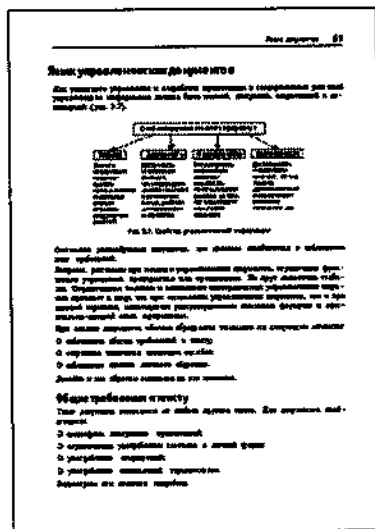


Рис. 3.2. Пример макета страницы книги



Макет текстового документа — это совокупность размещенных на странице и упорядоченных по определенным правилам объектов текстового документа.

Макет (рис. 3.2) является электронным оригиналом печатного издания, на основе которого создается нужное количество копий.

Характеристики текстового процессора

При подготовке текстового документа каждый объект необходимо сначала создать, затем отредактировать и выполнить его форматирование. После этого необходимо составить макет документа. Все эти операции можно выполнить в прикладной среде, которая называется *текстовым процессором*.



Текстовый процессор — прикладная среда для создания, редактирования, форматирования и макетирования текстового документа.

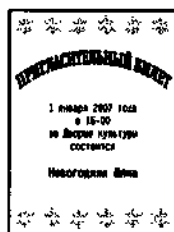
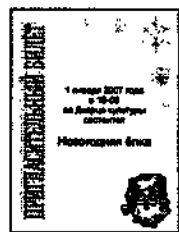
В чем заключаются преимущества создания текстового документа с помощью текстового процессора?



Во-первых, в электронный текст можно быстро и просто внести изменения. Представьте, что вы допустили ошибку в тексте поздравительной открытки. Вам придется взять новую открытку и переписать текст заново. Не зря в пословице говорится: «Что написано пером, не вырубишь топором». А в электронном документе можно делать сколько угодно исправлений, и они будут абсолютно незаметны, так

как на экране всегда отображается последняя редакция документа.

Во-вторых, нет необходимости продумывать заранее, как должен выглядеть готовый макет документа. Макет можно



создавать уже после набора текста, причем вариантов макета можно сделать столько, сколько потребуется. Например, набрав текст пригласительного билета один раз, можно оформить его несколькими способами (рис. 3.3) и распечатать приглашения в нужном количестве.

Рис. 3.3. Варианты макета пригласительных билетов

В-третьих, значительно увеличивается количество выразительных средств оформления текста и облегчается их использование.

В-четвертых, автоматизировано большинство рутинных работ, таких как поиск орфографических и синтаксических ошибок, поиск и замена слов и т. п.

В-пятых, специальные программы позволяют включать в текст фрагменты на другом языке, формулы, рисунки, таблицы и т. д. (рис. 3.4).

Наконец, создав электронный документ один раз, можно получить неограниченное число его копий.

Какие же документы можно создавать при помощи текстового процессора? Это могут быть простые документы, не требующие особого форматирования и макетирования: письма, заметки, сообщения. Для данного класса документов технологически важным является этап редактирования.

К более сложным следует отнести конверты, наклейки, наградные листы и т. п. Эти документы имеют простую структуру и чаще всего являются одностраничными. Здесь на первый план выходит форматирование.

Наиболее сложными являются документы, где текст чередуется с рисунками, таблицами, формулами. К ним можно отнести брошюры, газеты, книги, справочники и т. п. При создании таких документов каждый из технологических этапов является значимым. Особое значение приобретает умение составлять макет издания. Посмотрите на книгу, лежащую перед вами. Могла бы типография ее напечатать, не имея макета? Нет, данная книга так и осталась бы рукописью, несмотря на то что ввод и редактирование текста осуществлялись на компьютере.

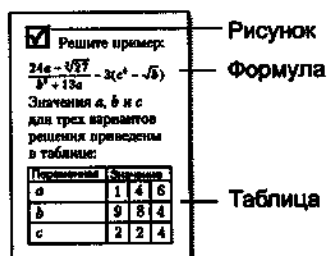


Рис. 3.4. Объекты, внедренные в текстовый документ

Объекты текстового документа и их параметры

Прежде чем продолжить работу в среде текстового процессора, надо составить представление об объектах, которыми он оперирует. Для этого познакомимся с классификацией объектов текстовых документов (рис. 3.5).

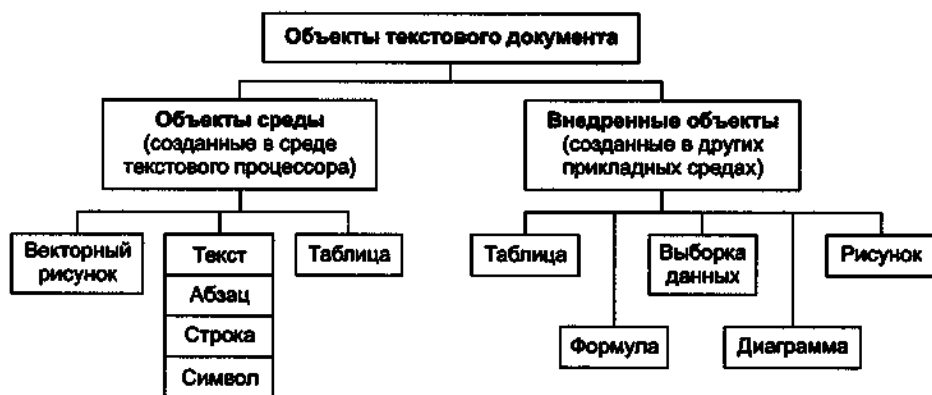


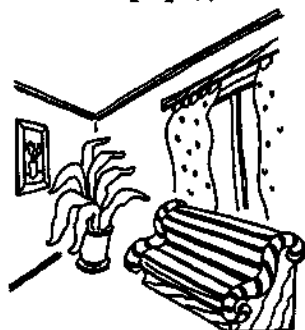
Рис. 3.5. Классификация объектов текстового документа

Классификация объектов, представленная на рис. 3.5, имеет два уровня. На первом уровне основанием классификации служит прикладная среда. Объекты делятся на две большие группы: объекты прикладной среды и внедренные объекты.

Объектами среды называются объекты документа, для создания, редактирования и форматирования которых не требуется вызывать отдельную программу.

Внедренными объектами называются объекты, которые первоначально создаются в другой прикладной среде.

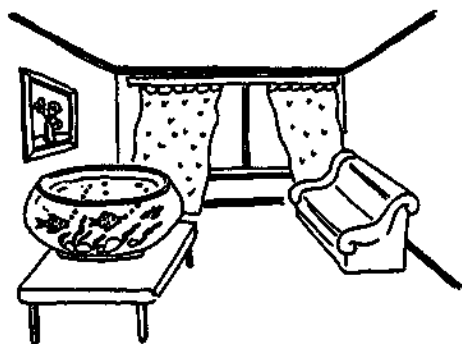
Проведем аналогию. Квартира является частью среды обитания городского жителя. Все здесь предназначено для него: мебель и посуда, одежда и обувь, осветительные и бытовые приборы. Но до появления городов человека большей частью окружала живая природа. Многие горожане скучают по утраченному общению с животным и растительным миром. Им хочется поселить в своей городской квартире какое-нибудь растение или, например, завести рыбок. Но цветок не может расти прямо на полу или на столе. Для нормальной жизни ему необходима его привычная среда — земля. Чтобы цветок рос в квартире, надо приобрести цветочный горшок, заполнить его землей и посадить в нее цветок. Таким образом,



Таким образом, для того чтобы объект мог существовать в новой среде, необходимо создать для него соответствующую среду. В текстовом процессоре это означает, что для внедрения объектов из других прикладных сред необходимо использовать соответствующие команды и инструменты.

внутри одной среды (городская квартира) появляется фрагмент другой среды (земля), обеспечивающий существование объекта (цветок).

Другим примером внедрения объекта одной среды в другую является аквариум. В этом случае водная среда с рыбками гармонично интегрируется в среду обитания человека.



- Основанием классификации на втором уровне является вид объекта (см. рис. 3.5). Рассмотрим подробнее подкласс объектов, создаваемых текстовым процессором. Начнем с объектов подкласса «текст». В тексте могут быть выделены следующие объекты: страница, абзац, строка, слово, символ.

Символ как объект характеризуется следующими основными параметрами: вид начертания, кегль, цвет (см. тему 15 в учебнике). Помимо основных параметров для символа можно также определить эффект, смещение и кернинг.

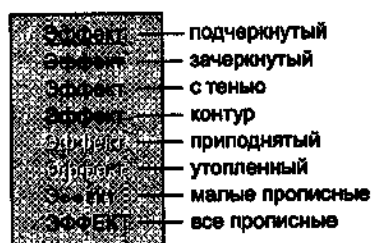


Рис. 3.6. Эффекты

Эффект — определяет, как символ будет представлен на экране и в документе. Наиболее распространенные эффекты приведены на рис. 3.6.

Смещение — определяет положение символа относительно базовой линии строки. Различают смещение вниз и вверх (рис. 3.7).



Рис. 3.7. Вертикальное смещение

Кернинг — межсимвольный интервал. Различают три основных значения данного параметра: нормальный, разреженный, уплотненный (рис. 3.8). Обычно нормальное значение кернинга определяется автоматически для заданного

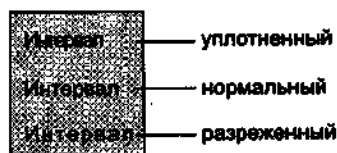


Рис. 3.8. Изменение межсимвольного интервала

начертания шрифта. При необходимости значение кернинга можно задать вручную.

Из объектов-символов образуются объекты-слова.

Слово представляет собой последовательность символов (букв, цифр, специальных знаков), кроме символа «пробел». Объект «слово» наследует все параметры объекта «символ». Единственным самостоятельным параметром слова является *количество символов*.

Строка. Из объектов-слов образуются объекты-строки. Слова в строках отделены друг от друга пробелами. Без пробела за словом могут следовать только знаки препинания, такие как запятая, точка, точка с запятой, двоеточие, скобка. Объект «строка» наследует все параметры объекта «слово». Самостоятельным параметром строки является *количество слов*. При форматировании строка подчиняется тем же правилам, что и символы и абзацы.

Абзац. Из объектов-строк образуются объекты-абзацы. Ввод абзаца всегда должен заканчиваться нажатием на клавишу Enter.

Признаком окончания абзаца является символ перевода строки (§). В обычном режиме этот символ на экране не отображается. Если включить режим отображения непечатаемых символов, то в конце каждого абзаца появится символ §.

Абзац наследует все параметры объекта «строка» и дополнительно характеризуется следующими параметрами: *втяжкой*, *абзацным отступом*, *выравниванием*, *интерлиньяжем*.

Втяжка. Абзацу можно задать отступ от левого и правого полей страницы. Форматирование абзаца выполняется с отступом в том случае, если необходимо выделить его содержание из общего контекста или когда вводимый текст является стихотворением (рис. 3.9).

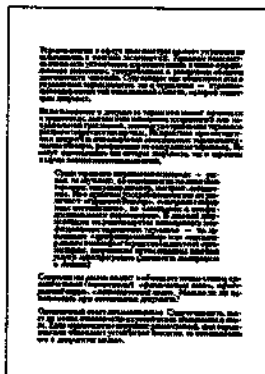


Рис. 3.9. Втяжка

Абзацный отступ. Первая из этих строк абзаца может начинаться с отступа или выступа, считая от левой границы. Различают три типа абзацного отступа: *положительный* («красная строка»), *отрицательный* («висячая строка», когда начало первой строки выдвигается влево по сравнению с остальными строками) и *нулевой*, когда все строки выровнены по левому краю одинаково. Примеры оформления первой строки абзаца приведены в табл. 3.1.

Таблица 3.1. Оформление первой строки абзаца

Тип отступа	Пример
Положительный абзацный отступ («красная строка»)	Текстовый процессор обеспечивает набор и редактирование текста, его хранение, иллюстрирование рисунками, подготовку к печати и распечатку на принтере.
Отрицательный абзацный отступ («висячая строка»)	Текстовый процессор обеспечивает набор и редактирование текста, его хранение, иллюстрирование рисунками, подготовку к печати и распечатку на принтере.
Нулевой абзацный отступ	Текстовый процессор обеспечивает набор и редактирование текста, его хранение, иллюстрирование рисунками, подготовку к печати и распечатку на принтере.

Выравнивание. Абзац можно выравнивать различными способами относительно полей страницы: по одной из границ (левой или правой), оставляя другую невыровненной, по обеим границам (за счет увеличения ширины пробелов между словами) либо по центру, чтобы каждая строка отстояла на равные расстояния от обеих границ. Примеры выравнивания абзаца приведены в табл. 3.2.

Таблица 3.2. Примеры выравнивания абзацев

Выравнивание	Пример
По левому краю	Текстовый процессор обеспечивает набор и редактирование текста, его хранение, иллюстрирование рисунками, подготовку к печати и распечатку на принтере.
По правому краю	Текстовый процессор обеспечивает набор и редактирование текста, его хранение, иллюстрирование рисунками, подготовку к печати и распечатку на принтере.
По центру	Текстовый процессор обеспечивает набор и редактирование текста, его хранение, иллюстрирование рисунками, подготовку к печати и распечатку на принтере.
По ширине	Текстовый процессор обеспечивает набор и редактирование текста, его хранение, иллюстрирование рисунками, подготовку к печати и распечатку на принтере.

Интерлиньяж — расстояние между строками в абзаце. Этот параметр может задаваться в пунктах и в строках. При измерении в строках величина интерлиньяжа зависит от выбранного начертания шрифта. Наиболее часто используются одинарный, полуторный и двойной интерлиньяж (рис. 3.10).

Пользователь, работающий с текстовым процессором, зачастую сталкивается с необходимостью задавать расстояния между абзацами.

Расстояние между строками внутри абзаца и расстояние между абзацами могут изменяться независимо друг от друга.

Общие характеристики объектов текстового документа приведена в табл. 3.3.

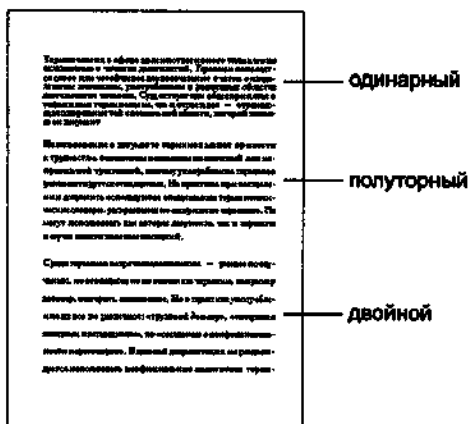


Рис. 3.10. Интерлиньяж

Таблица 3.3. Характеристики объектов текстового документа

Объект	Параметры
Символ	Начертание, кегль, цвет, эффект, смещение, кернинг
Слово	Количество символов. Наследуются свойства объекта «символ»
Строка	Количество слов. Наследуются свойства объекта «строка»
Абзац	Втяжка, абзацный отступ, выравнивание, интерлиньяж, расстояние между абзацами. Наследуются свойства объекта «строка»
Таблица	Количество строк и столбцов
Рисунок	Тип, размер, цвета и линии, положение, обтекание текстом

Способы выделения объектов текстового документа

Для дальнейшего изучения действий на объектами введем понятие текущего объекта.

Текущим объектом называется тот объект текстового документа, на который в данный момент оказывается воздействие в текстовом процессоре. В качестве текущего объекта текстового документа может выступать любой из объектов, показанных на рис. 3.5. Чтобы объект стал текущим, его необходимо *выделить*. При выделении изображение объекта становится инверсным (при выделении текста или таблицы) или помечается маркерами выделения (например, при выделении рисунка или формулы).

Для выполнения любых действий над объектами текстового документа необходимо запомнить следующее правило:



Выдели объект, а затем производи над ним действие.

Рассмотрим способы выделения различных текстовых объектов.

Выделение слова. Для выделения слова дважды щелкните левой кнопкой мыши на любом символе этого слова (третий щелчок выделит весь абзац).

Выделение строки. Для выделения строки установите указатель мыши слева от этой строки, на поле документа (при этом указатель примет вид стрелки, указывающей вправо вверх) и один раз щелкните левой кнопкой мыши.

Выделение абзаца. Для выделения абзаца выделите его первую строку (как указано выше) и, не отпуская левую кнопку мыши, проведите мышью вниз до последней строки выделяемого абзаца. Можно также трижды щелкнуть на любом символе внутри абзаца. Если вы хотите изменить параметры абзаца, то достаточно щелкнуть один раз в любом месте абзаца.

Выделение таблицы. Можно выделить как отдельные части таблицы (ячейку, строку, столбец), так и всю таблицу целиком. Для выделения ячейки таблицы подведите указатель мыши к ее левой границе (указатель примет вид черной стрелки, указывающей вправо вверх) и щелкните левой кнопкой мыши. Другой способ — щелкнуть мышью в первой ячейке таблицы и перейти в нужную ячейку, нажимая клавишу Tab.

Для выделения строки, столбца или всей таблицы нужно:

- ♦ установить курсор в любом месте объекта, который необходимо выделить;

- ◆ выбрать в меню команду Таблица ▶ Выделить ▶ Таблица, Таблица ▶ Выделить ▶ Столбец или Таблица ▶ Выделить ▶ Строка.

Выделение внедренного объекта. Для выделения любого внедренного объекта, в том числе и графического объекта, созданного непосредственно в среде текстового процессора, щелкните на нем мышью. На границах объекта (по углам и по серединам сторон) появятся маркеры выделения, говорящие о том, что объект выделен.

Контрольные вопросы

1. В чем состоит назначение текстового процессора?
2. Определите понятие «текстовый процессор».
3. Что понимают под текстовым документом?
4. Что такое макет документа и для чего он нужен?
5. Что понимается под созданием макета текстового документа?
6. Из каких объектов состоит текстовый документ?
7. Что такое внедренный объект?
8. Какие вы знаете параметры символа?
9. Что такое кернинг?
10. Что такое интерлиньяж?
11. Перечислите параметры объекта «строка».
12. Перечислите параметры объекта «абзац».

Тема 3.2 Создание и редактирование текстового документа

Выполнив задания этой темы, вы научитесь:

- вводить и редактировать текст;
- выполнять действия с различными объектами текстового документа;
- осуществлять поиск и замену текста;
- выполнять проверку орфографии.

Задание 3.1

Познакомьтесь со средой текстового процессора Word, технологией создания и сохранения текста, набрав текст, приведенный на рис 3.11, и сохранив его в файле.

При вводе текста не обращайте внимания на отсутствие некоторых запятых, так как это образец, на котором вы будете осваивать различные приемы редактирования.

РЕДАКТИРОВАНИЕ ТЕКСТОВОГО ДОКУМЕНТА

В текст, введенный с клавиатуры, может вкрасться ошибка. Все возникающие ошибки можно разделить на три типа: лишняя буква, неверная буква, пропущенная буква.

Для устранения ошибок используются соответствующие методы: удаление, замена, вставка.

Применение перечисленных методов поможет вам отредактировать текст, введенный с клавиатуры. После текст введенный с клавиатуры можно использовать для создания текстового документа.

Рис. 3.11. Образец текста (к заданию 3.1)

Технология работы

1. Запустите текстовый процессор Word.
2. В рабочем поле текстового процессора введите текст, приведенный на рис. 3.11. При вводе текста после второго и третьего

предложений нажмите клавишу Enter, чтобы получить текст, состоящий из трех абзацев.

3. Сохраните созданный текст в файле:

- выберите в меню команду **Файл** ▶ **Сохранить как**;
- в открывшемся диалоговом окне выберите папку, в которой надо сохранить файл, например **Мои документы**;
- в поле **Имя файла** укажите имя, например **Упражнение**; тип файла — **Документ Word (*.doc)** — задан по умолчанию;
- нажмите кнопку **Сохранить**.

4. Завершите работу текстового процессора Word, выбрав в меню команду **Файл ▶ **Выход**.**

Задание 3.2

Выполните основные операции по редактированию текстового документа:

- ◆ *редактирование слов в режимах **Вставка** и **Замена**;*
- ◆ *копирование фрагментов текста;*
- ◆ *удаление фрагментов текста;*
- ◆ *перемещение фрагментов текста;*
- ◆ *разделение и объединение абзацев;*
- ◆ *поиск слов и словосочетаний;*
- ◆ *замена слов и словосочетаний;*
- ◆ *проверка орфографии.*

Для выполнения задания воспользуйтесь созданным ранее текстом из файла **Упражнение.doc**. Документ, который должен быть получен в результате выполнения задания, показан на рис. 3.12.

Технология работы

1. Запустите текстовый процессор Word.

2. Откройте документ **Упражнение.doc:**

- выберите команду меню **Файл** ▶ **Открыть**;
- в списке **Папка** выберите папку, в которой сохранен документ;

РЕДАКТИРОВАНИЕ ТЕКСТОВОГО ДОКУМЕНТА

При вводе текста с клавиатуры могут вкрасться ошибки.

Все возникающие ошибки можно разделить на три типа: лишняя буква, неверная буква, пропущенная буква.

Для устранения ошибок используются соответствующие методы: удаление, замена, вставка.

Применение перечисленных методов поможет вам отредактировать введенный текст. После устранения ошибок текст можно использовать для создания текстового документа.

Рис. 3.12. Образец текста (к заданию 3.2)

- выберите в окне документ Упражнение.doc;
 - подтвердите выбор документа щелчком на кнопке Открыть.
3. Выполните редактирование слов в режимах Вставка и Замена:
- выделите в первом предложении слова *В текст, введенный* и напечатайте вместо них слова *При вводе текста*;
 - перейдите в режим Замена, нажав клавишу Insert;



Обратите внимание на то, что на строке состояния в окне редактора загорелся индикатор ЗАМ (включен режим Замена).

- исправьте в первом предложении слово *может* на *могут*, установив курсор перед буквой *ж* и набрав на клавиатуре буквы *гу*;
- исправьте слово *ошибка* на *ошибки*, установив курсор перед буквой *а* и напечатав букву *и*;
- вернитесь в режим вставки, повторно нажав клавишу Insert;



Обратите внимание на то, что индикатор ЗАМ в строке состояния погас.

- установите курсор после слова *клавиатуры* и удалите лишнюю запятую, нажав клавишу Delete.
4. Копирование и вставка текста. Выполните копирование фрагмента текста с помощью контекстного меню, скопировав

словосочетание *устранения ошибок* из третьего предложения в начало последнего:

- выделите слова *устранения ошибок* в третьем предложении;
- вызовите контекстное меню, щелкнув правой кнопкой мыши на выделенной области;
- выберите в контекстном меню команду Копировать;
- поместите указатель мыши перед словом *текст* в последнем предложении;
- снова вызовите контекстное меню щелчком правой кнопки мыши;
- выберите в контекстном меню команду Вставить.

5. Удаление фрагментов текста. Отработайте различные способы удаления фрагментов текста.

Вариант 1

- выделите в последнем предложении слова *введенный с клавиатуры*;
- выберите в меню команду Правка ► Очистить ► Содержимое.

Вариант 2

- выделите в предпоследнем предложении текста слово *введенный*;
- выберите в меню команду Правка ► Вырезать;
- выделите слова *с клавиатуры* вместе с предшествующей запятой;
- выберите в меню команду Правка ► Очистить ► Содержимое;
- поместите курсор перед точкой в конце предложения;
- вызовите контекстное меню щелчком правой кнопки мыши и выберите в нем команду Вставить.



Обратите внимание на то, что вы произвели удаление двумя разными способами: инструмент Очистить содержимое просто удалил выделенный фрагмент, а инструмент Вырезать переместил выделенный фрагмент в буфер обмена.

Восстановить удаленный фрагмент из буфера обмена можно только в том случае, если использовался инструмент Вырезать.

6. Перемещение текста с помощью мыши. Переместите фрагмент текста с помощью мыши:

- выделите слово *текст* в предпоследнем предложении;
- наведите указатель мыши на выделенное слово, нажмите левую кнопку мыши и, не отпуская ее, перетащите слово в конец предложения, отпустив кнопку мыши, когда маркер вставки (вертикальная черта) будет находиться перед точкой.

7. Задайте абзацный отступ — левую границу первой строки абзаца:

- поместите курсор щелчком мыши в любое место внутри первого абзаца;
- на линейке вверху страницы переместите мышью верхний маркер (*маркер абзацного отступа*) на отметку 3 см;
- повторите предыдущие действия, установив для второго и третьего абзацев абзацный отступ 1,5 см.

8. Разделение одного абзаца на два. Разделите первый абзац на два абзаца:

- поместите курсор перед словом *Все* в начале второго предложения;
- нажмите клавишу Enter.



При таком разделении новые абзацы сохраняют абзацный отступ и другие настройки исходного абзаца.

9. Объединение абзацев. Соедините второй и третий абзацы:

- поместите курсор в конец второго абзаца;
- удалите маркер конца абзаца, нажав клавишу Delete. Маркер конца абзаца можно увидеть, если нажать на панели инструментов кнопку



При таком объединении абзацев новый абзац сохраняет абзацный отступ и другие настройки первого абзаца.

10. Отмените объединение абзацев, выбрав в меню команду Правка ▶ Отменить удаление.

11. Повторите операцию разделения и объединения для других абзацев. Отмените произведенные действия.
12. Установите первому и второму абзацам новый абзацный отступ — 1,5 см, выполнив действия, аналогичные указанным в п. 7.
13. **Поиск заданного слова в тексте.** Для поиска в тексте слова *буква* выполните следующие действия:
 - установите курсор на начало текста;
 - выберите в меню команду Правка ► Найти;
 - в открывшемся диалоговом окне укажите в поле Найти слово *буква*;
 - нажмите кнопку Найти далее — первое найденное слово будет выделено в тексте;
 - найдите все вхождения слова *буква*, нажимая каждый раз кнопку Найти далее;
 - когда поиск будет произведен по всему документу, появится окно с вопросом, надо ли продолжить поиск от начала документа, — нажмите кнопку Нет;
 - чтобы закрыть диалоговое окно Найти, нажмите кнопку Отмена или клавишу Esc.
14. **Поиск с заменой.** Замените в тексте все слова *буква* словом *символ*:
 - установите курсор на начало текста;
 - выберите в меню команду Правка ► Заменить;
 - в открывшемся диалоговом окне укажите в поле Найти слово *буква*, а в поле Заменить на — слово *символ*;
 - нажмите кнопку Найти далее, чтобы первое найденное слово *буква* было выделено в тексте;
 - нажмите кнопку Заменить для замены выделенного слова и перехода к следующему слову;
 - замените таким образом все слова *буква* в данном тексте;
 - когда все слова в тексте будут заменены, программа сообщит об этом, отобразив информационное окно с соответствующим текстом;

- закройте диалоговое окно Найти и заменить щелчком на кнопке Закрыть;
 - включите режим Замена;
 - замените окончания *-ля* и *-ая* у прилагательных, которые были согласованы со словом *буква*, на окончания мужского рода *-ий* и *-ый*;
 - отключите режим Замена.
15. Замените в тексте слово *метод* словом *прием*, выполнив действия, аналогичные указанным в п. 14.
16. Проверьте, везде ли окончания слов в местах замены соответствуют правилам русского языка.
17. Замените во всем тексте буквы *т* буквой *ч*:
- установите курсор на начало текста;
 - выберите в меню команду Правка ► Заменить;
 - в диалоговом окне Найти и заменить в поле Найти напечатайте букву *т*, а в поле Заменить на — букву *ч*;
 - нажмите кнопку Заменить все;
 - закройте диалоговое окно Найти и заменить щелчком на кнопке Закрыть.
18. Проверка орфографии. Проверьте орфографию, используя встроенный словарь программы Word:
- установите курсор в начало текста;
 - выберите в меню команду Сервис ► Правописание;
 - поскольку в тексте есть ошибки, появится окно Правописание, а в тексте будет выделено первое слово, в котором программа обнаружила ошибку;
 - вы можете исправить слово с ошибкой, используя в окне Правописание кнопку Заменить и список вариантов замены;
 - в случае, если в окне Варианты нет правильных вариантов, исправьте слово вручную в поле Нет в словаре и нажмите кнопку Исправить;
 - для часто повторяющихся ошибок, таких как *чексч* вместо *текст*, используйте кнопку Заменить все;

- когда в тексте не будет ошибок, программа отобразит окно с сообщением Проверка орфографии завершена.



Компьютер не предлагает исправить такие слова, как «могуч» и «чипа», так как эти слова не содержат ошибок. Замените вручную слово «могуч» на «могут», а слово «чипа» — на слово «типа».

19. Сохраните документ, выбрав в меню команду Файл ▶ Сохранить.
20. Закройте программу, выбрав в меню команду Файл ▶ Выйти.

Задания для самостоятельной работы

Задание 3.3. Создайте и отредактируйте текст собственной сказки по мотивам русской народной сказки «Теремок». Главным действующим лицом будет Системный блок компьютера, к которому хотят подключиться Мышь, Клавиатура, Монитор, Принтер и Модем. Сохраните текстовый документ в файле Сказка.doc.

Задание 3.4. Создайте и отредактируйте стихотворение, каждое четверостишие которого начинается со слов «Компьютер — это...». Сохраните текстовый документ в файле Компьютер.doc.

Задание 3.5. Создайте и отредактируйте текст приглашения родителям на празднование юбилея школы. Сохраните текстовый документ в файле Приглашение.doc.

Задание 3.6. Создайте и отредактируйте текст с небольшим рассказом о вашем классе, отметив его положительные характеристики. Сохраните текстовый документ в файле Класс.doc.

Тема 3.3 Форматирование текста

Выполнив задания этой темы, вы научитесь:

- ⇒ форматировать объекты текстового документа;
- ⇒ изменять параметры символов и абзацев;
- ⇒ оформлять текст в виде списка;
- ⇒ форматировать текст в несколько колонок.

Задание 3.7

Отработайте различные варианты оформления шрифта для выделенных фрагментов текста.

Для выполнения задания воспользуйтесь созданным ранее файлом Упражнение.doc. В результате работы должен получиться текст, показанный на рис. 3.13.

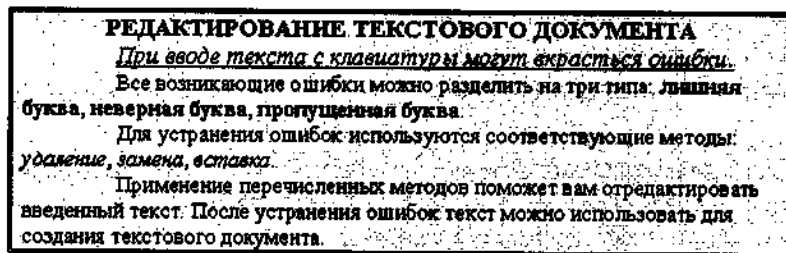


Рис. 3.13. Образец текста (к заданию 3.7)

Технология работы

Форматирование текста может выполняться за счет использования различных типов начертаний и размеров шрифта, инструментов оформления и заливки текста.

1. Запустите текстовый процессор Word и откройте документ Упражнение.doc так же, как вы делали это при выполнении задания 3.2.
2. Оформите заголовок текста **РЕДАКТИРОВАНИЕ ТЕКСТОВОГО ДОКУМЕНТА** полужирным шрифтом размером 14, используя кнопки панели инструментов и команды меню:

- выделите заголовок текста любым из известных вам способов;
 - выровняйте заголовок по центру, нажав кнопку  на панели инструментов;
 - выберите команду меню Формат ► Шрифт;
 - в открывшемся окне Шрифт выберите в списке шрифтов шрифт Times New Roman Cyr;
 - в поле Начертание выберите тип Полужирный;
 - в поле Размер выберите размер 14;
 - нажмите кнопку ОК.
3. Оформите предложение *При вводе текста с клавиатуры в него могут вкрасться ошибки* подчеркнутым курсивом размером 14, используя панель инструментов:
- выделите первое предложение текста;
 - измените начертание шрифта, поочередно нажав кнопки  и  на панели инструментов;
 - установите размер шрифта, выбрав размер 14 в списке на панели инструментов.
4. Выделите по образцу на рис. 3.13 остальные фрагменты текста:
- для слов *лишний символ, неверный символ, пропущенный символ* установите полужирное начертание шрифта (кнопка  на панели инструментов);
 - для слов *удаление, замена, вставка* — курсивное начертание шрифта (кнопка 
5. Заключите текст в рамку:
- выделите весь текст одним из известных вам способов;
 - выберите в меню команду Формат ► Границы и заливка;
 - в открывшемся окне на вкладке Граница выберите образец оформления рамки;
 - в списке Тип выберите двойную линию;
 - в списке Цвет выберите цвет рамки, например синий;
 - перейдите на вкладку Заливка и выберите цвет заливки, например бирюзовый;

- в списке Узор выберите значение 50 %;
 - в списке Цвет фона выберите белый цвет;
 - примените к тексту сделанные настройки, нажав кнопку ОК.
6. Сохраните документ.
7. Закройте текстовый процессор Word.

Задание 3.8

Познакомьтесь с основными технологическими операциями форматирования абзацев:

- ◆ изменение параметров абзаца с помощью команды меню Формат ► Абзац;
- ◆ изменение параметров абзаца с помощью панели инструментов;
- ◆ изменение параметров абзаца с помощью контекстного меню;
- ◆ использование команды меню Вставка ► Дата и время.

Для выполнения задания используйте документ Упражнение.doc. В результате работы должен получиться документ, показанный на рис. 3.14.

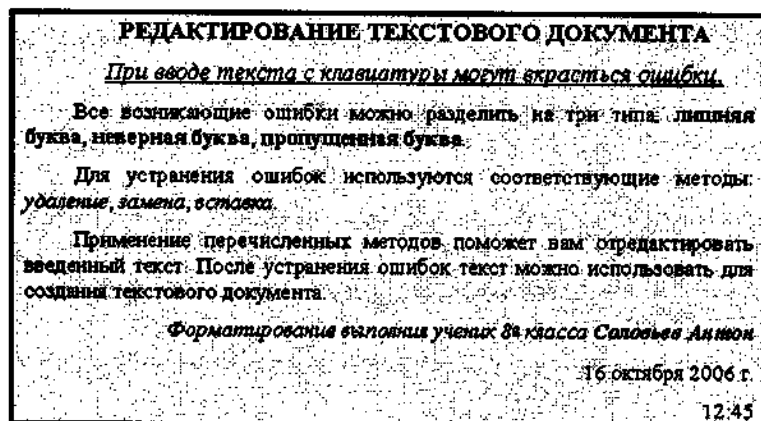


Рис. 3.14. Образец текста (к заданию 3.8)

Технология работы

1. Запустите текстовый процессор Word.
2. Откройте документ Упражнение.doc.
3. Выделите весь документ, выбрав команду меню Правка ► Выделить все.
4. Выберите команду меню Формат ► Абзац, в открывшемся окне выберите вкладку Отступы и интервалы.
5. Задайте новые параметры абзацев, указав перечисленные ниже значения в соответствующих полях:
 - выравнивание — по ширине;
 - отступ слева — 0,2 см,
 - отступ справа — 0,2 см;
 - отступ первой строки — 0,8 см;
 - интервал перед абзацем — 0 пт (пунктов);
 - интервал после абзаца — 6 пт;
 - межстрочный интервал — одинарный;примените сделанные изменения, нажав кнопку ОК.
6. Установите первым двум абзацам выравнивание По центру:
 - выделите два первых абзаца;
 - нажмите на панели инструментов кнопку .
7. Дополните текст еще одним абзацем:
 - установите курсор в конце последнего предложения и нажмите клавишу Enter;
 - введите с клавиатуры предложение *Форматирование выполнил ученик 8а класса Соловьев Антон*, заменив фамилию, имя и номер класса своими.
8. Отформатируйте фрагменты нового абзаца в соответствии с образцом (см. рис. 3.14) с помощью команды меню Формат ► Шрифт:
 - выделите все предложение и назначьте шрифту курсивное начертание;
 - выделите свои имя и фамилию и назначьте шрифту начертание Полужирный Курсив;
 - выделите букву класса и назначьте ей подчеркивание одной чертой и параметр надстрочный (верхний индекс).

9. Установите новому абзацу выравнивание вправо:
- поместите курсор в любое место внутри абзаца;
 - вызовите контекстное меню щелчком правой кнопки мыши и выберите в нем команду Абзац;
 - выберите вкладку Отступы и выравнивание и укажите в поле Выравнивание значение По правому краю;
 - закройте окно щелчком на кнопке ОК.
10. Дополните документ двумя абзацами с текущей датой и временем:
- поместите курсор в конец текста и нажмите клавишу Enter;
 - установите новому абзацу выравнивание По левому краю;
 - выберите команду меню Вставка ► Дата и время;
 - в появившемся диалоговом окне выберите формат даты 16 октября 2006 г. (доступные форматы показаны на примере текущей даты);
 - установите флажок Обновлять автоматически;
 - щелкните на кнопке ОК;
 - нажмите клавишу Enter, чтобы начать новый абзац;
 - выберите команду Вставка ► Дата и время еще раз;
 - в появившемся окне выберите формат времени 12:15 (доступные форматы показаны на примере текущего времени);
 - щелкните на кнопке ОК.
11. Сохраните документ.
12. Закройте текстовый процессор Word.

Задание 3.9*

Познакомьтесь с особыми формами текста и технологическими операциями, используемыми при их создании:

- ◆ оформление списка (бюллетеня) при помощи команды меню Формат ► Список;
- ◆ создание многоколоночного текста газетного типа при помощи команды меню Формат ► Колонки;
- ◆ создание колонтитулов.

Для выполнения задания воспользуйтесь созданным ранее файлом Упражнение.doc. В результате работы должен получиться документ, показанный на рис. 3.15.

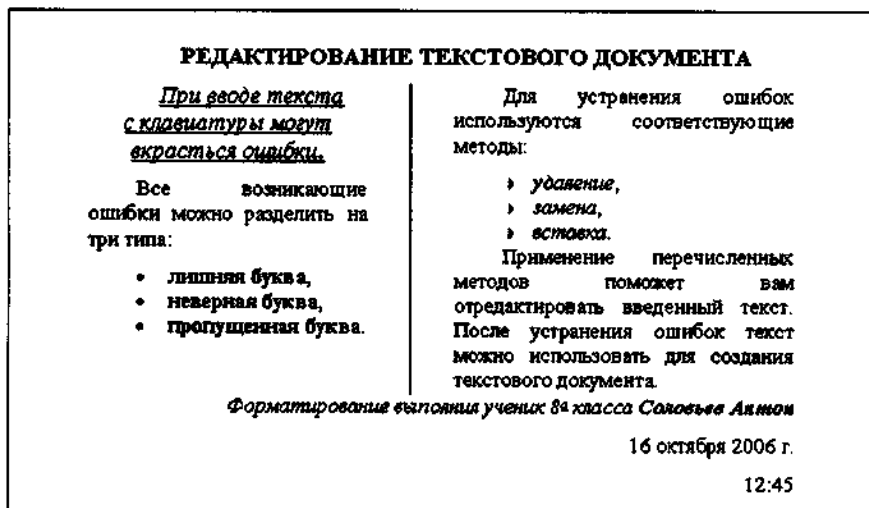


Рис. 3.15. Образец текста (к заданию 3.9)

Технология работы

1. Запустите текстовый процессор Word.
2. Откройте документ Упражнение.doc.
3. Удалите обрамление и заливку текста:
 - выделите весь текст любым из известных вам способов;
 - выберите команду меню Формат ► Границы и заливка;
 - в открывшемся окне выберите вкладку Граница;
 - в списке Тип выберите образец Нет;
 - перейдите на вкладку Заливка;
 - в поле Заливка выберите значение Нет заливки;
 - примените измененные параметры, щелкнув на кнопке ОК.
4. Оформите перечисления в тексте в виде списка:
 - расположите каждый из элементов перечислений (лишний символ, неверный символ, пропущенный символ; удаление,

замена, вставка) на отдельной строке, добавив после двоеточий и запятых знаки конца абзаца с помощью клавиши Enter;

- выделите три строки с первым перечислением и нажмите на панели инструментов кнопку ;
- выделите три строки со вторым перечислением и выберите команду меню Формат ▶ Список;
- в появившемся диалоговом окне щелкните на кнопке Изменить и выберите другой тип маркера (см. рис. 3.15);
- нажмите кнопку ОК.

5. Отформатируйте основную часть текста в две колонки:

- выделите основной текст (все абзацы, кроме заголовка и строк с подписью, датой и временем);
- выберите команду меню Формат ▶ Колонки;
- в появившемся окне укажите в поле Число колонок значение 2;
- в поле Промежуток укажите значение 1 см;
- установите флажок в поле Разделитель;
- примените измененные параметры щелчком на кнопке ОК.

6. Добавьте в текст колонтитулы:

- выберите команду меню Вид ▶ Колонтитулы — текстовый курсор переместится в область ввода верхнего колонтитула, а в окне программы появится панель инструментов Колонтитулы;
- введите в области ввода колонтитулов текст: *Справка по редактированию. Файл.*;
- вставьте после двоеточия пробел и название файла, выбрав на панели инструментов Колонтитулы команду Вставить автотекст ▶ Имя файла;
- задайте абзацу колонтитула выравнивание по центру с помощью кнопки ;
- щелкните на кнопке переключения верхнего и нижнего колонтитулов  на панели инструментов Колонтитулы — текстовый курсор переместится в область ввода нижнего колонтитула;

- напечатайте слово *Страница* и пробел;
- вставьте номер страницы, щелкнув на кнопке  на панели инструментов Колонтитулы;
- задайте абзацу выравнивание вправо с помощью кнопки ;
- вернитесь в основной текст, щелкнув на кнопке **Заккрыть** на панели инструментов Колонтитулы.

7. Сохраните документ.

8. Закройте файл, выбрав команду меню **Файл** ▶ **Заккрыть**.

Задания для самостоятельной работы

Задание 3.10. Откройте документ, созданный вами при выполнении самостоятельных работ в предыдущей теме (задания 3.3–3.6).

- ◆ Оформите заголовок текста полужирным шрифтом, размер 14.
- ◆ Все имена собственные в тексте выделите полужирным курсивом шрифтом Arial Cyr, размер 16.
- ◆ В каждом предложении выделите глаголы подчеркиванием.
- ◆ Сделайте обрамление двойной линией и тонировку текста зеленым цветом.
- ◆ Установите в первом абзаце выравнивание по центру; задайте отступы: справа — 1,4 см, слева — 2 см, отступ первой строки — 1 см; интервалы: межстрочный — полуторный, перед абзацем — 12 пт.
- ◆ Вставьте в документ текущую дату и время.
- ◆ Создайте бюллетень, содержащий список различных маркеров с их образцами, и придумайте для них названия. Для выполнения задания используйте команду меню **Формат** ▶ **Список**.
- ◆ Создайте верхний колонтитул с названием файла и нижний — с вашими фамилией и именем.

Тема 3.4 Оформление текста в виде таблицы и печать документа

Выполнив задания этой темы, вы научитесь:

- ⇒ создавать и редактировать таблицы;
- ⇒ печатать текстовый документ.

Задание 3.11*

Создайте таблицу получения цветов палитры (рис. 3.16) и сохраните ее в файле Цвета.doc.

Таблица цветов

Название цвета	Количество частей краски, %		
	Красная	Зеленая	Голубая
Черный	0	0	0
Оранжевый	255	179	0
Фиолетовый	185	6	255
Сиреневый	255	89	255
Желтый	255	255	134
Белый	255	255	255

Рис. 3.16. Таблица цветов палитры

Технология работы

1. Запустите текстовый процессор Word.
2. Создайте таблицу смешения цветов для палитры:
 - напечатайте заголовок *Таблица цветов* и нажмите клавишу Enter;
 - выберите команду меню Таблица ▸ Вставить ▸ Таблица;
 - в появившемся диалоговом окне укажите число строк — 8, число столбцов — 4;

- щелкните на кнопке Автоформат и выберите один из предложенных вариантов оформления таблицы из списка Стили таблицы; подтвердите выбор щелчком на кнопке ОК;
 - щелкните на кнопке ОК, чтобы добавить таблицу.
3. Объедините в первой строке таблицы вторую, третью и четвертую ячейки:
 - выделите ячейки, которые надо объединить: щелкните мышью во второй ячейке первой строки, нажмите левую кнопку мыши и, не отпуская ее, перемещайте мышшь до конца последней ячейки;
 - выберите в контекстном меню команду Объединить ячейки.
 4. Заполните таблицу данными в соответствии с рис. 3.16 и выделите заголовки, применив различные начертания шрифта.
 5. Сохраните документ в файле Цвета.doc.
 6. Закройте текстовый процессор Word.

Задание 3.12

Познакомьтесь с технологией подготовки документа к печати:

- ◆ *задание параметров страницы;*
- ◆ *просмотр документа перед печатью;*
- ◆ *установка конца страницы;*
- ◆ *печать текста.*

Для выполнения задания воспользуйтесь текстом, сохраненным в файле Упражнение.doc или Цвета.doc.

Технология работы

1. Запустите текстовый процессор Word.
2. Откройте документ.
3. Выберите команду меню Файл ► Параметры страницы.
4. В появившемся диалоговом окне на вкладке Поля установите следующие параметры страницы:
 - верхнее поле — 1,5 см;
 - нижнее поле — 1,5 см;
 - левое поле — 1,8 см;

- правое поле — 1,5 см;
 - ориентация страницы — альбомная.
5. Перейдите на вкладку **Источник бумаги** и задайте параметры колонтитулов:
- от края до верхнего колонтитула — 0,8 см;
 - от края до нижнего колонтитула — 1,1 см.
6. Перейдите на вкладку **Размер бумаги** и укажите размер страницы:
- ширина — 21 см;
 - высота — 14,8 см;
7. Щелкните на кнопке **ОК**, чтобы применить сделанные настройки и закрыть диалоговое окно.
8. Вставьте в текст символ конца страницы:
- установите курсор в конце документа;
 - выберите команду меню **Вставка** ▶ **Разрыв**;
 - в открывшемся окне установите переключатель **Начать: новую страницу** и щелкните на кнопке **ОК**.
9. Просмотрите документ перед печатью:
- выберите команду меню **Файл** ▶ **Предварительный просмотр**;
 - щелкните мышью на изображении страницы;
 - нажмите кнопку  для одновременного просмотра нескольких страниц;
 - проверьте корректность расположения колонтитулов, наличие номера страницы, имени файла и разделительной линии между колонками.
10. Напечатайте документ:
- выберите команду меню **Файл** ▶ **Печать**;
 - в появившемся диалоговом окне выберите весь диапазон страниц для вывода на печать, установив переключатель **все**;
 - укажите число копий — 1;
 - в поле **Напечатать** оставьте значение **Документ**;
 - щелкните на кнопке **ОК** для вывода документа на печать.
11. Сохраните документ.
12. Закройте текстовый процессор Word.

Задания для самостоятельной работы

Задание 3.13. Создайте и напечатайте таблицу, отражающую необходимые вам сведения о ваших друзьях и знакомых: фамилии, имена, адреса, телефоны, дни рождения и т. д.

Задание 3.14. Создайте и напечатайте таблицу, в которой были бы собраны основные сведения о каком-либо из ваших увлечений, например о музыкальных группах и исполнителях, художественных фильмах и актерах, марках автомобилей, спортсменах и т. д.

Тема 3.5* Включение в текстовый документ графических объектов

Выполнив задания этой темы, вы научитесь:

- создавать и редактировать рисунки, выполненные инструментами векторной графики;
- вставлять в текст и редактировать рисунки из коллекции картинок;
- вставлять в текст художественные заголовки;
- создавать и редактировать формулы.

Задание 3.15*

Создайте документ, содержащий схему — объект векторной графики (рис. 3.17).

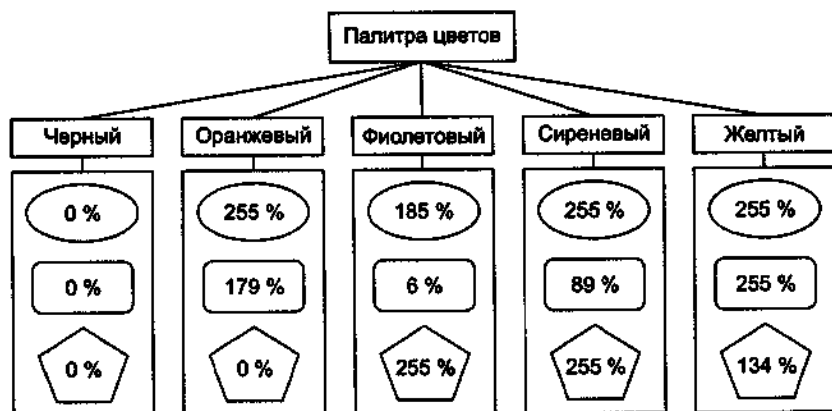


Рис. 3.17. Структурная схема палитры цветов

Технология работы

1. Запустите текстовый процессор Word.
2. Откройте файл Цвета.doc.

3. Создайте новый документ и сохраните его под именем **Схема.doc**:
- выберите команду меню **Файл** ▶ **Создать** — в правой части окна программы откроется область задач **Создание документа**;
 - щелкните в области задач на гиперссылке **Новый документ**;
 - выберите команду меню **Файл** ▶ **Сохранить**;
 - сохраните документ под именем **Схема.doc**.
4. Отобразите панель инструментов рисования, выбрав команду меню **Вид** ▶ **Панели инструментов** ▶ **Рисование** или щелкнув на кнопке **Рисование** на панели инструментов. 
5. Создайте рамку для надписи, используя инструменты панели **Рисование**:
- выберите инструмент **Прямоугольник**: в тексте документа появится прямоугольная область для создания рисунка — *полотно*; 
 - нарисуйте на полотне прямоугольную рамку аналогично тому, как вы делали это в графическом редакторе **Paint**;
 - выберите темно-синий цвет линии, вызвав палитру цветов щелчком на стрелке справа от кнопки **Цвет линий**; 
 - задайте толщину линии — 3 пт, щелкнув на кнопке **Тип линии**; 
 - отключите режим заполнения цветом внутренней части прямоугольника, щелкнув на стрелке справа от кнопки **Цвет заливки** и выбрав параметр **Нет заливки**. 
6. Добавьте надпись первой строки схемы — *Палитра цветов*:
- щелкните на кнопке **Надпись** на панели инструментов **Рисование**; 
 - щелкните мышью внутри прямоугольной рамки, созданной на предыдущем шаге, — появится рамка поля ввода текста;
 - наберите текст *Палитра цветов*;
 - сделайте невидимыми фон и рамку текстового поля, щелкнув сперва на стрелке справа от кнопки **Цвет линий** и выбрав значение **Нет линий**, а затем — на стрелке справа от кнопки **Цвет заливки** и выбрав параметр **Нет заливки**;

- выберите команду меню **Формат** ► **Шрифт**;
- в появившемся диалоговом окне выберите шрифт **Arial**;
- в поле **Начертание** выберите значение **Полужирный**;
- задайте размер шрифта — **14**;
- примените сделанные настройки, нажав кнопку **ОК**;
- переместите надпись мышью так, чтобы она располагалась по центру прямоугольника (см. рис. 3.17).



Каждый созданный объект располагается в рамке и его можно перемещать вместе с рамкой. Для редактирования объекта надо щелкнуть на нем мышью.

7. Создайте и расположите на полотне надписи и рамки, относящиеся к следующему уровню схемы, в соответствии с рис. 3.17:
 - выполните действия, аналогичные описанным в пп. 5 и 6. Текст можно скопировать через буфер обмена из файла **Палитра.doc**. Установите следующие параметры:
 - шрифт — **Arial**;
 - начертание — **курсив**;
 - размер — **12**;
 - толщина линии рамки — **1,5 пт**;
 - цвет линии выбирайте соответственно названию цвета в схеме.
8. Нарисуйте соединительные линии:
 - выберите инструмент **Линия** на панели инструментов **Рисование**; 
 - выберите толщину линии — **1 пт**;
 - нарисуйте линии, соединяющие надписи, аналогично тому, как вы это делали в графическом редакторе **Paint**.
9. Создайте и расположите надписи и рамки следующего уровня в соответствии с рис. 3.17, выполнив действия, аналогичные п. 7. Если размера полотна недостаточно, увеличьте его размер, переместив нижнюю границу за маркер выделения. Установите следующие параметры для объектов:
 - шрифт — **Arial**;
 - начертание — **обычный**;

- размер — 11;
- толщина линии рамок — 1 пт.

Для создания рамок используйте инструменты Овал, Скругленный прямоугольник, Правильный пятиугольник, доступные из меню Автофигуры ► Основные фигуры панели инструментов Рисование.

10. Создайте соединительные линии, относящиеся к следующему уровню схемы. Для этого выполните действия, аналогичные указанным в п. 8.
11. Щелкните на свободном пространстве полотна, чтобы появилась панель инструментов Полотно, и нажмите на ней кнопку Подобрать размер, чтобы обрезать выступающие части полотна.
12. Сгруппируйте созданные объекты в один объект:
 - выберите инструмент Выбор объектов на панели инструментов Рисование; 
 - обведите рамкой все объекты рисунка;
 - выберите команду меню Рисование ► Группировать.
13. Скопируйте рисунок в текстовый документ Цвета.doc:
 - скопируйте рисунок в буфер обмена, выбрав команду меню Правка ► Копировать;
 - выберите команду меню Окно ► Цвета.doc;
 - установите курсор в конце последнего абзаца и нажмите Enter;
 - выберите команду меню Правка ► Вставить.
14. Выполните форматирование рисунка:
 - щелкните на любом свободном месте полотна, чтобы выделить весь рисунок, и выберите команду меню Формат ► Полотно;
 - в открывшемся диалоговом окне перейдите на вкладку Размер;
 - установите флажок Сохранить пропорции;
 - в поле Масштаб по высоте укажите значение 80 %;
 - выберите вкладку Положение;
 - выберите образец обтекания текстом Вокруг рамки;
 - выберите горизонтальное выравнивание По центру;
 - примените настройки, щелкнув на кнопке ОК.
15. Сохраните документ.
16. Закройте текстовый процессор.

Задание 3.16*

Освойте технологические приемы вставки графических объектов в текстовый документ для следующих вариантов:

- ◆ вставка готовой картинку;
- ◆ редактирование и вставка рисунка;
- ◆ изменение размеров рисунка.

Для освоения технологии воспользуйтесь созданным ранее файлом Цвета.doc. Вставьте на титульный лист любую стандартную картинку из имеющейся коллекции картинок.

Технология работы

1. Запустите текстовый процессор Word.
2. Откройте документ Цвета.doc.
3. Подготовьте место для вставки:
 - поместите курсор в конце абзаца после слов *Таблица цветов*;
 - нажмите клавишу Enter для создания нового абзаца.
4. Вставьте готовую картинку:
 - выберите команду меню Вставка ► Рисунок ► Картинки — в правой части окна программы откроется область задач Коллекция клипов;
 - в области задач введите в поле Искать слово *бизнес* и нажмите кнопку Начать;
 - вставьте в текст понравившуюся вам картинку, щелкнув на ней мышью.
5. Выполните форматирование рисунка:
 - выделите рисунок, щелкнув на нем левой кнопкой мыши;
 - выберите команду меню Формат ► Рисунок;
 - в появившемся диалоговом окне выберите вкладку Размер;
 - сбросьте флажок Сохранить пропорции;
 - в поле Масштаб по высоте оставьте значение 100 %;
 - в поле Масштаб по ширине укажите значение 110 %;
 - выберите вкладку Положение;
 - выберите образец обтекания За текстом;
 - примените настройки, нажав кнопку ОК.

6. Сохраните документ, нажав кнопку Сохранить на панели инструментов. 
7. Закройте текстовый процессор.

Задание 3.17*

Вставьте в текстовый документ художественный заголовок.

Для освоения технологии воспользуйтесь созданным ранее файлом Цвета.doc; вставьте на титульный лист художественный заголовок в соответствии с рис. 3.18.

Таблица цветов

Рис. 3.18. Образец художественного заголовка (к заданию 3.17)

Технология работы

1. Запустите текстовый процессор Word.
2. Откройте документ Цвета.doc.
3. Подготовьте место для вставки художественного заголовка перед словами *Таблица цветов*:
 - поместите курсор в начало первой строки;
 - нажмите клавишу Enter для создания нового абзаца;
 - поместите курсор на новую (пустую) строку документа.
4. Создайте художественный заголовок:
 - выделите строку, содержащую слова *Таблица цветов*;
 - поместите ее в буфер обмена командой Правка ▸ Вырезать;
 - выберите команду меню Вставка ▸ Рисунок ▸ Объект WordArt;
 - в появившемся окне выберите нужный стиль надписи и нажмите кнопку ОК;
 - в следующем диалоговом окне вставьте содержимое буфера обмена, нажав комбинацию клавиш **Ctrl+V**;
 - выберите следующие параметры текста: шрифт — Arial, размер — 20, полужирный;
 - нажмите кнопку ОК;
 - щелчком мыши выделите художественный заголовок — появится панель инструментов WordArt;

- щелкните на кнопке Меню "Текст - Фигура" и выберите в раскрывшемся меню фигуру, в которую надо вписать текст, например Выпуклый верх.

5. Сохраните документ.

6. Закройте текстовый процессор.

Задание 3.18*

Создайте текстовый документ, содержащий формулы вычисления корней квадратного уравнения (рис. 3.19), используя инструмент создания и редактирования формул Microsoft Equations 3.0.

