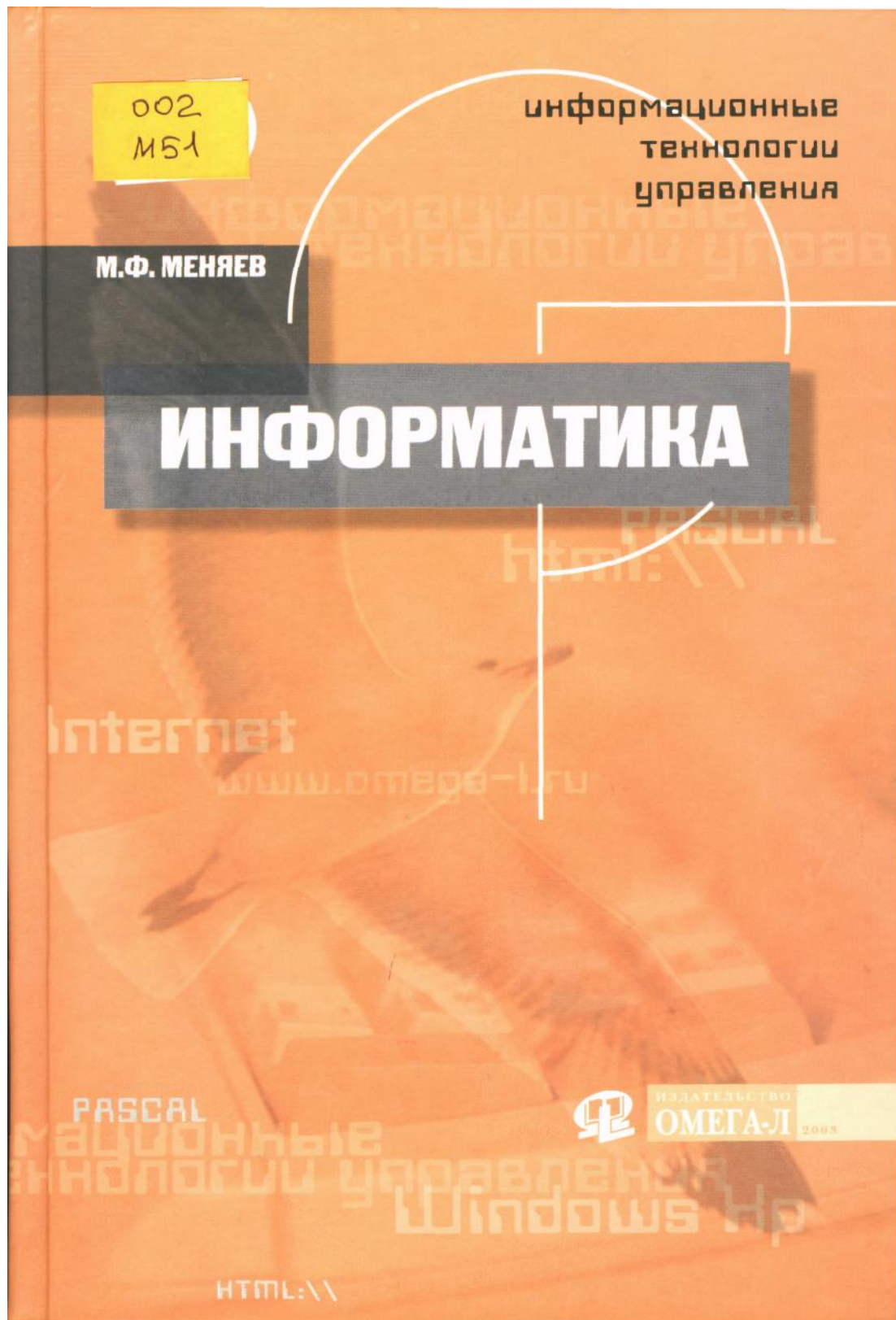


Материалы, размещенные в телекоммуникационной библиотеке и представленные в виде цитат,

допускается использовать исключительно в образовательных целях.

Запрещается тиражирование информационных ресурсов с целью извлечения коммерческой выгоды, а также иное их использование в нарушение соответствующих положений действующего законодательства по защите авторских прав.



М.Ф. Меняев

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ УПРАВЛЕНИЯ

Учебное пособие
в трех книгах

КНИГА 1

ИНФОРМАТИКА

Рекомендовано Советом УМО вузов России по образованию
в области менеджмента в качестве учебного пособия
по специальности «Менеджмент организации»



ИЗДАТЕЛЬСТВО
ОМЕГА-Л

Москва 2003

УДК 881.3
ББК 32.973
М51

Рецензенты:

Майоров С.А. — д.т.н., профессор, МГТУ им. Н.Э. Баумана, кафедра «Менеджмент»;

Павлов А.Н. — д.т.н., профессор, Российская Академия государственной службы при Президенте РФ, кафедра «Информационные структуры государственной службы»;
Кафедра ИБМ-4 и методическая комиссия ф-та ИБМ МГТУ им. Н.Э. Баумана

Меняев М.Ф.

М51 *Информационные технологии управления: Учебное пособие. В 3 кн.: Книга 1: Информатика.* — М.: Омега-Л; Высшая школа, 2003. - 464 с.

ISBN 5-98119-057-4 (Омега-Л)

ISBN 5-06-004632-X (Высшая школа)

В пособии рассматривается материал, раскрывающий содержание методов обработки информации на базе компьютерных технологий и позволяющий получить представление о технологической и программной составляющих информатики. В состав издания входит сборник практических заданий и контрольных упражнений, выполнение которых позволит познакомиться с основами MS DOS, получить навыки использования программы оболочки Norton Commander, а также методов работы в графических операционных средах семейства Windows, овладеть приемами составления программ с использованием языка программирования Турбо Паскаль 7.0 и

перейти к самостоятельной подготовке программ. Для этого в пособии изучаются конкретные тексты программных конструкций и даются задания для самостоятельной работы. Кроме того, отдельные главы посвящены работе в Интернете.

Для студентов университетов и колледжей, обучающихся по специализации «Информационные технологии управления», а также для всех желающих самостоятельно овладеть основами программирования.



УДК 881.3
ББК 32.973

© М.Ф. Меняев, 2003
© Омега-Л, 2003

ISBN 5-98119-057-4

ВВЕДЕНИЕ

Подготовка специалистов для управления организационными структурами предполагает не только обучение активному использованию информационных технологий (ИТ), но также изучение методов и способов их применения для оптимизации и развития системы управления организации.

Информационные технологии находятся в постоянном движении: изменяются их характеристики, совершенствуется структура, возникают новые образцы технических устройств, развиваются новые методы их применения. Все это ведет к объективным тенденциям, создающим предпосылки для оптимизации деятельности и развития организации. Возникает потребность в специалистах, сочетающих в себе знания в области менеджмента организации и навыки применения новых информационных технологий, позволяющие активно влиять на основные параметры и изменять бизнес.

С целью подготовки специалистов в области совершенствования методов управления организационными структурами на основе применения информационных технологий в университетах была введена специализация 061146 «Информационные технологии управления» на базе специализации 061100 «Менеджмент организации», программа обучения которой формируется на основе типовой государственной программы по направлению 521500 «Менеджмент».

При разработке концепции обучения в рамках специализации особое значение было уделено принципу соответствия между знаниями, умениями специалиста, которые требуются профессиональной средой для его будущей деятельности, и уровнем обучения, достигнутым при подготовке специалиста. Исходным моментом в этом соответствии являются требования профессиональной среды к навыкам и умениям специалиста. Практический опыт определяет перечень и содержание тех информационных технологий, которыми должен уметь воспользоваться менеджер для реализации принятой формы управления организацией.

Профессиональной средой менеджера является его практическая деятельность в системе управления организацией. Именно она определяет требования к знаниям менеджера в области ИТ: они должны позволить специалисту использовать ИТ в процессах оптимизации системы управления организации и сформировать соответствующие методы управления на базе новых информационных методов и технологий, а также позволить быстро адаптироваться к дальнейшим изменениям в информационной индустрии и методам управления.



РС — расширения системы базисного ПО;

СР — собственные разработки на основе базисного ПО и офисных систем

Схема взаимосвязи системы управления организации и содержания обучения специалистов в области информационных технологий

На схеме показана графическая иллюстрация принципа соответствия. ИТ организации здесь представлены в форме нескольких прямоугольников, отражающих следующие компоненты ИТ: технические и программные средства ИТ, офисное программное обеспечение (ПО), базисное ПО, настройки организации, а также двух квадратов: расширения системы (РС) и собственные разработки (СР).

Современная система управления организацией, как правило, содержит различные технические устройства: компьютеры, их программное обеспечение, принтеры, сканеры и т. п., которые базируются на определенной системной платформе. Эту часть ИТ организации следует определить как технические и программные средства ИТ.

Офисное ПО содержит операционные системы (платформы), текстовые, графические, табличные и другие среды, различный инструментарий для поддержки ПО. Выбор системной платформы определяет требования к установке базисного ПО, которое позволяет реализовать необходимую концепцию управления предприятием.

Базисное ПО содержит офисные системы, прикладные программы управления, реализующие, например, бухгалтерский, оперативный и другой учет, системы моделирования, справочные базы данных и т. п.

На основе базисного ПО могут выполняться расширения системы (РС), т. е. задание режимов и условий применения базисного ПО в конкретных условиях. На этом уровне могут осуществляться собственные разработки (СР) ПО, повышающие производительность информационной системы и соответственно эффективность системы управления организацией. Базисное ПО во многих случаях дополняется другими системами и программами, которые с помощью бесшовной технологии встраиваются в ИС управления организации.

Требования к знаниям менеджера в области ИТ, а также парк технических и программных средств организации формируют представления о совокупности навыков и умений, которыми должен овладеть специалист в процессе обучения в университете. Эти требования можно сгруппировать в упорядоченные уровни обучения, показанные на левом поле схемы.

Уровень «Основы информатики и программирования» предполагает организацию обучения, нацеленную на овладение обучаемыми методами классификации современной компьютерной технологии и основными приемами программирования с использованием языков высокого уровня. Содержание подготовки на первом уровне может поддерживаться учебным курсом «Информатика».

Второй уровень подготовки формирует умения и навыки в области применения офисных технологий в управленческой деятельности. Здесь также может найти отражение учебный материал, расширяющий возможности офисных приложений с помощью макропрограммирования, а также посвященный работе менеджера в Интернете, включая и формирование личных веб-страниц с использованием языка гипертекстовой разметки HTML. Этот уровень подготовки в государственной программе поддерживается учебным курсом «Информационные ресурсы в менеджменте».

На третьем уровне подготовки основное внимание уделяется базам данных и базам знаний. Здесь важным является получение навыков формирования баз данных для обработки фактографической

информации, использования языка запросов для проведения отдельных операций управления организацией.

Завершает процесс подготовки специалиста четвертый уровень обучения, где студент знакомится с методами построения информационных систем управления, их классификацией, возможностями организации бизнес-сообществ, умением сформировать специализированную службу по внедрению и активному функционированию информационной системы управления организацией. Здесь будущий специалист овладевает методами оптимизации и поиска новых путей развития бизнеса с использованием новых методов и средств информационных технологий.

Содержание учебного пособия ориентировано на образовательный стандарт по специальности 061100 «Менеджмент организации» и может быть использовано в учебных курсах соответствующих разделов стандарта: информатика, информационные ресурсы в менеджменте, информационные технологии управления и др. Пособие также содержит материал, необходимый при изучении учебных дисциплин регионального (вузовского) компонента: информационное обеспечение производства, системы управления базами данных организации, динамические системы управления и др.

Данное издание состоит из трех книг, изучение каждой из которых направлено на формирование соответствующих навыков.

Материал первой книги «Информатика» направлен на изучение методов классификации информационных технологий в управлении организацией. Кроме того, здесь даются основные сведения о языке гипертекстовой разметки (HTML).

Вторая книга «Информационные ресурсы» содержит учебный материал, касающийся обработки информации с помощью различных систем: текстовых, табличных, графических, презентационных, а также дает возможность получить навыки работы с базами данных при обработке фактографической информации. Материал этой книги направлен на формирование современного практического знания, позволяющего активно использовать офисные технологии в профессиональной деятельности менеджера.

В третьей книге «Системы управления организацией» содержатся сведения о различных методах управления организацией, основных представлениях системы качества, рассматриваются методы построения информационного пространства организации и анализируются структура и содержание интегрированных информационных систем управления.

Эта же книга вводит читателя в круг проблем активного использования глобальных информационных сетей для построения бизнес-сообществ с целью повышения эффективности управления организацией. Здесь обсуждаются вопросы подготовки веб-сайта предприятия, организации портала предприятия, системы управления клиентами и другие элементы динамической системы управления.

В каждой книге учебного пособия предусмотрена практическая часть, включающая в себя набор заданий, выполнение которых на компьютере в среде стандартных приложений и информационных систем позволит студенту самостоятельно освоить материал соответствующих учебных курсов. Также эта часть издания может быть использована преподавателями университетов для проведения практических занятий при организации обучения в университетах и для организации дистанционной формы обучения.

Каждая книга издания содержит словарь терминов и список рекомендуемой литературы.

Существуют различные подходы к решению проблемы обучения, однако для подготовки управленческих кадров наиболее распространенным является метод практического обучения. В его основе — фундаментальная научная и активная практическая подготовка. Используя эту традицию, мы также стремимся дать основные научные представления и в то же время создать условия для формирования навыков и умений с помощью практической формы обучения.

При отборе материала для учебного пособия учитывались и те ограничения, которые накладывает на учебный процесс время обучения. Это объясняет, в частности, то, что материал дается конспективно, рассматриваются лишь наиболее важные аспекты научного и технического знания, важные для формирования общей картины информационных процессов и используемые в дальнейшем обучении. Мы предполагаем, что при прочих равных условиях лучше стремиться к лаконичности. За рамками пособия остались, в основном, те моменты, которые могут быть самостоятельно изучены в процессе практической работы в информационном пространстве.

Содержание практического обучения здесь реализовано на базе метода активной рефлексии (The Active Reflection Together), применение которого предполагает изучение (лучше в процессе дискуссий с

педагогом) и обдуманное повторение обучаемым операций, необходимых для достижения поставленной цели. В процессе обучения выделяется три этапа: изложение теоретического материала; анализ способов применения полученного знания с помощью преподавателя или при самостоятельном изучении пособия (при построении конкретного алгоритма обработки информационного процесса); практическая реализация, т. е. процесс написания программы, ее отладки и апробации в компьютерной среде. Именно в процессе выполнения последнего этапа при достижении поставленной цели и формируются необходимые навыки. Важное значение предлагаемого учебного пособия проявляется в том, что оно поддерживает различные формы организации активного обучения. При этом формирование полезных навыков может осуществляться как под руководством преподавателя, так и самостоятельно, как в аудитории, так и дома, как в течение заданного временного интервала (в аудитории или в компьютерном зале), так и в процессе самонастраиваемого времени обучения. Важно то, что пособие позволит обучаемому «настроить» учебный процесс под свои возможности.

Текст издания апробирован в системе подготовки инженеров-менеджеров по специализации «Информационные технологии в управлении бизнесом» на кафедре Менеджмент факультета Инженерный бизнес и менеджмент МГТУ им. Н.Э. Баумана.

ОТ АВТОРА

Использование информационных технологий как в деле управления, так и в других областях технической и коммерческой деятельности предполагает формирование практических навыков работы с различными техническими устройствами и различными видами программного обеспечения. К первым следует отнести системные блоки, экраны, принтеры, сканеры, модемы и пр., а ко вторым — операционные системы, программы-оболочки операционных систем, графические операционные системы, текстовые и табличные процессоры, графические редакторы, языки программирования и др.

Предлагаемая читателю книга не просто учебное пособие в сложившемся понимании этого слова. Она — обучающая среда (Tutor-media), позволяющая на основе многократного практического действия сформировать навыки использования компьютера для разработки и программирования информационных систем.

Пособие требует постоянного диалога с компьютером для того, чтобы на основе теоретического материала, показанного в первых трех главах, работать в операционной среде и использовать различные вспомогательные программы, вводить и изучать многочисленные программы, тексты которых приведены в третьей главе пособия, контролировать процесс обучения, используя тестовые задания из четвертой главы издания.

Изложение материала ведется достаточно сухо — без подробного комментария. Это объясняется и необходимостью экономии времени читателя, и имеющейся у него возможностью получать более подробные справки в многочисленных учебниках, пособиях, справочниках.

Содержание учебного пособия не повторение учебника, а последовательно и четко сформулированные определения, которые необходимо рассмотреть, обсудить в процессе овладения навыками работы в операционной среде и программирования. Это достигается главным образом путем рассмотрения многочисленных текстов программ, приведенных в качестве примеров.

Для практического обучения работе в операционных средах следует воспользоваться соответствующими подглавами в конце первой и второй главы, которые определены как «Практикум». Для закрепления навыков программирования следует воспользоваться материалом практикума третьей главы пособия. Здесь приведены учебные задания и тексты программ, реализующих предлагаемые задания. Учебные задания сгруппированы по тематическим семинарам.

Учебные задания и предлагаемые тексты программ необходимо предварительно рассмотреть самостоятельно или обсудить на семинарских занятиях, а затем набрать и опробовать тексты программ на практических занятиях (лабораторных работах).

Лабораторный практикум, входящий в третью главу издания, следует рассматривать как составную часть Tutor-media пособия, которое дополняется сборником заданий к лабораторному практикуму и заданиям к тестам (рейтинговым работам), приведенным в четвертой главе издания.

Используя учебное пособие, студент может активизировать непосредственно процесс обучения. Для этого он может опереться на опыт педагога в области информатики, так и выполнить обучающий процесс самостоятельно, ориентируясь на тексты предложенных в пособии программ. Покажем обе методики.

В первом случае задания студентам для практической работы по программированию выдаются преподавателем на семинарских занятиях. Студенту, получившему задание, следует заранее составить алгоритм выполнения программы и написать текст на языке Турбо Паскаль. Необходимо учесть, что даже для тех, кто имеет достаточные знания в этой области, выполнение этой работы требует времени. Нередко, бегло просмотрев тексты заданий, особенно первые из них, у студентов создается впечатление, что они на перемене смогут быстренько «накидать» текст задачи. Но так думать наивно, поскольку, во-первых, задания постепенно усложняются, а во-вторых, навыки не формируются на перемене. Во втором случае, при самостоятельном изучении учебного курса, необходимы регулярная работа с материалом учебного пособия, отсутствие спешки, достижение результата. Именно результат, увиденный на экране компьютера, в десятки раз активизирует деятельность обучаемого, позволяет использовать положительные эмоции для закрепления навыков и обретения уверенности в правильности выбранного пути обучения.

При самостоятельном методе обучения сборник программ является также и средством для самоконтроля процесса обучения. В этом случае при обучении следует обратить внимание на тему конкретного занятия и после тщательного изучения текста программы ввести его в память компьютера и отладить программу. И здесь важно увидеть результат действия программы. Пусть вначале это будет повторение текста учебного пособия. Важно другое: понять смысл изучаемой конструкции, а затем применить ее в процессе выполнения «домашнего» задания. При необходимости следует выполнить несколько вариантов предложенных в сборнике задач (лучше не заглядывая в книгу).

Особое внимание рекомендуется уделить заданиям для самостоятельной работы (тестам), тексты которых приведены в четвертой главе пособия.

Тексты программ, приведенные в пособии, составлены в процессе многолетних учебных занятий со студентами. Некоторые из них возникли в результате влияния различных изданий, приведенных в списке литературы, другие написаны студентами в ходе выполнения лабораторного практикума и изначально отражают опыт большого числа преподавателей и студентов, однако каждое из заданий претерпело изменения, направленные на активизацию процесса обучения.

Материал издания можно рассматривать как учебное пособие по курсу «Информатика», который читается в российских университетах прежде всего при подготовке по специализации «Информационные технологии управления». Изучение пособия послужит хорошей основой для развития навыков алгоритмизации у будущих специалистов, что является обязательным условием формирования культуры управления организациями в современном обществе.

Деятельность менеджера в современном мире предполагает активное использование возможностей Интернета для поиска и передачи информации с целью реализации активного маркетинга в международном информационном сообществе.

Процесс формирования навыков работы в различных сервисах Интернета все время должен продолжаться в активной деятельности, однако на начальном этапе менеджеру необходимо овладеть основами работы в Интернете и методами формирования своей веб-страницы, чему посвящены отдельные главы.

Цель практического занятия «Информационные ресурсы Интернета» состоит: в овладении навыками использования специализированных программ (браузеров) для обращения к различным ресурсам Интернета; изучении основных приемов регистрации, настройки и поиска информации при использовании браузера Netscape Communicator; применении методов передачи информации в Интернете с помощью сервиса «Электронная почта» и др.

Цель занятия «Поиск и обработка информации в Интернете» направлена на овладение следующими навыками: поиск и копирование информации при помощи специализированных поисковых систем (машин), посещение веб-сайтов различных организаций и т. п. при помощи браузера Microsoft Explorer и др.

При выполнении задания по теме «Формирование текста веб-документа» обучаемые овладевают навыками разметки текста веб-документа с использованием языка гипертекстовой разметки HTML: организация заголовка и содержательной части документа, методы логического и физического форматирования строк, заголовков, списков, разделение на абзацы, перевод строки, применение специальных символов, организация гиперссылок, формирование оглавления документа и др.

Развитие навыков применения языка HTML осуществляется в процессе изучения последней темы «Списки и таблицы в веб-документах». Здесь рассматриваются методы установки списков и таблиц на страницах веб-документа с помощью языка HTML.

Учебное задание по пятой теме «Применение графики и фреймов для организации веб-документов» нацелено на овладение навыками: размещения графических элементов, применения графических элементов в качестве указателей ссылок, использования карт-изображений на полях документа, разработки фрейм-документов и др.

Приведенные в пособии примеры были подготовлены и апробированы в различных аудиториях при поддержке многих преподавателей МГТУ им. Н.Э. Баумана, за что выражаю им свою признательность. Также благодарю студентов, которые терпеливо прослушали курс и «проверили» на своем опыте обучения многие программы, вошедшие в это издание.

Особенно хочется поблагодарить к.т.н., доцента Алексея Федоровича Меняева, который сделал ряд ценных замечаний по тексту авторского оригинала, а также предложил алгоритмы и тексты некоторых программ.

Надеюсь на то, что в процессе работы с изданием у читателя возникнет желание его дополнить и сообщить об этом по адресу: [http:// www.omega-l.ru](http://www.omega-l.ru).

ГЛАВА 1. ОСНОВНЫЕ КОМПОНЕНТЫ КОМПЬЮТЕРНОЙ ТЕХНОЛОГИИ

1.1. Компьютерные технологии и информационное пространство

Современное значение компьютерной технологии состоит не только в том, что она обеспечивает быструю и эффективную обработку информации, создает условия для ее хранения и передачи, воздействует практически на все области деятельности мирового сообщества, способствует развитию нового знания, зарождению новых технологий, но и в том, что она позволяет передавать в общемировое информационное пространство достижения каждого отдельного индивидуума и дает возможность практически каждому жителю планеты пользоваться интеллектуальными достижениями всего человечества. Компьютер становится важнейшим элементом в системе коммуникации современного информационного общества.

На рис. 1.1 показана схема организации информационного пространства общества в своем взаимодействии с информационным пространством каждого члена общества. Из рисунка следует, что основными компонентами информационного пространства являются: база знаний каждого человека, его личная библиотека, персональный компьютер, совокупность внешних устройств, соединяющих с помощью компьютера внутренний мир личности и внешний мир, каналы коммуникации, включая Интернет (инфо-сообщество), и мировой интеллектуальный ресурс.



Рисунок 1.1. Схема организации информационного пространства общества

База знаний личности — это способность личности создавать смысловые конструкции, отражающие его восприятие мира.

Личная библиотека — совокупность документов в любой форме представления информации, которая создана конкретной личностью и хранится в его компьютере или на полке его библиотеки.

Внешние устройства — устройства фиксации и/или передачи информации, получаемой от инфо-сообщества по каналам связи. Особое значение здесь имеет глобальная информационная сеть Интернет, позволяющая обеспечить практически мгновенную взаимосвязь абонентов. К внешним устройствам следует отнести факс-модемные устройства связи, стримеры, сканеры, проекционные аппараты и панели, принтеры, устройства чтения и/или записи на оптические диски (CD), дискеты (флоппи-диски) и другие средства приема и/или передачи информации. Внешние устройства соединяются с компьютером с помощью соответствующих адаптеров.

Инфо-сообщество — сообщество личностей, формирующих информационное пространство общества с использованием компьютерной технологии.

Мировой интеллектуальный ресурс представляет собой понятие, охватывающее все произведения культуры и техники человечества, к которым обеспечен доступ с помощью информационных технологий.

1.2. Компьютер: структура и параметры

В современных системах управления понятие «информационная система» (ИС) определяет совокупность технических средств (ТС), программного обеспечения (ПО), методов организации процессов подготовки первичных данных и способов использования полученной информации для организации деятельности коллектива или отдельной личности.

1.2.1. Структурная схема компьютера

В качестве технической основы ИТ используется компьютер. **Компьютер** (электронно-вычислительная машина — ЭВМ) — техническое устройство, обеспечивающее прием, обработку, хранение и выдачу информации. Каждое из указанных действий выполняет соответствующий блок ЭВМ: устройство ввода информации, устройство обработки информации — центральный процессор и устройство вывода информации.

Центральный процессор (далее процессор) содержит следующие основные функциональные устройства: арифметико-логическое устройство (АЛУ), оперативное запоминающее устройство (ОЗУ) и устройство управления (УУ). Взаимосвязь устройств компьютера обеспечивается адресными и информационными шинами связи.

На рис. 1.2 показана структурная схема компьютера.



Рисунок 1.2. Структурная схема компьютера

Дадим краткую характеристику основным компонентам схемы.

ОЗУ — память компьютера, предназначенная для временного хранения информации, участвующей в процессе обработки: обрабатываемая команда программы, необходимые исходные данные, промежуточные данные и результаты выполненной операции.

АЛУ — арифметико-логическое устройство, выполняющее команды (операции) по обработке информации в соответствии с кодом выполняемой команды. Например, сложение, вычитание, умножение, логическое преобразование и т. п.

УУ — устройство управления, определяющее последовательность выполнения операций и ее элементов, а также выдающее управляющие сигналы для записи и чтения информации. УУ отвечает за синхронизацию всех процессов обработки информации в компьютере. АЛУ, УУ и ОЗУ располагаются на системной (материнской) плате системного блока компьютера.

ДЗУ — долговременное запоминающее устройство, обычно выполняемое на жестком магнитном диске (винчестере), предназначенное для хранения информации и программ. ДЗУ входит в системный блок компьютера.

Адаптеры (контроллеры портов ввода-вывода) — преобразователи информации, необходимые для согласования скорости передачи информации между внешними устройствами (низкоскоростными устройствами и устройствами системного блока). Адаптеры формируются в виде отдельных карт,

устанавливаемых на системную плату, снабженных разъемом, к которому и присоединяются внешние устройства.

Клавиатура компьютера предназначена для ввода информации и команд управления. Обычно она выполнена в виде самостоятельной горизонтальной панели с размещенными на ней группами клавиш. Как правило, клавиатура содержит от 101 до 103 клавиш, которые размещены по стандарту QWERTY, что определяет последовательность букв в верхнем ряду алфавитной группы клавиш (слева — направо).

Первая группа клавиш содержит символные клавиши с нанесенными на них знаками латинского и русского алфавитов и специальных символов, включая и знаки препинания, а также строку цифр. Переключение клавиш со строчных на прописные осуществляется с помощью одновременного нажатия соответствующей клавиши и клавиши <Shift>.

В правой части клавиатуры размещена группа цифровых клавиш, предназначенная для постоянного ввода чисел. В этом случае включен режим <NumLock>, который может быть отменен с помощью клавиши <Caps Lock>, тогда этот блок клавиатуры будет выполнять операции управления курсором.

На первой горизонтальной линии клавиатуры расположена группа Функциональных клавиш типа «F», их назначение может быть запрограммировано. В этом случае их использование определяется конкретной программой.

Группа клавиш управления курсором размещена внизу между первой и второй группами. Обозначения на этих клавишах определяют их действия по перемещению курсора по экрану, как правило, при работе в соответствующих текстовых, табличных или графических системах.

Дисплей (монитор) предназначен для отображения графической и текстовой информации. Способность дисплея воспроизводить информацию определяется не только его техническими параметрами, но и используемой видеокартой ввода-вывода (*видеоадаптером*).

В настоящее время используются только цветные мониторы (кроме карманных ПК) с разрешающей способностью не менее 640 ч 480 пикселей (обычно — 1024 ч 768). Это означает, что экран разбит на 640 столбцов и 480 строк, в пересечении которых размещена светящаяся точка. Помимо разрешающей способности, экран характеризуется и своим размером по диагонали, обычно это 15 или 17 дюймов. Еще одной характеристикой экрана является плоскость экрана. Плоский экран обозначается термином «Flatron».

Существуют три технологические конструкции монитора: на базе электронно-лучевой трубки (ЭЛТ), жидкокристаллические и плазменные. Последние две конструкции имеют значительно меньшие габариты (ось Z), в пределах 10 см, в то время как «глубина» монитора на основе ЭЛТ составляет 35—50 см.

КЭШ-память (от англ. *cash* — наличные) — устройство сверхоперативной памяти компьютера, предназначенное для промежуточного хранения данных и команд. Предназначено для уменьшения времени простоя процессора.

1.2.2. Дополнительные устройства ввода-вывода информации

Принтер — устройство, предназначенное для вывода информации на «твердый» носитель, в качестве которого используются качественные листы бумаги. По организации непосредственно процесса печати принтеры делят на матричные, струйные и лазерные.

Матричные принтеры снабжены печатающей головкой, содержащей матрицу из 24 игл, определенная совокупность которых ударяет по красящей ленте, что и обеспечивает печать необходимого символа.

Струйные принтеры формируют изображение микрокаплями специальных чернил. Этот способ печати обеспечивает более качественную печать по сравнению с матричными принтерами, дает возможность цветной печати, но более требователен к бумаге.

Лазерные принтеры обеспечивают наилучшее качество печати, близкое к типографскому. В принтерах используется принцип ксерографии:

изображение сначала формируется на специальном барабане с помощью управляемого лазерного луча в виде электрических зарядов, а затем при вращении барабана над пеналом с красителем к соответствующим точкам барабана прилипает тонкий слой этого красителя, далее при прижимании поверхности барабана к бумаге этот слой переносится на бумагу. Дальнейший поворот барабана счищает остатки красителя и снова подает его поверхность под блок формирования изображения. Бумага с нанесенным порошком проходит под блоком, осуществляющим прогрев текста и закрепляющим порошок на бумаге.

Разрешающая способность лазерных принтеров составляет от 300 точек на дюйм, что определяет размер отдельной точки в 0,08 мм, а скорость таких принтеров может составлять от 5 до 15 сек. на страницу с текстом. При выдаче сложных рисунков время печати увеличивается.

Мышь — управляет движением курсора по экрану и активизирует выполнение функции, закрепленной за определенным полем экрана.

Стример — устройство записи информации на магнитную ленту (имеет объем в несколько Мбайт). Применяется обычно в информационных системах, а также в системах управления организациями для создания страховочных копий информации, обработанной за определенный промежуток времени (день, неделя и т. д.).

Модем — специальное устройство, которое обеспечивает преобразование цифрового сигнала в аналоговый (модуляция) и обратно из аналоговой формы представления сигнала в цифровую (демодуляция). Используется для приема и передачи информации по телефонным каналам связи, как правило, для подключения к глобальной информационной сети Интернет.

Модемы выпускаются в двух конструктивных исполнениях: встраиваемые и внешние.

Оптические диски. В зависимости от возможности использования оптических носителей для записи и считывания информации наибольшее распространение получили диски с однократной записью CD-R (CD-Recordable) и диски с многократной записью CD-RW.

CD-ROM-диски предназначены для считывания однократно записанной на диск информации. Наибольшее распространение среди оптических носителей этого класса получили оптические компакт-диски CD-ROM (Compact disk — Read Only Memory). Они появились в 1985 г. на рынке баз данных и представляют собой отпечатанную из пластмассы 4,72-дюймовую 120 мм при толщине 1,2 мм) круглую пластинку, на плоскости которой можно записать до 700 Мбайт информации. Наиболее перспективными областями применения оптической технологии CD-ROM считают: формирование баз данных различного типа, формирование словарей и терминологических баз данных, подготовка каталогов издаваемых книг, хранение и поиск юридической информации, распространение технической документации и др.

Сканеры предназначены для преобразования графической или текстовой информации в электронные цифровые данные. Их работа характеризуется следующими параметрами: разрешающая способность от 19 до 62 точек (пикселей) на 1 см; возможность распознавания оттенков до 256 уровней на каждый элемент изображения.

При сканировании (оцифровке) текст или графика освещается высокоинтенсивным лучом определенной полосы спектра. Отраженный свет фокусируется на светочувствительном преобразователе, который представляет собой линейку, состоящую из светочувствительных ячеек. Длина линейки определяет ширину сканирующего документа. Количество ячеек в линейке задает горизонтальное разрешение сканера.

Расстояние (или период времени) между двумя последовательными считываниями определяет вертикальное разрешение сканера. Светочувствительные ячейки в сканере преобразуют световую энергию в электронные сигналы, представляющие собой данные, которые сканируются. Эти сигналы пропорциональны интенсивности отраженного света, получаемого данной ячейкой. Нулевой сигнал, соответствующий отсутствию отраженного света, представляет «чисто черный» пиксель. Максимальный сигнал соответствует «чисто белому» пикселю. Полутона или серые тона генерируются с помощью промежуточного аналогового сигнала, уровень которого находится между «чисто белым» и «чисто черным» пикселями.

Результирующие данные передаются в компьютер и помещаются в оперативную память для распознавания и дальнейшей обработки.

При сканировании документов 20 × 25 см необходимо иметь сканеры с разрешением до 62 точек на 1 см². Это означает, что сканирование предполагает считывание до 12,8 млн бит необработанной информации. Для передачи 1 Мбит требуется около 8 минут. При использовании параллельного интерфейса пропускная способность может быть повышена более чем в 8 раз.

Требования высокой скорости обработки информации, передачи тонов яркости и др. уже привели к необходимости создания специализированных процессоров изображений, использования сопроцессоров, организации прямого доступа к памяти, буферизации. Сканеры, применяемые для передачи текста или изображения, оформляются в виде отдельных устройств ввода, хранения, обработки и выдачи документов.

1.2.3. Классификация компьютеров

С технической точки зрения современные компьютеры можно классифицировать на:

- ◆ суперкомпьютеры, содержащие до миллиона компьютеров, ведущие обработку информации в реальном масштабе времени и обслуживающие общегосударственные вычислительные сети: гидрометеоцентры, системы ПРО, космические системы и т. п.;
- ◆ мэйнфреймы, характеризуемые мультипроцессорной архитектурой и предназначенные для работы с большим объемом информации в реальном режиме времени;
- ◆ рабочие станции, предполагающие одновременную параллельную работу нескольких десятков компьютеров; используются для обслуживания систем автоматизированного проектирования;
- ◆ серверы, файловые процессоры — многопроцессорные компьютеры, обладающие повышенными характеристиками систем хранения информации (винчестеров), что позволяет их использовать в качестве ядра информационных систем корпораций и крупных фирм. Серверы необходимы также для хранения и распределения информации между пользователями сети;
- ◆ персональные компьютеры (ПК), предназначенные для реализации корпоративных сетей, а также для развития системы обучения, прикладных научных работ, активной деятельности в Интернете;
- ◆ ноутбуки, отличающиеся относительно малыми размерами при сохранении основных характеристик ПК. Они используются, как правило, в деловой практике и в системах маркетинга;
- ◆ карманные (мобильные) ПК, предназначенные, как правило, для оперативной работы в Интернете по принципу «в любой географической точке — в любое время»;
- ◆ специальные компьютеры, используемые в системах управления различными подвижными объектами специального назначения;
- ◆ игровые компьютеры, предназначенные для создания интеллектуальной среды, обеспечивающей изменение эмоционального состояния пользователя;
- ◆ технологические компьютеры, представляющие собой системы управления технологическими процессами на производстве;
- ◆ бортовые компьютеры, являющиеся системами автоматического управления движением, диагностированием и информационного обеспечения различных аппаратов.

Для определения функционального содержания компьютера используют последовательность составляющих его характеристик.

Примером возможного описания компьютера и его составляющих может служить следующая последовательность данных:

Тип микропроцессора:	Intel Pentium 4;
Тактовая частота:	3 ГГц;
Кэш	128 Мбайт;
Объем оперативной памяти (RAM):	256 Мбайт;
Объем памяти жесткого диска (HDD):	120 Гбайт;
Наличие гибкого диска, его размер (FDD):	3,5";
Тип и размер экрана дисплея:	Flatron LCD F700 (18,1", 1280×1024);
Клавиатура:	101 Key;
Устройство чтения с оптического диска:	CD 52x.

1.2.4. Сети передачи данных

Понятие «*сеть*» определяется как система взаимосвязи удаленных одна от другой вычислительных и/или терминальных систем с целью взаимного обмена данными между ними.

Наиболее известны следующие типы сетевых соединений: компьютер с компьютером, компьютер с одним или несколькими терминалами, терминал с терминалом одного компьютера.

Сеть имеет четыре компонента:

- 1) аппаратный связной интерфейс, управляющий работой связных интерфейсов с сетью;
- 2) совместимый алфавит;
- 3) протокол, необходимый для координации передачи сообщений и данных по сети, а также для защиты от ошибок;

4) сеть в виде кабеля или других форм для передачи сигнала, соединяющая все компьютеры и терминалы.

Сеть может быть построена на основе одного или нескольких типов сетей. Различают следующие виды информационных сетей:

- ◆ внутренняя сеть, использующая прямую связь с помощью двужильного кабеля между несколькими компьютерами, расположенными в одном здании;
- ◆ сеть на базе двужильных (двучечных) или многожильных телефонных каналов, использующая модемы;
- ◆ сеть на базе модемов и коммутируемых аналоговых каналов (например, телефонная сеть общего пользования);
- ◆ сеть, основанная на использовании выделенных цифровых каналов и сетевых оконечных устройств;
- ◆ цифровая сеть интегрального обслуживания;
- ◆ сеть на базе мультимплексов, использующая аналоговые или цифровые каналы и модемы или сетевые оконечные устройства;
- ◆ сеть коммутации пакетов, основанная на использовании рекомендаций X.25;
- ◆ локальная сеть.

Важным аспектом построения локальной информационной сети является выбор ее конфигурации (топологии). В зависимости от предъявляемых к сети требований ее структура может быть различной:

- 1) шинной (bus topology), при которой каждая рабочая станция связана с центральным компьютером и все узлы подключены к общему линейному информационному каналу; в случае выхода из строя одной станции нарушается вся сеть;
- 2) звездообразной (star), при которой каждый терминал соединен с центральным компьютером своим собственным каналом связи: при выходе из строя одной из станций не происходит нарушения работы всей сети;
- 3) кольцевой (ring), при которой терминалы соединены друг с другом, что позволяет обеспечить их взаимосвязь, минуя центральный компьютер.

По небольшой удаленности рабочих станций сети от центрального компьютера ее определяют как локальную сеть.

Структура локальной компьютерной сети состоит из хост-компьютера (в небольших сетях его функции выполняет файловый процессор — сервер), который может объединить отдельные компьютеры с компьютерной системой предприятия, рабочих станций, терминалов для реализации электронных справочных систем, средств подключения к локальной сети и к другим компьютерным системам.

Объединение компьютерных технологий предполагает наличие мощных компьютерных региональных центров, обеспечивающих реализацию взаимосвязи удаленных друг от друга компьютеров.

1.3. Программное обеспечение компьютера

1.3.1. Классификация программного обеспечения

ПО компьютера содержит операционные системы, программы-оболочки операционных систем, системы программирования, системы управления базами данных, прикладные программы, информационные системы управления и другие программы.

Операционные системы предназначены для управления взаимодействием технических средств компьютера, а также для наиболее эффективного их использования.

Примеры: MS DOS, дисковая операционная система (ДОС) фирмы Microsoft; System (OS Apple), Операционная система для компьютеров Макинтош; OS/2 (IBM); UNIX: Solaris (SunSoft), UnixWare (Novell) и др.

Командный язык операционной системы — средство взаимодействия ОС и пользователя, позволяющее выполнить разнообразные действия по созданию и обращению к каталогу, разметке винчестера и дискет, запуску программы и др.

Рассмотрим методы функционирования операционной системы на примере дисковой операционной системы MS DOS. В процессе своего развития MS DOS прошла следующие исторические (с технической точки зрения) этапы:

1981 г. — появилась система для 16-разрядного микропроцессора (80086) — MS DOS 1.0, она занимала 6 Кбайт оперативной памяти (версии 1.1, 1.25);

1982 г. — MS DOS 2.0 - 24 Кбайт (версия 2.25);

1984 г. — разработка для микропроцессора 80286 — MS DOS 3.0 и версия 3.1 для поддержки локальной сети;

1984 г. — переход на 3" дискету привел к созданию версии 3.2;

1985—1987 гг. — создание микропроцессора 80386 привело к появлению на рынке новой версии MS DOS 3.3 объемом в 6 Кбайт;

1987 г. — выход на рынок операционной системы OS/2 для компьютерных систем фирмы IBM;

1988 г. — появление следующей версии ДОС — MS DOS 4.0 — 75 Кбайт, которая могла использовать графическую оболочку Shell;

1990 г. — выход на рынок графической оболочки Windows 3.0;

1991 г. — дальнейшее развитие ДОС привело к появлению MS DOS 5.0;

1992 г. — развитие графической операционной среды Windows 3.1;

1993 г. — формирование одного из окончательных вариантов ДОС -MS DOS 6.0, MS DOS 6.2;

1995 г. — появление на рынке операционной графической среды Windows 95;

1998 г. — переход на Windows 98, который обеспечивал бесшовную связь с браузером Internet Explorer;

1999 г. — разработка графической операционной системы Windows 2000 с активизацией функций мультимедийных устройств и расширением возможностей работы в Интернет-пространстве;

2001 г. — выход на рынок графической операционной системы Windows XP, показавшей новые возможности общения с помощью компьютерной технологии и усилившей защитные свойства программной системы при работе в Интернете.

Программы-оболочки операционных систем реализуют средства диалога между пользователем и компьютером на уровне DOS путем использования небольшого набора клавиш клавиатуры компьютера или мыши, например Norton Commander, PCtools.

Системы программирования представляют собой совокупность языка программирования, среды программирования, языкового компилятора.

Язык программирования — формальный язык для описания данных (информации и алгоритма (программы) их обработки), определяемый его лексикой и грамматикой.

Среда программирования — совокупность вспомогательных средств, позволяющих упростить процесс ввода, отладки и исполнения программы пользователя, написанной на соответствующем языке программирования.

Языковой компилятор — программа преобразования текста программы (программного кода) в текст команд, исполняемых компьютером (машинный код). Существуют также и программы-интерпретаторы, обеспечивающие построчный перевод инструкций с языка программирования в набор машинных команд. Интерпретаторы, характеризующиеся тем, что занимают малый объем оперативной памяти, используются, как правило, в маломощных компьютерах с ограниченными пространственными объемами.

В качестве языков программирования в настоящее время используют язык низкого уровня Assembler и языки высокого уровня: Basic, C++, Fortran, Pascal. Особое значение для разработок ПО для работы в графической операционной среде имеют языки: Visual Basic, VC и др.

Системы управления базами данных (СУБД) позволяют формировать и обрабатывать упорядоченные массивы данных. СУБД могут использовать свой язык программирования и свой компилятор. Например, Access, SQL и др.

Прикладные программы — совокупность программ пользователя компьютера, созданных или приобретенных им для обеспечения эффективного выполнения функций по обработке информации.

1.3.2. Файловая система хранения информации

В основе использования ресурсов компьютера лежит понятие «файловая система». На его основе реализуются методы разделения и управления памятью на внешних магнитных носителях — гибких и жестких магнитных дисках, CD.

Файл — это место постоянного хранения информации (программ текстов, баз данных и т. п.). Каждый файл имеет свое имя, зарегистрированное в каталоге (директории) носителя информации.

Каталог доступен пользователю через командный язык DOS — его можно озаглавить, просмотреть, переименовать. В нем можно изменять имена зарегистрированных файлов, заводить новые файлы и каталоги, удалять как файлы, так и каталоги.

Имя каталога жестко связано с именем диска.

В MS DOS накопители (магнитные и оптические диски) определяют корневые каталоги; они именуются последовательно A:, B:, C: и т. д.

На винчестере могут размещаться несколько корневых каталогов, при написании имен которых используются знаки латинского алфавита, например: D:, E:, F:, G:, ..., Z:.

Каталоги других уровней называются вложенными. Они могут иметь произвольные имена. Например: Student, ARC, AntiVirus (рис. 1.3).

Каталог, с которым в настоящий момент работает пользователь, называется *текущим*.

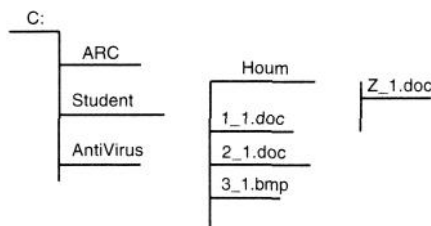


Рисунок 1.3. Древоподобная структура взаимосвязи каталогов и файлов различных уровней

В каталогах размещаются каталоги (подкаталоги) и файлы (программы, исходные данные, результаты, тексты и другая информация). Например, каталог (папка) с файлами студента, как показано на рис. 1.3, может размещаться в корневом каталоге C: под именем Student. Кроме нее, в корневом каталоге находятся каталог с архивирующими программами ARC и каталог с антивирусными программами AntiVirus.

В каталоге Student могут находиться файлы и вложенные каталоги. Например, текстовые файлы-отчеты о выполненных работах: 1_1.doc, 2_1.doc, другие файлы, а также вложенный каталог Houm с файлом заданий к первой лабораторной работе Z_1.doc.

Имя каталога в MS DOS не должно превышать 8 знаков. (Операционные графические среды не ограничивают число знаков в имени файла и каталога, однако при переходе в MS DOS «лишние» знаки будут утеряны и заменены одним знаком «~».)

В файловой системе используется понятие «*путь к файлу*» — маршрут от текущего каталога или от корневого каталога диска к тому каталогу, в котором находится необходимый файл. Задается последовательностью имен каталогов, разделенных символом «\» (левый слеш).

Если путь начинается с символа «\», то маршрут определяется от корневого каталога диска, иначе — от текущего каталога.

Например: c:\student\houm\Z_1.doc.

Каждому файлу присваивается имя и расширение. В имени файла может быть от 1 до 8 символов.

Расширение имеет не более 4 символов: точка и 1—3 символа, например: process.exe, fond.dbf, bcs_pr.bat. Расширение показывает на принадлежность файла к соответствующей программе обработки файла. Например, расширение .doc определяет принадлежность файла к текстовому процессору Word, расширение .dbf — к базе данных, расширение .txt — к текстовому файлу, расширения .com и .exe показывают, что данный файл является запускающей программой, и т. д.

Файл может не иметь расширения. В качестве символов в имени и расширении файла следует использовать только знаки латинского алфавита или разнообразные символы и цифры, кроме точки и пробела. (При работе в графической операционной среде можно использовать и кириллицу, и пробелы, однако при переходе в среду MS DOS они могут плохо читаться.) Запрещается использовать имена файлов, которые совпадают с логическими именами устройств компьютера: AUX, COM1 — COM3, LPT1 — LPT3, CON, PRN, NUL.

Полное имя файла — это последовательность знаков, определяющая путь к файлу и имя файла. Например, [дискковод:][\путь\]имя файла (в прямоугольных скобках обозначаются необязательные элементы).

Если дискковод не указан, то подразумевается текущий дискковод. Если путь не указан, то подразумевается текущий каталог.

1.4. Дискровая операционная система MS DOS

1.4.1. Состав и назначение MS DOS

DOS состоит из следующих основных модулей и программ:

- ◆ базовая система ввода-вывода (BIOS — Basic Input/Output System);
- ◆ блок начальной загрузки (BOOT RECORD);
- ◆ модуль расширения базовой системы ввода-вывода (IO.SYS);
- ◆ модуль обработки прерываний (MSDOS.SYS);
- ◆ командный процессор (COMMAND.COM);
- ◆ модуль сжатия данных (DBLSPACE.BIN);
- ◆ утилиты DOS (FORMAT.COM, FDISK.EXE, LABEL.EXE и др.);
- ◆ загружаемые драйверы устройств (HIMEM.SYS, RAMDRIVE.SYS и др.);
- ◆ файлы AUTOEXEC.BAT и CONFIG.SYS.

Основные функции дисковой операционной системы можно свести к двум основным:

- ◆ организация взаимодействия устройств компьютера в процессе выполнения программы;
- ◆ управление работой компьютера с помощью командного языка: запуск программ, форматирование дисков, работа с файлами, установка режимов работы дисплея, печать информации и др.

1.4.2. Распределение памяти в MS DOS

Карта оперативной памяти микропроцессора состоит из двух основных частей:

- ◆ основной памяти (conventional) — 0—640 К;
- ◆ области верхней памяти (Upper Memory Area — UMA) — 640—1024 К.

Для использования возможностей адресуемой памяти процессоров 80286 и старше используют расширенную (expanded — EMS) и дополнительную (extended — XMS) память.

Расширенная память (EMS). Действие расширенной памяти основано на принципе «замещения страниц» — виртуальной памяти. В основной памяти и в области верхней памяти выделяются окна в 64 К, в каждом из которых создаются по четыре страницы EMS. В страницах показываются адреса данных, которые не принадлежат адресному пространству процессора, а находятся на жестком диске. Соответствие логических адресов EMS физическим адресам определяется специальной программой, обеспечивающей доступ ко всей информации расширенной памяти.

Дополнительная память (XMS). Увеличение физического объема оперативной памяти позволяет размещать в ней область электронного (виртуального) диска, расширяя адресное пространство до нескольких Гбайт.

Высокая память (ХМА). Эта память дает возможность увеличивать адресное пространство процессора от 1 Мбайта до 1 Мбайта + 64 Кбайта.

Блоки верхней памяти (УМВ). Свободные области верхней памяти, зарезервированные для системных нужд, используются по принципу расширенной памяти, при этом эти области разделяются на блоки, куда загружаются данные и программы.

1.4.3. Файловая структура диска

Организация дискового пространства определяется схемой изменения данных на диске и построением базовой системы ввода-вывода.

Данные на диске размещены по секторам. Адрес сектора определяется следующими координатами:

- ◆ номер дорожки (цилиндра), для дискеты: 00—39 или 00—79;
- ◆ номер поверхности (стороны), для дискеты: 0 или 1;
- ◆ номер сегмента, для дискеты: от 9 до 15.

Базовая система ввода-вывода (BIOS) имеет следующие основные функции:

- ◆ тестирование устройств компьютера;
- ◆ вызов блока начальной загрузки с диска в оперативную память и передача ему управления для загрузки остальных модулей DOS;
- ◆ обработка системных прерываний для выполнения различных операций.

В начало диска устанавливается блок начальной загрузки (BOOT RECORD). Он размещается на диске при его форматировании независимо от того, является диск системным или нет. Его назначение – считывание в оперативную память модулей MS DOS: IO.SYS и MSDOS.SYS.

BOOT RECORD размещается на 0 стороне, в 1-м секторе 00 дорожки. Длина BOOT RECORD составляет 512 байт, что равно объему одного сектора (кластера).

При обращении к блоку начальной загрузки система выполняет следующие операции:

а) поиск на системном диске файлов IO.SYS и MSDOS.SYS (они размещены в начале диска последовательно. Если эти файлы не обнаружены, то диск считается несистемным. В этом случае выдается соответствующее сообщение и работа компьютера останавливается);

б) загрузку найденных файлов и передачу им управления. Модуль расширения базовой системы ввода-вывода (IO.SYS) имеет следующее назначение:

- ◆ определяет перечень внешних устройств (драйверов), подключаемых к компьютеру;
- ◆ завершает загрузку в память DOS: загрузка командного процессора с диска в оперативную память и передача ему управления.

Модуль обработки прерываний (MSDOS.SYS) предназначен для:

- ◆ обеспечения работы файловой системы, устройств ввода-вывода информации (дисплей, клавиатура, принтер, последовательные порты);
- ◆ завершения программ, преждевременного прерывания программ, обработки ошибок.

Основной программой дисковой операционной системы можно считать командный процессор — COMMAND.COM, который имеет следующие основные функции:

- ◆ прием и обработка команд, вводимых с клавиатуры или полученных из командного файла;
- ◆ выполнение внутренних (встроенных) команд;
- ◆ загрузка в память и выполнение внешних программ и утилит DOS;
- ◆ вывод на экран диагностических сообщений;
- ◆ запуск файла AUTOEXEC.BAT при загрузке машины.

При загрузке файла, имя которого не совпадает с именами внутренних команд, анализируется его расширение.

При расширении .bat файл считается командным, и строки этого файла исполняются последовательно. Каждая строка этого файла представляет собой команду, метку или комментарий. Если в строке находится имя вызываемой программы, то ей передается управление. После завершения работы программы управление возвращается командному процессору.

При расширении .exe осуществляется размещение программы в оперативной памяти и запуск программы.

При расширении .com программа занимает не более одного сегмента оперативной памяти (64 К), поэтому настройка адресов после ее загрузки не производится и программе передается управление.

Исполнение файлов с другими расширениями не осуществляется.

Модуль сжатия данных (DBLSPACE.BIN) предназначен для: сжатия данных при работе с дисковыми накопителями с целью экономии дискового пространства до 30%. Модуль загружается резидентно в оперативную память компьютера. Активизация модуля осуществляется с помощью драйвера DBLSPACE.SYS и DBLSPACE.EXE.

1.4.4. Утилиты MS DOS

MS DOS содержит более 70 утилит (программ) для выполнения различных функций. Они размещаются, как правило, в отдельном каталоге.

Файл конфигурации системы (CONFIG.SYS) определяет начальное состояние компьютера: подключаемые дополнительные драйверы, число буферов памяти для обмена информацией, клавиши прерывания исполняемой программы, тексты исходных сообщений, выдаваемые на экран, и т. д.

Основные команды файла CONFIG.SYS:

1) BREAK — прерывание выполнения программы при нажатии клавиш <Ctrl> + <C> или <Ctrl> + <Break>;

формат: BREAK ON, BREAK OFF;

2) BUFFERS — резервирование числа буферов для ускорения операций ввода-вывода информации;

формат: BUFFERS = *i,j*,

где *i* — число буферов (1—99);

j — объем буфера предввода (1—8): предназначен для считывания секторов на диске, непосредственно следующих за затребованным сектором.

Ускорение работы с дисками с помощью BUFFERS.DOS осуществляется за счет считывания данных с диска по секторам (512 байт). Если размер запрашиваемых данных больше размера сектора, DOS считывает несколько секторов, записывает их в заранее подготовленные буфера (532 байта ОЗУ) и дальше работает не с дисками, а с данными из этих буферов.

Максимальный объем буфера составляет 99, что затребует 51 Кбайт ОЗУ;

3) COUNTRY — установка национальных форматов: даты/времени и специальных символов; денежных единиц; десятичных разделителей; разделителей данных в списках. Здесь используется следующий формат:

COUNTRY = код_страны, кодовая_страница, файл_Country.sys. Пример: COUNTRY=001, 437, C:\DOS\COUNTRY.SYS;

4) DEVICE — подключение загружаемых драйверов устройств; Формат: DEVICE = файл_драйвера [параметры].

Пример: DEVICE=C:\DOS\ANSI.SYS;

5) DEVICEHIGH — загрузка драйверов устройств в область старшей памяти (UMB); его формат: DEVICEHIGH = файл_драйвера [параметры].

Пример: DEVICEHIGH=C:\MOUSE\MOUSE.SYS. Перед использованием директивы DEVICEHIGH необходимо записать: DEVICE=c:\DOS\HIMEM.SYS;

6) DOS=UMB.

DOS — загрузка операционной системы в область дополнительной памяти (XMS) или в блоки старшей памяти; в этом случае применяется формат: DOS = параметр.

Параметр: HIGH!LOW — DOS будет загружена в дополнительную (свыше 1 Мбайт)/основную память;

UMB!NOUMB — DOS будет/не будет свободными блоками старшей памяти (UMB);

HIGH,UMB — DOS загружается в дополнительную память (XMS), свободные блоки старшей памяти будут доступны прикладным программам.

Перед загрузкой необходимо записать: DEVICE=HIMEM.SYS;

7) FILES — задание числа одновременно открытых файлов; формат команды: FILES = число_файлов (число файлов задается в

диапазоне 8—255);

8) LASTDRIVE — определение максимального количества логических дисков компьютера;

формат: LASTDRIVE = символ, где символ — буква латинского алфавита от A до Z; например: LASTDRIVE=E;

9) MENUDEFAULT — установка пункта стартового меню DOS по умолчанию при загрузке системы;

формат: MENUDEFAULT = имя_блока, врем_задержка, где имя_блока — имя блока директив, принимаемое для выполнения по умолчанию;

врем_задержка — установка временной задержки перед выполнением блока директив, по умолчанию от 0 до 90. Например: MENUDEF-FAULT=NORMAL,5, где NORMAL — пункт меню;

10) MENUITEM — задание варианта обработки файла CONFIG.SYS в стартовом меню DOS;

формат: MENUITEM = имя_блока, текст, где имя_блока — блок директив обработки файла CONFIG.SYS;

текст — символьная строка до 70 символов. Параметр необязателен;

11) MENUCOLOR — определение цвета символов и фона в стартовом меню запуска;

формат: MENUCOLOR = X, Y, где X - код цвета символов, Y— код цвета фона;

12) REM — комментарий в строке CONFIG.SYS;

формат: REM-сообщение, где сообщение — любая символьная строка;

13) SHELL — указание места расположения командного процессора;

формат: SHELL = полное_имя_командного_процессора /ключи.

Ключи имеют следующее наполнение:

/E: — размер системного окружения от 160 до 32 768, кратное 16;

/P — процессор полностью резидентен в памяти и из него нельзя выйти, используя команду EXIT;

/MSG — в оперативной памяти сохраняются тексты аварийных сообщений. Ключ используется совместно с ключом /P.

Например: SHELL=C:\DOS\COMMAND.COM /E:1024 /P.

По этой команде размер системного окружения увеличивается до 1 К, а командный процессор резидентен в памяти. В этом случае файл AUTOEXEC.BAT должен содержать следующую строку:

SET COMSPEC=C:\DOS\COMMAND.COM. Пример содержания файла CONFIG.SYS:

BREAK=on — прерывание исполняемой программы при нажатии клавиш <Ctrl> + <Break>;

FILES=20 — открываются одновременно до 20 файлов (по ум. 8);

BUFFERS=30 — организуются 30 буферов по 512 байт (по ум. 2);

DEVICE=mouse.sys — вызывается драйвер mouse.sys.

Пример организации меню с помощью файла CONFIG.SYS:

MENUITEM BASE_CONFIG, BASE CONFIGURATION

MENUITEM FULL_CONFIG, FULL CONFIGURATION

[BASE_CONFIG]

DOS = HIGH

DEVICE = C:\DOS\HIMEM

[FULL_CONFIG]

INCLUDE BASE_CONFIG

DOS = UMB

DEVICE = C:\DOS\EMM386 RAM

DEVICE = C:\DOS\RAMDRIVE.SYS 512

Файл автозапуска (AUTOEXEC.BAT), осуществляющий настройку системы, вызывается операционной системой автоматически после COMMAND.COM.

AUTOEXEC.BAT состоит из набора команд, обеспечивающих вызов соответствующих программ.

Наиболее часто используются следующие команды:

1) PROMPT — установка вида приглашения системы;

2) @ — подавление вывода на экран строки командного файла, перед которой установлен символ «@»;

формат: @<строка командного файла>;

3) CALL — вызов программы с последующим возвращением к командному файлу;

формат: CALL <имя файла>;

4) CHOICE — выбор варианта командного файла;

формат: CHOICE /C:<ключи-символы> /N /S /T:c,nn <текст>, где /C — разрешение ввода-вывода информации, указанной в поле «ключи-символы»; ключи-символы — перечисление ключей, которые будут обрабатываться командой CHOICE; /N — запрещение вывода информации из поля «ключи-символы»; /S — использование верхнего и нижнего регистра для ввода знаков ключа; /T:c,nn — время паузы до ввода знака ключа (nn), c — вариант работы команды, принимаемый по умолчанию.

В результате выполнения команды выдается код завершения программы, который соответствует номеру выбранного ключа. Этот код определяет состояние условия ERRORLEVEL. Далее, используя команду условия IF и GOTO, вызывают необходимую программу.

Например, для вызова на экран меню, чтобы определить выбор необходимой программы после загрузки MS DOS: 1 — NC, 2 — DRVEB, 3 — PCTOOLS, следует записать в тело программы AUTOEXEC.BAT следующие команды:

CLS

ECHO A NORTON COMMANDER

ECHO B ANTWIRUS

ECHO C PCTOOLS

CHOICE /C:ABC CHOOSE AN OPTION

IF ERRORLEVEL 1 GOTO NC

IF ERRORLEVEL 2 GOTO DRVEB

IF ERRORLEVEL 3 GOTO PCTOOLS

:NC

```

NC
GOTO END
:DRVEB
DRVEB
GOTO END
:PCTOOLS
PCTOOLS
GOTO END
:END

```

5) ECHO OFF/ON — запрещение/разрешение отображать на экране команды прерываний работы программ или строки командного файла;

6) PAUSE — приостановка выполнения командного файла;

7) REM — комментарий в командном файле;

8) PATCH — задание последовательности каталогов, просматриваемых системой MS DOS при поиске выполняемых файлов;

9) SET — установка имени и параметров исполняемой программы. Для запуска резидентных программ используют следующие команды:

10) DOSKEY — обеспечение использования «быстрых» клавиш;

11) VSAFE — слежение за появлением вирусов в компьютере;

12) SMARTDRV — ускорение доступа к жесткому диску. Например:

```

ECHO OFF                                1
PROMPT $P$G                              2
REM ***** MOUSE SETUP *****        3
PATH C:\;C:\DOS;C:\NORTON;              4
SET TEMP=C:\WINDOUS\TEMP                 5
C:\MOUSE\MOUSE CENHANCE                  6
RFM *****                              7
NC                                         8

```

Покажем действие, выполняемое DOS после анализа каждой строки файла AUTOEXEC.BAT:

1 — отключить комментарий процесса исполнения команд;

2 — формат приглашения DOS-командной строки:

\$p — выдача имени текущего каталога;

\$g — выдача символа-разделителя > ;

\$t — выдача текущего времени;

\$d — выдача текущей даты и др. Для данного примера: C:\>;

3 и 7 — строки-комментарии — не обрабатываются;

4 — устанавливает маршруты для поиска файлов (в каталогах DOS, NORTON);

5 — указывает имена каталогов, где расположены рабочие файлы (базы данных, тексты и т. п.);

6 — вызов исполняемой программы MOUSE из каталога MOUSE (с указанием вида контроля исполнения программы CENHANCE);

8 — вызов программы Norton Commander.

Для обхода выполнения файлов настройки следует нажать клавишу <Shift> или <F5> при появлении надписи на экране «Запуск MS DOS».

1.4.5. Драйверы устройств в MS DOS

Драйверы — программы специального типа, ориентированные на управление внешними устройствами компьютера, в качестве которых могут выступать: дисплей, клавиатура, гибкие и жесткий диски, CD, принтер, графопостроители, планшеты, мышь, модемы (для связи по телефонным линиям), контроллеры локальных сетей, аналого-цифровые преобразователи и другое оборудование.

Работа отдельных устройств компьютера определяется встроенными драйверами, однако настройка этих устройств, а также подключение дополнительных устройств может быть определена с помощью программ-драйверов, обращение к которым помещается в файле CONFIG.SYS.

Приведем их перечень:

ANSI.SYS — обеспечивает дополнительные возможности: чтения текущего положения курсора, установку цвета символов и фона, переназначение клавиш, позиционирование курсора;

DISPLAY.SYS — поддерживает переключение кодовых страниц для монитора;
DBLSPACE.SYS — помещает модуль DBLSPACE.BIN на предназначенное ему место в памяти;
DRIVER.SYS — создает логический диск, который определяет дополнительный физический дисковод для гибких дисков;
EGA.SYS — организует изображение на дисплее при работе оболочки MS DOS Task Swapper с EGA-монитором;
EMM386.SYS — организует расширенную память и обеспечивает доступ к области старшей памяти на компьютерах с процессором типа 80386 и выше, которые имеют дополнительную память;
HIMEM.SYS — управляет использованием дополнительной памяти на компьютерах с процессором типа 80286 или выше;
RAMDRIVE.SYS — создает виртуальный диск в оперативной памяти компьютера;
SETVER.EXE — загружает таблицу соответствия версий системы MS DOS прикладным программам;
SMARTDRV.EXE — организует дополнительную или расширенную память (кэш) для дисковых накопителей, обеспечивая ускорение операций ввода-вывода.

1.5. Выполнение команд MS DOS

После включения компьютера и загрузки в режиме операционной системы на экране появится информация о текущем каталоге, например: C:\.

Для того чтобы ввести команду, ее необходимо набрать на клавиатуре, контролируя текст на экране дисплея. Затем следует нажать клавишу **<Enter>**.

В процессе ввода информации возникают различные ситуации, из которых можно выйти, используя ряд клавиш клавиатуры. Укажем основные из них.

- ** — удаление текущего символа;
- <BkSp>** [Backspace] — стирание предыдущего символа;
- <Ins>** — включение/выключение режима вставки пропущенных знаков;
- <клавиши-стрелки>** — перемещение курсора;
- <Esc>** — удаление всего текста из командной строки;
- <Shift + PrtSc>** — вывод текущего содержания экрана на принтер;
- <Ctrl> + <Alt> + ** — перезагрузка DOS.

Для прекращения выполнения любой команды необходимо одновременно нажать две клавиши **<Ctrl> + <C>** или **<Ctrl> + <Break>**.

Если необходимо приостановить процесс вывода информации на экран дисплея, то следует нажать клавиши **<Ctrl> + <S>**. Для продолжения вывода на экран нужно повторно нажать на эти клавиши.

Командная строка, набранная на клавиатуре и отправленная на выполнение нажатием клавиши **<Enter>**, копируется в буфер командной строки объемом в одну команду.

С помощью функциональных клавиш можно извлекать и редактировать содержимое этого буфера, повторно используя предыдущую команду. Приведем перечень этих клавиш:

- <F1>** — копирует из буфера один символ;
- <F2>** — копирует из буфера все символы до символа, введенного вслед **<F2>**;
- <F3>** — копирует все содержимое буфера;
- <F4>** — удаляет в буфере все символы до символа, введенного вслед **<F4>**;
- <F5>** — помещает текущую командную строку в буфер без ее выполнения, что позволяет ее редактировать с помощью функциональных клавиш;
- <F6>** — помещает в файл, копируемый с клавиатуры на диск, символ конца файла — **<Ctrl> + <Z>**.

Команды MS DOS состоят из непосредственно имени команды и параметров, разделенных пробелами. Имя команды следует набирать, используя только латинский алфавит.

Команда смены корневого каталога

Для смены корневого каталога следует набрать имя каталога, который должен стать текущим, и двоеточие, например:

- A:** — переход на каталог дисковода A:;
- C:** — переход на каталог дисковода C:.

Для выполнения команды необходимо нажать клавишу **<Enter>**.

Создание каталога

Для создания нового каталога используется команда **Md** (Make Directory). Формат команды:

Md [дискковод:]путь.

Например, Md c:\bcs — по этой команде будет создана директория BCS в корневом каталоге диска C:.

Просмотр содержания каталога

Вывод на экран содержания каталога осуществляется с помощью команды **Dir**. При выполнении команды показывается имя текущего каталога, его отношение к корневому каталогу, перечень содержащихся в нем файлов с указанием их объема, даты и времени и подкаталогов. Формат команды:

Dir [дискковод:]путь[имя_файла]/P/W.

В команде используются ключи /P и /W. Покажем их назначение:

/P — постраничный вывод информации на экран. Для перехода на следующую страницу экрана следует нажать любую клавишу;

/W — вывод имен файлов и подкаталогов без дополнительной информации.

Если в команде не указано имя диска или путь, то подразумевается текущий диск или каталог. Если указано имя файла, выводится информация о его содержании, например, команда Dir C:\bcs выводит на экран информацию о каталоге BCS. Если информацию необходимо записать в файл, то следует воспользоваться следующей командой:

Dir > report.txt — текущий каталог записывается в указанный файл.

Изменение текущего каталога

Для перехода в подкаталог и обратно следует применять команду **Cd** (Change Directory). Формат команды:

Cd [дискковод:] путь.

Например, **Cd dcs** — переход в директорию BCS из корневого каталога.

Если необходимо вернуться в каталог из подкаталога, следует записать: Cd ...

Удаление каталога

Для удаления каталога его следует сначала очистить, удалив все находящиеся там файлы и каталоги, а затем использовать команду **Rd** (Remove Directory). Формат команды:

Rd [дискковод:] путь.

Например: **Rd C:\client** — удаление подкаталога CLIENT в корневом каталоге диска C:.

Удаление файлов

Для удаления файлов используется команда **Del**. Ее формат: Del [дискковод:]путь\имя_файла.

Если не указано имя файла, то при выполнении команды будут удалены все файлы, находящиеся в указанной директории.

Команда **Del** не удаляет директории.

Для удаления группы файлов можно использовать знак «*».

Переименование файлов

Для переименования файлов следует применять команду **Ren** (Rename), ее формат:

Ren[дискковод:] [путь\]имя_файла имя_файла.

Первое имя файла в команде задает имя (имена) переименовываемых файлов, второе — новое имя файлов.

В именах файлов можно употреблять символы «*» и «?».

Символ «*» обозначает любое число любых символов в имени файла или его расширении.

Символ «?» обозначает один произвольный символ или отсутствие символа в имени файла или его расширении.

Например: **Ren c:*.doc *.txt** — команда переименует все файлы с расширением .doc в корневом каталоге диска C: в файлы со старыми именами, но с новым расширением .txt.

Копирование файлов

Для копирования файлов используется команда **Copy**. Формат команды:

Copy[диск:][путь\]имя_файла[диск:][путь\]имя_файла.

Команда выбирает из каталога, указанного в ее первом параметре, соответствующий файл и копирует в новый каталог под новым именем, если оно указано.

Символы «*» и «?» в имени файла во втором параметре позволяют сохранять фрагменты имени копируемого файла, например:

Copy a:\process.exe c:\bcs*.* — копирование программы process.exe с дискеты в директорию BCS, расположенную на диске C:;

Copy a:*.* c:\bcs*.* — копирование всех файлов с диска A: на диск C: в директорию BCS.

Команда **Copy** может быть применена для ввода информации в файл с клавиатуры компьютера и для вывода содержимого файла на печать. Для этого следует вместо имен файлов использовать обозначения соответствующих устройств, например:

Copy c:\report.txt con,

где con — консоль, содержащая клавиатуру для ввода и дисплей для вывода информации. При окончании ввода информации на экран дисплея следует одновременно нажать две клавиши <Ctrl> + <Z> или функциональную клавишу <F6>;

Copy report.txt prn, где prn — принтер.

Очистка экрана дисплея

Для очистки экрана дисплея применяется команда **Cls**.

Формат команды: **Cls**.

Форматирование (инициализация) дискет

Для форматирования (инициализации) дискет используется команда **Format**.

Формат команды:

Format <дискковод>:[Д>][5][y][4] .

/b — на диске резервируется место для системных файлов, которые позже копируются на диск командой SYS;

/s — в процессе форматирования на диск копируются системные файлы, и он становится загружаемым. Ключи /s и Д) несовместимы;

/v — перед началом форматирования запрашивается значение 11-сим-вольной метки тома;

/4 — на дисковом, рассчитанном на форматирование дискет емкостью в 1,2 Мбайт, форматировается диск объемом 360 Кбайт.

В процессе форматирования диска все находящиеся на нем файлы будут уничтожены.

1.6. Программа-оболочка Norton Commander

При работе на компьютере пользователь может активно использовать команды дисковой операционной системы. Однако написание самих команд, которые к тому же могут повторяться, — весьма трудоемкое занятие. Для облегчения этого процесса созданы специальные программы-оболочки, которые позволяют вызывать нужные команды и Файлы путем использования небольшого набора клавиш клавиатуры компьютера или щелчком мышью по обозначениям этих клавиш. Такие программы построены по принципу «указательного пальца», который однозначно показывает на нужный объект и говорит, что с ним сделать, используя небольшой набор требований типа приказа «хочу это».

Рассмотрим реализацию этого принципа на примере одной из наиболее удачных программ-оболочек — программы **Norton Commander** (NC).

Если на компьютере заранее записана программа Norton Commander, то следует набрать команду NC, чтобы начать работать с ней.

После запуска Norton Commander на экране компьютера появляются два прямоугольных окна, ограниченные двойной рамкой, которые называют панелями (рис. 1.4). В случае цветного экрана панели имеют синий цвет, а буквы на них отображаются белым цветом. В правом верхнем углу может располагаться небольшое окно с указанием текущего времени. Под панелями находится строка с мигающим курсором, на которой может быть записана команда DOS.

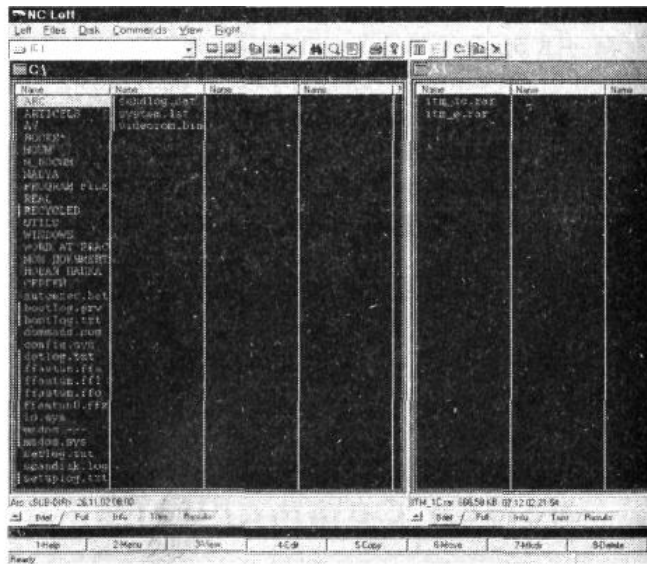


Рисунок 1.4. Панели программы Norton Commander

Ниже, под командной строкой DOS, располагается строка, поясняющая назначение функциональных клавиш для реализации операций Norton Commander.

На каждой панели экрана может располагаться различная информация: оглавление каталога, дерево каталогов на диске, информация об используемой оперативной и дисковой памяти.

Над панелями имеется указатель содержания панели. Если на панели показано оглавление каталога, то в указателе показано имя этого каталога и путь к нему. Если на панели показано дерево каталогов на диске, то выводится слово **Tree**. Если на панели отражена сводная информация об используемой памяти, то сверху выводится **Info**.

Панель с оглавлением каталога имеет четыре столбца. Если выводится полная информация, то в первом столбце слева, имеющем имя **Name**, показаны имена файлов, которые записаны строчными буквами, и подкаталогов, которые записаны прописными буквами. Во втором столбце, обозначенном словом **Size**, указано обозначение подкаталогов <SUB-DIR> или объем файлов в байтах.

Третий столбец **Date** содержит информацию о дате записи файла или подкаталога.

Четвертый столбец, обозначенный **Time**, указывает время занесения информации.

Самую верхнюю строку оглавления занимает информация в виде двух точек и знаков <UP-DIR>, которая указывает на работу в подкаталоге. В корневом каталоге эта строка отсутствует.

Работа с программой-оболочкой начинается с указания выбранного файла или каталога. Для этого с помощью стрелок клавиатуры устанавливается окно, выделенное контрастным (инверсным) цветом, соответствующий файл или каталог. В этом случае окно используется как палец малыша, обозначающий выбранную игрушку. Роль слов «хочу это» выполняет клавиша <Enter>, при нажатии на которую будет выполнена команда, соответствующая расширению файла. Расширения .com, .exe, .bat обеспечат выполнение указанного файла (программы).

Действия, выполняемые программой для других расширений, определяются файлом NC.EXT. При отсутствии этого файла оболочка не будет реагировать на клавишу <Enter> при указании на файлы с другими расширениями, кроме описанных в указанном файле.

Для перехода с одной панели экрана на другую следует использовать кнопку табуляции — <Tab>. При необходимости перехода из подкаталога в каталог (в родительский каталог) следует передвинуть окно на верхнюю строку оглавления с помощью клавиш-стрелок или вниз и нажать клавишу <Enter>.

Покажем назначение функциональных клавиш, о которых говорится³ в нижней строке экрана дисплея. <F1> — **Help** — помощь. При нажатии на эту клавишу на экране показывается краткая информация о назначении клавиш клавиатуры для выполнения операций в Norton Commander.

Помощь является контекстно-зависимой, т. е. на экране появляется разъяснение той ситуации, которая обрабатывается программой.

<F2> — **Menu** — работа с программами, указанными в пользовательском меню. В таком меню каждой вызываемой программе ставится в однозначное соответствие одна из клавиш клавиатуры, при нажатии на которую вызывается соответствующая программа.

Такой режим работы компьютера достаточно эффективен, если выполняется узкий круг программ или если на одном компьютере работают несколько пользователей, которым следует обращаться только к определенному перечню программ. Существует два типа пользовательских меню: главное и локальное. Они имеют одинаковые имена — `ps.mnu`, однако главное располагается вместе с Norton Commander, а локальное — в текущем каталоге.

Главное пользовательское меню вызывается в том случае, если в текущем каталоге нет локального пользовательского меню.

<F3> — View — просмотр содержимого файлов. Эта клавиша позволяет отразить содержимое файла в удобном для чтения виде.

Текстовые файлы показывают информацию, записанную по строкам с соблюдением абзацев, записи баз данных показываются как последовательности строк-реквизитов и т. д. В процессе просмотра используют клавиши перемещения курсора.

Совместное использование двух клавиш **<Shift> + <F3>** позволит ввести текст того файла, содержимое которого следует просмотреть. Нажатие клавиш **<Alt> + <F3>** показывает содержимое файла без его интерпретации с помощью ранее показанных программ.

<F4> — Edit — редактирование файла. Нажатие этой клавиши вызывает текстовый редактор Norton Commander, с помощью которого можно ввести с клавиатуры нужный текст программы или сообщения.

В качестве текстового редактора можно использовать другой (альтернативный) редактор, который будет вызван одновременным нажатием клавиш **<Alt> + <F4>**. Однако перед этим необходимо в главном меню **Norton Editor (<F9>)** выбрать пункт **Options** и установить курсор на строке **Editor**. После нажатия клавиши **<Enter>** переместить курсор на строку **External** и записать имя альтернативного редактора, после которого через пробел записать «**!!**».

<F5> — Copy — копирование файла. Эта клавиша позволяет реализовать команду DOS `Copy`, при этом функция оболочки заключается в указании того файла (или группы файлов), который следует копировать, и пути к месту хранения копии. Попутно программа проверяет возможность повторного копирования файла в ту же область памяти компьютера.

Сначала покажем, как выделить группу файлов. Для этого следует использовать клавишу **<Ins>**, которая нередко совмещена с клавишей **<0>** на цифровой части клавиатуры. Ее нажатие помечает отобранные файлы. Если следует пометить все файлы или файлы, имеющие совпадение отдельных частей имени, то необходимо воспользоваться клавишей **<+>**, при отмене выделенных файлов используют клавишу **<->**.

После выделения файлов для копирования устанавливается путь к месту хранения копируемых файлов. Наиболее простой способ состоит в размещении на неактивной панели экрана того каталога, куда направляются копии выбранных файлов.

Теперь следует нажать клавишу **<F5>**, на экране будут отображены имя и путь размещения первого из группы копируемых файлов. При необходимости следует внести коррективы и нажать клавишу **<Enter>**.

На следующем шаге осуществляется проверка наличия файла с подобным именем в новом каталоге. Если такой файл обнаруживается, то выдается сообщение о необходимости подтверждения возможности копирования этого файла. Здесь необходимо установить курсор на одну из команд: **Overwrite** — для продолжения копирования; **All** — продолжения копирования и других файлов группы без предупреждения; **Skip** — не копировать отобранный файл и перейти к другому.

<F6> — Move — переименование файла или каталога. Эта функция позволяет переименовывать файлы и каталоги, а также переносить их в другие каталоги. Методика выполнения этой команды аналогична методике копирования файлов.

<F7> — Mkdir — создание подкаталога. Новый каталог создается в текущем каталоге, если не указать полное имя каталога.

<F8> — Delete — удаление файла или каталога. Команда позволяет

удалить как отдельные файлы и пустые каталоги, так и их группы, если они предварительно помечены. Если файл имеет атрибут «только для чтения», то перед его удалением необходимо дать подтверждение для выполнения команды.

<F9> — Pulldn — вывод управляющего меню Norton Commander. Помощью меню Norton Commander можно установить наиболее удобную форму размещения информации на экране, изменить назначение отдельных клавиш и др.

После нажатия клавиши <F9> на верхней части экрана устанавливается горизонтальное меню. Один из пунктов является выделенным. Для выбора меню следует использовать горизонтальные стрелки клавиатуры и клавишу <Enter>, а затем вертикальные стрелки.

После вызова меню в верхней строке экрана загорается строка со следующими полями: **Left, Files, Disk, Commands, View** и **Right**. Выделение полей осуществляется перемещением курсора на соответствующее поле.

После нажатия клавиши <Enter> открывается подменю соответствующего поля.

Левая (left) и правая (right) панели имеют одинаковые поля:

Brief — выводится краткая информация о файлах (только имя файла);

Full — выводится полная информация о файлах;

Info — выводится общая информация о каталоге и диске на другой панели;

Tree — выводится дерево каталогов диска;

On/Off — выводится/ не выводится на экран левая панель.

Режимы сортировки файлов:

Name — по имени;

Extension — по расширению;

Time — по убыванию даты последней модернизации;

Size — по размеру в порядке убывания;

Unsorted — файлы выводятся по порядку записи в каталоге.

Re-read — повторно прочесть оглавление каталога.

<F10> — **Quit** — выход из Norton Commander.

Полезно запомнить следующие действия, закрепленные за клавишами при работе с NC:

<Ctrl> + <E> — вывод в командную строку предыдущей выполняемой команды;

<Ctrl> + <X> — вывод в командную строку команды, которая была введена после команды, находящейся в командной строке;

<Ctrl> + <Enter> — вывод в командную строку выделенного файла;

<Esc> — очистка командной строки;

<Ctrl> + <O> — убрать панели с экрана или вынести панели на экран;

<Ctrl> + <P> — убрать одну из панелей с экрана или вынести панель на экран;

<Ctrl> + <U> — поменять панели местами;

<Ctrl> + <F1> — убрать левую панель с экрана или вынести ее на экран;

<Ctrl> + <F2> — убрать правую панель с экрана или вынести ее на экран;

<Shift> + <F3> — просмотр запрашиваемого файла;

<Shift> + <F4> — редактирование файла;

<Shift> + <F5> — копирование файла;

<Shift> + <F6> — переименование или перемещение файла(ов).

Практикум: работа в среде MS DOS

Цель работы: отработать навыки работы в среде MS DOS.

1. Включите компьютер и перейдите на работу в режиме DOS. Нажмите на клавишу <F9>, затем <Y>.
2. Перейдите на диск **D:** (если его нет, то следует остаться на диске **C:**).
3. Просмотрите оглавление диска **D:** (этот диск, как правило, открыт для пользователя в компьютерных классах).
4. На рабочем диске создайте каталог с Вашим именем, применив латинский алфавит, и войдите в него. Используйте, например, имя Student.
5. Создайте в Вашем каталоге файл report.txt и запишите в него тему и цель лабораторной работы.
6. Еще раз, не выходя из своего каталога (Student), просмотрите содержание корневой директории диска **D:** и добавьте эту информацию в файл report.txt.
7. Создайте в Вашем каталоге два подкаталога.
8. Скопируйте с диска **D:\Student** файл report.txt в один из подкаталогов.
9. Измените имя одного из каталогов.
10. Просмотрите оглавление Вашего каталога (Student) и допишите его в файл report.txt.
11. Переименуйте файл в подкаталоге, присвоив ему новое имя с Расширением .bac.

12. Просмотрите файл report.txt.
13. Удалите все подкаталоги из Вашей директории.
14. Удалите все файлы из Вашего каталога.
15. Войдите в корневую директорию и удалите каталог с Вашим именем.
16. Установите на дисковод дискету и отформатируйте ее.
17. Скопируйте подготовленный файл на диск A:.
18. Прочитайте корневой каталог диска A:.
19. Перейдите на диск C:.
20. Прочитайте каталог диска C: и найдите каталог **Windows**.
21. Запустите графическую операционную систему **Windows XP**.
22. Закончите выполнение практической работы и вытащите дискету из дисковода.

ГЛАВА 2. ГРАФИЧЕСКАЯ ОПЕРАЦИОННАЯ СИСТЕМА WINDOWS XP



Среди современных графических операционных сред, выделяется семейство сред Windows, на базе которого реализуется большое число различных документных технологий.

Разработка и активное распространение графической операционной системы семейства Windows обусловлены необходимостью предметной организации применения компьютера.

При этом компьютер преобразовывается в документную коммуникационную систему, которая имеет дело с набором (множеством) документов. С этой целью непосредственно «экранный интерфейс» превращается в понятие «рабочий стол», который и реализуется с помощью соответствующей графической операционной системы, установленной на компьютере.