Корни квадратного уравнения можно вычислить по формуле:

$$x_{1,2} = \frac{-b \pm \sqrt{D}}{2a}$$

Если коэффициент b равен четному числу $b = 2k$, то корни вычисляются по формуле:

$$x_{1,2} = \frac{-k \pm \sqrt{k^2 - ac}}{a}$$

Рис. 3.19. Образец текста с формулами (к заданию 3.18)

Технология работы

1. Запустите текстовый процессор Word.
2. Создайте новый документ.
3. Наберите первое предложение в соответствии с рис. 3.19 и начните новый абзац.
4. Вставьте в текст первую формулу:
 - выберите команду меню Вставка ► Объект;
 - в появившемся диалоговом окне выберите тип объекта — Microsoft Equations 3.0.
5. Введите формулу в появившемся текстовом поле, используя для ввода математических знаков кнопки панели инструментов Формула:
 - наберите букву x ;
 - выделите букву x и выберите в меню команду Стилль ► Переменная;
 - поместите курсор справа от буквы x ;
 - щелкните на кнопке Шаблоны верхних и нижних индексов;
 - в раскрывшемся списке значков выберите значок нижнего индекса;

- введите через запятую индексы — 1,2;
 - переместите курсор в конец выражения, нажав клавишу управления курсором →;
 - введите знак =;
 - щелкните на кнопке Шаблоны корней и радикалов;
 - в раскрывшемся списке значков выберите значок вертикальной дроби обычного размера;
 - в поле над дробной чертой введите $-b$;
 - щелкните на кнопке со знаком «плюс-минус»;
 - щелкните на кнопке Шаблоны корней и радикалов;
 - в раскрывшемся списке значков выберите значок квадратного корня;
 - введите под знаком корня букву D ;
 - поместите курсор в поле ввода под знаком дроби;
 - введите $2a$;
 - завершите создание формулы, щелкнув в любом месте документа вне поля ввода формулы.
6. Наберите текст второго абзаца и начните новый абзац.
7. Введите вторую формулу, выполнив действия, аналогичные указанным в пп. 4 и 5.
8. Измените размер формул:
- выделите формулу щелчком мыши;
 - переместите мышью один из маркеров выделения, увеличив или уменьшив размер рамки.
9. Сохраните документ в файле Формула.doc.
10. Закройте текстовый процессор.

Задания для самостоятельной работы

Задание 3.19*. Нарисуйте схему классификации животных или птиц. Оформите название схемы художественным заголовком.

Задание 3.20*. Напишите рассказ о каком-либо событии вашей жизни и проиллюстрируйте его готовыми картинками из коллекции рисунков.

Задание 3.21*. Напечатайте фрагмент стихотворения и художественно оформите его, используя все известные вам инструменты текстового процессора.

РАЗДЕЛ 4 Освоение среды табличного процессора

Тема 4.1	Общая характеристика табличного процессора	126
Тема 4.2	Создание и редактирование табличного документа	143
Тема 4.3	Форматирование табличного документа. . .	150
Тема 4.4*	Использование функций и логических формул	157
Тема 4.5	Представление данных в виде диаграмм . .	167

Тема 4.1 **Общая характеристика табличного процессора**

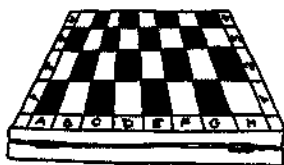
Изучив эту тему, вы узнаете:

- ➡ каковы назначение и основные возможности текстового процессора;
- ➡ из каких элементарных объектов состоит табличный документ;
- ➡ какие типы данных могут храниться в ячейках таблицы;
- ➡ что такое абсолютная и относительная адресация;
- ➡ правила записи и копирования формул и функций;
- ➡ что такое диаграмма и из каких объектов она состоит.

Назначение табличного процессора

Табличный процессор содержит набор компьютерных инструментов для работы с информацией, представленной в табличной форме — в виде *электронной таблицы*. Рабочая область электронной таблицы напоминает по своей структуре шахматную доску. Она состоит из строк и столбцов, имеющих свои имена.

Основное назначение табличного процессора — *автоматизация расчетов данных*, представленных в табличной форме. Результатом работы табличного процессора является документ в виде таблицы или диаграммы.



Например, в табличном процессоре можно вести классный журнал. Учителя могут заносить в него оценки учащихся, а встроенные формулы позволят вычислить средний балл для каждого ученика, общую успеваемость класса по предмету и т. п. Каждый раз, когда учитель вносит новую оценку, табличный процессор будет автоматически пересчитывать все результаты.

Характерной особенностью табличного процессора является то, что в нем данные и результаты вычислений представлены в таб-

личной форме. Для наглядности эти данные можно представить в графическом виде как диаграммы.

По сравнению со своей бумажной предшественницей электронная таблица предоставляет пользователю намного больше возможностей для работы. В клетки таблицы можно записывать не только числа, даты и тексты, но и логические выражения, функции и формулы. Формулы позволяют практически мгновенно производить пересчет и выводить в соответствующей ячейке новый результат при изменении исходных данных. Эта возможность позволяет активно использовать электронные таблицы в различных областях:

- ◆ для автоматизации вычислений;
- ◆ для представления результатов вычислений в виде диаграмм;
- ◆ для моделирования, когда исследуется влияние одних параметров на другие.

Объекты документа табличного процессора

Классификация объектов

Представление об объектах документа табличного процессора можно составить, познакомившись с их классификацией (рис. 4.1).



Рис. 4.1. Классификация объектов табличного документа

На первом уровне основанием классификации служит вид документа — таблица и диаграмма. Диаграмма является вспомогательным документом и без таблицы существовать не может. На втором уровне показаны основные объекты, которые образуют таблицу и диаграмму.

Объекты электронной таблицы

Таблица представляет собой сложный объект, состоящий из элементарных объектов: *строк*, *столбцов*, *ячеек*, *диапазонов ячеек* (рис. 4.2). Каждый элементарный объект имеет имя, которое определено разработчиками электронной таблицы.



Рис. 4.2. Элементарные объекты электронной таблицы

- ◆ **Строка.** Заголовки строк представлены в виде целых чисел, начиная с 1.
- ◆ **Столбец.** Заголовки столбцов задаются буквами латинского алфавита: сначала от A до Z, затем от AA до AZ, от BA до BZ и т. д.
- ◆ **Ячейка.** Адрес ячейки определяется ее положением в таблице и образуется из заголовков столбца и строки, на пересечении которых она находится. Сначала записывается заголовок столбца, а затем номер строки, например: A3, D6, AB46.
- ◆ **Диапазон ячеек.** Диапазон — это группа смежных ячеек: строка, несколько строк или часть строки; столбец, несколько столбцов или часть столбца; несколько смежных ячеек, образующих прямоугольную область. Диапазон ячеек задается указанием адресов его первой и последней ячеек, разделенных двоеточием.

Примеры:

- адрес ячейки, образованной пересечением столбца A и строки 7, — A7;
- адрес диапазона, образованного частью строки 3, — E3:G3;
- адрес диапазона, образованного частью столбца D, — D4:D8;
- адрес диапазона, имеющего вид прямоугольника с начальной ячейкой F5 и конечной ячейкой G8, — F5:G8.



Ячейка — элементарный объект электронной таблицы, расположенный на пересечении столбца и строки.

Строка — все ячейки, расположенные на одном горизонтальном уровне таблицы.

Столбец — все ячейки, расположенные в одном вертикальном ряду таблицы.

Диапазон ячеек — группа смежных ячеек, которая может состоять из строки, ее части или нескольких строк, столбца, его части или нескольких столбцов, а также из совокупности ячеек, охватывающих прямоугольную область таблицы; диапазоном можно также считать одну ячейку.

Таблица, создаваемая пользователем, представляет собой объект, характеризующийся определенным количеством строк и столбцов. Этому объекту присваивается имя, которое указывается в заголовке окна табличного документа. Над этим объектом можно производить действия, предусмотренные в табличном процессоре.

Параметры объектов таблицы представлены в табл. 4.1.

Таблица 4.1. Параметры объектов таблицы

Объект	Параметры объекта
Ячейка	Размер: ширина, высота. Тип данных: символьный, числовой, логический, дата, формула и др. Формат: заливка, обрамление, выравнивание, шрифт. Адрес: имя столбца (буква или буквы) и имя строки (число)
Строка	Наследуются все свойства объекта ячейка. Имя строки — целое число
Столбец	Наследуются все свойства объекта ячейка. Имя столбца — одна или две латинские буквы
Диапазон ячеек	Наследуются все свойства объекта ячейка. Адрес: адрес начальной ячейки и адрес последней ячейки, разделенные двоеточием
Таблица	Наследуются все свойства объекта ячейка

Объекты диаграммы

Диаграмма является объектом электронной таблицы и предназначена для представления данных в графической форме. Данные, находящиеся в одном столбце или в одной строке, называются **рядом**. Перед построением диаграммы сначала необходимо указать ряды, а затем перейти к выбору типа диаграммы. Для характеристики диаграммы служат следующие параметры: *имя, тип, область, размещение*.

- ◆ **Имя.** Диаграмме дается имя, под которым она включается в состав электронной таблицы.
- ◆ **Тип.** Табличный процессор позволяет строить диаграммы различных типов. Основные типы диаграмм показаны на рис. 4.3. Рассмотрим подробнее некоторые из них.
- **График** хорошо знаком вам по различным школьным задачам. На одном чертеже можно разместить несколько графиков, каждый из которых будет соответствовать своему ряду данных.



Рис. 4.3. Типы диаграмм

- **Гистограмма**, или **столбчатая диаграмма**, в отличие от круговых, может быть построена для нескольких рядов данных. Высота каждого столбика определяется значением в соответствующей ячейке. Примером гистограммы может

служить диаграмма распределения роста учеников каждого класса. Здесь каждый ряд представляет собой набор значений роста учеников каждого класса. Очевидно, что в электронной таблице каждый ряд занимает одну строку или один столбец.

- *Поверхностная диаграмма* строится только для нескольких рядов и представляет собой группу многослойных разноцветных поверхностей. Каждый слой соответствует одному ряду данных.
 - *Круговая диаграмма* (плоская или объемная) используется для графического отображения одного ряда значений. Таким рядом может быть, например, набор значений роста или веса учащихся. Каждый сектор такой диаграммы отражает относительную (выраженную в процентах) долю каждого значения из ряда от общей суммы всех значений.
 - *Смешанная диаграмма* используется для отображения нескольких рядов данных с использованием разных типов диаграмм.
 - ◆ **Область.** Ограничивает поле чертежа построения диаграммы.
 - ◆ **Размещение.** Диаграмма может размещаться либо на том же листе, что и таблица, либо на отдельном листе.
- Диаграмма состоит из следующих объектов (рис. 4.4): *ряд, ось, заголовок, легенда, область построения.*
- ◆ **Ряд.** Диаграмма может быть построена как по одному ряду, так и по нескольким рядам. Для выделенного диапазона ячеек построение диаграммы ведется по нескольким рядам данных. В этом случае в качестве каждого ряда принимается соответствующая строка или столбец выделенного диапазона. Ряд может содержать подписи отображаемых значений.
 - ◆ **Ось.** Каждая из осей диаграммы характеризуется следующими параметрами: *вид, шкала, шрифт, число.*
 - Вид определяет отображение внешнего вида оси на экране.
 - Шкала определяет минимальное и максимальное значения шкалы, цену основных и промежуточных делений, точку пересечения с другими осями.
 - Число определяет формат шкалы в соответствии с типами данных, находящихся в диапазоне.

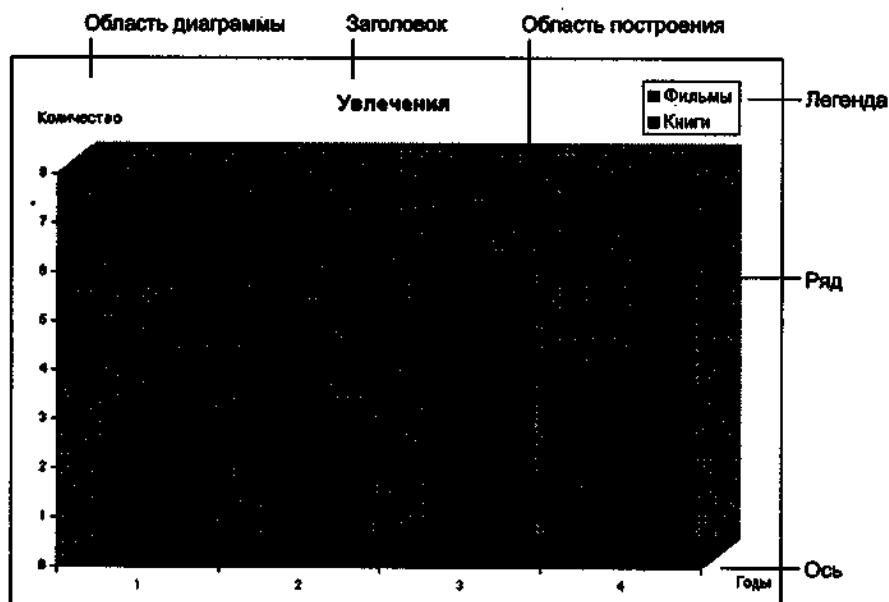


Рис. 4.4. Объекты диаграммы

- ◆ **Заголовок.** Заголовком служит текст, определяемый пользователем; как правило, заголовок размещается над диаграммой.
- ◆ **Легенда.** При выводе к диаграмме можно добавить легенду — список названий рядов (обозначений переменных).
- ◆ **Область построения.** Это область, ограниченная осями и предназначенная для размещения рядов данных. Для удобства анализа результатов на область построения может наноситься сетка.

Способы выделения объектов электронной таблицы

- ◆ Для выделения *ячейки* нужно щелкнуть на ней мышью или переместить на нее курсор с помощью клавиш управления курсором.
- ◆ Для выделения *строки* нужно щелкнуть на ее заголовке.
- ◆ Для выделения *столбца* нужно щелкнуть на его заголовке.
- ◆ Выделение *диапазона ячеек* может быть выполнено несколькими способами:

- при помощи мыши — перемещая мышь при нажатой левой кнопке от начала к концу диапазона;
 - при помощи клавиш управления курсором — при нажатой клавише Shift;
 - вводом с клавиатуры начального и конечного адресов ячеек диапазона, разделенных двоеточием.
- ◆ Для выделения *диаграммы* необходимо выбрать диаграмму двойным щелчком мыши, а затем щелкнуть на нужном объекте.

Типовые действия над объектами электронной таблицы

Все действия над объектами электронной таблицы, кроме форматирования, редактирования и вычислений, выполняются так же, как и описанные ранее действия над объектами текстового документа. Технологии работы с инструментами табличного процессора аналогичны технологии работы в среде текстового процессора. Особенности действий, связанных с форматированием объектов, мы рассмотрим в теме 4.3.

Данные электронной таблицы

Общие сведения

В табличных процессорах предусмотрены различные форматы представления данных. Форматы определяют типы данных электронной таблицы: символьные (текстовые), числовые, логические, даты и т. д.

Так, например, для переноса разных продуктов человек использует разные емкости, контейнеры и т. п. Для того чтобы принести домой разливное молоко, вы используете бидон или банку, а не сумку или сетку. Если же вы отправились за картошкой, то вам понадобится как раз сетка или мешок, в то время как бидон и банка окажутся бесполезными. Яйца можно принести и в сетке, и в бидоне, но для повышения сохранности лучше воспользоваться специальным контейнером.

От того, какой формат выбран для той или иной ячейки, будет



зависеть, какие действия табличный процессор сможет выполнять над ее содержимым.

Представьте себе, что в ячейку записаны цифры 120399. Как они будут восприняты табличным процессором? Какие действия будут над ними произведены? Если установлен текстовый формат, цифры будут восприняты как последовательность символов 1, 2, 0, 3, 9, 9. Если установлен числовой формат, они могут быть восприняты электронной таблицей как число. Если же ячейке назначен формат даты, они будут восприняты как 12 марта 1999 года.

Текстовый тип данных

Текстовые данные представляют собой некоторый набор символов. Если первый из них является буквой, кавычкой, апострофом или пробелом или если цифры в нем чередуются с буквами, то такая запись воспринимается как текст.

Действия над символьными данными производятся аналогично действиям над объектами в текстовом процессоре.

Примеры текстовых (символьных) данных:

<i>Расписание занятий</i>	<i>8 «А» класс</i>
<i>"236</i>	<i>001 счет</i>

Числовой тип данных

Числовые данные представляют собой последовательность цифр, которые могут быть разделены десятичной запятой и начинаться с цифры, знака числа (+ или -) или десятичной запятой. Над числовыми данными в электронной таблице могут производиться различные математические операции.

Примеры числовых данных:

<i>232,5</i>	<i>,546</i>
<i>-13,7</i>	<i>+100</i>



Запомните! Если в ячейке таблицы хранится последовательность цифр, начинающаяся с кавычки, то даже если такой набор цифр выглядит на экране как число, это — текст, и его нельзя использовать в вычислениях. Любые текстовые данные всегда воспринимаются как ноль.

Укажем косвенный признак, по которому можно отличить текстовые и числовые данные. Если значение в ячейке после ввода распознано табличным процессором как текст, то по окончании ввода (после нажатия на клавишу Enter) они автоматически выравниваются по левой границе ячейки. Данные, распознанные как числовые, после ввода выравниваются по правой границе ячейки (рис. 4.5). Однако этот признак не является показательным, если выравнивание в ячейках было задано вручную.

Текст	12,5
023	0,023
1456	-1255

Рис. 4.5. Выравнивание данных в ячейках

Логический тип данных

Логические данные используются в логических формулах и функциях. Данные этого типа отображаются в ячейке следующим образом: если вводится любое отличное от нуля число (целое или дробное), то после нажатия на клавишу Enter в ячейке будет выведено значение Истина. Ноль отображается в соответствующей ячейке как Ложь.

Это представление данных связано с понятием *логической переменной*, которая используется в алгебре логики. Она служит для описания высказываний, которые могут принимать одно из двух возможных значений: «истина» (логическая единица) либо «ложь» (логический ноль).

Тип данных — даты

Этот тип данных используется при выполнении таких функций, как добавление к дате числа, вычисление разности двух дат или пересчет даты вперед или назад. Пересчет чисел в даты производится автоматически в зависимости от заданного формата. Табличный процессор позволяет представлять вводимые числа как даты несколькими способами.

Примеры представления дат в разных форматах:

12 апреля 2006 12.04.2006

Апрель 2006 04.2006

Апрель 12 апреля

Формулы

Понятие формулы

Назначение электронной таблицы в первую очередь состоит в автоматизации вычислений. Для этого в ячейки таблицы вводятся **формулы**.

Ввод любой формулы начинается со знака равенства. Если его пропустить, то вводимая формула будет воспринята как текст.

В формулы можно включать числовые данные, знаки операций, различные функции, а также адреса объектов таблицы. Формулу, в которой содержатся адреса ячеек, можно сравнить с записью уравнения в математике, где вместо адресов ячеек используются переменные.

Адреса, которые используются в формулах, получили название **ссылок**. Ссылки позволяют связывать между собой любые ячейки электронной таблицы и производить необходимую обработку табличных данных.



Ссылка — это адрес объекта (ячейки, строки, столбца, диапазона ячеек), используемый при записи формулы.

Формулы представляют собой соединенные знаками арифметических и логических операций **операнды**. Операндом может быть конкретное значение, ссылка или функция.

Различают арифметические (алгебраические) и логические формулы.

Арифметические формулы. Арифметические формулы аналогичны математическим соотношениям. В них используются арифметические операции (сложение «+», вычитание «-», умножение «*», деление «/», возведение в степень «^»). При вычислении по формулам соблюдается принятый в математике порядок выполнения арифметических операций: сначала выполняется возведение в степень, затем — умножение и деление, после этого — сложение и вычитание. Операции одного уровня, такие как умножение и деление, выполняются слева направо.

Для изменения порядка выполнения арифметических операций используют круглые скобки. Действия над операндами, заключенными в круглые скобки, выполняются в первую очередь.

Результатом вычислений по арифметической формуле является число. При каждом изменении входящих в формулу операндов результат пересчитывается заново и отображается в соответствующей ячейке.

Пример вычисления по арифметическим формулам:

Пусть в ячейке C3 введена формула $=A1+7*B2$, а в ячейках A1 и B2 находятся числовые данные 3 и 5 (рис. 4.6, а). Тогда при вычислении по заданной формуле сначала будет выполнена операция умножения числа 7 на содержимое ячейки B2 (5) и к этому числу будет прибавлено содержимое ячейки A1 (3). Полученный результат, равный 38, будет отображен в ячейке C3, в которой находится формула (рис. 4.6, б).

В данной формуле A1 и B2 представляют собой ссылки на ячейки. Смысл использования ссылок состоит в том, чтобы при изменении значений операндов автоматически изменялся результат вычислений, выводимый в ячейке C3. Например, пусть значение в ячейке A1 стало равным 1, а значение в ячейке B2 — 10. Тогда в ячейке C3 появляется новое значение — 71 (рис. 4.6, в).

	A	B	C
	3		
		5	
			$=A1+7*B2$

а

	A	B	C
	3		
		5	
			38

б

	A	B	C
	1		
		10	
			71

в

Рис. 4.6. К примеру вычислений по арифметическим формулам

Логические формулы. Логическая формула содержит условие и определяет, истинно оно или ложно. Истинному выражению присваивается значение «истина» (1), а ложному — «ложь» (0).

Однотипные формулы

При работе с электронной таблицей часто возникает необходимость в заполнении некоторого диапазона ячеек формулами, имеющими одинаковую структуру, но разные значения переменных. Это означает, что они различаются ссылками. Подобные формулы получили название однотипных.



Однотипные (подобные) формулы — это формулы, которые имеют одинаковую структуру (строение) и различаются только конкретными ссылками.

Для упрощения и ускорения ввода однотипных формул используется следующий прием. Формулу вводят только в одну (начальную) ячейку, после чего ее копируют в другие ячейки.

Пример однотипных формул:

=A1+5	=A1*5	=A1+B3	=A1*B3	=(A1+B3)*D2
=A2+5	=B1*5	=A2+B4	=B1*C3	=(C3+D5)*F4
=A3+5	=C1*5	=A3+B5	=C1*D3	=(D4+E6)*G5
=A4+5	=D1*5	=A4+B6	=D1*E3	=(B4+C6)*E5

Относительная, абсолютная и смешанная адресация

В однотипных формулах могут быть использованы самые разные ссылки. Возможны однотипные формулы, в которых часть ссылок при переходе от одной формулы к другой изменяются закономерным образом, а другие входящие в формулу ссылки остаются неизменными для всех формул.

При копировании формулы в другое место таблицы прежде всего необходимо определить способ автоматического изменения входящих в нее ссылок. Для этого используются относительные, абсолютные и смешанные ссылки.

- ◆ **Относительная ссылка** используется в формуле в том случае, если она должна изменяться при копировании.

Пример:

Относительная ссылка записывается в обычной форме, например F3, E7. Во всех ячейках, куда она будет помещена после копирования, изменяется и буква столбца, и номер строки.

- ◆ **Абсолютная ссылка** используется в формуле в том случае, если при ее копировании не должны изменяться обе части: буква столбца и номер строки.

Пример:

В абсолютной ссылке перед буквой столбца и перед номером строки ставится символ \$, например \$F\$3, \$E\$7. При копировании во все ячейки формула будет помещена в неизменном виде.

- ◆ **Смешанная ссылка** используется, когда при копировании формулы меняется только какая-то одна часть ссылки — либо буква столбца, либо номер строки. При этом символ \$ ставится перед той частью ссылки, которая должна остаться неизменной.

Пример:

Смешанные ссылки с неизменяемой буквой столбца: \$C8, \$F12. *Смешанные ссылки с неизменяемым номером строки:* A\$5, F\$9.



Абсолютная ссылка — это ссылка, не изменяющаяся при копировании формулы.

Относительная ссылка — это ссылка, автоматически изменяющаяся при копировании формулы.

Смешанная ссылка — это ссылка, частично изменяющаяся при копировании.

Правило изменения относительных ссылок при копировании формул

Для правильного заполнения ячеек при копировании формул важно понимать *правило относительной ориентации*. Это правило объясняет, как изменяются входящие в скопированные формулы относительные или смешанные ссылки.



Правило относительной ориентации:

Относительные ссылки в формуле определяют взаимное расположение ячеек с исходными данными и ячейки, в которой хранится результат вычисления.

Поясним смысл этого правила. На рис. 4.7 показан пример записи в ячейку C2 формулы, содержащей относительные ссылки на данные, хранящиеся в ячейках A1 и B4. Эти относительные ссылки указывают на взаимное расположение в таблице ячеек с исходными данными — операндов A1 и B4 — и результата вычисления по формуле, записанной в C2.

	A	B	C
1			
2			=A1+B4
3			
4			

Рис. 4.7. Относительная ориентация ячеек

Правило относительной ориентации выполняется табличным процессором следующим образом:

- ◆ ссылка A1 указывает, что значение первого операнда берется из ячейки, расположенной на одну строку выше и на два столбца левее той, в которую будет помещена формула (здесь — C2);
- ◆ ссылка B4 указывает, что значение второго операнда берется из ячейки, смещенной на две строки вниз и на один столбец влево относительно места записи формулы (здесь — C2).

Правила копирования формул

1. Ввести формулу, указав в ней относительные и абсолютные ссылки. Такая формула представляет собой образец (шаблон), указывающий положение ячеек, в которых хранятся операнды, относительно положения ячейки с формулой.
2. После ввода исходной формулы необходимо скопировать ее в нужные ячейки. При копировании формул действует правило относительной ориентации ячеек, благодаря которому обеспечивается автоматическая настройка относительных ссылок во всех копиях формулы. Для запрета автоматического изменения ссылок при копировании следует использовать абсолютные ссылки в исходной формуле.

Порядок копирования формулы из ячейки в диапазон:

- ◆ выделите ячейку с исходной формулой;
- ◆ скопируйте формулу в буфер обмена;
- ◆ выделите диапазон ячеек, в который должна быть скопирована исходная формула;
- ◆ вставьте формулу из буфера, заполнив ею все ячейки выделенного диапазона.

Порядок копирования формул из одного диапазона в другой:

- ◆ выделите диапазон с формулами, которые должны быть скопированы;
- ◆ скопируйте формулы из выделенного диапазона в буфер обмена;
- ◆ установите курсор на первую ячейку диапазона, в который требуется скопировать формулы;
- ◆ вставьте формулы из буфера.

Пример копирования формул с использованием ссылок различных типов приведен на рис. 4.8: а) с относительными ссылками; б) с абсолютными ссылками; в) со смешанными ссылками.

а	A B C		
			5
		2	
=C1+B2			
б	A B C		
			5
		2	
=\$C\$1+\$B\$2			
в	A B C		
			5
		2	
=C1+B\$2			

			5
	2		
=C1+B2	=D1+C2	=E1+D2	

			5
	2		
=\$C\$1+\$B\$2	=\$C\$1+\$B\$2	=\$C\$1+\$B\$2	

			5
	2		
=C1+B\$2	=C1+C\$2	=C1+D\$2	

Рис. 4.8. Пример копирования формул с относительными (а), абсолютными (б) и смешанными ссылками (в)

Контрольные вопросы и задания

1. Перечислите объекты электронной таблицы.
2. Как задается адрес диапазона ячеек?
3. Перечислите параметры ячейки.
4. Из каких элементарных объектов состоит любая диаграмма?
5. Назовите основные типы диаграмм в табличном процессоре.
6. Назовите параметры, характеризующие любую диаграмму.
7. Назовите типы данных, используемых в табличном процессоре, и их особенности. Приведите примеры.
8. Что такое ссылка в электронной таблице и чем она отличается от адреса?
9. Что такое относительная адресация в формулах? Приведите примеры.
10. Что такое абсолютная адресация в формулах и как она обозначается? Приведите примеры.

11. Поясните на примере правило относительной ориентации.
12. Объясните на примерах правило автоматического изменения относительных ссылок при копировании.
13. Расскажите правила копирования формул.
14. Объясните на примерах результат копирования формулы, которая содержит:
 - а) константы и относительные ссылки;
 - б) относительные и абсолютные ссылки;
 - в) смешанные ссылки;
 - г) константы и все виды ссылок.

Тема 4.2 Создание и редактирование табличного документа

Выполнив задания этой темы, вы научитесь:

- ➡ создавать и заполнять таблицу данными;
- ➡ использовать в таблице простые формулы;
- ➡ копировать формулы в другие ячейки таблицы.

Задание 4.1

*Создайте таблицу, содержащую расписание движения поездов от станции Бологое до станции Тверь (рис. 4.9). Сохраните таблицу в файле *Расписание.xls*.*

В процессе выполнения задания вы познакомитесь с основными технологическими приемами ввода и редактирования данных в электронной таблице.

Станция	Прибытие	Отправление
Бологое		7:45
Академический	8:19	8:21
Верхний Волочек	8:48	8:53
Спирово	9:31	9:34
Калашниково	9:56	9:59
Лихославль	10:24	10:29
Чернуково	10:53	10:55
Тверь	11:12	

Рис. 4.9. Таблица Расписание
(к заданию 4.1)

Технология работы

1. Запустите табличный процессор.
2. Создайте новый документ или воспользуйтесь документом, созданным по умолчанию.
3. Заполните таблицу в соответствии с образцом (рис. 4.9):
 - щелчком мыши выберите ячейку, в которую будете вводить текст;
 - наберите текст и нажмите клавишу Enter;
 - выберите следующую ячейку и повторите действия, описанные выше.

Чтобы числа при вводе были автоматически распознаны как время в часах и минутах, разделяйте часы и минуты двоеточием, как показано на образце.



Обратите внимание! Текст в ячейках автоматически выравнивается по левому краю, а время — по правому. Используйте это свойство табличного процессора, чтобы определить, правильно ли он распознал введенный текст.

4. Отредактируйте название станции в ячейке A4:

- выберите ячейку A4;
- для перехода в режим редактирования значения ячейки выполните одно из возможных действий: нажмите клавишу F2, щелкните мышью в строке формул, где отображено значение ячейки, или дважды щелкните мышью в ячейке;
- замените слово *Верхний* на слово *Вышний* и нажмите клавишу Enter.

5. Аналогичным образом измените название станции *Черноруково* в ячейке A8 на *Черногубово*.

6. Выровняйте ширину столбца A так, чтобы был виден весь текст, помещенный в ячейку:

Способ 1 — с помощью мыши:

- Подведите указатель мыши к верхней строке таблицы, где размещены заголовки столбцов (буквы), и установите его на правой границе ячейки с буквой A, чтобы указатель принял вид вертикальной черты, пересеченной двусторонней горизонтальной стрелкой —
- дважды щелкните левой кнопкой мыши на разделительной черте — в результате ширина столбца будет подобрана автоматически по самому длинному слову, размещенному в ячейках этого столбца.

Можно также перетащить границу столбца, нажав левую кнопку мыши и переместив указатель в нужную сторону.

Способ 2 — с помощью меню:

- Выделите столбец A, щелкнув мышью на его заголовке;
 - выберите команду меню **Формат** ▸ **Столбец** ▸ **Автоподбор ширины**.
7. Аналогичным образом выровняйте ширину столбцов B и C так, чтобы все значения ячеек были видны полностью.
8. Сохраните документ:
- выберите команду меню **Файл** ▸ **Сохранить как**;

- в появившемся диалоговом окне выберите папку, в которой следует сохранить файл;
- в поле Имя файла укажите название файла — Расписание;
- в поле Тип файла оставьте тип, предложенный по умолчанию, — Книга Microsoft Office Excel (.xls);
- щелкните на кнопке Сохранить.

9. Закройте табличный процессор, выбрав команду меню Файл ► Выход.

Задание 4.2*

Дополните таблицу в файле Расписание.xls расчетами времени стоянок поезда в каждом населенном пункте. Вычислите суммарное время стоянок, общее время в пути и время, затрачиваемое поездом на передвижение от одного населенного пункта к другому.

В процессе выполнения задания вы познакомитесь со следующими технологическими операциями:

- ◆ перемещение данных;
- ◆ ввод расчетных формул;
- ◆ копирование формул;
- ◆ изменение формата данных.

В результате выполнения этого задания вы должны получить таблицу, приведенную на рис. 4.10.

Станция	Прибытие	Стоянка	Отправление	Время в пути
Бологое			7:45	
Академический	8:19	0:02	8:21	0:34
Вышний Волочек	8:48	0:05	8:53	0:27
Спирово	9:31	0:03	9:34	0:36
Калашниково	9:56	0:03	9:59	0:22
Лихославль	10:24	0:05	10:29	0:25
Черногубово	10:53	0:02	10:55	0:24
Тверь	11:12			0:17
Общее время стоянок		0:20		
Общее время в пути:				3:27

Рис. 4.10. Таблица Расписание после выполнения задания 4.2

Технология работы

1. Запустите табличный процессор.
2. Откройте документ *Расписание.xls*.
3. Переместите данные о времени отправления из столбца C в столбец D:
 - выделите диапазон C1:C8;
 - выберите команду меню *Правка* ► *Вырезать*;
 - установите курсор в ячейку D1;
 - выберите команду меню *Правка* ► *Вставить*;
 - подберите ширину столбца в соответствии с шириной текста.
4. Введите текст в ячейку C1:
 - выделите ячейку C1;
 - введите текст *Стоянка*;
 - подберите ширину столбца в соответствии с шириной текста.
5. Создайте формулу, вычисляющую время стоянки в населенном пункте:
 - выберите ячейку C3;
 - наберите на клавиатуре знак равенства;
 - для ввода адреса ячейки с первым значением (временем отправления) щелкните левой кнопкой мыши в ячейке D3;
 - наберите на клавиатуре знак «минус»;
 - щелкните мышью в ячейке B3;
 - нажмите клавишу *Enter*, чтобы завершить ввод формулы.



Обратите внимание! В ячейке, содержащей формулу, отображается результат вычислений. Саму формулу можно посмотреть в строке ввода.

6. Скопируйте формулу из ячейки C3 в каждую ячейку диапазона C4:C8 с помощью команды меню:
 - выделите ячейку C3, откуда надо скопировать формулу;
 - выберите команду меню *Правка* ► *Копировать* или аналогичную команду контекстного меню;

- выделите диапазон ячеек C4:C8, куда надо вставить копии формулы;
- выберите команду меню Правка ► Вставить или аналогичную команду контекстного меню.



Обратите внимание! При копировании формулы относительные ссылки автоматически изменяются относительно ее нового местоположения в соответствии с правилом их вхождения в формулу (см. тему 18 в учебнике).

7. Введите текст в ячейку E1:

- выделите ячейку E1;
- введите текст *Время в пути*;
- выровняйте ширину столбца в соответствии с шириной текста.

8. Создайте формулу расчета времени, затрачиваемого поездом на передвижение от одного населенного пункта к другому:

- выберите ячейку E3;
- наберите на клавиатуре знак равенства;
- введите адрес ячейки B3;
- наберите на клавиатуре знак «минус»;
- введите адрес ячейки D2;
- нажмите клавишу Enter.

9. Скопируйте формулу из ячейки E3 в каждую ячейку диапазона E4:E9 с помощью мыши:

- выделите ячейку E3, откуда надо скопировать формулу;
- наведите указатель мыши на правый нижний угол этой ячейки — указатель примет вид черного креста ;
- нажмите левую кнопку мыши и протащите рамку вниз на диапазон E4:E9.

10. Измените формат ячеек для диапазонов C2:C10 и E2:E10 (этот пункт следует выполнять только в том случае, если необходимо изменить формат представления времени):

- выделите диапазон ячеек C2:C10;
- выберите команду меню Формат ► Ячейки;

- в появившемся диалоговом окне на вкладке Число выберите формат Время, тип 13:30 (часы:минуты);
 - нажмите кнопку ОК;
 - выделите диапазон ячеек E2:E10 и повторите указанные действия.
11. Введите текст в ячейку A10:
- выберите ячейку A10;
 - введите текст *Общее время стоянок*;
 - выровняйте ширину столбца в соответствии с шириной текста.
12. Создайте формулу для вычисления общего времени стоянок поезда:
- выберите ячейку C10;
 - нажмите кнопку Автосумма на панели инструментов; 
 - выделите мышью диапазон ячеек C3:C8 и нажмите клавишу Enter.
13. Удалите содержимое ячейки C3:
- выберите ячейку C3;
 - выберите команду меню Правка ▸ Очистить или нажмите на клавишу Delete на клавиатуре.
- Обратите внимание: компьютер автоматически пересчитает сумму в ячейке C9.
14. Выберите команду меню Правка ▸ Отменить, чтобы отменить удаление.
15. Введите текст в ячейку A11:
- выберите ячейку A11;
 - введите текст *Общее время в пути*.
16. Создайте формулу для вычисления общего времени в пути:
- выберите ячейку E11;
 - наберите на клавиатуре знак равенства;
 - щелкните в ячейке B9, содержащей время прибытия в конечный пункт;
 - наберите на клавиатуре знак «минус»;

- щелкните в ячейке D2, содержащей время отправления из начального пункта;
- нажмите клавишу Enter.

17. Сохраните документ, нажав кнопку Сохранить.

18. Закройте табличный процессор, выбрав команду меню Файл ► Выход.

Задания для самостоятельной работы

Задание 4.3. Создайте таблицу расчета стоимости всех продуктов, используемых для приготовления завтрака, обеда и ужина в доме отдыха из расчета на одного человека.

Задание 4.4. Создайте таблицу, в которой отражаются затраты на покупку продуктов питания ежедневно, еженедельно, ежемесячно.

Задание 4.5. Создайте таблицу расчета на свободную тему.

Тема 4.3 **Форматирование табличного документа**

Выполнив задания этой темы, вы научитесь:

- форматировать объекты табличного документа различными способами;
- задавать разные форматы чисел;
- организовывать защиту данных;
- применять автоформатирование.

Форматирование табличных документов

Под форматированием табличного документа понимается ряд действий по изменению формы представления как самого документа, так и его объектов. Для форматирования объектов электронной таблицы помимо обычных способов, принятых в текстовом процессоре, используют свои собственные приемы:

- ◆ данные в ячейках могут быть представлены различным образом (в разных форматах);
- ◆ можно изменять ширину столбца или высоту строки, в которых хранятся данные;
- ◆ любой объект электронной таблицы может быть заключен в рамку или выделен специальным узором.

Форматирование любого объекта табличного документа выполняется с помощью команд раздела меню **Формат**. Различают форматы ячейки, строки и столбца.

Формат ячейки характеризуется следующими параметрами: *число, выравнивание, шрифт, граница, вид, защита*. Число определяет тип данных, хранящихся в ячейке, и формат представления этого числа. Выравнивание и шрифт используются так же, как и во всех других средах. Граница определяет внешнее обрамление ячейки (тип, толщину, штрих линии). Вид задает заливку и узор фона ячейки. Защита определяет уровень защиты данных в ячейке. В частности, можно защитить ячейку от изменения содержимого или скрыть формулы.

Формат строки позволяет регулировать высоту строки и управлять отображением строки в таблице. Высота строки определяется автоматически или вручную. При автоматическом определении высоты строки выбирается такое значение, чтобы все данные в строке были видны.

Формат столбца позволяет регулировать ширину столбца и управлять отображением столбца в таблице. Ширина столбца может регулироваться автоматически или вручную. При автоматической регулировке ширины столбца выбирается такое значение, чтобы все данные помещались в столбце в одну строку.

Отображение строк и столбцов. Любую строку и любой столбец в таблице можно скрыть. Это необходимо, когда строки и столбцы используются для записи промежуточных расчетов. Впоследствии скрытые строки и столбцы можно вновь вывести на экран.

Форматы данных

Для представления числовых данных можно применять различные форматы: общий, числовой (фиксированный), процентный, денежный, экспоненциальный и др.

- ◆ **Общий формат** используется по умолчанию и позволяет вводить любые данные (текст, числа, даты, время и т. д.), которые распознаются и форматируются автоматически.
- ◆ **Числовой (фиксированный) формат** обеспечивает представление чисел в ячейках с заданным пользователем количеством десятичных знаков. Например, если выбран формат, включающий три десятичных знака, то введенное в ячейку число 199 будет записано как 199,000, а число 0,12345 — как 0,123.
- ◆ **Процентный формат** обеспечивает представление числовых данных в форме процентов со знаком %. Например, если установлена точность в один десятичный знак, то при вводе числа 0,257 в ячейке будет отображено значение 25,7 %, а при вводе числа 257 — 25700,0 %.
- ◆ **Денежный формат** обеспечивает такое представление чисел, при котором каждые три разряда разделены пробелом, а после последнего десятичного знака указывается денежная единица размерности — р. (рублей). При этом пользователь может за-

дать определенную точность представления: с округлением до целого числа (0 десятичных знаков) или с заданным количеством десятичных знаков. Например, число 12345 будет записано в ячейке как 12 345 р. (при округлении до целого числа) и как 12 345,00 р. (при точности до двух десятичных знаков).

- ◆ *Экспоненциальный (научный) формат* обеспечивает представление вводимых чисел в виде двух частей:

- мантиссы, имеющей один разряд слева от десятичной запятой, и некоторого количества десятичных знаков справа от нее;
- порядка числа.

Например, число 12345 в экспоненциальном формате при заданной точности в два знака после запятой будет иметь вид 1,23E+04, где 1,23 — мантисса, а запись E+04 понимается как десятичный порядок и означает, что величина этого числа определяется как 1,23, умноженное на 10 в 4-й степени.

Изменение ширины столбца и высоты строки

Если длина вводимого числа превышает ширину ячейки (столбца), то вместо числа в ячейке выводятся знаки решетки (#####). При вводе текста, длина которого превышает ширину ячейку, на экране отображается только часть введенного текста в пределах установленной ширины ячейки. Количество символов, которое может быть отображено в ячейке, зависит не только от ее ширины, но и от выбранного типа шрифта (гарнитуры) и его размера (кегля).

Во всех перечисленных случаях возникает необходимость увеличить ширину столбца. Это можно сделать двумя способами:

- ◆ выбрать в меню команду **Формат** ▶ **Столбец** ▶ **Автоподбор ширины** или выбрать команду **Формат** ▶ **Столбец** ▶ **Ширина столбца** и в появившемся диалоговом окне задать нужное значение;
- ◆ установить курсор мыши на границу между именами столбцов (курсор изменит вид на ) и, удерживая нажатой левую кнопку мыши, передвинуть границу столбца в нужную сторону.

Аналогичным образом выполняются действия по изменению высоты строки.

Изменение внешнего вида объектов

В электронной таблице предусмотрены следующие возможности изменения внешнего вида объекта:

- ◆ отрисовка границ объекта;
- ◆ заливка фона объекта цветом и узором.

В табличных процессорах также предусмотрена возможность *автоматического форматирования* таблицы. Под автоматическим форматированием понимается изменение внешнего вида таблицы в соответствии с заданным образцом оформления (автоформатом).

Задание 4.6

Отформатируйте таблицу, сохраненную в файле Расписание.xls. Используйте возможности, предоставляемые средой табличного процессора:

- ◆ форматирование текста;
- ◆ изменение ширины и высоты столбца;
- ◆ задание формата чисел;
- ◆ задание типа выравнивания;
- ◆ отрисовку рамки и заливки;
- ◆ автоформатирование таблицы;
- ◆ организацию защиты данных.

В результате работы должна получиться таблица, представленная на рис. 4.11.

Железные дороги		Категория:		3	
Расписание движения поезда Бологое — Тверь					
Станция	Прибытие	Стоянок	Отправление	Время в пути	Цена
Бологое			7:45		
Ахдачинский	8:19	0:02	8:21	0:34	57,80р.
Вышний Волочек	8:48	0:05	8:53	0:27	45,90р.
Спирово	9:31	0:03	9:34	0:36	64,60р.
Калашниково	9:56	0:03	9:59	0:22	37,40р.
Лихославль	10:24	0:05	10:29	0:25	42,50р.
Наро-Фоминское	10:53	0:02	10:55	0:24	40,80р.
Тверь	11:12			0:17	28,90р.
Общее время					
стоянок:		0:20			
Общее время в					
пути:					
				3:27	317,90р.

Рис. 4.11. Таблица Расписание после форматирования (к заданиям 4.6 и 4.8)

Технология работы

1. Запустите табличный процессор.
2. Откройте документ *Расписание.xls*.
3. Вставьте две дополнительные строки для создания заголовка таблицы:
 - выделите строку 1;
 - выберите два раза команду меню Вставка ▶ Строки.
4. В ячейку A1 введите курсивом текст *Железные дороги*:
 - выделите ячейку A1;
 - нажмите кнопку Курсив на панели инструментов; 
 - наберите текст *Железные дороги*.
5. В ячейку A2 введите с клавиатуры текст *Расписание движения поезда Бологое — Тверь*:
 - выделите ячейку A2;
 - с помощью панели инструментов оформите ячейку шрифтом размера 12, полужирного начертания;
 - наберите текст;
 - выровняйте текст в ячейке по центру.
6. Отформатируйте заголовки столбцов расписания:
 - выделите строку 3;
 - выберите команду меню Формат ▶ Ячейки;
 - в появившемся диалоговом окне перейдите на вкладку Шрифт;
 - выберите полужирное начертание шрифта;
 - перейдите на вкладку Выравнивание;
 - выберите горизонтальное выравнивание По центру;
 - примените выбранные параметры, нажав кнопку ОК.
7. Задайте ширину каждого столбца так, чтобы она соответствовала длине заголовков (см. задание 4.1, п. 6).
8. Задайте высоту строки 2:
 - выделите строку 2;
 - выберите команду меню Формат ▶ Строка ▶ Высота;
 - в появившемся диалоговом окне задайте высоту строки — 22;
 - нажмите кнопку ОК.

9. **Задайте отрисовку границ и заливку для диапазона ячеек A2:F2, используя контекстное меню:**
- выделите диапазон ячеек A2:F2;
 - вызовите щелчком правой кнопки мыши контекстное меню;
 - выберите в контекстном меню команду **Формат ячеек**;
 - в появившемся диалоговом окне перейдите на вкладку **Граница**;
 - в поле **Тип линии** выберите образец с самой широкой линией;
 - выберите в качестве обрамления внешние границы ячеек;
 - перейдите на вкладку **Вид**;
 - выберите зеленый цвет заливки;
 - перейдите на вкладку **Шрифт**;
 - выберите белый цвет шрифта;
 - примените выбранные параметры, нажав кнопку **ОК**.
10. **Задайте автоформат таблицы:**
- выделите диапазон ячеек A3:F13;
 - выберите команду меню **Формат** ▶ **Автоформат**;
 - в появившемся диалоговом окне выберите подходящий тип автоформата;
 - нажмите кнопку **ОК**.
11. **Установите для ячеек A12 и A13 перенос текста по словам:**
- выделите диапазон ячеек A12:A13;
 - вызовите контекстное меню и выберите в нем команду **Формат ячеек**;
 - в появившемся диалоговом окне выберите вкладку **Выравнивание**;
 - установите флажок напротив параметра **переносить по словам**;
 - нажмите кнопку **ОК**.
12. **Подберите такую ширину столбца A, чтобы текст в ячейках строк 12 и 13 располагался в 2 строки.**
13. **Включите защиту таблицы:**
- выберите команду меню **Сервис** ▶ **Защита** ▶ **Защитить лист**;
 - в появившемся диалоговом окне установите флажок **Защитить лист и содержимое защищаемых ячеек**;

- в поле Разрешить всем пользователям этого листа сбросьте все флажки, кроме выделения незаблокированных ячеек и выделения заблокированных ячеек;
 - нажмите кнопку ОК.
14. Попробуйте ввести в ячейку D4 новое число — появится окно с сообщением о блокировке. Нажмите кнопку ОК.
15. Отключите защиту листа, выбрав команду меню Сервис ▸ Защита ▸ Снять защиту листа.
16. Сделайте ячейку D4 незащищаемой:
- выделите ячейку D4;
 - выберите команду меню Формат ▸ Ячейки;
 - перейдите на вкладку Защита;
 - сбросьте флажок Защищаемая ячейка;
 - щелкните на кнопке ОК.
17. Снова защитите лист, повторив действия, указанные в п. 13.
18. Попробуйте ввести в ячейку D4 другое время отправления, например 19:15, — предупреждение не появится, так как теперь эта ячейка является незащищенной.
19. Сохраните файл командой Файл ▸ Сохранить.
20. Закройте табличный процессор, выбрав команду меню Файл ▸ Выход.

Задания для самостоятельной работы

Задание 4.7. Отформатируйте созданную в предыдущей теме таблицу, применяя различные инструменты оформления.

Тема 4.4* Использование функций и логических формул

Выполнив задания этой темы, вы научитесь:

- использовать в формулах функции;
- создавать формулы с логическими функциями;
- изменять условия с логическими функциями.

Основные понятия и правила записи функции

В ячейки электронной таблицы можно вводить встроенные функции. Напомним, что с понятием функции тесно связано понятие аргумента, который надо задать, чтобы получить значение функции. Такими аргументами могут быть как конкретные данные, так и ссылки на них.

Функции в электронной таблице реализованы с помощью стандартных программ, которые можно вызывать по заранее известным именам. Количество различных функций, которые можно использовать в формулах, очень велико и может варьироваться в различных версиях табличных процессоров.

Все функции для удобства выбора и обращения к ним объединены по назначению в группы, называемые категориями: математические, статистические, текстовые, логические, финансовые, функции даты и времени и др. Использование любых функций в формулах происходит по одинаковым правилам:

- ◆ Каждая функция имеет свое уникальное имя.
- ◆ При обращении к функции после ее имени в круглых скобках указывается список аргументов, разделенных точкой с запятой.
- ◆ Ввод функции в ячейку начинается со знака равенства, а затем указывается ее имя.

Пример формулы, использующей функцию:

Для вычисления произведения сумм значений блоков ячеек A1:A9 и B7:C10 и сохранения результата в ячейке A10 можно использовать функцию СУММА (табл. 4.2). Для этого необходимо ввести в ячейку A10 формулу:

= СУММА(A1:A9)* СУММА(B7:C10)

Таблица 4.2. Некоторые функции табличного процессора Excel

Имя функции	Название и обозначение	Пример записи	Примечание
<i>Математические</i>			
SIN	Синус, $\sin(x)$	SIN(A1)	В ячейке A1 указывается угол в радианах
COS	Косинус, $\cos(x)$	COS(A1)	
TAN	Тангенс, $\tan(x)$	TAN(A1)	
ASIN	Арксинус, $\arcsin(x)$	ASIN(A2)	Значение ячейки A2 — число в диапазоне от -1 до 1
ACOS	Арккосинус, $\arccos(x)$	ACOS(A2)	
КОРЕНЬ	Квадратный корень, $\sqrt{x}$	КОРЕНЬ(A3)	Значение ячейки A3 — неотрицательное число
ABS	Модуль, $ x $	ABS(A4)	Значение ячейки A4 — любое число
LOG10	Десятичный логарифм, $\lg(x)$	LOG10(A5)	Значение ячейки A5 — положительное число
LN	Натуральный логарифм, $\ln(x)$	LN(A5)	
ПИ	Число π	ПИ()	Аргумент не указывается
<i>Статистические</i>			
МАКС	Максимум (список значений)	МАКС(A1:A10)	Возвращает максимальное из значений диапазона
МИН	Минимум (список значений)	МИН(A1:A10)	Возвращает минимальное из значений диапазона
СРЗНАЧ	Среднее значение (список значений)	СРЗНАЧ(A1:A10)	Возвращает среднее арифметическое значений диапазона

Имя функции	Название и обозначение	Пример записи	Примечание
СУММ	Сумма (список значений)	СУММ(A1:A10)	Возвращает сумму значений диапазона
<i>Логические</i>			
ЕСЛИ	Логическое условие: если (условие), то (выражение1), иначе (выражение2)	ЕСЛИ(A3<A4; A4-A3; A3-A4)	Если A3 < A4, функция возвращает значение A4 - A3; если нет, функция возвращает значение A3 - A4
И	Логическое И	И(A3<A4; B3<B4)	Возвращает значение ИСТИНА, если оба условия выполняются; иначе возвращает ЛОЖЬ
ИЛИ	Логическое ИЛИ	ИЛИ(A3<A4; B3<B4)	Возвращает значение ИСТИНА, если хотя бы одно условие выполняется; иначе возвращает ЛОЖЬ

Логические функции с простым условием

Остановимся более подробно на логических функциях, использование которых позволяет решать с помощью табличного процессора логические задачи. Для понимания и грамотного применения логических функций вспомним некоторые основные правила формальной логики.

В логической функции обязательно используется *условие*.

Условие представляет собой величины или выражения одного типа, связанные одним из знаков отношений: = (равно), <> (не равно), > (больше), < (меньше), >= (больше или равно), <= (меньше или равно).

Например: $A > 5$; $3 < 5$; $ax^2 + bx + c > 0$, и т. п.



Такие условия называются *простыми*. Они могут быть составлены из данных, ссылок и выражений.

Любое условие может принимать одно из двух возможных значений: ИСТИНА (логическая единица) или ЛОЖЬ (логический ноль).

Если условие составлено из констант (постоянных величин), то его значение не изменяется. Например, условие $3 < 5$ всегда имеет значение ИСТИНА. Если же в условие входят переменные величины, то оно может быть как истинным, так и ложным, в зависимости от значений переменных. Например, условие $A > 5$ будет иметь значение ИСТИНА, если $A = 7$, и значение ЛОЖЬ, если $A = 2$.

Для проверки условия в табличном процессоре используется функция ЕСЛИ. Функция ЕСЛИ имеет следующий формат записи:

ЕСЛИ(условие;выражение_если_истина;выражение_если_ложно)

Вычисляя значение этой функции при конкретных значениях, табличный процессор проверяет истинность записанного в ней логического условия и выводит значение выражения выражение_если_истина, если проверяемое условие истинно, или значение выражения выражение_если_ложно, если условие ложно.

Пример использования функции проверки простого условия:

Значение ячейки C10 зависит от соотношения значений ячеек A1 и A2:

- ◆ *если значение ячейки A1 больше значения ячейки A2, то в ячейке C10 надо записать число 1;*
- ◆ *если значение ячейки A1 меньше или равно значению ячейки A2, то в ячейке C10 надо записать число 0.*

Для выполнения этого задания надо ввести в ячейку C10 функцию

=ЕСЛИ(A>A2;1;0)

Логические функции со сложным условием

Сложным, или составным, условием называют несколько простых условий, связанных с помощью логических операций: И (логическое умножение), ИЛИ (логическое сложение) и др.

Рассмотрим некоторые жизненные ситуации, когда выбор вашего решения зависит от нескольких условий:

- ◆ Если вы стоите на перекрестке И горит зеленый свет светофора, то вам разрешено переходить улицу.
- ◆ Если у вас есть собака И вы ее выдрессировали, то любая ваша команда будет ей понятна.
- ◆ Если в субботу вечером вы пойдете с друзьями в театр ИЛИ в клуб, ИЛИ в гости, ИЛИ погулять, то вы отлично проведете время.



Функции И и ИЛИ имеют следующий формат записи:

И(условие1;условие2;...)

ИЛИ(условие1;условие2;...)

Для вычисления результата сложного условия необходимо знать, как определяется результат соответствующей логической операции (И, ИЛИ и др.). Ответ на этот вопрос дает *таблица истинности* (табл. 4.3). В этой таблице приняты обозначения: цифра 1 соответствует значению ИСТИНА, цифра 0 — значению ЛОЖЬ. В табл. 4.3 приведена таблица истинности для логических операций И и ИЛИ над двумя условиями, обозначенными как Условие1 и Условие2.

Таблица 4.3. Таблица истинности для логических операций И и ИЛИ

Условие1	Условие2	Условие1 И Условие2	Условие1 ИЛИ Условие2
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

Правила определения результата логических операций по таблице истинности:

- ☒ Результатом логической операции И будет ИСТИНА тогда и только тогда, когда истинны все перечисленные в аргументах условия (их количество не ограничено).
- ☒ Результатом логической операции ИЛИ будет ИСТИНА, если истинно хотя бы одно из перечисленных в аргументах условий.

Для выполнения указанных логических операций в электронной таблице как раз и предусмотрены логические функции И и ИЛИ, аргументами которых являются условия.

Для пояснения сказанного рассмотрим пример использования логической функции ЕСЛИ, в которую входит сложное логическое условие с функцией И.

Пример использования логических функций при проверке сложного условия:

Предположим, что электронная таблица используется для хранения и обработки среднего балла по школьному аттестату зрелости и оценок по результатам вступительных экзаменов в вуз (рис. 4.12). Пусть в ячейке A1 указан средний балл по аттестату, а в диапазоне A2:A5 хранятся экзаменационные оценки по четырем предметам, полученные на вступительных экзаменах. В ячейке A6 вычисляется сумма вступительных баллов ученика.

	A	B	C	D	E	F	G	H
1	4,6	16	Поздравляем, вы успешно сдали экзамены и приняты					
2	4		Сожалеем, но вы не прошли по конкурсу					
3	4							
4	4							
5	5							
6	17							
	Поздравляем, вы успешно сдали экзамены и приняты							

Рис. 4.12. К примеру использования логических функций

Требуется сравнить полученный суммарный балл с проходным баллом, который хранится в ячейке B1. В ячейке A7 должен быть выведен результат сравнения:

- ◆ если условие (суммарный балл $\geq$ проходному баллу И средний балл по аттестату > 4), то выводится текст из ячейки C1: «Поздравляем, вы успешно сдали экзамены и приняты»;
- ◆ если условие не выполнено, то выводится текст из ячейки C2: «Сожалеем, но вы не прошли по конкурсу».

Для решения поставленной задачи в ячейку A7 необходимо ввести логическую функцию ЕСЛИ следующего вида:

=ЕСЛИ(И(A6>=B1;A1>4);C1;C2)

Если оба простых условия (A6>=B1 и A1>4) истинны, то в соответствии с таблицей истинности будет истинно и проверяемое условие в функции ЕСЛИ, и в ячейке A7 будет выведен текст из ячейки C1. Иначе (то есть, если ложно хотя бы одно из условий, A6>=B1 или A1>4) в ячейке A7 появится текст из ячейки C2.

Задание 4.8*

В таблице Расписание.xls рассчитайте стоимость железнодорожных билетов для поездов разных категорий: пассажирские, скорые, фирменные. Цена билета зависит от времени движения поезда к пункту назначения и категории поезда. Каждой категории поезда соответствует определенный коэффициент, на который следует умножать время в пути (в минутах) для расчета цены билета: пассажирский — 0,5 р., скорый — 1,5 р., фирменный — 1,7 р. Организуйте таблицу Расписание.xls так, чтобы пересчет цены билета можно было бы осуществлять, изменяя значение ячейки Категория.