Выход на рынок графической оболочки Windows 3.0 следует отнести к началу 1990 г. В то время уже появились цветные дисплеи, значительно увеличилась разрядность и тактовая частота микропроцессоров. Объем винчестера достиг 40 Мбайт. В результате, на базе MS DOS 4.0 и появился первый представитель семейства Windows. В 1992 г. вышла новая версия графической операционной среды (оболочки) — Windows 3.1, которая просуществовала до середины 1990-х гг. К этому времени произошел значительный прогресс в технологии изготовления цветных экранов, а также в области микроэлектроники. Это привело к появлению компьютеров на базе 486 микропроцессоров. Одновременно развивалось системное ПО, и на рынке прочно обосновался один из последних вариантов DOS — MS DOS 6.0. Эти предпосылки позволили реализовать более эффективную операционную среду, которая принципиально отличалась от своей предшественницы. Изменился не только дизайн и методы навигации, значительно расширился круг операций, которые стали доступны пользователю, что значительно снизило требования к нему, связанные с умением работать в среде DOS. Но главное — это изменение метода работы с компьютером, связанное с переходом с управления файлами, устройствами и каталогами к управлению документами и устройствами с помощью их размещения на рабочем столе операционной системы. Через некоторое время выходит очередная редакция Windows 95, которая имела встроенный браузер Internet Explorer. Затем появляется расширение системы для активной деятельности пользователя в сетевых средах: Windows NT. Дальнейшее развитие этого направления шло как по линии разработки собственных элементов, присущих операционным системам, так и по линии усиления интеграции с Интернетом. В 1998 г. был осуществлен переход на Windows 98, который обеспечивал бесшовную связь с браузером Internet Explorer, а в 1999 г. был представлен новый член семейства — графическая операционная система Windows 2000, — отличавшийся активизацией функций мультимедийных устройств и расширением возможностей работы в Интернет-пространстве.

Параллельно с развитием вычислительной техники шли активные преобразования в средствах записи, передачи и воспроизведения информации. Расширились возможности устройств чтения и записи на оптические диски (CD), значительно увеличилась пропускная способность Интернета за счет оптоволоконных линий связи, появились средства распознавания и воспроизведения звука и т. д. Компьютерная технология стала тем элементом, который смог соединить все новые технологии для развития индивидуальных и профессиональных качеств каждой личности.

Дальнейшее развитие линии Windows привело в 2001 г. к выходу на рынок графической операционной системы Windows XP, которая создала техническую основу для развития новых возможностей в межличностном общении, предоставила компьютерную среду для работы в мобильном режиме, а также ввела в состав своей технологии работу с цифровыми фотографиями, усилившую защитные свойства программной системы при работе в Интернете. Новое поколение Windows раз-

делилось на две ветви: Windows XP Professional и Windows XP Home Edition, на базе последней из которых реализуется идея «народного компьютера».

Изменения в технологической оснащенности, которые произошли в семействе Windows, повлияли и на его дизайн. Он приобрел ярко выраженные графические черты. Однако в своих настройках Windows сохранил возможность использования традиционного оформления своих инструментов и приложений — режим **Классическая**.

Немного забегаая вперед, укажем, что оформление рабочего стола Windows XP определяется темой, которая выбирается в поле нового листа Тема вкладки настройки рабочего стола **Свойства: Экран** (рис. 2.1).

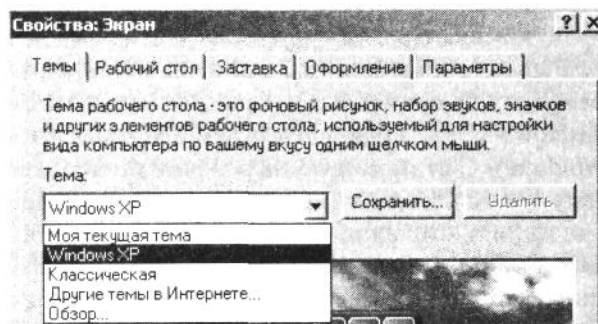


Рисунок 2.1. Лист Темы вкладки **Свойства: Экран**

Тема рабочего стола — это фоновый рисунок, набор значков и звуков, организации области задач и рабочих окон и других элементов рабочего стола, позволяющий наиболее эффективно использовать мощность компьютера. В системе различают следующие темы: моя текущая тема, Windows XP, классическая, другие темы в Интернете, выбор темы с помощью поиска подходящей.

Учитывая, что распространенность семейства Windows 9x пока значительно превосходит применение его новых, весьма современных отпрысков (типа Windows XP), сначала мы рассмотрим возможность применения темы **Windows XP**, а затем будем применять «Классическую тему» и используем эту возможность настройки на стандартное оформление рабочего стола.

2.1. Рабочий стол операционной с и с т е м ы

Рабочий стол, реализуемый с помощью графической операционной системы Windows XP, позволяет отображать потоки различной информации на экране. При этом он:

- ◆ обеспечивает оперативный переход от одного документа к другому, постоянно возвращаясь к одному или нескольким из них;
- ◆ реализует возможность подготовки и редактирования новых документов;
- ◆ подключает документную систему к каналам связи;
- ◆ создает условия для работы с удаленными документами и базами данных, т. е. использует различные формы информационной деятельности, не выходя (в физическом смысле) за пределы своего компьютера.

Решение подобных задач предполагает применение многопрограммных графических операционных систем, какой и является Windows XP (в дальнейшем будем использовать обобщающий термин Windows). С ее помощью на рабочем столе (экране) размещается тот документ, с которым ведется работа в настоящее время. Другие документы либо отодвигаются на второй план, либо замещаются различными метками (значками, закладками, пометками и тому подобными средствами), оставляя на экране информацию о ранее использованных документах и сохраняя оперативный доступ к ним (для их использования: чтения, преобразования, перемещения в памяти компьютера, передачи по каналам связи и т. п.).

В основе управления MS Windows лежит концепция аудиовизуального интерфейса. Эта среда имеет следующие преимущества:

- ◆ настраивается под конфигурацию конкретного компьютера и максимально использует его возможности;

- ◆ использует графические управляющие элементы, располагаемые пользователем на экране произвольным образом;
- ◆ реализует возможность одновременного выполнения нескольких программ, переноса между ними информации и перехода от одной из его программ к другой.

Изменение методов организации работы на компьютере привело к качественному изменению режима работы с документами. Теперь он осуществляет несколько отличительных от прежних операционных сред возможностей: прежде всего, это реализация многозадачного режима работы и многопоточной обработки данных.

Многозадачный режим определяет возможность одновременного выполнения нескольких заданий, т. е. запуска на выполнение нескольких программ, одновременное обращение к нескольким активизированным программам. Например, на рабочем столе одновременно могут быть активизированы: текстовый, табличный, графический редактор и редактор презентаций.

Многопоточная обработка данных означает разбивку задания на более мелкие задания, определяемые потоками. Например, поток управления вводом данных с клавиатуры, поток управления выводом информации на печать и т. п., что позволяет существенно ускорить процесс выполнения программы. Максимальное ускорение достигается при использовании многопроцессорных систем, в которых каждый поток программы выполняется на отдельном процессоре.

Другие две новые возможности обозначаются как WYSIWYG (What You See Is What You Get) — что видишь, то и получаешь, и Plug and Play -подключай и работай. Эти возможности позволяют видеть итоги своего труда при подготовке документов в процессе работы над ним (в том же виде и в тех же пропорциях, какие будут получены при их печати на бумагу) и свободного подбора и использования разнообразного оборудования как для получения твердых копий, так и для приема и передачи информации.

Семейство Windows использует только те программы, которые написаны специально для этой системы, следуя определенным соглашениям, благодаря чему они становятся универсальными: обладают единым пользовательским интерфейсом (т. е. используют однотипное размещение информации на экране компьютера, применение одинаковых клавиш для управления и др.), имеют общий механизм обмена информацией любого типа между частями одной программы или между различными программами.

Программа, загружаемая в Windows, называется *приложением* (Application). Программы, созданные для работы в Windows, называются *Windows-приложениями*.

При изучении методов организации рабочего стола специалиста в графической среде Windows необходимо овладеть следующими навыками:

- ◆ организация рабочего стола на экране компьютера;
- ◆ организация движения электронных документов;
- ◆ создание новых и размещение имеющихся документов и папок с документами на рабочем столе.

2.1.1. Основные элементы рабочего стола

На рис. 2.2 показан общий вид рабочего стола. Цвет поверхности стола, как и его фактуру, можно задать исходя из эстетических ориентиров с помощью контекстного меню.



1 — значки и ярлычки документов; 2 — область задач; 3 — переход к меню кнопки **Пуск**; 4 — кнопки окончания работы с системой; 5 — кнопка **Пуск**; 6 — область задач администратора рабочего стола; 7 — область системных задач; 8 — область задач пользователя; 9 — панель отложенных задач; 10 — графическое оформление рабочего стола; 11 — переключатель клавиатуры, индикатор текущей даты и времени

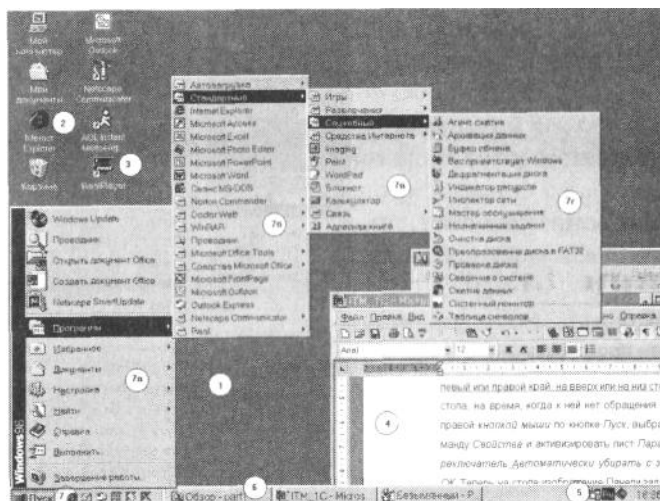
Рисунок 2.2. Тема **Windows XP** для оформления рабочего стола

В первоначальной установке в левой части стола расположены значки различных объектов рабочего стола. Они могут обозначать папки с документами, программные приложения и другие объекты. В процессе работы значки можно перемещать по столу, создавая оптимальные условия для плодотворной работы.

На рабочем столе находится обязательный элемент **Корзина**, куда переносились ненужные документы и программы, но которые в любой момент можно было бы извлечь на рабочий стол.

Область задач рабочего стола содержит поле имени администратора рабочего стола, поле вызова программ для работы в Интернете, поле используемых приложений, команду вызова меню программ рабочего стола **Все программы**, команды выхода из системы и завершения сеанса работы, панель инструментов рабочего стола, панель отложенных задач и область размещения переключателя клавиатуры, индикатора даты и времени, а также размещения других значков, обозначающих постоянно функционирующие приложения.

Для сравнения покажем оформление рабочего стола предыдущих версий Windows, которые обозначим как Windows 9x (рис. 2.3).



1 — рабочая поверхность стола; 2 — место размещения значков основных объектов; 3 — ярлык приложения; 4 — окно приложения; 5 — индикатор шрифта клавиатуры и таймер; 6 — панель задач; 7 — браузеры системы и кнопка **Пуск**, раскрывающая меню компьютерной системы: 7а — панель меню первого уровня, 7б — панель меню второго уровня, 7в — панель меню третьего уровня, 7г — панель меню четвертого уровня

Рисунок 2.3. Общий вид рабочего стола семейства **Windows 9x**

Здесь на рабочем столе система определяет ряд обязательных объектов, к которым относят: **Мой компьютер**, **Мои документы**, **Корзина** и **Портфель**. Остальные объекты, папки и ярлыки объектов являются необязательными и устанавливаются пользователем рабочего стола.

Для перехода к командам меню Пуск в Windows 9x следует сначала щелкнуть по кнопке **Пуск**, а затем по строке меню **Программы**. На рабочий стол вызывается меню второго уровня.

Переход на меню программ в среде Windows XP осуществляется после щелчка по строке **Все программы**, расположенной на поле области задач рабочего стола.

В этом случае в области задач появляется традиционное меню второго уровня кнопки **Пуск**, показанное на рис. 2.4.

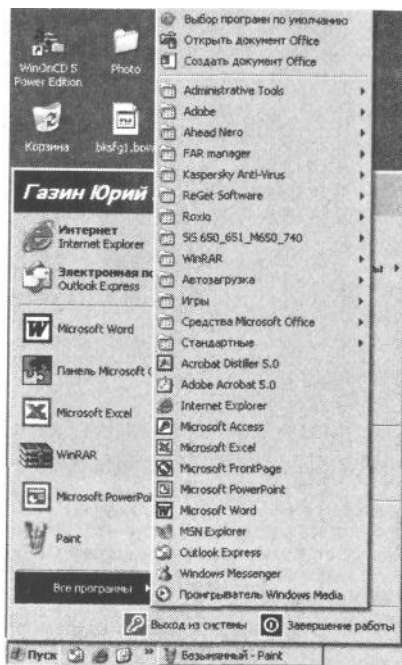


Рисунок 2.4. Меню строки Все программы

На рабочих столах Windows 9x и Windows XP используются как однотипные элементы, так и элементы, оформление и содержание которых различно. Сначала рассмотрим функциональное назначение элементов, используемых при оформлении обоих рабочих столов, показывая оба варианта обозначения.



Папки являются условным обозначением каталога в файловой системе компьютера. Папка может содержать другие папки, тогда последние определяются как вложенные папки.



В папках могут размещаться также программы, документы в форме файлов, программы обслуживания отдельных устройств системы (принтеры, диски и т. п., которые определяются как объекты и обозначаются соответствующим значком).



Для обозначения рабочих документов или приложений на рабочем столе используют ярлык. В его левом нижнем углу помещена отличительная стрелка. Файлы документов, приложения, показанные на рабочем столе, находятся в соответствующих каталогах. Их связь с рабочим столом осуществляется с помощью специального файла связи, описывающего документ. Этот файл связи и определяется как ярлык.



Ярлык объекта определяет путь к объекту, он его обозначает. Ярлык позволяет вызвать приложение, которое находится в соответствующем каталоге, непосредственно с рабочего стола системы. Он имеет свое название, расположенное внизу ярлыка. Все ярлыки и значки на рабочем столе активизируются двойным щелчком мыши.

Если ярлык связан с программным приложением, то он осуществляет его запуск, если папку или другой объект, то открывается соответствующее окно.

Ярлык объекта можно создать, щелкнув правой кнопкой мыши по объекту и выбрав пункт контекстного меню **Создать ярлык** или щелкнув правой кнопкой мыши на панели или в папке. На рабочем столе появится контекстное меню, на котором следует выбрать опцию **Создать**, а затем **Ярлык**.

На любом рабочем столе присутствуют специальные папки — Мой компьютер, Internet Explorer, Корзина и Мои документы.



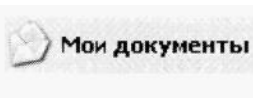
Папка Мой компьютер содержит значки всех накопителей на дисках, включая винчестер, гибкие диски, сетевой диск и диск CD-ROM. Здесь также располагаются папки: Панель управления и Принтеры.



Папка Internet Explorer обеспечивает быстрый доступ к ресурсам Интернета, включая WWW и электронную почту.

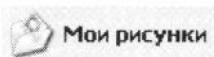


В папку Корзина попадают файлы, удаленные средствами Windows, при этом сам процесс удаления реализуется перетаскиванием значка объекта к значку Корзина. Удаленные в Корзину документы можно снова вернуть на рабочий стол.



Папка Мои документы обычно используется для размещения писем, отчетов, заметок и других документов. Эта папка может быть использована для оперативного перемещения подготовленных документов на рабочий стол.

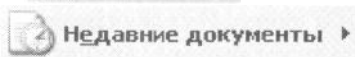
В этой папке обычно размещают часто используемые документы, уменьшая тем самым число значков и ярлыков на экране. На рабочем столе можно разместить значки различных объектов, перетаскивая их из окон различных папок или создавая новые ярлыки для обозначения документов или других объектов.



В Windows XP в области задач рабочего стола выделены папки Мои рисунки, Моя музыка, что позволяет администратору акцентировать свое внимание на документах с различными формами представления информации. Папка **Мои рисунки** предназначена для размещения цифровых фотографий, изображений, графических файлов. В папке **Моя музыка** предполагается размещение музыкальных и звуковых файлов.



В системах Windows 9x в поле команд меню **Пуск** используется команда



Документы, которая содержит список ранее открывавшихся документов.

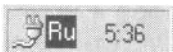
В системе Windows XP эта команда расширена и, разместившись в поле области задач рабочего стола, получила обозначение **Недавние документы**.

Если используется переносной компьютер (ноутбук), то на левом краю стола устанавливают значок программы **Max Time** — энергетического состояния батареи (аккумулятора).

На нижней части рабочего стола размещена панель задач, на которой помещаются клавиши — значки незавершенных объектов (работа которых была прервана хозяином рабочего стола). Щелчок мышью по этим клавишам позволяет активизировать приложение и немедленно продолжить работу прерванного объекта.



В левом нижнем углу установлена кнопка **Пуск**, с помощью которой запускается меню объектов системы (рис. 2.3, 7а—7г).

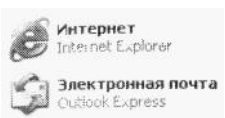


В правом конце панели задач установлена кнопка переключения алфавита клавиатуры (английский или русский) и указатель текущего времени (таймер). Для изменения алфавита следует щелкнуть мышью по кнопке переключателя и на появившемся окне выбрать строку **Русский или Английский** и еще раз щелкнуть мышью.

Панель задач Windows 9x можно перенести на любой край рабочего стола с помощью мыши: на левый или правый край, верх или низ стола. Панель задач можно убрать со стола на время, когда к ней нет обращения. Для этого следует щелкнуть правой кнопкой мыши по панели задач, выбрать в открывшемся меню команду **Свойства** и активизировать лист **Параметры** панели задач, затем включить переключатель **Автоматически убирать с экрана** и щелкнуть по кнопке ОК. Теперь на столе изображение панели задач отсутствует. Оно появится лишь в тот момент, когда курсор мыши пересечет соответствующий край стола.



Панель задач имеет отдельное поле, на которое устанавливаются программы-браузеры, позволяющие работать с веб-документами, и программы электронной почты, в данном случае это Internet Explorer и Netscape Communicator.



Аналогичные программы устанавливаются в верхней части области задач рабочего стола — это обозреватель Интернета и почтовая программа для работы с корреспонденцией Интернета.

Меню объектов системы представляет собой многоуровневое меню.

Среди основных команд панели меню первого уровня выделяются:

Программы — запускает второй уровень панели меню, обеспечивающий доступ к стандартным программам, включая программу Сеанс MS-DOS и папку **Стандартные**, а также к диспетчеру файлов **Проводник**;

Избранное — запускает панель меню второго уровня, на которой находятся папки **Каналы, Медиа, Ссылки, Мои документы**, а также список адресов **События Веба** и др., связанные с работой в веб-пространстве;

Документы — обеспечивает доступ к документам и папкам, к которым были последние обращения, включая и текущий сеанс;

Настройка — вызывает панель меню второго уровня, поддерживающую доступ к **Панели управления**, к папке **Принтеры** и к листам свойств **Панели задач**;

Поиск — осуществляет поиск файла или папки на компьютере или в сети, а также поиск необходимого компьютера в сети.

Справка — открывает справочную систему Windows;

Выполнить — обеспечивает быстрый запуск программ с помощью записи командной строки или поиск нужной программы.

В области задач Windows XP размещена команда **Выбор программ по умолчанию**. Эта команда позволяет установить доступ к соответствующим приложениям из главного меню рабочего стола и из других мест.

В нижней части панели меню первого уровня размещены команды **Завершение работы** в графической операционной системе и **Остановка работы** (для переносных компьютеров) в целях перехода на экономный режим работы. В верхней части панели меню располагается обращение к группе MS Office: **Открыть документ и Создать документ**. Здесь могут находиться обозначения наиболее важных для пользователя приложений, например **Проводник**.

Файлы в компьютерной системе размещаются в папках или в корневом каталоге. Папки могут располагаться как в других папках или в корневом каталоге, так и на рабочем столе. Для размещения новой папки на рабочем столе можно воспользоваться контекстным меню, которое вызывается с помощью щелчка правой кнопки мыши.

В появившемся контекстном меню, показанном на рис. 2.5 (на примере Windows 9x), следует перейти на строку **Создать** и щелкнуть мышью по позиции **Папка**. На поле рабочего стола появится заштрихованная область обозначения новой папки и поле под ней для записи имени новой папки. Остается записать имя новой папки (например, Новая) и щелкнуть мышью по полю окна или рабочего стола. То же следует сделать при изменении имени папки.

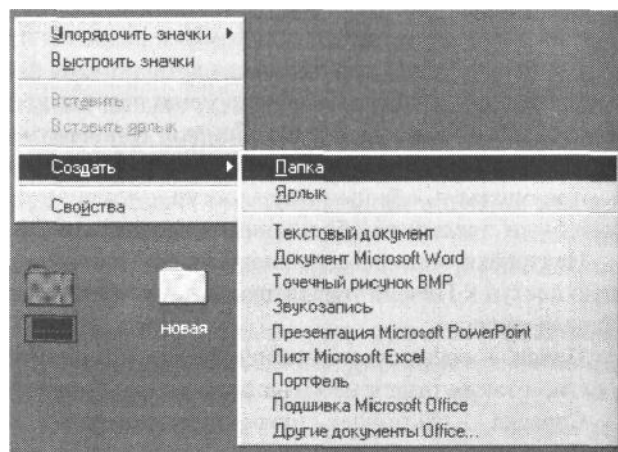
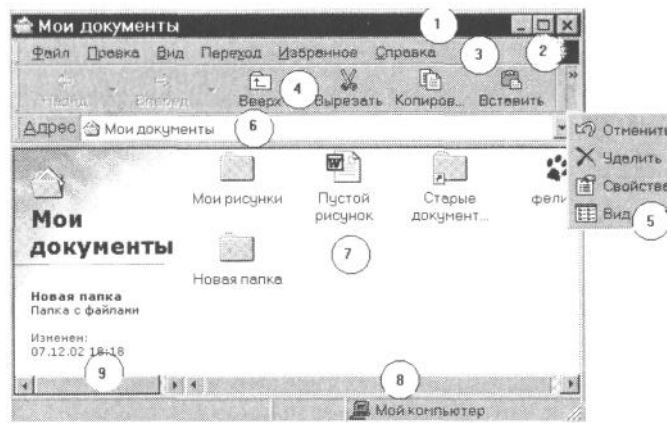


Рисунок 2.5. Применение контекстного меню для установки новой папки

Открыть папку можно двойным щелчком мыши по полю папки. На экране будет показано окно, содержащее строку управления положением и размерами папки на рабочем столе, органы управления папкой и рабочее поле папки (рис. 2.6).

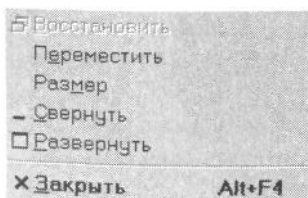


- 1 — область заголовка папки; 2 — кнопки управления размерами папки;
 3 — системное меню управления папкой; 4 — кнопочная панель управления папкой;
 5 — расширение кнопочной панели управления; 6 — адресная строка;
 7 — рабочее поле папки; 8 — ползунок линейки;
 9 — ползунок информационного поля папки

Рисунок 2.6. Окно папки **Мои документы** Windows 9x

Окно — прямоугольная область на экране дисплея, используемая для размещения программных групп и программных элементов, ввода или вывода информации и независимая от другой части экрана. Окно для приложения имитирует полный экран. Окно открывается при вызове приложения и закрывается по завершении работы с ним. Окно разворачивается при его максимизации, при этом оно занимает весь экран, и «свертывается» при его минимизации, при этом оно обозначается значком (пиктограммой).

На рис. 2.6 показано окно папки **Мои документы**, которое появляется после щелчка мышью по соответствующей папке, размещенной на рабочем столе. Рассмотрим подробнее основные компоненты этого окна.



В левом углу строки (области) заголовка папки помещается значок папки. Если щелкнуть по нему правой кнопкой мыши, то появится контекстное меню, с помощью которого можно выполнить следующие операции: **Восстановить**, **Переместить**, **Размер**, **Свернуть**, **Развернуть**, **Закрыть**. Последние три команды повторяются в правом краю строки заголовка в виде трех кнопок.

Левая кнопка свертывает окно и помещает его обозначение на панель команд, размещенную обычно в нижней части экрана. В этом случае работа с окном «откладывается» на некоторое время, но приложение не закрывается.

Средняя кнопка разворачивает окно, заполняя всю поверхность рабочего стола, закрывая остальные приложения полем активного окна. При повторном щелчке мышью по этой кнопке размеры окна уменьшаются. В этом случае, захватывая курсором мыши стороны обрамления окна или его углы, можно изменить размеры окна. Если окно максимизировано (развернуто), то эта функция отключается.

Правая кнопка закрывает активное окно.

Середина области заголовка позволяет перемещать окно папки по полю экрана. Для этого необходимо установить курсор мыши на область заголовка и, ведя мышью по полю экрана, передвинуть окно в нужное место рабочего стола.

Окно содержит опции (команды), которые позволяют управлять как окном, так и его содержимым (папками и файлами). Эти опции сгруппированы в отдельные списки, образующие операционное (системное) меню окна (см. рис. 2.6, 3): **Файл**, **Правка**, **Вид**, **Переход**, **Избранное** и **Справка**. Для знакомства с их содержанием следует самостоятельно изучить их, раскрыв предварительно созданное новое окно.

Так, для формирования новой вложенной папки следует раскрыть меню **Файл**, выбрать опцию **Создать** и перейти на правый список, щелкнув по позиции **Папку** (рис. 2.7).

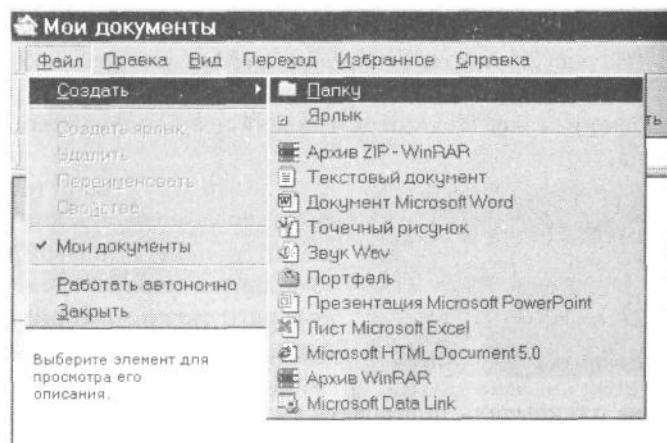


Рисунок 2.7. Операционное меню папки

Под строкой операционного меню размещена кнопочная панель управления окном (см. рис. 2.6, 4). Здесь в форме кнопок размещены отдельные, наиболее часто применяемые команды управления. Среди них: **Назад, Вперед, Вверх, Вырезать, Копировать, Вставить, Отменить,**

Удалить, Свойства, Вид. Если кнопки не помещаются на поле окна, то они заменяются двойной стрелкой, при щелчке по которой они раскрываются в виде дополнительной кнопочной панели, как показано на рис. 2.6, 5).

Кнопочная панель управления окном Windows XP имеет иное графическое содержание, но функционально почти не отличается от стандартной панели:



Кнопки **Назад** и **Вперед** позволяют перейти соответственно к ранее выполненной или к последующей команде. Кнопка **Вверх** дает возможность перейти на более высокий уровень иерархии в файловой системе. Эти кнопки позволяют осуществить навигацию по схеме, принятой в браузерах Интернета. Работу остальных кнопок несложно проверить самостоятельно.

Для перемещения объектов, размещенных на рабочем поле папки, следует щелкнуть правой кнопкой мыши по выделенному объекту и выбрать необходимую операцию из контекстного меню, например **Копировать, Удалить, Переместить** и т. д.

В тех случаях, когда число объектов, размещенных в папке, велико и они не помещаются на рабочем поле папки, то в окне устанавливается линейка прокрутки, снабженная ползунком (см. рис. 2.3, 8). С помощью линейки можно передвигать рабочее окно так, чтобы можно было рассмотреть необходимые объекты, размещенные в папке. Для передвижения рабочего окна следует захватить ползунок мышью и перетаскивать его по длине линейки до тех пор, пока не будет найден необходимый объект или не закончится сама папка. Для ускорения передвижения можно щелкать мышью по пространству между концами линейки и ползунком.

В поле окна папки выделяется его информационная часть (см. рис. 2.6, 9), которая содержит информацию о выделенном в папке объекте или объектах и другие сведения.

Оформление окна папки в Windows XP имеет некоторые отличия. Они хорошо видны на примере оформления окна папки **Мой компьютер** (рис. 2.8). Изменен дизайн кнопок управления окном, размещенных на строке имени папки. Используемое кнопочное меню аналогично меню Интернет-обозревателя. В поле окна встроена адресная строка, позволяющая осуществить непосредственный переход к документам Интернет-пространства и др.

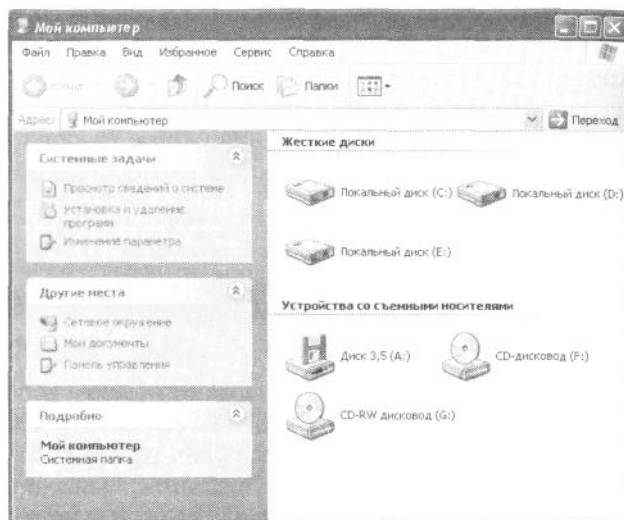


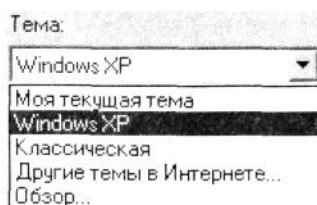
Рисунок 2.8. Окно папки **Мой компьютер** Windows XP

В левой части рабочей области окна помещена область задач окна приложения, на которой расположены те основные команды, которые могут использоваться при работе с информацией и приложениями текущего окна. Непосредственно в рабочей области окна в виде графических обозначений показаны основные объекты приложения.

2.1.2. Оформление рабочего стола

Для управления внешним видом поверхности рабочего стола следует вызвать вкладку **Свойства: экран**. Для этого следует щелкнуть правой кнопкой мыши по рабочему столу. В появившемся контекстном меню выбрать строку **Рабочий стол Active Desktop** и перейти на следующую панель меню **Настроить рабочий стол**.

Вызов вкладки можно осуществить и с помощью программы **Экран**, значок которой размешен на **Панели управления**. При двойном щелчке по значку **Экран** на рабочий стол будет вызвана вкладка **Свойства: Экран** (рис. 2.9, 2.10).



Свойства экрана описываются на шести листах вкладки для Windows 9x и на пяти листах — для Windows XP. Особое значение имеет первый лист вкладки **Темы** Windows XP, показанный на рис. 2.9. Предлагаемые темы оформления составляют список из пяти позиций, представленных в виде раскрывающегося списка **Тема**. Можно сформировать свою тему оформления и записать ее, используя кнопку **Сохранить**.

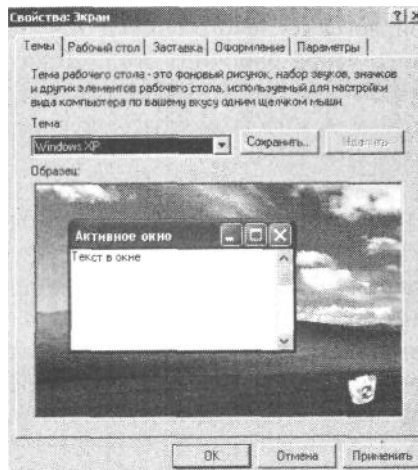


Рисунок 2.9. Лист Темы вкладки Свойства: Экран оформления рабочего стола Windows XP

Для оформления рабочего стола в Windows 9x используют вкладку, показанную на рис. 2.10.

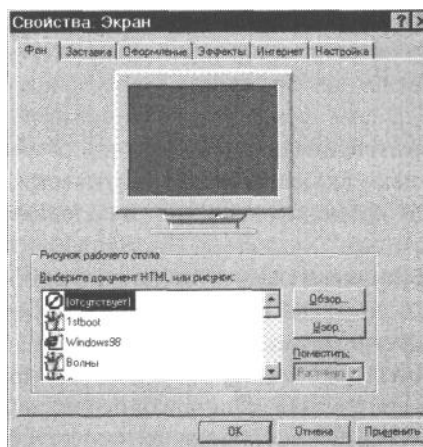


Рисунок 2.10. Лист Фон вкладки Свойства: Экран оформления рабочего стола Windows 9x

Вкладка содержит следующие листы: Фон, Заставка, Оформление, Эффекты, Интернет, Настройка.

Лист свойств **Фон** позволяет указать фоновый узор, который изображается на заднем плане рабочего стола (узор обивки стола). Узор является двуцветным изображением, размером 8x8 точек, которым покрывается весь экран. Первый цвет — черный, второй устанавливается на листе свойств **Оформление**. Каждый фоновый узор имеет свое имя, которое выбирается из левого списка окна свойств. Образец выбранного фонового узора рабочего стола показывается в центре листа свойств в виде экрана, заполненного узором.

На листе Фон можно разместить рисунок, который выбирается из списка стандартных графических файлов Windows. Рисунок является растровым изображением, которое Windows помещает на фоне экрана поверх узора. Рисунок может покрыть весь экран или занимать только его центр.

Для использования специально подготовленного рисунка, например с помощью графического редактора **Paint**, который размещен в папке **Стандартные** меню **Программы**, рисунок необходимо сформировать в виде файла с расширением .bmp, .dip или .rle и выбрать его, используя кнопку **Обзор**, расположенную под списком рисунков. Файлы с расширением .bmp и .dip будут показаны сразу, а файлы с расширением .rle (сжатый растровый файл) — после включения ;*.RLE к маскам имен файлов в поле ввода **Имя файла**.

Лист свойств **Заставка** позволяет включать динамическое изображение на экране при длительном простаивании компьютера во включенном состоянии. Первоначально это свойство предполагало содействовать предотвращению износа экрана, что было возможно ранее при статическом изображении. Современные экраны лишены такого недостатка, а сама функция динамической заставки позволяет получить дополнительное средство защиты от несанкционированного доступа. Заставка включается автоматически, после истечения заданного заранее времени ожидания, а ее выключение — при вводе знаков пароля.

На листе **Заставка** следует выбрать имя необходимого файла из списка, затем в поле ввода **Интервал** указать время ожидания (в минутах) системой начала запуска файла заставки. Переключатель **Пароль** позволяет осуществить защиту от несанкционированного доступа. Для его использования следует установить переключатель **Пароль** и щелкнуть по кнопке **Изменить**. В появившемся диалоговом окне **Изменение пароля** следует записать пароль. Если переключатель **Пароль** не включен, то работа заставки будет прервана при нажатии на кнопку мыши.

Большинство динамических заставок требует установки параметров. Они устанавливаются при нажатии на кнопку **Настройка**.

Лист свойств **Оформление** (рис. 2.11) позволяет задать цвета, шрифты и размеры различных элементов рабочего стола: цвет рабочего стола, цвета в окнах папок и программ, шрифт названий значков, размеры линеек прокрутки и др.

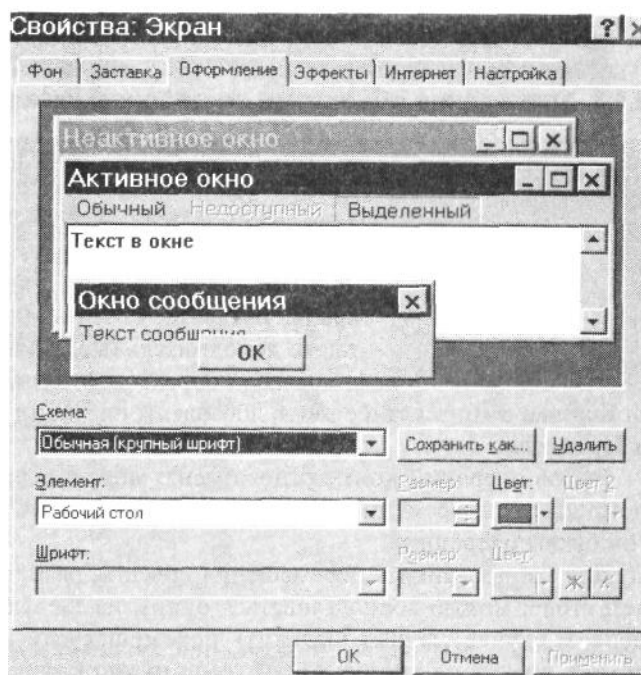


Рисунок 2.11. Лист **Оформление** вкладки **Свойства: Экран**

Windows предлагает список стандартных настроек, который размещен в поле **Схема**. Для создания оригинальной схемы настройки следует перейти к полю **Элемент** и указать его, затем показать размер этого элемента в рядом расположенном окне и, наконец, цвет фона элемента. Для этого следует щелкнуть по полю **Цвет**, изображенному в форме экрана, и щелкнуть по одной из ячеек в развернутой матрице цветов.

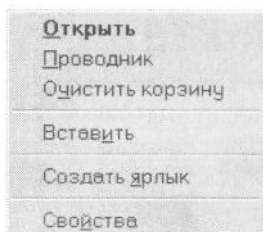
В заключение необходимо установить используемый в элементах интерфейса шрифт. Для выбора начертания (гарнитуры) шрифта следует воспользоваться стандартным списком поля **Шрифт**, установить размер шрифта, выбрать цвет текста и форму выделения: полужирный или курсив.

При установке своей схемы оформления экрана результаты преобразования интерфейса отражаются в верхней части листа. Для активизации новой настройки необходимо нажать на кнопку **Применить**, а для ее сохранения под новым именем — на кнопку **Сохранить как**.

Лист **Настройка** позволяет установить разрешение экрана и количество цветов, используемых драйвером дисплея, указать тип дисплея и другие его характеристики.

2.1.3. Управление объектами на рабочем столе

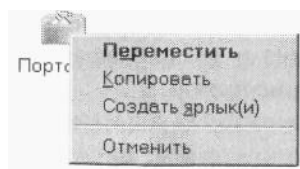
Управление объектами, размещенными на рабочем столе, можно осуществить с помощью контекстного меню, которое вызывается щелчком правой кнопкой мыши по рабочему столу. В меню имеются постоянные строки: **Открыть**, **Вставить**, **Создать ярлык** и **Свойства**, а также дополнительные, в зависимости от выполняемых объектом функций. Так, для



объекта **Корзина** в контекстное меню добавлены операции **Очистить корзину** и **Проводник**.

Вызов операций контекстного меню можно осуществить также с помощью одновременного нажатия на клавиши **<Ctrl>** и подчеркнутой буквы операции.

Для копирования и перемещения объекта, размещенного на рабочем столе, можно воспользоваться одним из следующих способов: в первом случае следует выделить перемещаемый или копируемый объект, нажать на кнопку **<Ctrl>**, если нужно копировать объект, или кнопку **<Shift>**, если его следует переместить, а затем перетащить мышью выбранный значок объекта на рабочее поле окна. В той же последовательности следует выполнить операцию по перемещению (копированию) объекта с рабочего поля окна на рабочий стол системы.



В ряде случаев операцию копирования (перемещения) следует выполнить по-другому: выделить обрабатываемый объект и перетащить его значок, используя правую кнопку мыши. При этом после перемещения значка объекта на поле окна появится его силуэт и распахнется контекстное меню, в котором и следует выбрать необходимую операцию.

2.2. Панель управления

Панель управления системой содержит программы управления отдельными устройствами компьютерной системы, которые собраны в окне **Панель управления** (рис. 2.12). Эту операцию можно также выполнить из окна **Мой компьютер**.

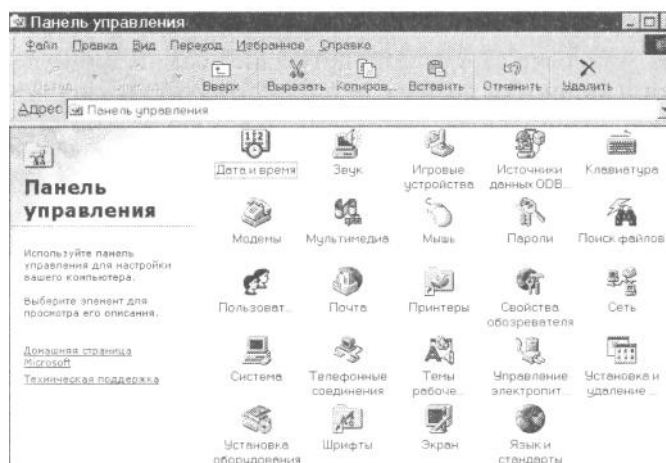


Рисунок 2.12. Окно **Панель управления** Windows 9x

Окно **Панель управления** обычно содержит следующие значки программ.

Экран. Устанавливает вид рабочего стола, его цветовую гамму, выбирает заставку для экрана в моменты длительного бездействия системы, управляет настройкой дисплея.

Дата и время. Устанавливает системную дату и время, определяет их индикацию на рабочем столе.

Звук. Устанавливает соответствие между системными событиями и заранее подобранными звуковыми сигналами.

Клавиатура. Устанавливает тип клавиатуры, частоту повтора клавиши, скорость мигания курсора, а также производит другие настройки. **Модемы.** Позволяет подключить модем к компьютеру.

Мультимедиа. Применяется для использования звуковых сигналов при разных событиях, применения различных звуковых устройств, микширования, установки звуковых компакт-дисков, определения параметров сжатия звуковых сигналов и графических изображений.

Мышь. Устанавливает функции левой и правой кнопок мыши, определяет их чувствительность, размер указателя и др.

Пароли. Управляет доступом к сети, а также из сети к компьютеру с помощью паролей.

Принтеры. Устанавливает и конфигурирует принтеры системы.

Сеть. Используется для установки сетевого имени компьютера, а также для ограничения доступа, определения сетевого адаптера, протоколов и обслуживания.

Система. Показывает информацию о ресурсах данного компьютера, установки параметров виртуальной памяти, управления устройствами и установки драйверов устройств, создает системную конфигурацию (например, для портативного компьютера).

Установка и удаление программ. Открывает листы свойств, позволяющие установить или убрать прикладную программу, удалить или добавить какие-либо компоненты Windows, а также создать загрузочный диск.

Установка оборудования. Запускает программу мастера, предназначенную для включения в конфигурацию системы новых устройств.

Язык и стандарты. Используется для установки даты, времени, числового формата, знака валюты и других национальных стандартов.

На поле окна также находятся настройки **Игровые устройства, Источники данных, Поиск файлов, Пользователи, Почта, Свойства обозревателя и др.**

Работа с папкой **Панель управления** в среде Windows XP предполагает два режима: обычный и переход к объектам по категориям. В первом случае выбор объекта управления осуществляется традиционно: щелчком мыши выбирается необходимый объект и осуществляется настройка его параметров. Рабочее окно панели управления, работающего в режиме **Обычный**, показан на рис. 2.13.