В процессе редактирования таблицы вы познакомитесь с основными технологическими операциями создания формул.

Технология работы

1. Запустите табличный процессор.
2. Откройте документ Расписание.xls.
3. Для задания категории поезда выполните следующие действия:
 - выберите ячейку D1 и введите текст Категория;
 - выберите ячейку E1 и введите первоначальное значение категории, равное 1.
4. Дополните таблицу столбцом Цена (см. рис. 4.11):
 - выберите ячейку F3;
 - введите текст Цена и нажмите клавишу Enter.

5. Задайте денежный формат чисел для диапазона ячеек F4:F11, используя контекстное меню:

- выделите диапазон ячеек F4:F11;
- вызовите контекстное меню и выберите в нем команду **Формат ячеек**;
- в появившемся диалоговом окне выберите вкладку **Число**;
- выберите в списке числовых форматов тип **Денежный**;
- если обозначение денежных единиц р. не будет выбрано автоматически, выберите его в списке **Обозначение**;
- в поле **Число десятичных знаков** укажите 2 знака;
- нажмите кнопку **ОК**.

6. Создайте в ячейке F5 логическую формулу для расчета стоимости билета.

Договоримся, что значение ячейки **Категория** для пассажирского поезда — 1, для скорого — 2, для фирменного — 3. Структура формулы должна быть следующей:

=ЕСЛИ(\$E\$1=1;МИНУТЫ(E5)*0,5;ЕСЛИ(\$E\$1=2;МИНУТЫ(E5)*1,5;
ЕСЛИ(\$E\$1=3;МИНУТЫ(E5)*1,7;"Неверное значение категории"))

Рассмотрим последовательность действий по созданию этой формулы:

- выделите ячейку F5;
- выберите команду меню **Вставка** ▸ **Функция**;
- в открывшемся диалоговом окне выберите категорию — **Логические**, функцию — **ЕСЛИ** и нажмите кнопку **ОК**;
- в окне ввода аргументов функции в поле **Лог_выражение** наберите **\$E\$1=1**: щелкните мышью в ячейке E1, нажмите клавишу F4, затем введите **=1**;
- в поле **Значение_если_истина** введите **МИНУТЫ(E5)*0,5**;
- нажмите кнопку **ОК**;
- скопируйте полученную формулу **ЕСЛИ(\$E\$1=1;МИНУТЫ(E5)*0,5)**, добавьте в нее перед последней скобкой точку с запятой и вставьте после нее скопированную формулу, заменив значение категории на 2, а коэффициент — на 1,5;

- добавьте после числа 1,5 еще одну точку с запятой и снова вставьте скопированную формулу, заменив значение категории на 3, а коэффициент — на 1,7;
 - добавьте после числа 1,7 точку с запятой и вставьте текст "Неверное значение категории";
 - нажмите клавишу Enter;
 - сверьте структуру формулы с образцом.
7. Скопируйте формулу из ячейки F5 в каждую ячейку диапазона F6:F11, используя раскрывающееся меню кнопки Вставить на панели инструментов:
- выделите ячейку F5, откуда надо скопировать формулу;
 - щелкните на кнопке Копировать на панели инструментов;
 - выделите диапазон ячеек F6:F11;
 - щелкните на стрелке справа от кнопки Вставить и выберите в открывшемся меню команду Формулы.



При копировании формулы абсолютная ссылка не изменяется.

8. Изменяя значение категории поезда в ячейке E1, проанализируйте результаты.
9. Измените ширину столбца F так, чтобы его ширина точно соответствовала длине заголовка.



Если данные не помещаются в ячейке, они заменяются знаками #####.

10. Измените ширину столбца F так, чтобы числовые данные были отображены полностью.
11. Сохраните документ и закройте табличный процессор командой Файл ► Выход.

Контрольные вопросы

1. Какие логические функции вам известны?
2. Что такое условие в логической функции и по каким правилам оно записывается? Приведите примеры.

3. В чем отличие сложного условия от простого? Приведите примеры.
4. Расскажите о сути логической операции И. Приведите примеры записи логической функции И.
5. Расскажите о сути логической операции ИЛИ. Приведите примеры записи логической функции ИЛИ.
6. Объясните, что такое таблица истинности. Приведите пример.
7. В чем отличие функции от формулы? Приведите примеры записи функции и формулы.

Задания для самостоятельной работы

Задание 4.9. Дополните созданные вами самостоятельно таблицы одним или несколькими столбцами, используя в них логические и прочие функции.

Тема 4.5 **Представление данных в виде диаграмм**

Выполнив задания этой темы, вы научитесь:

- ⇒ создавать различные типы диаграмм;
- ⇒ форматировать объекты диаграммы;
- ⇒ изменять размеры диаграммы.

Создание и редактирование диаграмм

Для создания диаграммы необходимо:

- ◆ выделить диапазон ячеек, содержащий данные для ее построения;
- ◆ нажать кнопку Мастер диаграмм на панели инструментов;
- ◆ выбрать тип диаграммы из предлагаемого набора;
- ◆ задать основные и дополнительные параметры диаграммы.

Редактирование диаграммы производится с помощью команд контекстного меню, позволяющих изменять ее параметры: заголовки, легенду, подписи рядов и данных. Можно добавлять новые данные для построения диаграммы или удалить построенные ранее диаграммы.

Форматирование диаграмм

Форматирование диаграммы производится с помощью команд меню Формат, которые позволяют:

- ◆ изменить тип диаграммы;
- ◆ изменить используемые шрифты;
- ◆ изменить цвет и узор объектов диаграммы;
- ◆ задать масштаб и формат вертикальной и горизонтальной осей;
- ◆ выбрать вид рамки, ограничивающей диаграмму.

Задание 4.10

*Используя данные таблицы **Расписание.xls**, создайте столбчатую диаграмму распределения времени стоянок по станциям и круговую объемную диаграмму распределения времени в пути. Отредактируйте диаграммы.*

В результате выполнения задания вы освоите основные технологические операции создания и редактирования диаграмм.

Технология работы

1. Запустите табличный процессор.
2. Откройте документ **Расписание.xls**.
3. Создайте столбчатую диаграмму распределения времени стоянок по станциям:
 - выделите диапазон ячеек **C5:C10**;
 - выберите команду меню **Вставка** ► **Диаграмма**;
 - в появившемся окне Мастера диаграмм выберите тип диаграммы — **Гистограмма**, вид — **объемный**;
 - щелкните на кнопке **Далее**.
4. На следующем шаге Мастера диаграмм задайте подписи по оси **X**:
 - на вкладке **Диапазон данных** установите переключатель **Ряды в строках**;
 - перейдите на вкладку **Ряд**;
 - щелкните в поле **Подписи по оси X** и выделите в таблице диапазон ячеек **A5:A10**;
 - задайте подпись ряда, щелкнув в поле **Имя** и выделив в таблице ячейку **C3**;
 - щелкните на кнопке **Далее**.
5. На следующем шаге Мастера диаграмм задайте подписи осей **X** и **Y**:
 - перейдите на вкладку **Заголовки**;
 - в поле **Название диаграммы** введите *Распределение стоянок*;
 - в поле **Ось X (категорий)** введите *Станция*;
 - в поле **Ось Y (значений)** введите *Время*;

- перейдите на вкладку **Легенда**;
 - сбросьте флажок **Добавить легенду**;
 - щелкните на кнопке **Далее**.
6. На следующем шаге Мастера диаграмм задайте расположение диаграммы, выбрав переключатель **Поместить диаграмму на отдельном листе**, и щелкните на кнопке **Готово**.
7. Измените шрифт заголовка диаграммы:
- щелкните мышью на заголовке диаграммы — вокруг текста появится рамка с маркерами выделения;
 - щелкните на заголовке правой кнопкой мыши и выберите в контекстном меню команду **Формат заголовка диаграммы**;
 - перейдите на вкладку **Шрифт**;
 - в появившемся диалоговом окне выберите полужирное начертание шрифта, размер — 16, цвет — синий;
 - щелкните на кнопке **ОК**.
8. Создайте диаграмму распределения времени в пути:
- выделите диапазон ячеек **E5:E11**;
 - выберите команду меню **Вставка ► Диаграмма**;
 - в появившемся окне Мастера диаграмм выберите тип диаграммы — **Круговая**, вид — **объемный**;
 - щелкните на кнопке **Далее**.
9. Задайте подписи по оси **X**:
- на вкладке **Диапазон данных** установите переключатель **Ряды в столбцах**;
 - перейдите на вкладку **Ряд**;
 - щелкните в поле **Подписи по оси X** и выделите в таблице диапазон ячеек **A5:A11**;
 - задайте подпись ряда, щелкнув в поле **Имя** и выделив в таблице ячейку **E3**;
 - щелкните на кнопке **Далее**.
10. На следующем шаге мастера задайте подписи данных на диаграмме:
- выберите вкладку **Подписи данных**;

- установите флажок Включить в подписи — значения;
 - перейдите на вкладку Легенда;
 - установите флажок Добавить легенду, выберите переключатель Размещение — справа;
 - щелкните на кнопке Далее.
11. Задайте расположение диаграммы, выбрав переключатель Поместить диаграмму на отдельном листе, и щелкните на кнопке Готово.
12. Сохраните документ и закройте табличный процессор командой Файл ► Выход.

Задания для самостоятельной работы

Задание 4.11. Представьте данные созданной в предыдущей теме таблицы в виде диаграмм. Используйте несколько типов диаграмм. Задайте им различное оформление.

Задание 4.12. Создайте таблицу успеваемости нескольких учеников по разным предметам. Отобразите эти данные на нескольких диаграммах. Используйте различные возможности оформления диаграмм.

РАЗДЕЛ 5 Освоение системы управления базой данных

Тема 5.1	Общая характеристика системы управления базой данных	172
Тема 5.2	Создание структуры базы данных и заполнение ее данными	187
Тема 5.3	Создание формы базы данных	193
Тема 5.4	Работа с записями базы данных	197
Тема 5.5	Разработка отчета для вывода данных . . .	203



Практические задания ориентированы на выполнение в среде системы управления базой данных (СУБД) Access.

Тема 5.1 **Общая характеристика системы управления базой данных**

Изучив эту тему, вы узнаете:

- ➡ что такое база данных;
- ➡ что такое поле и запись;
- ➡ каково назначение инструментов базы данных;
- ➡ какие существуют возможности обработки данных.

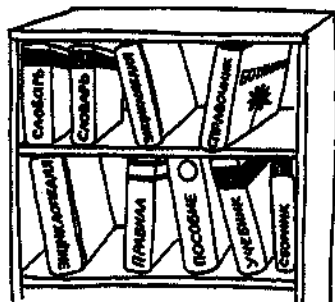
Назначение базы данных

Человеку свойственно систематизировать информацию обо всем, что его окружает: о растениях, животных, химических элементах, механизмах и т. п. Созданы многотомные справочники по самым разным областям человеческого знания, и каждый может воспользоваться этой информацией. Вас интересует журавль? Вы хотите узнать о нем больше? Пожалуйста! Возьмите справочник по орнитологии. В нем хранятся все имеющиеся сведения об интересующих вас птицах, а также обо всех остальных объектах этого класса.

Любой подобный справочник представляет собой хранилище информации. Допустим, что вам понадобилась информация о редком цветке. Вы должны найти специализированную библиотеку. А когда вы ее найдете, то окажетесь лицом к лицу с тысячами книг. И где-то среди них, может быть, отыщется нужная вам.

Конечно, существует система библиотечных каталогов, которая значительно облегчает поиск нужной информации. В конце концов вы найдете нужную книгу, но как это будет просто! А если поблизости нет библиотеки?

Все эти проблемы можно решить, используя компьютерные технологии обработки информации. Вы уже знаете, что существуют специальные програм-



мы, позволяющие готовить документы и книги, производить сложные вычисления, конструировать графические объекты и т. п. Для хранения и обработки большого количества информации люди тоже придумали специальные программы, называемые *системами управления базами данных*.

Но прежде чем вы начнете знакомиться с ними, надо понять смысл, который вкладывается в понятие *база данных*. Благодаря широкому использованию компьютеров оно настолько прочно вошло в наш лексикон, что многие, понимая его на интуитивном уровне, не задумываются над тем, что стоит за этим понятием, и связывают его исключительно с компьютером. Однако базы данных, хотя они и не назывались так раньше, существовали и до появления компьютеров в библиотеках, архивах, фондах, справочных бюро и других подобных организациях — в виде книг, справочников, брошюр, газет, журналов и т. п.

Под *базой данных* следует понимать совокупность специально организованных сведений, которые хранятся на каком-либо материальном носителе, например бумаге, магнитном или оптическом носителе, киноленте. Обращаем ваше внимание на то, что это не просто набор разрозненных данных. Они обязательно должны быть связаны между собой так, чтобы человек мог составить представление о каком-либо объекте, явлении, процессе.

Например, в базе данных, хранящей кулинарные рецепты, должна находиться информация о различных пищевых продуктах, мерах веса, способах приготовления пищи и т. д. Но никому не придет в голову помещать туда же данные о стоимости сельскохозяйственных машин. Другой пример: словарь терминов по компьютерной технике содержит разъяснения понятий только из этой области. Вы не найдете в нем сведений, разъясняющих смысл, например, термина из экономики или кулинарии.

Сведения, находящиеся в одной базе данных, должны быть связаны между собой по смыслу, логически. Это достигается путем группировки, объединения их по определенным параметрам. Такой процесс получил название *структурирования данных*. Следует заметить, что в результате структурирования данных должна появиться база данных в компьютерном или некомпьютерном варианте, что определяется тем, был ли при этом использован компьютер.



База данных — это совокупность структурированных данных определенного назначения.

Структурирование данных — это объединение данных по определенным параметрам.

Объекты базы данных

Любая база данных, создаваемая в компьютере, состоит из простейших объектов, называемых *полем* и *записью* (рис. 5.1). Совокупность полей и записей составляет более крупный объект — *таблицу*. С помощью этих компьютерных объектов можно составить описание (*модель*) любого реально существующего объекта или процесса.

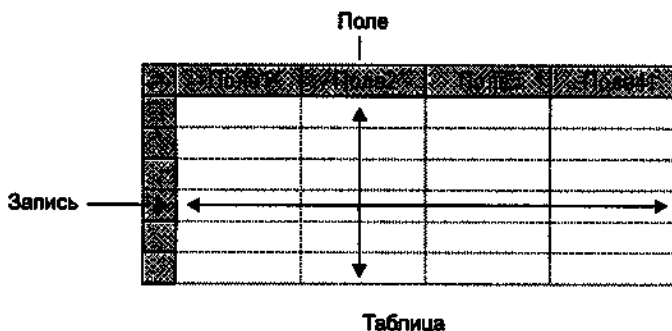


Рис. 5.1. Объекты базы данных

Поле

Любое поле предназначено для хранения множества значений одного параметра, характеризующего реальный объект. Каждое новое значение служит отличительным признаком описываемого объекта по сравнению с другими. Например, если мы рассматриваем человека, то поле *Цвет глаз* может содержать значение голубой, серый, зеленый, карий и т. п. Если же поле описывает такой параметр, как вес человека, то оно будет содержать числовые значения: 45, 57, 60 и т. п.

Одно поле не может служить хранилищем значений нескольких разнотипных параметров. Представьте себе, что вы создали базу данных по своему классу и описали внешний вид каждого

ученика всего в одном поле. При этом вы поместили туда данные и о росте, и о весе, и о цвете глаз, и о форме носа. В каждом поле у вас получится маленький рассказ о внешности ученика. Казалось бы, ничего страшного, ведь вся информация учтена и занесена в поле. Да вот беда: вы не сможете извлечь пользу из информации, которая записана таким образом. Например, как узнать, сколько голубоглазых детей учится в вашем классе или кто в классе самый высокий?



Поэтому на практике для каждого из параметров описываемого объекта отводят отдельное поле. При этом надо стремиться, чтобы информация, помещаемая в поле, была минимальной.

Так, например, для описания учащегося в классном журнале используется параметр «Адрес». Соответственно, в базе данных можно организовать поле — Адрес. Это поле будет содержать полные адреса всех учеников класса. Но обработку данных можно будет производить значительно эффективнее, если разбить это поле на несколько более мелких полей: Город, Улица, Дом, Квартира. Такое представление параметра «Адрес» в виде нескольких полей в базе данных позволит вам найти, например, всех учеников, живущих на одной улице или в одном доме.



Поле — это простейший объект базы данных, предназначенный для хранения значений параметра реального объекта или процесса.

Поле может характеризоваться следующими параметрами: *имя, формат данных, тип оформления.*

- ◆ **Имя поля.** Как и любой другой объект, поле при создании получает свое собственное имя. Имя поля состоит из последовательности символов — как правило, из латинских и русских букв и цифр. Именовать поля принято в соответствии с предполагаемым назначением. Например, если в поле будет храниться информация о возможных значениях цвета объекта, то разумно дать полю имя Цвет.
- ◆ **Формат данных.** Этот параметр поля служит для задания типа данных и характеризуется тем же набором значений,

что и в электронной таблице. Дополнительно используется формат *счетчик* (серия). Обычно поля такого формата заполняются автоматически.

- ◆ **Тип оформления.** Определяет внешнее представление данных в поле указанного формата. Например, вам нужно создать поле, в котором хранятся даты рождения учеников. При создании поля введите его имя: Дата рождения. Выберите тип данных Дата. В разделе Оформление (Формат поля) выберите наиболее удобный для вас способ записи, например 30.09.2006. Здесь первые две цифры означают день, вторые две — месяц, а последние четыре — год.

Запись

Вероятно, вы уже догадались, что одного поля недостаточно для того, чтобы описать какой-либо реально существующий объект. Все зависит от сложности выбранного для описания объекта. Иногда бывает достаточно небольшого количества полей, как,



например, для описания воздушного шара. Здесь вполне можно обойтись полями Название, Цвет, Объем, Форма. В поле Название мы запишем название конкретного экземпляра шара, например Шар1. В поле Цвет укажем цвет этого шара, например Синий или Красный; в поле Объем —

количество литров воздуха, которое может поместиться в шаре, например 7,5 или 3; в поле Форма — форму шара, например Круглый или Сердце. При описании сложных объектов, таких как, например, человек или растение, может потребоваться значительно большее количество полей.

Для того чтобы поля отражали информацию об одном и том же объекте, необходимо каким-то образом связать их между собой. Так в базе данных возникло понятие *записи*.



Запись — это совокупность логически связанных полей, характеризующих свойства описываемого объекта или класса объектов.

Значения записи отражают свойства конкретного объекта. Представьте себе, что имеется следующий набор полей: Имя,

Название, Фамилия, Форма листьев, Цвет лепестков, Успеваемость. Трудно представить себе реальный объект, для описания параметров которого нужны были бы все перечисленные поля. Но можно, исходя из смысла названий, разделить эти поля на две записи.

Давайте порассуждаем. Если в первую запись включить поле Имя, то очевидно, что поле Название нужно включить во вторую запись. Поле Фамилия ближе по смыслу к полю Имя, чем к полю Название. Включаем его в первую запись. Далее, объект, имеющий имя и фамилию, определенно не должен иметь ни листьев, ни лепестков. Поэтому поля Форма листьев и Цвет лепестков мы включим во вторую запись. А вот поле Успеваемость явно имеет отношение к имени и фамилии. Его включаем в первую запись.



Разобрав таким образом все имеющиеся поля, мы получили две записи (рис. 5.2).

	Поле 1	Поле 2	Поле 3
Запись 1	Имя	Фамилия	Успеваемость
Запись 2	Название	Форма листьев	Цвет лепестков

Рис. 5.2. Две записи, сформированные из предложенного набора полей

Запись характеризуется количеством и типами включенных в нее полей.

В базе данных обычно описывается не один объект, а несколько однотипных объектов. В рассмотренном примере с воздушными шарами в нашей базе могут быть описаны и красные, и синие, и большие, и маленькие шары. Поэтому в базе данных будет не одна, а несколько записей указанной структуры. Запись для каждого конкретного шара называется *экземпляром записи*. Экземпляров записи в базе данных может быть столько, сколько объектов данного вида мы описываем. Для простоты изложения в дальнейшем мы будем называть записью любой экземпляр записи.

Таблица

Совокупность экземпляров записи одной структуры называют *таблицей*. Таблица является документом базы данных и характеризуется следующими параметрами: *имя, структура записи, количество записей*.

- ◆ **Имя.** Как и любой компьютерный объект, таблица имеет собственное имя. Под этим именем она хранится в памяти компьютера.
- ◆ **Структура записи.** Этот параметр определяется количеством полей, которые соответствуют параметрам объекта. Каждое поле имеет свои формат и тип данных.
- ◆ **Количество записей.** Данный параметр указывает на то, сколько экземпляров записи содержит в себе данная таблица.

Структура базы данных

Прежде чем строить базу данных, необходимо определить, какие же параметры исследуемого объекта являются наиболее важными. Например, при построении модели библиотечного каталога к важнейшим параметрам можно отнести название книги, автора, издательство, тематику, а такими параметрами, как формат, толщина и вес книги, плотность и качество бумаги, можно пренебречь.

После того как вы отобрали интересующие вас параметры, характеризующие свойства объекта, для каждого из них создается соответствующее поле и вводится его значение. Этот процесс получил название *создание структуры базы данных*.

Создание структуры базы данных включает следующие этапы:

- ◆ определение параметров, характеризующих свойства объекта, которые будут служить полями базы данных;
- ◆ назначение каждому полю уникального имени;
- ◆ определение для каждого поля формата представления данных, то есть типа данных и необходимых ограничений.



Структура базы данных — это набор поименованных полей, описывающих свойства объектов одного класса.

Первым шагом при создании базы данных является определение ее структуры в виде набора полей. Вся последующая работа

с базой данных будет зависеть от того, насколько тщательно спланирована структура записи. В базе данных работа ведется с полями и записями, а не с отдельными ячейками, как в табличном процессоре. Структуру базы данных можно неоднократно изменять, добавляя и удаляя поля и изменяя их параметры. Но следует помнить, что необдуманные операции над заполненной базой данных могут привести к невозможным потерям информации.

Инструменты системы управления базой данных

Для организации работы с базами данных создан специальный комплекс программ, получивший название *системы управления базами данных (СУБД)*. СУБД предоставляет пользователю различные возможности поиска, сортировки, модификации и редактирования данных.

К основным инструментам СУБД можно отнести:

- ◆ инструменты для работы с записями и полями;
- ◆ инструменты управления отображением базы данных;
- ◆ инструменты обработки данных;
- ◆ инструменты вывода данных.

Инструменты для работы с записями и полями

К инструментам для работы с записями и полями относятся: *сортировка записей, вывод записей на экран, фильтр*.

- ◆ **Сортировка записей.** Позволяет представить записи в упорядоченном виде.
- ◆ **Вывод записей на экран.** При помощи этого инструмента можно скрыть часть записей. Это делается для удобства работы с определенным набором записей. Используя этот же инструмент, можно вывести на монитор ранее скрытые записи.
- ◆ **Фильтр.** Позволяет производить отбор данных по одному из имеющихся шаблонов. Фильтр по своим функциям можно сравнить с ситом. Просеивая через сито муку, мы отбираем только те частички, которые проходят сквозь ячейки сита. Применяя к таблице фильтр, мы выделяем только те записи, которые удовлетворяют заданному условию.

Инструменты управления отображением базы данных

База данных может быть представлена в двух режимах: как таблица и как форма.

- ♦ **Режим таблицы.** В этом режиме информация из базы данных выводится на экран в виде строк и столбцов и напоминает электронную таблицу (рис. 5.3). Вы можете видеть одновременно все записи, что облегчает их редактирование.

Название	Цвет	Объем	Форма
Шар 1	Синий	7,5	Круглый
Шар 2	Красный	3	Сердце
Шар 3	Голубой	5	Длинный
Шар 4	Зеленый	6	Круглый

Рис. 5.3. База данных в режиме таблицы

- ♦ **Режим формы.** В режиме формы вы будете видеть перед собой только одну запись, реализованную наглядно (рис. 5.4). При этом в нижней части окна формы находится панель с кнопками перехода по записям, позволяющая выбирать первую, последнюю, следующую или предыдущую записи.

The screenshot shows a form window with a dark background. On the left, there are labels for the fields: 'Название' (Name), 'Цвет' (Color), 'Объем' (Volume), and 'Форма' (Shape). On the right, there are corresponding input fields containing the values: 'Шар 1', 'Синий', '7,5', and 'Круглый'. At the bottom of the form, there is a navigation bar with several buttons and a status indicator showing '1'.

Рис. 5.4. База данных в режиме формы

Кроме переключения режимов эта группа включает в себя также инструмент для разработки дизайна формы. С его помощью можно изменить порядок перехода между полями, размер поля, его цвет. Можно вставить готовую картинку и нарисовать рисунок.

Инструменты обработки данных

База данных, как мы уже говорили, является не только хранилищем информации, но и мощным средством их обработки. Представьте себе, что вы занесли в компьютер информацию обо

всех растений на земном шаре. Вы описали все: название, цвет, форму, размер и т. д. У вас получился огромный список. Но как найти в этом списке самый большой цветок или, скажем, самое высокое дерево? Как узнать количество цветов, начинающихся с буквы «К»? Ответы на эти и другие вопросы станут для вас очевидными, если вы познакомитесь с возможностями обработки информации в базе данных.

База данных располагает следующими инструментами обработки данных: *поиск/замена, сортировка, группировка, фильтрация*.

- ◆ **Поиск/замена.** Позволяет находить в базе данных информацию согласно образцу, указанному пользователем. Например, вам надо найти информацию о цветке, названия которого вы не знаете, но о котором вам точно известно, что он желтого цвета. Тогда, выполнив команду основного меню Правка ▸ Найти, вы вводите образец для поиска — слово Желтый (рис. 5.5). Результатом поиска будет таблица, в которой окажутся все записи, содержащие слово Желтый хотя бы в одном из полей.

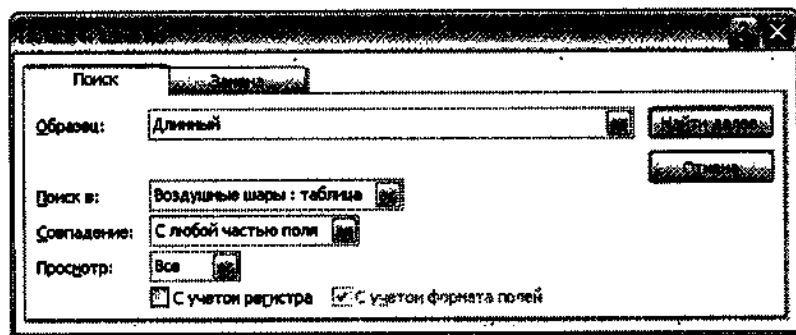


Рис. 5.5. Поиск значения по образцу

Теперь представьте себе, что при заполнении базы данных вы ошибочно указали место произрастания целого ряда растений. Вместо *Южная Америка* написали *Северная Америка*. Тогда вы вводите образец для поиска — Северная Америка, и образец для замены — Южная Америка (рис. 5.6). Далее вы можете использовать одну из трех имеющихся возможностей: продолжить поиск образца, заменить найденное значение, заменить все значения.

Если вы щелкнете на кнопке Продолжить, то перейдете к следующей записи, содержащей искомое значение, без изменения найденного значения. Если вы щелкнете на кнопке Заменить, то будет произведена замена только в текущей записи базы данных. Если вы выберете вариант Заменить все, то замена будет произведена сразу во всех записях базы данных.

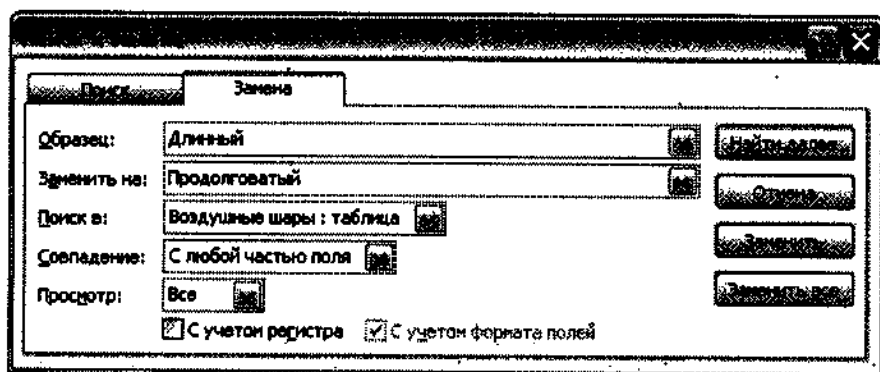


Рис. 5.6. Поиск и замена значения

- ◆ **Сортировка.** С помощью сортировки можно переставлять записи в нужном порядке. Например, вы хотите знать, у какого цветка стебель длиннее. Для этого записи сортируются по полю, которое описывает высоту растения, например Высота. При этом сортировку можно осуществлять в двух направлениях: *в порядке возрастания*, когда за меньшим значением следует большее (1, 2, 3, 4... или а, б, в, г...), или *в порядке убывания*, когда за большим значением следует меньшее (100, 99, 98... или я, ю, э...).

Сортировку можно выполнять одновременно по нескольким полям. Например, вам надо найти самый высокий цветок желтого цвета, название которого начинается на «Р». Для этого вам придется отсортировать три поля. Сначала переставим все записи в алфавитном порядке. Для этого отсортируем поле Название по возрастанию, то есть в алфавитном порядке. В качестве второго поля для сортировки укажем поле Цвет и тоже отсортируем его по возрастанию. Теперь отсортируем

поле Высота в порядке убывания. В результате проведенной сортировки все записи будут переставлены следующим образом:

- все записи будут выстроены по алфавиту;
 - в пределах каждой буквы записи будут отсортированы по цвету;
 - внутри каждого цвета записи будут отсортированы по высоте.
- ◆ **Группировка данных.** Группировка неразрывно связана с сортировкой. Дело в том, что при составлении отчетов имеется возможность организовывать отсортированные записи в группы двумя способами. Можно организовать группы только по первой букве. Например, вы отсортировали названия цветов по алфавиту. Укажите способ группировки только по первой букве. Тогда при формировании отчета сначала отдельной группой будут идти все цветы на букву «А», затем на букву «Б» и т. д.

Второй способ заключается в организации групп по изменению содержимого поля в целом. Например, вы отсортировали базу данных по полю Высота и указали способ группировки по изменению содержимого. Тогда все записи будут объединены в группы по мере того, как будет изменяться значение поля Высота.

- ◆ **Фильтрация.** Позволяет организовывать «сито», через которое можно «просеивать» записи базы данных. Фильтрация позволяет выбирать из всего множества записей только те, которые отвечают заданным условиям (рис. 5.7).

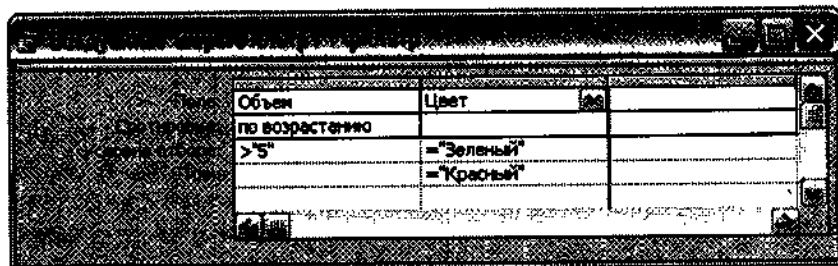


Рис. 5.7. Окно настройки фильтра

Параметры фильтрации, как и сортировки, задаются пользователем. Каждому новому фильтру присваивается собственное имя. После этого нужно указать тип фильтра. Существуют два типа фильтров: *простой фильтр* и *фильтр с формулой*.

По умолчанию предлагается создать простой фильтр, задав условия фильтрации. Делается это следующим образом:

- 1) сперва выбирается поле, для которого задается условие;
- 2) затем назначают способ сравнения (равно, больше, меньше и т. д.);
- 3) после этого указывают образец для сравнения.

При организации фильтра можно объединять несколько условий при помощи логических функций. Обычно используются две логические функции: логическое И и логическое ИЛИ.

Функция И объединяет два условия. Для того чтобы запись была отобрана фильтром, необходимо, чтобы она отвечала обоим условиям одновременно.

Например, нужно отфильтровать все записи, соответствующие желтым цветам высотой более двух метров. Сформируем первое условие:

- 1) в качестве поля для первого условия выберем поле Цвет;
- 2) в качестве способа сравнения укажем равно;
- 3) в качестве образца для сравнения введем слово Желтый.

Сформируем второе условие:

- 1) выберем поле Высота;
- 2) в качестве способа сравнения укажем больше;
- 3) для сравнения введем число 200.

Теперь объединим эти два условия, выбрав логическую функцию И.

Если в качестве логической функции выбрать ИЛИ, то фильтр отберет все записи, отвечающие первому условию, и все записи, отвечающие второму условию. В результате такой фильтрации в таблице будут отображены записи обо всех желтых цветах любой высоты и все записи о цветах высотой более двух метров любого цвета.

- ◆ **Вычисляемые поля.** Большие возможности для обработки данных открывают вычисляемые поля. В качестве значений в такие поля вводят формулы. В формулах могут содержаться числа, имена полей или функции. Любое вычисляемое поле должно начинаться со знака равенства. Например, вы работаете с базой данных кулинарных рецептов. В рецепте содержатся данные о расходе продукта на приготовление одной порции. Вам нужно приготовить пять порций. Тогда в качестве значения дополнительного поля вы вставляете формулу:

$$= \text{Расход} \times 5,$$

где Расход — имя поля, содержащего данные о расходе продукта на приготовление одной порции, 5 — число порций. Для каждой конкретной записи значение будет вычисляться по общей формуле.

Инструменты вывода данных

Результаты обработки информации, содержащейся в базе данных, могут быть выведены в виде *отчета*.

Под отчетом понимают специальную форму для вывода информации из базы данных. Форма может быть создана пользователем или выбрана из предлагаемого системой типового набора.

Действия по созданию отчета можно разбить на несколько этапов: *задание общих параметров оформления, включение полей, сортировка, группировка, установка фильтра, определение итогов*.

- ◆ **Задание общих параметров оформления.** На данном этапе определяются название отчета, его ориентация на странице и параметры шрифта.
- ◆ **Включение полей.** На этом этапе вы определяете, какие поля записи войдут в отчет. Чтобы добавить поле в отчет, надо выбрать его в списке доступных полей. Можно указать, требуется ли выводить имена полей на каждой странице, а также показывать ли итоговую информацию.
- ◆ **Определение итогов.** На этом этапе осуществляется итоговая обработка информации, приведенной в отчете.

О том, как выполняется сортировка, группировка и фильтрация, было рассказано ранее.

Для каждого из полей отчета вы можете задать способ обработки итогов. Например, если в отчете присутствует поле Вес и указан способ обработки Сумма, то вы узнаете суммарный вес по всем записям. Вы можете указать одновременно несколько способов итоговой обработки. На этом этапе также определяется порядок вывода итоговой информации в отчете. Вы можете разместить ее в конце каждой группы или в конце отчета, выводить ее под каждым столбцом или вместе со строками.

Контрольные вопросы

1. Для чего предназначена база данных?
2. Приведите определение базы данных и поясните его смысл. Приведите примеры баз данных.
3. Из каких объектов состоит база данных?
4. Дайте определение поля и приведите примеры полей.
5. Какие форматы данных существуют для объекта «поле»?
6. Что такое запись?
7. Как вы понимаете структуру базы данных?
8. Что такое СУБД и в чем состоят ее функции?
9. Что такое инструменты СУБД?
10. Какие группы инструментов СУБД вы знаете?
11. Чем отличается режим таблицы от режима формы?
12. Какие существуют возможности обработки данных?
13. Для чего используется группировка?
14. В каких направлениях может осуществляться сортировка?
15. Каков порядок построения простого фильтра?
16. Когда используется логическая функция ИЛИ, а когда — И?
17. В чем отличие фильтрации от сортировки?
18. В каких случаях нужен фильтр с формулой?
19. Для чего предназначен отчет?

Тема 5.2 Создание структуры и заполнение базы данных

Выполнив задания этой темы, вы научитесь:

- ⇒ работать с объектами базы данных;
- ⇒ создавать структуру базы данных;
- ⇒ редактировать полученную структуру;
- ⇒ изменять параметры объектов базы данных.

Задание 5.1

Создайте структуру базы данных, отражающую основные антропометрические сведения о каждом ученике вашего класса по следующим параметрам: фамилия, имя, количество полных лет, средний балл успеваемости, рост, вес, увлечения. Определите, какие из этих параметров важны при заполнении медицинской карты учащегося. Отредактируйте полученную структуру базы данных, удалив из нее ненужные поля.

Введите в структуру базы данных поля Номер и Пол. Отредактируйте существующее поле Полных лет, изменив имя поля и формат данных. Сохраните базу данных в файле Класс.mdb.

Технология работы

1. Запустите СУБД.
2. Создайте новую базу данных:
 - выберите команду меню Файл ► Создать;
 - в области задач Создание файла в правой части окна выберите щелчком мыши пункт Новая база данных;
 - в открывшемся окне выберите папку, в которой надо сохранить новую базу, и введите ее название — Класс.mdb.
3. В левой панели окна Класс: база данных выберите закладку Таблица, в правой панели выделите пункт Создание таблицы в режиме конструктора и нажмите кнопку Открыть на панели инструментов — откроется новое окно Таблица1.

4. Добавьте поля в таблицу в соответствии с табл. 5.1:

- в столбце Имя поля введите названия полей;
- в столбце Тип данных укажите соответствующий каждому полю тип данных (в дальнейшем — формат);
- в столбце Описание при желании можно ввести описание сведений, хранящихся в этом поле; текст описания будет показан в строке состояния при добавлении данных в поле, а также будет включен в описание объекта таблицы.

Таблица 5.1. Описание полей таблицы Карта

Имя поля	Тип данных	Свойства
Фамилия	Текстовый	—
Полных лет	Числовой	Формат — основной, число десятичных знаков — 0
Рост	Числовой	Формат — основной, число десятичных знаков — 0
Вес	Числовой	Формат — основной, число десятичных знаков — 0
Средний балл	Числовой	Формат — основной, число десятичных знаков — 1
Увлечения	Текстовый	—

5. Выберите команду меню Файл ► Сохранить на панели инструментов и сохраните таблицу с именем Карта. На вопрос о создании ключевого поля ответьте Нет.
6. Закройте таблицу Карта, щелкнув на кнопке управления окном Закреть.
7. В правой панели окна Класс: база данных (при выбранной закладке Таблицы) выделите пункт Карта и щелкните на кнопке Конструктор, чтобы снова открыть таблицу Карта в режиме Конструктора.
8. Удалите поля, которые не важны при заполнении медицинской карты учащегося:
 - щелкните мышью в ячейке с названием поля, которое следует удалить;

- выберите в меню команду Правка ► Удалить строки.

9. Добавьте в таблицу новые поля в соответствии с табл. 5.2:

- щелкните мышью в поле, перед которым надо вставить новую строку;
- выберите команду меню Вставка ► Строки;
- заполните новую строку значениями в соответствии с табл. 5.2.

Таблица 5.2. Дополнительные поля таблицы Карта

Имя поля	Положение	Тип данных	Оформление
Номер	Перед полем Фамилия	Счетчик	Новые значения — последовательные
Имя	После поля Фамилия	Текстовый	—
Пол	После поля Имя	Текстовый	Значение по умол- чанию — «М»

10. Измените поле Полных лет:

- щелкните мышью в ячейке с именем поля и замените имя Полных лет на Дата рождения;
- измените тип данных Числовой на Дата/время, выберите формат поля — Краткий формат даты.

11. Сохраните изменения в таблице, выбрав команду меню Файл ► Сохранить.

12. Завершите работу с СУБД, выбрав команду меню Файл ► Выход.

Задание 5.2

Заполните базу данных значениями, создав записи по образцу на рис. 5.8. Отредактируйте записи.

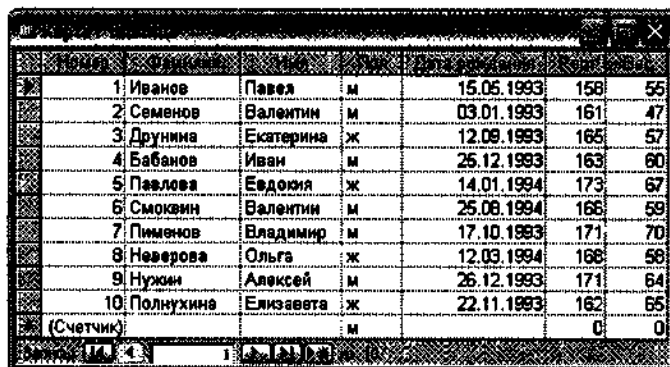
Технология работы

1. Запустите СУБД.

2. Откройте базу данных Класс.mdb:

- выберите команду меню Файл ► Открыть;

- в поле Папка выберите папку, в которой был сохранен файл базы;
- выделите файл Класс.mdb и щелкните на кнопке Открыть.



№	Фамилия	Имя	Пол	Дата рождения	Рост	Вес
1	Иванов	Павел	м	15.05.1993	158	55
2	Семенов	Валентин	м	03.01.1993	161	47
3	Друнина	Екатерина	ж	12.09.1993	165	57
4	Бабанов	Иван	м	25.12.1993	163	60
5	Павлова	Евдокия	ж	14.01.1994	173	67
6	Смоковин	Валентин	м	25.08.1994	166	59
7	Пименов	Владимир	м	17.10.1993	171	70
8	Неверова	Ольга	ж	12.03.1994	168	58
9	Нужин	Алексей	м	26.12.1993	171	64
10	Полнухина	Елизавета	ж	22.11.1993	162	65
(Счетчик)			м		0	0

Рис. 5.8. База данных с информацией об учениках

3. Откройте таблицу Карта в режиме таблицы:

- в левой панели окна Класс: база данных выберите закладку Таблица;
- в правой панели выделите пункт Карта;
- нажмите кнопку Открыть на панели инструментов.

4. Введите в базу данных первую запись:

- щелкните мышью в свободной ячейке в поле (столбце) Фамилия;
- введите значение поля Фамилия для первой записи — *Иванов* (записи будет автоматически присвоен порядковый номер 1 и в таблицу будет добавлена строка для следующей записи);
- нажмите клавишу Tab, чтобы перейти к заполнению следующего поля — Имя;
- введите значение имени для первой записи — *Павел* и перейдите к следующему полю, нажав клавишу Tab;
- значение *м*, введенное в поле Пол автоматически (по умолчанию), нас в данном случае устраивает, поэтому можно сразу перейти к вводу поля Дата рождения;

- введите значение даты рождения *15.05.93* и нажмите клавишу Tab;
 - введите значение поля Рост — 158 нажмите клавишу Tab;
 - введите значение поля Вес — 55 нажмите 2 раза клавишу Tab, чтобы сделать активным второе поле второй записи.
5. Введите аналогичным образом остальные записи по образцу на рис. 5.8.
6. Сохраните изменения в таблице, выбрав команду меню Файл ► Сохранить.
7. Измените значение поля Фамилия для первой записи:
- перейдите в ячейку поля Фамилия первой записи со значением *Иванов* с помощью клавиш управления курсором или клавиши Tab;
 - нажмите клавишу F2;
 - измените текст на *Иванченко*;
 - нажмите клавишу Tab.
8. Аналогичным образом внесите изменения в записи 3, 5 и 8:

Номер записи	Имя поля	Исходное значение	Отредактированное значение
3	Рост	165	170
5	Имя	Евдокия	Олимпиада
8	Дата рождения	12.03.1994	11.11.1993

9. Закройте СУБД (команда меню Файл ► Выход), не сохраняя последних изменений в базе данных.

Задания для самостоятельной работы

Задание 5.3. Создайте базу данных по книгам домашней библиотеки, введя следующие параметры: жанр книги, автор, название, год издания, комментарии.

Задание 5.4. Создайте базу данных музыкальных групп, задав, например, такие параметры: название группы, страна, основное

музыкальное направление группы, наиболее известный репертуар и т. п.

Задание 5.5. Создайте базу данных по интересующему вас виду спорта, отразив в ней место и время проведения соревнований, победителей и т. п.

Задание 5.6. Создайте базу данных на свободную тему.

Тема 5.3 Создание формы базы данных

Выполнив задания этой темы, вы научитесь:

- изменять параметры форматирования полей;
- создавать формы для базы данных;
- изменять местоположение полей на форме;
- включать в форму дополнительные элементы форматирования.

Задание 5.7

Просмотрите таблицу Карта в режиме таблицы. Научитесь изменять ширину и параметры форматирования полей таблицы.

Разработайте дизайн формы для записи, выберите оптимальное расположение полей на форме. Научитесь изменять оформление полей. Поместите на форму картинку. Используя встроенные средства СУБД, нарисуйте рамку.

В режиме формы вид записи должен соответствовать образцу на рис. 5.9.

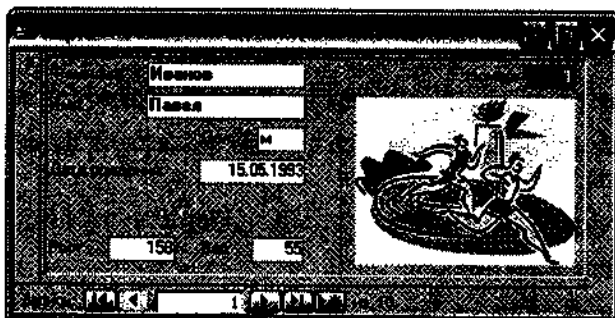


Рис. 5.9. Форма для ввода записей в базу данных

Технология работы

1. Запустите СУБД.
2. Откройте базу данных Класс.mdb.

3. Откройте таблицу Карта в режиме таблицы.
4. Установите ширину первого поля точно по ширине записи:

- выделите поле Номер, щелкнув на заголовке;
- выберите команду меню Формат ▶ Ширина столбца;
- в открывшемся диалоговом окне (рис. 5.10) щелкните на кнопке По ширине данных — ширина столбца будет подобрана по размеру самого длинного значения в поле или по ширине заголовка.

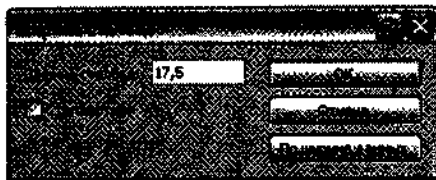


Рис. 5.10. Окно настройки ширины столбца

5. Аналогичным образом отформатируйте все поля таблицы и сохраните ее.
6. Установите параметры шрифта таблицы:
 - выберите команду меню Формат ▶ Шрифт;
 - выберите в открывшемся окне шрифт Times New Roman, начертание — полужирный, размер — 12.
7. Закройте таблицу, не сохраняя ее.
8. Создайте форму для записи таблицы Карта:
 - в левой панели окна Класс: база данных выберите закладку Формы;
 - щелкните на кнопке Создать на панели инструментов;
 - в открывшемся диалоговом окне выделите пункт Автоформа: в столбец;
 - в качестве источника данных выберите таблицу Карта;
 - нажмите кнопку ОК.

Поля вашей записи будут выведены на форму и расположены в том порядке, в котором они были внесены в таблицу.

9. Сохраните форму, выбрав команду меню Файл ▶ Сохранить и присвоив ей имя Form1.
10. Откройте созданную форму в режиме конструктора, выбрав команду меню Вид ▶ Конструктор.

11. Расположите поля на листе так, как показано на образце. Для перемещения поля:

- выделите поле щелчком мыши;
- переместите поле на новое место.



Чтобы посмотреть, как будет выглядеть готовая форма, переключитесь в режим формы, выбрав команду меню Вид ▶ Режим формы. Для продолжения редактирования вернитесь в режим конструктора (Вид ▶ Конструктор).

12. Установите ширину полей точно по длине записи:

- щелкните мышью на поле — по его контуру появятся маркеры выделения;
- наведите указатель мыши на маркер выделения, нажмите левую кнопку мыши и, не отпуская ее, перемещайте маркер в нужную сторону;
- отпустите кнопку мыши.

13. Измените цвет фона и шрифта для поля Номер:

- щелкните правой кнопкой мыши на поле Номер;
- выделите в контекстном меню команду Цвет заливки/фона и в открывшейся палитре выберите темно-серый цвет;
- выделите в контекстном меню команду Цвет текста и выберите белый цвет.

14. Измените начертание шрифта для поля Номер:

- щелкните на нем правой кнопкой мыши;
- выберите в контекстном меню команду Свойства;
- в открывшемся окне на вкладке Макет найдите параметр Насыщенность и назначьте ему значение Полужирный.

15. Установите полужирное начертание шрифта для полей Фамилия и Имя.

16. Поместите на форму картинку:

- щелкните мышью в любом свободном месте формы;
- выберите команду меню Вставка ▶ Рисунок;

- в открывшемся окне выберите папку с сохраненным в ней заранее рисунком на подходящую тему (например, медицина или спорт);
 - выделите файл с рисунком и щелкните на кнопке Вставить.
17. Подберите размер и расположение рисунка, чтобы он не закрывал текст:
- щелкните на рисунке мышью — по его контуру появятся маркеры выделения;
 - наведите указатель мыши на один из маркеров выделения, нажмите левую кнопку мыши и, не отпуская ее, перемещайте маркер в нужную сторону;
 - отпустите кнопку мыши;
 - подберите размер рамки рисунка, перемещая соответствующие маркеры;
 - после того как положение и размер рамки рисунка подобраны, щелкните на рисунке правой кнопкой мыши и выберите в контекстном меню команду Свойства;
 - на вкладке Макет найдите параметр Установка размеров и установите для него значение По размеру рамки.
18. Нарисуйте рамку:
- щелкните на кнопке Прямоугольник на панели элементов; 
 - нарисуйте рамку вокруг всех элементов формы.
19. Сохраните форму.
20. Перейдите в режим формы и просмотрите записи таблицы Карта.
21. Закройте СУБД, выбрав команду меню Файл ► Выход.

Задания для самостоятельного выполнения

Задание 5.8. Разработайте дизайн формы ввода данных для одной из баз данных, созданных самостоятельно в теме 5.2.

Тема 5.4 Работа с записями базы данных

Выполнив задания этой темы, вы научитесь:

- редактировать записи;
- осуществлять сортировку;
- создавать простые фильтры и фильтры с формулами;
- создавать вычисляемые поля.

Задание 5.9

Создайте модель страницы журнала для урока физкультуры. Отсортируйте экземпляры записи в таблице так, чтобы сначала шел список фамилий мальчиков, затем — список фамилий девочек. Создайте календарь поздравлений. Отредактируйте параметры поля Дата рождения, затем отсортируйте записи.

Технология работы

1. Запустите СУБД.
2. Откройте базу данных Класс.mdb.
3. Откройте таблицу Карта в режиме таблицы.
4. Разделите учащихся на мальчиков и девочек:
 - выберите щелчком мыши поле Пол;
 - выберите команду меню Записи ▶ Сортировка по убыванию.
5. Откройте таблицу в режиме конструктора, выбрав команду меню Вид ▶ Конструктор.
6. Отредактируйте поле Дата рождения:
 - щелкните мышью в поле Дата рождения;
 - на панели Свойства поля измените значение параметра Формат поля на Длинный формат даты (в этом формате число записывается цифрами, месяц — прописью, год полностью).
7. Вернитесь в режим таблицы.

8. Отсортируйте дни рождения в хронологическом порядке:
 - щелкните в любой ячейке поля Дата рождения;
 - выберите команду меню Записи ▸ Сортировка ▸ Сортировка по возрастанию.
9. Закройте СУБД, выбрав команду меню Файл ▸ Выход.

Задание 5.10

Создайте следующие простые фильтры:

- ◆ *Выше* — отбирает записи об учениках, чей рост более 170 см;
- ◆ *Легче* — отбирает записи об учениках, чей вес меньше или равен 65 кг;
- ◆ *Мало ест* — отбирает записи об учениках, чей рост больше 170 см, а вес меньше или равен 67 кг;
- ◆ *Отклонение* — отбирает записи об учениках, чьи данные находятся в диапазоне: $65 \text{ кг} < \text{вес} < 71 \text{ кг}$, $166 \text{ см} < \text{рост} < 171 \text{ см}$.

Технология работы

1. Запустите СУБД.
2. Откройте базу данных Класс.mdb.
3. Откройте таблицу Карта в режиме таблицы.
4. Создайте фильтр Выше:
 - выберите команду меню Записи ▸ Фильтр ▸ Расширенный фильтр — появится диалоговое окно КартаФильтр1;
 - в строке Поле выберите из списка имя фильтруемого поля — Рост;
 - в строке Условие отбора введите >170 ;
 - выберите команду меню Фильтр ▸ Применить фильтр — на экране появятся результаты фильтрации (рис. 5.11).

5	Павлова	Евдокия	ж	14.01.1994	173	67
7	Пименов	Владимир	м	17.10.1993	171	70
9	Нужин	Алексей	м	26.12.1993	171	64

Рис. 5.11. Результат работы фильтра Выше

5. Создайте фильтр Легче:

- выберите команду меню Записи ► Фильтр ► Расширенный фильтр;
- в открывшемся окне в строке Поле выберите из списка имя фильтруемого поля Вес (вместо поля Рост);
- в строке Условие отбора введите ≤ 65 ;
- выберите команду меню Фильтр ► Применить фильтр — на экране появятся результаты фильтрации (рис. 5.12).

1	Иванов	Павел	м	15.05.1993	158	55
2	Семенов	Валентин	м	03.01.1993	161	47
3	Друнина	Екатерина	ж	12.09.1993	165	57
4	Бабанов	Иван	м	25.12.1993	163	60
5	Смокин	Валентин	м	25.08.1994	166	59
6	Неверова	Ольга	ж	12.03.1994	168	58
9	Нужин	Алексей	м	26.12.1993	171	64
10	Полнухина	Елизавета	ж	22.11.1993	162	65

Рис. 5.12. Результат работы фильтра Легче

6. Создайте фильтр Мало ест:

- выберите команду меню Записи ► Фильтр ► Расширенный фильтр;
- в открывшемся окне в строке Поле выберите из списка имя фильтруемого поля — Рост;
- в строке Условие отбора введите > 170 ;
- во второй ячейке строки Поле выберите из списка имя второго фильтруемого поля — Вес;
- в строке Условие отбора введите ≤ 65 ;
- выберите команду меню Фильтр ► Применить фильтр — на мониторе появятся результаты фильтрации (рис. 5.13).

9	Нужин	Алексей	м	26.12.1993	171	64
---	-------	---------	---	------------	-----	----

Рис. 5.13. Результат работы фильтра Мало ест

7. Создайте фильтр Отклонение:

- выберите команду меню Записи ► Фильтр ► Расширенный фильтр;

- в открывшемся окне в строке Поле выберите из списка имя фильтруемого поля — Рост;
- в строке Условие отбора введите $>=171$;
- в строке или введите $<=166$;
- во втором столбце строки Поле выберите из списка имя фильтруемого поля — Вес;
- в строке Условие отбора введите >71 ;
- в строке или введите <65 ;
- выберите команду меню **Фильтр** ▶ **Применить фильтр** — на экране появятся результаты фильтрации.

8. Закройте СУБД, выбрав команду меню **Файл** ▶ **Выход**.

Задание 5.7.1.

Создайте фильтр с формулой Идеальный вес, отбирающий записи об учениках, чей вес пропорционален росту. Пропорция вычисляется по формуле:

Идеальный вес = Рост – 100

Допустимое отклонение — 3 кг.

Технология работы

1. Запустите СУБД.
2. Откройте базу данных Класс.mdb.
3. Откройте таблицу Карта в режиме таблицы.
4. Выберите команду меню **Записи** ▶ **Фильтр** ▶ **Расширенный фильтр**.
5. В строке Поле выберите из списка имя фильтруемого поля — Рост.
6. Создайте формулу для фильтра:
 - щелкните правой кнопкой мыши в строке Условие отбора и выберите в контекстном меню команду **Построить** — откроется диалоговое окно **Построитель выражений**;
 - в левом списке выберите двойным щелчком пункт **Таблицы** — **Карта**;
 - в среднем списке выберите пункт **Рост** и щелкните на кнопке **Вставить** — в верхнем поле появится ссылка на поле Рост;

- * щелкните на кнопке $\leftarrow \rightarrow$ (минус), наберите в верхнем поле число 100 и еще раз щелкните на кнопке $\leftarrow \rightarrow$;
- * выберите в среднем списке пункт Вес и щелкните на кнопке Вставить;
- * выберите в левом списке пункт Операторы, в правом — знак $\leq$ и щелкните на кнопке Вставить;
- * введите в верхнем поле число 3, чтобы завершить формулу;
- * щелкните на кнопке ОК.

В результате этих действий в поле Условие отбора появится формула:

$[Карта]![Рост] - 100 - [Карта]![Вес] \leq 3$

7. Выберите команду меню **Фильтр** $\triangleright$ **Применить фильтр** — на мониторе появятся результаты фильтрации.
8. Закройте СУБД, выбрав команду меню **Файл** $\triangleright$ **Выход**.

Задание 11

Рассчитайте оптимальный вес учащихся, создав в базе данных запрос Идеальный вес с вычисляемым полем и введя в него формулу $Рост - 100$.

Технология работы

1. Запустите СУБД.
2. Откройте базу данных Класс.mdb.
3. Создайте новый запрос:
 - в левой панели окна Класс: база данных выберите закладку Запросы;
 - в правой панели выберите пункт Создание запроса в режиме конструктора;
 - * щелкните на кнопке Открыть на панели инструментов.
4. В диалоговом окне Добавление таблицы выберите таблицу Карта, щелкните на кнопке Добавить, а затем — на кнопке Закрыть.
5. Создайте поле запроса Фамилия:
 - установите указатель мыши в таблице на поле Фамилия;
 - * нажмите левую клавишу мыши и перетащите ее в первую ячейку строки Поле.

6. Аналогичным образом создайте поля Имя и Вес, перетащив их названия из таблицы во вторую и третью ячейки строки Поле.
7. Добавьте в запрос вычисляемое поле:
 - щелкните правой кнопкой мыши в свободной ячейке строки Поле и в открывшемся контекстном меню выберите команду Построить;
 - в окне построителя выражений выберите таблицу Карта, поле Рост и поместите его в верхнее поле щелчком на кнопке Вставить;
 - щелкните на кнопке $\leftarrow \rightarrow$ и введите в верхнее поле число 100;
 - нажмите кнопку ОК.
8. Замените в ячейке с формулой Выражение1: [Карта].[Рост]-100 слово Выражение1 на Оптимальный вес.
9. Сохраните запрос с именем Идеальный вес.
10. Посмотрите результат выполнения запроса с вычисляемым полем, выбрав команду меню Запрос ► Запуск (рис. 5.14).

№	Имя	Рост	Вес	Идеальный вес
1	Иванов	Павел	58	55
2	Семенов	Валентин	61	47
3	Друнина	Екатерина	65	57
4	Бабанов	Иван	63	60
5	Павлова	Евдокия	73	67
6	Смокин	Валентин	66	59
7	Пименов	Владимир	71	70
8	Неверова	Ольга	68	58
9	Нужин	Алексей	71	64
10	Полнухина	Елизавета	62	65

Рис. 5.14. Результаты выполнения запроса с вычисляемым полем Идеальный вес

11. Закройте СУБД, выбрав команду меню Файл ► Выход.

Задания для самостоятельной работы

Задание 5.13. Для созданной в теме 5.2 базы данных произведите все возможные варианты фильтрации. Добавьте в них вычисляемые поля.

Тема 5.5 Разработка отчета для вывода данных

Выполнив задания этой темы, вы научитесь:

- ⇒ создавать отчеты;
- ⇒ устанавливать список полей для включения в отчет;
- ⇒ задавать порядок сортировки данных в полях;
- ⇒ устанавливать способ подведения итогов.

Задание 5.14

Создайте отчет Анализ группы. Данный отчет должен наглядно представлять информацию о росте и весе учеников.

Технология работы

1. Запустите СУБД.
2. Откройте базу данных Класс.mdb.
3. Создайте новый отчет:
 - в левой панели окна Класс: база данных выберите закладку Отчеты;
 - щелкните на кнопке Создать на панели инструментов;
 - в открывшемся окне выберите пункт Мастер отчетов;
 - выберите в качестве источника данных таблицу Карта;
 - нажмите кнопку ОК.
4. Выберите поля для включения в отчет:
 - в списке Доступные поля выберите поле Фамилия и нажмите кнопку > — поле Фамилия появится в списке Выбранные поля;
 - аналогичным образом добавьте поля Имя, Рост, Вес;
 - нажмите кнопку Далее.
5. Добавьте группировку по полю Рост:
 - выделите поле Рост и нажмите кнопку >;
 - нажмите кнопку Далее.

6. Установите порядок сортировки:
 - в списке 1 выберите поле Вес;
 - щелкните на кнопке по возрастанию, чтобы изменить порядок сортировки на сортировку по убыванию;
 - в списке 2 выберите поле Фамилия;
 - оставьте порядок сортировки по возрастанию.
7. Установите способ подведения итогов:
 - нажмите кнопку Итоги;
 - в открывшемся окне установите для поля Вес флажки для значений Avg (Средний), Min (Минимальный), Max (Максимальный);
 - нажмите кнопку ОК;
 - нажмите кнопку Далее.
8. Выберите вид макета и ориентацию страницы:
 - установите переключатель напротив макета ступенчатый;
 - установите переключатель напротив ориентации альбомная;
 - нажмите кнопку Далее.
9. Выберите стиль отчета Обычный, нажмите кнопку Далее.
10. Введите имя отчета — Анализ группы.
11. Установите переключатель дальнейших действий в положение Просмотреть отчет.
12. Нажмите кнопку Готово — готовый отчет будет выведен на экран.
13. Закройте СУБД, выбрав команду меню Файл ► Выход.

Задания для самостоятельной работы

Задание 5.15. Для созданной в теме 5.2 базы данных разработайте отчет и произведите с его помощью вывод данных.

Задание 5.16. Создайте базу данных Записная книжка и поместите в нее 30 записей. Сформируйте итоговый отчет так, чтобы информация в нем была представлена как в обычной записной книжке.

Коммуникации в глобальной сети Интернет

Тема 1	Возможности Интернета	206
Тема 2	Среда браузера Internet Explorer	211
Тема 3	Поиск информации в сети Интернет	216
Тема 4	Язык разметки гипертекста HTML	223
Тема 5	Веб-страница с графическими объектами	234
Тема 6	Веб-страница с гиперссылками	237
Тема 7	Возможности электронной почты	240

Тема 6.1 Возможности Интернета

Изучив эту тему, вы узнаете:

- каково назначение Интернета;
- какие существуют виды коммуникаций;
- что такое модель «клиент — сервер»

Компьютерная система Интернет

Сеть Интернет состоит из множества объектов. Тем не менее мы говорим о ней в единственном числе. Почему? Мы всегда говорим о совокупности объектов в единственном числе, если, взаимодействуя, они воспринимаются нами как единое целое, то есть система.

Компьютерная сеть является системой, а компьютеры — ее объектами. В современном мире существует множество компьютерных сетей. Сначала появились *локальные сети* — системы компьютеров, которые находятся в непосредственной близости друг от друга и соединены между собой при помощи специального кабеля. Соединяя компьютеры, находящиеся в ведении одной организации, но разделенные большими расстояниями, при помощи уже существующих каналов связи, например телефонной линии, получают *корпоративные сети*. Такими являются, например, банковские сети. *Региональные сети* объединяют компьютеры в пределах региона или города.

В каждой локальной, корпоративной или региональной сети существуют собственная *система адресации* (способ присвоения имен компьютерам) и свои *протоколы* (правила передачи сообщений). Для того чтобы существующие сети представляли собой систему, необходимо соединить их между собой. Это означает создание такого протокола передачи данных, который не зависел бы ни от одной из существующих компьютерных сетей, а также создание общего для всех сетей адресного пространства. Когда эти проблемы были решены, появилась всемирная сеть Интернет (англ. *Internet* — *international net*, международная сеть).

Сетевые коммуникации

Компьютеры и компьютерные сети создавались людьми для удовлетворения их потребностей. Одной из важных потребностей человека является общение. Формально общение между людьми представляет собой обмен сведениями, то есть взаимные передачу и восприятие информации. Для обозначения такого обмена используется специальный термин — *коммуникация*.

Например, разговор между людьми представляет собой речевую коммуникацию, причем средством коммуникации в данном случае является устная речь. Однако для использования этого средства коммуникации людьми, удаленными друг от друга, нужны специальные технические средства передачи речи на расстояние. Пример такой удаленной коммуникации — разговор по телефону, возможный благодаря существованию *телефонной сети*.

Другой пример удаленной коммуникации — переписка по почте. Средством коммуникации в данном случае является письмо. Для передачи рукописного или печатного текстового сообщения на расстояние можно воспользоваться услугами *почтовой службы*.

Современным и все более распространенным средством коммуникации является электронное сообщение, которое создается на компьютере. Для передачи электронных сообщений используют *компьютерные сети*. В компьютерной сети существует несколько видов коммуникаций.

Во-первых, существует множество способов создания электронных сообщений. Каждому виду данных соответствует свой способ передачи, то есть свой вид коммуникации.

Во-вторых, можно, используя одно и то же средство коммуникации, реализовать разные виды коммуникации. Например, речевая коммуникация может быть реализована по разным принципам: беседа — «один на один» или конференция — «один ко многим».

В-третьих, существуют специфические виды коммуникаций, которые разрабатывают и предоставляют службы, обеспечивающие информационные услуги. Например, различные телефонные службы готовят информационные сообщения, которые вы можете прослушать, набрав определенный номер телефона, то есть запросить и получить определенный информационный продукт.

Почтовая служба, как известно, предоставляет специфическую услугу — пересылку не только писем, но и посылок. Компьютерная сеть также предоставляет коммуникационные услуги, к числу которых относятся перечисленные ниже.

E-mail, или *электронная почта* — наиболее распространенный вид сетевой коммуникации. Электронное письмо представляет собой обычный текст с подготовленным специальным образом заголовком. Услуга компьютерной сети состоит не только в том, чтобы переслать подготовленное таким образом электронное сообщение с компьютера на компьютер, но и в том, чтобы представить его в виде, удобном для чтения и ответа.

FTP, или *пересылка файлов*, — один из самых «древних» видов сетевой коммуникации. Любой компьютер изначально представляет собой хранилище файлов. В компьютерных сетях некоторые компьютеры в результате установки на них специальных программ становятся библиотеками файлов и предоставляют их для получения бесплатно (*freeware*) и условно бесплатно (*shareware*). Эта услуга компьютерной сети называется так же, как и специальные правила, или протокол (*protocol*), регламентирующий передачу (*transfer*) файлов (*file*) с компьютера на компьютер, — *File Transfer Protocol*, или *FTP*. Необходимо заметить, что любое электронное сообщение на компьютере представляет собой файл. Однако при пересылке по компьютерным сетям файлами называют только заархивированные документы, готовые программы и пр., то есть такие электронные сообщения, которые не имеет смысла сразу «разворачивать» на экране, так как прежде они должны быть «прочитаны» и «обработаны» компьютером.

Telnet — особый вид сетевой коммуникации. Для описания его сути используем следующий сравнительный образ. Представьте себе, что вы пользуетесь двумя столами, находящимися в разных помещениях, например в разных офисах одной организации. В данный момент вы находитесь в первом офисе и сидите за одним из столов, но у вас возникла необходимость воспользоваться другим столом, например посмотреть какой-то документ, оставшийся на нем. То есть у вас появляется необходимость получить *удаленный доступ* к столу, находящемуся далеко от вас. Компьютерная сеть может предоставить вам возможность, находясь за одним компьютером, получить удаленный доступ к дру-

тому компьютеру в рамках имеющихся у вас прав на использование его ресурсов.

WWW — новейший вид сетевой коммуникации. WWW расшифровывается как World Wide Web — всемирная информационная паутина. Средством коммуникации между компьютерами, входящими в систему WWW, являются *гипертексты*. Подобные тексты содержат слова, которые «реагируют» на движение мыши. На экране монитора гипертекст выглядит как текст, в котором некоторые слова выделены цветом и подчеркиванием. Выделенные слова являются *гиперссылками*. Щелчок мыши на гиперссылке приведет к тому, что на экран будет выведен связанный с этой ссылкой электронный текст, возможно, хранящийся на другом компьютере сети.

Модель «клиент — сервер»

Итак, как только люди удалены друг от друга, им для реализации коммуникации необходимы услуги какой-либо службы. Так, для обмена письмами люди отправляются на почту, то есть становятся клиентами почтовой службы. На почте их обслуживают — предоставляют почтовый сервис. Сервис означает умение обслужить: сначала принять заказ от клиента, а затем точно и правильно исполнить его. Модель отношений «клиент — сервис» реализована во многих областях нашей жизнедеятельности. По этой же модели организованы и сетевые коммуникации в Интернете.

Обмен данными в компьютерной сети происходит между компьютерами, поэтому для того чтобы компьютер стал клиентом, на нем должна быть установлена *программа-клиент*, а для того чтобы компьютер «умел» обслуживать клиентов, на нем должна быть установлена *программа-сервер*. Слова «сервер» и «сервис» — родственные и означают «обслуживание». Такие две взаимосвязанные программы существуют в Интернете для каждого вида сетевых коммуникаций.

Программы, установленные на компьютере, «наделяют» его теми или иными функциями, которые он выполняет во время их работы. Так что на одном компьютере может быть установлена и программа-сервер, и программа-клиент одновременно. Кроме

того, следует заметить, что сервером называют как специальным образом укомплектованный компьютер, так и программу.

Контрольные вопросы

1. Что следует понимать под термином «Интернет»?
2. Какие виды коммуникаций вы знаете?
3. Что является средством коммуникации в сети Интернет?
4. Какие виды сетевых коммуникаций можно использовать в Интернете?
5. Что такое модель «клиент — сервер»?

Тема 6.2 Среда браузера Internet Explorer

Выполнив задания этой темы, вы научитесь:

- работать в среде браузера Internet Explorer;
- просматривать веб-страницы.

Сеть Интернет представляет собой систему компьютеров, на которых установлены специальные программы — WWW-серверы (веб-серверы). Именно веб-серверы делают доступными для пользователей сети Интернет размещаемые на них электронные сообщения, созданные в виде гипертекста и называемые *веб-страницами*, или *веб-документами*. Совокупность всех размещенных на веб-серверах веб-страниц составляет информационную систему World Wide Web.



World Wide Web — это система, объектами которой является гипертекстовая информация, предоставляемая компьютерами сети Интернет с помощью программы веб-сервер.

Страницы в книге читают, а веб-страницы — просматривают, используя для этого специальные программы. Просматривать — по-английски *to browse*, соответствующее существительное — *browser* (произносится как «браузер»).



Браузер — это программа, предназначенная для просмотра веб-страниц.

Программа-браузер является программой веб-клиентом, то есть позволяет пользователю запрашивать, получать и изучать информацию, размещенную на веб-серверах.

Существует много программ-браузеров. У них одна основная функция — обеспечить отображение веб-страниц. Но все они предоставляют пользователю разные возможности при работе с этими документами. Например, на маломощных компьютерах используются браузеры, которые не способны отображать графику. Графические браузеры, как правило, предоставляют пользователю

возможность включать и выключать отображение изображений. Эта важная особенность браузера полезна в том случае, если передача информации по сети происходит медленно. Наиболее известными браузерами, имеющими самые широкие возможности при отображении веб-страниц, являются *Netscape Navigator* и *Microsoft Internet Explorer*.

Задание 6.1

Познакомьтесь с основными возможностями и функциями браузера Internet Explorer. При работе с браузером изучите основные поля главного окна браузера, команды меню, научитесь работать с закладками.

Технология работы

1. Запустите приложение Internet Explorer.
2. В главном окне браузера (рис. 6.1) основную часть рабочей области составляет *область просмотра* веб-страницы. Если веб-страница целиком не помещается в окне, то можно использовать вертикальную и горизонтальную полосы прокрутки для полного просмотра, как и в других приложениях Windows. Кроме того, можно открыть сразу несколько окон для просмотра веб-страницы. Для этого воспользуйтесь командой меню **Файл** ► **Создать** ► **Окно**.
3. Рассмотрите главное окно браузера и изучите основные поля: поле заголовка окна, строку состояния, строку адреса.
 - *Поле заголовка* содержит название загруженной страницы.
 - *Строка состояния* отображает ход загрузки данных, то есть состояние работы браузера. Найти эту строку легко — она расположена в нижней части окна. Одна из функций строки состояния состоит в следующем: если переместить курсор мыши на слово, являющееся гиперссылкой, то в строке состояния отобразится адрес веб-страницы или файла, содержание которого связано с этой ссылкой. Это позволяет вам, прежде чем щелкнуть на этой гиперссылке, узнать, куда она ведет.
 - *Строка адреса* содержит полный адрес документа, отображаемого в окне браузера. Строка адреса находится над областью просмотра главного окна.

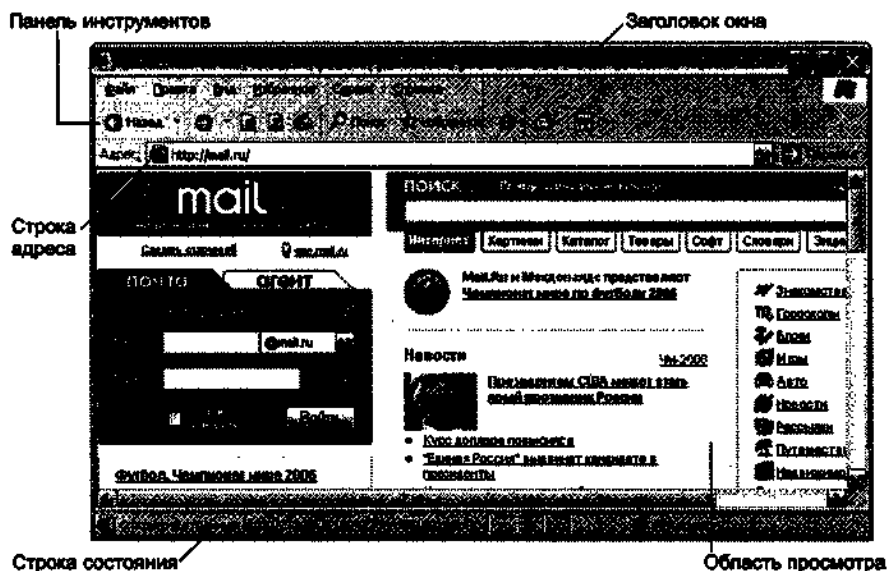


Рис. 6.1. Главное окно браузера Microsoft Internet Explorer

4. Познакомьтесь с кнопками *панели инструментов* окна браузера Internet Explorer.

Панель инструментов находится под строкой меню и включает часто используемые команды браузера, чтобы предоставить пользователю возможность быстро выбирать команды с помощью мыши. Кнопки панели инструментов снабжены текстовыми надписями, из которых понятны их функции.

- Кнопка **Избранное** открывает панель, в которой отображаются сохраненные ранее ссылки на веб-страницы (закладки). Щелкните мышью на кнопке **Избранное**. В открывшейся панели (рис. 6.2) познакомьтесь со списком уже сделанных закладок. Щелкните мышью на одной из строк панели и изучите информацию, появившуюся в главном окне браузера. На другие веб-страницы перейдите, используя гиперссылки.
- Кнопка **Добавить** в верхней части панели **Избранное** используется для создания собственной закладки. Сделать закладку — значит дать браузеру команду запомнить адрес открытой в браузере веб-страницы.

- Кнопка Назад используется для перехода к веб-страницам, просмотренным ранее. Для перехода щелкните мышью на кнопке. Просмотрите содержание веб-страниц в обратном порядке и сделайте анализ достоинств и недостатков такого «блуждания» по WWW, начавшегося с некоторой начальной страницы.

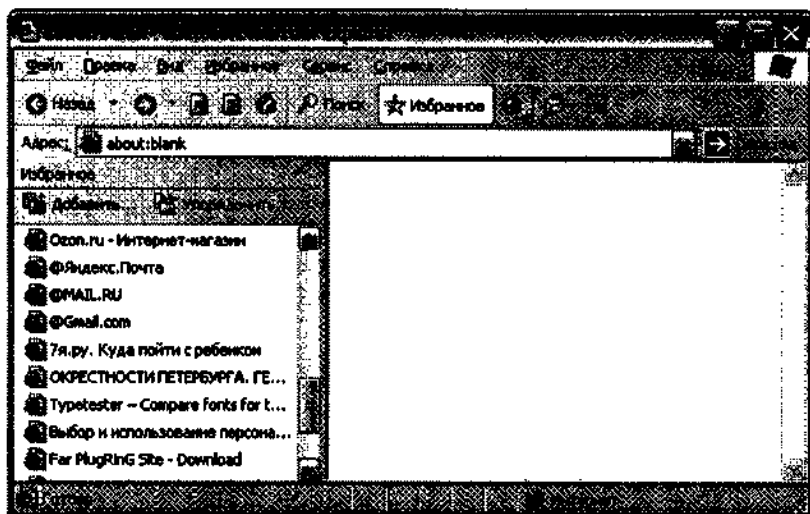


Рис. 6.2. Панель Избранное

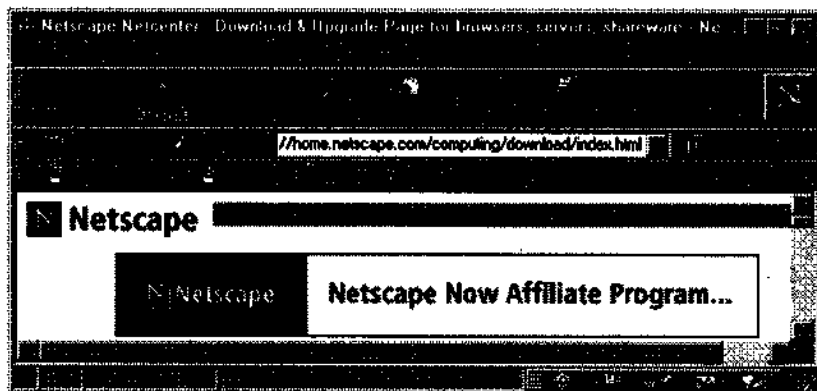


Рис. 6.3. Окно браузера Netscape Navigator

Задания для самостоятельной работы

Задание 6.2. Рассмотрите окно браузера Netscape Navigator (рис. 6.3). Эта программа является условно бесплатно распространяемым продуктом. Ее можно загрузить на компьютер с веб-страницы:

<http://home.netscape.comcomputing/download/index.html>.

Задание 6.3. Назовите различия окон браузеров Netscape Navigator и Internet Explorer.

Задание 6.4. Назовите различия панелей инструментов браузеров Netscape Navigator и Internet Explorer.

Тема 6.3 Поиск информации в сети Интернет

Выполнив задания этой темы, вы научитесь:

- ⇒ разбирать структуру адреса Интернета;
- ⇒ искать информацию по заданным адресам;
- ⇒ искать информацию по ключевым словам.

Веб-сервер — это компьютер, на котором установлено специальное программное обеспечение. **Веб-сайт** — это место на веб-сервере (слово *сайт* происходит от английского слова и в переводе на русский означает *место*). Адреса веб-серверов и веб-сайтов печатаются в газетах, журналах; их можно увидеть на экране телевизора в конце программы или услышать из уст диктора любимой радиостанции.

В сети Интернет принята числовая *система адресации*. Это означает, что каждый компьютер имеет свой уникальный адрес, в чем-то подобный почтовому индексу. Адрес разделен на четыре блока, например: 194.135.81.254.

Компьютеры предпочитают числа, а люди — имена, так как имена легче запомнить. Поэтому в Интернете кроме числовой системы адресации была введена очень удобная *система доменных имен*. Согласно этой системе, компьютеры в сети Интернет относятся к той или иной тематической или географической области. Область — по-английски *domain* (домен), поэтому система имен и получила название доменной.

Доменное имя составляется из имен нескольких доменов, отделенных друг от друга точками. Доменов в имени обычно два или три, реже — четыре и более. Первым справа стоит домен верхнего уровня, затем уровни доменов убывают.

Рассмотрим адрес `edusom.spb.ru`. Домен высшего уровня `ru` означает, что веб-сервер с этим именем зарегистрирован в Российской Федерации. Затем идет домен второго уровня — `spb`, который означает, что сайт зарегистрирован в Санкт-Петербурге. Домен третьего уровня — `edusom` — сообщает название организации, на которую зарегистрирован данный доменный адрес.

Таким образом, имя `educom.spb.ru` в Интернете принадлежит Комитету по образованию Администрации Санкт-Петербурга.

Вся последовательность символов, используемых в адресе, называется *URL*. Это аббревиатура слов *Uniform Resource Location*, обозначающая не только местонахождение веб-документа в сети, но и способ его передачи. Пользователи Интернета часто используют термины «адрес» и «URL» как синонимы.

Для поиска интересующей вас информации необходимо указать браузеру адрес веб-страницы, на которой она находится, или адрес веб-сервера, где размещена эта страница.

Если вы не знаете точного адреса сервера, то для поиска информации в сети Интернет можно воспользоваться *поисковыми системами*. Каждая поисковая система — это обширная база *ключевых слов*, связанных с веб-страницами, на которых они встретились. Ключевыми являются любые слова, которые встречаются в документе основными. Для создания баз ключевых слов используются специальные программы, которые делают автоматический обход веб-серверов в сети и просматривают страницу за страницей. Это так называемые *программы-роботы*, которые выполняют индексирование ключевых слов, то есть подсчитывают, сколько раз встретилось каждое из них. URL каждого ключевого слова запоминается в базе данных и может быть восстановлен по соответствующему запросу.

Поисковые системы отличаются друг от друга по логике своей работы: языку запроса, представлению адресной информации и выходных документов, совместимости с другими поисковыми системами и т. д.

Для поиска адреса сервера с интересующей вас информацией надо ввести в поисковой системе ключевое слово, несколько слов или фразу. Таким образом вы посылаете поисковой системе *запрос*. В ответ она выведет список адресов веб-страниц, зачастую очень длинный, на которых встречаются искомые ключевые слова.



Как бы вам ни было интересно продвигаться «в глубь» Интернета, начав с любой из этих гиперссылок, — остановитесь и перейдите к следующей. Работая в сети Интернет, всегда следуйте заранее намеченному плану, в противном случае по окончании сеанса связи или, как обычно говорят, сессии, вы не сможете найти ответ на вопрос, который обязательно должны задать себе: «Что я сегодня узнал, нашел, в чем определился?»

Задание 6.5

Научитесь искать информацию по известным адресам.

Для выполнения этого задания желательно заранее иметь адреса веб-серверов (веб-сайтов). Используя подготовленные заранее или предлагаемые ниже адреса, познакомьтесь с информацией, хранящейся на этих серверах.

Заинтересовавшую вас информацию сохраните в файле. Сохраните ссылки на некоторые веб-страницы в Избранном.

Технология работы

1. Запустите приложение Internet Explorer.
2. Наберите в строке Адрес адрес веб-сервера города Санкт-Петербурга — www.spb.ru (рис. 6.4).

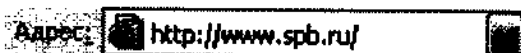


Рис. 6.4. Строка адреса с адресом официального веб-сервера Санкт-Петербурга



В строке Адрес перед непосредственным адресом сервера указаны символы <http://>, обозначающие протокол, который используется при передаче данной информации.

3. Перемещайте мышь и следите за изменением изображения указателя. Найдите гиперссылки и используйте их для перехода на следующие страницы веб-сервера. Изучите информацию, размещенную на веб-страницах сервера Санкт-Петербурга.
4. Введите в строке Адрес следующий URL: <http://www.nlr.ru/res>. Это электронные информационные ресурсы Российской национальной библиотеки, находящейся в Санкт-Петербурге (рис. 6.5).
5. Научитесь сохранять текст веб-страницы, открытой в окне браузера. Для этого используйте команду Файл ► Сохранить как и задайте имя файла.
6. Изучая информацию на веб-серверах, делайте необходимые закладки. Для этого используйте команду Избранное ► Добавить в избранное.

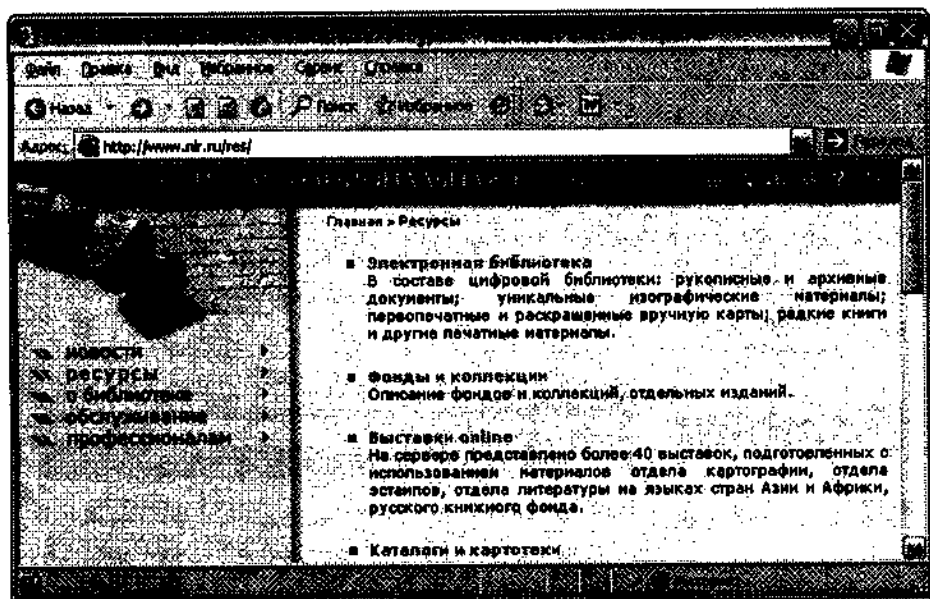


Рис. 6.5. Электронные информационные ресурсы Российской национальной библиотеки (Санкт-Петербург)

Задание 6.6

Научитесь искать информацию по ключевым словам.

Выполняя это задание, вы познакомитесь с несколькими российскими и зарубежными поисковыми системами. Обратите внимание на проблему ключевых слов, которые необходимо использовать для описания интересующей вас информации. Сохраните адреса заинтересовавших вас страниц.

Технология работы

1. Запустите приложение Internet Explorer.
2. В строку Адрес введите URL поисковой системы Rambler: www.rambler.ru.
3. Поле для запроса в любой поисковой системе представляет собой текстовое поле с кнопкой Найти, Искать, Поиск, Go или Search. Найдите поле для ввода запроса на веб-странице поисковой

системы Rambler, введите запрос, например *электронная информация*, и щелкните мышью на кнопке Поиск или нажмите Enter (рис. 6.6).

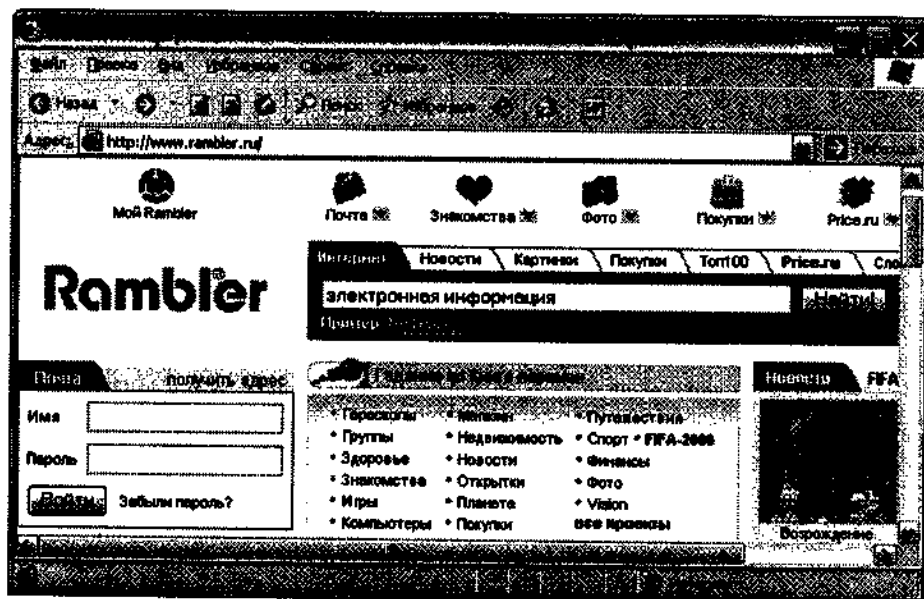


Рис. 6.6. Главная страница информационно-поисковой системы Rambler

4. В приведенном примере ключевым было не слово, а фраза. Результаты поиска представляют собой список гиперссылок, сопровождающихся кратким комментарием. Просматривайте комментарии к каждой из найденных гиперссылок и принимайте решение, на какой из них следует щелкнуть мышью, чтобы более подробно изучить информацию на соответствующей веб-странице. Исследуйте несколько наиболее интересных гиперссылок и перейдите к следующему пункту задания.
5. Введите в строке Адрес URL другой поисковой системы — Яндекс: www.yandex.ru (рис. 6.7). Яндекс, как и Rambler, — одна из крупнейших российских поисковых систем. Это означает, что базы ключевых слов этих поисковых систем содержат наибольшее количество ссылок на веб-страницы, находящиеся на русскоязычных веб-серверах.

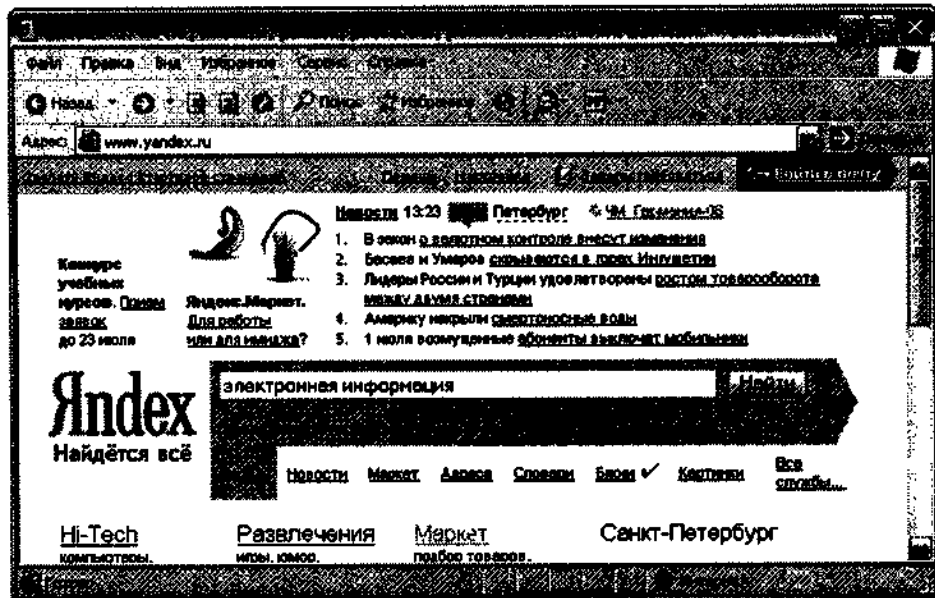


Рис. 6.7. Главная страница информационно-поисковой системы Яндекс

6. Введите ключевую фразу *электронная информация* в поле поиска на главной странице поисковой системы Яндекс. Сравните списки предлагаемых ссылок. Сделайте необходимые закладки.
7. Познакомьтесь с зарубежными поисковыми системами:

- <http://www.altavista.com>;
- <http://www.google.com>;
- <http://www.yahoo.com>.

Обратите внимание на то, что поле для запуска поиска по ключевым словам имеет название Search (Искать) или Go (Начать). Несмотря на то что это зарубежные поисковые системы, используйте в качестве ключевой прежнюю фразу — *электронная информация*. Проанализируйте результаты поиска.

Задания для самостоятельной работы

Задание 6.7. Исследуйте информацию, находящуюся в сети Интернет по следующему адресу: www.ic.al.lg.ua/resources (поиск людей и организаций в Интернете).

Задание 6.8. Сохраните в файле содержание веб-страницы, находящейся по адресу: karabas.actor.ru/doc/search.html (как искать информацию в Интернете).

Задание 6.9. Проверьте, существуют ли в сети веб-сайты широко известных фирм, радиостанций и телепрограмм. Например, фирма DOLE — всемирно известный поставщик бананов. Проверьте, ей ли принадлежит веб-сайт с адресом www.dole.com.

Задание 6.10. Познакомьтесь с русской поисковой системой Aport: www.apor.ru.

Задание 6.11. Изучите описания, различия и особенности русских поисковых систем по следующим веб-страницам:

- <http://www.nlr.ru/res/inv/ic/rambler.htm>;
- <http://www.nlr.ru/res/inv/ic/yandex.htm>;
- <http://www.nlr.ru/res/inv/ic/apor.htm>.

Задание 6.12. С помощью любой поисковой системы найдите другую поисковую систему.

Тема 6.4* Язык разметки гипертекста HTML

Выполнив задания этой темы, вы изучите:

- ➡ историю создания гипертекста;
- ➡ основы языка HTML;
- ➡ правила создания веб-документов в редакторе HTML.

Путешествуя по Интернету, вы, наверное, не раз задавали себе вопрос: как самому создать веб-страницу? Веб-страницы создаются с помощью *языка разметки гипертекста HTML* (аббревиатура английских слов *Hypertext Markup Language*). Язык HTML был создан для того, чтобы, подготавливая веб-страницу, не надо было думать о различии в программном и аппаратном обеспечении компьютеров, на которых эта страница будет просматриваться. Вопрос создания системно-независимых файлов актуален до сих пор, поскольку компьютерная сеть объединяет принципиально разные компьютеры как с точки зрения программного обеспечения и операционных систем, так и с точки зрения аппаратной части.

Из истории гипертекста

Более 25 лет назад появился новый вид организации данных — гипертекст. Термин «гипертекст» был впервые описан Тэдом Нельсоном в 1969 году. Информация, подготовленная в виде гипертекста, — это электронная информация, работать с которой можно только на компьютере. Невозможно создать ее точную бумажную копию.



Гипертекст — это документ, содержащий ссылки на другие документы.

Чем отличается гипертекст от привычного текста?

Вспомните, как выглядит энциклопедический словарь. Если в тексте, разъясняющем то или иное понятие, встречается слово,

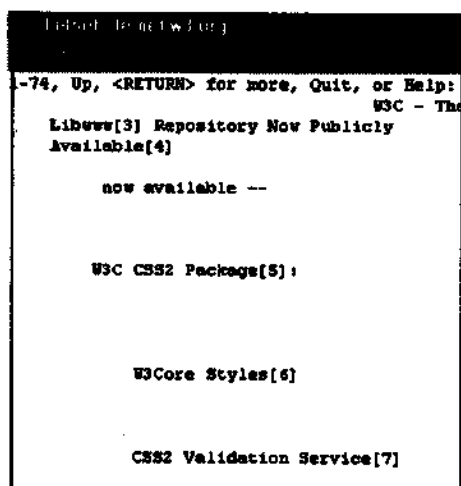


Рис. 6.8. Отображение ссылки в гипертексте цифрой

которое также имеет объяснение в этом словаре, то оно выделено *курсивом* — вы можете найти его и прочитать соответствующий текст. Выделенное слово мы обычно называем *ссылкой*. С помощью ссылки открывается возможность получить дополнительную информацию по выделенному слову. Аналогичным образом организован гипертекст.

«На заре» сети Интернет в гипертексте ссылка выделялась цифрой, следующей за ней в квадратных скобках (рис. 6.8). Чтобы «открыть» ссылку, надо было ввести этот номер — ему соот-

ветствовал определенный информационный ресурс. С такой организацией гипертекста можно встретиться и сейчас, например в играх MUD.

Техника чтения обычного текста состоит в том, что, закончив чтение одной страницы, вы приступаете к следующей. Технология гипертекста позволяет вам свободно переходить со страницы на страницу, следуя заинтересовавшим вас ссылкам, — никакого заранее установленного порядка чтения не существует.

Часто ссылка выделяется на фоне экрана монитора с помощью изменения цвета текста и подчеркивания (рис. 6.9). В таком случае о ней говорят «гиперссылка» или «гиперсвязь».

Гиперссылкой могут считаться слово, фраза или графический элемент, если указатель мыши, установленный на этом объекте, меняет свое изображение.



Гиперссылка — это выделенный объект, связанный с другим файлом и реагирующий на щелчок мыши.

Число гиперссылок в Интернете растет с каждым днем. Прежде существовала серьезная проблема создания гипертекстового документа, так как они создавались вручную. Теперь можно вос-

пользоваться различными специальными средствами для создания структуры гиперссылок.

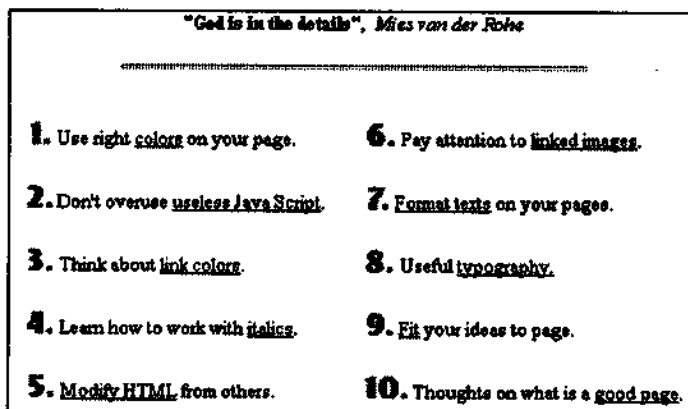


Рис. 6.9. Отображение ссылки в гипертексте цветом или подчеркиванием

Гипертекст с помощью гиперссылок связывает между собой множество документов. Сплетаются данные и самого пользователя, и информационных источников, находящихся на компьютерах, расположенных в разных частях света. Таким образом, один объект соединяется множеством связей с другими объектами. Каждый из этих объектов может представлять собой как обычный, привычный текст, так и гипертекст. Но в таком случае он также, в свою очередь, будет соединен множеством связей с другими объектами.

Основы языка HTML

С помощью языка HTML создается текстовый файл, имеющий расширение .html или .htm, который называется HTML-файлом. Этот файл отображается с помощью браузера в виде веб-страницы. На рис. 6.10 представлены веб-страница и соответствующий ей HTML-файл.

С точки зрения языка HTML, гипертекст — это текст, содержащий специальные разметочные указатели, или так называемые флаги разметки. Обычно используется термин *тег*.

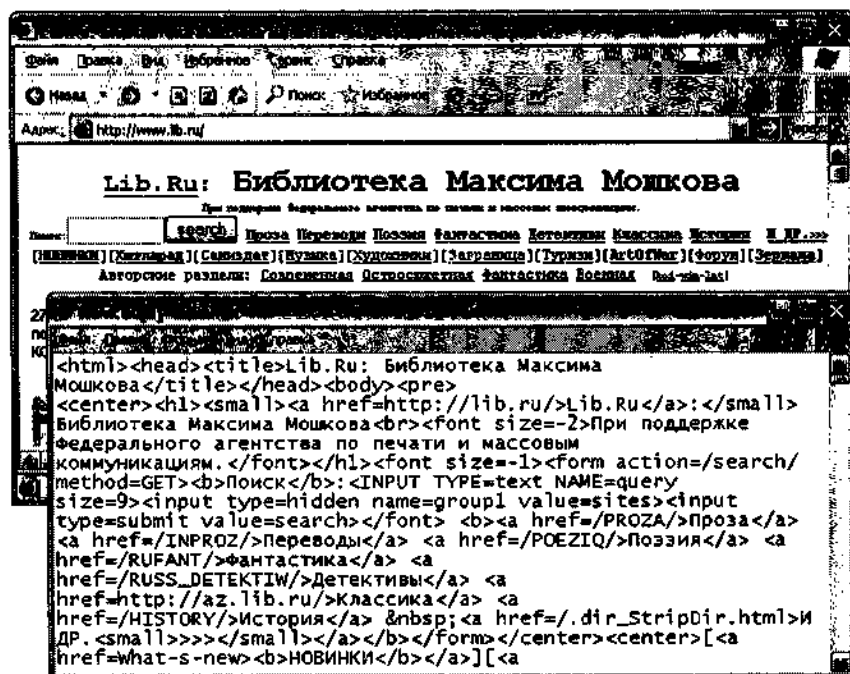


Рис. 6.10. Веб-страница и текст соответствующего ей HTML-файла



Теги — это инструкции браузеру, определяющие способ отображения текста.

Тег всегда начинается с открывающей угловой скобки (знака <) и завершается закрывающей угловой скобкой (знаком >). Заглавные и строчные буквы при написании тегов не различаются.

Существуют два типа тегов — парные и непарные. Парные теги можно сравнить со скобками алгебраического выражения. Парные теги характеризуют текст с того места, где употреблен открывающий тег пары, до того места, где находится закрывающий тег, указывающий на окончание его действия, — такой же тег, но начинающийся с наклонной черты — символа /.

Теги делятся на категории по выполняемым ими функциям — структурные теги (табл. 6.1), теги символов (табл. 6.2), теги форматирования абзацев (табл. 6.3), теги оформления списков (табл. 6.4) и др.

Таблица 6.1. Структурные теги

Название тега	Комментарий
<HTML> </HTML>	Тег, указывающий браузеру, что далее следует HTML-файл. Этот тег как бы обрамляет документ — весь текст должен находиться внутри этого тега. Далее надо разбить документ на две части — заголовок и собственно текст
<HEAD> </HEAD>	Внутри этого тега будет находиться информация о документе, состоящая из нескольких частей
<BODY> </BODY>	Внутри этого тега содержится информация, которая будет отображена в области просмотра браузера. Этот тег может иметь несколько параметров, описывающих цвет фона окна просмотра, рисунок в нем, цвет текста и т. п.

Таблица 6.2. Виды форматирования символов на веб-странице

Название тега	Комментарий
 	Слово, заключенное в этот тег, будет отображено жирным шрифтом
<I> </I>	Слово, заключенное в этот тег, будет отображено курсивом
<U> </U>	Слово, заключенное в этот тег, будет отображено подчеркнутым шрифтом

Таблица 6.3. Теги управления внешним видом веб-страницы

Название тега	Комментарий
<P>	Тег «параграф», отделяет абзацы друг от друга, ставится в конце абзаца. В соответствии с международным стандартом абзацы разделяются пустыми строками и не начинаются с красной строки
 	Следующее за этим тегом содержание HTML-файла будет просто начинаться с новой строки
<HR>	Этот тег указывает на то, что браузер должен отобразить горизонтальную линию, идущую через весь экран. Он может иметь несколько параметров, задающих ширину и цвет линии
<PRE> </PRE>	С помощью этого тега вставляется предварительно отформатированный текст
<H1> </H1>	Тег, определяющий заголовок документа. После буквы H указывается уровень заголовка — от 1 до 5. Чем меньше цифра, тем больше шрифт, отображающий заголовок

Таблица 6.4. Теги оформления списков данных на веб-странице

Название тега	Комментарий
	Этот тег служит для представления абзаца — элемента списка
 	Этот тег служит для представления информации в виде упорядоченного списка, абзацы которого последовательно нумеруются. Каждый абзац списка помечается тегом
 	Этот тег отображает неупорядоченный (маркированный) список в виде последовательности помеченных элементов. Каждый абзац списка помечается тегом
<DL> </DL>	Тег для вывода списка определений или словаря терминов. Каждый элемент такого списка состоит из двух частей — термина и его определения. Термин предваряется тегом <DT>, а определение тегом <DD>

Средства создания HTML-файлов

Существует множество программных продуктов, с помощью которых создают HTML-файлы. Все они делятся на три категории:

- ◆ редакторы;
- ◆ шаблоны;
- ◆ программы-преобразователи.

Использование шаблонов и программ-преобразователей позволяет пользователю не иметь представления о том, что такое теги. Однако язык HTML постоянно развивается и обогащается все новыми возможностями. Соответствующие изменения в программные продукты вносятся намного реже и не в полном объеме. Уже имея начальные знания о тегах, вы сможете постоянно расширять и углублять их, изучая и анализируя понравившиеся веб-страницы. Таким образом, сначала необходимо научиться пользоваться редакторами для создания HTML-файлов, а затем, имея базовые знания, использовать шаблоны и преобразователи.

Существует много редакторов для составления веб-документов. Наиболее известными являются HTML Writer, HTML Assistant, WebEdit, HomeSite.

Задание 6.13*

Ознакомьтесь со средой HTML-редактора HomeSite и научитесь использовать описанные выше теги для создания веб-страниц.

Для выполнения этого задания необходимо, чтобы на компьютере была установлена бесплатно распространяемая программа HomeSite. Ее можно легко найти в Интернете с помощью поисковых систем или, например, на сайте www.design.rsu.ru/club/software.shtml.

Технология работы

1. Для выполнения задания подготовьте рабочий стол Windows: запустите приложения HomeSite и Internet Explorer. С помощью приложения HomeSite вы будете создавать страницу, а с помощью браузера Internet Explorer — просматривать ее.
2. Познакомьтесь с главным окном приложения HomeSite (рис. 6.11). Активизируйте панель QuickTab.
3. Начните редактирование веб-страницы с создания HTML-файла:
 - с помощью команды меню File ▶ New (Файл ▶ Новый) или кнопки на панели инструментов New (Новый) создайте бланк нового документа (рис. 6.12);
 - воспользуйтесь командой меню File ▶ Save as и задайте имя файла, например global.htm, поместив его в предназначенном для этого каталоге.

Структура HTML-файла описывается с помощью тегов, перечисленных в табл. 6.1.

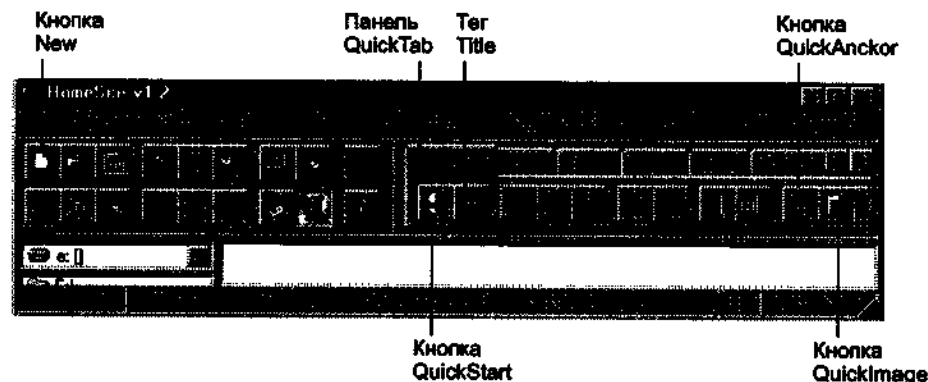


Рис. 6.11. Главное окно HTML-редактора HomeSite

4. Для определения структуры HTML-файла щелкните на кнопке QuickStart (Быстрый старт) (рис. 6.11). На экране появится диалоговое окно, в котором можно не делать никаких изменений и принять установленные по умолчанию цвета. В результате все структурные теги будут внесены в текст файла (рис. 6.12). Сохраните изменения, сделанные в файле.

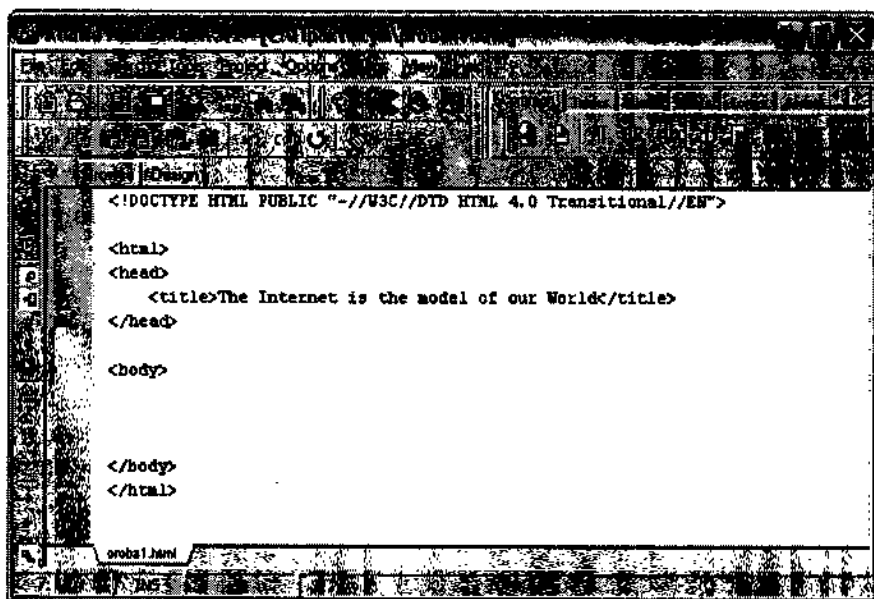


Рис. 6.12. Основные структурные теги HTML-файла

5. Заголовок HTML-файла можно было сделать сразу, в окне QuickStart, но можно это сделать и позже. Для этого необходимо использовать тег `<TITLE>`. Щелкните мышью внутри тега `<HEAD>` и воспользуйтесь кнопкой `Ttl` (см. рис. 6.11). Наберите заголовок. Это не заголовок веб-страницы, а заголовок HTML-файла. Конечно, они должны совпадать по смыслу, однако для того чтобы содержание тега `<TITLE>` отображалось на любом компьютере, необходимо написать его латинскими буквами, например на английском языке. В нашем примере файл называется *The Internet is the model of our World* (рис. 6.13). Сохраните сделанные в файле изменения.

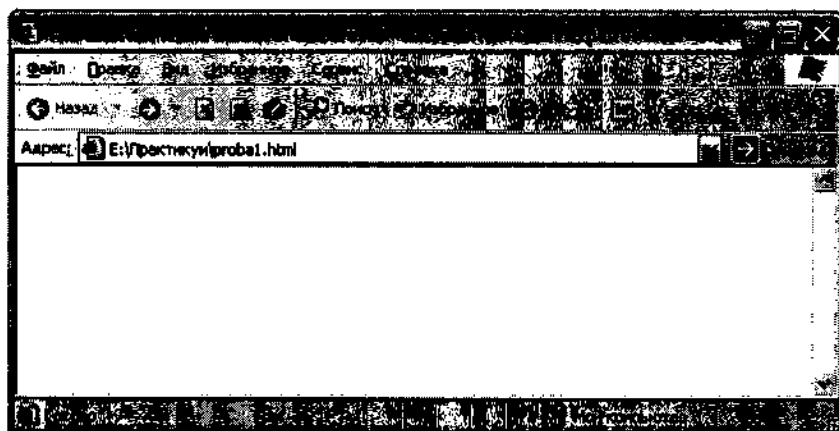


Рис. 6.13. Добавление тега <TITLE> — названия HTML-файла

6. Просмотрите созданную страницу в браузере. Для этого в Internet Explorer в меню Файл выберите пункт Открыть, нажмите кнопку Обзор, перейдите в каталог, где сохранен только что созданный файл, и щелкните мышью на его имени.
7. Научитесь форматировать и оформлять текст веб-страницы. Для этого используйте кнопки Fonts (Шрифты) и Lists (Списки) на панели QuickTab. Эти кнопки позволят вам быстро использовать теги, приведенные в табл. 6.2–6.4.
8. Изучите содержание и теги приведенного ниже текстового файла и продолжите редактирование файла proba1.htm. Можно придумать свой текст.

<HTML>

<HEAD>

<TITLE>The World and The Internet</TITLE>

</HEAD>

<BODY>

<H2>Мир и Интернет</H2><HR>

<i>Система.</i> Что есть система? <i>Система</i> - это некоторое множество <i>элементов</i>, находящихся в отношениях и связях друг с другом, но при этом образующих определенную <i>целостность, единство.</i><HR>

<i>Наш мир.</i> Мир, в котором мы живем, - представляет ли он собой **<i>систему?</i>** Каждый из нас, читающих эти строки, думает о своем. Для кого-то такой **<i>мир</i>** - это его семья. Каждый член этой семьи - это элемент множества, и все вместе они образуют собой целостность, потому что **<i>умеют понимать, договариваться друг с другом.</i>** Может быть, мир, который представляет собой систему, - это наш город? Элементы такого множества разнородны - это и отдельные горожане, и целые семьи, и более крупные общности людей, но систему они представляют собой потому, что образуют определенную целостность - **<i>умеют понимать и договариваться друг с другом</i>**. Как было бы прекрасно, если бы люди разных стран, регионов, национальностей, **<i>умели понимать и договариваться друг с другом</i>**! Тогда весь наш мир - наша **<i>планета Земля</i>** - представлял бы собой **<i>систему</i>**.

</BODY>

</HTML>

9. Посмотрите, как отображается созданный вами файл в браузере (рис. 6.14).

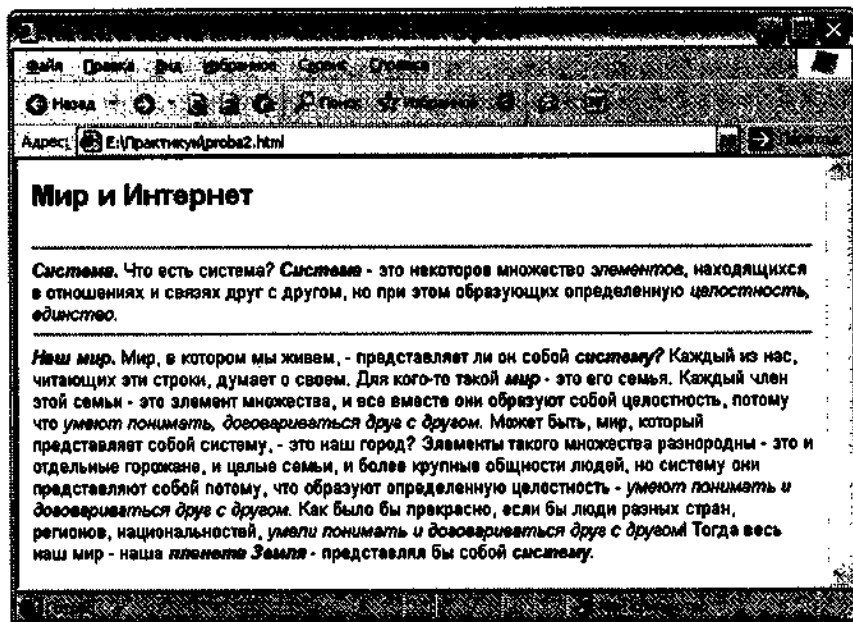


Рис. 6.14. Созданная веб-страница

Задания для самостоятельной работы*

Задание 6.14. Создайте веб-страницу о любимых улицах вашего города.

Задание 6.15. Создайте веб-страницу о своих друзьях и о себе.

Задание 6.16. Создайте веб-страницу на тему из области фантастики.

Задание 6.17. Создайте веб-страницу на свободную тему.

Тема 6.5* **Веб-страница с графическими объектами**

К графическим объектам относится все, что нельзя назвать текстом: отсканированные фотографии и рисунки, изображения, созданные на компьютере, и т. п. Иллюстрации делают привлекательной любую веб-страницу.

Тег, описывающий графический файл на веб-странице, выглядит следующим образом:

```
<IMG SRC="имя файла">
```

Имя файла заключается в кавычки.

Для оформления веб-страницы необходимо сначала подготовить графический файл в любом редакторе или отсканировать изображение, а затем перевести его в формат, удобный для передачи по сети.

Графическая информация не сохраняется непосредственно в HTML-файле. Рисунки и фотографии хранятся каждый в своем файле и передаются на компьютер пользователя по *отдельному запросу браузера*. Для пересылки по компьютерным сетям используются специально предусмотренные форматы графических файлов.

Формат GIF (от англ. *Graphics Interchange Format* — формат взаимодействия графикой) был разработан компанией CompuServe Incorproation для пересылки графических файлов по сети. Файлы с графикой в этом формате имеют расширение .gif. Этот формат идеально подходит для монохромных, 16-цветных, а также 256-цветных рисунков, не содержащих большого количества мелких деталей.