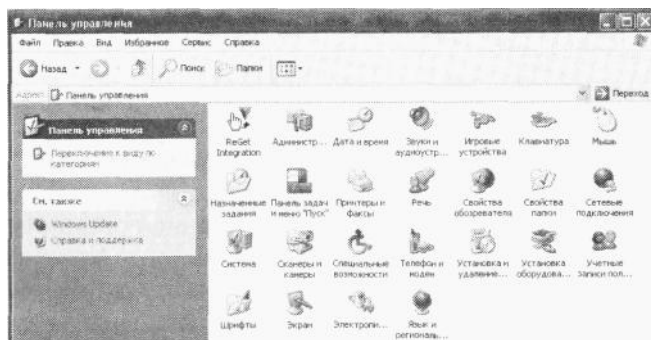


Рисунок 2.13. Окно **Панель управления** Windows XP

В области задач окна **Панели управления** размещена команда **Переключение к виду по категориям**. Выбор объекта в этом режиме имеет несколько отличий, которые связаны с его оформлением, показанным на рис. 2.14. Здесь на первом уровне выбора объекта системы осуществляется выбор группы управления, затем на втором уровне конкретизируется объект управления и в заключение на третьем уровне выбора объекта управления выполняются операции по работе с соответствующим объектом.

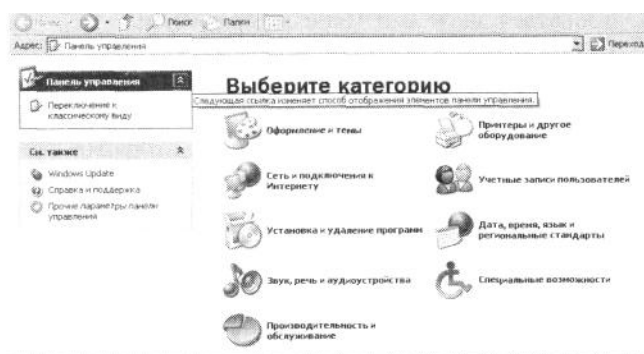


Рисунок 2.14. Окно категорий **Панели управления**

Основные объекты управления разделены на следующие группы: **Оформление и темы; Сеть и подключение к Интернету; Установка и удаление программ; Звук, речь и аудиоустройства; Производительность и обслуживание; Принтеры и другое оборудование; Учетные записи пользователей; Дата, время, язык и региональные стандарты; Специальные возможности.**

В поле окна первого уровня выделена область задач, которая содержит необходимые инструменты для перехода к выполнению возможных последующих операций.

Выбор категории объекта **Панели управления** позволяет перейти на второй уровень конкретизации объекта управления системы. Для работы на этом уровне используется окно, показанное на рис. 2.15. Для позиции **Принтеры и другое оборудование** на втором уровне устанавливается окно выбора задания.

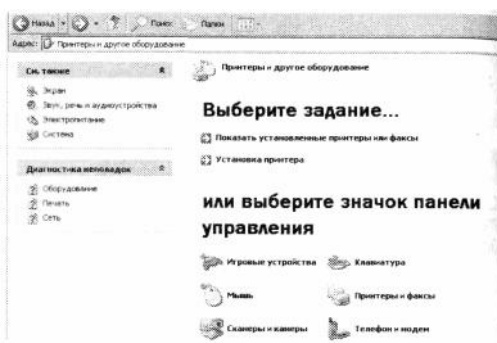


Рисунок 2.15. Окно выбора задания

На поле этого окна активизируется одна из команд: **Показать установленные принтеры или факсы** и **Установка принтера**. Можно также воспользоваться значками **Панели управления**.

Выбор значка или команды переводит диалог на третий уровень. Для демонстрации операций третьего этапа покажем содержание окна **Принтеры и факсы** (рис. 2.16). Здесь на поле области задач приведены команды установки принтера и настройки и отправки факсов, а на рабочем поле окна показаны подключенные к компьютеру принтеры. Щелчок по полю объекта вызывает соответствующее окно настройки.

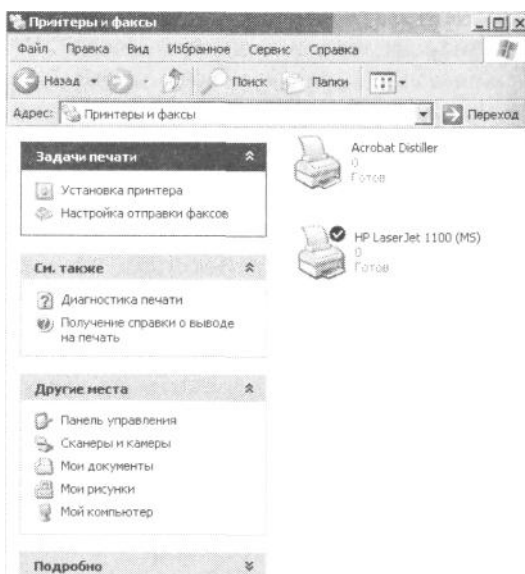


Рисунок 2.16. Окно папки Принтеры и факсы

2.3. Построение файловой системы

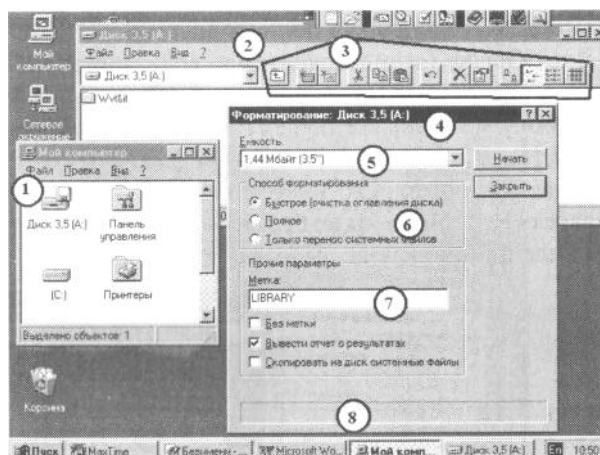
2 3.1. Форматирование диска



Форматирование гибкого диска в Windows XP можно осуществить различными методами, например с помощью программ папки **Мой компьютер**. Для реализации этого метода следует вызвать на рабочий стол папку **Мой компьютер** и на его поле выбрать дисковое устройство A:, затем вызвать контекстное меню, щелкнув по значку диска правой кнопкой мыши. Далее необходимо выбрать строку **Форматировать**. На поле рабочего стола появится соответствующий диалог.

В диалоге **Форматирование** (рис. 2.17) находятся следующие элементы: **Емкость**. В раскрывающемся списке следует выбрать емкость диска из ряда: 1,44 Мбайт, 720 Кбайт (рис. 2.17, 5).

Быстрое (очистка оглавления диска). Предполагает форматирование ранее размеченных дисков. При этом не анализируется состояние поверхности диска (рис. 2.17, 6).



1 — значок драйвера гибкого диска 3,5 (A:) в окне папки **Мой компьютер**; 2 — окно **Диск 3,5 (A:)**; 3 — панель инструментов окна **Диск 3,5 (A:)**; 4 — диалог **Форматирование**; 5 — емкость дискеты; 6 — переключатели способа форматирования; 7 — поле параметров форматирования; 8 — индикатор процесса форматирования

Рисунок 2.17. Форматирование диска

Форматирование диска **Полное**. При форматировании осуществляется контроль состояния всех секторов дисков.

Только перенос системных файлов. На системный диск переносятся обновленные системные файлы.

Метка. В поле окна заносится текст метки диска (рис. 2.17, 7).

Вывести отчет о результатах. Переключатель вызывает на рабочий стол информацию о результатах форматирования: общее число байтов, число плохих секторов и их объем, а также другую информацию.

На нижней строке диалога установлен графический индикатор процесса форматирования (рис. 2.17, 8).

Другой метод обращения к диалогу **Форматирование** осуществляется с помощью системного меню программы **Проводник**.

2.3.2. Управление папками

Эффективное использование папки осуществляется в том случае, когда правильно установлены ее параметры. Для их установки следует открыть необходимую папку и в меню команды Вид активизировать позицию **Свойства папки**. На рабочий стол будет вызвана вкладка, содержащая листы: **Общие**, **Вид** и **Типы файлов** (рис. 2.18). Каждый лист можно вызвать, щелкнув по нему мышью.

Лист **Общие** (он не появляется, если листы свойств были вызваны из окна программы **Проводник**) дает возможность управлять способом отображения на экране новых окон, определяя параметры просмотра (рис. 2.18). В поле листа **Обновление рабочего стола** установлены переключатели, с помощью которых можно настроить рабочий стол как обозреватель Интернета (окно браузера Internet Explorer), на работу в обычном виде или на основе выбранной настройки.

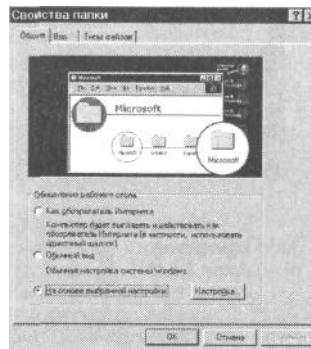


Рисунок 2.18. Лист **Общие** вкладки **Свойства папки**

Первое положение переключателя позволяет оформить окно папки рабочего окна по технологии обозревателя Интернета, что не только изменит дизайн окна папки, но и повлияет на способ активизации объектов — для их открытия будет достаточно одного щелчка мышью. Второе положение переключателя дает возможность использовать стандартный режим оформления рабочего окна папки. В третьем положении режим оформления окна папки настраивается пользователем. В этом случае следует щелкнуть по кнопке **Настройка** и установить требуемые параметры. Здесь, в частности, можно установить режим открытия каждой папки в новом окне, при этом предыдущая папка будет по-прежнему отображаться в своем окне, либо режим отображения содержимого всех открываемых папок в новом окне. Применение отдельных окон позволяет обмениваться документами, программами и другой информацией между различными окнами. Использование единственного окна вносит порядок на рабочий стол.

Лист **Вид** позволяет управлять содержанием, показываемым в папке при ее открытии. Лист состоит из двух областей: **Представление папок** и **Дополнительная настройка** (см. рис. 2.19). В верхней области листа **Представление папок** можно или использовать текущее представление папки для отображения других папок в том же виде, или сбросить установку для всех папок окна.

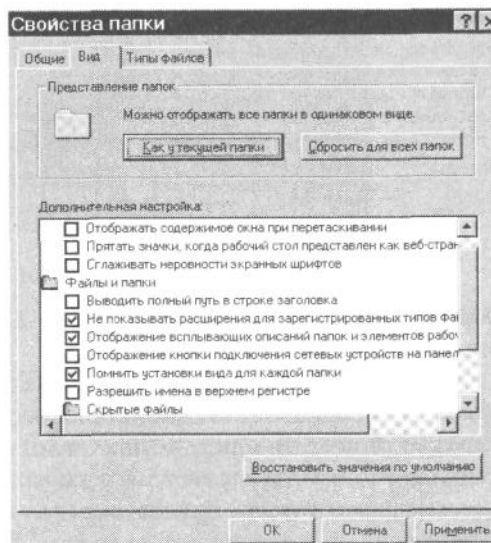


Рисунок 2.19. Лист **Вид** вкладки **Свойства папки**

В нижней области листа устанавливается дополнительная настройка с помощью установки флажков в соответствующих строках настройки. Если настройка не соответствует ожиданиям, то можно щелкнуть по кнопке **Восстановить значения по умолчанию**.

Лист **Типы файлов** (рис. 2.20) позволяет просмотреть список зарегистрированных расширений, зарегистрировать новое расширение, удалить расширение из списка или изменить параметры регистрации объекта в папке. Для этого следует воспользоваться соответствующими тремя кнопками: **Новый тип**, **Удалить** и **Изменить**. Выполнение операции предполагает поиск необходимого файла в списке, показанном в поле **Зарегистрированные типы файлов**, выделение найденного файла и переход на соответствующую операцию с помощью функциональной кнопки.

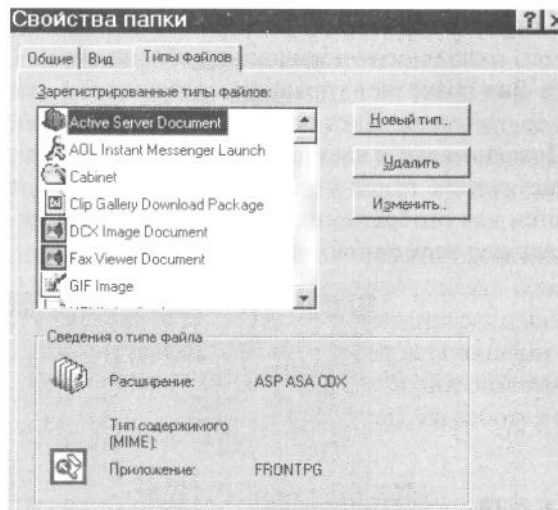
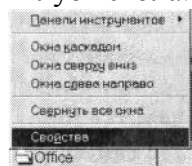


Рисунок 2.20. Лист **Типы файлов** вкладки **Свойства папки**

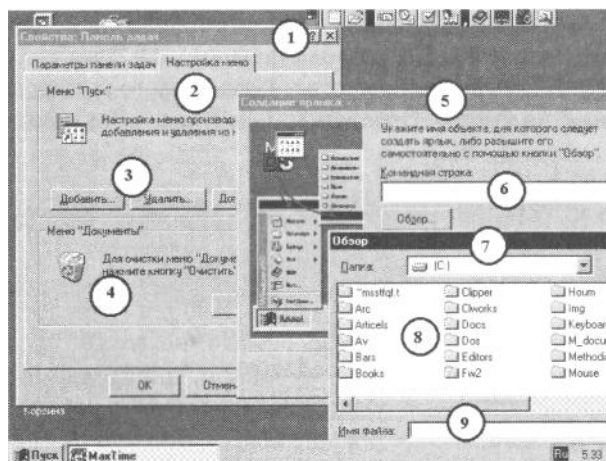
2.4. Настройка панели задач

Панель задач рабочего стола обычно устанавливается на его нижний край (см. рис. 2.3, б), однако ее можно установить на любой край стола. Для этого следует поместить курсор мыши на поле панели задач и перетащить мышью на соответствующий край рабочего стола. Можно также изменить размеры панели задач. Для этого следует захватить мышью верхний край панели задач и перетащить ее мышью вглубь стола.



Для настройки панели задач следует установить курсор мыши на ее поверхность и щелкнуть правой кнопкой. В появившемся контекстном меню следует выбрать строку **Свойства** и щелкнуть мышью.

На рабочий стол вызывается вкладка, показанная на рис. 2.21, 1). Она содержит два листа: **Параметры панели задач** и **Настройка меню**.



1 — заголовок вкладки; 2 — лист **Настройка меню**; 3 — кнопки управления **Настройка меню**; 4 — поле для очистки меню **Документы**; 5 — диалог **Создание ярлыка**; 6 — командная строка; 7 — диалог **Обзор**; 8 — поле обзора командной строки; 9 — поле имени и типа файла

Рисунок 2.21. Работа с панелью задач

Лист **Параметры панели задач** имеет четыре флажка: **Расположить поверх всех окон**, **Автоматически убирать с экрана**, **Мелкие значки в главном меню**, **Отображать часы**.

Флажок **Расположить поверх всех окон** позволяет размещать панель задач поверх окон всех приложений, которые вызваны на рабочий стол.

Флажок **Автоматически убирать с экрана** дает возможность убирать с экрана панель задач через несколько секунд после прекращения работы с ней. Возвращение панели осуществляется автоматически

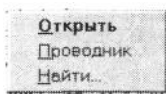
после пересечения курсора мыши соответствующего края рабочего стола. Флажки **Мелкие значки в главном меню** и **Отображать часы** включают в себя соответствующие функции.

Лист **Настройка меню** (см. рис. 2.21, 2) позволяет установить (или удалить) дополнительные строки в меню кнопки **Пуск**, обеспечивающие запуск соответствующих приложений, загруженных в файловую систему. Для установки приложения следует щелкнуть по кнопке **Добавить** и в новом диалоговом окне с помощью кнопки **Обзор** указать место расположения и имя устанавливаемого приложения. Выполнение операции удаления приложения из списка меню кнопки **Пуск** осуществляется с использованием кнопки **Удалить**. Для очистки списка **Документы** из меню кнопки **Пуск** первого уровня следует щелкнуть мышью по кнопке **Очистить** в поле **Меню «Документы»** (рис. 2.21, 4).

2.5. Настройка меню кнопки Пуск



Кнопка **Пуск**, размещенная на панели задач, позволяет вызвать многокаскадное меню, определяемое как **Главное меню**. Кроме того, она обеспечивает вызов на рабочий стол необходимых приложений. Определение содержания списка приложений, их разделение на отдельные каскады (панели) и составляет суть настройки меню кнопки **Пуск**.



Для настройки меню кнопки **Пуск** следует правой кнопкой мыши вызвать контекстное меню, на поле которого щелкнуть по строке **Открыть**. В результате на рабочий стол будет вызвано окно папки **Главное меню** (рис. 2.22).

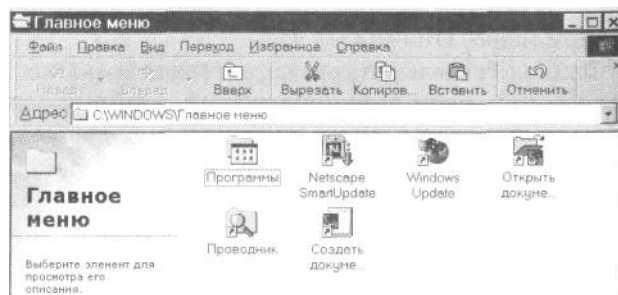


Рисунок 2.22. Окно папки **Главное меню**

На рабочем поле окна показаны значки и ярлыки приложений, которые запускаются из верхней области, а также меню строки **Программы** первой панели (каскада) меню кнопки **Пуск**. С помощью операционного меню окна можно выполнить различные операции как по установке новых приложений на поле первой панели меню, так и по изменению вида значка, установке условий запуска приложений и т. п.

Для установки, редакции или удаления объектов на других панелях меню следует активизировать значок **Программы**, размещенный на рабочем окне **Главное меню**.

Контекстное меню кнопки **Пуск** позволяет осуществить поиск файлов в дисковой памяти системы. Для этого следует щелкнуть по строке **Найти** контекстного меню. На рабочий стол будет вызвано окно **Найти: Все файлы**, показанное на рис. 2.23.

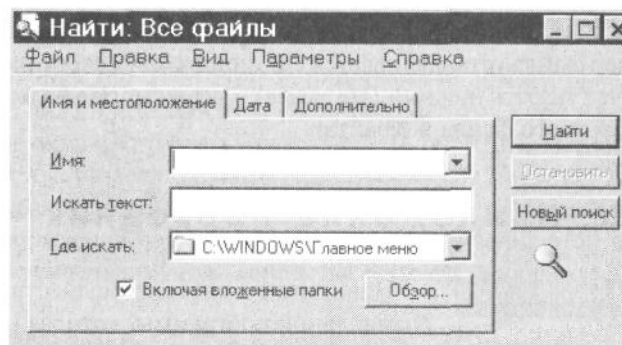


Рисунок 2.23. Окно **Найти: Все файлы** меню кнопки **Пуск**

Окно содержит три листа: Имя и местоположение, Дата и Дополнительно.

Лист **Имя и местоположение** содержит три поля: Имя, Искать текст, Где искать. Поле **Имя** представляет собой раскрывающийся список, в котором следует найти подходящее имя или самостоятельно указать имя искомого файла или папки (при этом нет различия в используемых знаках: строчные или прописные).

В поле **Искать текст** требуется записать необходимый текст, который станет ключом поиска.

С помощью поля **Где искать** можно сузить или расширить область поиска.

После заполнения необходимых атрибутов поиска: что искать и где искать, следует щелкнуть по кнопке **Найти**. Результат поиска будет отражен в раскрывающемся под окном списке. Если поиск не даст результата, то требуется уточнить критерии поиска, предварительно щелкнув мышью по кнопке **Новый поиск**.

Лист **Дата** позволяет определить рамки поиска по временному признаку. В частности, он содержит два переключателя: Все файлы и Найти все файлы. Последний соединен с раскрывающимся списком: **Измененные, Созданные и Открытые**.

При активизации положения переключателя **Найти все файлы** можно использовать три позиции: **Между, За последние месяцы, За последние дни**.

Позиция **Между** активизирует два раскрывающихся списка, при помощи которых можно установить временной интервал поиска. Другие две позиции имеют счетчики, позволяющие указать количество последних месяцев и дней соответственно для установки глубины поиска.

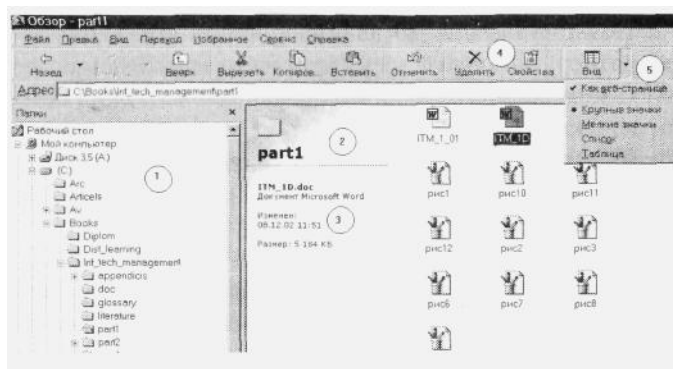
Лист **Дополнительно** позволяет указать в раскрывающемся списке тип искомого файла, а также ограничить поиск размером файла. Размер файла устанавливается с помощью двух счетчиков, на которых следует указать нижний (Не менее) и верхний (Не более) пределы объемов искомого файла в Кбайтах.

2.6. Приложение Проводник

Навигация в системе осуществляется с помощью специальной программы, которая называется **Проводник**. Обращение к этой программе можно выполнить из контекстного меню, а также из меню **Программы** кнопки **Пуск**.

Проводник позволяет не только отразить с помощью графических элементов структуру файловой системы, но и обеспечить оперативное управление как файлами, так и папками системы. Он позволяет свести процессы управления в файловой системе к операциям по активизации файлов и команд управления с помощью щелчков мышью по соответствующим кнопкам и значкам рабочего окна программы **Проводник**.

Окно программы **Проводник** (рис. 2.24) имеет две панели (левую и правую), операционное меню и панель инструментов. В левом окне размещается дерево связей, отражающее структуру файловой системы. Папки, имеющие вложенные папки, обозначаются знаком «+», который заменяется на знак «—», если открыть структуру соответствующей папки (каталога). Активизация (выделение) папки осуществляется щелчком мыши по ее значку.



1 — дерево папок диска или вложенной папки; 2 — поле содержания текущей папки; 3 — параметры выделенного объекта текущей папки; 4 — панель инструментов окна программы **Проводник**; 5 — раскрывающееся расширение кнопки **Вид**

Окно текущей папки (см. рис. 2.24, 2) помещено на правой панели рабочего окна. Его вид определяется кнопкой Вид, размещенной на панели инструментов окна программы **Проводник** (см. рис. 2.24, 5). Меню кнопки Вид имеет два раздела. Первый раздел содержит позицию **Как веб-страница**, которая использует дизайн браузера Internet Explorer. Пример подобного оформления показан на рис. 2.25. Отключение этой позиции приведет к стандартному (для семейства Windows) изображению.

Второй раздел меню кнопки **Вид** содержит четыре позиции: Крупные значки, Мелкие значки, Список и Таблица.

Имя	Разм...	Тип	Изменен
ITM_1_01	300 КБ	Документ Micr...	04.12.02 17:55
ITM_1D	5 184 ...	Документ Micr...	08.12.02 11:51
ITM_1C	651 КБ	Архив WinRAR	07.12.02 21:54
рис1	770 КБ	Точечный рис...	04.12.02 18:49
рис10	236 КБ	Точечный рис...	07.12.02 18:34

Рисунок 2.25. Оформление списка объектов в текущей папке при выборе позиции **Таблица** меню кнопки **Вид**

Позиция **Крупные значки** выводит построчно перечень объектов текущей папки в виде крупных значков. Каждый значок снабжен именем объекта.

Позиция **Мелкие значки** изображает построчно объекты текущей папки в форме мелких значков.

Позиция **Список** размещает значки объектов по столбцам. Позиция **Таблица** выдает информацию по каждому объекту: имя объекта, размер, тип, дата и время установки на рабочем столе (см. рис. 2.25).

Соотношение размеров панелей рабочего окна можно регулировать, перетаскивая линию раздела панелей мышью вправо и влево.

Если обозначения объектов в текущей папке не помещаются на правой панели рабочей области, то на соответствующей панели появляются линейки прокрутки.

Поиск папок или файлов в программе **Проводник** осуществляется следующим образом. В меню **Сервис** окна **Проводник** следует активизировать команду **Поиск**, а далее выбрать строку **Файлы и папки** и щелкнуть мышью. На поле рабочего стола будет вызвано окно поиска, в котором следует указать объект и режимы поиска (см. рис. 2.25). Окно поиска можно вызвать и с помощью контекстного меню, которое появляется щелчком правой кнопкой мыши по правой области рабочего окна **Проводник**.

В последних версиях Windows функции поиска значительно расширены. Так, в команде **Поиск** меню **Сервис** установлены позиции: В Интернете, С помощью Microsoft Outlook и Людей.

Первая из позиций предназначена для автоматического поиска информации, размещенной в Интернете. Вторая позиция — **С помощью Microsoft Outlook** — позволяет автоматически перейти к подготовке письма в сервисе электронной почты. Третья позиция — **Людей** — дает возможность отыскать нужный почтовый адрес в адресной книге электронной почты и перейти к подготовке письма.

Активизация (открытие, запуск) объекта из папки программы **Проводник** осуществляется двойным щелчком мыши по его значку или с помощью команды **Открыть** в меню **Файл**. Если объект является программой, то она будет запущена, если объект является документом, то будет активизирована и та среда, в которой он был создан. Если объект не распознан, то появится диалоговое окно, с помощью которого необходимо указать ту программу, в среде которой подготовлен документ или работает программа.

Открыть объект можно с помощью контекстного меню, которое появляется при щелчке правой кнопкой мыши по значку. В появившемся списке следует выбрать позицию **Открыть** и щелкнуть мышью, если объект не будет опознан, то необходимо выбрать позицию **Открыть с помощью**.

Панель инструментов **Проводника** (рис. 2.26) содержит ряд кнопок, число которых может быть увеличено. Их использование позволяет быстро выполнить следующие операции (слева направо):

- ◆ перейти к ранее выполненной команде (рядом с кнопкой имеется значок раскрытия списка кнопки);
- ◆ перейти к последующей команде (если был возврат к ранее выполненной операции);
- ◆ перейти на один уровень вверх (показывается папка более высокой иерархии);

- ◆ вырезать выделенный объект (при этом он перемещается в промежуточную память и ждет операции по вставке в соответствующую папку);
- ◆ копировать выделенный объект (при этом его копия переводится в промежуточную память и ждет операции по вставке);
- ◆ вставить объект из промежуточной памяти в область активизированной папки;
- ◆ отменить действие ранее выполненной операции;
- ◆ удалить выделенные объекты в **Корзину**;
- ◆ вывести на рабочий стол лист свойств выделенного объекта;
- ◆ дублировать соответствующие операции меню **Вид**.



1 — Назад; 2 — Вперед; 3 — Вверх; 4 — Вырезать; 5 — Копировать; 6 — Вставить; 7 — Отменить; 8 — Удалить; 9 — Свойства; 10 — Вид; 11 — раскрытый список кнопки **Назад**

Рисунок 2.28. Панель инструментов Проводника

Внизу окна **Проводника** находится строка состояния. Она содержит информацию о текущем выделенном объекте **Проводника**. Для выключения или включения строки состояния следует активизировать позицию **Строка состояния** команды Вид системного меню.

Для выделения объекта необходимо щелкнуть один раз мышью по его значку, для выделения группы смежных объектов следует выделить первый объект, затем нажать клавишу **<Shift>** и щелкнуть по значку последнего из группы объектов. Если необходимо выделить группу из отдельных объектов, то нужно нажать и не отпускать клавишу **<Ctrl>** и щелкать мышью по отдельным объектам из группы. Последнюю операцию можно выполнить, используя клавиши-стрелки, перемещая курсор по полю значков и нажимая клавишу **<Space>** для выделения объекта, не забывая держать нажатой клавишу **<Ctrl>**.

Проводник можно также использовать для копирования, перемещения и переименования файлов, что осуществляется тем же методом, который применяется на рабочем столе. Для выполнения указанных операций следует на правой панели выделить копируемый(ые) объект(ы), а на левой — создать папку, куда следует разместить объект(ы). Затем необходимо правой кнопкой мыши перетащить объект(ы) с правой панели на значок соответствующей папки левой панели и выбрать необходимую команду из контекстного меню: **Вырезать** или **Копировать**.

Другой способ копирования и перемещения осуществляется с помощью кнопок панели инструментов: **Вырезать**, **Копировать**, **Вставить**. При этом методе следует выделить объект или группу объектов и нажать на кнопку **Вырезать** (для перемещения) или **Копировать** (для копирования). Затем необходимо на любое поле **Проводника** вызвать папку документов, предназначенную для приема новых объектов, и выделить ее. Нажать на кнопку **Вставить**.

Переименовать объект можно с помощью дополнительного (одиночного!) щелчка мышью по выделенному полю имени объекта и записи на это поле нового имени. Завершить операцию следует нажатием на клавишу **<Enter>**. После выполнения операции необходимо нажать клавишу **<F5>** для обновления изображения рабочего стола.

2.7. Приложение Корзина



С помощью приложения **Корзина** (раньше это приложение обозначалось как **Мусорка**), значок которого обязательно находится на рабочем столе, осуществляются операции удаления и восстановления объектов. Для восстановления объекта с помощью **Корзины** следует ее открыть, выбрать имя удаленного объекта из предложенного списка, щелкнуть на

нем правой кнопкой мыши и выбрать команду **Восстановить**. Даже если удалена папка, она будет восстановлена полностью! Следует помнить, что файлы, удаленные в сеансе MS DOS, в **Корзину** не попадают.

Корзину можно очистить, что повлечет окончательное удаление объектов. Для этого следует воспользоваться командой **Очистить корзину** в меню **Файл**.

Можно удалить отдельные объекты из **Корзины** с целью увеличения дискового пространства системы. Для этого следует выделить удаляемые объекты и нажать на клавишу **** или воспользоваться командой **Удалить** на панели инструментов **Корзины**.

2.8. Приложение Портфель



Портфель является специальным типом системной папки, с помощью которого определяется место, время и факт изменения файла, его объем. Следует отметить, что в Windows XP это приложение не имеет большого значения. Приложение предназначено для реализации возможности работы с документом на нескольких компьютерах одновременно. Его можно использовать для следующих целей:

- ◆ копирование документов с одного компьютера в **Портфель** другого;
- ◆ модификация документа в **Портфеле**;
- ◆ возвращение документа в исходную систему, где находится оригинал документа;
- ◆ сверка копии документа с оригиналом;
- ◆ информация о последней копии документа и ее модификации;
- ◆ передача изменений и их соединение в последней версии документа.

Суть работы **Портфеля** можно проиллюстрировать следующим примером. Если передать на другой компьютер документ из **Портфеля**, а затем внести изменения в документ с помощью нового компьютера и снова вернуть в **Портфель** первого компьютера, то система проинформирует об изменениях и устранил различия между двумя документами, исправив более раннюю версию.

Наиболее простой способ использования **Портфеля** связан с возможностью его копирования на дискету. Для этого необходимо передать подготовленные документы в **Портфель**, затем переписать **Портфель** на дискету. Теперь можно работать с файлами дискеты, используя другие компьютеры. При возвращении на исходный компьютер следует воспользоваться командами **Портфеля** для модификации файлов. Создание **Портфеля** осуществляется в корневом каталоге или на рабочем столе. **Портфель** должен находиться на том компьютере, на который перемещается документ для дальнейшей работы с ним.

Для создания **Портфеля** необходимо щелкнуть правой кнопкой мыши по выделенному объекту и выбрать команду **Отправить**, в новом списке активизировать опцию **Портфель**. Выделенный документ будет скопирован в папку **Портфель**.

Использование **Портфеля** с помощью дискеты предполагает выполнение следующей последовательности операций:

1. Копирование в **Портфель** документов, предназначенных для работы на других компьютерах.
2. Создание **Портфеля** и перемещение его на дискету с помощью ведения мышью значка **Портфель** или с поля рабочего стола, или из окна папки **Мой компьютер**, или окна **Проводник**.
3. Установка диска на другой компьютер и перетаскивание необходимых файлов на новый рабочий стол для облегчения доступа к этим файлам. Для перемещения следует использовать правую кнопку мыши и выбрать из контекстного меню команду **Переместить**.
4. Перемещение откорректированных файлов в **Портфель**, а оттуда вместе с **Портфелем** снова на дискету.

Для обновления и синхронизации содержимого файлов необходимо:

1. Открыть **Портфель**, размещенный на дискете, или переместить его на рабочий стол.
2. Выделить документы в **Портфеле** и выбрать из меню окна последовательно команды **Портфель** и **Обновить выделенные объекты**. Для обновления всех документов следует выбрать команду **Обновить все**. Если оригинал и копия не обновлялись, то будет выведена соответствующая надпись — при их обновлении будет показано диалоговое окно, на котором будет установлен значок документа, время изменения документа, требуемая операция для обновления документа и состояние исходной копии документа.

3. Выбрать команду **Обновить. Портфель** заменит исходную копию новой. Если во время отсутствия документа (размещенного на дискете) в копию документа, оставшуюся на компьютере, были внесены изменения, то **Портфель** заменит документ, размещенный на дискете, если изменилось содержание той копии, что находилась на дискете, то изменится файл, оставшийся в компьютере. Если изменились обе копии, то изменять содержимое копии необходимо вручную.

4. Щелкнуть правой кнопкой мыши на документе **Портфеля** и в контекстном меню активизировать одну из команд: **Заменить копию исходным документом, Заменить исходный документ копией, Пропустить.**

2.9. Приложение Блокнот

Блокнот (рис. 2.27) представляет собой простой текстовый редактор, предназначенный для записи оперативной информации с целью ее последующей обработки с помощью более развитых редакторов, например Word Pad Windows, Word MS Office и др.

В **Блокноте** можно писать только одним встроенным шрифтом, в нем отсутствуют какие-либо средства форматирования текста, нельзя вставлять графические объекты. Файлы, созданные этим редактором, имеют расширение .txt, однако для их использования в DOS необходима конвертация, так как кодировка русскоязычных алфавитов DOS и Windows не совпадает.

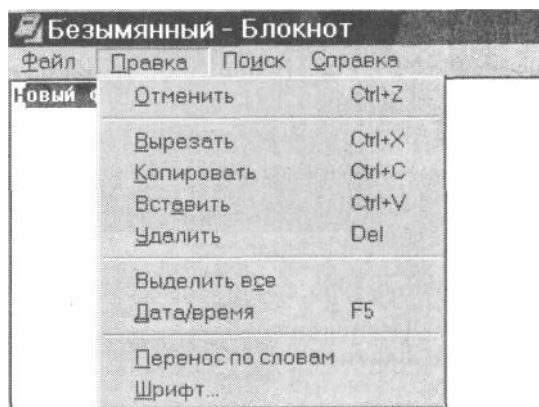


Рисунок 2.27. Фрагмент рабочего окна **Блокнот** с раскрытым меню **Правка**

Системное меню редактора состоит из четырех пунктов:

Файл (File) — запись, чтение и печать редактируемого файла;

Правка (Edit) — выделение, вставка, копирование текстовых фрагментов, вставка текущей даты в любое место документа, включение/ отключение переноса строк;

Поиск (Search) — поиск образца текста;

Справка (Help) — система справки.

2.10. Приложения Назначенные задания и Дата и время



Приложение **Назначенные задания** позволяет сформировать задания, которые будут выполнены операционной системой в указанное время.

Запуск приложения осуществляется двойным щелчком мыши по значку папки **Планировщик**, расположенному в правой части (рядом с переключателем алфавитов) панели задач.

На рис. 2.28 показаны две возможности установки задания: **Добавить задание** и **Запуск Мастера настройки**. В первом случае следует самостоятельно указать объект активизации и время активизации объекта. Во втором случае на рабочий стол вызывается вкладка **Запуск мастера настройки**, содержащая три листа: **Задание**, **Расписание** и **Настройка** (см. рис. 2.28). На листе **Задание** следует указать объект файловой системы текстом или с помощью кнопки **Обзор**, снабдив этот объект необходимым текстовым комментарием. На листе **Расписание** следует показать время и периодичность запуска объекта. На третьем листе — **Настройка** — требуется установить граничные временные условия, а также ограничения, связанные с переменной источника питания в системе.

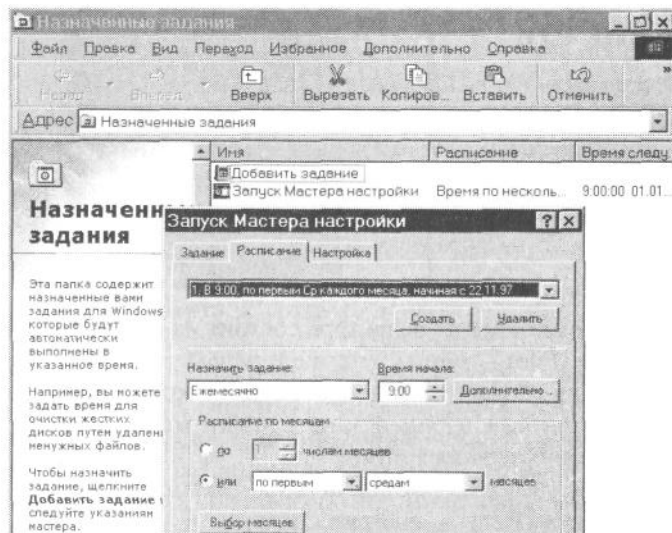


Рисунок 2.21. Окно папки **Назначенные задания** и вкладка **Запуск Мастера настройки**

Окно снабжено операционным меню, позволяющим выполнять различные операции управления рабочим окном приложения и самими файлами заданий.



Приложение **Дата и время** вызывает на поверхность стола календарь текущего времени и изображение циферблата (рис. 2.29). Оно вызывается или щелчком мыши по текущему времени, показываемому в правом углу панели задач, или из папки **Настройка** двойным щелчком мыши по значку **Дата и время**.



Рисунок 2.29. Приложение **Дата и время**

2.11. Приложение Paint



Основная цель изучения этой подглавы состоит в овладении следующими навыками:

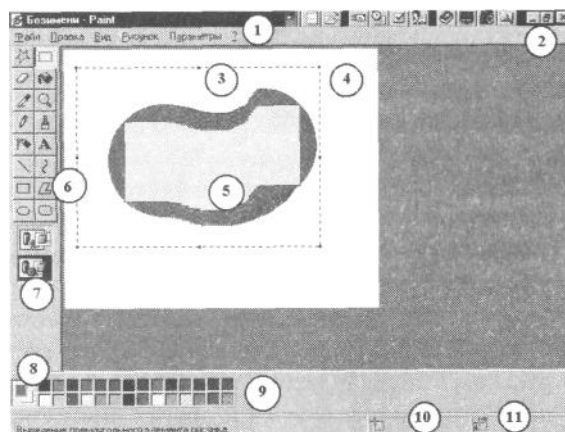
- ◆ вызов и прекращение работы в графическом редакторе **Paint** (имя редактора трудно однозначно перевести на русский язык, и поэтому во всех русскоязычных версиях Windows сохранено его американское имя, но представляется, что его можно обозначить русским словом **Этюдник** — небольшой ящик, для красок, карандашей, ластиков, бумаги и других инструментов и материалов, которыми пользуются живописцы в своей работе);
- ◆ выбор и использование различных инструментов для рисования;
- ◆ изображение прямых, ломаных линий, окружностей, эллипсов, овалов, прямоугольников, многоугольников, нанесение штриховки;
- ◆ начертание плоских и трехмерных изображений;
- ◆ раскрашивание изображений;
- ◆ перемещение по экрану, изменение размеров и сохранение фрагментов изображений;
- ◆ нанесение текста на рисунок;
- ◆ использование буфера обмена для перенесения изображений на документы.

2.11.1. Основные элементы графического редактора Paint

Графический редактор **Paint** предназначен для создания, хранения, перемещения и вывода на печать различных изображений (рисунков). Его можно вызывать из других программ для вставки графических объектов.

Paint является однодокументным приложением, т. е. в каждый момент времени он обрабатывает только одно изображение. Графический редактор **Paint** обычно размещается на панели третьего уровня меню кнопки **Пуск**, куда можно попасть, открыв папку **Программы** на панели первого уровня, а затем **Стандартные** на панели второго уровня.

После загрузки приложения на экране появляется окно, элементы которого представлены на рис. 2.30.



1 — заголовок окна; 2 — системное меню; 3 — область выделения; 4 — рабочее поле рисунка; 5 — рисунок; 6 — инструменты приложения; 7 — указатель переднего плана; 8 — указатель цвета; 9 — цветовая палитра; 10 — координаты курсора мыши; 11 — координаты выделенной области

Рисунок 2.30. Окно графического редактора Paint

Поле рисунка размещается на рабочем поле экрана и ограничено контурной линией с выделенными точками по периметру. Поле рисунка может иметь различные размеры, устанавливаемые с помощью выбора позиции **Атрибуты** пункта **Рисунок** системного меню редактора (рис. 2.31).

Для изменения размеров поля рисунка следует либо установить новые габаритные размеры в диалоговом окне **Атрибуты**, указав единицу измерения, либо, ухватив мышью за одну из выделенных контурных точек поля рисунка, переместить контурные линии листа на новое место.

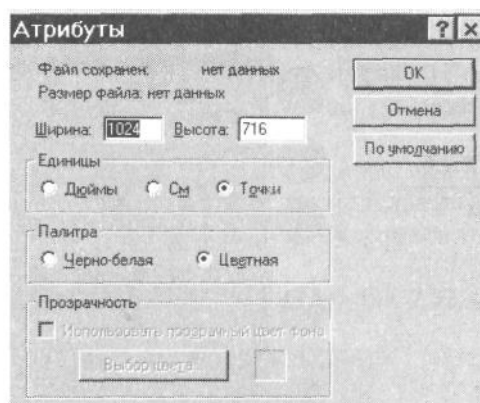


Рисунок 2.31. Диалоговое окно Атрибуты

При загрузке нового рисунка приложение выделяет рабочую область листа. Если размеры рисунка превосходят рабочую область, то после предупреждения размер листа увеличивается до размера рисунка. Если размеры рисунка превосходят размеры рабочего поля редактора, то появляются линейки прокрутки, которые позволяют перемещаться по полю рисунка. Перемещать рисунок по полю экрана

можно также с помощью перетаскивания мыши, предварительно выделив все поле листа. В этом случае экран показывает только часть изображения – окно экрана движется по изображению.

Курсор редактора на рабочем поле имеет вид, определяемый выбранным инструментом, а вне рабочего поля принимает форму стрелки.

2.11.2. Системное меню графического редактора Paint

Меню **Файл** (рис. 2.32) предназначено для выбора выполняемой стандартной операции по созданию, сохранению, вызову, считыванию графических файлов и выводу изображений на печать.