Формат JPEG был разработан для хранения и отображения фотографий объединенной группой экспертов по фотографии. Файлы с графикой в этом формате имеют расширение .jpg или .jpeg.

Задание 6.18*

Разместите на веб-странице графическое изображение (рисунок).

Для выполнения этого задания кроме редактора HTML и браузера нужен любой графический редактор, с помощью которого будет подготовлена графическая иллюстрация. Затем, преобразовав графический объект в формат, допустимый для компьютерных сетей, и пользуясь соответствующими экранными кнопками, вы вставите иллюстрацию в свою веб-страницу.

Технология работы

1. Подготовьте иллюстрацию в графическом редакторе, например Paint Shop Pro (рис. 6.15), и сохраните его в формате GIF или JPEG либо найдите готовый рисунок в одном из указанных форматов.

В нашем примере создан лягушонок, поэтому рисунок имеет имя `frog1.bmp`, а после преобразования, соответственно, — `frog1.gif`.



Рис. 6.15.
Рисунок
в формате BMP

2. Подготовьте рабочий стол: запустите редактор HTML и браузер.
3. Поместите подготовленный графический файл на веб-страницу. Для этого создайте в HomeSite новый HTML-файл или используйте подготовленный заранее файл с текстом. Щелкните мышью в любом месте внутри тега `<BODY>` и нажмите кнопку QuickImage (см. рис. 6.11).
4. В открывшемся окне (рис. 6.16) заполните поля. Обязательным для заполнения является поле Source, в котором указывается имя подготовленного графического файла. Нажмите кнопку ОК, чтобы подтвердить вставку рисунка. HTML-файл примет вид, показанный на рис. 6.17.
5. Сохраните измененный HTML-файл.
6. Посмотрите, как теперь выглядит HTML-файл в браузере.

Задания для самостоятельной работы*

Задание 6.19. Отсканируйте или найдите в Интернете фотографию улицы вашего города и вставьте ее в подготовленную ранее веб-страницу.

Задание 6.20. Вставьте графические иллюстрации в созданные ранее веб-страницы.

Тема 6.6* Веб-страница с гиперссылками

Выполнив задания этой темы, вы научитесь:

⇒ вставлять гиперссылки в веб-страницу.

Гиперссылки делают информационную систему World Wide Web распределенной и тем самым отличают ее от любой другой гипертекстовой информации.

Для вставки в веб-страницу гиперссылки используется тег <A>:

текст гиперссылки

Слова *текст гиперссылки* являются так называемым *якорем* этой гиперссылки. Эти слова при просмотре HTML-страницы в браузере будут выделены цветом и подчеркиванием. Вместо слов *имя файла* в кавычках может быть написано имя файла, находящегося в этой же папке, или может быть указан полный путь к файлу, находящемуся на этом же или на другом компьютере.

Рисунок также может быть оформлен как гиперссылка.

Задание 6.21*

Разместите на веб-странице гиперссылку.

Выполняя это задание, вы научитесь использовать кнопку редактора HTML для вставки гиперссылки на веб-страницу.

Технология работы

1. Подготовьте рабочий стол: запустите и расположите на рабочем столе редактор HTML и браузер.
2. С помощью кнопки QuickStart (см. рис. 6.11) создайте структуру нового HTML-файла и задайте его имя.
3. Щелкните на кнопке QuickAnchor (см. рис. 6.11) для вставки гиперссылки на веб-страницу.
4. В открывшемся окне (рис. 6.18) в первом поле введите следующий URL: <http://www.ict.nsc.ru/rus/docs/html-gd>. Остальные поля можно не заполнять.
5. Обратите внимание на мигающий курсор и введите фразу *Краткое руководство по HTML*. Слова *Краткое руководство*

по *HTML* связывают данную веб-страницу с другой страницей, находящейся на другом компьютере.

6. Сохраните файл и просмотрите результат в браузере. Щелкните мышью на сделанной вами гиперссылке и перейдите на веб-страницу, размещенную на другом компьютере (рис. 6.19).

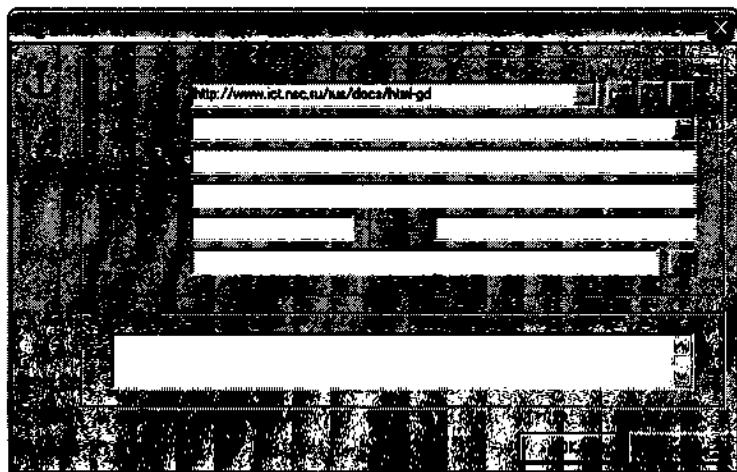


Рис. 6.18. Окно Anchor с заполненным полем гиперссылки

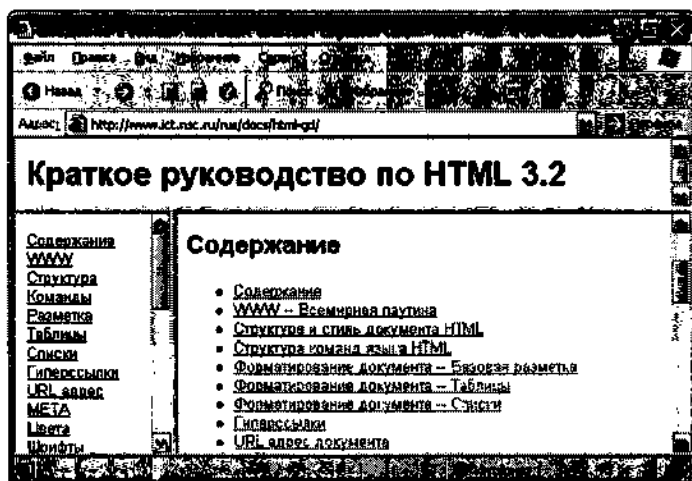


Рис. 6.19. Полезная для вас информация на другом компьютере

7. Повторите пп. 3 и 4 этого задания, введя в открывшемся окне <http://www.darsbydesign.com/index.html>. Обратите внимание, где мигает курсор, и щелкните на кнопке QuickImage. Задайте имя созданного ранее графического файла frog1.gif и щелкните на кнопке OK. В результате этих действий якорем гиперссылки будет являться рисунок. Текст HTML-файла показан на рис. 6.20.
8. Сохраните и просмотрите в браузере созданный HTML-файл. Перейдите по сделанной гиперссылке.

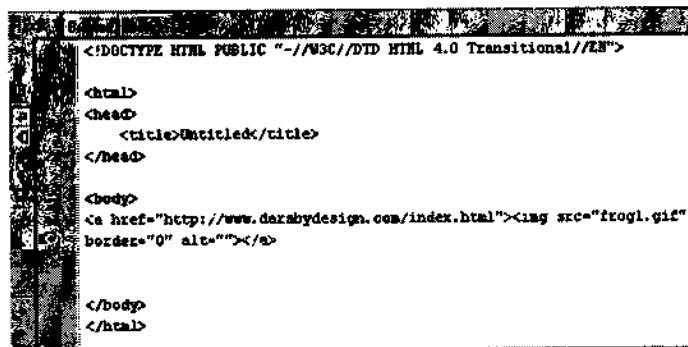


Рис. 6.20. Текст HTML-файла с гиперссылкой, якорем которой служит рисунок

Задания для самостоятельной работы*

Задание 6.22. Найдите с помощью поисковой системы веб-страницу с информацией о своем городе. Подберите слово или фразу, которая могла бы служить якорем для гиперссылки на эту страницу, и создайте гиперссылку на нее на созданной ранее веб-странице о своем городе.

Задание 6.23. Создайте файл, в котором рассказывается, например, о вашем любимом занятии. Сделайте гиперссылку на этот файл с созданной ранее веб-страницы, где вы рассказываете о себе или о своих друзьях.

Задание 6.24. Разместите гиперссылки на созданной ранее веб-странице с текстом на свободную или фантастическую тему.

Тема 6.7 Возможности электронной почты

Выполнив задания этой темы, вы научитесь:

- ⇒ создавать, отправлять и принимать электронные сообщения;
- ⇒ присоединять к электронному сообщению файлы.

Электронная почта, или e-mail (от англ. *electronic mail*), — это самый распространенный вид сетевой коммуникации. Каждый пользователь электронной почты имеет собственный электронный почтовый адрес. Он получает e-mail в той компьютерной системе, в которой регистрируется. Можно зарегистрироваться в нескольких системах и иметь сразу несколько электронных адресов.

В сети Интернет электронные адреса пользователей состоят из двух частей, разделенных символом @. Перед символом @ указывается имя пользователя, а после него — доменное имя почтового сервера, например: info@educom.spb.ru.

Электронное письмо представляет собой обычный текст с подготовленным специальным образом заголовком, в котором содержится основная информация о письме (рис. 6.21).

По электронной почте можно пересылать не только текстовые сообщения, но и файлы любых форматов: документы, графику, программы и др.

Для использования возможностей электронной почты необходима специальная программа. Одной из наиболее удобных почтовых программ является *Outlook Express* — приложение, поставляемое вместе с операционной системой Windows.

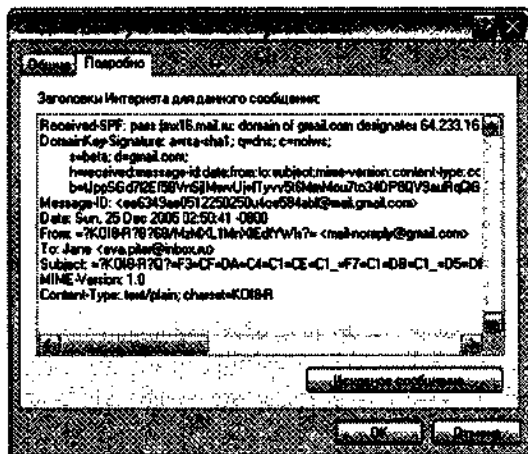


Рис. 6.21. Заголовок электронного письма

Задание 6.25

Изучите основные функции программы электронной почты Outlook Express.

Технология работы

1. Запустите приложение Outlook Express.
2. Познакомьтесь с главным окном Outlook Express (рис. 6.22). Пользуясь стандартными возможностями Windows-приложений, настройте главное окно.

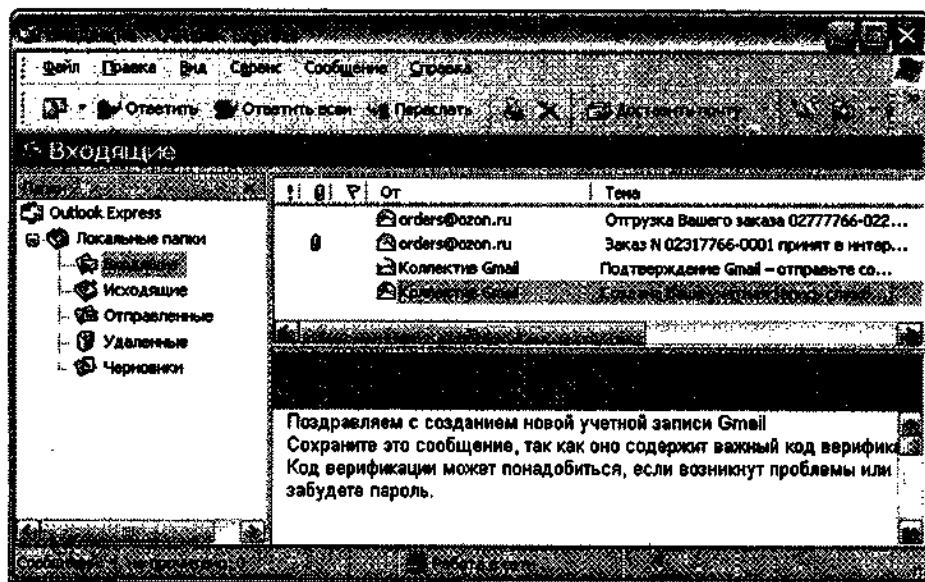


Рис. 6.22. Главное окно приложения Outlook Express

3. Создайте новое сообщение. Для этого щелкните мышью на кнопке Создать сообщение на панели инструментов (рис. 6.22).
4. Заполните поля в открывшемся окне (рис. 6.23). Обязательным для заполнения является поле Кому, содержащее адрес получателя. Если остальные поля останутся пустыми, то будет отправлено так называемое *пустое письмо*.

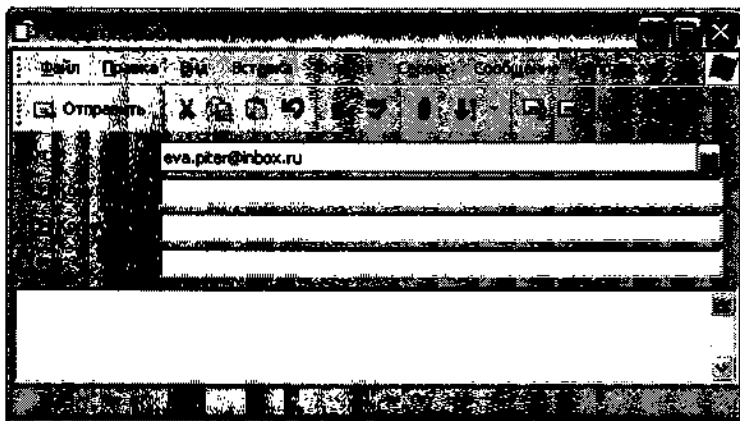


Рис. 6.23. Окно создания электронного сообщения

В поле Тема указывается тема письма (о чем оно), а в рабочем поле записывается собственно текст письма.

В поле Копия указывается адрес электронной почты, по которому должна быть направлена копия письма.

5. Отправьте письмо щелчком на кнопке Отправить. Письмо переместится в специальную папку Исходящие (см. рис. 6.22) и во время ближайшего сеанса связи будет отправлено получателю.
6. Научитесь работать с полученной почтой: читать сообщения, отвечать на них, удалять ненужные и перемещать в другие папки для хранения.
 - Для получения почты необходимо установить соединение с почтовым сервером и щелкнуть мышью на кнопке Доставить почту. Все настройки соединения уже подготовлены системным администратором.
 - Для прочтения письма щелкните мышью на папке Входящие (см. рис. 6.22), а затем — на заголовке письма в списке сообщений.
7. Создайте новое сообщение и присоедините к письму файл. Для этого воспользуйтесь кнопкой Вложить файл в сообщение. В открывшемся окне выберите папку, в котором лежит нужный файл, выделите его и щелкните на кнопке Вложить.

8. Переместите электронное письмо в другую папку:

- найдите сообщение, которое надо переместить, в списке сообщений и щелкните на нем правой кнопкой мыши;
- в появившемся контекстном меню выберите команду Переместить в папку или Копировать в папку;
- в открывшемся диалоговом окне выберите нужную папку (или создайте новую) и щелкните на кнопке ОК.

9. Познакомьтесь с другими возможностями, предоставляемыми контекстным меню.**Задания для самостоятельной работы**

Задание 6.26. Создайте и отправьте письмо с указанием адреса веб-сайта или адреса e-mail.

Задание 6.27. Создайте рисунок в графическом редакторе и заархивируйте его. Напишите письмо в текстовом редакторе. Создайте сообщение без текста, присоедините к нему оба файла и отправьте его.

Часть 2

Основы алгоритмизации и программирования

Раздел 7 Программирование в среде ЛОГО

Раздел 8 Основы алгоритмизации

РАЗДЕЛ 7 Программирование в среде ЛОГО

Тема 7.1	Знакомство с основным инструментарием среды	246
Тема 7.2	Понятие программы	255
Тема 7.3	Последовательный алгоритм	263
Тема 7.4	Циклический алгоритм	267
Тема 7.5	Процедура и модуль	275
Тема 7.6	Процедура с параметрами	282
Тема 7.7	Переменная в алгоритме	290
Тема 7.8	Логика в среде программирования	293



Примеры и задания записаны на русском ЛОГО среды ЛогоМиры 2 (<http://www.int-edu.ru/logo>). Они также действуют в средах Logo-Writer и MSWLogo.

Краткий словарь языка ЛОГО для разных сред приведен в приложении.

Тема 7.1 Знакомство с основным инструментарием среды

Выполнив задания этой темы, вы научитесь:

- ⇒ вводить команды на языке ЛОГО;
- ⇒ измерять и изменять значения параметров объекта;
- ⇒ производить вычисления.

Интерфейс среды программирования

Каждая программа для компьютера разрабатывается на одном из специальных языков — языков программирования. В этом разделе вы познакомитесь с языком ЛОГО. Название ЛОГО образовано от древнегреческого слова *logos*, которое означает *мысль, слово*.

Основу ЛОГО, как и любого другого языка, составляет алфавит. Алфавит ЛОГО включает буквы — русские (кириллица) и английские (латиница), цифры и другие символы, имеющиеся на клавиатуре. Алфавит используется для образования слов.

В любом языке программирования есть специальные *встроенные* слова, значение которых определено заранее. Из этих слов составляются предложения, которые называют *командами*.



Команда — это предложение на языке программирования для указания действия объекту.

Правило записи команды: на первом месте пишется название команды, а затем указываются ее параметры.

Параметр команды — это слово, число или выражение, уточняющее действие команды.

Интерфейс программной среды ЛОГО состоит из трех окон: *рабочего поля, командного центра (поля команд) и листа программ*. После загрузки приложения ЛОГО на экране появляются два окна — командный центр и рабочее поле.

На рабочем поле отображаются графический исполнитель в виде Черепашки и результаты его действий. Черепашка может перемещаться по рабочему полю в разных направлениях и рисовать.

Договоримся называть верхнюю границу рабочего поля Севером, нижнюю границу — Югом, правую — Востоком, а левую — Западом. Обычно после загрузки нового листа Черепашка находится в центре рабочего поля, и голова ее направлена на Север.

Командный центр служит для ввода команд Черепашке на языке ЛОГО.

Команды для графического исполнителя

Черепашка является исполнителем для создания графических объектов на рабочем поле. Посмотрите внимательно команды языка ЛОГО для управления Черепашкой (табл. 7.1). Для некоторых команд принята краткая форма.



По правилам оформления и результату действия команда в краткой форме записи ничем не отличается от команды в полной форме.

Расстояние на рабочем поле измеряется в шагах Черепашки. Один шаг равен одному пикселу. Угол поворота головы Черепашки измеряется в градусах.

Задание 7.1

Установите Черепашку в центр командой `домой`. Введите команду `по`. Используя команду `вперед`, передвиньте Черепашку до верхней границы рабочего поля. Подсчитайте, сколько шагов прошла Черепашка. С помощью команды `назад` передвиньте Черепашку до нижней границы рабочего поля. Подсчитайте количество пройденных Черепашкой шагов. Верните Черепашку в центр поля командой `домой`.

После этих действий на экране появилась вертикальная линия, разделяющая рабочее поле на правую и левую равные части.



Обратите внимание: один шаг Черепашки равен одному пикселу.

Команды `домой`, `по`, `пп`, `сотри` состоят только из названия, а команды `вперед` и `назад` требуют указания параметра — количества шагов. Если в команде, требующей параметра, не указать его, то в командном центре появится сообщение об ошибке.

Таблица 7.1. Графические команды

Название команды и краткая форма	Параметр	Действие исполнителя
вперед, вп	число шагов	Продвигается в направлении головы на указанное число шагов
назад, нд	число шагов	Продвигается в направлении, противоположном направлению головы, на указанное число шагов
нов_х	число шагов от оси X	Перемещается по горизонтали в точку с указанной координатой по оси X (абсциссой)
нов_у	число шагов от оси Y	Перемещается по вертикали в точку с указанной координатой по оси Y (ординатой)
сотри	нет	Стирает рисунок, состояние Черепашки сохраняется
домой	нет	Устанавливается в центр рабочего поля, голова направлена на Север
пп	нет	Поднимает перо
по	номер цвета	Опускает перо
нов_цвет, нц	номер цвета	Принимает цвет с указанным номером
нов_фон, нф	нет	Рабочее поле принимает цвет с указанным номером
сч	нет	Становится невидимым
пч	число градусов	Становится видимым
направо, пр	число градусов	Поворачивается направо относительно направления, заданного головой черепашки
налево, лв	число градусов	Поворачивается налево относительно направления, заданного головой черепашки
нов_курс, нк	число градусов	Поворачивается по часовой стрелке относительно Севера



Запомните правила записи команд:

- ☑ **Правило точного названия.** Название команды должно быть записано без ошибок — точно как в словаре.
- ☑ **Правило одного слова.** Между буквами в названии команды не должно быть пробелов.
- ☑ **Правило пробела.** Если в команде есть параметр, то он отделяется от названия пробелом. Если в одной строке записывается несколько команд, то они разделяются пробелами.

Задание 7.2

Используя команду `нов_х`, определите расстояние от центра до границ видимого рабочего поля по горизонтали.

При выполнении заданий 7.1 и 7.2 Черепашка перемещалась по рабочему полю, но направление ее головы не изменялось.

Задание 7.3

Введите команды, каждую на своей строке:

```
направо 30  
налево 30  
нов_курс 90
```

Черепашка сначала поворачивается направо, затем — налево. После выполнения последней команды голова Черепашки смотрит на Восток. Обратите внимание на то, что центр Черепашки не меняет свое положение на рабочем поле.



Вспомните, что такое угол и как он измеряется. Что такое прямой угол, развернутый угол, полный угол?

Задание 7.4

Верните Черепашку «домой». Наберите в командном центре приведенную ниже последовательность команд (по три в каждой строке). Команды отделите друг от друга пробелом. Какая фигура появилась на рабочем поле? Где оказалась Черепашка?

```
по вперед 50 направо 90  
вперед 50 направо 90 вперед 50  
направо 90 вперед 50 направо 90  
домой
```

Вычисления на компьютере

В среде программирования ЛОГО можно не только рисовать графические объекты, но и вычислять значения математических выражений. Математическое выражение используется как параметр в любой команде, где требуется число. Для изменения порядка вычислений в математическом выражении в языке ЛОГО, как и в математике, применяются круглые скобки.

В среде ЛОГО допустимы целые и дробные числа. Дробная часть числа записывается в десятичном виде и отделяется от целой части точкой.

Обозначения арифметических действий и некоторых математических операций в языке ЛОГО приведены в табл. 7.2.

Таблица 7.2. Основные арифметические действия и математические функции

Действие	Обозначение в ЛОГО	Пример записи	Результат
Сложение	+	5 + 2	7
Вычитание	-	25 - 10	15
Умножение	*	25 * 12	300
Деление	/	121 / 11	11
Получение случайного числа	случайный (сл)	сл 100	число в интервале от 0 до 99
Вычисление остатка от деления	остаток	остаток 100 16	4
Получение целой части числа	целое	целое 13,345	13
Получение числа с противоположным знаком	минус	минус 567	-567
Округление числа	округли	округли 256,56	257

Задание 7.5

Черепашка должна нарисовать отрезок длиной 150 шагов, а затем встать на середину этого отрезка. Запишите последовательность действий Черепашки.

Введите команды:

вп 150 нд 150 / 2

Черепашка окажется точно посередине нарисованного отрезка. Для вывода на экран результата вычисления или текстовых данных используются команды вывода текста (табл. 7.3).

Таблица 7.3. Команды вывода на экран

Название команды	Результат исполнения
пиши	Указанный текст или число выводится в текстовое окно на рабочем поле
покажи	Указанный текст или число выводится в командном центре

Задание 7.6

Выведите в текстовом окне надпись: «Решение примера: $25 + 27$ » и результат вычисления указанной суммы.

Введите команды:

пиши [Решение примера: $25 + 27$]

пиши $25 + 27$

Информационная модель среды ЛОГО

В средах программирования существуют инструменты для измерения значений параметров объектов. В среде ЛОГО эти инструменты называются *датчиками*.

После выполнения команды изменяется определенный параметр рабочего поля или Черепашки. По командам движения изменяется положение Черепашки на рабочем поле и может появиться линия — траектория ее движения. Команды управления пером изменяют положение пера. Команды установки цвета изменяют цвет фона рабочего поля или Черепашки, а команды поворота изменяют направление ее головы. С помощью датчиков можно узнать значения всех этих параметров.

Так как и Черепашка, и рабочее поле являются объектами, для них, как и для любого объекта, можно составить информационные модели (табл. 7.4 и 7.5).

Таблица 7.4. Информационная модель объекта «Черепашка»

Параметр объекта		Название датчика в ЛОГО	Действие объекта
Название	Значения		
Положение на Рабочем поле	[0 0], [100 125]...	x_коор y_коор место	Передвижение по рабочему полю
Направление головы	0, 90, 270...	курс	Поворот
Состояние пера	поднято, опущено	нет	Изменение состояния пера
Цвет	1, 15, голубой...	цвет	Изменение своего цвета
Видимость	видима, невидима	нет	Показаться/спрятаться

Таблица 7.5. Информационная модель объекта «Рабочее поле»

Параметр объекта		Название датчика в ЛОГО	Действие над объектом
Название	Значения		
Цвет фона	0, белый, 15...	фон	Изменение цвета фона
Цвет поля под пером Черепашки	0, черный, 45...	цвет_поля, цп	Закрашивание фигур, ограниченных контуром

Задания для самостоятельной работы

Задание 7.7. Нарисуйте пунктирную тропинку, состоящую из трех отрезков длиной по 20 шагов каждый с расстояниями между ними тоже по 20 шагов. Перепишите команды из командного центра в тетрадь и нарисуйте полученную тропинку (допустим, что сторона одной тетрадной клетки — 10 шагов Черепашки).

Задание 7.8. Наберите в одной строке командного центра указанные ниже команды:

сотри по вп 50 пр 90 вп 50 пр 90 вп 50 пр 90 вп 50 домой

Задание 7.9. Используя команды вперед и направо, установите Черепашку около самой границы рабочего поля в положения, показанные на рис. 7.1. Для каждого варианта ответьте на вопрос: где окажется Черепашка после выполнения команды по вп 100?

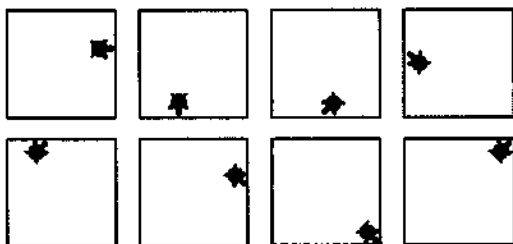


Рис. 7.1. К заданию 7.9

Задание 7.10. Запишите в тетрадь последовательность команд для рисования фигур, изображенных на рис. 7.2.

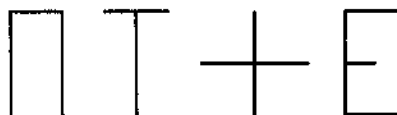


Рис. 7.2. К заданию 7.10

Задание 7.11. Какие графические объекты нарисует Черепашка, выполняя последовательности команд а), б), в), если перед их выполнением она смотрела на Север? Перепишите команды в тетрадь, используя сокращенную форму записи. Нарисуйте эти фигуры в тетради.

- а) по назад 40 направо 90 вперед 80 налево 90
вперед 40 направо 90 назад 80 налево 90 пп домой
- б) сотри по направо 90 вперед 40 налево 90
вперед 40 направо 90 вперед 40 пп домой
- в) сотри по назад 100 направо 90 вперед 40
налево 90 вперед 100 назад 100 направо 90
вперед 10 направо 90 вперед 5 пп домой

Задание 7.12. Придумайте другие последовательности команд, с помощью которых можно нарисовать фигуры, которые получились при выполнении задания 7.11.

Задание 7.13. Черепашка находится в левом нижнем углу рабочего поля. Запишите последовательность команд, выполнив которые, она нарисует лабиринт (рис. 7.3).

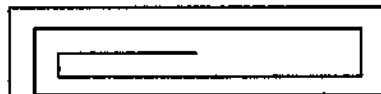


Рис. 7.3. К заданию 7.13

Задание 7.14. Какие команды должна выполнить Черепашка, чтобы на рабочем поле появились узоры, изображенные на рис. 7.4? (Будем считать, что одна клетка на рисунке равна 10 шагам Черепашки.)

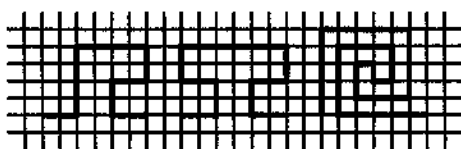


Рис. 7.4. К заданию 7.14

Задание 7.15. Вы знаете, что на почтовых конвертах необходимо указывать индекс получателя письма. Сортировку писем по почтовым индексам выполняют специальные машины. Они должны «узнавать» цифры. Поэтому для изображения этих цифр выработаны определенные правила. Вы видите их на рис. 7.5. Размеры отрезков указаны в шагах Черепашки.

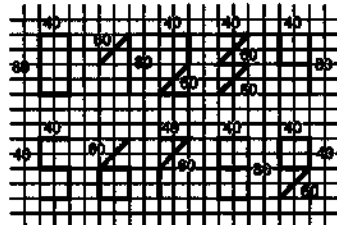


Рис. 7.5. Начертание цифр почтового индекса (к заданию 7.15)

Каждая цифра изображается в прямоугольнике со сторонами 40 на 80 шагов. Будем считать, что перед рисованием цифры Черепашка находится в левом нижнем углу этого прямоугольника.

Запишите в тетрадь последовательности команд, следуя которым, Черепашка могла бы нарисовать каждую цифру почтового индекса.

Задание 7.16. С помощью команды покажи выведите значения следующих дробей: $1/2$; $1/5$; $1/3$; $1/6$; $1/9$. Сколько знаков десятичной дроби получилось в каждом случае?

Задание 7.17. С помощью команды покажи выведите в командный центр значение суммы частного от деления числа 200 на число 8 и разности чисел 50 и 211.

Задание 7.18. Установите Черепашке цвет с номером на 10 большим, чем текущий. Выведите в командный центр значения абсциссы и ординаты Черепашки в текущем положении.

Задание 7.19. Используя команды `нов_x` и `нов_y`, переместите Черепашку по горизонтали и по вертикали на случайные расстояния. Введите в командный центр значения абсциссы и ординаты Черепашки в текущем положении.

Задание 7.20. Опишите любым способом алгоритм вычисления площади прямоугольника.

Тема 7.2 Понятие программы

Выполнив задания этой темы, вы научитесь:

- ⇒ разрабатывать алгоритм для исполнителя;
- ⇒ записывать и редактировать текст алгоритма;
- ⇒ оформлять алгоритм в виде программы;
- ⇒ выполнить программы, записанные на листе программ.

Рассмотрим порядок разработки алгоритма и программы на примере создания графических объектов — цифр 0 и 1.

Задание 7.21

Разработайте алгоритм рисования цифры 0 в соответствии с рис. 7.5.

При выполнении задания рассмотрим три этапа создания алгоритма (см. учебник).

Этап 1. Разработка приближенного варианта алгоритма

Определим цель разработки алгоритма — создание графического объекта в виде прямоугольника высотой 80 шагов и шириной 40 шагов. Начальное положение — левая нижняя вершина. Составим для себя следующий план действий:

1. Установить графический указатель в начальное положение.
2. Нарисовать прямоугольник, начав с левой нижней вершины.
3. Установить графический указатель в исходное положение.

Этап 2. Разработка уточненного варианта алгоритма для среды

Цифру 0 будет рисовать исполнитель Черепашка на рабочем поле. Поэтому следует уточнить разработанный на первом этапе алгоритм для этого исполнителя, используя действия, которые допустимы в среде ЛОГО:

1. Опустить перо.
2. Нарисовать левую сторону прямоугольника.

3. Изменить направление движения — направо на 90° .
4. Нарисовать верхнюю сторону прямоугольника.
5. Изменить направление движения — направо на 90° .
6. Нарисовать правую сторону прямоугольника.
7. Изменить направление движения — направо на 90° .
8. Нарисовать нижнюю сторону прямоугольника.
9. Изменить направление движения — направо на 90° .

Этап 3. Запись алгоритма на языке Исполнителя

Чтобы исполнитель понял этот алгоритм, нужно записать его на понятном исполнителю языке:

```
по
  вперед 80 направо 90
  вперед 40 направо 90
  вперед 80 направо 90
  вперед 40 направо 90
```

После выполнения этих команд на рабочем поле будет нарисован прямоугольник, а Черепашка окажется в исходном положении. Такая последовательность команд для достижения поставленной цели будет *алгоритмом на языке ЛОГО*.

Задание 7.22

Выполните более сложную задачу — нарисуйте цифру 1 в соответствии с рис. 7.5. Этот графический объект должен быть нарисован в прямоугольнике. Составьте алгоритм рисования по этапам так же, как это сделано в задании 7.21.

Этап 1. Разработка приближенного варианта алгоритма

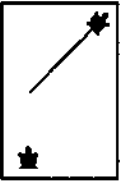
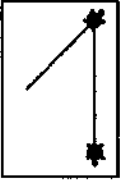
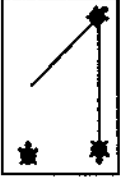
1. Выбрать начальное положение — середину левой стороны прямоугольника.
2. Установить Черепашку в начальное положение.
3. Соединить отрезком начальную точку и правую верхнюю вершину прямоугольника.
4. Нарисовать правую сторону прямоугольника.

Этапы 2 и 3. Разработка уточненного варианта алгоритма для среды. Запись алгоритма на языке исполнителя

Уточним алгоритм для исполнителя, наметив промежуточные цели (подцели). Для этого разделим графический объект на фрагменты и составим алгоритм рисования каждого фрагмента на языке ЛОГО (табл. 7.6).

Действия, перечисленные в первой графе табл. 7.6, уточняют разработанный на первом этапе алгоритм рисования цифры 1 для исполнителя — Черепашки. Во второй графе тот же алгоритм представлен в виде команд, которые можно ввести в командный центр и выполнить.

Таблица 7.6. Алгоритм рисования цифры 1

Действия для достижения подцели	Команды для Черепашки	Значения параметров Черепашки и рабочего поля после выполнения команд	
Нарисовать первый фрагмент	пп вп 40 пр 45 по вп 60	Черепашка в правом верхнем углу прямоугольника, голова повернута на 45° вправо относительно Севера. Нарисован первый фрагмент	
Нарисовать второй фрагмент	пр 135 вп 80	Черепашка в правом нижнем углу прямоугольника, голова смотрит на Юг. Нарисована цифра 1	
Вернуться в начальное положение	пр 90 пп вп 40 пр 90	Черепашка в левом нижнем углу прямоугольника, голова смотрит на Север. Перо поднято. Нарисована цифра 1	



Обратите внимание! Целесообразно команды, которые нужны для достижения промежуточной цели, писать в одной строке. Тогда, сравнивая получившийся результат с намеченным, легко находить и исправлять ошибки.

После окончания сеанса работы в среде ЛОГО информация в командном центре не сохраняется. На следующем сеансе для построения уже известных объектов придется снова набирать команды.

В состав интерфейса среды ЛОГО входит *лист программ*. В этом окне вы набираете и редактируете текст алгоритма, записанный на языке исполнителя в виде последовательности команд. Переход на следующую строку осуществляется нажатием клавиши Enter. С помощью мыши можно выделить фрагмент текста или изменить положение курсора. Фрагмент текста можно перенести, используя промежуточную память (*буфер*), в другое место листа программ или в командный центр.

Содержимое рабочего поля и листа программ сохраняется в файле.

Если алгоритму, записанному на листе программ, дать имя, то этот алгоритм становится *программой*.

Рассмотрим подробнее, как создавать программу.

Сначала вы формируете цель, описывая параметры объекта и создавая информационную модель. Затем разрабатываете алгоритм его рисования по этапам.

На первом этапе разрабатывается примерный алгоритм достижения поставленной цели. На втором этапе этот алгоритм детализируется с учетом среды его будущего выполнения.

На третьем этапе алгоритм подвергается дальнейшему уточнению и полностью записывается на языке программирования с помощью команд. Полученной последовательности команд присваивают имя и записывают ее на листе программ. Теперь для выполнения составленного алгоритма достаточно обратиться к программе по ее имени.

Любой алгоритм предназначен для решения определенной задачи. Каждый новый алгоритм, записанный на листе программ и получивший *имя*, становится *программой*. Таким образом пополняется набор инструментов исполнителя в среде ЛОГО.

Во время подготовки текста программы в окне программ никакие действия на рабочем поле не выполняются.

Подготовив тексты программ, вы закрываете лист программ и переходите в командный центр. При этом проводится предварительная проверка текстов всех программ, расположенных на данном листе программ. При обнаружении ошибок появляется сообщение в командном центре.



Если на листе программ выявлена хотя бы одна ошибка, то программы этого листа выполняться не будут.

Программы создаются не только для описания графических объектов. В программе можно описать алгоритм для демонстрации мультфильма или реализации игры. При выполнении программы создается некоторая новая среда, в которой живут придуманные вами объекты. Программирование позволяет строить с помощью компьютера фантастические виртуальные миры и действовать в них в соответствии с заданными в них законами.



В любой среде программирования существуют строгие правила оформления программы:

- ☒ Программа имеет заголовок, в котором указано ее имя.
- ☒ Программа содержит тело, в котором на языке программирования описан алгоритм.
- ☒ Программа имеет конец, который является признаком завершения текста программы.

В среде программирования ЛОГО эти правила выполняются следующим образом:

Заголовок записывается на отдельной строке. Перед *заголовком* можно писать пояснения к программе. Заголовок начинается со слова *это*, после которого через пробел надо написать *имя* программы.

Имя программы не должно содержать пробелов. Имя программы не должно совпадать ни с одним встроенным словом языка ЛОГО. Одинаковые имена разных программ приводят к путанице. Ниже представлены примеры правильных заголовков:

это кораблик
это единица
это jkl

Так как алгоритм составляется для определенного действия, то ему принято давать имя, говорящее о результате выполнения программы. Например, по имени программы кораблик можно догадаться, что эта программа рисует кораблик, а house — дом. Имя jkl не говорит ни о чем, и понять, для чего предназначена программа с таким именем, невозможно.

Тело программы, то есть алгоритм на языке ЛОГО, обязательно начинается с новой строки. Алгоритм записывается так же, как последовательности команд, вводимые в командном центре. Текст программы должен обязательно заканчиваться словом конец, записанным на новой строке. На листе программ может быть записано много программ. Слово конец отмечает конец каждой программы. В теле программы (до слова конец) не допускается использование нового заголовка (слова это). После слова конец можно написать пояснения и поместить другую программу.

Примеры правильно оформленных программ:

а) программа рисования цифры 0 (см. задание 7.21):

```
это цифра_0  
по вперед 80 направо 90 вперед 40 направо 90  
вперед 80 направо 90 вперед 40 направо 90  
конец
```

б) программа рисования цифры 1 (см. табл. 7.6):

```
это цифра_1  
пп вперед 40 направо 45 по вперед 60  
направо 135 вперед 80  
направо 90 пп вперед 40 направо 90  
конец
```

Записав программу на листе программ, вы дополнили словарь ЛОГО новым словом — именем программы, например цифра_0 или цифра_1. Теперь это слово можно ввести в командном центре или применить его в другой программе как команду.

Для вызова программы в командном центре или в теле другой программы используется только имя этой программы (без слова это).

Если при выполнении новой программы в ее алгоритме обнаруживается незнакомое слово, отсутствующее в словаре и на листе программ, в командном центре появляется сообщение: Не знаю как выполнить <слово>.

Задания для самостоятельной работы

Задание 7.23. Откройте лист программ. Наберите текст программы рисования цифры 0 (цифра_0). Закройте лист программ и убедитесь, что в командном центре не появилось сообщение об ошибке. Если обнаружена ошибка, снова откройте лист программ, найдите и исправьте ошибку и снова закройте лист программ. Если ошибок нет, выполните команду цифра_0, введя это имя в командном центре. На рабочем поле должна появиться цифра 0.

Задание 7.24. Запишите алгоритмы для рисования остальных цифр почтового индекса в виде программ в среде ЛОГО.

Задание 7.25. Составьте алгоритмы рисования фигур, приведенных на рис. 7.6. Для выполнения этого задания подготовьте в тетради рисунки, отметьте на них начальное положение Черепашки и промежуточные цели. Числа на рисунках указывают размер отмеченных углов в градусах. Размеры других углов рассчитайте или подберите сами. Размеры отрезков произвольны.

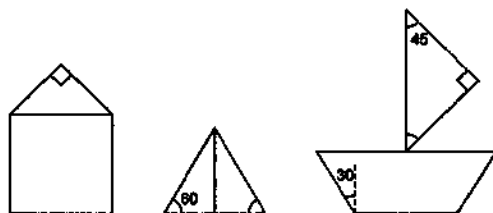


Рис. 7.6. К заданию 7.25

Задание 7.26. Придумайте и нарисуйте в тетради лабиринт. Опишите алгоритм рисования этого лабиринта для Черепашки. Затем, задав начальное положение Черепашки — у входа, «объясните» ей, как проходить лабиринт.

Задание 7.27. Вы должны объяснить человеку, не знающему русского языка, как приготовить яичницу. Изобразите этот алгоритм в тетради с помощью рисунков, расположенных в определенном порядке. Может ли Черепашка выполнить ваш алгоритм?

Задание 7.28. Опишите двумя способами (на русском языке и на языке математических символов) алгоритм получения суммы частного от деления числа 200 на число 8 и разности чисел 211 и 50.

Задание 7.29. Запишите на листе программ программу рисования буквы «П». Введите в командном центре команду буква_п.

это буква_п

по вл 80 пр 90 вл 40 пр 90 вл 80 пр 180 пп

конец

Задание 7.30. Запишите на листе программ программу рисования бантика:

```
это бантик  
по вп 40 пр 120 вп 80 лв 120  
вп 40 лв 120 вп 80 пр 120 пп  
конец
```

Введите в командном центре команду бантик.

Задание 7.31. В теле программы песочные_часы используется записанная ранее программа бантик:

```
это песочные_часы  
нк 270 бантик нк 0  
конец
```

Записав текст этой программы, закройте лист программ и выполните команду песочные_часы.

Задание 7.32. Опишите объект «прямоугольник» со сторонами 100 и 60 шагов и составьте программу для его построения.

Задание 7.33. Найдите и объясните ошибки, допущенные при написании программ:

- а) это ерунда вп 10 нд 10 конец
- б) это пп
по буква_п вп 40 буква_п конец
- в) это двойной бант
по бантик вп 10 бантик пп
конец

Задание 7.34. Изучите объекты, представленные на рис. 7.7. Найдите и назовите элемент, который встречается во всех трех объектах. Составьте описание каждого объекта и алгоритм его построения. Дайте имена полученным алгоритмам и запишите соответствующие программы на лист программ. Проверьте новые команды.

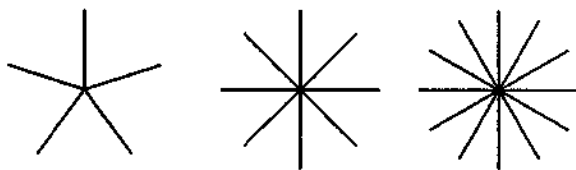


Рис. 7.7. К заданию 7.34

Тема 7.3 Последовательный алгоритм

Выполнив задания этой темы, вы научитесь:

- ⇒ составлять информационную модель графического объекта;
- ⇒ разрабатывать и записывать последовательный алгоритм.

Задание 7.35

Составьте алгоритм построения домика (рис. 7.8) с условием, что Черепашка не должна проходить дважды по одной и той же линии. Начальное положение выберите самостоятельно. Стороны квадрата и треугольника равны.

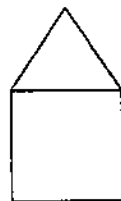


Рис. 7.8. К заданию 7.35

Этап 1. Разработка приближенного варианта алгоритма

Домик состоит из крыши и основания. Основание представляет собой квадрат, крыша — равносторонний треугольник. Низ крыши и верх основания — один и тот же отрезок. Дважды его рисовать нельзя. Составим примерный алгоритм рисования домика:

1. Нарисовать квадрат, начиная с верхней вершины, и вернуться в ту же точку.
2. Нарисовать крышу — оставшиеся стороны треугольника.

Этап 2. Разработка уточненного варианта алгоритма для среды

Опишем объект, который должна нарисовать Черепашка, в виде информационной модели (табл. 7.7).

Необходимо задать начальное и конечное состояния Черепашки и рабочего поля. Начальное положение Черепашки обычно выбирается относительно объекта, который она будет рисовать. В соответствии с выбранным алгоритмом это должна быть одна

из верхних вершин квадрата, допустим, левая. Голова Черепашки направлена вдоль основания крыши. Перо должно быть опущено. Конечное положение Черепашки не будем оговаривать.

Таблица 7.7. Информационная модель объекта «Домик»

Параметр объекта		Действие объекта
Название	Значение	
Длина стороны	50 шагов	Рисовать
Число сторон в основании	4	Стирать
Соотношение сторон в основании	Все стороны равны	
Величина угла между сторонами основания	90°	
Число сторон у крыши	2	
Соотношение сторон крыши	Все стороны равны	
Величина угла между сторонами крыши	60°	
Цвет контура домика	Текущий	

Для этой задачи начальное состояние рабочего поля не важно. Но после выполнения алгоритма на нем должен появиться домик.

Этап 3. Запись алгоритма на языке исполнителя

Представим алгоритм построения домика на языке для исполнителя — Черепашки (табл. 7.8).

Все команды, приведенные во втором столбце табл. 7.8, выполняются в порядке записи друг за другом. Такой алгоритм называется *линейным*, или *последовательным*. На основании алгоритма, полученного во втором столбце табл. 7.8, напомним программу:

```

это домик
  вп 50 пр 90  вп 50 пр 90
  вп 50 пр 90  вп 50
  пр 30  вп 50
  пр 120  вп 50
конец

```

Таблица 7.8. Алгоритм рисования домика в ЛОГО

Действие для достижения подцели	Команды для Черепашки	Значения параметров Черепашки и рабочего поля
Нарисовать основание	вп 50 пр 90 вп 50 пр 90 вп 50 пр 90 вп 50	Нарисован квадрат справа вниз от начального положения Черепашки. Черепашка в верхней левой вершине нарисованного квадрата, голова направлена на Север
Нарисовать крышу	пр 30 вп 50 пр 120 вп 50	Нарисована крыша над основанием. Черепашка в правой верхней вершине, голова направлена вдоль стороны крыши

Получится ли домик, изображенный на рис. 7.8, если изменить порядок выполнения алгоритма: сначала нарисовать крышу, а затем — основание?

Задания для самостоятельной работы

Задание 7.36. Дан прямоугольник со сторонами 30 и 60 шагов. Составьте программу его рисования на рабочем поле.

Задание 7.37. Составьте программу вычисления площади прямоугольника со сторонами 30 и 60 шагов. Какую команду вы используете для вывода полученного значения в командный центр?

Задание 7.38. Какую фигуру нарисует Черепашка после выполнения следующих действий:

- вперед на 30 шагов,
- влево на 60° ,
- назад на 30 шагов,
- влево на 60° ,
- вперед на 30 шагов?

На какой угол должна повернуться Черепашка, чтобы, закончив выполнение этого алгоритма, вернуться в начальное положение? Напишите выражение для вычисления угла поворота Черепашки, возвращающего ее в исходное положение.

Задание 7.39. Нарисуйте в тетради квадрат со стороной 100 шагов (10 клеток) и обозначьте начальное положение Черепашки — в левом нижнем углу этого квадрата, голова ее направлена на Север. Подготовьте таблицу по образцу табл. 7.7, в которой укажите параметры графического объекта «Квадрат». Опишите начальное и конечное состояния Черепашки и рабочего поля. Запишите программу построения квадрата на рабочем поле в таблице по образцу табл. 7.8. В результате выполнения программы Черепашка должна вернуться в начальное положение.

Тема 7.4 Циклический алгоритм

Выполнив задания этой темы, вы научитесь:

- ⇒ записывать циклический алгоритм;
- ⇒ рисовать равносторонний треугольник;
- ⇒ составлять алгоритм построения любого правильного многоугольника и окружности;
- ⇒ составлять алгоритм построения объектов с повторяющимися элементами.

Способ записи циклического алгоритма

Задание 7.40

Нарисовать несколько одинаковых квадратов в разных местах рабочего поля, используя программу, созданную в задании 7.39 (тема 7.3).

Выполните алгоритм:

1. Переместить Черепашку в центр рабочего поля. Записать в командном центре команду вызова программы рисования квадрата. Ввести эту команду, нажав клавишу Enter. На рабочем поле будет нарисован квадрат. Черепашка вернется в начальное положение — в левый нижний угол квадрата.
2. Переместить Черепашку в другую точку рабочего поля и снова выполнить ту же команду. На экране появится еще один квадрат тех же размеров, а Черепашка окажется в левом нижнем углу этого квадрата.
3. Переместить Черепашку в новое начальное положение и повернуть голову направо на 30° . Затем выполнить ту же команду. На экране снова будет нарисован квадрат тех же размеров, но развернутый относительно оси Y вправо на 30° . Черепашка окажется в левом нижнем углу этого квадрата.
4. Конец алгоритма

Во всех пунктах задания 7.40 получался один и тот же результат при выполнении последовательного (линейного) алгоритма:

вп 100 пр 90
вп 100 пр 90
вп 100 пр 90
вп 100 пр 90

Подчеркните в этом алгоритме одинаковые команды: команду `вп 100` — одинарной чертой, команду `пр 90` — двойной чертой. Вы видите, что пара команд `вп 100 пр 90` повторяется 4 раза.

Во всех языках программирования существует способ сокращенной записи повторяющихся действий — *циклический алгоритм*.



В среде программирования циклический алгоритм состоит из тела цикла и условия:

- ☒ тело цикла содержит команды, которые должны повторяться;
- ☒ условие указывает, сколько раз или до каких пор повторяются команды тела цикла.

В языке программирования ЛОГО для записи циклического алгоритма применяется команда `повтори`. В этой команде два параметра: первый задает количество повторений (*условие цикла*), второй — список команд, которые должны повторяться (*тело цикла*). С помощью команды `повтори` алгоритм построения квадрата можно записать короче:

`повтори 4 [вп 100 пр 90]`

Здесь число 4 — *условие цикла*, а команды `вп 100 пр 90` — *тело цикла*. Тело цикла всегда заключается в квадратные скобки.

Задание 7.41

Введите команду:

`повтори 6 [лв 45 вп 30 пр 90 вп 30 лв 45]`

Черепашка нарисует зигзагообразную линию с шестью зубцами. Алгоритм создания одного зубца описан в теле цикла.

Задание 7.42

Введите команду:

`повтори 4 [повтори 4 [вп 40 пр 90] лв 90]`

Здесь в тело одного цикла вложен второй цикл.

Вложенный цикл описывает алгоритм рисования квадрата. В соответствии с внешним циклом Черепашка 4 раза рисует квадрат, поворачиваясь после рисования каждого на 90° . В результате вы увидите на экране большой квадрат, составленный из четырех маленьких. Длину стороны большого квадрата легко подсчитать.

Для построения многих геометрических объектов могут применяться циклические алгоритмы. Среди таких объектов особое место занимают правильные многоугольники. Правильным называется выпуклый многоугольник, у которого все углы и стороны равны. Правильными многоугольниками являются, например, равносторонний треугольник, квадрат. В геометрии, чтобы нарисовать многоугольник с большим количеством сторон, обычно используют такой способ: разбивают окружность на сектора с одинаковыми углами и соединяют точки на окружности отрезками. Примеры правильного пятиугольника и правильного восьмиугольника приведены на рис. 7.9.

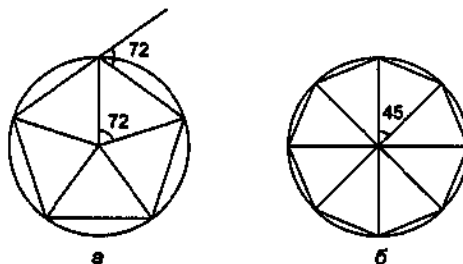


Рис. 7.9. Построение правильного пятиугольника (а) и правильного восьмиугольника (б)

Задание 7.43

Введите команды:

вп 50 пр 60

вп 50 пр 60

вп 50 пр 60

Черепашка нарисовала ломаную линию из трех равных отрезков. Какие параметры в указанных командах нужно изменить, чтобы начало и конец ломаной линии соединились, образовав правильный треугольник? Подумайте, почему предложенная последовательность команд не привела к такому результату.

Задание 7.44

Постройте правильный треугольник.

Модель объекта показана на рис. 7.10. На ней отмечены начальное и конечное положения Черепашки и углы между направлениями сторон треугольника. Как видим, Черепашка должна поворачиваться на угол, смежный с углом треугольника. Ей следует повернуться три раза, чтобы получилось три угла.

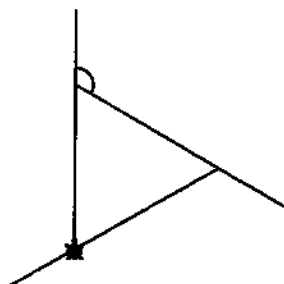


Рис. 7.10. К заданию 7.44

У треугольника с равными сторонами равны и все углы, следовательно, равны и их смежные углы. Поскольку Черепашка возвращается в начальное положение, значит, она делает полный оборот, то есть сумма углов поворотов Черепашки составляет 360° . Легко подсчитать, на какой угол надо каждый раз поворачиваться Черепашке, чтобы нарисовать правильный треугольник.

Графические объекты с повторяющимися элементами

Задание 7.45

Постройте правильный шестиугольник.

Информационная модель правильного шестиугольника со стороной 50 шагов представлена в табл. 7.9.

Таблица 7.9. Информационная модель объекта «Правильный шестиугольник»

Параметры объекта		Действия объекта
Название	Значение	
Длина стороны	50 шагов	Рисовать
Число сторон	6	Стирать
Соотношение сторон	Все стороны равны	
Соотношение углов	Все углы равны	
Величина угла, смежного с углом между сторонами	$360/6$ градусов	
Цвет контура	Текущий	

Начальное состояние Черепашки — текущее. Ее конечное состояние должно совпадать с начальным. Начальное состояние рабочего поля — текущее. После выполнения алгоритма на рабочем поле должен быть нарисован шестиугольник со стороной 50 пикселей.

Составим алгоритм рисования шестиугольника (табл. 7.10).

Таблица 7.10. Алгоритм рисования правильного шестиугольника

Действия для достижения цели	Команды для Черепашки	Значения параметров Черепашки и рабочего поля
Повторить 6 раз следующие действия: вперед на 50 шагов, повернуть голову направо на угол, равный $360/6$ градусов	повтори 6 [вп 50 пр $360 / 6$]	Нарисован шестиугольник справа от начального положения Черепашки. Черепашка вернулась в начальное положение

Обобщим предложенный метод для построения правильного многоугольника с любым количеством сторон. Обозначим буквой A длину стороны многоугольника, буквой N — количество углов многоугольника. Такой объект описан в табл. 7.11, а алгоритм его построения приведен в табл. 7.12.

Таблица 7.11. Информационная модель объекта «Правильный многоугольник»

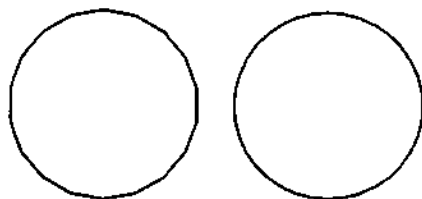
Параметры объекта		Действия объекта
Название	Значение	
Длина стороны	A пикселей	Рисовать
Число сторон	N	Стирать
Соотношение сторон	Все стороны равны	
Соотношение углов	Все углы равны	
Величина угла, внешнего к углу между сторонами	$360/N$ градусов	
Цвет контура	Текущий	

Таблица 7.12. Алгоритм рисования правильного многоугольника

Действия для достижения цели	Команды для Черепашки	Значения параметров Черепашки и рабочего поля
Повторить N раз: вперед на A шагов, повернуть голову направо на угол, равный $360/N$	повтори N [вп A пр $360 / N$]	Нарисован многоугольник справа от начального положения Черепашки. Черепашка вернулась в начальное положение

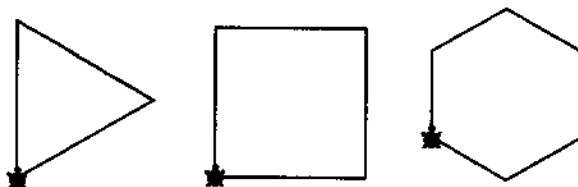
В командах на языке ЛОГО числовые параметры заменены буквами, как это делается в алгебре. Вы можете использовать предложенный алгоритм для построения любого правильного многоугольника с заданной длиной стороны, подставляя вместо букв конкретные числа.

Посмотрите на рис. 7.11. На нем изображены правильные 18-угольник со стороной 10 шагов и 36-угольник со стороной 5 шагов. На глаз их почти не отличить от окружности. Чем больше сторон у правильного многоугольника, тем больше он будет похож на окружность.

**Рис. 7.11.** Правильные 18-угольник (слева) и 36-угольник (справа)

Задания для самостоятельной работы

Задание 7.46. Постройте правильные многоугольники (рис. 7.12) с помощью команды повтори. Черепашка должна всегда возвращаться в начальное положение.

**Рис. 7.12.** К заданию 7.46

Задание 7.47. Найдите ошибки, допущенные при написании команд:

- а) [вп 20 нд 20 пр 90 вп 20 лв 90] повтори 5
- б) повтори 20 раз [вп 10 жди 10]
- в) повтори 100 (пр 90 вп 20 лв 90 вп 20)

Задание 7.48. Что нарисует Черепашка, выполнив следующие команды:

- а) по повтори 10 [вп 20 нд 20 пр 90 вп 20 лв 90] пп
- б) по повтори 15 [пр 90 вп 20 лв 90 вп 20] пп
- в) по повтори 3 [вп 50 пр 60] пп

Задание 7.49. На какой угол должна поворачиваться Черепашка, начертив очередную сторону: а) квадрата; б) треугольника; в) шестиугольника?