Меню **Файл** имеет следующие группы опций:

1. Группа опций для работы с графическим файлом (см. рис. 2.32, 1):

Создать — вызывает на рабочую область редактора новый лист, окрашенный цветом фона;

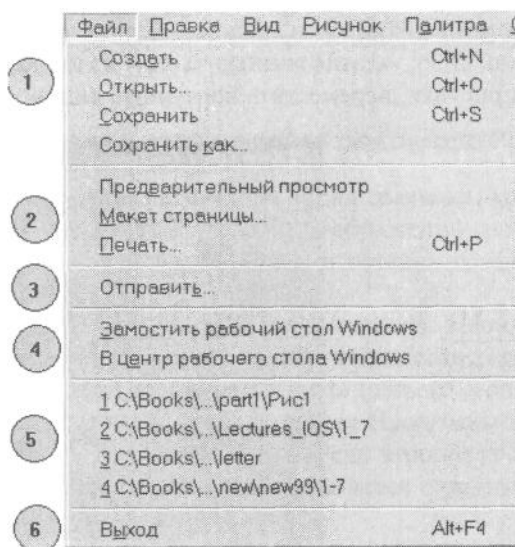


Рисунок 2.32. Меню Файл

Открыть — вызывает диалоговое окно, на котором показан текущий каталог. Если необходимо вызвать файл, расположенный в другом каталоге, то следует сначала указать имя диска, затем выбрать из списка соответствующий каталог, указать щелчком мыши выбранный файл и двойным щелчком загрузить файл рисунка в рабочую область редактора;

Сохранить — записывает подготовленное изображение в файл с текущим именем;

Сохранить как... — запоминает созданный рисунок в файле с новым именем.

2. Группа опций для работы с подготовленным рисунком: **Предварительный просмотр**, **Макет страницы**, **Печать** (см. рис. 2.32, 2). Эти опции позволяют просмотреть в реальных пропорциях подготовленный для печати рисунок и осуществить вывод его на печать. В процессе печати вызывается диалоговое окно, с помощью которого устанавливаются: качество печати (черновик или качественная), объем выводимого изображения (полностью или выделенный фрагмент), масштаб изображения, растянуть или сжать изображение (для матричных принтеров) и количество копий.

3. Опция **Отправить** (см. рис. 2.32, 3), которая осуществляет создание папки **Входящие** для передачи рисунка по факсу, электронной почте или по сети Microsoft NetWork.

4. Группа опций, обеспечивающая использование графического файла для создания рисунка на рабочем столе Windows XP (см. рис. 2.32, 4).

Замостить рабочий стол Windows — установка подготовленного изображения в качестве рисунка для рабочего стола системы. Для восстановления рисунка рабочего стола следует открыть папку **Настройка** первой панели меню кнопки **Пуск**, затем папку **Панель управления** второй панели, активизировать значок **Экран**. На появившейся вкладке **Свойства: Экран** выбрать лист **Фон** и в поле **Рисунок** выделить нужный рисунок;

В центр рабочего стола Windows — подготовленное изображение заполняет центр рабочего стола Windows.

5. Область имен подготовленных графических файлов, выделяющее несколько строк меню для указания имен ранее использовавшихся файлов (см. рис. 2.32, 5). Область применяется для оперативного вызова файлов (рис. 2.32, 5).

6. Опция **Выход**, завершающая работу в Paint (см. рис. 2.32, 6).

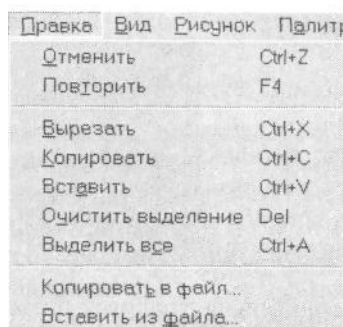


Рисунок 2.33. Меню Правка

Команда **Правка** (рис. 2.33) содержит следующие опции:

Отменить — указывает на отмену последних выполненных операций, например рисования линии, заливки контура, вырезки и перемещения фрагмента и т. п. Команду можно реализовать нажатием комбинации клавиш **<Ctrl> + <Z>**. Команда отменяет только ранее выполненные действия в обратном порядке: последнее действие отменяется первым;

Повторить — восстанавливает отмененное действие;

Вырезать — удаляет с листа выделенный фрагмент рисунка и переводит его в область промежуточного хранения (буфер);

Копировать — копирует выделенную область рисунка в область промежуточного хранения (буфер);

Вставить — устанавливает рисунок из области промежуточного хранения в верхний левый угол листа (рабочей области редактора). Для дальнейшей работы с этим фрагментом его надо перетащить с помощью мыши в нужное место изображения и щелкнуть мышью вне поля фрагмента или по кнопке **Выделение фрагмента**. Если в области промежуточного хранения находится текст, то перед его вызовом необходимо установить текстовый курсор;

Очистить выделение — удаляет выделенный фрагмент;

Выделить все — выделяет все поле рисунка;

Копировать в файл — записывает под новым именем отредактированное изображение или выделенный фрагмент;

Вставить из файла — устанавливает рисунок или фрагмент из графического файла.

Меню **Вид** содержит 6 следующих опций (рис. 2.34):

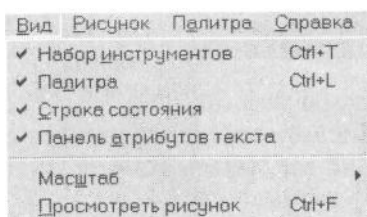


Рисунок 2.34. Меню Вид

Набор инструментов — устанавливает метку на левой части строки и определяет необходимость изображения панели инструментов на рабочем поле редактора;

Палитра — устанавливает палитру цветов инструмента и фона (бумаги) рисунка;

Строка состояния — устанавливает внизу экрана строку с указанием места положения курсора на рабочем листе, координаты правого нижнего угла области выделения;

Панель атрибутов текста — при записи текста на поле рисунка автоматически устанавливает панель форматирования, на которой можно изменить кегль, форму выделения и гарнитуру шрифта;

Масштаб — вызывает панель, на которой можно установить масштаб изображения для приближения рисунка и указать процент уменьшения рисунка по вертикали и горизонтали;

Просмотреть рисунок — убирает с экрана все обрамление рисунка, оставляя на экране только рисунок.

Меню **Рисунок** (рис. 2.35), предназначено для управления шрифтом при вводе текста. Меню содержит следующие опции:

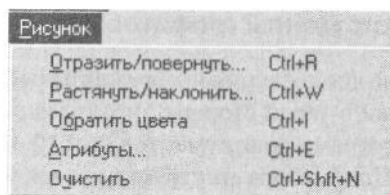


Рисунок 2.35. Меню Рисунок

Отразить/повернуть — вызывает диалоговое окно **Отражение и поворот**, которое включает в себя кнопки: Отобразить слева направо (т. е. создать зеркальное отображение рисунка относительно вертикали); Отобразить сверху вниз; Повернуть на угол 90°, 180°, 270°;

Растянуть/наклонить — активизирует диалоговое окно **Растяжение и наклон**, которое состоит из двух областей: **Растянуть** и **Наклонить**.

В каждой области имеются два текстовых окна, в которых следует указать проценты и градусы изменения геометрии поля рисунка относительно горизонтали и вертикали.

Следует пояснить, что операция выполняется в том случае, если осуществлено выделение рисунка или его фрагмента. При выполнении операции наклона по горизонтали происходит перемещение верхней линии выделения относительно нижней, которая во время выполнения операции остается неподвижной. При наклоне рисунка по вертикали перемещается правая линия выделения, а левая сторона выделенной области остается неподвижной;

Обратить цвета — заменяет цвета рисунка на противоположные, например белый на черный, синий на желтый, черный на белый и т. д.;

Атрибуты — сообщает и позволяет изменить размеры листа рисунка; единицу измерения указанных размеров: дюймы, сантиметры, пиксели; используемую палитру: черно-белая или цветная;

Очистить — заменяет на листе изображение цветом фона;

Непрозрачный фон — устанавливает активность фона при перемещении фрагментов рисунка.

Меню **Палитра** позволяет изменить, сохранить и загрузить новую палитру.

2.11.3. Инструменты графического редактора Paint

Средства для создания изображения собраны в инструментарий, расположенный в левой стороне экрана и представляющий собой блок из меню пиктограмм инструментов (рис. 2.30, 6).

Для выбора инструмента нужно указать курсором на соответствующую пиктограмму и щелкнуть мышью, при этом пиктограмма выделяется дополнительным цветом.

Для работы с выбранным инструментом курсор следует перенести в область рисунка и вести мышью с нажатой левой кнопкой.

Перечислим стандартный набор инструментов сверху вниз и слева направо.

Выделение произвольной области. Инструмент предназначен для вырезания, копирования и перемещения произвольных фигур. Для выделения произвольной области необходимо установить курсор в форме крестика в начальное положение и тащить мышью, обводя предназначенную для выделения часть изображения и щелкая на каждой точке изменения направления линии выделения. Если отпустить кнопку мыши до того, как будет замкнута линия, начальная и конечная точки замкнутся прямой линией, а контур выделения станет пунктирным.

Выделение. Применяется для выделения участков рисунков с целью их копирования, вырезания, перемещения. Граница выделенной области помечается пунктирной линией. Для отмены выделения следует щелкнуть мышью за пределами выделенной области.

Выделенный текст можно переместить по рабочему полю редактора. Для этого требуется:

- ♦ установить курсор внутри выделенной области;

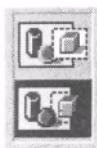
- ◆ нажать левую кнопку мыши;
- ◆ тащить до нужного места;
- ◆ отпустить кнопку мыши;
- ◆ щелкнуть мышью вне выделенного контура.

Использование выделения позволяет создать некоторые специальные эффекты изображения, например создание шлейфа. Для этого следует выполнить следующие операции:

- ◆ выделить нужную область;
- ◆ нажать клавишу <Shift>;
- ◆ поместить курсор в область выделенного фрагмента, нажать на кнопку мыши и тащить его по рабочему полю.

Если нажать клавишу <Shift>, то при дальнейшем перемещении фрагмента не будут появляться новые копии.

При нажатии на правую кнопку мыши будет перемещаться весь фрагмент, и он полностью закроет цветовой фон.





При использовании кнопок выделения на панели размера инструментов показаны два способа выделения объектов на рисунке (см. рис. 2.30, 7). Верхнее изображение показывает, что в процессе перемещения выделенного объекта он будет перемещаться со своим фоном (прозрачный фон). Нижний рисунок определяет режим перемещения с прозрачным фоном. В этом случае перемещается только объект, а цвет листа определяется цветом того места, куда перемещается объект. Внутренняя часть выделенной и перенесенной области заполняется текущим цветом.


Выбор режима перемещения определяется щелчком мыши по соответствующему рисунку.


Перемещение при нажатой клавише <Shift> приводит к размазыванию выделенного фрагмента по выделенному пути.


Перемещение при нажатой клавише <Ctrl> позволяет размножить фрагмент: переместить его в нужное место и отпустить кнопку мыши, затем снова нажать и переместить в другое место и т. д.


 **Ластик/Цветной ластик.** Используется для снятия фрагментов изображения при перемещении инструмента по его поверхности. Ластик оставляет неизменным цвет фона. Всегда имеет форму квадрата выбранного размера (определяется размером линии в области выбора размера инструмента).


 **Заливка.** Применяется для заливки краской обрамленных участков рисунка. Заливка используется для закраски (заливки) цветом инструмента (первым цветом) внутренней части любого замкнутого контура. Для этого нужно поместить острие курсора инструмента в любое место внутри контура. Если цвет заливки неудачен или краска «растеклась» за пределы контура, то следует использовать команду **Отменить** из меню **Правка**.

 **Выбор цветов.** С помощью этого инструмента можно определить цвет фрагмента или линии рисунка. После щелчка по кнопке курсор в виде сканера необходимо привести на выбранный участок изображения, на левой панели (под панелью инструментов) показывается определенный системой цвет.


 **Масштаб.** На панели размера инструмента устанавливается переключатель: 1x, 2x, 6x, 8x, на котором следует выбрать требуемое увеличение фрагмента рисунка.


 **Карандаш.** Позволяет наносить произвольные линии, штриховку и другие элементы на поле рисунка, применяется для рисования произвольных линий, раскрашивания частей изображения и т. п. Для его использования необходимо щелкнуть на соответствующей пиктограмме и поместить карандаш в нужную точку рисунка, затем следует нажать на левую кнопку мыши и водить карандашом по рисунку.


 **Кисть.** С ее помощью наносится плоская линия заданного цвета. При работе с карандашом и кистью следует задать их размер (толщину) и форму, а также установить цвет краски (первый цвет) на панели размера инструмента, которая размещена под панелью инструментов. Если во время работы щелкнуть правой кнопкой мыши, то изображение исчезнет.


 **Распылитель (Пульверизатор, Баллончик).** Разбрызгивает краску кругами задаваемого на панели размера инструмента радиуса. Распылить краску можно непрерывно, перемещая мышью с нажатой кнопкой: плотность слоя краски зависит от скорости движения мыши или от числа накладываемых слоев.


Кисть, карандаш, распылитель и ластик при нажатой клавише <Shift> работают строго в горизонтальном направлении.

 **Надпись.** Дает возможность ввести в изображение текст. После активизации этой кнопки курсор следует перенести на поле рисунка и выделить место записи текста, протаскив мышью по диагонали. Появится пунктирная рамка, обрамляющая поле текста, а в ней текстовый курсор, позволяющий ввести текст. Если нажать на клавишу <Enter>, то курсор перейдет на следующую строку. После щелчка на поле рисунка выделенный текст устанавливается на изображении и становится его частью. Текст можно выделить и переместить на другое место листа. С помощью инструмента Выделение текст, так же как и другие части изображения, можно подвергнуть различным преобразованиям: сжать или растянуть, инвертировать, отобразить зеркально, придать ему дополнительный наклон и т. д.


 **Линия.** Для рисования линии необходимо установить курсор на ее начало, нажать на кнопку мыши и тащить ее по любому пути в конечную точку, после чего отпустить кнопку мыши. Чтобы изобразить строго вертикальную или горизонтальную линию, следует нажать клавишу <Shift>. Если во время работы щелкнуть правой кнопкой мыши, то изображение исчезнет.

 **Кривая (Дуга, Изгиб).** Позволяет нанести плавные изгибающиеся линии. Для этого необходимо установить курсор на начало линии и, не отпуская кнопку мыши, перетащить ее в конечную точку. После этого следует отпустить кнопку мыши. На рисунке появится тонкая линия. Перевести курсор мыши на предполагаемую точку изгиба линии и тащить его (вместе с линией) в заданном направлении. В это время, передвигаясь по линии, можно менять точку изгиба. После получения линии нужной кривизны отпустить кнопку, а затем еще раз щелкнуть ею. Если во время работы щелкнуть правой кнопкой мыши, то изображение исчезнет. Волнистая линия получается последовательным изображением изогнутых линий.

 **Прямоугольник.** Для его изображения следует перенести курсор в точку любого угла изображаемой фигуры, перетащить мышью в противоположный угол создаваемого изображения и отпустить кнопку мыши. Если во время работы щелкнуть правой кнопкой мыши, то изображение исчезнет.

 **Многоугольник.** Позволяет изобразить контур любых многоугольников. Выбрав эту пиктограмму, следует установить курсор в одном из углов многоугольника и, нажав на левую кнопку мыши, переместить ее во второй угол и отпустить кнопку мыши. Снова нажать кнопку мыши и переместить ее в третью точку фигуры. Повторяя показанные действия, необходимо обвести весь контур многоугольника, снова возвратившись в точку его первого угла. Для отмены изображения нестроенного многоугольника следует щелкнуть правой кнопкой мыши.

Проведение строго горизонтальных и вертикальных линий обеспечит нажатие на клавишу <Shift>.

 **Скругленный прямоугольник.** Для его изображения необходимо выбрать соответствующую пиктограмму, перевести курсор в то место, где должен находиться один из углов прямоугольника. Нажать на кнопку мыши и тащить курсор в сторону противоположного угла. После появления необходимой фигуры отпустить кнопку мыши. Для построения точного квадрата или окружности следует выполнить описанную операцию при нажатой клавише <Shift>. Если во время работы щелкнуть правой кнопкой мыши, то изображение исчезнет. **Эллипс.** Для его изображения следует перенести курсор в точку любого угла, находящегося вне изображаемой фигуры, затем перетащить мышью в противоположный угол создаваемого изображения и отпустить кнопку мыши. Если во время работы щелкнуть правой кнопкой мыши, то изображение исчезнет.

2.11.4. Вставка графических изображений

Техника вставки графического изображения состоит в использовании последовательности операций вставки и копирования фрагментов изображения. Для этого следует использовать две группы команд. К первой относятся команды, расположенные в верхней части меню **Правка: Вырезать, Копировать и Вставить**. Все они используют для запоминания фрагмента изображения область промежуточного хранения Windows. Вторая группа команд собрана в нижней части меню **Правка: Копировать в файл, Вставить из файла**.

Область промежуточного хранения, или **карман**, представляется как некоторая область памяти, в которую передается информация на хранение. Емкость кармана не ограничена, но может вместить только один фрагмент, который теряется при выходе из среды Windows или при загрузке в него другого фрагмента.

Для передачи фрагмента изображения в карман следует выполнить следующую последовательность команд:

- ◆ выделить фрагмент и выбрать команду **Вырезать**. Эта команда удалит фрагмент из рабочей области и передаст его на хранение в карман. Образовавшаяся дырка в рабочей области закрашивается текущим цветом фона;
- ◆ скопировать из кармана выделенное изображение и поместить его на рабочее поле с помощью команды **Вставить**. При этом один экземпляр фрагмента остается в кармане, а второй возвращается на поле изображения.

В левом углу экрана появится изображение кармана. Остается переместить изображение на нужное место рисунка.

Практикум: Работа в графической операционной системе семейства WINDOWS

1. Настройка рабочего стола системы.

Цель работы: сформировать навыки организации рабочего стола в среде информационной офисной системы, овладеть приемами оформления рабочего стола, построения файловой системы, создания персональных папок и установки ярлыков на рабочем столе.

1.1. Проведите подготовительные операции.

1.1.1. Вызовите меню настройки рабочего стола:

а) включите компьютер и после его загрузки на свободном поле рабочего стола щелкните правой кнопкой мыши;

б) в появившемся контекстном меню выберите опцию **Свойства** (или **Рабочий стол Active Desktop** и далее **Настроить рабочий стол**), щелкните левой кнопкой мыши;

в) на вкладке **Свойства: Экран** щелчком мыши выберите первый лист вкладки: **Фон**.

1.1.2. Установите рисунок на поле рабочего стола:

а) на поле **Рисунок рабочего стола** выделите щелчком мыши рисунок из предлагаемого списка документов и рисунков (например, **Волны**);

б) в раскрывающемся списке **Расположить** установите строку **Рядом** и ознакомьтесь с рисунком на демонстрационном окне.

1.1.3. Измените содержание рисунка на поле рабочего стола:

а) выберите новую строку **По центру** в раскрывающемся списке **Поместить**, переведите курсор на первую строку в списке рисунков (на ней установлена позиция **Отсутствует**) и щелкните по кнопке **Узор**;

б) на новой вкладке **Фоновый узор** выберите из предлагаемого списка необходимый узор (например, **Волны**) и щелкните по кнопке **Изменить**. На экране появится увеличенный рисунок узора. На его поле с помощью щелчков мышью можно изменить цвет отдельных участков рисунка на противоположный. Выполнив 2—3 изменения в рисунке, щелкните по кнопкам **Изменить** и **Готово**. На вкладке **Фоновый узор** нажмите на кнопку **ОК**. На вкладке **Свойства: Экран** щелкните по кнопке **Применить**. На рабочем столе появится новый рисунок фонового узора.

1.1.4. Увеличьте рисунок на рабочем столе:

а) вернитесь к списку рисунков;

б) выберите новый файл с рисунком из списка, например **Пузырьки**, выберите строку **Растянуть** в раскрывающемся списке **Расположить** и нажмите на кнопку **Применить**. Увеличенный рисунок будет установлен на поле рабочего стола.

1.1.5. Удалите рисунок (фоновый узор) с поля рабочего стола:

а) для удаления фонового узора следует снова вернуться в режим изменения фонового узора, вызвать вкладку **Фоновый узор** и в списке **Узор** активизировать позицию **Отсутствует**;

б) щелкните по кнопке **Изменить**, закройте вкладку **Фоновый узор**. Нажмите на кнопку **Применить**.

1.1.6. Откажитесь от использования рисунков на поле рабочего стола. 1.2. Установите заставку на поле рабочего окна.

1.2.1. Выберите изображение для заставки:

а) перейдите на второй лист **Заставка** вкладки **Свойства: Экран**;

б) раскройте список **Заставка** и выберите, например, строку **В мире Windows**, щелкните по кнопке **Просмотр**;

в) ознакомьтесь с изображением в режиме **Заставка**, для окончания режима просмотра — передвиньте мышь по коврику: контрольное изображение исчезнет, а система восстановит вкладку **Свойства: Экран**.

1.2.2. Установите новые параметры воспроизведения заставки:

а) щелкните по кнопке **Настройка**. На новой вкладке выполните следующие действия: измените положение бегунка **Скорость**, введите новое значение в поле **Плотность заполнения** и нажмите на кнопку **ОК**. Снова щелкните по кнопке **Просмотр**.

1.2.3. Смените заставку:

а) вернитесь на поле листа **Заставка**;

б) из списка **Заставка** выберите новый рисунок, например **Объемный текст**. Измените параметры текста, предварительно щелкнув по кнопке **Настройка** и вызвав диалог **Параметры заставки «Объемный текст»**: введите новый текст в поле **Строка**, например Ваше имя или индекс студенческой группы, просмотрите результат;

в) вернитесь на поле диалога **Параметры заставки «Объемный текст»** и проведите дополнительную редакцию формы объемного текста: выберите с помощью ползунка размер изображения текста: **Мелкий** — **Крупный**, необходимое разрешение: **Низкое** — **Высокое**, скорость передвижения рисунка по экрану: **Низкая** — **Высокая**; установите рисунок поверхности текста: **Сплошной** или **С текстурой** (в последнем случае система переходит в режим выбора цвета); определите стиль движения объемного текста с помощью раскрывающегося списка **Стиль движения**: **Без вращения**, **Качели**, **Волны**, **Произвольный**. Вызовите

заставку на поле экрана и убедитесь в правильности выполненных операций.

1.2.4. Установите время вызова заставки:

а) в поле **Интервал** листа **Заставка** вкладки **Свойства: Экран** установите минимальное время ожидания (1 мин.) и нажмите на кнопку **Применить**. Проверьте действие выполненной операции;

б) вернитесь на лист **Заставка** и установите интервал 10 мин.

1.3. Оформите элементы рабочего стола (окно сообщения, активное окно и др.).

1.3.1. Выберите схему оформления рабочего стола:

а) перейдите на третий лист **Оформление** вкладки **Свойства: Экран**. Выберите одну из схем оформления рабочего стола, например **Розовая**. С помощью кнопки **Цвет** раскройте палитру для закраски рабочего стола. Выберите щелчком мыши один из предлагаемых цветов, щелкните по кнопке **Применить**;

б) передвинув окно вкладки **Свойства: Экран** в правый нижний угол, оцените выполненные действия.

1.3.2. Смените схему оформления рабочего стола:

а) раскройте список **Схема** и выберите строку **Обычная Windows**;

б) нажмите на кнопки **Применить** и **ОК**.

1.4. Установите новую папку (каталог).

1.4.1. Создайте на рабочем столе новую папку с присвоением ей имени индекса учебной группы, например Student 1:

а) переведите курсор на свободное место поля экрана, щелкните правой кнопкой мыши;

б) в появившемся контекстном меню выберите опцию **Создать** и, переведя курсор на поле нового контекстного меню, щелкните мышью по строке **Папку**;

в) под значком новой папки установлено поле для записи необходимого имени папки. Щелкните мышью по этому полю и нажмите на клавишу **<BackSpace>**, чтобы его очистить.

г) запишите новое имя папки: индекс группы или Ваше имя. Нажмите на клавишу **<Enter>**.

1.4.2. Перенесите созданную папку на поле меню **Пуск**:

а) выделите щелчком мыши созданную папку;

б) перетащите левой кнопкой мыши изображение папки на область кнопки **Пуск**, отпустите кнопку мыши; в) раскрыв меню **Пуск**, убедитесь в том, что папка находится на его поле.

1.5. Оформите папку документов.

1.5.1. Управление папкой документов:

- а) раскройте новую папку, размещенную на рабочем столе, двойным щелчком левой кнопки мыши;
- б) уберите панель инструментов и адресное окно с помощью строки Вид системного меню;
- в) восстановите панель инструментов и окно адреса на поле окна папки;
- г) перенесите рабочее окно папки в центр рабочего стола, увеличьте размеры рабочего окна папки, захватив мышью край или угол окна;
- д) раскройте окно на весь экран;
- е) восстановите исходное изображение;
- ж) удалите подготовленную папку в **Корзину**.

1.5.2. Изменение содержание меню **Пуск**:

а) удалите строку о подготовленной папке из меню **Пуск**: щелкните по панели задач правой кнопкой мыши, вызовите опцию Свойства и на появившейся вкладке **Свойства** выберите лист **Настройка меню**;

б) в верхнем поле вкладки щелкните по кнопке **Удалить**. В новом диалоге **Удаление ярлыков и папок**, в поле дерева каталога с помощью ползунка найдите строку с названием ранее подготовленной папки и щелкните по ней мышью, затем нажмите на кнопку **Удалить** и щелкните по кнопкам **Закреть** и **ОК**.

1.6. Управление файлами и документами с помощью программы **Проводник**.

1.6.1. Сформируйте новую папку (каталог):

а) с помощью меню **Пуск** найдите программу **Проводник** и запустите ее: переведите курсор на кнопку **Пуск** и щелкните мышью, на первой панели меню найдите папку **Программы** и переведите на нее курсор. Рядом появится вторая панель. На ней найдите строку **Проводник** и щелчком мыши активизируйте это приложение;

б) на диске С: найдите каталог Student (если его нет, то создайте его), переведите курсор на диск С:, в меню **Файл** выберите опцию **Создать**, переведите курсор на правое поле нового меню на строку

Папку, щелкните мышью, в новом окне на правой панели окна приложения запишите имя создаваемой папки (например, Student). Эти же операции можно выполнить щелчком правой кнопки мыши по полю активизированной папки, размещенной в правой области **Проводника**;

в) в папке **Student** сформируйте новую папку с именем индекса студенческой группы, например ИБМ-4 (см. предыдущий пункт задания).

1.6.2. Активизируйте программы с помощью меню кнопки **Пуск**:

а) в меню **Пуск** в папке **Программы** найдите папку **Стандартные**, откуда запустите программу **WordPad** или **Блокнот**. Запишите на поле его рабочего окна тему лабораторной работы и свои фамилию, имя и отчество, сохраните подготовленный текстовый документ под именем 1_1_ФИО: в меню **Файл** выберите опцию **Сохранить как...**, раскройте список **Папка**, выберите каталог С:, найдите папку **Student** и дважды щелкните по ней, затем перейдите на поле папки с именем индекса Вашей группы. Запишите на поле **Имя файла** 1_1_ФИО, а на поле **Тип файла** выберите строку **Текстовый документ**, щелкните по кнопке **Сохранить**. Закройте **WordPad**;

б) перейдите на поле **Проводника**, раскройте ранее созданную папку и убедитесь в наличии подготовленного текстового файла на его поле.

1.6.3. Установите ярлык для документа:

а) создайте ярлык для документа 1_1_ФИО и перенесите его на поле рабочего стола: щелкните правой кнопкой мыши по имени файла, в контекстном меню выберите опцию **Создать ярлык** и щелкните мышью, нажмите на левую кнопку мыши и перетащите ярлык на рабочий стол экрана;

б) замените рисунок ярлыка файла: вызовите контекстное меню, щелкнув правой кнопкой мыши по новому файлу, выделите опцию **Свойства**, перейдите на лист **Ярлык** и щелкните по кнопке **Изменить значок**. Выберите подходящее изображение из стандартного ряда и установите его;

в) отложите работу с **Проводником**: перенесите его обозначение на поле строки панели задач;

г) с помощью подготовленного ярлыка раскройте документ 1_1_ФИО, расположенный на рабочем столе. Убедившись в эффективности выполненных операций, закройте приложение **WordPad**.

2. Работа с дискетами.

Цель работы: овладеть приемами форматирования дискет, перемещения и копирования объектов, удаления и восстановления файлов.

2.1. Отформатируйте дискету:

а) установите дискету в дисковод компьютера;

- б) правой кнопкой мыши вызовите контекстное меню для диска А;
- в) в контекстном меню выберите опцию **Форматировать** и, используя диалоговое меню, проведите полное форматирование дискеты, обратив внимание на установку емкости диска.

2.2. Копируйте файлы:

а) в левой панели **Проводника** (на дереве каталогов) найдите текущую папку с файлом 1_1_ФИО (ее содержание будет показано на правой панели **Проводника**);

б) щелкните по полю имени файла правой кнопкой мыши, выберите в контекстном меню опцию **Копировать** и щелкните левой кнопкой мыши;

в) выберите на дереве каталогов диск А: и щелкните правой кнопкой мыши. В контекстном меню выберите опцию **Вставить**;

г) создайте новую папку на дискете, например Student_2;

д) осуществите операцию копирования файла на дискету другим способом: предварительно на дереве каталога у диска А: щелкните по знаку +, чтобы раскрыть структуру файлов диска А:, переведите курсор мыши на поле копируемого файла, нажмите на правую кнопку мыши и перетащите значок файла на левую панель **Проводника**, совместив изображение с каталогом А:/Student_2. Отпустите кнопку мыши, выберите опцию **Копировать**, щелкните по ней мышью.

2.3. Удалите файлы:

а) на поле имени файла, размещенного в каталоге А:/Student_2, **1_1_ФИО** щелкните правой кнопкой мыши и выберите команду Удалить, в новом диалоге подтвердите намерение;

б) удалите ярлык файла и ярлык новой папки с рабочего стола: выделите мышью и перетащите изображение компонентов в **Корзину**.

2.4. Восстановите файлы:

а) на рабочем столе выделите приложение **Корзина** и активизируйте его двойным щелчком мыши;

б) выделите в списке удаленных тот файл, который следует восстановить. В меню **Файл** активизируйте опцию **Восстановить**;

в) удалите файл с дискеты;

г) повторите операцию копирования, используя другой алгоритм: щелкните по значку копируемого файла правой кнопкой мыши, выберите в контекстном меню опцию **Копировать**, переведите курсор на значок диска А: и щелкните по нему тоже правой кнопкой, в контекстном меню выберите опцию **Вставить**;

д) раскройте диск А: и убедитесь в выполнении операции копирования;

е) перейдите на рабочий каталог;

ж) вытащите дискету из дисковода.

3. Применение антивирусных программ.

Цель работы: овладеть приемами защиты документов от компьютерных вирусов.

3.1. Разверните рабочее окно **Проводника** и найдите антивирусную программу, например **drweb.exe**. Если папка с антивирусной программой неизвестна, то следует открыть меню **Сервис**, выбрать опцию **Поиск** и заполнить соответствующие поля (искать следует в поле всего дискового пространства!).

3.2. Найдите опцию **Лечить**, запишите: А:*. * и запустите программу.

3.3. С помощью меню антивирусной программы завершите операцию.

4. Архивирование файлов.

Цель работы: овладеть приемами архивирования файлов.

4.1. В рабочей папке найдите ранее созданный текстовый файл.



4.2. Вызовите программу архивирования файлов, например **WinRAR**, ярлык которой обычно размещен на рабочем столе.

Если приложение находится внутри файловой системы, то его следует отыскать. Для этого можно воспользоваться **Проводником** или щелкнуть правой кнопкой мыши по кнопке стартового меню **Пуск**. В этом случае появится контекстное меню с командой **Найти**. Далее следует активизировать соответствующую команду и в диалоге поиска задать имя приложения и условия поиска.

4.3. Самостоятельно установите и активизируйте режим архивирования текстового файла, используя кнопку **Добавить** на панели инструментов архиватора (рис. 2.36).

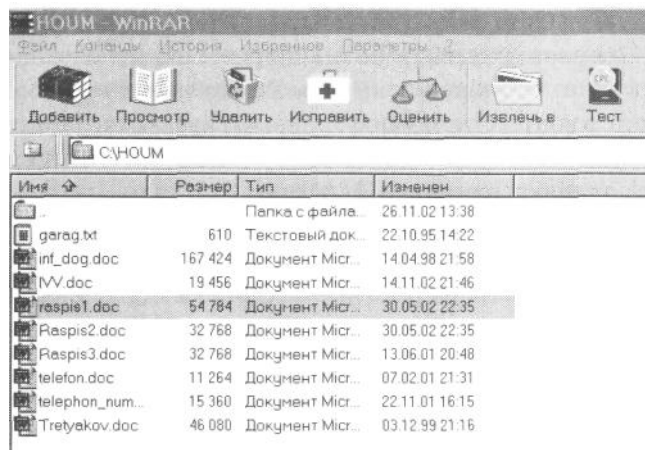


Рисунок 2.36. Рабочее окно приложения WinRAR

В диалоге Имя и параметры архива (рис. 2.37) укажите место размещения архивированного файла, используя кнопку Обзор, размещенную справа от поля Архив. По умолчанию архивный файл будет направлен в папку Мои документы. Щелкните по кнопке ОК.

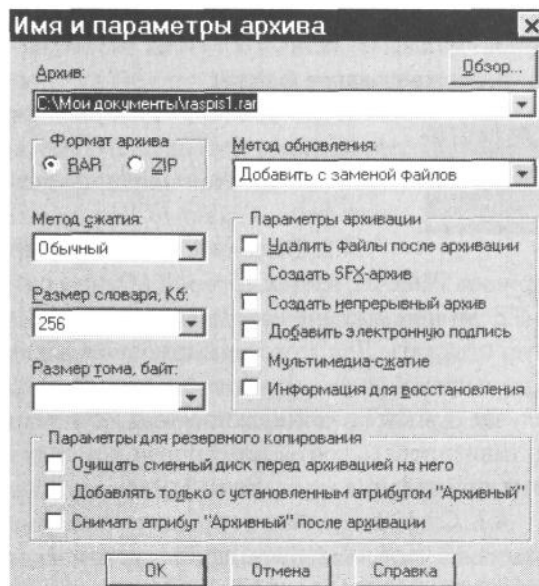


Рисунок 2.37. Диалог установки параметров архивирования и имени файла

На поле индикатора архивирования будет показан процесс архивирования и время течения предполагаемого и фактического процессов архивирования, после окончания которого приложение снова вернется к своему рабочему окну.

Следует оценить объем нового файла.

4.4. Удалите текстовый файл упражнения. Снова вызовите архиватор и разархивируйте файл с текстом упражнения, используя кнопку Извлечь. Если будет указан конкретный архивированный файл, то появятся две кнопки Извлечь и Извлечь в. С помощью первой разархивированный файл помещается в текущую папку, с помощью второй – в новую, указанную пользователем, папку.

4.5. Закройте все приложения (программы и документы).

5. Подготовка рисунков в графической среде операционной системы.

Цель работы: овладеть навыками работы в графическом редакторе Paint: выбор и использование различных инструментов, изображение прямых, окружностей, эллипсов, кривых, прямоугольников, многоугольников, начертание плоских и объемных изображений, раскрашивание, перемещение по экрану, изменение размеров и сохранение фрагментов изображений, нанесение текста на рисунок, использование буфера обмена для переноса изображений на документы других приложений, подготовка рисунков для деловых документов.

5.1. Подготовка простых рисунков.

5.1.1. Вызовите и подготовьте новый рабочий документ:

а) вызовите графический редактор Paint: в меню Пуск выберите папку Программы, с поля которой переведите курсор на строку Стандартные, на следующей панели меню щелкните по строке Paint;

б) сохраните новый документ под именем 2_1_ФИО в рабочей папке: в меню Файл выберите опцию Сохранить как.... В раскрывающемся списке Папка диалогового окна выберите строку C:, в поле рабочего окна диалога найдите рабочую папку Student, двойным щелчком раскройте ее и дважды щелкните по рабочей папке (например, ИБМ-4). В строке Имя файла запишите 2_1_ФИО и щелкните по кнопке Сохранить.

5.1.2. Примените основные инструменты графического редактора:

а) нарисуйте линию. На панели инструментов выберите пиктограмму Линия: щелкните по соответствующей области на панели инструментов, переведите курсор в центр рабочей области, нажмите на левую кнопку мыши и проведите линию (5—6 см) (протащите мышь вправо);

б) выделите область рисунка. Замените инструмент на **Выделение** (щелкните по кнопке в форме пунктирного прямоугольника). Переведите курсор слева над линией и протащите курсор вниз вправо так, чтобы ранее изображенная линия оказалась в поле пунктирного прямоугольника. Отпустите кнопку мыши;

в) удалите выделенный фрагмент. Щелкните по выделенному полю правой кнопкой мыши. В контекстном меню выберите опцию **Очистить выделение**, щелкните мышью (контекстное меню дублирует основные опции меню графического редактора);

г) измените толщину линии. Щелкните по пиктограмме **Линия**. Под панелью инструментов откроется область размеров инструмента. Выберите на ней необходимую толщину линии и нарисуйте прямые линии разной толщины. Очистите рабочую область редактора;

д) изобразите прямые линии. Проведите линию, одновременно нажимая на клавишу <Shift> — если угол наклона небольшой, то получится горизонтальная линия, если угол наклона около 45°, то линия пойдет под углом 45°, дальнейший поворот угла наклона приведет к изображению строго вертикальной линии и т. д. Проведите горизонтальную, вертикальную и наклонную линии;

е) очистите экран. Выберите инструмент **Выделение**, щелкните правой кнопкой мыши по полю экрана и в контекстном меню выберите опцию **Выделить все**, нажмите на клавишу . Повторите эту же операцию, используя меню **Правка**;

ж) изобразите многоугольник. Щелкните по кнопке инструмента **Многоугольник**. Изобразите небольшой треугольник: проведите линию первой стороны треугольника, отпустите кнопку мыши, снова нажмите на нее, проведите вторую сторону, отпустите мышь, снова нажмите и замкните линии фигуры;

з) перенесите изображение. Выделите область треугольника. Переведите курсор на выделенную область (форма курсора изменится на крест в виде стрелок), протащите мышь с нажатой левой кнопкой по полю экрана и отпустите кнопку;

и) копируйте изображение. Выделите область треугольника. Протащите выделенную область при нажатой клавише <Ctrl>;

к) запишите изображение в промежуточную память (карман). Выделите область треугольника, откройте контекстное меню и вызовите опцию **Копировать**. Удалите выделенную область;

л) восстановите изображение из промежуточной памяти. В меню **Правка** выберите команду **Вставить** (эту операцию можно выполнить, используя контекстное меню);

м) поверните изображение. Перенесите восстановленный рисунок в левую часть, а его три копии (одна под другой) — в правую часть рабочего окна. Переведите курсор на верхний правый выделенный рисунок и вызовите контекстное меню. Выберите опцию **Отразить/повернуть**. В появившемся диалоге установите режим **Отразить слева направо**. Выделите следующий правый рисунок и в опции **Отразить/ повернуть** используйте режим **Отразить сверху вниз**. Перейдите на нижний правый рисунок и покажите, как действует режим **Повернуть на угол**;

н) наклоните изображение. Выделите правые рисунки и удалите их. Перенесите две копии левого рисунка в правую область. Вызвав контекстное меню верхней кнопки, выберите опцию **Растянуть/наклонить**. В диалоге выберите поле **Наклонить**, где в строке **По горизонтали** укажите 45°. Оцените изменения и повторите операцию для режима **По вертикали** 45°. Перейдите на нижнюю правую копию рисунка и в диалоге **Растяжение и наклон** установите сначала параметр режима **Растя-**

нуть по горизонтали, равный 150%, а затем измените параметр режима **Растянуть по вертикали** на 150%. Уясните эффект от действия режимов **Растянуть** и **Наклонить**.

5.1.3. Сформируйте простой рисунок:

а) начертите 8-лучевую звезду, используя вертикальную линию: изобразите вертикальную линию (5—6 см) средней толщины, в правой части экрана отобразите две ее копии (в верхнем правом углу и посередине), первую копию поверните на угол 90°, вторую копию поверните на угол 45° и скопируйте ее на свободное место в низ правой части экрана. На третьей копии, используя опцию **Отразить**, получите отраженное изображение наклонной линии. Перетащите три линии на четвертую, изображенную в левой части экрана;

б) уменьшите рисунок. Выделите область звезды. Переведите курсор на границу угла выделенной области и протащите курсор внутрь поля, уменьшая изображение до 2 см;

в) вырежьте рисунок. Выделите область рисунка. В контекстном меню выберите опцию **Вырезать**. При этом рисунок помещается в карман, а рабочее окно очищается;

г) восстановите рисунок, используя меню **Правка** и опцию **Вставить**.

5.1.4. Запишите рисунок в графический файл:

а) изобразите созвездие: захватывая рисунок звезды из левого верхнего угла рабочего окна, копируйте его по полю рисунка в соответствии с замыслом;

б) выделите полученное изображение «звездного неба», активизируйте контекстное меню и выберите опцию **Копировать в файл**;

в) укажите место каталога (папки), имя файла и нажмите на кнопку **Сохранить**.

5.1.5. Раскрасьте изображение:

а) выберите инструмент **Эллипс**. Переведите курсор на поле экрана и протащите мышью по диагонали, повторите операцию с нажатой клавишей <Shift>. В результате будет изображен круг. Очистите экран;

б) нарисуйте три пересекающиеся окружности, «поставленные» на квадрат. Щелкните по инструменту **Заливка**;

в) щелкните мышью по соответствующей краске палитры редактора — установите цвет раскраски;

г) щелкните по полю квадрата;

д) смените инструмент на **Распылитель** и, изменяя размер струи, изобразите три шаровые поверхности на поле ранее нарисованных кругов;

е) очистите рабочее поле редактора и выберите инструмент **Прямоугольник**. Выберите в поле формы (размера) инструмента нижнюю форму: заштрихованный прямоугольник без окантовки;

ж) нанесите прямоугольник на поверхность листа (он не виден);

з) щелкните левой кнопкой мыши по палитре редактора. Поле прямоугольника изменит свой цвет: левая кнопка мыши изменяет цвет линий и заливки, а правая кнопка мыши — цвет бумаги (подложки).

5.1.6. Запишите текст на поле рисунка:

а) организуйте подрисовочную подпись: щелкните по инструменту **Надпись**, переведите курсор на свободное поле рабочего окна и протащите курсор по диагонали (в результате будет выделено поле для записи текста);

б) вызовите на экран панель форматирования текста (если ее нет): в меню Вид выберите опцию **Панель атрибутов текста**;

в) выберите подходящий кегль (размер шрифта), начертание (шрифт) и способ выделения текста (полуужирный, курсив, подчеркивание);

г) щелкните мышью в выделенной области и запишите необходимый текст, например **Учебный рисунок**, щелкните мышью;

д) вызовите инструмент Выделение, выделите область текста и перетащите текст под рисунок.

5.1.7. Обрамите рисунок:

а) с помощью инструмента **Прямоугольник** установите рамку рисунка;

б) организуйте двойную рамку для рисунка, используя линии разной толщины;

в) запомните подготовленное изображение под именем **2_2_ФИО**.

5.2. Подготовка сложного рисунка.

В качестве сложного рисунка возьмем рисунок товарного знака завода спортивных изделий. Для этого используем изображение пары гантелей весом 3 кг, изготовленных на заводе «Спорт» (рис. 2.38).



Рисунок 2.38. Эскиз товарного знака

5.2.1. Вызовите новый документ:

а) в меню **Файл** выберите команду **Сохранить как...**;

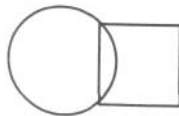
б) в соответствующее поле диалогового окна запишите имя создаваемого файла и нажмите кнопку

Сохранить.

5.2.2. Подготовьте контур будущего рисунка:

а) выберите инструмент **Линия**, установите вторую сверху толщину линии инструмента и щелкните мышью;

б) выберите инструмент **Эллипс**, установите первую из списка форм: прямоугольник без подложки и нарисуйте на левой части рабочей области круг диаметром 5 см;

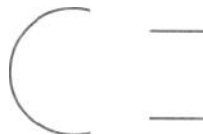


в) выберите инструмент **Прямоугольник**, переведите курсор на правую часть окружности, на 1 см ниже осевой линии;

г) нажмите кнопку мыши и тащите ее сначала вверх до пересечения с окружностью, а затем вправо на 3 см, отпустите кнопку мыши;

д) изобразите линию сопряжения шара с цилиндром. Для этого: выберите инструмент **Масштаб**, переведите курсор на область сопряжения окружности и прямоугольника, щелкните мышью;

е) выберите инструмент **Ластик** и удалите линии окружности и прямоугольника в области сопряжения, сотрите левую и правую вертикальные линии прямоугольника;



ж) выберите инструмент **Кривая**, переведите курсор на точку обрыва окружности, протащите мышью к точке обрыва прямоугольника и отпустите кнопку мыши, переведите курсор на тонкую линию в точку предполагаемого изгиба и, нажав на кнопку мыши, оттяните линию во внутреннюю область рисунка на 3—5 мм, отпустите кнопку и щелкните мышью;



з) повторите предыдущую операцию для другой пары точек разрыва между окружностью и прямоугольником;

и) выделите нарисованную фигуру и перенесите ее копию в правую область рабочего окна;

к) поверните копию рисунка на 180° и совместите правую часть рисунка с левой, используя инструмент **Выделить**;

л) изобразите выпуклую ручку у гантели: выделите область рукоятки (между точками сопряжения) и удалите ее, с помощью инструмента **Кривая** изобразите утолщения на ручке гантели;

м) покажите две гантели на рисунке: одну за другой, удалите невидимые линии второй гантели;

н) выберите инструмент **Распылитель** и нанесите краску по краям рисунка, создавая ощущение его объемности, изобразите тень от предмета;

о) покажите вес гантели: на свободном поле рисунка напишите: 3 кг. Текст выделите и скопируйте на шаровые поверхности гантели;

п) запишите логотип завода-изготовителя и нанесите его на рукоятки гантелей: ООО Спорт™ и покажите зарегистрированные права на это изделие ®: отдельно нарисуйте круг и напишите букву R, затем перенесите одно изображение на другое и, выделив полученное, перенесите к тексту «Спорт». Полученную надпись установите сверху рисунка;

р) установите рамку на общий рисунок.

5.3. Осуществите обмен документами между различными приложениями офисной системы.

5.3.1. Выделите область рисунка.

5.3.2. Копируйте рисунок в карман.

5.3.3. Вызовите текстовый редактор, откройте новый документ (файл) и вставьте на его поле из кармана подготовленный рисунок.

5.3.4. Запишите поясняющий текст к рекламному рисунку, сохраните текстовый файл как 2_3_ФИО.

5.4. Изобразите рекламный рисунок на изделие фирмы.

Фирма может производить различные товары, например компьютеры, радиолокационное оборудование, литые, холодильные установки, двигатели, спортивные товары, галантерею и т. д. На рисунке следует отобразить основной профиль деятельности фирмы, установить зарегистрированный логотип фирмы.

5.4.1. Используя инструменты графического редактора, изобразите подробный контур основной выпускаемой продукции (например, профиль автомобиля, самолета, ракеты, компьютера и т. п.).

5.4.2. Раскрасьте рисунок, используя различные краски для фрагментов рисунка продукции (например, кузова, колес и аксессуаров автомобиля).

5.4.3. Разработайте фоновый рисунок.

5.4.4. Поместите на поле рисунка (сверху справа) имя фирмы и знак регистрации товарного знака фирмы.

5.4.5. Выделите непосредственно поле рисунка, скопируйте рисунок в файл (в личную папку каталога **Student**) и в буфер обмена.

5.4.6. Перейдите в текстовый редактор (например, **Word** или **WordPad**) и создайте текстовый файл рекламы изделий фирмы для помещения ее в печатное издание (газета, журнал, рекламный листок и т. п.), укажите реквизиты фирмы.

5.4.7. Перенесите в этот файл подготовленный рисунок и закрепите его в тексте рекламы.

5.4.8. Сохраните файл подготовленного документа в личной папке каталога **Student**.

5.4.9. Перенесите подготовленные документы на личную дискету, проверив ее на наличие компьютерных вирусов.

5.4.10. Закройте все приложения (графический и текстовый редакторы).

5.4.11. Вытащите из дисковода личную дискету.

ГЛАВА 3. ОСНОВЫ ПРОГРАММИРОВАНИЯ НА ТУРБО ПАСКАЛЬ 7.0

3.1. Процесс программирования

Программирование в большинстве случаев определяется как процесс подготовки программ для их решения (применения) на компьютере, состоящий из следующих этапов: составление алгоритма задачи, т. е. составление «плана решения» задачи в виде набора операций, описание «плана решения» на языке программирования (создание программы), трансляция программы на машинный язык. Последний этап программирования в большинстве своем связан с процессом отладки программы, внесением в текст программы необходимых коррективов и т. п., что определяется как доводка программы.

Обратим внимание на первую часть процесса программирования - алгоритмизацию задачи. Сначала определим, что термин «алгоритм» определяет совокупность команд (инструкций), позволяющих четко и однозначно определить процесс выполнения задачи.

Алгоритм можно описать в форме отдельных текстовых инструкций (приказов), в виде последовательного набора отдельных операций, кодов команд и т. п., однако наиболее зримым алгоритм становится в том случае, когда он показан в форме взаимосвязанных графических элементов, обозначающих, например, процессы ввода данных, вычисления значений, определения условий

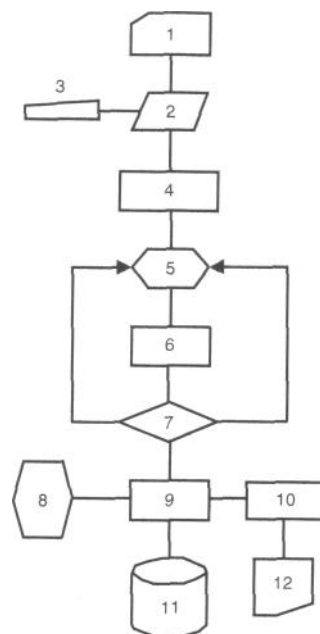
продолжения выполнения вычислений, вывода результатов и т. п., пример начертания которых показан на рис. 3.1.



Рисунок 3.1. Обозначение графических элементов, используемых для составления графической модели алгоритма

На поле графических элементов указывают основные операции, которые должны быть выполнены, например ввод начальных условий (с клавиатуры), ввод значений массива A (с магнитного диска или ленты), вычисление параметра C , определение условия продолжения вычислений, вывод результатов на печать (экран) и т. д.

На рис. 3.2 показан фрагмент построения алгоритма решения некоторой вычислительной задачи, которая предполагает вычислить некоторый массив данных по результатам анализа вводимой с клавиатуры информации, используя при этом ряд коэффициентов, которые «защиты» в тело программы.



1 — запуск программы и ввод значений констант; 2 — запрос на ввод исходных данных (параметров); 3 — ввод исходных данных с клавиатуры; 4 — выполнение необходимых операций вычисления; 5 — вычисление отдельных значений с помощью повторяющихся операций (цикла); 6 — операции продолжения обработки информации; 7 — проверка условий вычисления (условие К); 8 — выдача текущей информации на экран; 9 — формирование данных для хранения; 10 — подготовка данных для выдачи на бумажный носитель (твёрдая копия); 11 — запись подготовленного массива данных на диск; 12 — текстовый документ

Рисунок 3.2. Графическая модель алгоритма

Последовательность выполнения алгоритма можно пояснить следующим образом. При запуске приложения устанавливаются значения констант и начальные значения переменных. После этого система запрашивает с клавиатуры исходные данные для проведения вычислений. Затем она выполняет первый блок вычислений и переходит к циклической обработке, например к вычислению матрицы. На этом этапе программа поочередно изменяет индексы у элементов матрицы, повторяя цикл вычисления постоянное число раз (n). На следующем этапе последовательные операции обработки информации продолжаются. Затем программа переходит к выбору дальнейшего пути решения задачи, используя операцию сравнения значения текущей переменной с некоторой константой K , которая была определена либо при вводе данных с клавиатуры, либо вычислена в процессе выполнения программы, либо задана изначально при запуске программы.

В процессе анализа программа определяет необходимость продолжения вычисления по условию либо перехода к записи информации на диск с выдачей информации на экран, либо к выдаче данных на твердый носитель (бумагу).

В приведенном примере в качестве элементов взаимосвязи между отдельными блоками программы использовались линии связи. Обычно они не имеют стрелок, так как алгоритм разворачивается сверху вниз по полю бумаги. Однако чтобы подчеркнуть цикличность процесса, применяют стрелки. Например, при формировании циклических процедур. Также, чтобы подчеркнуть ввод массива данных, для начертания процесса используют объемные стрелки.

3.2. Интегрированная инструментальная оболочка Турбо Паскаля

Вызов системы Турбо Паскаль 7.0 осуществляется с помощью команды **Turbo** из соответствующего каталога (папки). После этой команды на верхней строке экрана терминала отображается главное меню. Меню состоит из опций — вариантов действий. Их десять: **File, Edit, Search, Run, Compile, Debug, Tools, Options, Window** и **Help**.

Обращение к опциям меню осуществляется с помощью клавиши **<F10>** или с помощью одновременного нажатия на клавиши **<Alt>** и подчеркнутой в команде буквы. Отказ от выполнения команд меню происходит в результате нажатия клавиши **<Esc>**.

Для работы с выбранной опцией достаточно перевести на ее обозначение курсор и нажать клавишу **<Enter>**.

3.2.1. Опции главного меню

File — управляет файлами (выбор, загрузка, запись на диск, открытие нового файла, вывод на печать и др.), выход из программы, временный выход в оболочку DOS.

File|New

Открытие нового окна редактирования и нового файла с именем Noname <цифра><цифра>.pas.

File|Open

Формирование в диалоговом окне экрана списка текущей директории. Используя клавишу **Open**, выбранный файл загружается во вновь открываемое окно. При выборе клавиши **Replace** файл загружается в активное окно редактирования.

File|Save

Запись файла из активного окна на диск.

File|Save as

Запись файла из активного окна на диск под другим именем.

File|Save all

Запись на диск всех измененных файлов из открытых окон.

File|Change dir

Изменение текущего каталога с помощью выбора нужного каталога на отображенном дереве каталогов или указания полного адреса нового каталога.

File|Print

Вывод на печать файла из активного окна редактирования.

File|Printer setup

Обработка текста файла перед выводом его на печать, например для выделения заголовков, отдельных слов и т. п.

File|Dos shell

Временный выход в операционную систему. Для возврата в среду Турбо Паскаля необходимо использовать команду **File|Exit**.

File|Exit

Выход из среды Турбо Паскаля и удаление ее из оперативной памяти компьютера.

Edit — позволяет создавать и редактировать исходные файлы: изменять текст, выделять и перемещать фрагменты текста, создавать карман для фрагментов текста и др.

Edit|Undo

Отмена всех изменений, внесенных в процессе предыдущего редактирования.

Edit|Redo

Отмена действий, выполненных с помощью команды **Edit|Undo**.

Edit|Cut

Перемещение выделенного фрагмента текста в карман. Выделенный фрагмент удаляется из текста.

Edit|Copy

Копирование выделенного фрагмента текста в карман.

Edit|Paste

Перемещение текста из кармана или выделенного фрагмента текста в окне **Clipboard** в то место активного окна, которое отмечено курсором.

Edit|Clear

Удаление выделенного фрагмента и очистка кармана.

Edit|Show clipboard

Открытие окна кармана.

Search — организует поиск информации в редактируемом тексте программы на основе заданного фрагмента текста.

Search|Find

Поиск информации в тексте программы по задаваемой последовательности символов, условиям начертания знаков, направлению и области поиска.

Search|Replace

Замена в тексте файла одной последовательности символов на другую.

Search|Search again

Просмотр условий ранее проведенного поиска.

Search|Go to line number

Поиск фрагмента текста по заданному номеру строки в тексте программы.

Search|Show last compiler error

Установка курсора на строку с ошибкой, обнаруженной компилятором программы.

Search|Find error

Определение места ошибки, обнаруженной в процессе выполнения программы.

Search|Find procedure

Поиск процедуры или функции, используемой в программе.

Search|Previous browser

Восстановление содержания закрытого последним окна браузера.

Браузер — программное средство, позволяющее находить информацию об используемых в программе именах объектов (**Objects**), глобальных символах программы или модуля (**Globals**), именах всех модулей, используемых программой (**Units**), глобальном символе, помеченном курсором в основной программе (**Symbol**).

Run — проводит компиляцию, компоновку и выполнение программы.

Run|Run

Компиляция, компоновка и выполнение программы. Нажатие клавишей <Ctrl>+<Break> приостанавливает ход выполнения программы. Повторное нажатие завершает выполнение программы.

Run|Step over

Пошаговое выполнение программы без сканирования содержания процедур и функций.

Run|Trace into

Пошаговое выполнение программы. При использовании подпрограмм рассматриваются операторы выполняемых процедур и функций.

Run|Go to cursor

Компиляция и выполнение программы до строки в программе, на которой установлен курсор.

Run|Program reset

Прекращение текущего сеанса отладки. Закрываются все файлы, используемые программой, и освобождается память, отведенная под программу.

Run|Parameters

Возможность задать строку символов, определяющую входные параметры программы, аналогичные параметрам командной строки.

Compile — компилирует и преобразует исходный код в откомпилированный модуль и выполняемый файл, находит ошибки выполнения, выдает системную информацию.

Compile|Compile

Компиляция программы или модуля, находящегося в активном окне редактирования.

Compile|Make

Компиляция только тех программ и модулей, в тексты которых вносились изменения.

Compile|Build

Перекомпиляция программы и всех модулей независимо от времени их редактирования.

Compile|Destination

Определение места размещения программного кода: в оперативной памяти — Memory, на диске — Disk.

Compile|Primary file

Указание имени файла с расширением .pas, который будет обрабатываться в режимах **Run**, **Make**, **Build**.

Compile|Clear primary file

Очищение всех опций **Primary file**.

Compile|Information

Информация о результатах компиляции.

Debug — организует режим отладки программы.

Debug| Breakpoints

Выдача информации о всех точках останова в программе: номер строки, условие и регулярность срабатывания.

Debug|Call stack

При временной остановке хода выполнения программы показ стека, на дне которого находится имя главной программы, а на вершине — имя подпрограммы, получившей управление в последний момент перед остановкой.

Debug|Register

Определение состояния регистров центрального процессора компьютера.

Debug|Watch

Активизация окна наблюдения.

Debug|Output

Активизация окна вывода, в котором размещается запрашиваемая у пользователя и выводимая программой информация.

Debug|User screen

Просмотр результатов работы программы не в окне, а на полном экране.

Debug|Evaluate/modify

Просмотр значений любой переменной программы.

Debug|Add watch

Добавление в окно наблюдений нового выражения из программы. Значение выражения вызывается нажатием клавиши <F6>.

Debug|Add breakpoint

Определение новой точки останова в программе.

Tools — формирует оперативный доступ к внешним программам из среды программирования Турбо Паскаля.

Tools|Messages

Показ сообщений внешних программ-фильтров о содержании файлов.

Tools|Go to next

Переход из окна Messages к следующему выбранному сообщению анализируемого файла в окне редактирования.

Tools|Go to previous

Переход из окна Messages к предыдущему сообщению анализируемого в окне редактирования файла.

Options — управляет режимами компиляции и компоновки программ.

Options|Compiler

Открытие диалогового окна, с помощью которого можно установить параметры, определяющие работу компилятора, аналогично директивам компилятора.

Options|Memory sizes

Открытие диалогового окна, с помощью которого задаются размеры используемой оперативной памяти.

Options|Linker

Открытие диалогового окна, с помощью которого можно установить параметры, определяющие работу компоновщика.

Options|Debugger

Открытие диалогового окна, с помощью которого можно установить параметры, определяющие работу отладчика.

Options|Directories

Возможность задать каталоги, из которых извлекаются или в которые помещаются файлы.

Options|Tools

Открытие диалогового окна, с помощью которого можно включать в список вызываемых из среды программ новую программу, отредактировать ее параметры, удалить программу из списка.

Options|Environment

Возможность задать условия работы в среде.

Options|Open

Открытие файла конфигурации, в котором сохраняются установленные параметры интегрированной среды.

Options|Save

Сохранение конфигурации среды в файле, открытой командой меню **Options|Open**.

Options|Save as

Выведение диалогового окна, с помощью которого выбирается имя файла конфигурации, отличного от заданного командой меню **Options|Open**, в который затем записываются установленные параметры интегрированной среды.

Window — открывает и определяет положение на экране окна для редактирования, наблюдения, вывода и помощи.

Window|Tile

Размещение на экране всех открытых окон. Их размеры одинаковы и они не перекрывают друг друга.

Window|Cascade

Расположение всех открытых окон друг за другом, оставляя для обозрения заголовки окон.

Window|Close all

Закрытие всех открытых окон.

Window|Refresh display

Восстановление содержимого экрана после просмотра результатов работы программы.

Window|Size/move

Изменение места расположения окна на экране с помощью клавиш-стрелок или мыши.

Window|Zoom

Увеличение размеров окна или их уменьшение до исходных размеров.

Window|Next

Активизация следующего по порядку окна.

Window|Previous

Активизация окна, которое было открыто непосредственно перед текущим окном.

Window|Close

Закрытие текущего окна.

Window|List

Вывод на экран списка всех открытых окон.

Help — выдает справочную информацию о языке Турбо Паскаль и его интегрированной среде. Вызов **Help** осуществляется также нажатием клавиши <F1>, а справку об интересующем слове — <Ctrl>+<F1>, предварительно переместив на это слово курсор.

Help|Contents

Вывод в диалоговое окно системы справочной информации, снабженной оглавлением.

Help|Index

Поиск справки по списку ключевых слов.

Help|Topic search

Справка по лексике и грамматике языка программирования. То же — клавиши <Ctrl>+<F1>.

Help|Previous topic

Восстановление предыдущего окна помощи.

Help|Using help

Инструкции по использованию окна **Help**.

Help|Files

Добавление и исключение файлов со справочной информацией.

Help|Compiler directives

Вывод перечня директив компилятора.

Help|Reserved words

Показ информации о зарезервированных словах языка.

Help|Standart units

Вывод перечней стандартных модулей.

Help|Turbo pascal language

Передача информации о синтаксических элементах языка.

Help|Error messages

Вывод сведений о системе сообщений об ошибках.

Help|About

Вывод окна с информацией о версии пакета и авторском праве на него.

3.2.2. Назначение функциональных клавиш

Назначение функциональных клавиш для текущего состояния среды:

<F1> — Help (помощь) — вызов на экран вспомогательной информации, связанной с текущим состоянием системы;

<F2> — File|Save, сохранение на диске файла, находящегося в окне редактирования;

<F3> — File|Open, открытие файла;

<F4> — Run|Go to cursor, выполнение команды до строки, в которой находится курсор;

<F5> — Window|Zoom, увеличение (или уменьшение) активного окна;

<F6> — Window|Next, активизация следующего окна;

<F7> — Run|Trace into, пошаговое выполнение редактируемой программы с заходом в подпрограммы;

<F8> — Run|Step over, пошаговое выполнение редактируемой программы без захода в подпрограммы;

<F9> — Compile|Make, компиляция программы, находящейся в окне редактирования;

<F10> — Menu — переход к меню.

Если нажать и некоторое время подержать клавишу <Alt>, в нижней строке будут перечисляться те функции, которые можно выполнить при ее комбинировании с другими клавишами:

<Alt>+<F1> — Help|Previous topic, восстановление предыдущего окна помощи;

<Alt>+<F3> — Window|Close, закрытие активного окна;

<Alt>+<F5> — Debug|User screen (сохраненный экран), активизация окна пользователя;

<Alt>+<F6> — Window|Previous, активизация ранее активного окна;

<Alt>+<F7> — Tools|Go to previous, переход к предыдущей строке окна сообщений;

<Alt>+<F8> — Tools|Go to next, переход к следующей строке окна сообщений;

<Alt>+<F9> — Compile|Compile, компиляция файла, находящегося в активном окне редактирования;

<Alt>+<X> — File|Exit, выход из среды системы;

<Alt>+<O> — Window|List, вывод на экран списка всех открытых окон;

<Alt>+<S>+<F> — Search|Find, поиск заданного текста в файле;

<Alt>+<S>+<R> — Search|Replace, поиск заданного текста и его замена.

При использовании комбинаций с клавишей <Ctrl> можно вызвать следующие команды:

<Ctrl>+<F1> — Help|Topic search, справка о языковой конструкции;

<Ctrl>+<F2> — Run|Program reset, завершение отладки программы;

<Ctrl>+<F3> — Debug|Call/stack, вывод на экран имен активных блоков;

<Ctrl>+<F4> — Debug|Evaluate/modify, просмотр значения выражения, изменение значения переменной;

<Ctrl>+<F5> — Window|Size/move, изменение размера и положения активного окна;

<Ctrl>+<F6> — Debug|Add/watch, активизация окна наблюдения;

<Ctrl>+<F7> — Debug|Watch, добавление выражения в окно наблюдения;

<Ctrl>+<F9> — Run|Run, запуск программы;

<Ctrl>T<Ins> — Edit|Copy, помещение выделенного текста в карман;

<Ctrl>+ — Edit|Clear, удаление выделенного текста.

Комбинации с клавишей <Shift> вызывают команды:

<Shift>+ — Edit|Cut, удаление выделенного текста из файла и помещение его в карман;

<Shift>+<Ins> — Edit|Paste, перемещение текста из кармана в активное окно;

<Shift>+<F1> — Help|Index, введение словаря справочной системы;

<Shift>+<F6> — Window|Previous, активизация предыдущего окна.

3.2.3. Текстовый редактор

Размер листа: 126 символов x 4535 строк. Клавиши:

<PgUp>, <PgDn> — перемещение окна на страницу;

<Home> — курсор в начало строки;

<End> — курсор в конец строки;

<Ctrl>+<PgUp> — курсор в начало текста;

<Ctrl>+<PgDn> — курсор в конец текста;

 - удаление текущего символа;

<Ctrl>+<Y>- удаление текущей строки;

<Ctrl>+<Q>+<L> — восстановление текущей строки;

<Ctrl>+<K>+ — пометка начала блока;

<Ctrl>+<K>+<K> — пометка конца блока;
<Ctrl>+<K>+<Y> — стирание блока;
<Ctrl>+<K>+<C> — копирование блока;
<Ctrl>+<K>+<V> — перемещение блока;
<Ctrl>+<K>+<W> — запись блока на диск;
<Ctrl>+<O>+<I> — автоотступ (отказ, восстановление).

3.2.4. Операции

Арифметические операции:

* — умножение;

/ — деление;

+ — сложение;

- — вычитание;

Div — целочисленное деление;

Mod — остаток от деления;

Shl — левый сдвиг: *i Shl j* (*i* на *j* влево);

Shr — правый сдвиг: *i Shr j* (*i* на *j* вправо).

Логические операции:

Not — логическое НЕ;

Or — логическое ИЛИ;

Xor — логическое ИЛИ-НЕ (исключающее ИЛИ);

And — логическое И.

Операции отношения:

= — равно;

<> — не равно;

< — меньше;

> — больше;

<= — меньше, равно;

>= — больше, равно;

In — принадлежность к множеству.

Пример 1. Выведение результата выполнения операций над двумя целыми числами.

```
Program 03_1;  
Var  
  n,m: Integer;  
Begin  
  Write('Введите два целых числа: '); ReadLn(n,m);  
  WriteLn('n + m = ',n + m);  
  WriteLn('n*m = ',n*m);  
  WriteLn('n/m = ',n/m,' ',n/m:5:2);  
  WriteLn('n Div m = ',n Div m);  
  WriteLn('n Mod m = ',n Mod m);  
  WriteLn;  
  WriteLn('Not n = ',Not n);  
  WriteLn('Not m = ',Not m);  
  WriteLn('n And m = ',n And m);  
  WriteLn('n Or m = ',n Or m);  
  WriteLn('n Xor m = ',n Xor m);  
  WriteLn;  
  WriteLn('n Shl m = ',n Shl m);  
  WriteLn('n Shr m = ',n Shr m);  
End.
```

3.3. Язык программирования Турбо Паскаль 7.0

Любой язык характеризуется лексикой и грамматикой.

3.3.1. Лексика языка

Лексика языка (от греч. *lexikos*) — словарный состав языка программирования. Турбо Паскаль оперирует следующими лексическими единицами: символами, словами, именами, числами, строками, данными, стандартными функциями, выражениями и др.

Алфавит: буквы; цифры (арабские); специальные символы: +, —, /, \, =, >, < и т. д.; служебные слова: And, Or, Array, Begin, End и др.

Слова — совокупность символов, имеющая самостоятельный смысл.

Имена (идентификаторы) — применяются для обозначения констант, переменных, границ, процедур, файлов. Ограничения: начинаются с буквы; внутри нет пробелов; используют только буквы, цифры и знак «подчеркивание»; длина ≤ 63 символа; не используют стандартные имена, зарезервированные для обозначения служебных слов и функций: Abs, Arctan, Char, Exp и др.

Например: x, summa_5, LR1 .

Числа — различают целые, действительные и константы.

Целые числа выбирают из диапазона от -32768 до $+32767$.

Пример 2. Написание программы определения максимального целого числа, которым может оперировать компьютер.

```
Program 03_2;  
Begin  
  Write('Максимальное целое = ',MaxInt);  
End.
```

Действительные числа изменяются в диапазоне: $2.9E - 39$ до $1.7E + 38$, где E — десятичное основание порядка числа: $2.9E - 39 = 2,9 \times 10^{-39}$. Например: $-8.3E-2$, $7E3$.

Пример 3. Введение двух чисел с клавиатуры и выдача частного от деления первого числа на второе.

```
Program 03_3;  
Var  
  n1, n2: Integer;  
  x: Real;  
Begin  
  WriteLn;  
  Write('N1 = '); ReadLn(n1);  
  Write('N2 = '); ReadLn(n2);  
  x := n1/n2;  
  WriteLn('N1/N2 = ',x);  
End.
```

Строки — последовательность символов языка, заключенная в апострофы, например: 'result', 'x = '.

Пример 4. Введение последовательности символов с клавиатуры и выдача ее на экран.

```
Program 03_4;  
Var  
  L: String;  
Begin  
  WriteLn;  
  Write('Введите строку: '); ReadLn(L);  
  WriteLn('Введенная строка => ',L);  
End.
```

Данные — стандартные типы данных, которые определяют форму их представления в компьютере. К основным типам данных относят:

Integer — целый;

Real — действительный;

Boolean — логический (булевский), принимает значения 1 или 0;

Char — символьный: символ запоминается в виде кода и др.

Данные различают на константы и переменные.

Константы — неизменяемые значения данных любого типа.

Переменные — изменяемые значения данных любого типа:

- ◆ простые переменные: tabl, sumnia, ...

- ◆ переменные с индексами: A[5], B[i,j], ...
- ◆ строковая переменная: String[n], где n <= 255.

Стандартная функция — совокупность команд, имеющая определенное имя и выполняющая соответствующую функцию. Пример: Abs(x), Sqr(x), Sin(x), ...

Выражение — последовательность выполняемых операций, разделенных знаками операций: +, -, *, /, Div, Mod, Not, Or, And, <, >, <=, >=, In, порядок применения которых показан в табл. 3.1.

Пример 5. Определение знаков 4-значного числа, их произведения, суммы первых и последних знаков числа.

```

Program 03_5;
Uses Crt;
Var a,b,b1,c,c1,d,p,x,s12,s34: Integer;
    m: Char;
Begin
  Repeat
    ClrScr;
    Write('x = '); ReadLn(x);
    a := x Div 1000;
    b1 := (x Div 100);
    b := (b1 Mod 10);
    c1 := (x Mod 100)*100;

    c := c1 Div 1000;
    d := (x Mod 10);
    WriteLn('a = ',a);
    WriteLn('b = ',b);
    WriteLn('c = ',c);
    WriteLn('d = ',d);
    p := a * b * c * d;
    s12 := a + b;
    s34 := c + d;
    WriteLn('Произведение чисел равно: ',p:5);
    If s12 = s34 Then
      WriteLn('Сумма двух первых чисел равна сумме двух последних чисел и равна: ',s12:5)
    Else
      Begin
        WriteLn('Сумма двух первых чисел не равна сумме двух последних чисел');
        WriteLn('s12 = ',s12:5,' s34 = ',s34:5);
      End;
    Write('Ввести новое число?(y=да, n=нет) '); ReadLn(m);
  Until m = 'n';
End.

```

Таблица 3.1

Порядок выполнения операций

Приоритет	Операция	Обозначение операции
1	Отрицание	Not
2	Тип умножения	*, /, Div, Mod, And
3	Тип сложения	+, -, Or
4	Отношения	=, <>, <=, <, >, >=, In

При изменении последовательности выполнения операций следует применять скобки!

Например: (x = y) And (y = z).

Возведение в степень: если $y = a^n$, то, прологарифмировав, получим: $\ln(y) = n \cdot \ln(a)$; обратное преобразование можно осуществить следующим образом: $\exp(\ln(y)) = \exp(n \cdot \ln(a))$, что при $a > 0$ можно преобразовать к виду: $y = \exp(n \cdot \ln(a))$.

Пример 6. Написание программы вычисления площади треугольника по его сторонам.

```

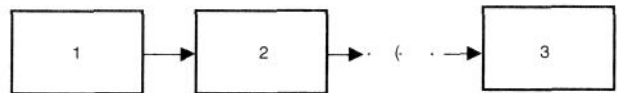
Program 03_6;
Label 1,2;
Var
  a,b,c,p,s: Real;
Begin
  WriteLn;
  WriteLn('Введите исходные данные: ');
  ReadLn(a,b,c);
  If (a+b<c) Or (a+c<b) Or (b+c<a) Then Goto 1;
  p := (a+b+c)/2;
  s := sqrt(p*(p-a)*(p-b)*(p-c));
  Write('Площадь треугольника: ');
  WriteLn('S = ',s);
  ReadLn;
  Goto 2;
1: WriteLn('Исправьте исходные данные');
2: End.

```

3.3.2. Грамматика языка

Грамматика языка (от греч. *grammatike*) — система языковых форм, синтаксических конструкций, образующих основу для построения множества текстов программ на соответствующем языке программирования. В грамматику входит синтаксис языка — структура программы.

Структура программы показана на рис. 3.3. Она содержит заглавие; раздел описаний; «(» — операторную скобку Begin, раздел операторов; «)» — операторную скобку End.



1 — заглавие; 2 — раздел описаний; 3 — раздел операторов

Рисунок 3.3. Структура программы

Заглавие содержит служебное слово **Program** и имя программы. Формат: Program <имя программы>; . Имя программы содержит не более 8 знаков языка Паскаль. Например: Program Ex_1; Program L05_3; .

Раздел описаний содержит:

- ◆ описание используемых процедур и функций — Uses Crt, Graph;
- ◆ описание меток — Label 1,2,3;
- ◆ перечень констант — Const L=80; Pr=124;
- ◆ описание типов переменных:

Var

```

x,y: Real;
Ml, index: Integer;
a,b,c: Char;
flag: Boolean;
st: String;
text: String[25];
Xa,Ya: Word;

```

Описать — означает указать имя и тип переменной!

- ◆ описание подпрограмм пользователя.

Раздел операторов имеет следующую структуру:

Begin

<Последовательность выполняемых операторов>;

End.

Комментарий в тексте программы

Комментарий представляет собой произвольную последовательность любых символов, обрамленную ограничителями: {...} или (* ... *).

Комментарии с одностипными ограничителями нельзя вкладывать друг в друга!

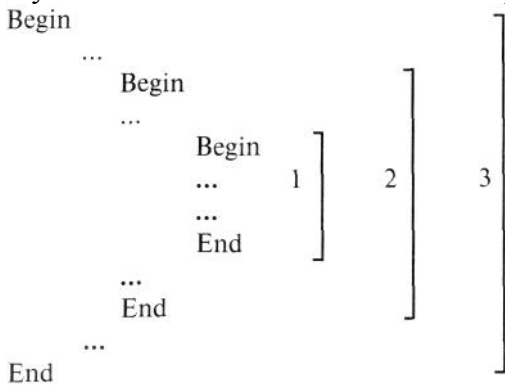
Недопустимо: {...{...},...} или (*...(*...*)...*).

3.4. Операторы

Операторы классифицируются следующим образом:

- а) операторы ввода и вывода данных;
- б) простые операторы:
 - ◆ присваивания :=;
 - ◆ перехода Goto [метка];
 - ◆ процедуры InitGraph(d,r,'d:\turbo');
 - ◆ пустой;
- в) структурированные операторы;
- г) составной оператор — последовательность операторов, заключенная в операторные скобки: Begin ... End.

Глубина вложенности составных операторов:



3.4.1. Операторы ввода и вывода данных

Ввод и вывод данных реализуется с помощью следующих операторов:

```
Read(a1,a2,...,aN);
ReadLn(a1,a2,...,aN);
ReadLn;
```

где a1, a2, a3, ... , aN — имена переменных.

Например, если при использовании строки Read(a,b,c) необходимо ввести числа a = 10, b = -15, c = 24, то следует набрать на клавиатуре: 10 -15 24 и нажать клавишу <Enter>. В этом случае при вводе значения переменных разделяют знаком «пробел»! При записи строки ReadLn(a,b,c) следует после набора каждого числа нажимать на клавишу <Enter>:

- ◆ 10[<Enter>]
- ◆ -15[<Enter>]
- ◆ 24[<Enter>].

При вводе значения переменных разделяют знаком «ввод»!

Оператор ReadLn; обеспечивает режим ожидания нажатием клавиши <Enter>. Его используют для приостановки процесса выполнения программы.

```
Например: ReadLn(a);
          ReadLn;
          ReadLn(b); .
```

Для вывода данных применяют операторы:

Write(a1,a2,...,aN); — вывод информации одной строкой;

WriteLn(a1,a2,...,aN); — переход на новую строку после выдачи информации;

WriteLn; — пропуск строки,

где a1, a2, ... , aN — имена переменных.

Ширина поля вывода:

- ◆ если выводится действительное число (Real), то на бумаге будет напечатано: 'знак мантииссы 'мантисса'(mE) 'знак порядка' порядок (p);
- ◆ если выводится целое число (Integer), то напечатано целое число;
- ◆ если логическая переменная (Boolean), то текст: True или False;
- ◆ если знак (Char), то символ на одной позиции строки.

Форма выводимого выражения (переменной) может быть определена программистом, тогда в операторе Write следует указать:

a: MinWidth: DecPlaces,

где a — переменная, MinWidth — минимальная ширина поля знаков, DecPlaces — количество десятичных знаков в дробной части вещественного числа (указывается только для вещественных чисел). **Замечания:**

1) если MinWidth не указано, то следующий параметр выводится вслед за предыдущим без разделения;

2) если MinWidth опущен, то он принимается равным 23 (по умолчанию), если MinWidth < 10, то он равен 10;

3) если DecPlaces = 0, то дробная часть не выводится; если DecPlaces > 18, он принимается равным 18.

Например: Write('A = ',a:10:5); A = 75,234567 будет напечатано: A = 75.23456,

Write('B = ',b:5); B = 34 будет напечатано: B = 34.

Пример 7. Ввод и вывод имен и значений двух чисел.

```
Program 03_7;
Var
  n1, n2: Integer;
  xr: Real;
Begin
  WriteLn;
  Write('N1 = '); ReadLn(n1);
  Write('N2 = '); ReadLn(n2);
  xr := n1/n2;
  WriteLn;
  WriteLn('N1/N2 = ',xr:5:2);
  ReadLn;
End.
```

Позиционирование печати позволяет установить место знака на строке с помощью пропусков (вывода пробелов).

Формат: Write(' :y); где y — константа, указывающая на число пробелов.

Например: Write(a:5, ' :5,'Сумма = ',b:7). В данном случае между значениями чисел A и B будет показан пропуск в 5 пробелов.

В тексте операторов Write и WriteLn могут находиться выражения, которые вычисляются и выводятся, но не сохраняются в памяти компьютера!

3.4.2. Простой оператор перехода Goto

Оператор Goto позволяет изменить стандартный последовательный порядок выполнения операторов и перейти к выполнению программы, начиная с заданного оператора.

Формат оператора: Goto[M], где M — метка в программе.

В системе программирования Турбо Паскаль допускается использовать в качестве меток целые, константы и идентификаторы, состоящие из символов. Меткой может быть, например, целое число без знака из диапазона от 0 до 9999 или другой идентификатор, например: m1, flag, st23 и т. п.

Метка маркирует оператор, на который передается управление; она отделяется от него символом «:».

При использовании оператора перехода Goto должны соблюдаться следующие правила:

- 1) метка, которая указывается в операторе перехода, должна находиться в том же блоке или модуле, что и сам оператор перехода. Другими словами, не допускаются переходы из процедуры или функции или внутрь нее;
- 2) переход извне внутрь структурного оператора (т. е. переход на более глубокий уровень вложенности) может вызвать непредсказуемые эффекты, хотя компилятор не выдает сообщения об ошибке;
- 3) оператор, следующий за оператором Goto, должен быть обязательно помечен.

3.4.3. Структурированные операторы

Рассмотрим следующие конструкции, применяемые в программировании:

1. Переход по условию.
2. Повторение операций:
 - ◆ заданное число раз;
 - ◆ с постпроверкой условия;
 - ◆ по команде с клавиатуры.

Для реализации показанных конструкций применяют операторы условия, повтора (цикла), выбора.

Оператор условия If

Условный оператор — это структурированный оператор, предназначенный для выделения из составляющих его операторов одного, который и выполняется в дальнейшем. Структурированные операторы обычно включают в себя другие операторы, к которым относятся составные операторы, передачи управления и операторы цикла.

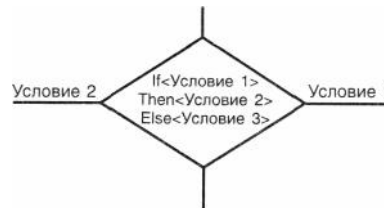
Формат оператора:

```
If <условие> Then <оператор1> Else <оператор2>;
```

где If означает «если», Then — «то», Else — «иначе».

Например: If $k \bmod 5 = 0$ Then

```
r := r1  
Else r := r2; .
```



В логическом выражении (условии) должен получаться результат, имеющий стандартный булевский тип. Если результатом выражения является истинное значение (True), то выполняется оператор, следующий за ключевым словом Then. Если результатом выражения является значение False и присутствует ключевое слово Else, то выполнятся оператор, следующий за ключевым словом Else. Если ключевое слово Else отсутствует, то выполняется следующий по порядку оператор.

При необходимости выполнения более одного оператора, в случае, когда выражение принимает значение True или False, следует использовать составной оператор.

Логическая схема работы оператора:

```
If x > max Then X <= max  
Begin  
  max := x;  
  If x > max  
    Then max := x  
Else  
  Goto 2;  
...  
End; .
```

Если опустить часть Else <оператор2>, то будет выполнена только первая часть оператора!

Пример 8. Определение знака вводимого числа.


```

Program 03_8;
Label 1;
Var
  x: Integer;
Begin
1: WriteLn;
  ReadLn(x);
  If x < 0 Then Write ('Число отрицательное')
  Else
    If x > 0 Then Write ('Число положительное')
  Else
    WriteLn('Число равно нулю');
  Goto 1;
End.

```

Операторы повтора (цикла)

Оператор цикла For. Формат:

For <параметр цикла> := <начальное зн.> To <конечное зн.> Do <оператор>;

где For означает «для», To — «до», Do — «выполнить»;

параметр цикла — переменная порядкового типа (Integer);

начальное зн. — начальное значение переменной;

конечное зн. — конечное значение переменной;

оператор — исполняемый оператор (последовательность операторов).

Например, логическая схема работы оператора For $i := 1$ To n Do $s := s+1$ будет иметь вид, представленный на рис. 3.4.

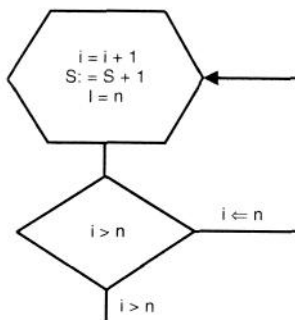


Рисунок 3.4. Описание действия логических операторов

Переменная в конструкции изменяется с шагом +1.

Пример 9. Определение суммы квадратов чисел от 1 до 9.

```

Program 03_9;
Var
  number, sqnumber, sum: Integer;
Begin
  WriteLn;
  sum := 0;
  For number := 1 To 9 Do
    Begin
      sqnumber := number * number;
      sum := sum + sqnumber;
    End;
  WriteLn('Сумма = ',sum);
End.

```

Пример 10. Вычисление корней квадратного уравнения вида: $a*x^2+b*x+c=0$.

```

Program 03_10;
Uses Crt;

```

```

Var
  a,b,c,d,r1,r2: Real;
  i,j: Integer;
Begin
  ClrScr;
  WriteLn('Введите коэффициенты уравнения а, в, с: ');
  Write('a= '); ReadLn(a);
  Write('b= '); ReadLn(b);
  Write('c= '); ReadLn(c);
  d := b*b - 4.0*a*c;
  If d < 0.0 Then
    WriteLn('Вещественных корней нет!')
  Else
    If a = 0.0 Then
      Begin
        r1 := -c/b;
        WriteLn('Уравнение имеет один корень: ',r1:8:4);
      End
    Else
      If d = 0.0 Then
        Begin
          r1 := -b/(2.0*a);
          WriteLn('Корни равны: ',r1:8:4);
        End
      Else
        Begin
          r1 := (-b + sqrt(d))/(2.0 * a);
          r2 := (-b - sqrt(d))/(2.0 * a);
          WriteLn('Первый корень: ',r1:8:4,'второй корень: ',r2:8:4);
        End;
      End;
    For i := 10 To 500 Do
      Begin
        j := i * 10;
        Sound(j);
        Delay(15);
        NoSound;
      End;
    End.

```

Оператор цикла с шагом —1. Формат:

For <...> DownTo <...> Do ...; .