Задание 7.50. На какой угол повернулась Черепашка, придя к начальному положению после рисования: а) квадрата; б) треугольника; в) шестиугольника?

Задание 7.51. Запишите в командном центре команды для построения правильного 9-угольника с длиной стороны 10 шагов (см. табл. 7.12) и выполните их.

Задание 7.52. Изменив команды, написанные для построения 9-угольника, нарисуйте следующие многоугольники:

- а) 15-угольник со стороной 10 шагов;
- б) 18-угольник со стороной 10 шагов;
- в) 36-угольник со стороной 5 шагов.

Задание 7.53. Нарисуйте стебель и 16 одинаковых игольчатых лепестков (рис. 7.13).

Задание 7.54. Нарисуйте пучок, состоящий из 30 расходящихся лучей длиной по 60 шагов. Угол между крайними лучами — 60° (рис. 7.14).

Задание 7.55. Нарисуйте полумесяц, являющийся половиной 360-угольника со стороной 1 шаг (рис. 7.15).

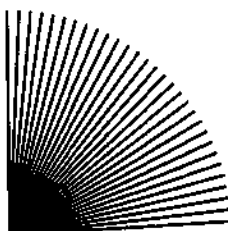
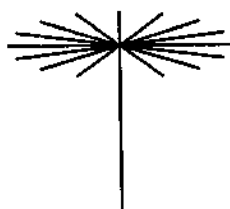


Рис. 7.13. К заданию 7.53 **Рис. 7.14.** К заданию 7.54 **Рис. 7.15.** К заданию 7.55

Задание 7.56. Подсчитайте путь, который проходит Черепашка, когда рисует:

- а) квадрат со стороной 30 шагов;
- б) треугольник со стороной 30 шагов;
- в) шестиугольник со стороной 30 шагов.

Задание 7.57. Подсчитайте путь, который проходит Черепашка, когда рисует:

- а) 18-угольник со стороной 10 шагов;
- б) 36-угольник со стороной 5 шагов;
- в) 360-угольник со стороной 0,5 шага.

Задание 7.58. Напишите программу квадрат для построения квадрата со стороной 20 шагов. Составьте информационные модели объектов, представленных на рис. 7.16. Напишите программы для построения этих объектов. В программах используйте команду квадрат.

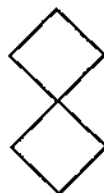
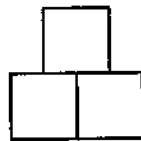
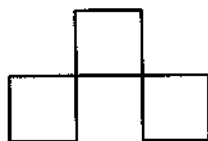
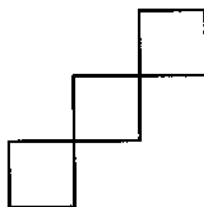


Рис. 7.16. К заданию 7.58

Тема 7.5 Процедура и модуль

Выполнив задания этой темы, вы научитесь:

- ⇒ использовать вспомогательные алгоритмы для создания сложных объектов;
- ⇒ выполнять правило сохранения состояния при разработке процедуры;
- ⇒ составлять модули, используя вспомогательные программы — процедуры.

Встречаясь с новым объектом, системой или процессом, человек всегда стремится выделить знакомые для себя элементы и определить взаимоотношения между ними. Подобный процесс называют *анализом*.

Рассмотрим графические объекты, изображенные на рис. 7.16. Все они составлены из квадратов со стороной 30 пикселей. Составим информационную модель композиции «Лесенка из квадратов» (табл. 7.13), опираясь на разработанный алгоритм построения квадрата и соответствующую ему программу квадрата.

Таблица 7.13. Информационная модель объекта «Лесенка из квадратов»

Параметры объекта		Действие объекта
Название	Значение	
Количество квадратов	3	Рисовать
Соотношение квадратов	Все одинаковые	Стирать
Соотношение вершин квадратов	Левая нижняя вершина следующего квадрата совпадает с правой верхней вершиной предыдущего	
Соотношение углов между квадратами	Все углы равны	
Цвет контура	Текущий	

Начальное состояние Черепашки — в левой нижней вершине нижнего квадрата, голова направлена на Север. Конечное со-

стояние Черепашки совпадает с начальным. Начальное состояние рабочего поля — очищено от рисунков. После выполнения алгоритма на рабочем поле должна быть нарисована лесенка из квадратов.

Алгоритм рисования лесенки из квадратов приведен в табл. 7.14. Надо выполнить следующие действия: нарисовать квадрат, передвинуть Черепашку в правый верхний угол нарисованного квадрата. Эти действия повторяются столько раз, сколько квадратов в лесенке. Программа рисования лесенки будет следующей:

это лесенка

повтори 3 [квадрат вп 30 пр 90 вп 30 лв 90]

пп повтори 3 [пр 90 нд 30 лв 90 нд 30]

конец

Таблица 7.14. Алгоритм рисования лесенки из квадратов

Действия	Команды для Черепашки	Значения параметров Черепашки и рабочего поля
Повторить 3 раза действия: нарисовать квадрат, перейти в правую верхнюю вершину	повтори 3 [квадрат вп 30 пр 90 вп 30 лв 90]	Нарисована лесенка из квадратов. Черепашка в правой верхней вершине последнего квадрата
Вернуться в начальное состояние	повтори 3 [пр 90 нд 30 лв 90 нд 30]	Черепашка в левой нижней вершине первого квадрата, голова направлена на Север

Особенностью программы лесенка является использование вспомогательной программы рисования квадрата, которая вызывается командой квадрат. В средах программирования вспомогательные программы обычно называют *процедурами*.



Процедура — это вспомогательная программа, которая может вызываться из другой программы.

Различают встроенные и пользовательские процедуры. Все команды ЛОГО обращаются к встроенным процедурам. вспомо-

гательные программы, написанные вами и помещенные на лист программ, являются пользовательскими процедурами.

Так, программа квадрат является пользовательской процедурой. Применяя процедуру, вы должны точно представлять, в каком состоянии находится Черепашка перед ее выполнением и в каком состоянии она окажется после ее выполнения. От этого зависит выполнение всей программы, где используется пользовательская процедура. Поэтому при создании процедуры надо соблюдать *правило сохранения начального состояния исполнителя*:



Процедура перед своим завершением должна восстановить исходные значения параметров исполнителя.

На рис. 7.17 представлены объекты, которые получились из сложного объекта — лесенки. Объект, который составляется из более простых объектов и может использоваться для создания более сложных объектов, называют *модулем*. В современном производстве модули (или блоки) используются очень широко, например при проектировании домов, при изготовлении корпусной мебели и пр. Детский конструктор предоставляет возможность детям воплощать свои строительные фантазии, используя готовые модули. Компьютер также собирается из отдельных модулей, которые легко заменяются.

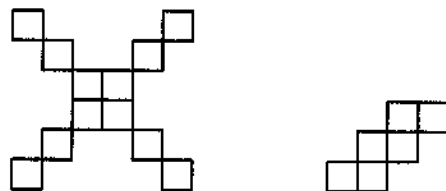


Рис. 7.17. Композиция из нескольких лесенок



Программный модуль — это совокупность процедур, связанных определенными правилами.

Программу, которая вызывает пользовательские процедуры, тоже называют модулем. Программный модуль применяется для создания более сложных объектов, в отличие от процедуры, которая используется для создания элементарных объектов. В таком случае программу рисования лесенки, созданную с помощью процедуры квадрат, можно воспринимать как программный модуль и использовать для конструирования всевозможных композиций (рис. 7.17).

Одни и те же процедуры могут применяться в разных модулях. В свою очередь, модули могут применяться как для создания отдельных объектов, так и для создания более сложных модулей (рис. 7.18). Такой подход к программированию называется *процедурным*.

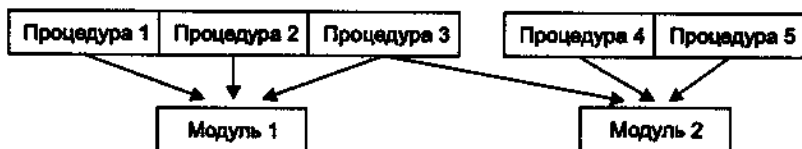


Рис. 7.18. Построение модулей на основе процедур

Рассмотрим пример создания модуля четырехлистника, который используется для рисования цветка (рис. 7.19).

Сначала надо провести анализ структуры графического объекта «Цветок»:

- ◆ цветок состоит из 16 лепестков;
- ◆ каждый лепесток образован двумя дугами;
- ◆ элементарным объектом является дуга, равная четверти окружности.

На основании анализа предлагаем технологию создания программы рисования цветка:

- ◆ написать процедуру для рисования дуги;
- ◆ учитывая эту процедуру, разработать модуль, рисующий четырехлистник в соответствии с алгоритмом, приведенным на рис. 7.20;
- ◆ используя модуль четырехлистника, нарисовать разнообразные цветы с любым количеством лепестков, кратным четырем.

Например, чтобы нарисовать цветок, изображенный на рис. 7.19, нужно 4 раза выполнить следующие действия:

1. Нарисовать четырехлистник.
2. Повернуть графический указатель вправо на $22,5^\circ$.

Для получения цветка с большим количеством лепестков нужно увеличить число повторений и уменьшить угол поворота Черепашки.

Создание программ путем объединения модулей называется *модульным программированием*.

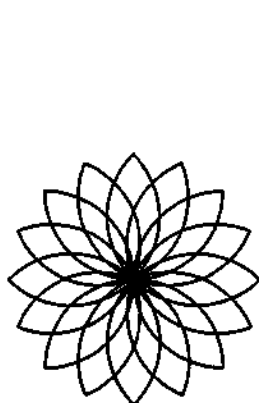


Рис. 7.19. Графический объект «Цветок»

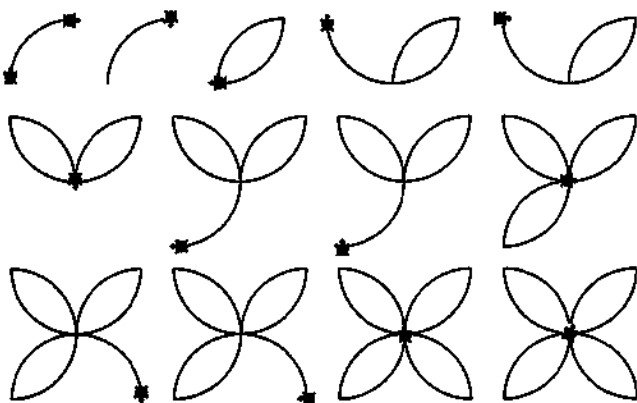


Рис. 7.20. Алгоритм создания модуля для рисования четырехлистника

Задания для самостоятельной работы

Задание 7.59. Составьте процедуру рисования элементарного объекта (дуги) и программный модуль для рисования четырехлистника для рис. 7.19. Напишите программы для рисования разных цветков, используя модуль четырехлистника.

Задание 7.60. Выделите в каждой фигуре, изображенной на рис. 7.21, свой повторяющийся элемент. Напишите процедуру для их построения. Составьте программы для рисования каждой фигуры, используя написанные процедуры.

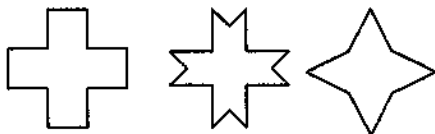


Рис. 7.21. К заданию 7.60

Задание 7.61. Составьте программы для рисования снежинок (рис. 7.22). В чем сходство и различие этих программ?



Рис. 7.22. К заданию 7.61

Задание 7.62. Составьте программы для рисования звездочек (рис. 7.23). В чем сходство и различие этих программ?



Рис. 7.23. К заданию 7.62

Задание 7.63. Зачет: конкурс по модульному программированию.

а) Даны процедуры фрагм1 и фрагм2:

```
это фрагм1
  вп 12 пр 90 вп 12
конец
это фрагм2
  вп 12 лв 90 вп 12
конец
```

Используя только процедуры фрагм1 и фрагм2, команды поворота и повтори, построить фигуры, приведенные на рис. 7.24.

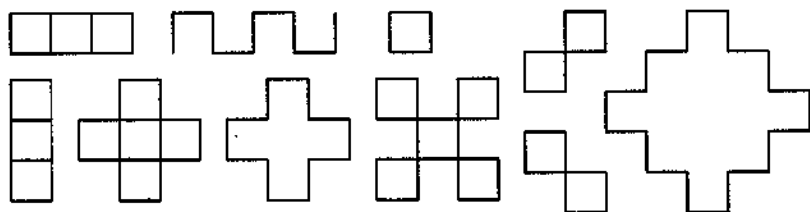


Рис. 7.24. К заданию 7.63, а

б) В Греции есть река Меандр со столь извилистыми берегами, что трудно понять, в какой стороне начало реки, а в какой — конец. Такое же название носит классический греческий орнамент, представляющий собой извилистую кривую или ломаную линию (рис. 7.25). Составьте алгоритмы и программы для рисования меандров.

в) Составьте алгоритмы и программы рисования фигур, предложенных на рис. 7.26, выделив в них элементарные объекты и написав процедуры для их построения. Доработайте свои алгоритмы так, чтобы получились раскрашенные фигуры.

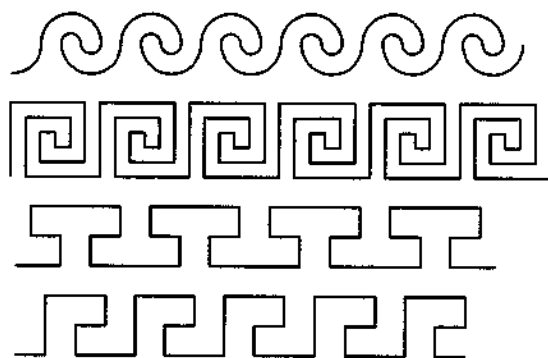


Рис. 7.25. Узоры-меандры (к заданию 7.63, б)



Рис. 7.26. К заданию 7.63, в

- г) Придумайте композицию, состоящую из различных фигур, программы для которых уже содержатся на вашем листе программ. Устанавливайте Черепашку в определенные точки видимого поля, используя команды `нов_х`, `нов_у`, `нов_курс`. Придумайте и расскажите в классе историю про вашу композицию.

Тема 7.6 Процедура с параметрами

Выполнив задания этой темы, вы научитесь:

- ⇒ анализировать объект, выделяя в нем настраиваемые параметры;
- ⇒ описывать в процедуре настраиваемые формальные параметры;
- ⇒ задавать значения фактических параметров при вызове процедуры.

Процедура с одним параметром

До сих пор в программах для создания графических объектов длина отрезка, угол поворота, цвет задавались определенными числами. Однако часто возникает необходимость построить такой же объект, но с другими значениями параметров. Рассмотрим, как это сделать, на примере процедуры рисования квадрата. Для этого проведем анализ свойств объекта «Квадрат» и выявим общие и специфические параметры.

Этап 1. Перечислим параметры, характеризующие квадрат на рабочем поле:

- а) количество сторон (всегда 4);
- б) соотношение сторон (всегда равны);
- в) длины сторон (например, 60 пикселей);
- г) соотношение углов (всегда равны);
- д) величины углов (все по 90°);
- е) цвет контура (например, синий);
- ж) цвет внутри контура (например, желтый);
- з) координаты расположения на рабочем поле (например, нижняя левая вершина в точке с координатами $x = 0, y = 0$);

Этап 2. Выявим среди перечисленных характеристик параметры, общие для любого квадрата.

Параметры а), б), г), д) определяют общие свойства квадрата. Изменение их значений превратит квадрат в другую фигуру.

- ◆ Параметры в), е), ж), з) характеризуют конкретный квадрат. Математическое определение объекта «Квадрат» не зависит от значений этих параметров.

Этап 3. Перечислим действия, которые можно выполнить над объектом на рабочем поле.

Квадрат можно нарисовать в любом месте рабочего поля. Изменяя цвет, длину стороны и место расположения квадрата, можно построить сложные объекты, например орнаменты из квадратов.

При использовании квадрата в другом объекте цвет контура и положение начальной вершины на рабочем поле зависят от места этого квадрата в сложном объекте. Значения таких параметров можно принять текущими.

Результаты анализа свойств квадрата представлены в табл. 7.15.

Таблица 7.15. Информационная модель объекта «Квадрат»

Параметры объекта		Действия объекта
Название	Значение	
Число сторон	4	Рисовать
Соотношение сторон	Все стороны равны	Стирать
Соотношение углов	Все углы равны	
Величина угла, внешнего к углу между сторонами	90°	
Длина стороны	Можно изменять	Приобретать указанные размеры
Цвет контура	Текущий	Устанавливать указанный цвет контура
Цвет внутри контура	Текущий	Закрашивать внутри контура
Координаты расположения левой нижней вершины	Текущие	Располагать в разных местах рабочего поля

Вспомним процедуру квадрат, которая строит квадрат со стороной 30 пикселей:

```
это квадрат  
повтори 4 [вп 30 пр 90]  
конец
```

Сравним эту процедуру с процедурами построения квадратов с другими длинами сторон, например, 80 и 60 пикселей:

```
это кв80  
повтори 4 [вп 80 пр 90]  
конец  
это кв60  
повтори 4 [вп 60 пр 90]  
конец
```

Процедуры `квадрат`, `кв80` и `кв60` предназначены для рисования квадратов, но стороны получаемых квадратов различны. Удобно было бы иметь одну процедуру, позволяющую подставлять в нее заданную длину стороны. Сторона квадрата рисуется командой `вп`, следовательно, именно в этой команде нужно указать, что ее параметр может быть переменным.

Программная среда позволяет создавать процедуры с изменяющимися значениями параметров. Для этого параметру присваивается имя, которое должно быть указано в заголовке процедуры. В команде, использующей настраиваемый параметр, вместо конкретного значения (числа) пишется имя, объявленное в заголовке.

Например, процедура рисования квадрата с изменяемой длиной стороны может выглядеть так:

```
это квадрат сторона  
повтори 4 [вп :сторона пр 90]  
конец
```

В заголовке процедуры появилось еще одно слово — *имя параметра*. Это имя (`сторона`) используется в команде `вп` и показывает, что конкретные данные для этой команды будут указаны позже. Здесь вместо указания конкретного числа используется так называемый *формальный параметр*.

 **Формальный параметр** — это параметр, значение которого в процедуре может быть любым. Имя формального параметра задается в заголовке процедуры.

В языке ЛОГО двоеточие (:) перед именем параметра показывает, что в данной команде используется значение параметра, которое будет известно при выполнении процедуры.

Вызывая процедуру с параметром, необходимо после ее имени задать *фактическое значение* этого параметра. Например, для построения квадрата со стороной 50 шагов следует использовать команду `квадрат 50`.



Фактический параметр — это значение, задаваемое при вызове процедуры. Это значение заменит формальный параметр при выполнении команды вызова процедуры.

Задание 7.64

1. Запишите на листе программ процедуру:
это квадрат сторона
повтори 4 [вп :сторона пр 90]
конец
2. Введите команду `квадрат` без фактического параметра. В командном центре появится сообщение: Не хватает входных данных для `квадрат`.
3. Исправим ошибку, задав значение фактического параметра: `квадрат 100`. На рабочем поле будет нарисован квадрат со стороной 100 пикселей. При вводе команды `квадрат 80` на рабочем поле появится квадрат со стороной 80 пикселей.
4. Какие значения можно давать параметру? Очевидно, только те, что могут быть применены в команде, использующей этот параметр. Проверим это. Введем в командном центре: `квадрат s`. Появится сообщение: Не могу выполнить `s`.
5. Введем команду: `квадрат "s`. Появится сообщение: `вп` не допускает `s` на входе.

Задание 7.65

Исследуйте, каковы наименьшее и наибольшее значения параметра `сторона` в процедуре `квадрат`.

Вводите в командный центр поочередно команду `квадрат` с разными значениями параметра и передвигайте Черепашку на новое место.

```
по квадрат 2 пп нов_х 2
по квадрат 150 пп нов_у -170
по квадрат 500 пп нов_х -300
по квадрат 700
```

Что вы видите на рабочем поле при самом маленьком фактическом параметре, а что — при самом большом? Почему?



Заломните правила создания и вызова процедуры с параметром:

- ☒ При создании процедуры в заголовке после имени процедуры записывается имя формального параметра. В теле процедуры это имя указывается в командах, использующих этот параметр.
- ☒ При вызове процедуры после ее имени обязательно указывается фактический параметр.

Процедура с несколькими параметрами

Количество параметров в процедуре зависит от описания объекта. Рассмотрим пример создания правильного многоугольника, свойства которого отображены в табл. 7.16.

В соответствии с представленным описанием необходимо иметь несколько изменяемых параметров: число сторон, длина стороны, цвет контура.

Напишем процедуру `прав_мн_угол`, которая рисует правильный многоугольник текущим цветом. Используем алгоритм из табл. 7.12 (тема 7.4):

```
это прав_мн_угол а n
повтори :n [вп :а пр 360 / :n]
конец
```

Для изменения цвета необходимо в данную процедуру добавить еще один параметр:

```
это прав_мн_угол а n с
нц :с
повтори :n [вп :а пр 360 / :n]
конец
```

Чтобы на рабочем поле нарисовать синий шестиугольник с длиной стороны 50 пикселей, введем команду: `прав_мн_угол 50 6 5`. Здесь первый фактический параметр (число 50) обозна-

чает длину стороны, второй (число 6) — количество сторон (вершин), а третий (число 5) — цвет.

Таблица 7.16. Информационная модель объекта «Правильный многоугольник»

Параметры объекта		Действия объекта
Название	Значение	
Соотношение сторон	Все стороны равны	Рисовать
Соотношение углов	Все углы равны	Стирать
Величина угла, внешнего к углу между сторонами	Полный угол, разделенный на число сторон	
Число сторон	Можно изменять	Приобретать указанное число сторон
Длина стороны	Можно изменять	Приобретать указанные размеры
Цвет контура	Можно изменять	Устанавливать указанный цвет контура
Цвет внутри контура	Текущий	Закрашивать внутри контура
Координаты расположения начальной вершины	Текущие	Располагать в разных местах рабочего поля



Запомните правила создания и вызова процедуры с несколькими параметрами:

- ☒ При описании процедуры в заголовке после ее имени записываются имена формальных параметров, разделенные пробелом.
- ☒ В теле процедуры имена появляются как параметры в командах.
- ☒ При вызове процедуры после ее имени записываются все фактические параметры в порядке, соответствующем порядку записи формальных параметров в заголовке процедуры.

Задания для самостоятельной работы

Задание 7.66. Составьте процедуру для рисования правильного шестиугольника с длиной стороны, заданной формальным параметром.

Задание 7.67. Составьте процедуру для рисования правильного треугольника с длиной стороны, заданной формальным параметром.

Задание 7.68. Составьте процедуру для рисования условной окружности — правильного 36-угольника с длиной стороны, заданной формальным параметром.

Задание 7.69. Напишите процедуру рисования закрашенного квадрата с длиной стороны 80 пикселей и изменяемым параметром — цветом контура. Какие изменения надо внести в имеющийся алгоритм построения квадрата?

Задание 7.70. Напишите программы для рисования композиций, представленных на рис. 7.27. Для описания каждой композиции заполните табл. 7.17 аналогично тому, как заполнялась табл. 7.16.

Установите начальное и конечное состояния Черепашки и рабочего поля.

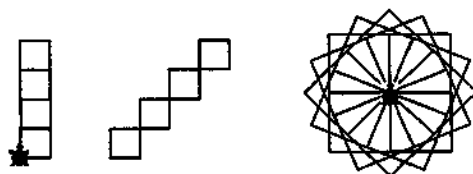


Рис. 7.27. К заданию 7.70

Таблица 7.17. Шаблон информационной модели объекта

Параметры объекта		Действия объекта
Название	Значение	
Количество квадратов		Рисовать
Соотношение квадратов		Стирать
Соотношение вершин квадратов		
Соотношение углов между квадратами		
Цвет контура		

Задание 7.71. Напишите процедуру рисования полуокружности как половины 360-угольника с переменной длиной стороны и углом поворота после каждого шага, равным 1° .

Задание 7.72. Используя процедуру, созданную при выполнении предыдущего задания, напишите программы для рисования составных графических объектов (рис. 7.28).

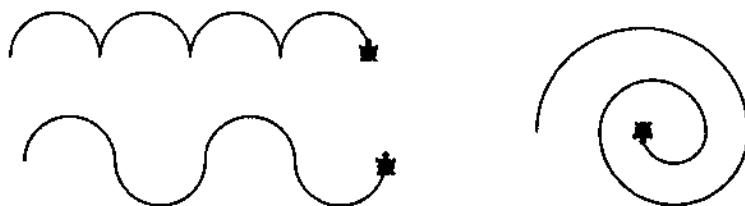


Рис. 7.28. К заданию 7.72

Задание 7.73. Модифицируйте программу `прав_мн_угол`, добавив еще один параметр — угол поворота. Теперь угол поворота можно задавать независимо от количества сторон. Какой графический объект описывает такая программа?

Задание 7.74. Опишите элементарный объект, из которого составлены снежинки (см. рис. 7.22). Напишите процедуру рисования элементарного объекта с параметром, управляющим размером этого объекта.

Задание 7.75. Составьте информационную модель объекта «Куб», который можно изобразить на плоскости как объемную фигуру. Какие элементы добавляются к квадрату, чтобы на экране можно было увидеть куб? Представьте алгоритм рисования куба в виде таблицы.

Задание 7.76. Используя дугу, равную четверти окружности, опишите объект «Летающая птица». Составьте программный модуль для изображения птицы, который вызывает процедуру рисования дуги. Напишите программу создания стаи летящих птиц, используя модуль рисования одной птицы.

Тема 7.7 Переменная в алгоритме

Выполнив задания этой темы, вы научитесь:

- ⇒ задавать и изменять значение переменной;
- ⇒ запоминать в одной переменной список значений;
- ⇒ использовать значение переменной для вычисления параметров команд.

На рисунке 7.29 изображен объект «Тоннель». Он состоит из 40 восьмиугольников, причем сторона каждого следующего на 1 пиксел длиннее текущего. Мы можем указать в процедуре длину самого маленького восьмиугольника как изменяемый параметр:

```
это тоннель1 s  
повтори 40 [прав_мн_угол :s 8 вп 1]  
конец
```

Процедура тоннель1, повторенная 40 раз, позволит нарисовать 40 восьмиугольников, но все они будут с одинаковой заданной длиной стороны. Для получения тоннеля, изображенного на рис. 7.29, значение s надо увеличивать на каждом проходе цикла. Иначе говоря, значение параметра s должно храниться в памяти компьютера, чтобы исполнитель мог его менять по ходу выполнения программы. В программировании для этого используют объект *переменная*.

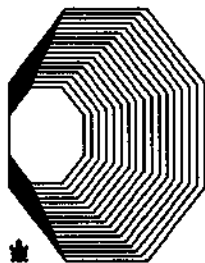


Рис. 7.29. Объект Тоннель



Переменная — это объект в программе, имеющий имя и изменяемое значение. Для каждой переменной выделено определенное место в памяти компьютера.

В любой среде программирования различаются имя и значение переменной.

Имя переменной показывает, в каком месте памяти компьютера хранится значение переменной.

Значение переменной считывается из указанного ее именем места памяти. Значение переменной применяется как фактический параметр в команде и в выражении.

Переменную можно создать, то есть указать ее имя и присвоить ей значение. В среде ЛОГО для создания переменной приме-

няется команда **пусть**. Эта же команда изменяет значение уже существующей переменной. В языке ЛОГО явно показывается различие между именем и значением переменной: имя предваряется двойной кавычкой ("), а значение — двоеточием (:). Символ **:** обозначает действие *взять данные из области памяти, названной именем переменной*. В некоторых языках программирования имя и значение переменной могут не различаться по виду.



Запомните правило работы с переменной в среде ЛОГО:

Команда **пусть** имеет два параметра. Первый указывает имя переменной, второй задает выражение для получения значения этой переменной.

Рассмотрим пример. В переменную **a1** надо записать число 30, в переменную **a2** — число 6, а в переменную **сумм** — сумму переменных **a1** и **a2**.

Для выполнения этих действий вводим команды:

пусть "a1 30

пусть "a2 6

пусть "сумм :a1 + :a2

Результат выполнения этих команд на экран компьютера не выводится.

Командой **пусть** можно также изменить значение уже созданной переменной. Например, для увеличения значения переменной **s** на 1 вводим:

пусть "s :s + 1

В этом случае из памяти считывается старое (текущее) значение переменной **s** и увеличивается на 1, а затем новое значение записывается в то же место памяти.

Переменные служат для хранения значений фактических параметров и других данных, используемых при выполнении процедуры. При описании процедуры формальный параметр обозначает имя переменной. При вызове процедуры в памяти компьютера создается переменная с этим именем и в нее записывается значение фактического параметра.

В одной переменной можно запомнить более одного значения. Например, положение Черепашки на рабочем поле задается абсциссой и ординатой. Эти координаты можно измерить датчиком место и записать в одну переменную командой:

пусть "положение место

Тогда в переменной положение окажется список из двух значений. Вы увидите их, если введете команду:

```
покажи :положение
```

Команда `нов_место :положение` устанавливает Черепашку в указанное место рабочего поля.

Задания для самостоятельной работы

Задание 7.77. Запишите на листе программ программу для рисования тоннеля (см. рис. 7.29), которая вызывает процедуру построения правильного многоугольника `прав_мн_угол s` (параметр `s` показывает длину стороны самого маленького восьмиугольника). Проверьте работу этой программы:

```
это тоннель :s  
повтори 40 [прав_мн_угол :s 8 пусть "s :s + 1]  
конец
```

Задание 7.78. Измените программу тоннель так, чтобы тоннель не увеличивался от меньшего восьмиугольника к большему, а уменьшался от большего к меньшему.

Задание 7.79. Создайте в теле программы тоннель переменную `s` — счетчик для номера цвета, начальное значение счетчика задайте равным 9. На каждом шаге цикла увеличивайте значение переменной `s` на 10 и устанавливайте цвет Черепашки равным значению `s`. Тоннель получится разноцветным.

Задание 7.80. Каким будет значение переменной `z` после выполнения следующих команд:

- а) пусть "z 10 вп :z + 5
- б) пусть "z 10 вп :z пусть "z :z + 5
- в) пусть "z 10 повтори 5 [пусть "z :z + 1]
- г) пусть "z 105 / (30 + 5)
- д) пусть "z 105 / 30 + 5

Задание 7.81. Сохраните координаты точки, в которой находится в данный момент Черепашка, в переменной `ст_место`:

```
пусть "ст_место место
```

Передвиньте Черепашку мышью в другую точку, а затем выполните команду:

```
мм :ст_место
```

Черепашка вернется на прежнее место.

Тема 7.8 **Логика в среде программирования**

Выполнив задания этой темы, вы научитесь:

- ⇒ записывать операции сравнения;
- ⇒ записывать логическое выражение;
- ⇒ описывать разветвляющийся алгоритм;
- ⇒ описывать рекурсивный процесс;
- ⇒ моделировать микромиры.

Инструменты сравнения

Знаменитый Шерлок Холмс при расследовании применял законы логики, что позволяло ему находить выход из любой ситуации и распутывать самые замысловатые истории.

Лучшим способом освоения понятий логики является их практическое применение при составлении алгоритмов и проверке алгоритмов на компьютере.

В качестве примера использования законов логики рассмотрим задачу о лабиринте. Как научить Черепашку проходить по лабиринту?

Прежде всего построим объект «Лабиринт», который состоит из закрашенных квадратиков (плиток) одного цвета. Процедуры для построения лабиринта из элементарных объектов приведены в табл. 7.18.

Задание 7.82

Запишите на листе программ процедуры, приведенные в табл. 7.18. Проверьте работу этих процедур. Опишите объект «Лабиринт», состоящий из плиток одного заданного цвета и одного заданного размера. Какие параметры нужны для этого объекта? Какой инструмент нужен для рисования такого объекта? Напишите процедуру лабиринт, которая рисует лабиринт, начиная с текущего положения Черепашки (рис. 7.30).

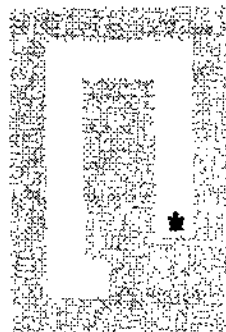


Рис. 7.30.
Лабиринт

Таблица 7.18. Инструменты-процедуры для построения лабиринта

Объект	Параметры	Инструмент (процедура)
Элемент закрашенной плитки	р — длина стороны элемента; с — цвет элемента	это эл_пл р цв по нц :цв повтори 3 [лв 90 вп :р] конец
Закрашенная плитка, состоящая из элементов закрашенной плитки	ст — длина стороны плитки; цв — цвет плитки	это плитка ст цв пусть "е 1 пусть "хн х_коор пусть "ун у_коор повтори :ст - 1 [эл_пл :е :цв пусть "е :е + 1] нов_х :хн нов_у :ун нк 0 конец

Исполнитель Черепашка установлена в начало лабиринта. Она может идти только по плиткам лабиринта. Следовательно, исполнитель должен измерить цвет плитки, на которой он находится, и сравнить его с заданным значением. В зависимости от результата сравнения Черепашка выберет направление движения.

Для выполнения операций сравнения в любой среде программирования имеются инструменты, представленные в табл. 7.19. Как видно из таблицы, значение операции сравнения может быть только истина (да) или ложь (нет).

Таблица 7.19. Инструменты сравнения

Обозначение	Форма записи	Значение операции сравнения
>	$x1 > x2$	да (истина), если значение $x1$ больше значения $x2$, нет (ложь) — в противном случае
<	$x1 < x2$	да (истина), если значение $x1$ меньше значения $x2$, нет (ложь) — в противном случае
=	$x1 = x2$	да (истина), если значение $x1$ равно значению $x2$, нет (ложь) — в противном случае

Операция сравнения используется в качестве параметра в командах. Например, вывести значение операции сравнения в командный центр можно с помощью команды покажи:

```
покажи color > 8  
покажи 100 / 3 < 50
```

Если для анализа состояния объекта требуется осуществить несколько проверок, составляется логическое выражение, включающее несколько операций сравнения. В логическом выражении операции связываются логическими действиями И, ИЛИ и НЕ. Например, чтобы проверить, совпадает ли цвет Черепашки с цветом пиксела под ее пером и одновременно с цветом фона, введите команду:

```
покажи и (цвет = цп) (цвет = фон)
```

Выбор действий по условию

Рассмотрим такую ситуацию. Черепашка гуляет по рабочему полю. На рабочем поле расположено озеро определенного цвета. Черепашка, попав на озеро, должна «спрятаться».

Для моделирования ситуации сначала нужно нарисовать объект «Озеро», изображенный на рис. 7.31 и описанный в табл. 7.20. ★ Для этого потребуется специальный инструмент — процедура озеро.

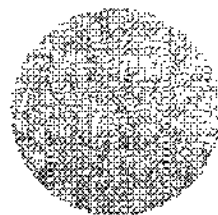


Рис. 7.31. Озеро

Для создания объекта «Озеро» введите команду:

```
озеро 1 360 85
```

Инструмент-процедура фортуна заставляет Черепашку менять свое положение на рабочем поле случайным образом:

```
это фортуна  
пп нов_x -300 + сл 600  
нов_y -180 + сл 360  
конец
```

Задание 7.83

Составьте и запишите на листе программ процедуры озеро и фортуна. Проверьте работу этих процедур. Проверьте в разных точках рабочего поля значение выражения «цвет под пером равен 85», устанавливая Черепашку на озере и вне озера и вводя команду покажи цп = 85.

Таблица 7.20. Информационная модель объекта «Озеро»

Параметры объекта		Инструмент (процедура)
Название	Значение	
Цвет	цв	это озеро ст чв цв
Форма	Правильный многоугольник	пп нов_х сл 100 нов_у сл 100 пусть "тц цвет нц :цв по
Длина стороны	ст	прав_мн_угол :ст :цв
Число вершин	чв	пр 90 пп вл :ст по крась
Начальное положение исполнителя	Случайное в области: $0 < x < 100$ $0 < y < 100$	нд :ст лв 90 нц :тц
Конечное положение исполнителя	Черепашке возвращается исходный цвет, и она располагается вне озера	пп нов_х -100 нов_у -100 конец

Как научить Черепашку самостоятельно менять свое состояние? В среде ЛОГО есть команды, с помощью которых можно выбрать нужные действия, проанализировав условие:

- ◆ команда **если** позволяет реализовать неполную форму разветвляющегося алгоритма;
- ◆ команда **если_иначе** позволяет реализовать полную форму разветвляющегося алгоритма.

Варианты возможных действий задаются списком команд. *Список команд* — это последовательность команд, заключенная в квадратные скобки. Команды в списке отделяются друг от друга пробелом.

Формат команды для неполной формы разветвляющегося алгоритма:

если *условие* [*список команд*]

Команда состоит из ключевого слова **если** и двух входных параметров: *условия* и одного *списка команд*. Условие представляет собой одну операцию сравнения или логическое выражение. Список команд содержит те команды, которые надо выполнить, если условие выполняется (значение выражения да, или «исти-

на»). Если значение условия «ложь», то команды, записанные в квадратных скобках, не выполняются.

Формат команды для полной формы разветвляющего алгоритма:
если_иначе условие [список команд 1] [список команд 2]

Команда состоит из ключевого слова *если_иначе* и трех входных параметров: *условия* и двух *списков команд*. Первый список команд содержит те команды, которые надо выполнить, если условие выполняется (истинно). Второй список команд содержит команды, которые надо выполнить в случае, когда условие ложно.

Теперь научим Черепашку «нырять под воду» — становиться невидимой. На русском языке мы объяснили бы объекту, что он должен делать, таким образом:

Если цвет под пером голубой (номер 85), то стань невидимой, подожди некоторое время и снова покажись.

В приведенном выше объяснении после слова «если» записано условие: *цвет под пером голубой (номер 85)*.

На языке ЛОГО это условие будет выглядеть так: *цп = 85*.

Действия, которые должна выполнить Черепашка, записаны после слова «то»: *стань невидимой, подожди некоторое время и снова покажись*.

На языке ЛОГО эти действия выражаются командами: *сч жди 10 пч*.

Разветвляющийся алгоритм для Черепашки выглядит следующим образом:

если *цп = 85* [*сч жди 10 пч*]

Задание 7.84

Нарисуйте на рабочем поле озеро командой *озеро 1 360 85* и установите Черепашку вне озера.

а) Введите команду:

если *цп = 85* [*сч жди 10 пч*]

Вы видите, что при цвете под пером, не равном 85, никаких действий не выполняется.

б) Переместите Черепашку на озеро. Введите в командном центре команду:

если *цп = 85* [*сч жди 10 пч*]

Черепашка исчезла на некоторое время, а потом появилась опять.

Теперь пусть Черепашка на берегу оставляет квадратный коврик. Ваш алгоритм должен предусматривать один список действий при выполнении условия и другой — в противном случае:

Если цвет под пером голубой (номер 85), то стань невидимой, подожди некоторое время и покажись, иначе нарисуй квадрат со стороной 10 шагов.

Запишем этот алгоритм на языке Лого:

```
если_иначе цп = 85 [сч жди 10 пч] [по нц 65 цв_кв 10 ]
```

Задание 7.85

Нарисуйте на рабочем поле озеро командой `озеро 1 360 85` и установите Черепашку вне озера.

а) Введите команду:

```
если_иначе цп = 85 [сч жди 10 пч]  
[по нц 65 плитка 10 45]
```

Вы видите, что при цвете под пером, не равном 85, Черепашка рисует коврик.

б) Переместите Черепашку на озеро. Введите в командном центре команду:

```
если_иначе цп = 85 [сч жди 10 пч]  
[по нц 65 плитка 10 45]
```

Черепашка исчезла на некоторое время, а потом появилась опять.

Создание процесса в среде программирования

В заданиях 7.84 и 7.85 Черепашка один раз выполняла действия, изменяющие ее состояние. Что получится, если эти действия повторять в течение некоторого времени t ?

Процедура фортуна забрасывает Черепашку в случайную точку рабочего поля. Что мы будем наблюдать на экране, если запускать ее несколько (t) раз?

Задание 7.86

Нарисуйте на рабочем поле озеро командой `озеро 1 360 85` и установите Черепашку вне озера.

а) Введите команду:

```
повтори 100 [фортуна]
```

Черепашка «летает» по рабочему полю.

б) Введите команду:

```
повтори 100 [фортуна если цп = 85 [сч жди 10 пч]]
```

Поведение Черепашки изменилось. Попадая на озеро, она исчезает на некоторое время, а потом снова появляется.

в) Введите в командном центре команду:

```
повтори 100 [фортуна
```

```
если_иначе цп = 85 [жди 10 сч жди 10 пч]
```

```
[плитка 10 45]]
```

Теперь Черепашка то исчезает в озере, то рисует коврик на берегу.

Последовательную смену состояний объекта в результате произведенных действий называют *процессом*. Процесс обязательно разворачивается во времени. В среде программирования за единицу времени можно принять один проход цикла. Циклический алгоритм является одним из способов моделирования процесса в среде программирования

В среде ЛОГО можно создать повторяющийся процесс и другим способом. Известно, что в теле процедуры допускается вызов любой процедуры, в том числе и той же самой. Оформим алгоритм изменения состояния Черепашки как процедуру:

```
это изм_сост
```

```
фортуна
```

```
если_иначе цп = 85 [жди 10 сч жди 10 пч] [плитка 10 45]]
```

```
конец
```

Для моделирования процесса прогулки вокруг озера предложим следующую программу:

```
это прогулка
```

```
изм_сост
```

```
прогулка
```

```
конец
```

Выполнение алгоритма прогулка напоминает известную сказку про белого бычка: по команде изм_сост Черепашка изменяет свое состояние, затем снова вызывается программа про -

гупка и все начинается сначала. Указанный способ вызова процедуры, прогулка называется *рекурсией*.



Рекурсивный алгоритм — это алгоритм повторения, в котором процедура по ходу ее выполнения обращается к самой себе.

Процесс, основанный на рекурсивном вызове, называется **рекурсивным**.

В природе существует множество рекурсивных процессов: курица несет яйцо, из яйца появляется цыпленок, который вырастает в курицу, которая несет новые яйца; на дереве образуется семечко, семечко падает в землю и из него вырастает новое дерево, и т. д. Рекурсивный алгоритм применяется для моделирования подобных рекурсивных процессов.

Рассмотрим новый графический объект — «Спираль». Пусть Черепашка рисует некоторый отрезок и поворачивается на определенный угол. На каждом проходе рекурсии отрезок должен уменьшаться, а угол оставаться неизменным:

```
это спираль с угол
вп :с пр :угол спираль :с - 5 :угол
конец
```

Введя команду `по спираль 50 90`, вы будете наблюдать бесконечный процесс рисования прямоугольной спирали.

Вы можете ограничить спираль, задав, например, минимальный допустимый отрезок.

```
это спираль с угол
если :с < 5 [стоп]
вп :с пр :угол спираль :с - 5 :угол
конец
```

Таким образом, бесконечный рекурсивный процесс можно завершить, применив разветвляющийся алгоритм.

Задание 7.87

Сравните подготовленную вами ранее процедуру лабиринт, которая строит лабиринт из квадратных цветных плиток указанных размера и цвета, с процедурой, приведенной ниже:

```
это лабиринт ст цв
  пусть "хт х_коор пусть "ут у=коор пусть "цт цвет
```

```
сч повтори 4 [плитка :ст :цв нов_х х_коор + :ст]  
повтори 3 [плитка :ст :цв нов_у у_коор + :ст]  
нов_у у_коор - :ст нов_х х_коор - :ст  
повтори 6 [плитка :ст :цв нов_х х_коор - :ст]  
пп нов_х :хт нов_у :ут нц :цт нк 90 пч  
конец
```

В приведенном алгоритме Черепашка после рисования лабиринта помещается в середину первой плитки, а ее голова смотрит на Восток.

Используя программу лабиринт, нарисуйте на рабочем поле лабиринт из голубых (цвет 85) плиток размером 30 шагов. Измерьте цвет под пером Черепашки, выведя его значение в командный центр.

Задание 7.88

Черепашка находится в лабиринте. Вводя несколько раз команды `вп 30` покажи `цп = 85`, проведите ее до поворота. Черепашка окажется вне лабиринта. Какое слово появится в командном центре — «истина» или «ложь»? Что теперь должна делать Черепашка?

Как научить Черепашку самостоятельно проходить лабиринт, если она уже стоит в нем и смотрит вдоль правильного пути? В этой задаче взаимодействуют два объекта — Черепашка и лабиринт. Черепашка должна знать параметры второго объекта — длину стороны и цвет плитки.

В этом алгоритме возникает необходимость разработать вспомогательный алгоритм `ищи_ход`. Черепашке надо вернуться на предыдущую плитку и повернуться на прямой угол.

Сравните придуманный вами алгоритм с представленным в табл. 7.21.

Задание 7.89

Запишите программы `путь_по_лаб` и `ищи_ход` на листе программ. Проверьте, как Черепашка выбирается из лабиринта. Выйдет ли Черепашка из лабиринта? Нужно ли задать еще один параметр объекта «Лабиринт», чтобы Черепашка распознала правильный путь?

Таблица 7.21. Программы прохождения по лабиринту

Словесное описание	Программа
Путь по лабиринту Повторить t раз: если под пером — цвет лабиринта то вперед на s шагов, иначе назад на s шагов Ищи ход Конец повторения Конец	это путь_по_лаб t с $цл$ повтори : t [если_иначе $цл = :цл$ [вп : s] [нд : s ищи_ход : s]] конец
Ищи ход налево на 90° вперед на s шагов Конец	это ищи_ход s лв 90 вп : s конец

Черепашка странно ведет себя в лабиринте. На каком-то участке она в панике бежит по одним и тем же плиткам до тех пор, пока не кончится отведенное время. Очевидно, ей необходима «нить Ариадны», чтобы отмечать уже пройденные участки. Например, Черепашка может оставлять след на каждой пройденной плитке:

```

это путь_по_лаб1 с  $цл$ 
пп вп : $s$ 
если_иначе  $цл = :цл$  [нд : $s$  по штамп пп вп : $s$ ]
[если_иначе цвет = : $цл$  [стоп]
[нд : $s$  лв  $90$ ]]
путь_по_лаб1 : $s$  : $цл$ 
конец

```

Теперь Черепашка поворачивается и ищет путь, по которому она еще не проходила. Здесь уместно применить рекурсию.

Задание 7.90

Запишите процедуру путь_по_лаб1 на листе программ. Проверьте, как теперь Черепашка выбирается из лабиринта. Сколько времени ей потребуется? Нужно ли задать еще один параметр объекта «Лабиринт», чтобы Черепашка остановилась в конце лабиринта? Предложите свой вариант решения.

Моделирование системы

Для моделирования в среде программирования некоторой системы необходимо понять, для чего вам нужна эта модель, то есть поставить цель. Затем вы описываете объекты, которые будут составлять систему, и, конечно, их параметры. Вам надо будет задать правила взаимодействия этих объектов. Одним словом, для моделирования вам придется подготовить инструмент — процедуры и модули.

Моделирование можно сравнить с представлением в театре. Вы создаете свой маленький мир — микромир, готовите декорации, маски, костюмы, подбираете актеров, учите с ними роли. Затем запускаете процесс изменения параметров определенного объекта. То, что происходит потом, кажется загадочным, но постепенно вы разбираетесь в ходе событий, иногда обнаруживаете ошибки в описании объектов или их взаимодействия и в конце концов достигаете поставленной цели.

Представьте, что Черепашка обучена нескольким профессиям и отправилась путешествовать по рабочему полю. Судьба забрасывает Черепашку в разные части света. В зависимости от положения на координатной плоскости Черепашка выполняет ту или иную известную ей работу, например рисует определенный объект.

Цель нашего моделирования — освоить законы логики. Для этого вы познакомитесь с историями из жизни Черепашки в созданном микромире. Каждая история происходит в течение времени t , поэтому ее можно моделировать с помощью цикла.

На листе программ вам нужно подготовить инструменты для рисования объектов, создания процессов из жизни Черепашки в роли разных мастеров. Необходимые процедуры описаны в табл. 7.22 и 7.23.

История 1. Черепашка-лесник

Всякий раз, попадая на поле справа от вертикальной границы $x = 60$, Черепашка должна сажать дерево. Цвет дерева выбирается случайно в интервале от 51 до 58. Высота дерева должна быть не более 20 шагов Черепашки.

История 2. Черепашка-садовник

Всякий раз, попадая на поле слева от вертикальной границы $x = 0$, Черепашка должна сажать цветок. Цвет цветка выбирается случайно в интервале от 41 до 48. Количество лепестков цветка — не более 8.

Задание 7.91

Запишите процедуры, перечисленные в табл. 7.22 и 7.23, на листе программ и проверьте, как они работают.

Таблица 7.22. Инструменты-процедуры для создания объектов
(к заданию 7.91)

Объект	Параметры		Инструмент-процедура
	Название	Значение	
Дуга правая	Угол поворота Крутизна	а с	это дуга_пр а с повтори :а [вп :с пр 1] конец
Лепесток из двух дуг	Нет	нет	это лепесток дуга_пр 90 0.25 пр 90 дуга_пр 90 0.25 пр 90 конец
Цветок	Количество лепестков Цвет лепестка	к ц	это цветок к ц по нц :ц нк 0 вп 30 повтори :к [лепесток пр 360 / :к] пп конец
Ветка	Высота	в	это ветка в вп :в нд :в лв 15 вп :в нд :в пр 30 вп :в нд :в лв 15 нд :в конец
Дерево	Высота Цвет	в ц	это дерево в ц нц :ц нк 0 по лв 15 ветка :в / 2 пр 30 ветка :в / 2 лв 15 нд :в / 4 лв 15 ветка :в пр 30 ветка :в лв 15 нд :в / 4 пп конец
Плитка	Сторона Цвет	с ц	плитка :с :ц (см. табл. 7.18)

Таблица 7.23. Инструменты для создания процессов

Процесс (действия за один проход цикла)	Инструмент (процедура)
Судьба, забрасывающая исполнителя Черепашку в случайную точку	это фортуна пп нов_х -300 + сл 600 нов_у -180 + сл 360 конец
Посадка дерева со случайными высотой и цветом	это лесник дерево сл 20 51 + сл 8 конец
Посадка цветка со случайными числом лепестков и цветом	это садовник цветок сл 8 41 + сл 8 конец

Задание 7.92

Напишите по словесному алгоритму программу история_1 для моделирования первой истории в течение времени t:

Алгоритм истории 1

Повторить t раз следующие действия:

Судьба забросила Черепашку в случайную точку.

Если $x > 60$ (справа от вертикальной границы), то выполнить роль лесника

Конец цикла

Конец истории 1

Задание 7.93

Напишите по словесному алгоритму программу история_2 для моделирования второй истории в течение времени t:

Алгоритм истории 2

Повторить t раз следующие действия:

Судьба забросила Черепашку в случайную точку.

Если $x < 0$ (слева от вертикальной границы), то выполнить роль садовника

Конец цикла

Конец истории 2

Задания для самостоятельной работы

Задание 7.94. Выведите в командный центр с помощью команды покажи результаты сравнения:

- а) $2 \times 2 = 4$;
- б) $21 / 7 > 3$;
- в) $7 \times 3 = 21$.

Задание 7.95. Что выведет Черепашка на рабочее поле после выполнения следующих команд:

- а) пусть "а 12 нц 12 пиши :а = цвет
- б) нов_х 200 пиши х_коор > 0
- в) нов_у -150 пиши у_коор < -100
- г) пусть "b у_коор < -100 пиши :b

Задание 7.96. Нарисуйте блок-схему алгоритма прохождения лабиринта.

Задание 7.97. Приведите примеры, когда выбор тех или иных действий зависит от некоторого условия.

Задание 7.98. Нарисуйте в тетради координатную плоскость. Покажите на ней, что будет изображено на рабочем поле после выполнения команд:

- а) сотри история_1 5;
- б) сотри история_2 6.

Сколько объектов-деревьев и объектов-цветков будет нарисовано на экране? Где они будут расположены?

Задание 7.99. Развитие задачи про лабиринт. Добавьте объекту «Лабиринт» еще один параметр, который показывал бы, что Черепашка вышла из лабиринта и может остановиться. Опишите словами и блок-схемой алгоритм прохода лабиринта с учетом анализа значения нового параметра.

Задание 7.100. Составьте процедуру рисования спирали, которая раскручивается, увеличивая после каждого поворота длину отрезка.

Задание 7.101. Что будет выведено в командный центр по команде:

- а) покажи $113 < 50$;
- б) покажи $(\text{целое } 25 / 5) = 0$;
- в) покажи цвет = цп.

Задание 7.102. Какую команду надо ввести, чтобы в командном окне было печатано да, если цвет под Черепашкой равен 0, и нет, если цвет не равен 0?

РАЗДЕЛ 8 Основы алгоритмизации

Тема 8.1	Линейные алгоритмы	308
Тема 8.2	Разветвляющиеся алгоритмы	317
Тема 8.3	Циклические алгоритмы. Цикл с предусловием	331
Тема 8.4	Цикл с известным числом повторений . . .	347
Тема 8.5	Цикл с постусловием	362

Тема 8.1 **Линейные алгоритмы**

Выполнив задания этой темы, вы научитесь:

- ▣ представлять линейный алгоритм в различных формах;
- ▣ использовать линейные алгоритмы при решении задач;
- ▣ применять переменные для хранения данных;
- ▣ осуществлять ввод-вывод информации.

Как вам уже известно, существуют различные типы алгоритмов и разные формы их представления.

Для представления линейного алгоритма в виде блок-схемы используются блоки ввода-вывода, выполнения действий, вызова вспомогательного алгоритма.

Для представления линейного алгоритма в виде программы используются операторы ввода-вывода, оператор присваивания, оператор вызова вспомогательного алгоритма.

Задание 8.1

Коллекция Эрмитажа содержит более 2 800 000 единиц хранения. Если у каждого музейного экспоната задержаться всего на 5 минут и проводить в Эрмитаже по 8 часов каждый день, то может не хватить жизни, чтобы ознакомиться со всей коллекцией. Требуется вычислить суммарное время просмотра всей коллекции в минутах, часах, днях, годах, «жизнях», считая, что средняя продолжительность жизни в России составляет 70 лет.

Словесный алгоритм**Начало алгоритма**

1. Введите количество экземпляров коллекции.
2. Рассчитайте время просмотра всех экземпляров:
 - в минутах;
 - в часах;
 - в днях;
 - в годах;
 - в «жизнях».
3. Выведите результаты расчетов.

Конец алгоритма**Алгоритм в виде блок-схемы**

Рис. 8.1. Блок-схема алгоритма (к заданию 8.1)

Алгоритм в виде программы

В табл. 8.1 приведена программа к заданию на школьном алгоритмическом языке Кумир (с пояснениями). В табл. 8.2 приведены тексты программ на языках программирования Паскаль и Visual Basic.

Таблица 8.1. Программа на Кумире с пояснениями (к заданию 8.1)

№	Программа на Кумире	Пояснения	Тест
1	алг Коллекция	Начало программы	
	нач цел n, m, ch, d, g, life	Описание переменных: количество экземпляров, минуты, часы, дни, годы, «жизни»	
2	вывод "Введите количество экземпляров коллекции"	Приглашение к вводу числа	
3	ввод n	Ввод числа	2800000

№	Программа на Кумире	Пояснения	Тест
4	<pre> m := n * 5 ch := div(m, 60) d := div(ch, 8) g := div(d, 365) life := div(g, 70) </pre>	Расчет времени: в минутах в часах в днях в годах в «жизнях» с использованием цело- численного деления	<pre> 14000000 233333 29166 79 1 </pre>
5	вывод "количество:" вывод "минут", m вывод "часов", ch вывод "дней", d вывод "лет", g вывод "жизней", life	Вывод расчетов	<pre> + + + + + </pre>
6	кон	Конец алгоритма	

Таблица 8.2. Примеры программ на Паскале и Visual Basic (к заданию 8.1)

№	Программа на Паскале	Программа на Visual Basic
1	program Kollekcij;	Private Sub Kollekcij_Click()
	var n, m, ch, d, g, life: longint;	Dim n As Long, m As Long, ch As Long, d As Long, g As Long, life As Long
	begin	
2	write('Введите количество экземпляров коллекции: ');	n = Val(InputBox("Введите кол-во экземпляров"))
3	readln(n);	
4	m := n * 5;	m = n * 5
	ch := m div 60;	ch = m Div 60
	d := ch div 8;	d = ch Div 8
	g := d div 365;	g = d Div 365
	life := g div 70;	life = g Div 70

№	Программа на Паскале	Программа на Visual Basic
5	writeln('количество:');	Print "количество:"
	writeln('минут: ', m);	Print "минут: ", m
	writeln('часов: ', ch);	Print "часов: ", ch
	writeln('дней: ', d);	Print "дней: "; d
	writeln('лет: ', g);	Print "лет: "; g
	writeln('жизней: ', life);	Print "жизней: "; life
6	end.	End Sub

Задание 8.2

Требуется рассчитать параметры прямоугольного треугольника с углом 30° по заданному катету, лежащему против угла 30° .

Словесный алгоритм

Начало алгоритма

1. Введите длину катета.
2. Рассчитайте гипотенузу по введенному катету, исходя из свойств треугольника.
3. Рассчитайте второй катет по теореме Пифагора.
4. Выведите расчеты.

Конец алгоритма

Алгоритм в виде блок-схемы



Рис. 8.2. Блок-схема алгоритма (к заданию 8.2)

Алгоритм в виде программы

В табл. 8.3 приведена программа к заданию на школьном алгоритмическом языке Кумир. В табл. 8.4 приведены тексты программ на языках программирования Паскаль и Visual Basic.