Пример 11. Подсчет квадратов чисел от 9 до 1 в обратном порядке.

```

number, sqnumber: Integer;
Begin
  WriteLn;
  For number := 9 DownTo 1 Do
    Begin
      sqnumber := number * number;
      WriteLn(number, ' => ',sqnumber);
    End;
  End.
Program 03_11;
Var

```

Выход из цикла по Exit. Exit — команда выхода — прекращает выполнение цикла и передает управление следующему за ним оператору.

Оператор цикла While с предпроверкой условия. Формат: While <условие> Do <оператор>; где While означает «пока выполняется условие», Do — «выполнить»;

условие — выражение логического типа: True — выполняется оператор, и снова проверяется условие, False — оператор прекращает свою работу, например:

```

eps:= 1;
Whileeps/2+1 > 1 Do eps := eps/2;
WriteLn('Epsilon = ',eps); .

```

Пример 12. Вычисление значений функций на интервале [7,8] с шагом 0,1 и оформление результатов вычислений в виде таблицы, сопроводив вывод информации звуковым сигналом.

```

Program 03_12;
Uses Crt;
Var
  t,f1,f2,f3: Real;
  a,b,s,g: Integer;
  ch: Char;
Begin
  ClrScr;
  TextColor(13);
  WriteLn('-----');
  WriteLn('| T | F1 | F2 | F3 |');
  WriteLn('-----');
  TextColor(15);
  t := 7;
  While t <= 8 Do Begin
    f1 := sqrt(t)*t-19.8*t+562.8;
    f2 := t-13*t;

    f3 := (sqrt(f1)+sqrt(f2))/100;
    t := t+0.1;
    WriteLn (' ',t:2:1,' | ',f1:2:1,' | ',f2:2:1,' | ',f3:2:1,"");
  End;
  WriteLn('-----');
End.

```

Оператор цикла Repeat ... Until с постпроверкой условия. Формат: Repeat <тело цикла> Until <условие>;

где Repeat означает «повторять», Until — «до тех пор, пока не будет выполнено условие».

Например:...

```

cr := 13; {Код клавиши <Enter>}
Repeat
  ReadLn(ch);
  WriteLn(ch, ' ',ord(ch));
Until Ord(ch) = cr; .

```

Пара Repeat... Until подобна паре Begin ... End.

Пример 13. Вывод значения кода нажатой клавиши.

```

Program 03_13;
Var
  mark: Char;
Const
  cr = 13; {Код клавиши <Enter>}
Begin
  Repeat
    ReadLn(ch);
  Until Ord(mark) = cr;
End.

```

Оператор выбора Case

Формат:

Case <ключ выбора> Of <список выбора> Else <оператор> End; где Case означает «выбрать», Of- «из списка», Else — «иначе», End -«окончание оператора»;

ключ выбора — выражение (переменная) порядкового типа; например: m: 1..5;

lt: char;

список выбора — множество конструкций вида: <константа выбора> : <оператор>;

константа выбора — переменная ключа выбора; например:

```

1: WriteLn('Y');
'+': a := x+y;

```

оператор — оператор Паскаля, например:

```

Var
  an: Integer;
  rm: 0..2;

```

```

...
rai := an mod 3;
Case rm Of
0: WriteLn(an, 'кратно 3');
1: WriteLn(an, 'кратно 3 с избытком Г');
2: WriteLn(an, 'кратно 3 с недостатком Г')
Else
End; .

```

Использование Else позволяет избежать «зависания» программы!

Пример 14. Проведение классификации букв алфавита.

```

Program 03_14;
Label 1;
Var
  letter: Char;
  lettype: 1..5;
Begin
1: Write('Введите букву алфавита: ');
  ReadLn(letter);
  Case letter Of
    'b', 'd' : lettype := 1;
    'c', 'j', 'q': lettype := 2;
    'a': lettype := 3;
    'r', 'z', 'h': lettype := 4;
    'e', 'g', 'i', 'k', 'p': lettype := 5
  Else
  End;
  WriteLn('Буква ', letter, ' в классе ', lettype);
  Goto 1;
End.

```

Любому из операторов списка выбора может предшествовать несколько констант выбора, разделенных запятыми. Например:

```

Var ch: Char;
Begin
  ReadLn(ch);
  Case ch Of
    'n', 'N', '№', 'H': WriteLn('Hex');
    'У', 'У', 'Д', 'Д': WriteLn('7Ia')
  Else
  End;.

```

Ключ выбора (селектор) определяет множество используемых в операторе меток, которые должны иметь тот же тип, что и селектор.

Особенностью оператора Case является то, что при его выполнении активизируется лишь тот оператор среди множества операторов, метка которого совпадает с текущим значением селектора. Если ни одна метка не совпадает со значением селектора, то выполняется оператор (он может быть составным), расположенный вслед за ключевым словом Else.

При организации оператора Case, необходимо соблюдать следующие правила:

- 1) метки, используемые внутри оператора Case, локальны и не могут быть описаны вне данного оператора;
- 2) в качестве меток не допускается использование переменных диапазонного типа;
- 3) не допускается передача управления извне какому-либо помеченному оператору, расположенному внутри оператора Case.

Пример 15. Написание программы, выполняющей арифметические действия над вводимыми аргументами (калькулятор).

```

Program 03_15;
Var
  operation: Char;
  x,y: Integer;
  z: Real;
  flag: Boolean;
Begin
  flag := False;
  Repeat
    WriteLn;
    Write('Введите аргумент x > ');
    ReadLn(x);
    Write('Введите аргумент y > ');
    ReadLn(y);
    Write('Операция > ');
    ReadLn(operation);
  Case operation Of
    '+' : z := x + y;
    '-' : z := x - y;
    '*' : z := x * y;
    '/' : z := x / y
  Else
    flag := True;
  End;
  If Not flag Then WriteLn('Результат = ',z:6:2);
Until flag;
End.

```

3.6. Типы данных

На рис. 3.5 приведена классификация типов данных, используемых в языке Турбо Паскаль.

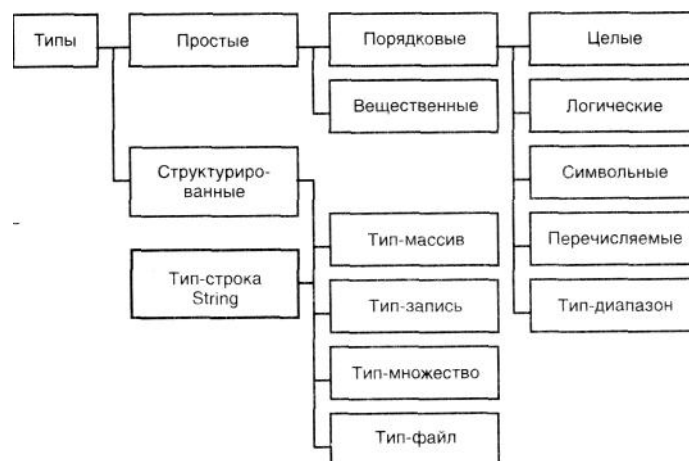


Рисунок 3.5. Классификация типов данных

Тип определяет множество объектов, множество допустимых операций над ними и формат их представлений в компьютере.

3.5.1. Простые типы данных

Порядковый тип данные характеризуется тем, что переменной соответствует порядковый номер — целое число. Этот тип данных разделяется на пять типов: целый, логический, символьный, перечисляемый типы и тип-диапазон.

К данным *целого типа* относят: Byte, Shortint, Word, Integer, Longint — типы, каждый из которых занимает определенное число байтов в памяти компьютера и имеет свой диапазон значений (табл. 3.2).

Таблица 3.2

Целый тип	Длина	Диапазон значений
-----------	-------	-------------------

Byte	1	0 ... 255
Shortint	1	-128 ... 127
Word	2	0 ... 65535
Integer	2	-32768 ... 32767
Longint	4	-2147483648 ... 2147483647

Логический тип — это тип данных, любой элемент которого может принимать лишь два значения: True и False, причем True > False.

Символьный тип — это множество символов, коды ASCII которых лежат в диапазоне от 0 до 255. Код ASCII — American Standard Code for Information Interchange (стандартный код США для обмена информацией).

Код ASCII содержит 128 символов, каждый из которых представляет собой 7-разрядный двоичный. Коды сгруппированы следующим образом:

0 ... 31 — служебные коды;

32 ... 47 и 58—63 — знаки препинания, арифметики и т. п.;

48 ... 57 — цифры;

96 ... 122 и 65—90 — знаки латинского алфавита;

128 ... 255 — знаки национальных алфавитов, псевдографики и др.

Использование знаков псевдографики позволяет формировать простейшие рисунки для организации текста выводимых сообщений.

На рис. 3.6 показано использование знаков псевдографики для изображения рамки таблицы.

Примеры кодов ASCII:

Примеры кодов ASCII:

```

179 — | 180 — †
...
191 — † 192 — †
193 — † 194 — †
195 — † 196 — †
197 — † ...
217 — † 218 — †

```

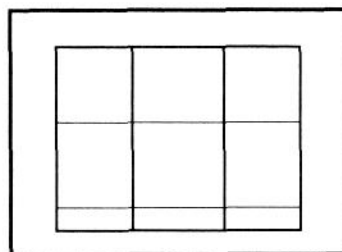


Рисунок 3.6. Изображение с использованием знаков псевдографики

Знак заносится в строку Write с помощью цифровой части клавиатуры при нажатой клавише <Alt>.

Пример 16. Проведение табулирования функции, вывод результатов в виде таблицы и сравнение результатов работы с примером 03_15.

```

Program 03_16;
Uses Crt;
Const pn=0; pk=1; hp=0.05;
Var p,t1,t2: Real;
Begin
  ClrScr;
  p := pn;
  WriteLn('+-----+');
  WriteLn(' | p | t1 | t2 | ');
  WriteLn('+-----+-----+-----');
  Repeat
    t1 := 2.3*p*p*p*p+0.5*p*p*p+5.95*p*p-2.5*p+1.5;
    t2 := 21.7+9.98*exp(p*ln(2));
    WriteLn(' | p:3:1, | | t1:5:0, | | t2:5:0, | | ');

    p := p+hp;
  Until p>pk;
  WriteLn('+-----+');
  Repeat Until KeyPressed;
End.

```

Перечисляемый тип данных задается перечислением возможных значений объекта.

Например:

Type

```

colors = (Red, White, Blue);
Month = (Jan,feb,mar,apr,may,jun,jul,aug,sep,oct,nov,dec); .

```

Тип-диапазон данных, определяет границы изменения данных.

Формат: <мин. значение> ... <макс. значение> .

Например:

Type

```

digit = '0'..'9';
dig=13..21;
alfa = 'A'..'Z'; .

```

*Символы «..» рассматриваются как один символ
и пробелы недопустимы!*

Левая граница диапазона не должна превышать правую!

3.5.2. Структурированные типы данных Тип-массив

Переменные, организованные в массивы данных, описываются с помощью лексической единицы языка Турбо Паскаль массив.

Массив представляет собой формальное объединение нескольких однотипных объектов (чисел, символов, строк и т. п.).

Формат:

<имя массива> : Array[диапазон индекса массива] Of <тип элементов>; .

Например:

Var

```

a:Array[1..10] Of Real;
b:Array[0..50]Of Char;
c: Array[-3..4] Of Boolean; .

```

Пример 17. Формирование массива из S чисел и определение среднего арифметического этих чисел, минимального и максимального из них.

```

Program 03_17;
Const
  N = 50;
Var
  M: Array[1..N] Of Real;
  I: Integer;
  MAX, MIN: Real;
  SUM, S, Between: Real;
Begin
  Write('Введите размер массива: ');
  ReadLn(S);
  For I := 1 To S Do
    Begin
      Write('M['I,'] = ');
      ReadLn(M[I]);
    End;
  SUM := 0; MAX := M[1]; MIN := M[1];
  For I := 1 To S Do
    Begin
      SUM := SUM + M[I];
      If M[i] < MIN Then MIN := M[I]
      Else
      If M[I] > MAX Then MAX := M[i];
    End;
  Between := SUM/S;
  WriteLn;
  WriteLn('Мин = ',MIN:8:2,' Max = ',MAX:8:2,' Среднее =',Between:8:2);
End.

```

В тех случаях, когда используются несколько массивов, имеющих одинаковый размер и включающих данные одного типа, следует описать этот массив в разделе Type. Например:

```

Type
  digit = Array[0..9] Of Char;
  matr = Array[1..10] Of Integer;
  a = Array[1..5,1..4]Of Real;

```

```

Var
  let, leta: digit;
  artic, art, kart: a;
  beta, alpha, delta: matr; .

```

Элементы матрицы в компьютере записываются по столбцам, как показано в следующем примере:

```

a[1,1]a[2,1]
a[1,2] a[2,2] = a[1,1], a[1,2], a[2,1], a[2,2];

```

Например, для организации ввода и вывода двумерной матрицы следует записать:

```

Begin
  Write('Число строк в матрице? >');
  ReadLn(N);
  Write('Число столбцов в матрице? >');
  ReadLn(M);
  For I := 1 To N Do {Ввод значений элементов матрицы}
    For J := 1 To M Do
      Begin
        Write('A['I,']J,'] = ');
        ReadLn(A[i,j]);
      End;
  For I := 1 To N Do {Вывод массива в виде матрицы}
    Begin
      For J := 1 To M Do
        Write(A[I,J], ' ');
      WriteLn;
    End;

```

Массив позволяет:

1) хранить элементы вектора, матрицы в памяти ЭВМ с теми же обозначениями, как они используются в математических записях;

- 2) организовать циклические вычисления с исходными данными, значения которых изменяются не только по линейному, но и по любому нелинейному закону;
- 3) уменьшить объем программ, сделать их более наглядными, а значит, и более надежными;
- 4) составлять программы в более общем виде, задавая размеры массивов при решении задачи в процессе ввода исходных данных.

Массивы символов

Описание для одномерного массива символов имеет вид:

```
Type
a = Array[1..80] Of Char;
```

```
Var
mb: a; .
Или:
```

```
Var
b: Array[1..80] Of Char; .
```

Для двумерного массива описание имеет вид:

```
Type
d = Array[1..10] Of Char;
s = Array[1..5] Of d;
Или:
```

```
Type
s = Array[1..5] Of Array[1..10] Of Char
```

```
Var
z: s;
Или:
```

```
Var
z: Array[1..5, 1..10] Of Char; .
```

Массив строк

Описание массива строк имеет вид:

```
Var
a: Array[1..n] Of String[m];
```

где n — число строк в массиве;

m — длина строки (число знаков в строке).

Если параметр m не указан, то длина каждой строки равна 255 знакам.

Тип-строка String

Формат: `String[N]`, где $N = 1..255$, N — константа порядкового типа.

Строка — цепочка символов, составляющих одномерный массив: `Array[0..N] Of Char`.

Нулевой элемент [0] массива содержит текущую длину строки; например:

```
Var
st: String;
k := Ord(st[0]); .
```

Пример 18. Определение количества гласных и согласных во вводимой строке.

```
Program 03_18;
Uses Crt;
```

```

Const
  n = 30;
Var
  a: String[n];
  p: String[7];
  g,s,i,j,k: Integer;
Begin
  ClrScr;
  p := 'aejiouy';
  WriteLn('Введите текст: ');
  k := 0;
  Repeat
    k := k+1;
    Read(a[k]);
  Until (k=n) Or (a[k]='. ');
  WriteLn;
  g := 0; s := 0;
  For i := 1 To k Do
    For j:= 1 To 7 Do
      If a[i] = p[j] Then g := g+1;
  s := k-g;
  WriteLn('Гласных',g);
  WriteLn('Согласных',s);
  If g > s Then WriteLn('Гласных > согласных');
  If g = s Then WriteLn('Кол-во гласных = кол-ву согласных');
  If s > g Then WriteLn('Согласных > гласных');
End.

```

Для строк определены следующие операции, процедуры и функции.

Операция сцепления — «+», ее действие можно проиллюстрировать следующими примерами:

st := 'a' + 'b';

st := st + 'c'; { st содержит abc }

Операции отношения — =, <, >, <>, <=, >=. Используются в основном при проверке условий.

Процедура Delete. Формат: Delete(St, Pos, Num).

Удаляет участок строки St, содержащий Num символов, начиная с позиции Pos, где St — строковая переменная;

Pos, Num — выражения целочисленного типа.

Например, если St принимает значение 'abcdefg', то в результате выполнения процедуры Delete(St,2,4) получим St='afg'.

Процедура Insert. Формат: Insert(Obj, Target, Pos).

Выполняет вставку строки Obj в строку Target начиная с позиции Pos, где Obj — строковое выражение;

Target — строковая переменная;

Pos — целочисленное выражение.

Например, если строка St принимает значение 'abcdefg', то в результате выполнения процедуры Insert('xx',St,3) получим St='abxxcdefg'.

Процедура Str. Формат: Str(Value[:Width[:Decimals]],St).

Выполняет преобразование числового значения Value в строку символов и запоминает результат в строку St,

где Value — параметр записи целочисленного или вещественного типа;

Width — общая ширина поля;

Decimals — количество символов в дробной части;

St — строковая переменная.

Например: если I принимает значение 1234, то в результате выполнения процедуры Str(I:5,St) переменная St принимает значение «1234».

Процедура Val. Формат: Val(St, Value, Code).

Преобразует строку символов St во внутреннее представление целой или вещественной переменной Value, которое определяется типом этой переменной,

где St — строка символов;

Value — целая или вещественная переменная;

Code — параметр, равный 0, если преобразование прошло успешно, или равен номеру позиции, где обнаружена ошибка.

Функция Length. Формат: Length(St).

В результате выполнения этой функции получается число, равное длине строкового выражения St (число символов в строке).

Пример 19. Определение количества раз встречаемости каждой буквы алфавита во вводимой строке.

```
Program 03_19;
Uses Crt;
Var
  st: String;
  a: Array[1..26] Of Char;
  n: Array[1..26] Of Integer;
  i,j: Integer;
Begin
  ClrScr;
  WriteLn('Введите строку: ');
  ReadLn(st);
  For i := 1 To 26 Do
    Begin
      a[i] := Chr(Ord('a')-1+i);
      n[i] := 0;
      For j := 1 To Length(st) Do
        If st[j] = a[i] Then n[i] := n[i] + 1;
      WriteLn('Буква', ' ', a[i], ' ', 'встречается', ' ', n[i], ' ', 'раз');
    End;
  End.
```

Функция Concat. Формат: Concat(St1, St2,...,StN).

Результатом выполнения функции Concat становится строка, состоящая из последовательности строк St1, St2,..., StN.

Функция Copy. Формат: Copy(St, Pos, Num).

Результатом выполнения функции Copy является подстрока, содержащая Num символов строки St, начиная с позиции Pos, где St — строковое выражение;

Pos, Num — целочисленные выражения.

Например, если строка принимает значение 'abcdefg', то после выполнения функции Copy(St,3,2) получим 'cd'.

Пример 20. Ввод в строку предложения, определение числа слов в нем и вывод слов построчно.

```
Program 03_20;
Uses Crt;
Var
  st: String[80];
  a,b,c,dl: Integer;
Begin
  ClrScr;
  b := 0; c := 0; dl := 0;
  WriteLn('Введите предложение: ');
  ReadLn(st);
  WriteLn('-----');
  For a := 1 To (Length(st)) Do
    If (st[a] = ' ') Or (st[a] = '.') Then
      Begin
        dl := a-c; b := b+1;
        WriteLn(Copy(st,c,dl)); c := a+1;
      End;
  WriteLn('-----');
  WriteLn('Количество слов -', b);

  ReadLn(b);
End.
```

Функция Pos. Формат: Pos(Subst, St).

Передаёт номер позиции, с которой начинается подстрока Subst в строке St; например:

```

Var s : String;
Begin
  s:='123.5';
  While Pos(" ", s) > 0 Do (Преобразовать пробелы в нули)
    s[Pos(" ", s)] := '0';
End.

```

Пример 21. Замена слова в предложении на вводимое с клавиатуры.

```

Program 03_21;
Uses Crt;
Var
  sl,st: String;
  mark: Char;
  a,b,dl: Integer;
Begin
  mark := 'y';
  Repeat
    ClrScr;
    WriteLn;
    WriteLn('Введите предложение: ');
    ReadLn(st);
    Repeat
      WriteLn('Какое слово заменить?');
      ReadLn(sl);
      a := Pos(sl,st);
      If a=0 Then WriteLn("Такого слова в предложении нет");
    Until a<>0;
    WriteLn('На какое слово заменить: ');
    ReadLn(sl);
    b := a;
    Repeat
      b := b + 1;
    Until (st[b] = ' ') Or (st[b] = '.');
    dl := b - a;
    Delete(st,a,dl);
    Insert(sl,st,a);
    WriteLn(st);

    WriteLn('Ввести новое предложение? (y/n)');
    ReadLn(mark);
  Until (ch = 'n') Or (ch = 'N');
End.

```

Функция Uprcase. Формат: Uprcase(Ch).
Преобразует строчную букву в прописную.

Примеры:

```

Var
  x: Real;
  y: Integer;
  st, st1: String;
  ...
  st := Concat('3','06',47); {st содержит 30647}
  st1 := Copy(st,3,length(st)-2); {st1 содержит 647}
  Insert('-',st1,2); {st1 содержит 6-47}
  Delete(st,pos('0',st),3); {st содержит 37}
  Str(pi:6:2,st); {st содержит 3.14}
  st1 := '3,1415';
  Val(x,st1,y); . {Ошибка y=2}

```

Тип-запись

Тип-запись определяется в виде совокупности компонентов (полей записи). Формат:

<имя типа> = Record

<поле 1> : <тип переменных>;

<поле 2> : <тип переменных>;

...
<поле N> : <тип переменных>;

End; .

Например:

Type

student = Record

name: String;

group: Word;

End;

Var

excellent: student; .

Обращение (доступ) к полю записи: <переменная>.<поле>; .

Например: excellent.name := 'Smirnov'; .

Поля могут иметь дальнейшее разделение (вложенные поля), они определяются в разделе описания:

Var

place: Record

university: String;

list: student;

End;.

Доступ к данным поля: <имя записи>.<имя поля>.<вложенное поле>....

Для примера, доступ к полю Name необходимо определить следующим образом: place.list.name .

Для упрощения доступа к полям записи применяется оператор присоединения With. Структура оператора:

With <переменная> Do <оператор> .

Например: без оператора With ... Do необходимо записать: place. list.name := 'Smirnov', после оператора With place.list Do можно обращаться: name := 'Smirnof' .

Далее в программе переменная Name определяется как подполе поля List в записи Place, например: name := 'Smit'.

Тип-множество

Множества представляют собой наборы упорядоченных объектов, число которых от 0 до 256.

Описание типа: <имя типа> = Set Of <базовый тип>;

Word, Integer, Longint.

Конструктор множества: список элементов множества в квадратных скобках.

Например:

Type

digitChar = Set Of '0'..'9';

digit = Set Of 0..9;

Var

s1,s2,s3: digitChar;

m1,m2,m3: digit;

...

s1 := ['1','2','3'];

s2 := ['2','5'];

s3 := ['2','1','3'];

m1 := [0..4,6];

m2 := [4,5];

m3 := [2,3]; .

Для множеств определены следующие операции:

* — пересечение множеств, например: m1*m2 = [4], результат содержит общий для обоих множеств m1 и m2 из предыдущего примера элемент 4;

+ — объединение множеств, например: s1+s2 = ['1','2','3','5'], результат содержит элементы первого и недостающие элементы из второго множества;

- — разность множеств, например: m1-m2 = [0,1,2,3,6], результат содержит элементы первого множества, не принадлежащие второму;

= — проверка эквивалентности, например:

If $s1 = s3$ Then True, означает: если оба множества эквивалентны, то: $s1 = s3$ возвращает True;

\diamond — проверка неэквивалентности, например:

If $m1 \diamond m2$ Then True, определяет: если оба множества неэквивалентны, то: $m1 \diamond m2$ возвращает True;

\leq — проверка вхождения, например, если записано If $m3 \leq m1$ Then True, то это означает: если первое множество включено во второе, то: $m3 \leq m1$ возвращает True;

\geq — проверка вхождения, например, если показано If $m1 \geq m3$ Then True, то это определяет: если второе множество включено в первое, то: $m1 \geq m3$ возвращает True;

In — проверка принадлежности, например, при записи <выражение> In <множество> выдается знак True, если выражение (его результат) принадлежит множеству. Как, например:

3 In $m3$ — True;

2*3 In $m1$ — True;

2*3 In $m2$ — False.

Пример 22. Выделение из 250 целых чисел всех простых. В качестве алгоритма используем построение, известное как «решето Эратосфена». Оно заключается в следующем:

- ♦ формирование множества простых чисел [2..N] {BeginSet};
- ♦ передача в множество простых чисел 1 {PrimerSet};
- ♦ организация цикла;
- ♦ передача в множество простых чисел следующего (+1) числа;
- ♦ удаление из BeginSet всех других, кратных: 2*next, 3*next и др.;
- ♦ повторение цикла, пока BeginSet не станет пустым.

Program 03_22;

Const

N = 250;

Type

SetOfNumber = Set Of 1..N;

Var

n1,next,i: Word;

BeginSet, PrimerSet: SetOfNumber;

{Исходное множество и множество простых чисел}

Begin

BeginSet := [2..N]; {Формирование множества от 2 до N}

PrimerSet := [1]; {Первое простое число}

next := 2; {Следующее простое число}

While BeginSet \diamond [] Do {Основной цикл}

Begin

n1 := next; {n1 — кратное очередному простому}

While n1 \leq N Do {Удаление непростых чисел из BeginSet}

Begin

BeginSet := BeginSet - [n1];

n1 := n1+next; {Следующее кратное}

End; {Конец цикла удаления}

PrimerSet := PrimerSet + [next];

Repeat

{Следующее простое — первое невычеркнутое из исходного множества}

Inc(next);

Until(next In BeginSet) Or (next > N);

End; {Конец основного цикла}

For i := 1 To N Do

If i In PrimerSet Then Write(i:8);

WriteLn;

End.

Тип-файл

Файл (от англ. *file* — картотека, подшивка) — линейная последовательность компонентов одного типа.

Размер файла (количество компонентов) не определяется, компоненты не имеют индексов! Первый компонент файла считается нулевым — за последним компонентом записывается признак конца файла (Eof).

Структура файлового типа:

Type <имя файла> = File Of <тип файла> .

Например:
TypeF = FileOfInteger;
Var fv: F; ...

или:

Var fv: File Of Integer; .

Для файла определены следующие процедуры и функции.

Процедура Assign — связывает внешний файл, размещенный на диске, с файловой переменной.

Формат: Assign (fv/имя файла').

Внешним файлом является поименованный файл на диске. Это имя присваивается fv, и все остальные действия над fv будут фактически предполагать действия над дисковым файлом.

Assign не употребляется для файла, который в настоящий момент используется (открыт)!

Процедура Rewrite — создает и открывает новый внешний файл с именем, определенным переменной fv.

Формат: Rewrite(fv).

В процессе выполнения процедуры реализуются следующие операции:

- ◆ создания внешнего файла с именем fv;
- ◆ открытия внешнего файла с именем fv;
- ◆ установка указателя файла на начало файла (нулевой номер);
- ◆ удаление любого уже существующего файла с таким же именем.

Файл открыт для записи и закрыт для чтения!

Например, если следует записать в файл row ряд из десяти целых чисел, то необходимо написать:

Assign(fv, 'row');

Rewrite(fv);

For i := 1 To 10 Do Write(fv,i); .

Процедура Write — инициация файла для записи. Записывает переменную(ые) в соответствующий(ие) компонент(ы) файла. Формат: Write(fv,v1 [,v2,v3,...vn]).

Например, запись в файл chis десять первых натуральных чисел: Assign(fv,'chis'); Rewrite(fv);

For i := 1 To 10 Do Write(fv,i); . Следует учесть, что:

- ◆ переменные и компоненты файла должны быть одного и того же типа;
- ◆ признак конца файла будет записан после последнего компонента;
- ◆ если идентификатор файловой переменной не описан, то оператор выполняется как оператор вывода информации на экран!

Процедура Close — закрывает внешний файл, ассоциированный с файловой переменной fv.

Формат: Close(fv).

Файл следует закрывать после всех случаев работы с ним!

Процедура Reset — подготавливает к чтению с начала файла fv. Она открывает внешний файл, ассоциированный с именем fv.

Формат: Reset(fv).

Замечания:

- ◆ если файл с заданным именем на существует, возникает ошибка;
- ◆ если файл уже открыт, то его нужно закрыть, а затем повторно открыть;
- ◆ указатель файла стоит на нулевом компоненте (начале файла);
- ◆ если файл пуст, то указатель файла стоит на признаке конца файла.

Процедура Read — вводит символы, строки и числа.

Формат: Read(fv,v) или Read(fv,v1[,v2,...vn]),

где fv — файловая переменная;

v — переменная ввода;

v1,...,vN — список ввода.

При каждом считывании в переменную указатель файла переводится к следующему компоненту.

Конец строки (v) помечается клавишей <Enter> (ASCII 13), конец файла (конец ввода) — символом Eof(ASCII 26) = <Ctrl>+<Z>.

Процедура Seek — перемещает указатель файла к заданному компоненту.

Формат: (fv,n), где n — выражение целого типа.

Для расширения файла следует переместить указатель файла в конец файла: Seek(fv,FileSize(fv)).

Функция Eof(fv) — указывает конец файла. Выдает True, если указатель на последней записи или записей в файле нет, False — в других случаях.

Функция FileSize(fv) — определяет текущий размер файла (число компонентов в файле). Так как компоненты fv нумеруются, начиная с 0, это число на 1 больше номера последнего компонента.

При использовании функции FileSize(fv) файл должен быть открыт!

Функция FilePose(fv) — определяет номер текущего компонента в файле. Если указатель находится в начале файла, то FilePose(fv) равен 0.

Пример 23. Запись в файл значения функции $y = \sin(x)$ на интервале [a,b] с шагом hx.

```
Program 03_23;
Type TF = File Of Real;
Var
  f: TF;
  x,a,b,hx,y: Real;
  i,k: Integer;
Begin
  WriteLn('Введите интервал изменения аргумента и шаг: ');
  ReadLn(a,b,hx);
  Assign(f,'a:\fil.dat');    {Связь переменной f с внешним файлом}
  Rewrite(f);                {Открыть файл для записи}
  x := a; k := Trunc((b-a)/hx+1);
  {Определение числа повторений в цикле}
  For i := 1 To k Do
    Begin
      y := sin(x);
      Write(f,y);            {Запись вычисленных значений y в файл}
      x := x + hx;
    End;
  Close(f);                  {Закрыть файл}
  WriteLn('Созданный файл a:\fil.dat содержит элементы');
  WriteLn('с номерами 0 ..',k-1);
End.
```

Пример 24. Вычисление среднего арифметического элементов от m до n из файла $y = \sin(x)$.

```
Program 03_24;
Type TF = File Of Real;
Var
  f: TF;
  x,sr: Real;
  i,k,m,n: Integer;
Begin
  WriteLn('Введите начальный и конечный номера элементов файла');
  ReadLn(m,n);

  Assign(f,'a:\fil.dat');    {Связь переменной с внешним файлом}
  Reset(f);                  {Открыть файл для чтения}
  Seek(f,m);                  {Установить указатель файла на номер m}
  sr := 0; k := n-m+1;
  For i := 1 To k Do
    Begin
      Read(f,x);
      sr := sr+x;
    End;
  Close(f);
  sr := sr/k;
  WriteLn('Среднее значение равно: ',sr:6:2);
End.
```


Пример 25. Организация ввода символического файла и выдача строчных букв из него.

```
Program 03_25;
Type t = File Of Char;
Var
  fv: t;
  s: Char;
Begin
  Assign(fv,'a:\simv.ch');
  Rewrite(fv);
  While s <> '.' Do
    Begin
      Read(s);
      Write(fv,s);
    End;
  Close(fv);
  Reset(fv);
  While Not Eof(fv) Do
    Begin
      Read(fv,s);
      If(s >= 'a') And (s <= 'z') Then
        Write(S:2);
    End;
  Close(fv);
End.
```

Дополнительные сведения о работе с внешними файлами

Управление внешними файлами заключается в предоставлении возможности устанавливать различные имена для обрабатываемых файлов, перенаправлять ввод и вывод файлов на различные устройства компьютера.

Установка имени обрабатываемого файла осуществляется с помощью процедуры `Assign(intname,extname)`.

В процедуре используются параметры:

- ◆ `intname` — имя файла, упоминаемое в программе;
- ◆ `extname` — любое выражение типа `String`, оно может быть задано:
 - как текст — «a:\stud.txt»;
 - как символическая константа, определенная в разделе (`const`);
 - как переменная, значение которой вводится пользователем, например:

```
Var datafile: Text;
  extname: String[14];
...
WriteLn('Укажите имя внешнего файла:');
ReadLn(extname);
Assign(datafile,extname);
...

```

Для организации ввода данных с консоли параметр `extname` определяет имя 'con:' как имя внешнего файла в предложении `Assign`, например `Assign(datafile,'con:')`.

При направлении данных на принтер параметр `extname` должен быть обозначен как 'lst:' — имя внешнего файла в предложении `Assign` (`list device` — печатающее устройство). Например:

```
Var data : Text;
...
Assign(data,'lst:');
Rewrite(data);
...
WriteLn(data,...);
{Если использовать Write, то последняя строка не будет напечатана}
Close(printer); .

```

Если использовать 'con:' или опустить 'lst:', то информация будет выведена на экран компьютера!

Например:

```

Varfflel,ffle2:Text;
  devicename: String[14];
  ...
  WriteLn('Укажите имя устройства для вывода: ');
  ReadLn(devicename);
  {Устройство вывода задается с клавиатуры}
  Assign(file 1,devicename);
  Rewrite(file1); {Направить file1 на указанное устройство вывода}
  ...
  Assign(file2,'con:');
  Rewrite(file2);      {Направить file2 на консоль}
  ...

```

Если файловая переменная связана с внешним устройством, то все выводимые значения будут направлены на соответствующее внешнее устройство!

Например:

```

WriteLn(file1,'перВbrii');      {Слово 'Первый' вывести на консоль}
WriteLn(file2,'ВТОpou');
WriteLn(output,'ТpeTHU');
{Слово Третий' через стандартный файл вывода output вывести на} {экран}
WriteLn('4eТВepTbin');
{Слово 'Четвертый' по умолчанию выводится в файл output, а затем} {на экран}
При вводе используется стандартный файл ввода Input.

```

Логические устройства в Турбо Паскале

Логическое устройство рассматривается как внешний файл с определенным именем, которое связывается с переменной типа Text.

Турбо Паскаль различает следующие логические устройства:

Con: — пульт управления (консоль); может быть использован как для ввода, так и для вывода;

Trn: — терминальное устройство (терминал); используется как для ввода, так и для вывода;

Kbd: — устройство ввода с клавиатуры (клавиатура);

Lst: — устройство вывода на печать (обычно принтер).

Например, назначения: Assign(file1,'Lst:'); Assign(file2,'Kbd:').

Каждому логическому устройству соответствует стандартный файл: Con, Trn, Lst.

Например:

Read(Con,x,y) — считывает значения x,y;

WriteLn(Lst,x,y) — печатает значения x,y на принтере.

При использовании стандартных файлов предложения с использованием процедур Assign, Reset, Rewrite, Close не допускаются!

3.6. Процедуры и функции

Процедуры и функции в Турбо Паскале — самостоятельные фрагменты программы (подпрограммы), снабженные именем и предназначенные для многократного выполнения определенного набора операторов, реализующие самостоятельные смысловые части алгоритма.

Упоминание в программе имени подпрограммы называется **вызовом подпрограммы**.

Подпрограммы могут иметь свои константы, переменные, типы и подпрограммы следующего (более низкого) уровня.

Из программы С можно обратиться к А и В, но нельзя обратиться непосредственно к процедурам А1, ..., АN, а также к любым именам, объявленным в них!

Обозначив текст программы А_і между скобками Begin и End знаком {А_і}, можно показать графическую интерпретацию взаимосвязи отдельных процедур и функций и основной программы, представленную на рис. 3.7.

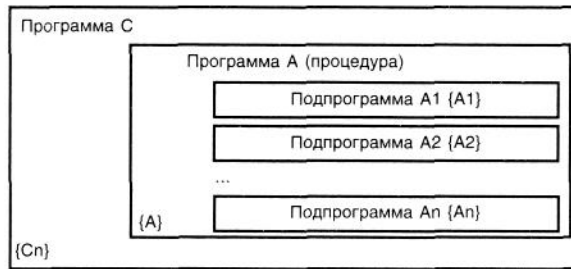


Рисунок 3.7.

Из подпрограмм сохраняется доступ ко всем именам верхнего уровня (из подпрограммы ВN можно обратиться к подпрограмме А).

Каждая процедура или функция описывается в разделе описаний программы!

Размещение подпрограмм в разделе описаний программы показано на следующем примере:

```

Program C;
Var ...
Procedure A1;
Var...
Begin ...End;  {A1}
...
Begin ... End.  {C}

```

Описать подпрограмму — означает указать ее заголовок и тело в разделе описаний программы.

Заголовок процедуры:

Procedure<имя>[(<список формальных параметров>)]; .

Список формальных параметров — перечисление всех имен и типов формальных параметров (если они есть), например:

Procedure rfile(n: String; Var dat: Data).

Параметры в списке отделяются друг от друга знаком «;».

Пример 26. Возведение вещественного числа в вещественную степень.

```

Program 03_26;
Var x,y: Real;
Function Power(a,b: Real): Real;
Begin
  If a > 0 Then
    Power := Exp(b * Ln(a))
  Else
    If a < 0 Then
      Power := Exp(b * Abs(a))
    Else
      If b = 0 Then
        Power := 1
      Else
        Power := 0;
  End;
Begin {03-26a}
  Repeat
    ReadLn(x,y); {Фактические параметры}
    WriteLn(power(x,y):12:2, power(x,-y):12:2);
  Until Eof; {<Ctrl>+<Z> +<Enter>}
End. {03-26a}

```

Число и типы формальных параметров должны совпадать с числом и типами фактических параметров!

Формальный параметр может быть параметром-значением (a,b: Real) или параметром-переменной (Var: a, b: Real).

Если в подпрограмме используется параметр-значение, то после выполнения подпрограммы его значение восстанавливается.

При указании в подпрограмме параметра-переменной ее значение из подпрограммы передается в вызывающую программу.

Пример 27. Написание программы управления файлом с помощью меню, использующего функциональные клавиши.

```
Program 03_27;
Uses Crt, DOS;
Var c,i,u: Char;
    t: Char;
    f: Text;
    r: SearchRec;
    o,p: String;
Procedure Menu;
Begin
    WriteLn(' 1) Создать файл');           {Creation}
    WriteLn(' 2) Найти файл');           {Find}
    WriteLn(' 3) Добавить в файл');      {App}
    WriteLn(' 4) Вывести файл');         {Read_to}
    WriteLn(' 5) Переименовать файл');   {Rename}
    WriteLn(' 6) Удалить файл');        {Erase}
    WriteLn(' 7) Выход');
End;
```

```

Procedure Creation;      {#49}
Var t: Integer;
Begin
  ClrScr;
  Write('Введите имя файла: '); ReadLn(o);
  WriteLn('Введите текст: ');
  Write(' ');
  Assign(f,o+'.txt');
  Rwrite(f);
  While u <> #13 Do Begin
    u := ReadKey;
    Write(u);
    Write(f,u);
  End;
  Close(f);
  WriteLn;
  WriteLn('Файл записан');
  WriteLn('Для перехода в меню нажмите любую клавишу');
  c := ReadKey;
End;
Procedure Find;        {#50}
Begin
  ClrScr;
  WriteLn('Файлы с расширением .txt: ');
  Findfirst(*.txt',archive,r);
  While DosError = 0 Do Begin      {Функция модуля DOS}
    Write(r.name,' «);
    Findnext(r);
  End;
  WriteLn;
  WriteLn('Для перехода в меню нажмите любую клавишу');
  c := ReadKey;
End;
Procedure App;        {#51}
Begin
  ClrScr;
  Write('Введите имя файла: ');
  ReadLn(o);
  Assign(f,o+'.txt');
  Append(f);
  Repeat
    u := ReadKey;
    Write(u);
    Write(f,u);
  End;

```

```

Until u = #13;
Close(f);
WriteLn;
WriteLn('Информация добавлена, для продолжения нажмите');
WriteLn ('любую клавишу');
End;
Procedure Read_to;           {#52}
Begin
  ClrScr;
  Write('Введите имя файла: ');
  ReadLn(o);
  ClrScr;
  Assign(f,o+'.txt');
  Reset(f);
  Write(' ');
  While Not SeekEof(f) Do Begin {Функция пропуска конца строки}
    Read(f,u);
    Write(u);
  End;
  WriteLn;
  WriteLn('Для перехода в меню нажмите любую клавишу');
  c := ReadKey;
End;
Procedure Ren;              {#53}
Begin
  ClrScr;
  Write('Введите старое имя файла: ');
  ReadLn(o);
  Write('Введите новое имя файла: ');
  ReadLn(p);
  Assign(f,o+'.txt');
  Rename(f,p+'.txt');
  WriteLn('Файл переименован!');
  WriteLn('Для продолжения нажмите любую клавишу');
  c := ReadKey;
End;
Procedure Delete;
Begin
  ClrScr;
  Write('Введите имя файла (без расширения): ');
  ReadLn(o);
  Assign(f,o+'.txt');
  Reset(f);
  Erase(f);
  WriteLn('Файл удален');
  WriteLn('Для продолжения работы нажмите любую клавишу');
  c := ReadKey;
End;
Begin                       {Тело основной программы}
  Repeat
    ClrScr;
    TextColor(11);
    menu;
    i := ReadKey;
    Case i Of
      #49: creation;         {Создать файл}
      #50: find;             {Найти файл}
      #51: app;              {Добавить в файл}
      #52: read_to;         {Прочитать файл}
      #53: ren;             {Переименовать файл}
      #54: delete;          {Удалить файл}
    End;
  Until i = #55;
End.

```

Пример 28. Создание базы данных для соревнований по тяжелой атлетике и выдача сведений о призерах.

```

Program 03_28;
Uses Crt, DOS;
Type
  sp = Record
    country: String[20];
    surname: String[20];
    result: Integer;
  End;
  ar = Array[1..5] Of sp;
Label cont;
Var
  a: ar;
  f: File Of sp;
  i: Byte;
  fn: String[12];
  c: Char;
Procedure sort(Var a: ar);
Var t: sp;
  i,j: Byte;
Begin
  For i := 1 To 5 Do
    For j := i To 5 Do
      If a[i].result < a[j].result Then
        Begin
          t := a[i];
          a[i] := a[j];
          a[j] := t;
        End;
      End;
    End;
  End;
Begin
  TextColor(White);
  TextBackground(Blue);
  ClrScr;
  WriteLn('Создайте базу данных о соревнованиях по тяжелой атлетике');
  GotoXY(10,25);
  Write('Введите имя файла: ');
  ReadLn(fn);
  GotoXY(1,2);
  Assign(f,fn);
  If search(fn,"") <> " Then Begin
    WriteLn('Файл уже существует');
    Reset(f);
    For i := 1 To 5 Do Read(f,a[i]);
    Goto cont;
  End;
  WriteLn('Требуется 5 записей. Формат ввода: ');
  WriteLn('Страна: Фамилия: Результат: ');
  Rewrite(f);
  For i := 1 To 5 Do
    Begin
      GotoXY(2,3+i);
      ReadLn(a[i].country);
      GotoXY(21,3+i);
      ReadLn(a[i].surname);
      GotoXY(41,3+i);
      ReadLn(a[i].result);
    End;
  End;
  For i := 1 To 5 Do
    Write(f,a[i]);
  Close(f);
  cont : WriteLn('Отсортировать файл (по местам) (y/n)?');
  c := ReadKey;
  If (c = 'y') Or (c = 'Y')
    Then

```

```

    sort(a);
    WriteLn('Место: Страна: Фамилия: Результат: ');
    For i := 1 To 5 Do
        WriteLn(' ',a[i].country:20,a[i].surname:20, a[i].result:10);
    c := ReadKey;
    ClrScr;
End.