Таблица 8.3. Программа на Кумире с пояснениями (к заданию 8.2)

№	Программа на Кумире	Пояснения	Тест
1	алг Прямоугольный_треугольник	Начало программы	
	нач вещ a, b, c	Описание переменных	
2	вывод "Введите катет"	Приглашение к вводу числа	
3	ввод a	Ввод числа	50
4	$c := a * 2$	Расчет гипотенузы	100
5	$b := \sqrt{c * c - a * a}$	Расчет катета	86,6
6	вывод "Длина 1-го катета", a вывод "Длина 2-го катета", b вывод "Длина гипотенузы", c	Вывод расчетов	
7	кон	Конец алгоритма	

Таблица 8.4. Примеры программ на Паскале и Visual Basic (к заданию 8.2)

№	Программа на Паскале	Программа на Visual Basic
1	program Prym_treug;	Private Sub Prym_treug_Click()
	var a, b, c: real;	Dim a As Single, b As Single, c As Single
	begin	
2	write('Введите катет: ');	a = Val(InputBox("Введите катет"))
3	readln(a);	
4	$c := a * 2$;	$c = a * 2$
5	$b := \sqrt{c * c - a * a}$;	$b = \text{Sqr}(c^2 - a^2)$
6	writeln('Длина 1-го катета: ', a); writeln('Длина 2-го катета: ', b); writeln('Длина гипотенузы: ', c);	Print "Длина 1-го катета: ", a Print "Длина 2-го катета: ", b Print "Длина гипотенузы: ", c
7	end.	End Sub

Задание 8.3

На памятнике Пифагору высечен чертеж вписанного в цилиндр шара, так как Пифагор нашел соотношение между их объемами. Современный почитатель гения Пифагора решил создать объемный памятник в честь этого открытия. Городские власти определили статус памятника — скульптура малой формы, и выделили для него небольшую площадь. Требуется рассчитать объем цилиндра и вписанного в него шара по заданной площади основания памятника (цилиндра) и убедиться в правильности выведенного Пифагором соотношения.

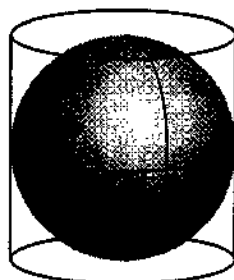


Рис. 8.3. К заданию 8.3

Словесный алгоритм

Начало алгоритма

1. Введите значение площади основания цилиндра.
2. Рассчитайте радиус основания цилиндра.
3. Рассчитайте объем цилиндра.
4. Рассчитайте объем шара.
5. Найдите соотношение между объемами цилиндра и шара.
6. Выведите расчеты.

Конец алгоритма

Алгоритм в виде блок-схемы

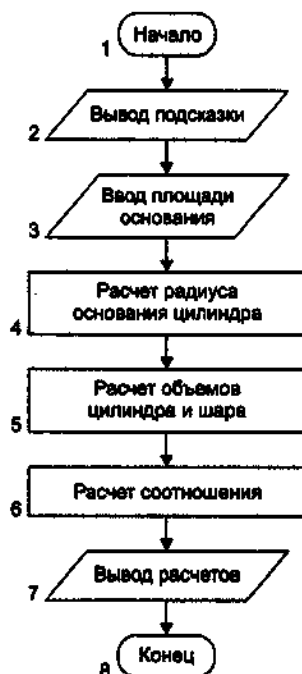


Рис. 8.4. Блок-схема алгоритма (к заданию 8.3)

Алгоритм в виде программы

В табл. 8.5 приведена программа к заданию на школьном алгоритмическом языке Кумир. В табл. 8.6 приведены тексты программ на языках программирования Паскаль и Visual Basic.

Таблица 8.5. Программа на Кумире с пояснениями (к заданию 8.3)

№	Программа на Кумире	Пояснения	Тест
1	алг Памятник_Пифагору	Начало программы	
	нач вещ r, s, vshara, vcil, k	Описание переменных	
2	вывод "Введите площадь основания цилиндра"	Приглашение к вводу числа	
3	ввод s	Ввод площади основания	314 (см ²)
4	$r := \sqrt{s / 3.1415926}$	Расчет радиуса	10.0 (см)
5	$vcil := s * 2 * r$	Расчет объема цилиндра	6278.5 (см ³)
	$vshara := 4 / 3 * 3.1415926 * r ** 3$	Расчет объема шара	4185.7 (см ³)
6	$k := vcil / vshara$	Расчет отношения	1.5
7	вывод "Радиус основания =", r вывод "Объем цилиндра =", vcil вывод "Объем шара, вписан- ного в цилиндр =", vshara вывод "Отношение =", k	Вывод расчетов	
8	кон	Конец алгоритма	

Таблица 8.6. Примеры программ на Паскале и Visual Basic (к заданию 8.3)

№	Программа на Паскале	Программа на Visual Basic
1	program Pifagor;	Private Sub Pifagor_Click()
	var r, s, vshara, vcil, k: real;	Dim r As Single, s As Single, vshara As Single, vcil As Single, k As Single

№	Программа на Паскале	Программа на Visual Basic
	begin	
2	write('Введите площадь основания: ');	s = Val(InputBox("Введите площадь основания ", "Пифагор"))
3	readln(s);	
4	r := sqrt(s / pi);	r = Sqr(s / 3.1415926)
5	vcil := s * 2 * r; vshara := 4 / 3 * pi * sqr(r) * r;	vcil = s * 2 * r vshara = 4 / 3 * 3.1415926 * r^3
6	k := vcil / vshara;	k = vcil / vshara
7	writeln('Радиус основания: ', r:6:1); writeln('Объем цилиндра: ', vcil:6:1); writeln('Объем шара: ', vshara:6:1); writeln('Отношение: ', k:3:1);	Print "Радиус основания ="; FormatNumber(r, 1) Print "Объем цилиндра = "; FormatNumber(vcil, 1) Print "Объем шара ="; FormatNumber(vshara, 1) Print " Отношение ="; FormatNumber(k, 1)
8	end.	End Sub

Контрольные вопросы и задания

К заданию 8.1

1. Разработчик алгоритма к заданию 8.1 (рис. 8.1), введя переменную n , хотел придать алгоритму свойство массовости. Какие еще переменные следует ввести, чтобы алгоритм соответствовал этому свойству в полной мере?
2. В приведенном последовательном алгоритме порядок вывода расчетных данных можно изменять. Какие команды в приведенных программах нельзя переставлять? Почему?
3. На блок-схеме представлены два блока вывода информации на экран. В чем их различие?
4. Запишите в тетради имена переменных, которые были использованы в процессе решения задания. Напишите под ними значения переменных, полученные в результате тестирования.
5. Что происходит в результате выполнения блока 3 представленного алгоритма?

6. Добавьте в программу блок вычислений, определяющий, сколько экспонатов в день удастся посмотреть посетителю.

К заданию 8.2

1. Замените в программе формулу расчета катета $b = a\sqrt{3}$ и убедитесь, что результат от этого не изменится.
2. Используя готовый каркас блок-схемы, заполните его таким образом, чтобы по новой блок-схеме вычислялись высота и площадь равнобедренной трапеции с углом при основании 45° (рис. 8.5). Задание выполните в тетради.

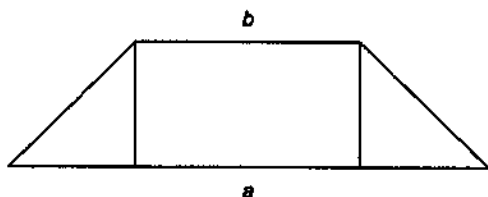


Рис. 8.5. Чертеж для вычисления высоты и площади равнобедренной трапеции

3. Напишите текст программы на Кумире или другом языке для полученной блок-схемы.
4. Что надо изменить в условии задачи, чтобы расширить границы применимости алгоритма?
5. Из пояснительного рисунка видно, что $a > b$. Что произойдет, если при вводе a и b это соотношение будет нарушено?

К заданию 8.3

1. При расчетах радиуса и объемов используется константа 3,1415926. Что нужно изменить в программе, чтобы не набирать ее каждый раз заново?
2. В примере программы на языке Кумир тип используемых переменных описан следующим образом: вещь $г$, s , $vshara$, $vcil$, k . Что означает эта запись? Почему для переменных выбран такой тип?
3. В формуле вычисления объема шара используется формула $V = r^3$. В примерах программ на разных языках она записана по-разному. Есть ли здесь ошибки? Объясните, что означают разные записи? Придумайте такой вид записи, который будет верен для всех языков.
4. Можно ли изменить последовательность операторов расчета?

Тема 8.2 Разветвляющиеся алгоритмы

Выполнив задания этой темы, вы научитесь:

- ➡ использовать различные структуры ветвления;
- ➡ использовать простые и сложные условия;
- ➡ использовать вложенные ветвления;
- ➡ использовать структуру множественного выбора при наличии более чем двух ситуаций;
- ➡ тестировать алгоритм в пошаговом режиме для проверки всех его ветвей.

Для представления разветвляющегося алгоритма в виде блок-схемы используются блоки принятия решения. Для представления разветвляющегося алгоритма в виде программы используются условные операторы и операторы выбора (если вариантов выбора больше двух).

Задание 8.4

Требуется разработать алгоритм проверки принадлежности введенного числа данной арифметической прогрессии. Прогрессия задается двумя последовательными членами.

Словесный алгоритм

Начало алгоритма

1. Введите два последовательных члена арифметической прогрессии.
2. Введите произвольное целое число.
3. Найдите разность (d) арифметической прогрессии.
4. Найдите разность между введенным числом и членом арифметической прогрессии.
5. Найдите остаток от деления нацело найденной разности на d .
6. Организуйте проверку остатка:
 - если остаток от деления равен 0, выведите сообщение: «Число принадлежит рассматриваемой арифметической прогрессии»;

- иначе выведите сообщение: «Число не принадлежит рассматриваемой арифметической прогрессии».

Конец алгоритма

Алгоритм в виде блок-схемы

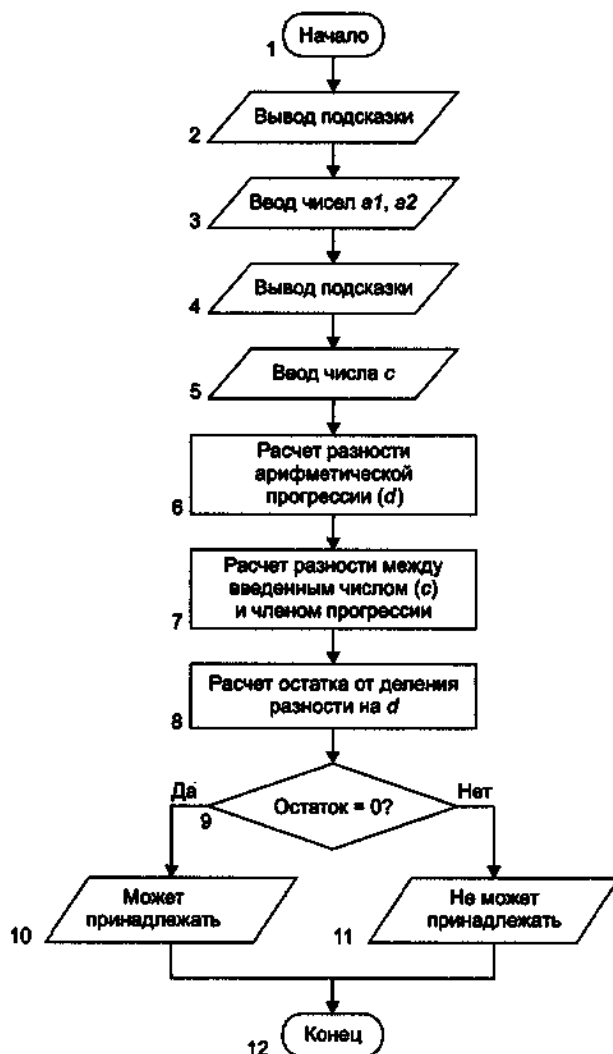


Рис. 8.6. Блок-схема алгоритма (к заданию 8.4)

Алгоритм в виде программы

В табл. 8.7 приведена программа к заданию на алгоритмическом языке Кумир. В табл. 8.8 приведены тексты программ на языках программирования Паскаль и Visual Basic.

Таблица 8.7. Программа на Кумире с пояснениями (к заданию 8.4)

№	Программа на Кумире	Пояснения	Тестирование					
			Тест 1		Тест 2		Тест 3	
1	алг Прогрессия	Начало программы						
	нач цел a1, a2, c, d, razn, ost	Описание переменных						
2	вывод "Введите два последовательных члена арифметической прогрессии"	Приглашение к вводу чисел						
3	ввод a1, a2	Ввод чисел	3	7	3	7	3	7
4	вывод "Введите произвольное целое число"	Приглашение к вводу числа						
5	ввод c	Ввод числа	18		11		-1	
6	d := a2 - a1	Расчет разности арифметической прогрессии	4		4		4	
7	razn := c - a1	Расчет разности между введенным числом и членом арифметической прогрессии	15		8		-4	
8	ost := mod(razn, d)	Расчет остатка от деления разности на d	3		0		0	

№	Программа на Кумире	Пояснения	Тестирование		
			Тест 1	Тест 2	Тест 3
9	если $ost = 0$	Проверка равенства остатка 0	Нет	Да	Да
10	то	То			
	вывод "Число", с, " принадлежит рассматриваемой арифметической прогрессии:", a1, ", ", a2	Вывод текста на экран	-	+	+
11	иначе	Иначе			
	вывод "Число", с, " не принадлежит рассматриваемой арифметической прогрессии:", a1, ", ", a2	Вывод текста на экран	+		
	все	Конец Если			
12	кон	Конец алгоритма			

Таблица 8.8. Примеры программ на Паскале и Visual Basic (к заданию 8.4)

№	Программа на Паскале	Программа на Visual Basic
1	program Progress;	Private Sub Progress_Click()
	var a1, a2, c, d, razn, ost: integer;	Dim a1 As Integer, a2 As Integer, c As Integer, d As Integer, razn As Integer, ost As Integer
	begin	
2	write('Введите два последовательных члена арифметической прогрессии: ');	a1 = Val(InputBox("Введите 1-й член арифметической прогрессии", "Прогрессия")) a2 = Val(InputBox("Введите 2-й член арифметической прогрессии", "Прогрессия"))
3	readln(a1, a2);	

№	Программа на Паскале	Программа на Visual Basic
4	write('Введите произвольное целое число: ');	c = Val(InputBox("Введите произвольное целое число", "Прогрессия"))
5	readln(c);	
6	d := a2 - a1;	d = a2 - a1
7	razn := c - a1;	razn = c - a1
8	ost := razn mod d;	ost = razn Mod d
9	if ost = 0 then	If ost = 0 Then
10	writeln('Число ', c, ' принадлежит арифм. прогрессии: ', a1, ', ', a2)	Print "Число "; c; " принадлежит арифм. прогрессии: "; a1; ", "; a2
	else	Else
11	writeln('Число ', c, ' не принадлежит арифм. прогрессии: ', a1, ', ', a2);	Print "Число "; c; " не принадлежит арифм. прогрессии: "; a1; ", "; a2
		End If
12	end.	End Sub

Задание 8.5

Из «Арифметики» таджикского ученого Авиценны (X–XI вв.) известно следующее свойство целых чисел: если число, будучи разделено на 9, дает в остатке 1 или 8, то квадрат этого числа, деленный на 9, даст 1. Требуется подтвердить верность свойства или опровергнуть его.

Словесный алгоритм

Начало алгоритма

1. Введите целое число.
2. Найдите остаток от деления этого числа на 8.
3. Организуйте проверку остатка на равенство 1 или 8:
 - если остаток от деления равен 1 или 8, то:
 - а) найдите квадрат введенного числа;
 - б) найдите остаток от деления квадрата числа на 9;

в) организуйте проверку остатка от деления: если остаток равен 1, то выведите сообщение «Свойство верно», иначе выведите сообщение «Свойство не верно»;

- иначе (остаток от деления не равен 1 и остаток от деления не равен 8) выведите сообщение «Остаток от деления $\neq 1$ и Остаток от деления $\neq 8$ ».

Конец алгоритма

Алгоритм в виде блок-схемы

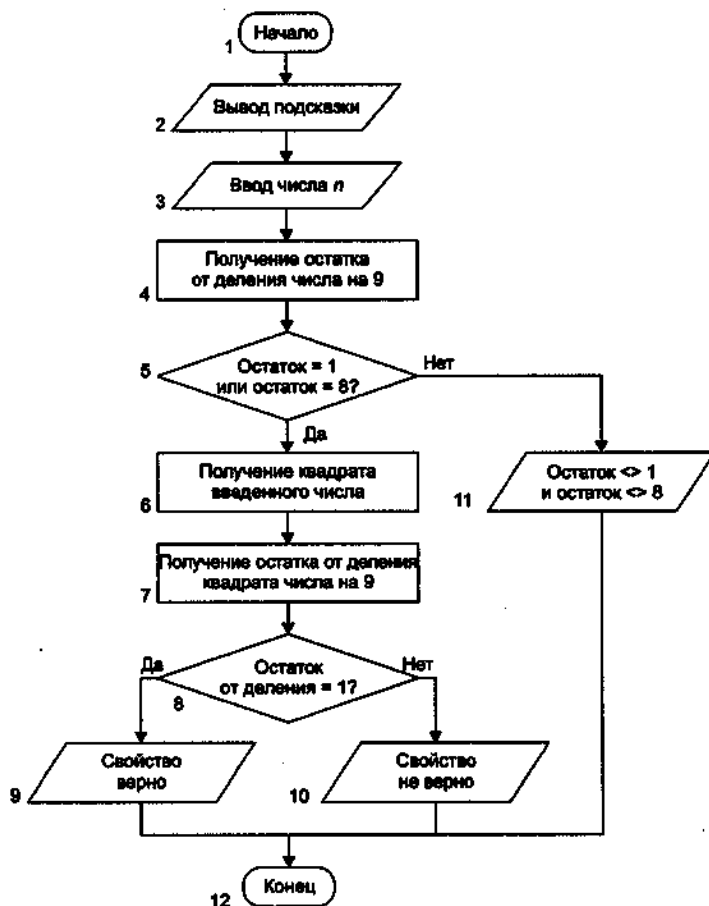


Рис. 8.7. Блок-схема алгоритма (к заданию 8.5)

Алгоритм в виде программы

В табл. 8.9 приведена программа к заданию на алгоритмическом языке Кумир. В табл. 8.10 приведены тексты программ на языках Паскаль и Visual Basic.

Таблица 8.9. Программа на Кумире с пояснениями (к заданию 8.5)

№	Программа на Кумире	Пояснения	Тестирование		
			Тест 1	Тест 2	Тест 3
1	алг Свойства_чисел	Начало программы			
	нач цел n, kvn, ost1, ost2	Описание переменных			
2	вывод "Введите целое число"	Приглашение к вводу числа			
3	ввод n	Ввод числа	14	17	10
4	ost1 := mod(n, 9)	Получение остатка от деления на 9	5	8	???
5	если (ost1 = 1) или (ost1 = 8)		Нет	Да	???
	то				
6	kvn := n * n			289	???
7	ost2 := mod(kvn, 9)			1	???
8	если ost2 = 1	Проверка, равен ли 1 остаток от деления квадрата числа на 9	-	Да	???
	то	То			
9	вывод "Свойство верно"	Вывод текста на экран	-	+	???
10	иначе	Иначе			

№	Программа на Кумире	Пояснения	Тестирование		
			Тест 1	Тест 2	Тест 3
	вывод "Свойство не верно"	Вывод текста на экран	—	—	???
	все	Конец Если (8)			
11	иначе	Иначе			
	вывод "Остаток ≤ 1 и Остаток ≤ 8 "	Вывод текста на экран	+	—	???
	все	Конец Если (5)			
12	кон				

Таблица 8.10. Примеры программ на Паскале и Visual Basic
(к заданию 8.5)

№	Программа на Паскале	Программа на Visual Basic
1	program Avicenna;	Private Sub Avicenna_Click()
	var n, kvn, ost1, ost2: integer;	Dim n As Integer, kvn As Integer, ost1 As Integer, ost2 As Integer
	begin	
2	write('Введите целое число: ');	n = Val(InputBox("Введите целое число", "Свойства"))
3	readln(n);	
4	ost1 := n mod 9;	ost1 = n Mod 9
5	if (ost1 = 1) or (ost1 = 8) then	If (ost1 = 1) Or (ost1 = 8) Then
	begin	
6	kvn := n * n;	kvn = n * n
7	ost2 := kvn mod 9;	ost2 = kvn Mod 9
8	if ost2 = 1 then	If ost2 = 1 Then

№	Программа на Паскале	Программа на Visual Basic
9	writeln('Свойство верно')	Print "Свойство верно"
	else	Else
10	writeln('Свойство не верно')	Print "Свойство не верно"
	end	End If
	else	Else
11	writeln('Остаток ≤ 1 и Остаток ≤ 8 ');	Print "Остаток ≤ 1 и Остаток ≤ 8 "
		End If
12	end.	End Sub

Задание 8.6

Требуется определить тип треугольника по двум введенным углам. При выполнении задания необходимо учесть ситуации некорректного ввода данных, например: 90, 90 или 120, 80.

Словесный алгоритм

Начало алгоритма

- Введите два угла треугольника в градусах.
- Организуите проверку типа треугольника:
 - если сумма двух углов меньше 180° , то вычислите значение третьего угла и рассмотрите три ситуации:
 - если все углы острые, то выведите сообщение «Треугольник остроугольный»;
 - если один из углов равен 90° , то выведите сообщение «Треугольник прямоугольный»;
 - в противном случае выведите сообщение «Треугольник тупоугольный»;
 - иначе (если сумма углов больше 180°) выведите сообщение «Некорректный ввод».

Конец алгоритма

Алгоритм в виде блок-схемы

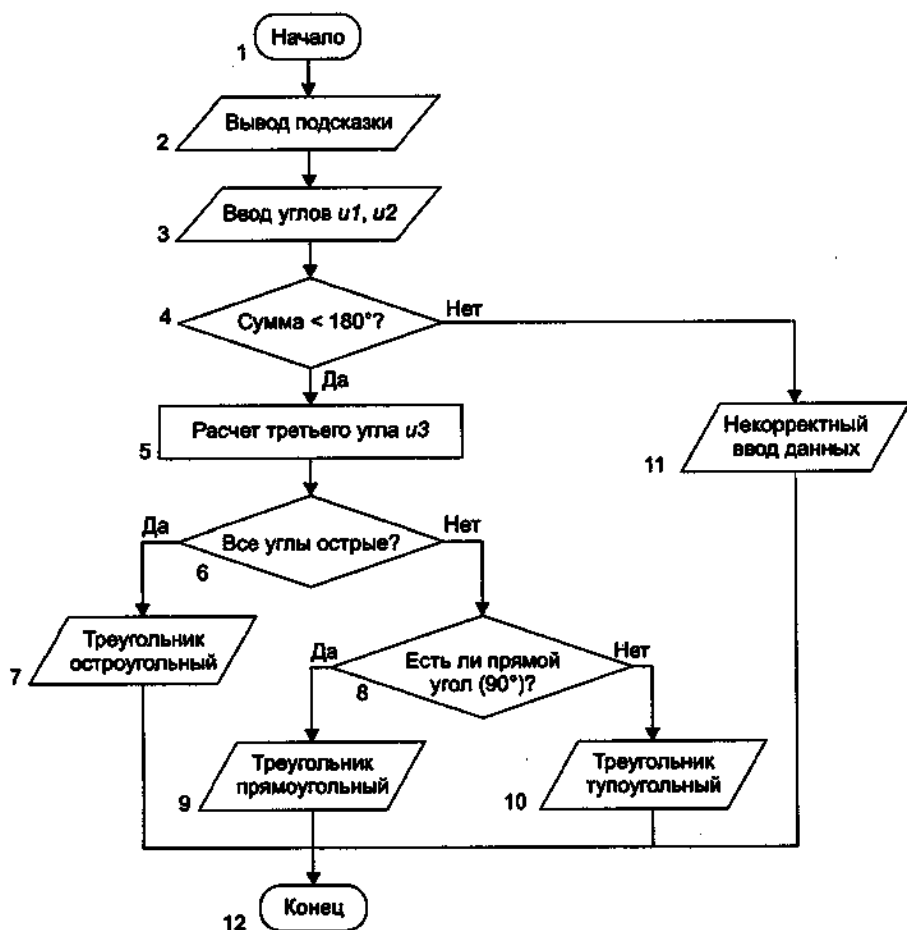


Рис. 8.8. Блок-схема алгоритма (к заданию 8.6)

Фраза «один из углов равен 90° » в словесном алгоритме понятна человеку. Для компьютера ее следует детализировать, рассмотрев три ситуации (для каждого из углов u_1 , u_2 , u_3). На алгоритмическом языке эта проверка может выглядеть следующим образом: $((u_1 = 90) \text{ и } (u_2 < 90) \text{ и } (u_3 < 90))$ или $((u_2 = 90) \text{ и } (u_1 < 90) \text{ и } (u_3 < 90))$ или $((u_3 = 90) \text{ и } (u_1 < 90) \text{ и } (u_2 < 90))$.

и $(u1 \neq 90)$ и $(u3 \neq 90)$) или $((u3 = 90)$ и $(u1 \neq 90)$ и $(u2 \neq 90)$)).

Чтобы упростить проверку, в алгоритм должен быть введен блок, обеспечивающий условие «сумма углов = 180». После этого достаточно рассмотреть выполнение условия « $(u1 = 90)$ или $(u2 = 90)$ или $(u3 = 90)$ ».

Алгоритм в виде программы

В табл. 8.11 приведена программа к заданию на алгоритмическом языке Кумир. В табл. 8.12 приведены тексты программ на языках Паскаль и Visual Basic.

Таблица 8.11. Программа на Кумире с пояснениями (к заданию 8.6)

№	Программа на Кумире	Пояснения	Тестирование					
			Тест 1		Тест 2		Тест 3	
1	алг Тип_треуголь- ника	Начало программы						
	нач цел u1, u2, u3	Описание переменных						
2	вывод "Введите два угла треуголь- ника в граду- сах"	Приглашение к вводу углов						
3	ввод u1, u2	Ввод углов	100	80	120	45	70	60
4	если $(u1 + u2) < 180$	Если сумма двух углов меньше 180°	Нет		Да		Да	
	то	То						
5	$u3 := 180 - (u1 + u2)$	Расчет третьего угла	-		15		50	
	выбор	Выбор из трех ситуаций	-					

№	Программа на Кумире	Пояснения	Тестирование		
			Тест 1	Тест 2	Тест 3
6	при ($u1 < 90$) и ($u2 < 90$) и ($u3 < 90$):	Если все углы острые	–	Нет	Да
7	вывод "Треугольник остроугольный"	Вывод текста на экран	–	–	+
8	при ($u1 = 90$) или ($u2 = 90$) или ($u3 = 90$):	Если какой-то из углов равен 90°	–	Нет	–
9	вывод "Треугольник прямоугольный"	Вывод текста на экран	–	–	–
	иначе	Иначе			
10	вывод "Треугольник тупоугольный"	Вывод текста на экран	–	+	–
	все	Конец Выбора			
	иначе	Иначе			
11	вывод "Некорректный ввод"	Вывод текста на экран	+	–	–
	все	Конец Если (4)			
12	кон	Конец алгоритма			

Таблица 8.12. Примеры программ на Паскале и Visual Basic (к заданию 8.6)

№	Программа на Паскале	Программа на Visual Basic
1	program Treug;	Private Sub Treug_Click()
	var u1, u2, u3: integer;	Dim u1 As Integer, u2 As Integer, u3 As Integer
	begin	

№	Программа на Паскале	Программа на Visual Basic
2	write('Введите два угла треугольника: ');	u1 = Val(Inputbox("Введите 1-й угол треугольника"))
3	readln(u1, u2);	u2 = Val(Inputbox("Введите 2-й угол треугольника"))
4	if (u1 + u2) < 180 then begin	If (u1 + u2) < 180 Then
5	u3 := 180 - (u1 + u2);	u3 = 180 - (u1 + u2)
6	if (u1 < 90) and (u2 < 90) and (u3 < 90) then	If (u1 < 90) And (u2 < 90) And (u3 < 90) Then
7	writeln('Треугольник остроугольный')	Print "Треугольник остроуголь- ный"
8	else if (u1 = 90) or (u2 = 90) or (u3 = 90) then	ElseIf u1 = 90 Or u2 = 90 Or u3 = 90 Then
9	writeln('Треугольник прямоугольный')	Print "Треугольник прямоуголь- ный"
	else	Else
10	writeln('Треугольник тупоугольный');	Print "Треугольник тупоуголь- ный"
	end	End If
	else	Else
11	writeln('Некорректный ввод');	Print "Некорректный ввод"
		End If
12	end.	End Sub

Контрольные вопросы и задания

К заданию 8.4

1. Какое сообщение будет получено в результате выполнения алгоритма и программ, если введенное число s будет равно $a1$ или $a2$?

2. Могут ли быть введены различающиеся по знаку или отрицательные числа a_1 и a_2 ?
3. Что надо изменить в блок-схеме и программе, чтобы они работали с тремя последовательными членами геометрической прогрессии (a_1 , a_2 — являются членами, s — следует проверить)?
4. Найдите на блок-схеме (см. рис. 8.6) блок ветвления и определите, является ли ветвление полным или нет.

К заданию 8.5

1. Заполните таблицу тестирования для числа 10 (см. табл. 8.9).
2. Почему в 7-й строке табл. 8.9 тестирования (первый тест) ничего нет?
3. Достаточно ли представленных в табл. 8.9 тестов, чтобы проверить все ситуации, которые могут возникнуть при выполнении программ (все ветви алгоритма)?
4. Можно ли объединить оба условия проверки (пп. 5 и 8) в одно сложное условие? Напишите логическое выражение для подобной проверки.
5. Составьте самостоятельно фрагмент блок-схемы алгоритма для приема менеджера на работу по следующим условиям:
 - возраст от 30 до 40 лет;
 - знание персонального компьютера или стаж работы по специальности не менее 5 лет.

К заданию 8.6

1. Почему при формировании сложного условия (см. табл. 8.11, п. 6) использована логическая связка И, а не ИЛИ?
2. Почему при формировании сложного условия (см. табл. 8.11, п. 8) использована логическая связка ИЛИ, а не И?
3. В алгоритме и программе тупоугольный треугольник определяется по веткам «иначе» (не прямоугольный и не остроугольный). Напишите самостоятельно сложное условие, определяющее, является ли треугольник тупоугольным.
4. Выполните тестирование программы для угла 90° .
5. Дополните алгоритм и программу блоком проверки положительных значений углов.

Тема 8.3 Циклические алгоритмы. Цикл с предусловием

Выполнив задания этой темы, вы научитесь:

- ➡ использовать цикл с предусловием для организации повторяющихся действий;
- ➡ тестировать циклические алгоритмы;
- ➡ использовать цикл Пока как универсальный для решения разного вида задач;
- ➡ использовать переменные разного типа для организации цикла.

Цикл с предусловием относится к циклам с неизвестным числом повторений.

В цикле с предусловием сначала проверяется выполнение условия продолжения цикла. Если условие истинно (да, true), то выполняется тело цикла, а иначе (нет, False) цикл завершается. Особенностью этого цикла является то, что если при 1-й проверке условие ложно, то тело цикла не выполнится ни разу.

Специального блока для реализации цикла с предусловием в блок-схемах нет.

Блок-схема алгоритма реализуется при помощи блока принятия решения, выполнения действий и др.

Блок-схемы алгоритмов, содержащих циклы, легко узнаваемы, так как содержат возврат на предыдущие блоки («петлю»). Во всех языках программирования есть специальные операторы, реализующие этот цикл.

Задание 8.7

Существуют простые правила определения делимости чисел на числа 3, 4, 5:

- ◆ *на 3 без остатка делятся числа, сумма цифр которых делится на 3;*

- ◆ на 4 без остатка делятся числа, у которых две последние цифры составляют число, делящееся на 4;
- ◆ на 5 без остатка делятся числа, заканчивающиеся на цифры 5 и 0.

Впервые эти правила были сформулированы в знаменитой «Книге Абака» итальянского математика Леонардо Фибоначчи (XII век). Требуется проверить делимость введенных чисел на 3 по первому из перечисленных правил.

Словесный алгоритм

Начало алгоритма

1. Введите число.
2. Пока цифры числа не закончатся:
 - а) выделите очередную цифру как остаток от деления на 10;
 - б) прибавьте эту цифру к общей сумме;
 - в) удалите обработанную цифру из числа, получив новое число в виде частного от деления на 10.
3. Проверьте, делится ли полученная сумма на 3 без остатка:
 - если делится, то сообщите, что исходное число делится на 3;
 - иначе сообщите, что исходное число не делится на 3.

Конец алгоритма

Алгоритм в виде блок-схемы

На рис. 8.9 приведена блок-схема, составленная по словесному алгоритму.

Алгоритм в виде программы

В табл. 8.13 приведена программа к заданию на алгоритмическом языке Кумир. В табл. 8.14 приведены тексты программ на языках Паскаль и Visual Basic.

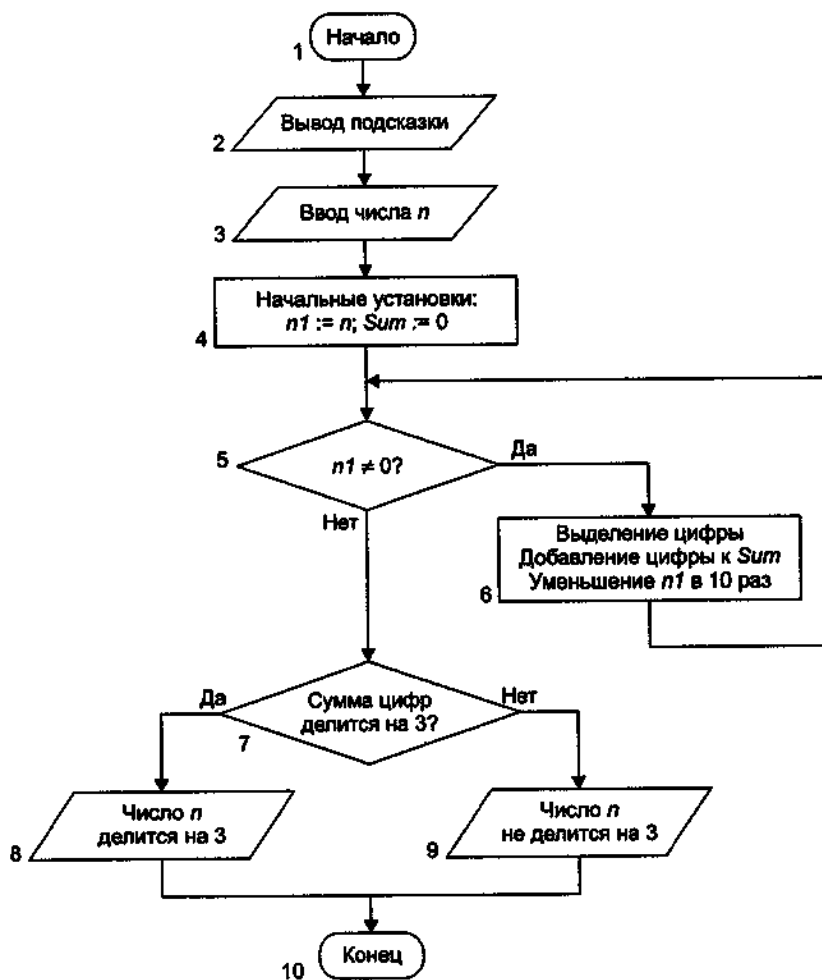


Рис. 8.9. Блок-схема алгоритма (к заданию 8.7)

Таблица 8.13. Программа на Кумире с пояснениями (к заданию 8.7)

№	Программа на Кумире	Пояснения	Тестирование			
1	алг Делимость_на_3	Начало программы	Введите число			
	нач цел n, n1, sum, cifra	Описание переменных				
2	вывод "Введите число"	Приглашение к вводу				
3	ввод n	Ввод проверяемого числа	528			
4	n1 := n	Начальные установки: чтобы сохранить значение n, запоминаем его в n1; обнуляем сумму цифр	528			
	sum := 0		0			
5	нц пока n1 <> 0	Пока введенное число не равно 0:	528 <> 0 Да	52 <> 0 Да	5 <> 0 Да	0 <> 0 Нет
6	cifra := mod(n1, 10)	Выделение очередной цифры	8	2	5	—
	sum := sum + cifra	Добавление ее к сумме цифр	8	10	15	—
	n1 := div(n1, 10)	Удаление цифры из числа	52	5	0	—
	кц	Конец цикла	К п. 5	К п. 5	К п. 5	—
7	если mod(sum, 3) = 0	Если сумма делится на 3 (остаток от деления по модулю 3 равен 0)	—	—	—	mod(15, 3) = 0 Да
	то	То				
8	вывод n, "делится на 3"	Вывод числа n и сообщения "Число делится на 3"	—	—	—	+

№	Программа на Кумире	Пояснения	Тестирование			
	иначе	в противном случае:				
9	вывод n, " не делится на 3"	?	-	-	-	
	все	Конец Если				
10	кон	Конец программы				

Таблица 8.14. Примеры программ на Паскале и Visual Basic (к заданию 8.7)

№	Программа на Паскале	Программа на Visual Basic
1	program Delimost;	Private Sub Delen_Click()
	var n, n1, sum: word; cifra: 0..9;	Dim n As Integer, n1 As Integer, cifra As Integer, sum As Integer
	begin	
2	write('Введите число: ');	n = Val(InputBox ("Введите число"))
3	readln(n);	
4	n1 := n;	n1 = n
	sum := 0;	sum = 0
5	while n1 <> 0 do	Do While n1 <> 0
	begin	
6	cifra := n1 mod 10;	cifra = n1 Mod 10
	sum := sum + cifra;	sum = sum + cifra
	n1 := n1 div 10	n1 = n1 \ 10
	end;	Loop
7	if sum mod 3 = 0 then	If sum Mod 3 = 0 Then
8	writeln('Число ', n, ' делится на 3')	Print "Число " & n & " делится на 3"
	else	Else
9	writeln('Число ', n, ' не делится на 3')	Print "Число " & n & " не делится на 3"
		End If
10	end.	End Sub

Задание 8.8

Леонардо из Пизы, известный как Фибоначчи, был первым из великих математиков Европы позднего Средневековья. Числовой ряд, носящий в наше время имя Фибоначчи, вырос из проблемы с кроликами, которую Фибоначчи изложил в своей «Книге Абака», написанной в 1202 году. Он выглядит так:

1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144,...

В этом ряду каждое следующее число, начиная с третьего, равно сумме двух предыдущих (рис. 8.10). Требуется составить алгоритм проверки принадлежности введенного числа ряду чисел Фибоначчи.

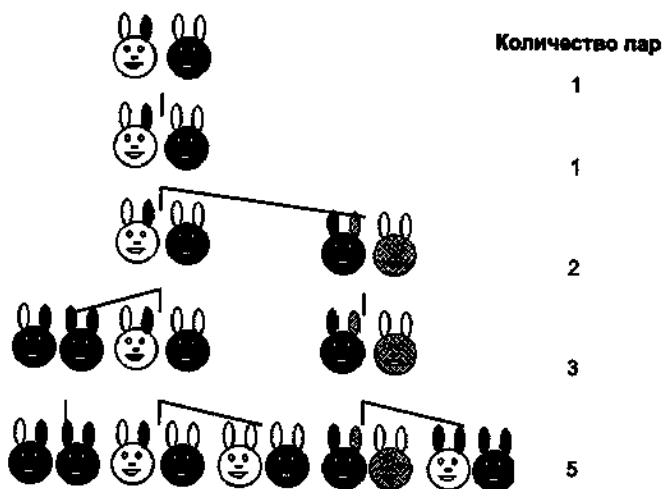


Рис. 8.10. К заданию 8.8

Словесный алгоритм

Начало алгоритма

1. Введите число.
2. Установите значение первых трех чисел Фибоначчи: 1, 1, $1 + 1$ (сумма двух предыдущих чисел).
3. Пока введенное число больше очередного числа Фибоначчи, возьмите два последних числа и получите из них новое число Фибоначчи.

4. Если число Фибоначчи, полученное по выходу из цикла, равно введенному (n) или было введено число $n = 1$, то сообщите «Да» (введено число Фибоначчи), в противном случае — сообщите «Нет» (введенное число не является числом Фибоначчи).

Конец алгоритма

Алгоритм в виде блок-схемы

На рис. 8.11 приведена блок-схема, составленная по словесному алгоритму.

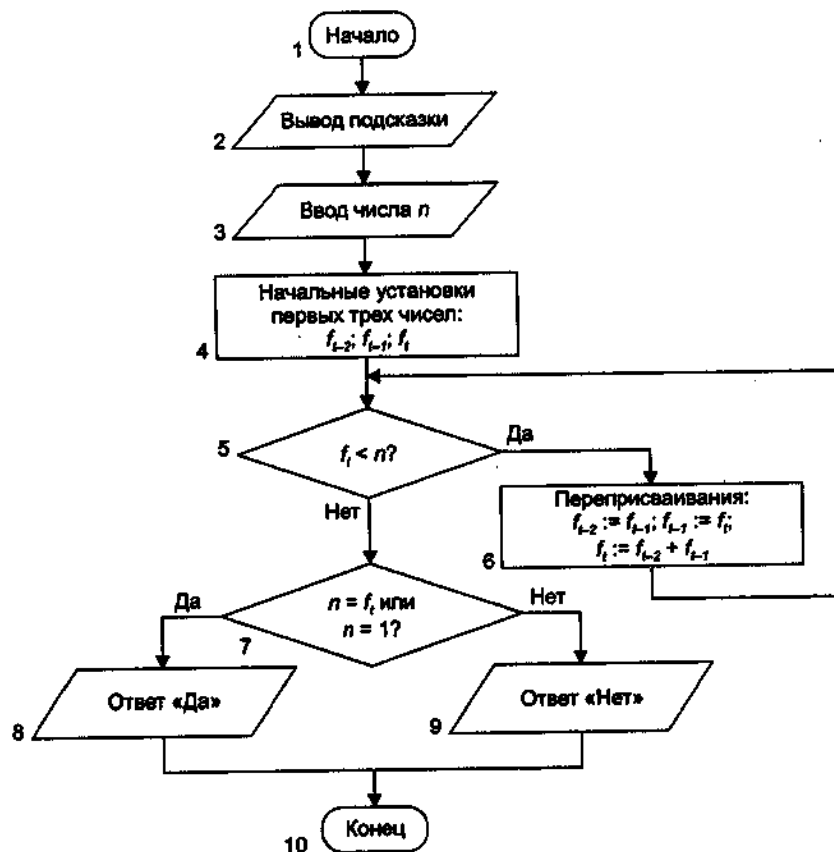


Рис. 8.11. Блок-схема алгоритма (к заданию 8.8)

Алгоритм в виде программы

В табл. 8.15 приведена программа на алгоритмическом языке Кумир. В табл. 8.16 приведены тексты программы на языках программирования Паскаль и Visual Basic.

Таблица 8.15. Программа на Кумире с пояснениями (к заданию 8.8)

№	Программа на Кумире	Пояснения	Тестирование				
1	алг Фибоначчи	Начало алгоритма					
	нач цел f1, f2, f3, n	Описание переменных					
2	вывод "Введите число"	?	Введите число				
3	ввод n	?	13				
4	f1 := 1	Начальные установки: 3 первых числа Фибоначчи	1				
	f2 := 1		1				
	f3 := f1 + f2		2				
5	нц пока f3 < n	Пока введенное число больше текущего значения числа Фибоначчи	2<13 Да	3<13 Да	5<13 Да	8<13 Да	13<13 Нет
6	f1 := f2	Переприсваивание и нахождение следующего числа Фибоначчи	1	2	3	5	—
	f2 := f3		2	3	5	8	—
	f3 := f1 + f2		3 К п.5	5 К п.5	8 К п.5	13 К п.5	—
	кц	Конец цикла Пока					

№	Программа на Куирире	Пояснения	Тестирование				
7	если ($n = f3$) или ($n = 1$)	Если текущее число Фибоначчи равно введенному или введенное число = 1					13-13 Да
8	то	То сообщение "Да" (число является числом Фибоначчи)					
	вывод n, " является числом Фибоначчи"						+
9	иначе	Иначе сообщение "Нет" (число не является числом Фибоначчи)					
	вывод n, " не является числом Фибоначчи"						
	все	Конец Если					
10	кон	Конец алгоритма					

Таблица 8.16. Примеры программ на Паскале и Visual Basic
(к заданию 8.8)

№	Программа на Паскале	Программа на Visual Basic
1	program Fib;	Private Sub Fib_Click()
	var f1, f2, f3, N: word;	Dim n As Integer, f1 As Integer, f2 As Integer, f3 As Integer
	begin	

№	Программа на Паскале	Программа на Visual Basic
2	write('Введите проверяемое число: ');	n = Val(InputBox("Введите число", "Число"))
3	readln(n);	
4	f1 := 1; f2 := 1; f3 := f1 + f2;	f1 = 1; f2 = 1; f3 = f1 + f2
5	while f3 < n do	Do While f3 < n
6	begin f1 := f2; f2 := f3; f3 := f1 + f2 end;	f1 = f2 f2 = f3 f3 = f1 + f2 Loop
7	if (n = f3) or (n = 1) then	If (n = f3) Or (n = 1) Then
8	writeln('Да')	Print n ; "Да"
	else	Else
9	writeln('Нет');	Print n; "Нет"
		End If
10	end.	End Sub

Задание 8.9

В телевизионном эфире в США проводится марафон, цель которого — сбор средств для поддержки незащищенных слоев населения. Во время эфира слушатели отправляют на адрес студии телеграммы с указанием суммы пожертвования и цели, на реализацию которой пойдет эта сумма, например: «Посылаю 521\$ в фонд помощи безработным. Смит».

В конце марафона должна быть объявлена общая сумма пожертвований. Требуется составить алгоритм выделения из текста конкретной телеграммы числовых данных, заканчивающихся знаком \$, для дальнейшего суммирования.

Словесный алгоритм

Начало алгоритма

1. Запросите текст телеграммы.
2. Поместите текст в строку *st*.
3. Определите длину строки *n*.
4. Пока не закончатся все символы в строке или пока не встретится знак \$, рассмотрите три ситуации:
 - а) если символ — цифра, то получите цифровой эквивалент символа; добавьте полученную цифру в следующую позицию числа, из которого будет сформирована сумма пожертвования, и перейдите к следующему символу, увеличив счетчик символов: $i = i + 1$;
 - б) если символ — «\$», то установите признак окончания цифр *d*, который будет признаком досрочного выхода из цикла;
 - в) если это другой символ (буква, знак препинания и т. п.), то перейдите к следующему символу, увеличив счетчик символов: $i = i + 1$.
5. Проанализируйте признак окончания цифр *d*. Если он равен 1, то сообщите сумму пожертвования, если нет — сообщите, что указания о сумме пожертвования в телеграмме нет.

Конец алгоритма

Алгоритм в виде блок-схемы

На рис. 8.12 приведена блок-схема, составленная по словесному алгоритму.

Алгоритм в виде программы

В табл. 8.17 приведена программа на алгоритмическом языке Кумир. В табл. 8.18 приведены тексты программы на языках программирования Паскаль и Visual Basic.

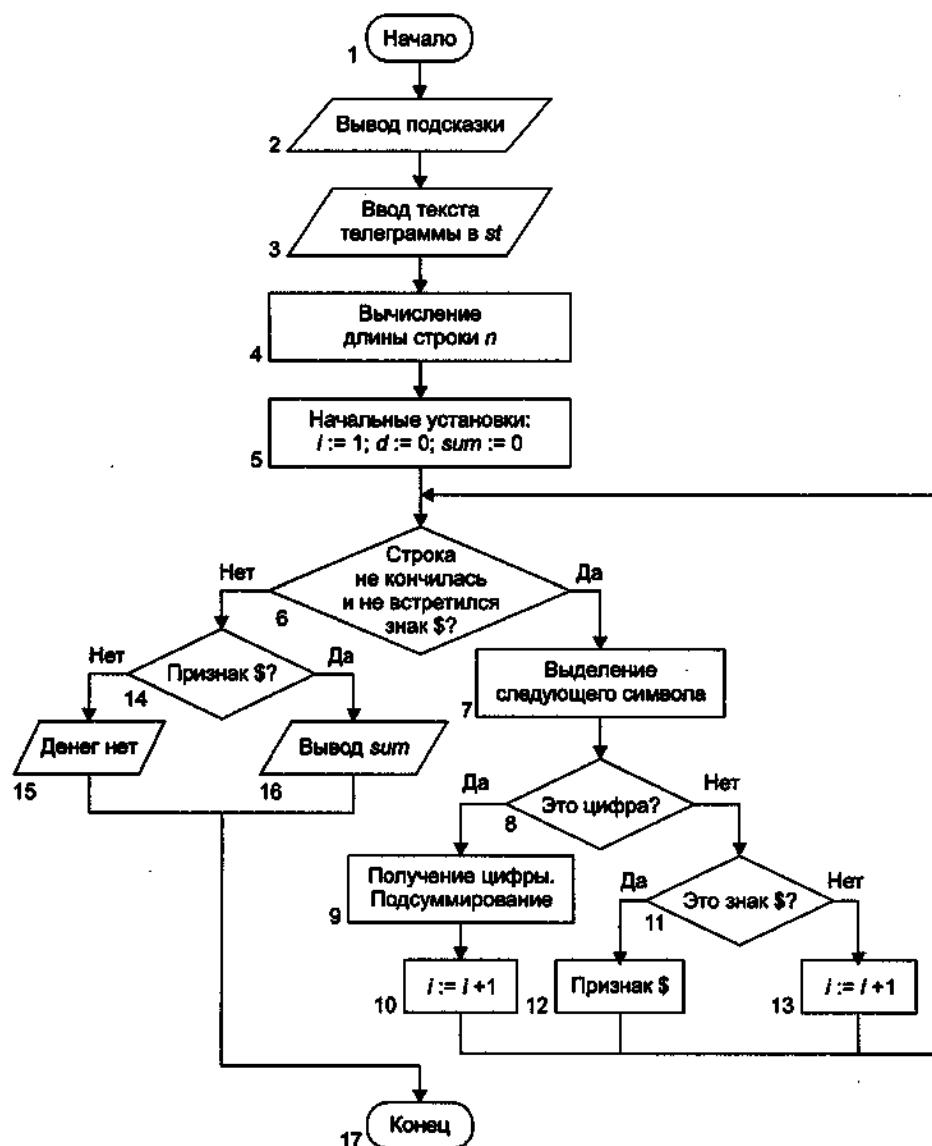


Рис. 8.12. Блок-схема алгоритма (к заданию 8.9)

Таблица 8.17. Программа на Кумире (к заданию 8.9)

№	Программа на Кумире	Пояснения
1	алг Марафон	Начало алгоритма
	нач лит st, сим sim, цел sum, ch, i, n, d	Описание переменных
2	вывод "Введите строку"	Вывод приглашения
3	ввод st	Ввод текста телеграммы
4	n := длин(st)	Подсчет длины строки
5	i := 1 d := 0 sum := 0	Начальные установки: счетчик символов = 1; обнуление признака \$; обнуление суммы взноса
6	нц пока (i <= n) и (d = 0)	Начало цикла: пока не закончилась строка и не появился \$
7	sim := st[i]	Выделение очередного символа
	выбор	Выбор одной из трех ситуаций:
8	при sim >= "0" и sim <= "9":	Если выделенный символ — цифра, то:
9	ch := цел(sim) - цел("0")	Нахождение цифрового эквивалента символа;
	sum := sum * 10 + ch	Добавление цифры в младший разряд к сдвинутым влево разрядам суммы
10	i := i + 1	Увеличение счетчика символов
11	при sim = "\$":	Если выделенный символ — \$,
12	d := 1	Установка признака \$: d = 1
13	иначе	Иначе
		Увеличение счетчика символов
	i := i + 1	
	все	Конец выбора
	кц	Конец цикла Пока
14	если d = 1	Если есть признак \$
15	то	То
	вывод "сумма в \$ =", sum	Вывод накопленной суммы

№	Программа на Кумире	Пояснения
16	иначе	Иначе
	вывод " нет суммы в \$"	Вывод сообщения, что в тексте нет упоминания о сумме взноса
	все	Конец ветвления "Если"
17	кон	Конец алгоритма

Таблица 8.18. Примеры программ на Паскале и Visual Basic (к заданию 8.9)

№	Программа на Паскале	Программа на Visual Basic
1	program Tele;	Private Sub Command1_Click()
	var st: string; sum: longint; i, ch, n: byte; sim: char; d: integer;	Dim st As String, sim As String, sum As Long, ch As Byte, i As Byte, n As Byte, d As Integer
	begin	
2	writeln('Введите текст');	st = InputBox("Введите текст ", "Сумма", "Посылаю 521\$ в фонд помощи безработным. СМит.")
3	readln(st);	
4	n := length(st);	n = Len(st) 'длина строки
5	i := 1;	i = 1
	d := 0;	d = 0
	sum := 0;	Sum = 0
6	while (i <= n) and (d = 0) do	Do While (i <= n) And (d = 0)
	begin	
7	sim := st[i];	sim = Mid(st, i, 1)
	case sim of	Select case sim
8	'0'..'9':	Case "0" to "9"
9	begin	
	ch := ord(sim) - ord('0');	ch = Asc(sim) - Asc("0")
	sum := sum * 10 + ch;	Sum = Sum * 10 + ch
10	i := i + 1	i = i + 1
	end;	
11	'\$':	Case "\$"

№	Программа на Паскале	Программа на Visual Basic
12	d := 1	d = 1
13	else	Case Else
	i := i + 1	i = i + 1
	end;	End select
	end;	Loop
14	if d = 1 then	If d = 1 Then
15	writeln('Получена сумма ', sum)	Print "Сумма в \$ = "; Sum
	else	Else
16	writeln('Нет суммы в \$');	Print "Нет суммы в \$"
		End If
17	end.	End Sub

Контрольные вопросы и задания

К заданию 8.7

1. Для чего нужны начальные установки в алгоритме?
2. Пройдите алгоритм по блок-схеме для числа 521 (см. рис. 8.9). Какое значение будет в переменных *n1* и *Sum* перед выходом из цикла?
3. Добавьте в любую из программ оператор вывода переменных *cifra* и *Sum*. Какое значение и сколько раз будет выведено для числа 222?
4. Что произойдет, если пользователь введет число *n* равным нулю?
5. Напишите комментарий к п. 9 программы на Кумире (см. табл. 8.13).

К заданию 8.8

1. Какие действия выполняются в блоке 6 (см. рис. 8.11) и соответствующем фрагменте программы на Кумире (см. табл. 8.15)?
2. Какое условие проверяется в данном алгоритме при входе в цикл с предусловием?
3. Запишите в тетради тело цикла с предусловием, используемое в данном алгоритме.

4. Придумайте и запишите пример ситуации, когда тело цикла не выполняется ни одного раза.
5. В блоке 3 алгоритма (см. рис. 8.11) было введено число 5. Сколько раз выполнится тело цикла?
6. Напишите пояснение к строкам 2 и 3 программы на Кумире (см. табл. 8.15).
- 7*. В блоке 6 программы на Кумире (см. табл. 8.15) производится переприсваивание содержимого ячеек (предыдущей, предыдущей и текущей). Можно ли поменять местами операторы присваивания? Ответ обоснуйте.
- 8*. Можно ли в данном алгоритме обойтись только двумя переменными $f1$ и $f2$?

К заданию 8.9

1. В блоке 4 блок-схемы (см. рис. 8.12) определяется длина строки n . Для чего это делается? Где далее используется эта переменная?
2. В алгоритме используется цикл с предусловием. Может ли возникнуть такая ситуация, что тело цикла не исполнится ни разу? Придумайте и запишите в тетради пример телеграммы, текст которой приведет к такой ситуации.
3. Что произойдет в алгоритме, если в адрес телемарафона придет телеграмма из России: «Я пенсионерка, но хочу пожертвовать 3 доллара в фонд помощи бездомным животным. Татьяна». Предложите вариант алгоритма, учитывающего подобную ситуацию.
4. Добавится ли к сумме число, указанное в телеграмме: «Я родился в 1900 году, средств не имею, но считаю, что нужно помогать старикам. Джон»?
- 5*. Если вы отлаживаете программы на Паскале или на Visual Basic, упростите поиск упоминания доллара (знак или текст).
6. Что означает сложное условие $sim \geq "0"$ и $sim \leq "9"$ (см. п. 8 табл. 8.17)?
7. Чем различаются ветвления на блок-схеме и в примерах программ?

Тема 8.4 Цикл с известным числом повторений

Выполнив задания этой темы, вы научитесь:

- ➡ создавать алгоритмы с известным числом повторений;
- ➡ тестировать циклический алгоритм с известным числом повторений в пошаговом режиме;
- ➡ изменять параметры цикла.

Для реализации циклического алгоритма с известным числом повторений в блок-схемах используется специальный блок (см. блок 3 на рис. 8.13). Для программной реализации цикла с известным числом повторений используются специальные операторы.

Задание 8.10

Часто требуется обработать оценки, полученные учащимися или студентами в результате каких-либо испытаний (контрольной работы, экзамена). Обработка, как правило, сводится к подсчету количества хороших (или плохих) оценок, среднего балла, и к нахождению максимального показателя. Требуется смоделировать процесс ввода оценок с клавиатуры с одновременным набором статистики.

Словесный алгоритм

Начало алгоритма

1. Для каждого из учащихся сделайте следующие действия:
 - а) введите его оценку;
 - б) приплюсуйте ее к сумме оценок (для дальнейшего подсчета средней оценки);
 - в) сравните оценку с текущим максимумом: если они равны, то увеличьте счетчик максимальных оценок; иначе, если оценка больше текущего максимума, замените текущий максимум новой оценкой, а счет лучших оценок начните заново.
2. Сообщите данные по накопленной статистике.

Конец алгоритма

Алгоритм в виде блок-схемы

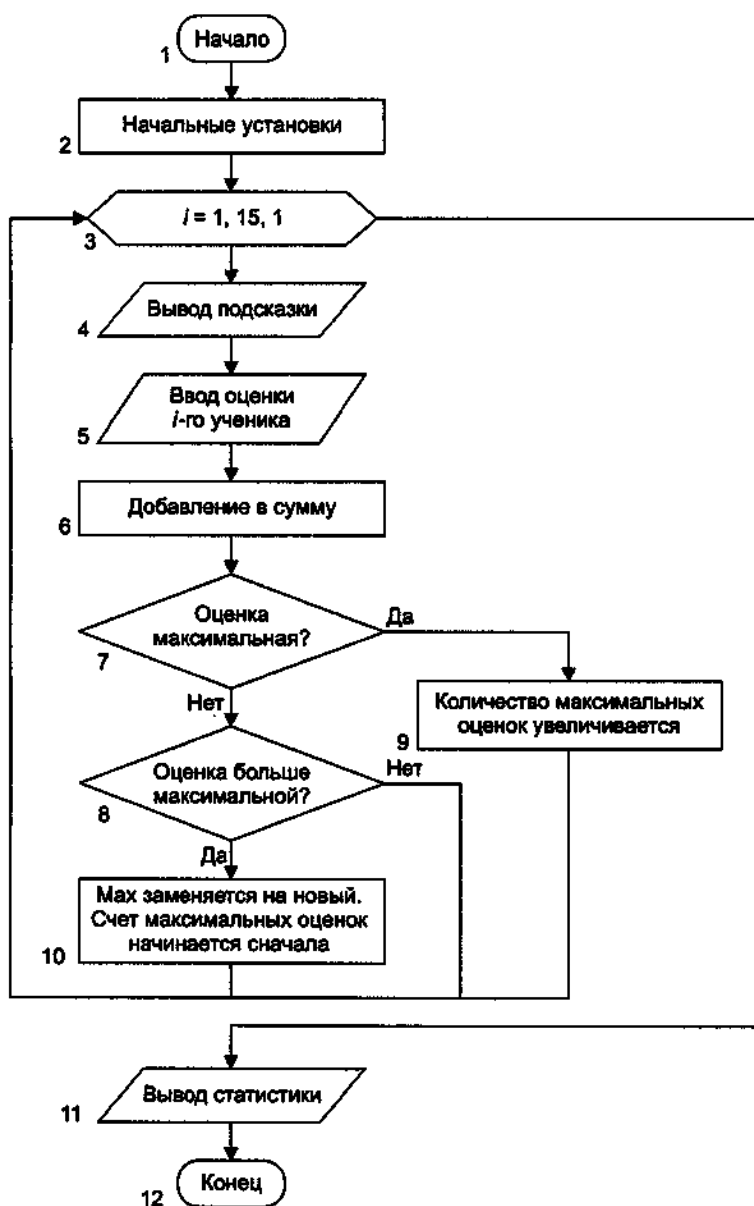


Рис. 8.13. Блок-схема алгоритма (к заданию 8.10)

Алгоритм в виде программы

В табл. 8.19 приведена программа на алгоритмическом языке Кумир. В табл. 8.20 приведены тексты программы на языках программирования Паскаль и Visual Basic.

Таблица 8.19. Программа на Кумире с пояснениями (к заданию 8.10)

№	Программа на Кумире	Пояснения	Тестирование				
1	алг Оценки	Начало программы					
	нач цел i, ma, sum, kolmax, otc	Описание переменных					
2	ma := 1; sum := 0; kolmax := 0	Начальные установки. Текущий максимум устанавливается заведомо минимальным, чтобы первая же оценка >1 его заменила					
3	нц для i от 1 до 15	Цикл для i от 1 до 15	i				
			1	2	3	4	5
4	вывод "Введите оценку", i, "-го ученика"	Приглашение к вводу очередной оценки					
5	ввод otc	Ввод оценки	2	3	5	4	5
6	sum := sum + otc	Накопление суммы оценок	2	5	10	14	19
7	если otc = ma	Если введенная оценка равна той, что считается максимальной	2=1 Нет	2=3 Нет	3=5 Нет	4=5 Нет	5=5 Да
	то	То					

№	Программа на Кумире	Пояснения	Тестирование				
8	kolmax := kolmax + 1	Увеличивается счетчик макси- мальных чисел	-	-	-	-	2
	иначе	В противном случае					
9	если otc > ma	Вложенное условие Если	2>1 Да	3>2 Да	5>3 Да	4>5 Нет	-
	то	Если введенная оценка больше той, что считает- ся максимальной					
10	ma := otc	Меняем макси- мальную на нее	2	3	5	-	-
	kolmax := 1	Счет максималь- ных оценок начи- нается снова	1	1	1	-	-
	все	Конец если otc > ma					
	все	Конец если otc = ma					
	кц	Конец цикла для					
11	вывод "Сред- няя оценка = ", sum / 15	Подсчет и вывод средней оценки	3.8				
	вывод "Луч- шая оценка = ", ma	Вывод максимальной оценки	5				
	вывод "Ко- личество лучших оце- нок:", kolmax	Вывод количества максимальных оценок	2				
12	кон	Конец программы					

Таблица 8.20. Примеры программ на Паскале и Visual Basic (к заданию 8.10)

№	Программа на Паскале	Программа на Visual Basic
1	program Ocenki;	Private Sub Command1_Click()
	const n = 15;	Const n = 15
	var i, ma, sum, kolmax, otc: integer;	Dim i As Integer, ma As Integer, sum As Integer, kolmax As Integer, otc As Integer
	begin	
2	ma := 1; sum := 0; kolmax := 0;	ma = 1; sum = 0; kolmax = 0
3	for i := 1 to n do	For i = 1 To n
	begin	
4	writeln('Введите оценку ', i, '- го ученика');	otc = Val(InputBox("Введите оценку " & i & "-го ученика", "Оценки"))
5	readln(otc);	
6	sum := sum + otc;	sum = sum + otc
7	if otc = ma then	If otc = ma Then
8	kolmax := kolmax + 1	kolmax = kolmax + 1
9	else if otc > ma then	ElseIf otc > ma Then
	begin	
10	ma := otc;	ma = otc
	kolmax := 1	kolmax = 1
	end;	End If
		Next
11	writeln('Средняя оценка: ', sum / n:4:1);	Print Print " Средняя оценка: "; FormatNumber(sum / n, 1)
	writeln('Лучшая оценка: ', ma);	Print " Лучшая оценка: "; ma

№	Программа на Паскале	Программа на Visual Basic
	writeln('Количество лучших оценок', kolmax)	Print " Количество лучших оценок: "; kolmax
	end	
12	end.	End Sub

Задание 8.11

Кто из вас хотя бы раз в жизни не считал, сколько дней осталось до любимого праздника? Требуется разработать алгоритм, в котором производится подсчет числа дней до Нового года от текущей даты.

Словесный алгоритм

Начало алгоритма

1. Введите текущую дату: день, месяц и год.
2. Для каждого месяца, начиная с текущего, проанализируйте, сколько в нем дней. Добавьте полученное число к общей сумме дней.
3. Вычтите из полученной суммы число дней, которые уже прошли в текущем месяце.
4. Сообщите количество дней до Нового года.

Конец алгоритма



Иногда применение вложенных блоков принятия решения делает схему громоздкой. Здесь в блок-схему введен блок выбора по индексам месяцев. Этот блок применяется, когда вариантов выбора по условию больше двух. Он соответствует оператору Выбор (Case).

Алгоритм в виде блок-схемы

На рис. 8.14 приведена блок-схема, составленная по словесному алгоритму.

Алгоритм в виде программы

В табл. 8.21 приведена программа на алгоритмическом языке Кумир. В табл. 8.22 приведены тексты программы на языках программирования Паскаль и Visual Basic.

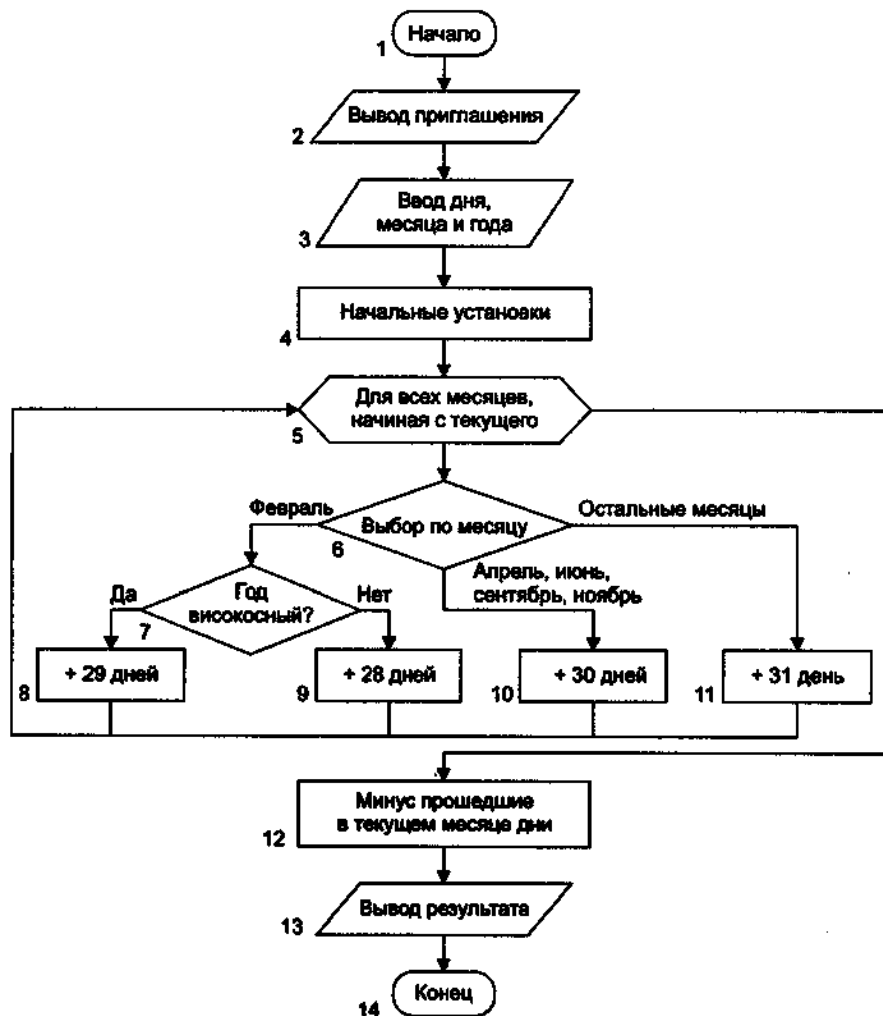


Рис. 8.14. Блок-схема алгоритма (к заданию 8.11)

Таблица 8.21. Программа на Кумире с пояснениями (к заданию 8.11)

№	Программа на Кумире	Пояснения
1	алг До_Нового_Года	Начало алгоритма
	нач цел d, m, god, i, kolday	Описание переменных

№	Программа на Кумире	Пояснения
2	вывод "Введите день, месяц, год"	Вывод приглашения для ввода
3	ввод d, m, god	Ввод дня, месяца и года
4	koday := 0	Начальные установки: обнуление счетчика дней
5	нц для i от m до 12	Начало цикла: для всех месяцев, начиная с текущего
	выбор	Выбор по номеру месяца i
6	при i = 2:	Для февраля:
7	если mod(god, 4) = 0	Если год високосный (в XXI веке — делится на 4 без остатка)
	то	То
8	koday := koday + 29	К счетчику дней добавляется 29
	иначе	Иначе
9	koday := koday + 28	К счетчику дней добавляется 28
	все	Конец если
10	при (i = 4) или (i = 6) или (i = 9) или (i = 11): koday := koday + 30	Для апреля, июня, сентября и ноября к счетчику дней добавляется 30
	иначе	Для всех остальных месяцев:
11	koday := koday + 31	К счетчику дней добавляется 31
	все	Конец выбора
	кц	Конец цикла
12	koday := koday - d	Из счетчика вычитается текущий день (дни, уже прошедшие в текущем месяце)
13	вывод "Количество дней до Нового Года: ", koday	Вывод результата
14	кон	Конец алгоритма

Таблица 8.22. Примеры программ на Паскале и Visual Basic (к заданию 8.11)

№	Программа на Паскале	Программа на Visual Basic
1	program New_Year;	Private Sub New_Year_Click()
	var d: 1..31; m, i: 1..12; god, kolday: word;	Dim d As Integer, m As Integer, god As Integer, i As Integer, kolday As Integer
	begin	
2	writeln('Введите день, месяц, год');	d = Val(InputBox("Введите день", "Дата")) m = Val(InputBox("Введите месяц", "Дата")) god = Val(InputBox("Введите год", "Дата"))
3	readln(d, m, god);	
4	kolday := 0;	kolday = 0
5	for i := m to 12 do	For i = m To 12
6	case i of	Select Case i
7		Case 2
8	2: if god mod 4 = 0	If god Mod 4 = 0
9	then kolday := kolday + 29	Then kolday = kolday + 29
	else kolday := kolday + 28;	Else kolday = kolday + 28
10	4, 6, 9, 11: kolday := kolday + 30;	Case 4, 6, 9, 11
		kolday = kolday + 30
11	else	Case Else
	kolday := kolday + 31;	kolday = kolday + 31
		End Select
	end;	Next
12	kolday := kolday - d;	kolday = kolday - d
13	writeln('Количество дней до Нового года - ', kolday)	Print "Количество дней до Нового года: "; kolday
14	end.	End Sub

Задание 8.12

Требуется смоделировать фрагмент теста, проверяющего навыки устного счета в начальной школе. Этот фрагмент (своеобразный конструктор теста) должен позволить учителю, задав любую последовательность цифр со знаками сложения и вычитания (строку), тут же получить и сохранить ответ для последующего сравнения с ответом учащегося. Для подсчета правильного ответа в алгоритм должен быть заложен механизм перевода текстовой строки в последовательность цифр и арифметических знаков.



Проверку на корректность ввода данных в алгоритме можно опустить, так как он предназначен преподавателю, готовящему тест.

Словесный алгоритм

Начало алгоритма

1. Запросите и введите строку, состоящую из цифр и знаков $\leftarrow + \rightarrow$ и $\leftarrow - \rightarrow$.
2. Определите длину строки.
3. Занесите числовой эквивалент первого символа (цифры) в сумму.
4. Для каждого символа строки сделайте следующее:
 - а) прочтите символ;
 - б) рассмотрите три ситуации:
 - если это символ $\leftarrow + \rightarrow$, установите признак знака равным 1;
 - если это символ $\leftarrow - \rightarrow$, то установите признак знака равным -1;
 - иначе (остается только символ-цифра) определите вес цифры и приплюсуйте ее к сумме с соответствующим знаком.
5. Выведите результат.

Конец алгоритма

Алгоритм в виде блок-схемы

На рис. 8.15 приведена блок-схема, составленная по словесному алгоритму.

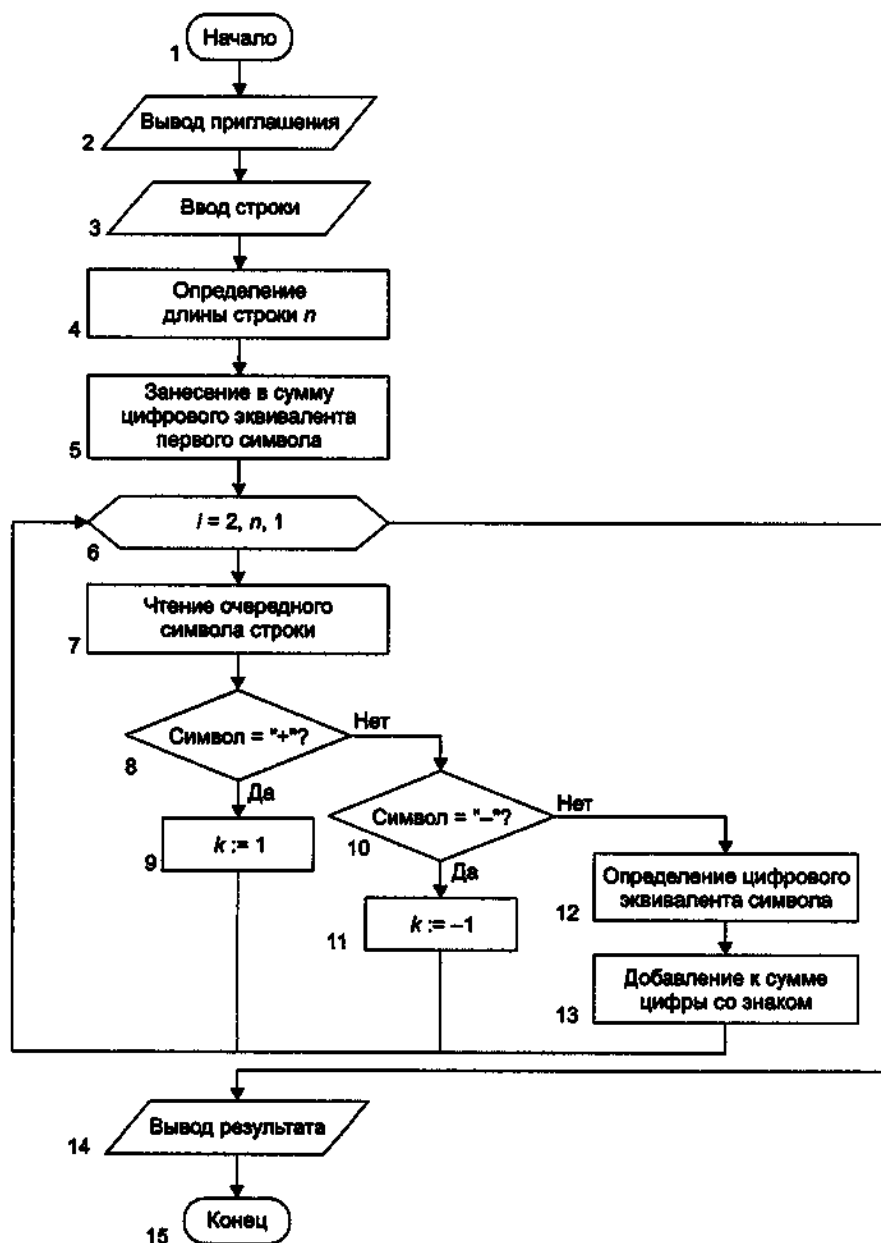


Рис. 8.15. Блок-схема алгоритма (к заданию 8.12)

Алгоритм в виде программы

В табл. 8.23 приведена программа на алгоритмическом языке Кумир. В табл. 8.24 приведены тексты программы на языках программирования Паскаль и Visual Basic.

Таблица 8.23. Программа на Кумире с пояснениями (к заданию 8.12)

№	Программа на Кумире	Пояснения
1	алг Устный счет	Начало алгоритма
	нач цел n, i, k, ch, sum, сим sim, лит st	Описание переменных
2	вывод "Введите выражение "	Приглашение к вводу строки
3	ввод st	Ввод строки
4	n := длин(st)	Определение длины строки
5	sum := цел(st[1]) - цел("0")	Получение цифрового эквивалента первого символа – цифры (от кода символа отнимается код 0) и занесение его в сумму
6	нц для i от 2 до n	Начало цикла: для всех символов, начиная со второго
7	sim := st[i]	Чтение очередного символа строки
	выбор	Выбор из трех возможных вариантов:
8	при sim = "+": k := 1	Если символ «+»
9		k := 1
10	при sim = "-": k := -1	Если символ «-»
11		k := -1
12	иначе	Иначе (символ – цифра):
	ch := цел(sim) - цел("0")	Нахождение «веса» цифры
13	sum := sum + k * ch	Добавление цифры в сумму с соответствующим знаком
	все	Конец иначе

№	Программа на Кумире	Пояснения
	кц	Конец цикла
14	вывод "Значение арифметического выражения: ", st; "=", sum	Вывод результата выражения
15	кон	Конец алгоритма

Таблица 8.24. Примеры программ на Паскале и Visual Basic (к заданию 8.12)

№	Программа на Паскале	Программа на Visual Basic
1	program Ustno;	Private Sub Command1_Click()
	var n, i, k, ch, sum: integer; st: string; sim: char;	Dim n As Integer, i As Integer, k As Integer, ch As Integer, sum As Integer, st As String, sim As String
	begin	
2	write('Введите выражение: ');	st = InputBox("Введите выражение", "Значение выражения", "9 - 6 + 4 + 5 - 1")
3	readln(st);	
4	n := length(st);	n = Len(st)
5	sum := ord(st[1]) - ord('0');	sum = Asc(Left(st, 1)) - Asc("0")
6	for i := 2 to n do	For i = 2 To n
	begin	
7	sim := st[i];	sim = Mid(st, i, 1)
	case sim of	Select Case sim
8	'+' : k := 1;	Case "+"
9		k = 1
10	'-' : k := -1;	Case "-"
11		k = -1
	else	Case Else

№	Программа на Паскале	Программа на Visual Basic
	<code>begin</code>	
12	<code>ch := ord(sim) - ord('0');</code>	<code>ch = Asc(sim) - Asc("0")</code>
13	<code>sum := sum + k * ch</code>	<code>sum = sum + k * ch</code>
	<code>end</code>	
	<code>end; {case}</code>	End Select
	<code>end; {for}</code>	Next
14	<code>writeln('Значение арифметического выражения: ', st, ' = ', sum)</code>	<code>Print "Значение арифметического выражения: "; st; "="; sum</code>
15	<code>end.</code>	End Sub

Контрольные вопросы и задания

К заданию 8.10

1. Определите по блок-схеме (см. рис. 8.13), сколько учеников участвовало в контроле знаний?
2. Что означает запись `i = 1, 15, 1` в блоке, обозначающем цикл?
3. Что означает левая и правая части оператора присвоения `sum = sum + otc`? Что и чему будет присваиваться?
4. Почему в начальных установках переменной `max` присваивается значение 1, несмотря на то что оценка 1 не ставится за контрольные работы и экзамены?
5. Изобразите в тетради фрагмент блок-схемы цикла Для, который будет суммировать числа от 1 до 100.
6. Изобразите в тетради блок-схему алгоритма, вычисляющего произведение всех нечетных чисел от 1 до 29.
7. Останутся ли оценки, которые были введены в алгоритме, после его завершения?

К заданию 8.11

1. Для чего нужны начальные установки в алгоритме?
2. Где в программе на Кумире учитывается, что определенное количество дней в текущем месяце уже прошло и не должно учитываться в сумме?

3. Когда в программах используется оператор выбора? Можно ли заменить его операторами Если?
4. Выпишите в тетрадь, как выглядит оператор выбора на разных языках.
5. Значение какой переменной в программах влияет на выбор той или иной ветки? Что реально означает эта переменная?
6. Найдите в программе на Кумире место, где определяется, является ли год в XXI веке високосным.

К заданию 8.12

1. Определите по блок-схеме (см. рис. 8.15), почему цикл Для начинается со второго символа, а не с первого?
2. На блок-схеме условные блоки 8 и 10 обеспечивают выбор из трех возможных вариантов: символ «+», символ «-» и символ-цифра. Найдите в примерах программ фрагменты, соответствующие этому множественному выбору.
3. При помощи каких функций определяется длина введенной строки в разных языках? Выпишите их в тетрадь.
4. Определите по примерам программ, какой тип у переменной st?
5. В третьем блоке блок-схемы (см. рис. 8.15) была введена следующая строка: 6+3-2-4+5. Определите по программе на Кумире:
 - длину строки n;
 - какие значения примут переменные sim и k после первого прохода цикла;
 - какие значения будут находиться в переменных sim, k, sum и ch после завершения цикла.

Тема 8.5 Цикл с постусловием

Выполнив задания этой темы, вы научитесь:

- ⇒ использовать цикл с постусловием для решения задач;
- ⇒ выбирать тот или иной вид цикла в зависимости от поставленной задачи;
- ⇒ использовать типовые алгоритмы поиска минимума и максимума;
- ⇒ использовать данные разного типа при решении задач.

Цикл с постусловием относится к циклам с неизвестным числом повторений.

В цикле с постусловием условие окончания цикла проверяется в конце оператора цикла. Если условие ложно (нет, False), то тело цикла повторяется, а иначе (если истинно — да, True) цикл завершается. Особенностью этого цикла является то, что тело цикла выполняется хотя бы один раз. Специального блока для реализации цикла с постусловием в блок-схемах нет, но во всех языках программирования есть специальные операторы, реализующие этот цикл.

Задание 8.13

В некоторых случаях окружность заменяют правильным n -угольником. Например, если надо получить круглую игровую площадку, то трудно найти циркуль для ее очерчивания. Тогда достаточно вбить в землю колышек с веревкой, равной предполагаемому радиусу, и сделать отметки, вращая натянутую веревку. Остается только соединить полученные отметки.

При увеличении числа сторон такого n -угольника его периметр стремится к длине окружности. Требуется рассчитать параметры n -угольника, чтобы его периметр был равен длине окружности с заданной абсолютной погрешностью.

Словесный алгоритм

Начало алгоритма

1. Задайте радиус окружности.
2. Вычислите для него длину окружности.

3. Последовательно удваивайте число сторон n -угольника, рассчитывайте его периметр и сравнивайте с длиной окружности, пока не добьетесь требуемой точности.

4. Сообщите результаты.

Конец алгоритма

Алгоритм в виде блок-схемы

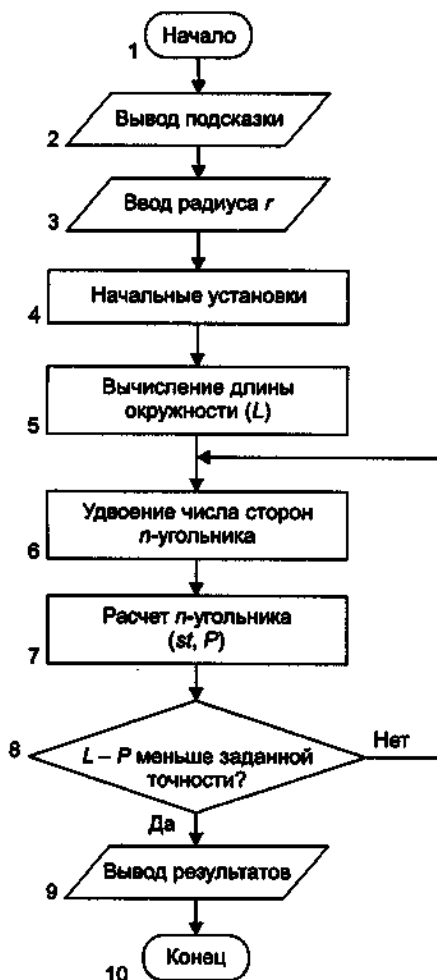


Рис. 8.16. Блок-схема алгоритма (к заданию 8.13)

Алгоритм в виде программы

В табл. 8.25 приведена программа на алгоритмическом языке Кумир. В табл. 8.26 приведены тексты программы на языках программирования Паскаль и Visual Basic.

Таблица 8.25. Программа на Кумире с пояснениями (к заданию 8.13)

№	Программа на Кумире	Пояснения
1	алг многоугольник	Начало программы
	нач вещ e , π , r , l , p , st цел n	Описание переменных
	$\pi := 3.14159$	
	$e := 0.1$	Заданная точность
2	вывод "Введите радиус"	Приглашение на ввод
3	ввод r	?
4	$n := 3$	Начальные установки
5	$l := 2 * \pi * r$	?
	нц	Начало цикла
6	$n := n * 2$	Удвоение числа сторон n -угольника
7	$st := 2 * r * \sin(\pi / n)$	Расчет n -угольника: стороны и периметра
	$p := st * n$	
8	кц_при $(l - p) < e$	Выход из цикла при достижении заданной точности
9	вывод "L=", l , " p=", p , " n=", n	Вывод результатов
10	кон	Конец алгоритма

Таблица 8.26. Примеры программ на Паскале и Visual Basic (к заданию 8.13)

№	Программа на Паскале	Программа на Visual Basic
1	program Mnougoug;	Private Sub Mnougougolnik_Click()
	const $e = 0.1$; var n : byte; r , st , l , p : real;	Const $e = 0.1$ Const $\pi = 3.1415$ Dim n As Integer, r As Single, st As Single, l As Single, p As Single
	begin	

№	Программа на Паскале	Программа на Visual Basic
2	write('r=');	r = Val(InputBox("Введите радиус"))
3	readln(r);	
4	n := 3;	n = 3
5	l := 2 * pi * r;	l = 2 * pi * r
	repeat	Do
6	n := n * 2;	n = n * 2
7	st := 2 * r * sin(pi / n);	st = 2 * r * Sin(pi / n)
	p := st * n;	p = st * n
8	until (1 - p) < e;	Loop Until (1 - p) < e
9	writeln('L=', l:6:2, 'P=', p:6:2, 'N=', n)	Print "L="; FormatNumber(l, 2); " P="; FormatNumber(p, 2); " N="; n
10	end.	End Sub

Задание 8.14

Бывают ситуации, когда надо найти информацию по образцу в неупорядоченном (по алфавиту, номеру и пр.) массиве. Например, ученик может искать книги для реферата в домашней библиотеке, квартиросъемщик — свою фамилию в журнале вызова электрика и т. п.

Требуется создать алгоритм поиска в массиве заданной фамилии и обеспечить запоминание ее порядкового номера. По найденному номеру можно впоследствии найти соответствующие этому человеку данные (адрес, дату рождения и др.) в других массивах, связанных с данным.

Словесный алгоритм

Начало алгоритма

1. Введите фамилию.
2. Сравнивайте ее с очередным элементом массива, пока не найдете такую же или пока не закончится список.
3. Если фамилия найдена, сообщите ее номер в списке, если нет — сообщите о том, что фамилия не найдена.

Конец алгоритма

Алгоритм в виде блок-схемы

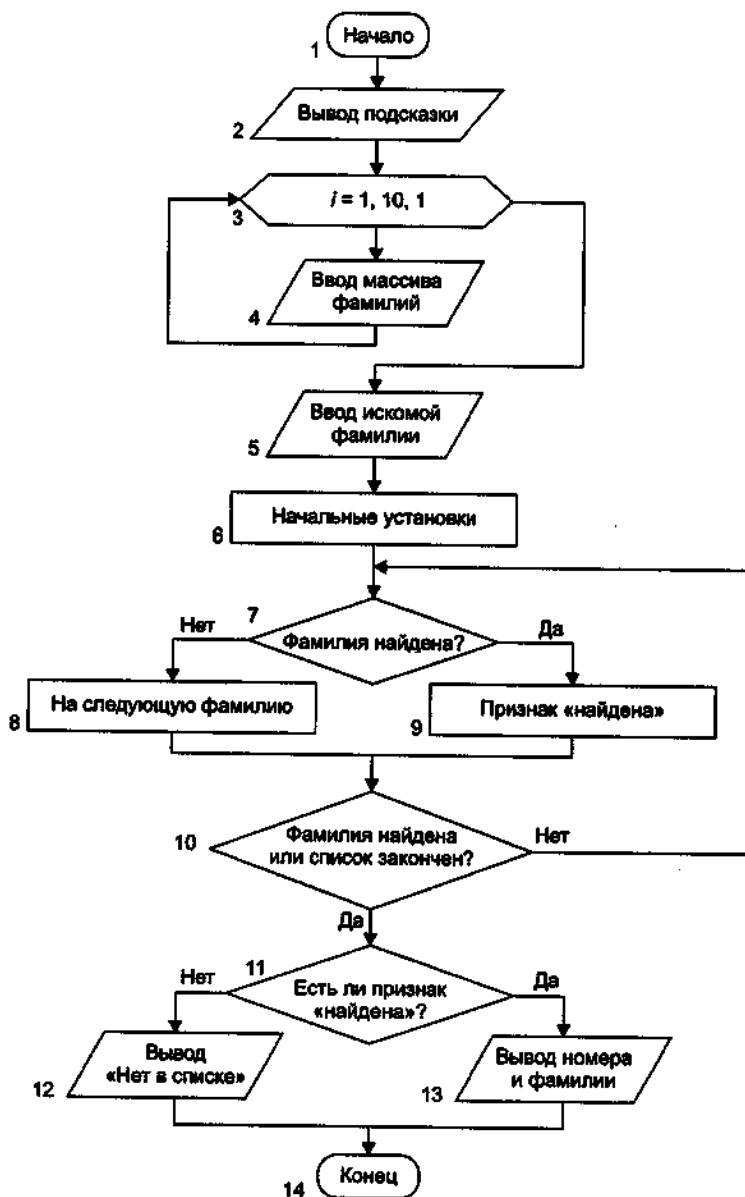


Рис. 8.17. Блок-схема алгоритма (к заданию 8.14)



В алгоритме имеется два цикла. Первый цикл — цикл заполнения массива — является вспомогательным, поэтому не был описан в словесном алгоритме. Он добавлен для того, чтобы проверить работу основного алгоритма.

Алгоритм в виде программы

В табл. 8.27 приведена программа на алгоритмическом языке Кумир. В табл. 8.28 приведены тексты программы на языках программирования Паскаль и Visual Basic.

Таблица 8.27. Программа на Кумире с пояснениями (к заданию 8.14)

№	Программа на Кумире	Пояснения
1	алг Поиск_фамилии	Начало программы
	нач литтаб f[1:10], лит fam, цел i, лог poisk	Описание переменных
2	вывод "Введите фамилии, на- жимая Enter после каждой фа- милии"	Приглашение на ввод
3	нц для i от 1 до 10	?
4	ввод f[i]	Ввод очередного элемента массива фамилий
	кц	Конец цикла Для
5	вывод "Введите искомую фа- милию"	Ввод искомой фамилии
	ввод fam	
6	poisk := нет	Начальные установки
	i := 1	
	нц	?
7	если f[i] = fam	Если очередной элемент массива равен искомому
	то	То
8	poisk := да	Установка признака «Найдена»

№	Программа на Кумире	Пояснения
	иначе	Иначе
9	$i = i + 1$	Переход к следующей фамилии
	все	Конец Если
10	кц_при poisk или $(i > 10)$	Конец цикла при признаке «Найдена» или по концу списка
11	если poisk	Если фамилия найдена
	то	То
12	вывод "Фамилия находится на "; i; "месте"	Вывод индекса фамилии в списке
	иначе	Иначе
13	вывод "Нет такой фамилии"	Вывод сообщения «Нет такой фамилии»
	все	Конец Если
14	кон	Конец программы

Таблица 8.28. Примеры программ на Паскале и Visual Basic (к заданию 8.14)

№	Программа на Паскале	Программа на Visual Basic
1	program Mass;	Private Sub Poisk_v_mass_Click()
	var f: array[1..10] of string; i: byte; fam: string; poisk: Boolean;	Dim f(1 To 10) As String Dim i As Byte, fam As String, poisk As Boolean
	begin	
2	writeln('Введите фамилии');	
3	for i := 1 to 10 do	For i = 1 To 10
4	readln(f[i]);	f(i) = Inputbox("Введите фамилию ")
		Next

№	Программа на Паскале	Программа на Visual Basic
5	writeln('Введите искомую фамилию');	fam = InputBox("Введите искомую фамилию", "Поиск в массиве")
	readln(fam);	
6	poisk := False;	poisk = False
	i := 1;	i = 1
	repeat	Do
7	if f[i] = fam then	If f(i) = fam Then
8	poisk := True	poisk = True
	else	Else
9	i := i + 1;	i = i + 1
		End If
10	until poisk Or (i > 10);	Loop Until poisk Or i > 10
11	if poisk then	If poisk Then
12	writeln('Фамилия находится на ', i, ' месте')	Print " Фамилия находится на "; i; "месте"
	else	Else
13	writeln('Нет такой фамилии');	Print " Нет такой фамилии"
		End If
14	end.	End Sub

Задание 8.15*

Слышали ли вы о «золотом» прямоугольнике? Если от него отсечь квадрат, то остается прямоугольник с такими же пропорциями (отношением сторон), то есть полученный прямоугольник тоже будет «золотым». Этот процесс можно продолжать до бесконечности. На этой же пропорции базируются все «золотые» геометрические фигуры. Отрезки «золотой» пропорции выражаются бесконечными дробями 1,618 или 0,618.

Если вписать в квадраты, отсекаемые от прямоугольника, четверти окружности, то получается знаменитый «золотой» завиток, являющийся основой многих орнаментов, архитектурных деталей и пр.

Требуется определить по введенным сторонам, является ли прямоугольник «золотым». Принять допущение: считать прямоугольник «золотым», если одно и то же соотношение с заданной точностью (абсолютная погрешность — 0,01) повторилось 5 раз. Вывести «золотое» соотношение.

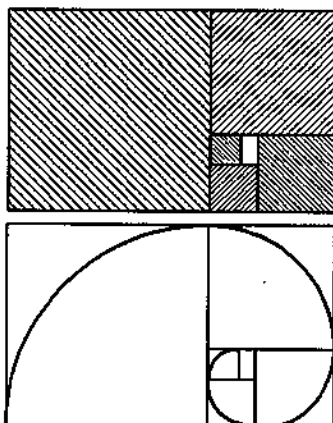


Рис. 8.18. К заданию 8.15

Словесный алгоритм

Начало алгоритма

1. Введите параметры прямоугольника.
2. Определите, какая сторона является большей, какая — меньшей.
3. Найдите и сообщите первое отношение большей стороны к меньшей.
4. Пока не закончатся 5 экспериментов или не выявится неравенство последующих отношений, выполняйте следующие действия:
 - а) определите стороны нового прямоугольника после отсечения квадрата и найдите их соотношение;
 - б) если это соотношение не равно предыдущему с заданной точностью, то завершите эксперименты.
5. Если все 5 попыток прошли успешно, то выведите сообщение «прямоугольник золотой», иначе — «прямоугольник не золотой».

Конец алгоритма

Алгоритм в виде блок-схемы

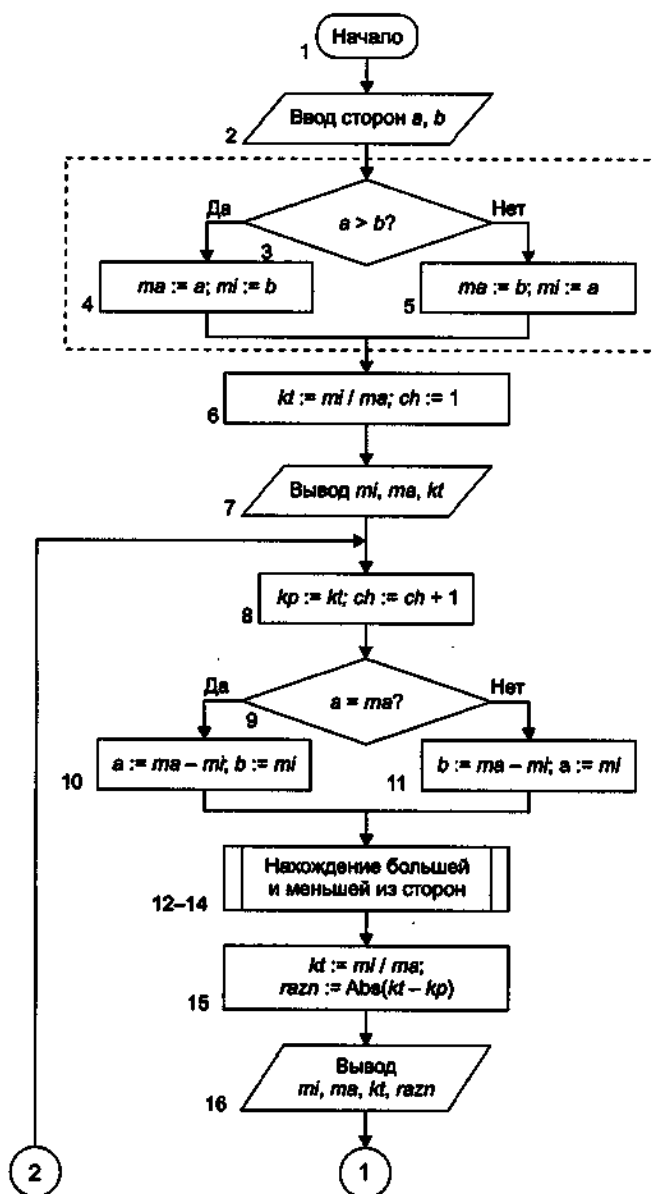


Рис. 8.19. Блок-схема алгоритма (к заданию 8.15)

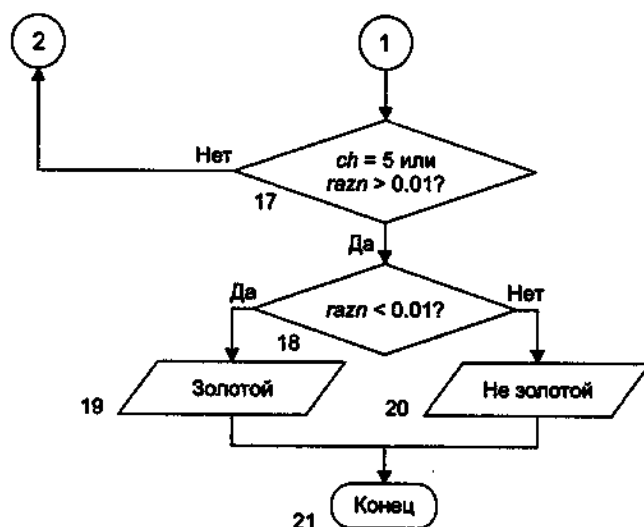


Рис. 8.19. Блок-схема алгоритма к заданию 8.15 (продолжение)



На блок-схеме (рис. 8.19) пунктирной линией обведены блоки, которые будут многократно повторяться далее в цикле. Это означает, что данные блоки должны быть оформлены как вспомогательный алгоритм.

Алгоритм в виде программы

В табл. 8.29 приведена программа на алгоритмическом языке Кумир с пояснениями. В табл. 8.30 приведены тексты программы на языках программирования Паскаль и Visual Basic.

Таблица 8.29. Программа на Кумире с пояснениями (к заданию 8.15)

№	Программа на Кумире	Пояснения
1	алг Золотая пропорция	Начало программы
	нач вещ a, b, mi, ma, ch, kr, kt, razn	Описание переменных
2	вывод "Введите стороны прямоугольника"	Ввод сторон
	ввод a, b	
3	если a < b	Если a < b

№	Программа на Кумире	Пояснения
	то	То
4	$mi := a; ma := b$	Определение соотношения сторон: a — меньшая, b — большая
	иначе	В противном случае
5	$mi := b;$ $ma := a$	Определение сторон: a — большая, b — меньшая
	все	Конец Если
6	$kt := mi / ma;$ $ch := 1;$	Вычисление отношения меньшей стороны к большей Начальная установка счетчика попыток
7	вывод "стороны:", mi , " ", ma , "пропорция =", kt	Вывод значений сторон и первой пропорции
	нц	Начало цикла с постусловием
8	$kp := kt;$ $ch := ch + 1$	Пропорция уже рассмотренного прямоугольника становится предыдущей для нового прямоугольника Увеличение счетчика попыток
	"Отсечение квадрата" от прямоугольника:	Комментарий
9	если $a = ma$	Если a — большая
	то	То
10	$a := ma - mi;$ $b := mi$	?
	иначе	Иначе
11	$b := ma - mi;$ $a := mi$	?
	все	Конец Если
12	если $a < b$	Вычисление длин большей и меньшей сторон в новом прямоугольнике
	то	

№	Программа на Кумире	Пояснения
13	ma := b; mi := a	
	иначе	
14	ma := a; mi := b	
	все	
15	kt := mi / ma; razn := abs(kt - kp)	Расчет новой пропорции и модуля разности пропорций для нового и старого прямоугольников
16	вывод "стороны:", mi, " ", ma, "пропорция =", kt, "Разница K =", razn	Вывод расчетов
17	кц при (razn > 0.01) или (ch = 5)	Выход из цикла, если выполнены все 5 попыток или была неудачная попытка
18	если razn < 0.01	Если вышли при разнице < 0,01
	то	То
19	вывод "Золотой"	Сообщается, что прямоугольник золотой
	иначе	В противном случае
20	вывод "Не золотой"	Сообщается, что прямоугольник не золотой
	все	Конец Если
21	кон	Конец программы

Таблица 8.30. Примеры программ на Паскале и Visual Basic (к заданию 8.15)

№	Программа на Паскале	Программа на Visual Basic
1	program Zolotoe;	Private Sub Zolotoe_click()
	var a, b, ma, mi, kp, kt, razn: real; ch: byte;	Dim a As Single, b As Single, ma As Single, mi As Single, kp As Single, kt As Single, ch As Byte
	begin	
2	writeln('Введите стороны ');	a = Val(InputBox(" Введите сторону"))

№	Программа на Паскале	Программа на Visual Basic
	readln(a, b);	b = Val(InputBox("Введите сторону "))
3	if a < b then	If a < b Then
4	begin ma := b; mi := a end	ma = b; mi = a
5	else	Else
	begin ma := a; mi := b end;	ma = a; mi = b
		End If
6	kt := mi / ma; ch := 1;	kt = mi / ma; ch = 1
7	writeln(mi:6:1, ' ', ma:6:1, 'k=', kt:6:4, ' - первый прямоугольник');	Print mi, ma, "k="; FormatNumber(kt, 3); " - первый прямоугольник"
	repeat	Do
8	ch := ch + 1; kp := kt;	ch = ch + 1; kp = kt
9	if a = ma then	If a = ma Then
10	begin a := ma - mi; b := mi end	a = ma - mi; b = mi
	else	Else
11	begin b := ma - mi; a := mi end;	b = ma - mi; a = mi
		End If
12	if a < b then	If a < b Then
13	begin ma := b; mi := a end	ma = b; mi = a
	else	Else
14	begin ma := a; mi := b end;	ma = a; mi = b
		End If
15	kt := mi / ma; razn := abs(kt - kp);	kt = mi / ma; razn = Abs(kt - kp)
16	writeln(mi:6:1, ' ', ma:6:1, 'k=', kt: 6:3, ' Разница k =', razn: 6:4)	Print mi, ma, "k="; FormatNumber(kt, 4); "; Разница k ="; FormatNumber(razn, 4)
17	until (razn > 0.01) or (ch = 5);	Loop Until (razn > 0.01) Or (ch = 5)
18	if razn < 0.01 then	If razn < 0.01 Then
19	writeln('Золотой')	Print "Золотой"

№	Программа на Паскале	Программа на Visual Basic
	else	Else
20	writeln('Не золотой');	Print "Не золотой"
		End if
21	end.	End Sub

Контрольные вопросы и задания

К заданию 8.13

1. В примерах программ сторона многоугольника рассчитывается через синус центрального угла. В каких единицах измерения должен быть указан угол?
2. Условием окончания цикла является отношение $(l - p) < e$. Что оно означает?
3. Найдите на блок-схеме (см. рис. 8.16) и в примерах программ операторы, составляющие тело цикла с постусловием, и выпишите их в тетрадь.
4. Существуют ли такие ситуации, когда тело цикла в цикле с постусловием не будет выполнено ни разу?
5. Напишите пояснение к пп. 3 и 5 программы на Кумире.
- 6*. Зачем при тестировании программы на Visual Basic (рис. 8.20) расчетные данные выводятся с такой точностью, если точность $e = 0,1$?

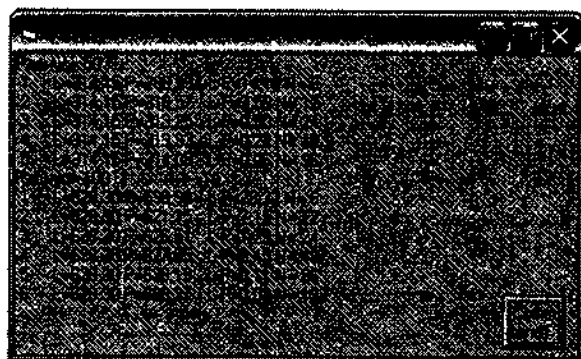


Рис. 8.20. Пример тестирования программы на Visual Basic (к заданию 8.13)

К заданию 8.14

1. В пп. 3 и 6 примеров программ (см. табл. 8.27 и 8.28) начинаются два разных цикла. Напишите в тетради их назначение и разновидности.
2. Сколько элементов в строке фамилий? Где можно узнать это число?
3. Определите по примерам программ тип переменной `poisk`. Какие значения может принимать эта переменная?
4. Условие окончания цикла выглядит так: `poisk` или $(i > 10)$. Что означает это сложное условие?
- 5*. Заполните таблицу истинности для указанного сложного условия.

<code>poisk</code>	$i > 10$	Выход из цикла
False (0)	False (0)	?
False (0)	True (1)	?
True (1)	False (0)	?
True (1)	True (1)	?

6. Выпишите в тетрадь, как описана на разных языках строка фамилий?
7. Если искомая фамилия является последней в списке, по какому условию осуществился выход из цикла?

К заданию 8.15

1. Для чего нужны начальные установки в алгоритме?
2. При решении задачи принято допущение: считать прямоугольник «золотым», если одно и то же соотношение с заданной точностью (абсолютная погрешность — 0,01) повторилось 5 раз. Найдите места в примерах программ, где учитывается это соображение.
3. Что надо изменить в блок-схеме и программах, чтобы пользователь сам указывал абсолютную погрешность?
4. После выхода из цикла анализируется переменная `gazn`. Можно ли анализировать переменную `ch`?

5*. Заполните таблицу истинности для сложного логического выражения $(\text{razn} > 0.01) \text{ Or } (\text{ch} = 5)$.

Число попыток $\text{ch} = 5$	$\text{Razn} > 0.01$	Выход из цикла
False (0)	False (0)	?
False (0)	True (1)	?
True (1)	False (0)	?
True (1)	True (1)	?

6. Напишите пояснения к пп. 10 и 11 программы на Кумире (см. табл. 8.29).
- 7*. Золотое соотношение обеспечивают стороны, длина которых соответствует числам Фибоначчи (чем больше числа, тем точнее пропорция). Это видно на первом тесте (рис. 8.21). Что нужно изменить в программе, чтобы рассматриваемое отношение сторон поменялось с 0,618 на 1,618?
- 8*. На рис. 8.21 изображено два теста. Какого теста не хватает для проверки всех ветвей алгоритма?

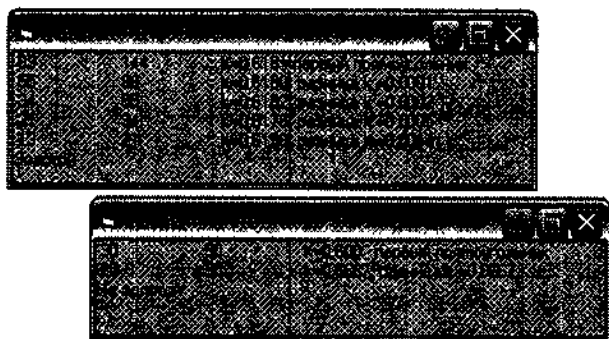


Рис. 8.21. Примеры тестирования программы на Visual Basic (к заданию 8.15)

9. Что нужно изменить (добавить) в блок-схеме, чтобы проверялось условие равенства сторон исходного прямоугольника? Что даст подобное изменение?

Приложение

Справочные сведения по среде программирования ЛОГО

Таблица А.1. Основные команды среды программирования ЛОГО

Название команды (краткая форма)			Назначение инструмента
ЛогоМиры	LogoWriter	LOGO	
Относительное перемещение (команды)			
вперед (вп)	вперед (вп)	forwad (fd)	Черепашка продвигается вперед по направлению головы
назад (нд)	назад (нд)	back (bk)	Черепашка продвигается назад в направлении, противоположном направлению головы
направо (пр)	направо (пр)	rigth (rt)	Черепашка поворачивается направо относительно головы
налево (лв)	налево (лв)	left (lt)	Черепашка поворачивается налево относительно головы
курс_на	–	–	Голова текущей Черепашки поворачивается к Черепашке, имя которой указано
Абсолютное перемещение (команды)			
нов_x	новх	setx	Черепашка перемещается по горизонтали относительно центра координат
нов_y	нову	sety	Черепашка перемещается по вертикали относительно центра координат
домой	домой	home	Черепашка перемещается в центр координат

Название команды (краткая форма)			Назначение инструмента
ЛогоМиры	LogoWriter	LOGO	
нов_курс (нк)	новкурс (нк)	seth	Голова Черепашки поворачивается по часовой стрелке на угол относительно Севера
Перемещение (датчики)			
х_коор	хкоор	xcor	Сообщает значение абсциссы Черепашки в текущем положении
у_коор	укоор	ycor	Сообщает значение ординаты Черепашки в текущем положении
путь	доточки	distance	Сообщает расстояние от Черепашки до указанной точки
—	угол	towards	Сообщает величину угла, образованного направлением головы Черепашки и линией, соединяющей Черепашку с указанной точкой
курс	курс	heading	Сообщает величину угла, образованного линией, идущей на Север, и линией, соединяющей Черепашку с указанной точкой
Состояние Черепашки (команды)			
пп	пп	pu	Перо поднять
по	по	pd	Перо опустить
пс	пс	pe	Перо стереть
сч	сч	ht	Спрятать Черепашку
пч	пч	st	Показать Черепашку
Управление цветом (команды)			
нов_цвет (нц)	новцвет (нц)	setc	Установить Черепашке указанный цвет
нов_фон (нф)	новфон (нф)	setbg	Установить указанный цвет фона
крась	крась	fill	Закрасить фигуру текущим цветом Черепашки

Название команды (краткая форма)			Назначение инструмента
LogoМиры	LogoWriter	LOGO	
Управление цветом (датчики)			
цвет	цвет	color	Сообщает текущий цвет Черепашки
ц_поля (цп)	цп	colorunder	Сообщает цвет пиксела под пером Черепашки
Вывод текста (команды)			
пиши	пиши (п)	print (pr)	Выводит указанное значение в текстовом окне рабочего поля
покажи	покажи	show	Выводит указанное значение в командном центре
Управление алгоритмом (команды)			
пусть	пусть	make	Создание переменной
жди	жди	wait	Пауза на указанное время
повтори	повтори	repeat	Организация цикла
если если_иначе	если еслииначе	if ifelse	Организация алгоритма разветвления
выход	выход	output (op)	Завершение датчика (функции)
стоп	стоп	stop	Остановка выполнения текущей программы
Математические инструменты (датчики)			
+ - * /	+ - * /	+ - * /	Знаки арифметических действий
округли	округли	round	Сообщает число, округленное до целого
целое	целый	int	Сообщает целую часть числа
остаток	остаток	remainder	Сообщает число — остаток от деления
случайный (сл)	случайный (сл)	random	Сообщает целое положительное случайное число
минус	минус	minus	Сообщает число, округленное до целого

Содержание

Введение	3
--------------------	---

ЧАСТЬ 1. ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ В ПРИКЛАДНЫХ ПРОГРАММНЫХ СРЕДАХ

Раздел 1. Освоение системной среды Windows XP . . 5

Тема 1.1. Параметры файла и действия над файлом	6
Тема 1.2. Параметры папки и действия над папкой	11
Тема 1.3. Работа с окнами графического интерфейса	16
Тема 1.4*. Настройка параметров Рабочего стола	21
Тема 1.5. Запуск программ (приложений)	27
Тема 1.6. Работа в Windows как в многозадачной среде	31
Тема 1.7*. Работа с папками и файлами с помощью меню и панели инструментов Проводника	36
Тема 1.8. Технология и способы обмена данными	45
Тема 1.9. Антивирусная защита информации	54
Тема 1.10*. Создание архивных файлов	60

Раздел 2. Освоение среды графического редактора 65

Тема 2.1. Общая характеристика графического редактора	66
Тема 2.2. Создание и редактирование рисунка	74
Тема 2.3. Создание и редактирование рисунка с текстом	80

Раздел 3. Освоение среды текстового процессора 83

Тема 3.1. Общая характеристика текстового процессора	84
Тема 3.2. Создание и редактирование текстового документа	97
Тема 3.3. Форматирование текста	105
Тема 3.4. Оформление текста в виде таблицы и печать документа	113
Тема 3.5*. Включение в текстовый документ графических объектов	117

**Раздел 4. Освоение среды
табличного процессора 125**

Тема 4.1. Общая характеристика табличного процессора	126
Тема 4.2. Создание и редактирование табличного документа	143
Тема 4.3. Форматирование табличного документа	150
Тема 4.4*. Использование функций и логических формул	157
Тема 4.5. Представление данных в виде диаграмм	167

**Раздел 5. Освоение системы управления
базой данных 171**

Тема 5.1. Общая характеристика системы управления базой данных	172
Тема 5.2. Создание структуры и заполнение базы данных	187
Тема 5.3. Создание формы базы данных	193
Тема 5.4. Работа с записями базы данных	197
Тема 5.5. Разработка отчета для вывода данных	203

**Раздел 6. Коммуникации в глобальной сети
Интернет 205**

Тема 6.1. Возможности Интернета	206
Тема 6.2. Среда браузера Internet Explorer	211
Тема 6.3. Поиск информации в сети Интернет	216
Тема 6.4*. Язык разметки гипертекста HTML	223
Тема 6.5*. Веб-страница с графическими объектами	234
Тема 6.6*. Веб-страница с гиперссылками	237
Тема 6.7. Возможности электронной почты	240

**ЧАСТЬ 2. ОСНОВЫ АЛГОРИТМИЗАЦИИ
И ПРОГРАММИРОВАНИЯ****Раздел 7. Программирование в среде ЛОГО 245**

Тема 7.1. Знакомство с основным инструментарием среды	246
Тема 7.2. Понятие программы	255
Тема 7.3. Последовательный алгоритм	263
Тема 7.4. Циклический алгоритм	267

Тема 7.5.	Процедура и модуль	275
Тема 7.6.	Процедура с параметрами	282
Тема 7.7.	Переменная в алгоритме	290
Тема 7.8.	Логика в среде программирования	293
Раздел 8. Основы алгоритмизации		307
Тема 8.1.	Линейные алгоритмы	308
Тема 8.2.	Разветвляющиеся алгоритмы	317
Тема 8.3.	Циклические алгоритмы. Цикл с предусловием	331
Тема 8.4.	Цикл с известным числом повторений	347
Тема 8.5.	Цикл с постусловием	362
Приложение. Справочные сведения по среде программирования ЛОГО		379