```

Структура функции:

```

Function <имя>:(var <p1>:<тип>[,...,<pN>:<тип>]):<тип>;
Begin
... (тело подпрограммы);
End;

```

где p1,p2,...,pN — имена переменных функции.

Использование в программе:

```
<pP>:=<имя функции>(pP1[,...,pPN]);
```

где pP — переменная основной программы;

pP1,...,pPN — значения переменных основной программы, передаваемые в подпрограмму-функцию.

Тип переменной pP и тип функции (Function), значение которой присваивается этой переменной, должны совпадать!

3.7. Модуль в Турбо Паскале

Модуль — это автономно компилируемая программная единица, доступная для основной программы, содержащая, как правило, библиотеку процедур и функций основной программы, размещаемая в отдельном сегменте памяти, что позволяет увеличить объем программного кода.

Модуль состоит из заголовка и трех составных частей, любая из которых может быть пустой:

```

Unit<имя>
Interface<интерфейсная часть>
Implementation<исполнительная часть> {Выполнение}
Begin<иницилирующая часть>
End.

```

Имя модуля должно совпадать с именем дискового файла модуля!

Например, модулю Unit Global должен соответствовать файл Global.pas, модулю Unit LP_141 должен соответствовать файл LP_141.pas.

Связь основной программы с модулем устанавливается предложением Uses<список модулей>; .

Например: Uses Crt, LP_141, Global;.

Если модуль использует другие модули, то предложение Uses следует непосредственно за Interface, либо за Implementation.

Интерфейсная часть модуля содержит объявления всех глобальных объектов модуля: типов, констант, переменных, подпрограмм.

При объявлении глобальных подпрограмм в интерфейсной части указываются только их заголовки.

Например:

```

Unit PL;
Interface
Uses Crt;
Procedure WindowS(x1,y1,x2,y2: Integer); .

```

При использовании модуля PL становится возможным применение процедуры WindowS в основной программе.

Исполнительная часть модуля содержит описания программ, объявленных в интерфейсной части модуля; например:

```
Unit PL;
```



```

Interface
Uses Crt;
Procedure WindowS(x1,y1,x2,y2: Integer);
Procedure WindowI(x1,y1,x2,y2: Integer);
...
Implementation
Procedure WndowS;
Procedure WindowI;
...

```

В заголовке подпрограммы можно опустить список формальных переменных и их тип (и тип результата для функций).

Иницилирующая часть модуля размещает исполняемые операторы. Если таких операторов нет, то иницилирующая часть может отсутствовать.

Пример 29. Создание модуля, содержащего процедуры организации окон.

```

Unit 03_29;
Interface
Uses Crt;
Procedure Window1(x1,y1,x2,y2 : Integer);
Procedure Window2(x1,y1,x2,y2 : Integer);
Procedure Window3(x1,y1,x2,y2 : Integer);
Implementation
Procedure Window1;
Var x: Byte;
Begin
  Window(x1,y1,x2,y2);
  TextBackground(White); TextColor(Black);
  ClrScr;
  x := (x2 - x1) Div 2 - 5;
  GotoXY(x,1); Write('Ввод данных');
End;
Procedure Window2;
Var x: Byte;
Begin
  Window(x1,y1,x2,y2);
  TextBackground(Yellow); TextColor(Blue);
  ClrScr;
  x := (x2 - x1) Div 2 - 5;
  GotoXY(x,1); Write('Вычисления');
End;
Procedure Window3;
Var x: Byte;
Begin
  Window(x1,y1,x2,y2);
  TextBackground(Cyan); TextColor(Black);
  ClrScr;
  x := (x2 - x1) Div 2 - 2;
  GotoXY(x,1); WriteLn('Меню');
  WriteLn;
  WriteLn('Ввести <F1>');
  WriteLn('Записать <F2>');
  WriteLn('Загрузить <F3>');
  WriteLn('Очистить <F4>');

  WriteLn('Вычислить <F5>');
  Write('Выход <F9>');
End;.

```

Пример 30. Подготовка программы, использующей «оконный» модуль.

```

Program 03_30;
Uses Crt, 03_29;
Const
  x11 = 3; y11 = 2; x12 = 58; y12 = 12;
  x21 = 3; y21 = 15; x22 = 58; y22 = 23;
  x31 = 62; y31 = 2; x32 = 76; y32 = 12;
Var
  ch: Char;
Begin
  Window1(x11,y11,x12,y12);
  Window2(x21,y21,x22,y22);
  Window3(x31,y31,x32,y32);
  ch := ReadKey;
  TextMode(Co80);           {Восстанавливаем экран}
End.

```

При компиляции основной программы все упоминающиеся в программе модули должны быть откомпилированы и находиться в том же каталоге.

В процессе компиляции основной программы на базе модулей, размещенных в файлах с расширением .pas, создаются программные модули с расширением .tpr.

Организация выхода из программы осуществляется с помощью меню, использующего функциональные клавиши. Суть этого метода станет ясной после анализа текста следующего примера:

```

Uses Crt;
Var
  ch: Char;
  b: Boolean;
Begin
  ...
  b := True;
  While b Do
    Begin
      ch := ReadKey;
      Case ch Of
        #67: b := False;      {Выход по клавише <F9>}
      End;
    End;
  End;
End.

```

3.7.1. Библиотечный стандартный модуль Crt

Модуль Crt обеспечивает управление текстовым режимом работы экрана: перемещает курсор по экрану, изменяет цвет фона (экрана) и символов (знаков на экране), создает окна, управление звуком и др. Его назначение — создание и обновление различного рода окон, меню и других атрибутов диалоговых программ. Модуль состоит из подпрограмм, оформленных в виде библиотеки (модуля).

Модуль Crt выполняет различные процедуры и функции.

Процедура ClrScr — очищает экран или окно: экран заполняется цветом фона, а курсор устанавливается в верхний левый угол экрана или окна.

Процедура TextColor — определяет цвет выводимых символов.

Формат: TextColor(Color); Color: Byte.

Процедура TextBackground — определяет цвет фона.

Формат: TextBackground(Color); Color: Byte.

Окраска фона осуществляется после использования процедуры очистки экрана ClrScr!

Для установки цвета можно использовать соответствующие коды -константы цветов (табл. 3.3).

Таблица 3.3

Константы модуля Crt

Const	Константы цветов	Описание цвета
Black	= 0	Черный
Blue	= 1	Синий
Green	= 2	Зеленый
Cyan	= 3	Голубой
Red	= 4	Красный
Magenta	= 5	Фиолетовый
Brown	= 6	Коричневый
LightGray	= 7	Светло-серый
DarkGray	= 8	Темно-серый
LightBlue	= 9	Светло-синий
LightGreen	= 10	Светло-зеленый
LightCyan	= 11	Светло-голубой
LightRed	= 12	Розовый
LightMagenta	= 13	Светло-фиолетовый
Yellow	= 14	Желтый
White	= 15	Белый
Blink	= 128	Мерцание символа

После восстановления исходного цвета знаков на экране компьютера необходимо обратиться к процедуре Write/WriteLn, так как только в этом случае в специальную переменную TextAttr модуля Crt заносятся цветовые определения!

Пример 31. Написание текста желтыми буквами на синем фоне, приостановка работы, а затем восстановление черного цвета экрана и белого цвета знаков на экране.

```

Program 03_31;
Uses Crt;
Var
  mark: Char;
Begin
  TextBackground(Blue); TextColor(Yellow);
  ClrScr;
  WriteLn('Информатика');
  mark := ReadKey;
  TextBackground(Black); TextColor(White);
  ClrScr;
  WriteLn('Информатика');
  mark := ReadKey;
  ClrScr;
  mark := ReadKey;
End.

```

Пример 32. Показ цветовых возможностей экрана.

```

Program 03_32;
Uses Crt;
Var
  color: Byte;
Begin
  Repeat
    ReadLn(color);
    TextColor(color);
    WriteLn('Цвет знаков = ',color);
  Until Eof;
End.

```

Процедура Window — определяет размещение текстового окна на экране.

Формат: Window(x1,y1,x2,y2); x1,x2,y1,y2: Byte,

где x1, y1 — координаты левого верхнего угла окна;

x2, y2 — координаты правого нижнего угла окна.

Условия выполнения процедуры: $x_2 > x_1$, $y_2 > y_1$.

В процессе выполнения процедуры:

- курсор помещается в левом верхнем углу окна;
- окно заполняется цветом фона;
- курсор не выходит за пределы окна.

*Окно заливается заданным цветом после использования
процедуры ClrScr!*

Границы текущего окна запоминаются в двух глобальных переменных модуля Crt: WindMin — хранит x_1 и y_1 , WndMax — хранит x_2 и y_2 .

Пример 33. Создание белого окна на синем фоне.

```
Program 03_33;
Uses Crt;
Var
  mark: Char;
Begin
  TextBackground(Blue); TextColor(Yellow);
  ClrScr;
  Window(3,2,60,12);
  TextBackground(White); TextColor(Black);
  ClrScr;

  WriteLn('Окно в Информатику');
  mark := ReadKey;
  Window(1,1,80,25);
  TextBackground(Blue); TextColor(White);
  ClrScr;
  WriteLn('Окно во весь экран');
  mark := ReadKey;
  ClrScr;
  Window(1,1,80,25);
  TextBackground(Black); TextColor(White);
  ClrScr;
  WriteLn('Восстановление цвета экрана');
  ReadLn;
  mark := ReadKey;
End.
```

Процедура TextMode — задает соответствующий текстовый режим работы адаптера.

Формат: TextMode(Mode); Mode: Word, где Mode — код текстового режима:

BW40	= 0	Черно-белый режим 40x25
Co40	= 1	Цветной режим 40x25
BW80	= 2	Черно-белый режим 80x25
Co80	= 3	Цветной режим 80x25
Mono	= 7	Используется с MDA
Font8x8	= 256	Используется для загружаемого шрифта в режиме 80x43 или 80x50

*Код режима TextMode запоминается с помощью глобальной переменной
LastMode, при ее выполнении сбрасываются установки цвета и окон,
экран очищается, курсор устанавливается в верхнем левом углу экрана.*

Пример 34. Иллюстрация работы различных режимов процедуры TextMode.

```
Program 03_34;
Uses Crt;
Procedure Disp(s: String);
```

```

{Выводит на экран сообщение о режиме работы дисплея и ждет реакцию}
{пользователя}
Begin
  WriteLn(S);
  WriteLn('Нажмите клавишу <Enter>');
  ReadLn;
End;
Var
  LM: Word;
Begin
  LM := LastMode; {Запоминание режима работы дисплея}
  TextMode(Co40);
  Disp('Режим 40x25');
  TextMode(Co80);
  Disp('Режим 80x25');
  TextMode(Co40+Font8x8);
  Disp('Режим Co40+Font8x8');
  TextMode(Co80+Font8x8);
  Disp('Режим Co80+Font8x8');
  TextMode(LM); {Восстановление режима работы дисплея}
End.

```

Процедура GotoXY — переводит курсор в нужное место экрана или текущего окна.

Формат: GotoXY(X,Y); X,Y: Byte,

где X,Y — новые координаты курсора.

Пример 35. Образование трех окон с заголовками.

```

Program 03_35;
Uses Crt;
Var
  mark: Char;
Begin
  TextBackground(Blue); TextColor(White);
  ClrScr;
  GotoXY(28,1);
  WriteLn('Информационная система');
  mark := ReadKey;
  Window(3,2,58,12); {Белое окно}
  TextBackground(White); TextColor(Black);
  ClrScr; GotoXY(28,1); Write('Ввод');
  mark := ReadKey;
  Window(3,15,58,23); {Желтое окно}

  TextBackground(Yellow); TextColor(Blue);
  ClrScr; GotoXY(26,1); Write('Результат');
  mark := ReadKey;
  Window(62,2,76,12); {Голубое окно}
  TextBackground(Cyan); TextColor(Black);
  ClrScr; GotoXY(7,1); Write('Меню');
  mark := ReadKey;
  Window(1,1,80,25); {Очистка экрана}
  TextBackground(Black); TextColor(White);
  ClrScr;
  WriteLn;
End.

```

Процедура ClrEol — стирает часть строки от текущего положения курсора до правой границы окна (экрана). Положение курсора остается неизменным.

Процедура DelLine — уничтожает всю строку с курсором в текущем окне (на экране), все строки ниже удаляемой сдвигаются вверх на строку.

Процедура InsLine — вставляет пустую строку в месте расположения курсора, а нижние строки сдвигаются вниз на одну строку.

Процедура LowVideo — устанавливает пониженную яркость экрана.

Процедура NormVMeo — задает нормальную яркость экрана.

Процедура HighVideo — устанавливает повышенную яркость экрана.

Пример 36. Показ изменений яркости экрана при использовании различных процедур изменения яркости экрана.

```
Program 03_36;
Uses Crt;
Begin
  LowVideo;
  WriteLn('Пониженная яркость');
  NormVideo;
  WriteLn('Нормальная яркость');
  HighVideo;
  WriteLn('Повышенная яркость');
End.
```

Процедура **Sound** — выдает сигнал на встроенный динамик компьютера с заданной частотой звука. Формат: **Sound(F)**; F: Word; . Например: **Sound(1000)** .

Процедура NoSound — выключает динамик компьютера. Формат: **NoSound**; .

Процедура Delay — обеспечивает задержку работы программы на заданный интервал времени (в миллисекундах). Формат: **Delay(t)**; t: Word; . Например: **Delay(500)**; .

Пример 37. Показ с помощью динамика компьютера восходящей и нисходящей музыкальной гаммы.

```
Program 03_37;
Uses Crt;
Const
  F: Array[1..12] of Real = (130.8, 138.6, 146.8, 155.6, 164.8, 174.6,
                           185.0, 196.0, 207.7, 220.8, 233.1, 246.9);
                           {Массив частот 1-й октавы}
  Temp = 100;                {Темп исполнения}
Var
  k,n: Integer;
Begin
  {Восходящая гамма}
  For k := 0 To 3 Do
    For n := 1 To 12 Do
      Begin
        Sound(Round(F[n]*(1 shl k)));
        Delay(Temp);
        NoSound;
      End;
    For k := 3 DownTo 0 Do
      {Нисходящая гамма}
      For n := 12 DownTo 1 Do
        Begin
          Sound(Round(F[n]*(1 shl k)));
          Delay(Temp);
          NoSound;
        End;
      End;
End.
```

Функция ReadKey — возвращает значение символа клавиатуры (Char); форма записи **a := ReadKey**;

Ввод символа с помощью этой функции не сопровождается эхо-повтором.

Каждому символу клавиши клавиатуры в компьютере ставится в соответствие определенный код, состоящий из 2—3 цифр. Ряд функциональных клавиш имеют код, состоящий из дополнительной цифры. Такой код определяется как расширенный код клавиши.

Расширенный код клавиш — #0 + код клавиши символа (ASCII): к ним относятся клавиши <F1>—<F10>, <Ins>, <Home>, , <End>, <PgUp>, <PgDn>.

#0 используется для указания программе на генерацию расширенного кода, например: <Home> — 0 71, <PgUp> — 0 73, <PgDn> — 0 81, <Ins> — 0 82, . . .

Пример 38. Показ расширенного кода любой клавиши.

```

Program 03_38;
Uses Crt;
Var
  mark: Char;
Begin
  ClrScr;
  Repeat
    mark := ReadKey;
    If mark <> #0 Then
      WriteLn(Ord(mark))
    Else
      WriteLn('0', Ord(ReadKey):8);
  Until mark = #27;           {27 — расширенный код <Esc>}
End.

```

Функция KeyPressed — указывает состояние буфера клавиатуры:

False — буфер пуст;

True — в буфере есть хотя бы один символ, не прочитанный программой.

*Обращение к функции KeyPressed не задерживает
исполнения программы.*

Примеры использования функции KeyPressed для организации ожидания выхода из программы:

```

1: Uses Crt;
   Repeat
   ...
   Until KeyPressed; .

2: Uses Crt;
   Var
     C: Char;
   Begin
     While KeyPressed Do
       C := ReadKey;
     ...
   End.

```

Функция WhereX — возвращает текущую горизонтальную координату курсора.

Функция WhereY — возвращает текущую вертикальную координату курсора.

Например: GotoXY(X2-X1+1, WhereY); .

3.7.2. Библиотечный стандартный модуль Graph

Модуль Graph — библиотека графических подпрограмм, обеспечивающих использование цветовой и разрешающей способностей экрана.

Настройка графических процедур с конкретным адаптером достигается за счет подключения соответствующего графического драйвера.

Драйвер — специальная программа, осуществляющая управление соответствующими техническими средствами компьютера.

Для поддержки технических средств компьютера используются следующие драйверы:

- ◆ CGA.BGI — драйвер для CGA, MCGA (BGI — Borland Graphics Interfase);
- ◆ EGAVGA.BGI — драйвер для EGA, VGA;
- ◆ HERC.BGI — драйвер для монохромного Hercules;
- ◆ ATT.BGI — драйвер для AT&T6300 (400 строк);
- ◆ PC3270.BGI - драйвер для IBM 3270 PC;
- ◆ IBM8514.BGI — драйвер для IBM 8514 и др. Введем следующие понятия.

Графический режим работы экрана — реализация управления свечением совокупности близко расположенных точек — пикселей, светимость которых управляется с помощью программы.

Разрешающая способность экрана (разрешение экрана) — общее количество пикселей и количество цветов (оттенков), которыми может светиться каждый из них.

Графическая страница — область оперативной памяти компьютера (карта экрана), используемая для запоминания информации о светимости (цвете) каждого пикселя.

Различают следующие графические режимы работы для каждого типа адаптеров.

Адаптер CGA эмулирует 5 графических режимов, среди них режимы низкого разрешения экрана — 320x200 пикселей:

палитра 0: светло-зеленый, розовый, желтый + черный,

палитра 1: светло-голубой, светло-фиолетовый, белый + черный,
палитра 2: зеленый, красный, коричневый + черный,
палитра 3: голубой, фиолетовый, светло-серый + черный;
и высокого разрешения экрана — 640x200 при использовании двух цветов, при чем один из них — всегда черный.

Графический адаптер CGA использует одну страницу. Адаптер EGA эмулирует графические режимы CGA и режимы низкого разрешения: 640x200, 16 цветов, 4 страницы; высокого разрешения: 640x350, 16 цветов, 1 страница.

Адаптер VGA эмулирует режимы адаптеров CGA, EGA и режим высокого разрешения: 640x480, 16 цветов, 1 страница.

Адаптер SVGA эмулирует режим адаптера VGA, достигая режима разрешения 1024x768, 256 цветов (оттенков).

Для модуля Graph определены следующие процедуры и функции.

Управление графическим режимом

Процедура InitGraph — иницирует графический режим работы адаптера.

Формат:

```
InitGraph(Driver,Mode,'Path');
```

```
Var Driver, Mode: Integer;
```

```
Path: String; .
```

Переменные:

Driver — тип графического драйвера: Const. Диапазон значений:

Detect = 0; {Режим автоопределения типа драйвера}

CGA = 1;

MCGA = 2;

EGA = 3;

VGA = 9; .

Mode — режим работы графического адаптера. Значения Mode для EGA:

Const

EGALo = 0; {640x200, 16 цветов}

EGAHi = 1; {640x350, 16 цветов}

EGAMonoHi = 3; {640x350, 2 цвета}

AraVGA:

Const

VGALo = 0; {640x200}

VGAMed=1; {640x350}

VGAHi = 2; {640x480}

Path — полное имя файла драйвера (путь + имя файла).

Процедура SetGraphMode — устанавливает новый режим работы адаптера.

Формат: SetGraphMode(Mode: Integer); .

Mode — код устанавливаемого режима.

Например, обращение к процедуре InitGraph, использующей драйвер EGA.BGI, который находится в каталоге TP\BGI на диске C, и устанавливающей режим работы 640x350,16 цветов, следует показать следующим образом: Uses Graph; Var

```
Driver, Mode: Integer; Begin
```

```
Ddriver := EGA; {Драйвер}
```

```
Mode := EGAHi; {Режим работы}
```

```
InitGraph(Driver,Mode,'C:\TP\BGI');
```

```
...
```

При автоматическом определении типа драйвера в текст предыдущего примера следует внести изменения:

```
...
```

```
Driver := Detect;
```

```
InitGraph(Driver,Mode,'C:\TP');
```

```
...
```


При неопределенном значении Mode режим работы драйвера определяется максимальным значением константы Mode.

Процедура CloseGraph — восстанавливает текстовый режим работы экрана.

Формат: CloseGraph; .

Функция GraphResult — записывает результат обращения к графическим процедурам. Значение GraphResult = 0, если обращение успешно, иначе — отрицательное число — код ошибки. Например:

GraphResult = —1 означает, что не инициирован графический режим;

GraphResult = -2 означает, что не определен тип драйвера;

...

GraphResult = —14 — неправильный номер шрифта.

Наиболее частая ошибка — неправильно определено место расположения файла GRAPH.TPU, чтобы избежать ошибки, необходимо: в Турбо-среде в диалоговом окне Options/Directories в поле Unit Directories указать каталог, в котором расположен этот файл.

После обращения к функции GraphResult признак ошибки сбрасывается, и повторное обращение к ней вернет ноль.

Функция GraphErrorMsg — дает текстовое сообщение коду ошибки.

Формат: GraphErrorMsg(Code: Integer): String.

Code — код ошибки, возвращаемый функцией GraphResult.

Например:

Var

Error: Integer;

Begin

...

Error := GraphResult;

If Error <> grOk; { grOk = 0, если нет ошибок }

WriteLn(GraphErrorMsg(Error)); .

Управление цветом и палитрой

Процедура SetColor — устанавливает текущий цвет линий и символов.

Формат: SetColor(Color: Word); .

Color — цвет линий и символов.

В модуле Graph используются константы для задания цвета модуля Crt.

Процедура SetBkColor — устанавливает цвет фона экрана.

Формат: SetBkColor(Color: Word);

Color — цвет фона экрана.

Пример 39. Переход в графический режим и установка синего фона.

```
Program 03_39;
Uses Crt, Graph;
Var
  Driver, Mode, error: Integer;
  ch: char;
Begin
  Driver := Detect;
  InitGraph(Driver, Mode, 'C:\Pascal');
  SetGraphMode(0);
  SetBkColor(Blue);
  ch := ReadKey;
  CloseGraph;
End.
```

Построение фигур из линий

Процедура SetFillStyle — устанавливает стиль (тип и цвет) заполнения прямоугольника на экране.

Формат: SetFillStyle(Fill, Color: Word); .

Fill — тип заполнения (периодически повторяющийся узор), определяется значением константы, выбираемой из следующего ряда:

```
const
EmptyFill = 0; { Фоном }
SolidFill = 1; { Сплошное }
LineFill = 2; { - - - - }
LtSlashFill = 3; { \ \ \ \ \ }
SlashFill = 4; { Утолщенными /// }
BkSlashFill = 5; { Утолщенными \\ \ }
LtBkSlashFill = 6; { \ \ \ \ \ \ }
HatchFill = 7; { + + + + + + + }
XhatchFill = 8; { x x x x x x x x }
InterLeaveFill = 9; { Прямоугольная клеточка }
WideDotFill = 10; { Редкими точками }
CloseDotFill = 11; { Частыми точками }
UserFill = 12;
{ Определяется пользователем при обращении к процедуре }
{ SetFillPattern }
Color — цвет заполнения.
```

Процедура Rectangle — вычерчивает прямоугольник с указанными координатами углов.
Формат: Rectangle(x1,y1,x2,y2: Integer); .

Прямоугольник вычерчивается с использованием текущего цвета и текущего стиля линий.

Процедура Bar — заполняет прямоугольную область экрана.

Формат: Bar(x1,y1,x2,y2); .

x1, y1 — координаты левого верхнего угла экрана;

x2, y2 — координаты правого нижнего угла экрана.

Процедура закрашивает прямоугольник текущим образцом узора и текущим цветом, установленным процедурой SetFillStyle.

Пример 40. Показ на экране стандартных типов заполнения.

```
Program 03_40;
Uses Crt; Graph;
Var
  d,r,e,k,j,x,y: Integer;
Begin
  d := Detect;
  InitGraph(d,r,"");
  e := GraphResult;
  If e <> grOk Then
    WriteLn(GraphErrorMsg(e))
  Else
    Begin
      x := GetMaxX Div 6;
      y := GetMaxY Div 5;
      For j := 0 To 2 Do
        For k := 0 To 3 Do Begin
          Rectangle((k+1)*x,(j+1)*y,(k+2)*x,(j+2)*y);
          SetfillStyle(k+j*4,j+1);
          Bar((k+1)*x+1,(j+1)*y+1,(k+2)*x-1,(j+2)*y-1);
        End;
      If ReadKey = #0 Then k := Ord(Readkey);
      CloseGraph;
    End;
  End.
```

Пример 41. Показ различной закрашки прямоугольников на экране.

```

Program 03_41;
Uses Graph, Crt;
Var
  d, r, e: Integer;
Begin
  d := Detect;
  InitGraph(d, r, 'C:\TP7');
  d := GetMaxX Div 8;
  r := GetMaxY Div 8;
  Rectangle(d,r,7*d,7*r);
  SetViewPort(d+1,r+1,7*d-1,7*r-1,ClipOn);
  Repeat
    SetFillStyle(Random(12), Random(succ(GetMaxColor)));
    Bar(Random(GetMaxX), Random(GetMaxY),
    Random(GetMaxX), Random(GetMaxY));
    Delay(500);

  Until KeyPressed;
  CloseGraph;
End.

```

Процедура Bar3D — трехмерное изображение параллелепипеда с закрасленной передней гранью.
 Формат: Bar3D(X1,Y1,X2,Y2,Depth: Integer; Top: Boolean); .

X1,Y1 — координаты левого верхнего угла экрана;

X2,Y2 — координаты правого нижнего угла экрана;

Depth — глубина изображения (третье измерение);

Top — способ изображения верхней грани; определяется одной из констант:

Const

TopOn = True — верхняя грань параллелепипеда вычерчивается,

TopOff= False — без верхней грани.

Используется текущий стиль линий SetLineStyle, текущий цвет SetColor, передняя грань заливается текущим стилем заполнения SetFillStyle.

Пример 42. Иллюстрация возможностей применения процедуры Bar3D.

```

Program 03_42;
Uses Crt, Graph;
Var
  d,r: Integer;
Begin
  d := Detect;
  InitGraph(d,r,"");
  Bar3D(80,100,120,180,15,TopOn);           {С верхней гранью}
  OutTextXY(80,190,'TopOn'); Delay(1000);
  Bar3D(250,150,290,180,15,TopOff);       {Без верхней грани}
  OutTextXY(250,190,'TopOff'); Delay(1000);
  Bar3D(430,50,450,150,15,TopOn);         {«Стоит» на следующем}
  OutTextXY(400,160,'Press Enter'); ReadLn;
  Bar3D(390,150,495,180,15,TopOn);
  Delay(1000);
  Bar3D(230,350,320,450,15,TopOff);       {Без верхней грани}
  OutTextXY(230,300,'Press Enter'); ReadLn;
  SetLineStyle(3,0,1);
  SetFillStyle(LtSlashFill, Yellow);
  Bar3D(230,250,320,350,15,TopOn);       {«Поставлен» сверху}

  ReadLn;
  CloseGraph;
End.

```

Работа с линиями

Процедура DrawPoly — изображает произвольную ломаную линию, заданную координатами точек излома.

Формат: DrawPoly(N: Word; Var Points); .

N — количество точек излома, включая обе крайние;

Points — переменная типа PointType, содержащая координаты точек излома.

Например:

Type

```
PointType = Record
```

```
  x,y: Word;
```

```
End; .
```

При использовании процедуры применяется текущий цвет и стиль линии.

Пример 43. Изображение на экране графика функции синуса.

```
Program 03_43;
Uses Graph;
Const N = 100;
Var
  d,r: Integer;
  m: Array [0..N+1] Of PointType;
  k: Word;
Begin
  d := Detect;
  InitGraph(d,r,"");
  For k := 0 To N Do      {Вычисление координат графика}
    With m[k] Do
      Begin
        x := Trunc(k*GetMaxX/N);
        y := Trunc((GetMaxY-20)*(-Sin(2*Pi*k/N)+1)/2) + 10;
      End;
    m[succ(N)].x := m[0].x;      {Прямая линия на графике}
    m[succ(N)].y := m[0].y;
  DrawPoly(N+2,m);
  ReadLn;
  CloseGraph;
End.
```

Процедура Line — изображает линию между указанными координатами начала и конца.

Формат: Line(X1,Y1,X2,Y2: Integer); .

X1, Y1 — координаты начала линии;

X2, Y2 — координаты конца линии.

Процедура LineTo — изображает линию от текущего положения указателя до точки с указанными координатами.

Формат: LineTo(X,Y: Integer); .

Процедура LineRel — изображает линию от текущего положения указателя до точки, заданной приращениями его координат.

Формат: LineRel(DX,DY: Integer); .

DX,DY — приращения координат.

В процедурах Line, LineTo, LineRel линии вычерчиваются текущим стилем и текущим цветом.

Пример 44. Изображение на правой стороне экрана случайных линий случайным цветом, а на левой — разноцветных точек.

```

Program 03_44;
Uses Graph, Crt;
Var x1,x2,y1,y2, DR,Mo: Integer;
    ch := Char;
Begin
  DR := detect;
  InitGraph(DR,Mo,"");
  SetBkColor(black); SetColor(Blue);
  x1 := 1; y1 := 1; x2 := GetMaxX Div 2-2; y2 := GetMaxY;
  Bar(1,1,GetMaxX-1,GetMaxY-1);
  Bar(x1+2,y1+2,GetMaxX Div 2-2,GetMaxY);
  SetViewPort(x1+1,y1-1,x2-1,y2-1,Clipon);
  While Not KeyPressed Do
    Begin
      SetColor(Random(GetMaxColor));
      MoveTo(200*Round(Abs(Sin(x1))),150*Round(Abs(Cos(Y1))));
      LineTo(Random(getmaxx),Random(GetMaxY));
    End;
  ch := ReadKey;
  Bar(GetMaxX Div 2+2,1,GetMaxX-2,GetMaxY-2);
  SetViewPort(GetMaxX Div 2+3,1,GetMaxX-3,GetMaxY-3,Clipon);
  Repeat
    x1 := Random(x2-x1); y1 := Random(GetMaxY);
    PutPixel(x1,y1,Random(GetMaxColor));
  Until KeyPressed;
  CloseGraph;
End.

```

Процедура SetLineStyle — устанавливает новый стиль вычерчивания линий.

Формат: SetLineStyle(Type,Pattern,Thick: Word); .

Type — тип линии, определяется константой из следующего ряда:

Const

SolidLn = 0; {Сплошная линия}

DottedLn = 1; {Точечная линия}

CenterLn = 2; {Штрих-пунктирная линия}

DashedLn = 3; {Пунктирная линия}

UserBitLn = 4. {Узор линии определяет пользователь}

Pattern — образец линии: состоит из двух байтов, каждый бит которых соответствует светящемуся пикселю в линии (16 пикселей); этот отрезок повторяется по всей длине линии. Pattern указывается только для линий, вид которых определяет пользователь (например, для Type = UserBitLn).

Thick — толщина линии, параметр может принимать два значения:

Const

NormWidth = 1; {Толщина в один пиксель}

ThickWidth = 3. {Толщина в три пикселя}

Пример 45. Показ образцов стандартных стилей и линий, задаваемых пользователем.

```

Program 03_45;
Uses Crt, Graph;
Const
  style: Array[0..4] Of String[9] = ('SolidLn','DottedLn', 'CenterLn', 'DashedLn',
  'UserBitLn');
Var
  d,r,e,i,j,dx,dy: Integer;
  p: Word;
Begin
  d := Detect;
  InitGraph(d, r, "");
  e := GraphResult;
  If e <> grOk Then WriteLn(GraphErrorMsg(e))
  Else
    Begin

```

```

dx := GetMaxX Div 6;
dy := GetMaxY Div 10;
For j := 0 To 1 Do           {Вывод стандартных линий}
  Begin                     {Для двух толщин}
    For i := 0 To 3 Do     {Четыре типа линий}
      Begin
        SetLineStyle(i,0,j*2+1);
        Line(0,(i+j*4+1)*dy,dx,(i+j*4+1)*dy);
        OutTextXY(dx+10,(i+j*4+1)*dy,style[i]);
      End;
    End;
  j := 0;                   {Вывод образца и изображение линии}
  dy := (GetMaxY+1) Div 25;
  Repeat
    OutTextXY(320,j*dy, 'Pattern: ');
    GotoXY(50,j+1);
    ReadLn(p);              {Введите 1, 7, 127, 65520}
    If p <> 0 Then
      Begin
        SetLineStyle(UserBitLn,p, NormWidth);
        Line(440,j*dy+4,600,j*dy+4);
        Inc(j);
      End;
    Until p = 0;
  CloseGraph;
End;
End.

```

Процедура SetWriteMode — устанавливает способ взаимодействия вновь выводимых линий с уже существующим на экране изображением.

Формат: SetWriteMode(Mode: Integer); .

Mode = 0 — выводимые линии накладываются на существующее изображение;

Mode = 1 — накладываемая линия не изменяет ее изображение.

Для задания параметра Mode можно использовать модульные константы:

Const

CopyPut = 0;

XorPut = 1;2.

Пример 46. Имитация секундомера.

```

Program 03_46;
Uses Graph, Crt;

```

```

Var
  d,r,i,x0,y0,x1,y1,x2,y2: Integer;
  xasp,yasp: Word;
Const
  dr = 0.9;
Begin
  d := detect; InitGraph(d,r,'C:\TP');
  i := GraphResult;
  If i<>grOk Then
    WriteLn(grapherrormsg(i))
  Else
    Begin
      SetColor(Yellow);
      x0 := getmaxx Div 2;
      y0 := getmaxy Div 2;
      Getaspratio(xasp,yasp);
      r := y0;
      Circle(x0,y0,r); Circle(x0,y0,round(r*dr));
      For i := 0 To 59 Do
        Begin
          x1 := x0+Round(dr*r*sin(2*i*pi/60));
          x2 := x0+Round(r*sin(2*pi*i/60));
          y1 := y0-Round(dr*r*xasp*cos(2*pi*i/60)/yasp);
          y2 := y0-Round(r*xasp*cos(2*pi*i/60)/yasp);
          Line(x1,y1,x2,y2);
        End;
      SetFillStyle(7,9);
      FloodFill(x0,y0,Yellow);
      SetTextStyle(4,0,5);
      OutTextXY(236,250,'Pascal');
      OutTextXY(300,40,'12');
      OutTextXY(310,390,'6');
      OutTextXY(120,220,'9');
      OutTextXY(490,220,'3');
      SetWriteMode(XorPut);
      Repeat
        For i := 0 To 59 Do
          If Not KeyPressed Then
            Begin
              x2 := x0+Round(dr*r*Sin(2*i*Pi/60));
              y2 := y0-Round(dr*r*xasp*Cos(2*Pi*i/60)/yasp);
              Line (x0,y0,x2,y2);
              Delay(1000);
              Line(x0,y0,x2,y2);
            End;
          Until KeyPressed;
        End;
      End;
    End.

```

Пример 47. Изображение процесса «приземления» аппарата. Суть задачи заключается в том, чтобы показать перемещение геометрической фигуры (ромба) на экране сверху вниз, слева направо.

```

Program 03_47;
Uses Graph, Crt;
Label 1;
Var
  x,dt,mo,c,i,y1: Integer;
  y: Real;
Procedure Tar(x,y,c: Integer);
Begin
  SetColor(c);
  MoveTo(x,y);
  Linerel(50,15);
  Linerel(-50,15);
  Linerel(-50,-15);
  Linerel(50,-15);
  Line(x+10,y+25,x+15,y+35);
  Line(x-10,y+25,x-15,y+35);
End;
Begin
  dt := detect;
  Initgraph(dt,mo,'C:/TP');
  SetColor(2); SetBkColor(0);
  Line(0,340,640,340);
  SetFillStyle(4,2);
  FloodFill(1,341,2);
  y := 0;
  For i := 1 To 2 Do
    Begin
      For x := 50 To 590 Do
        Begin
          Tar(x,round(y),5);
          Delay(2);
          Tar(x,round(y),0);
          y := y + 0.1;
          If KeyPressed Then Goto 1;
        End;
      End;
    End;
  End;

```



```

    For x := 590 DownTo 50 Do
        Begin
            Tar(x,round(y),5);
            Delay(2);
            Tar(x,round(y),0);
            If KeyPressed Then Goto 1;
            y := y+0.1;
        End;
    End;
For x := x To 320 Do
    Begin
        Tar(x,round(y),5);
        Delay(2);
        Tar(x,round(y),0);
    End;
For y1 := round(y) To 305 Do
    Begin
        Tar(x,y1,5);
        Delay(10);
        Tar(x,y1,0);
    End;
1: Tar(320,305,5);
    For i := 1 To 500 Do
        Begin
            Sound(i); Delay(10);
        End;
    For i := 1 To 20 Do
        Begin
            Delay(20);
            Sound(500);
            Delay(40);
            NoSound;
        End;
    Repeat Until KeyPressed;
    CloseGraph;
End.

```

Процедура MoveTo — устанавливает новое текущее положение указателя.

Формат: MoveTo(X,Y); .

X, Y — координаты левого верхнего угла окна (экрана).

Процедура MoveRel — устанавливает новое положение указателя в относительных координатах.

Формат: MoveRel(DX,DY: Integer); .

DX, DY — приращения новых координат указателя соответственно по горизонтали и вертикали.

Приращения задаются относительно текущего положения курсора!

Пример 48. Рисование белого окна с рамкой на синем фоне.

```

Program 03_48;
Uses Crt, Graph;
Const
  x11 = 26; y11 = 38; x12 = 564; y12 = 270; {Окно 1}
Var
  Driver, Mode, Error: Integer;
  ch: Char;
Procedure Window1(x1,y1,x2,y2: Integer);
Begin
  SetFillStyle(1,Cyan);
  Bar(x1,y1,x2,y2);
  SetColor(LightGray);
  SetLineStyle(0,0,3);
  Rectangle(x1-3,y1-3,x2+3,y2+3);
  MoveTo(x1+250,y1+5);
  SetColor(DarkGray);
  OutText('Input');
End;
Begin
  driver := Detect;
  InitGraph(Driver, Mode, "");
  SetGraphMode(1);
  SetBkColor(Blue);
  SetColor(White);
  MoveTo(220,10);
  OutText('Information system');
  Window1(x11,y11,x12,y12);
  ch := ReadKey;
  CloseGraph;
End.

```

Управление экраном, окном, страницей

Процедура ClearDevice — очищает графический экран и заполняет фон цветом, заданным процедурой SetBkColor.

Формат: ClearDevice; .

Процедура ClearViewPort — очищает графическое окно и заполняет цветом из текущей палитры, если окно не задано — очищает весь экран.

Формат: ClearViewPort; .

Процедура GetAspectRatio — возвращает два числа, позволяющих оценить соотношение сторон экрана.

Формат: GetAspectRatio(VarX, Y: Word); .

Процедура используется для построения правильных геометрических фигур: окружностей, квадратов и т. п. Например, для построения квадрата со стороной в L пикселей по вертикали следует записать:

```

GetAspectRatio(Xasp, Yasp);
Rectangle(x1,y1,x1+L*round(Yasp^asp),y1+L); .

```

Если L определяет длину квадрата по горизонтали, то следует записать:

```

Rectangle(x1,y1,x1+L,y1+L*round(Xasp^asp)); .

```

Процедура SetAspectRatio — устанавливает масштабный коэффициент отношения сторон графического экрана.

Формат: SetAspectRatio(X, Y: Word); .

X, Y — устанавливаемые соотношения сторон.

Пример 49. Показ 30 окружностей с разными соотношениями сторон экрана.

```

Program 03_49;
Uses Crt, Graph;
Const
  R = 50;
  dx = 1000;
Var
  d,m,e,k: Integer;
  Xasp,Yasp: Word;
Begin
  d := Detect;
  InitGraph(d,m,"");
  GetAspectRatio(Xasp,Yasp);
  For k := 0 To 30 Do
    Begin
      SetAspectRatio(Xasp+k*dx,Yasp);
      Circle(GetMaxX Div 2,GetMaxY Div 2,R);
      Delay(300);
    End;
  ReadLn;
  CloseGraph;
End.

```

Построение криволинейных фигур

Процедура Ellipse — изображает эллипсную дугу.

Формат: Ellipse(X,Y: Integer; BegA,EndA,RX,R Y: Word); .

X, X — координаты центра;

BegA, EndA — начальный и конечный углы дуги;

RX, R Y — горизонтальный и вертикальный радиусы эллипса в пикселях.

Пример 50. Показ эллипсов при разных отношениях радиусов.

```

Program 03_50;
Uses Crt, Graph;
Var
  d,r,e: Integer;
  xa,ya: Word;
Begin
  d := Detect;
  InitGraph(d,r,"");
  OutTextXY(30,100,'RX = RY, 180 — 90');
  Line(10,200,160,200); Line(80,135,80,265);
  Ellipse(80,200,180,90,50,50);
  Delay(1000);
  OutTextXY(245,100,'RX = 5*R Y, 360 ');
  Line(190,200,410,200); Line(300,135,300,265);
  Ellipse(300,200,0,359,100,20);
  Delay(1000);
  OutTextXY(455,100,'AspectRatio, 0 — 270');
  Line(440,200,600,200); Line(520,135,520,265);
  GetAspectRatio(Xa,Ya);
  Ellipse(520,200,0,270,50,Round(50*(Xa/Ya)));
  ReadLn;
  CloseGraph;
End.

```

Процедура Arc — изображает дугу окружности.

Формат: Arc(X,Y: Integer; BegA,EndA,R: Word); .

X, X — координаты центра;

BegA, EndA — начальный и конечный углы дуги;

R — радиус.

Углы отсчитываются против часовой стрелки! Нулевой угол соответствует горизонтальному вектору слева направо.

Пример 51. Изображение дуг в осях XY

```

Program 03_50;
Uses Crt, Graph;
Var
  d,r,e: Integer;
  Xasp, Yasp: Word;
Begin
  d := Detect;
  InitGraph(d,r,"");
  GetAspectRatio(Xasp, Yasp);           {Отношение сторон экрана}
  r := Round(Yasp*GetMaxY/5/Xasp);     {r=1/5 от верт. размера экрана}
  d := GetMaxX Div 2;                   {Смещение второго графика}
  e := GetMaxY Div 2;                   {Положение горизонтальной оси}
  Line(0,e,r*5 Div 2,e);                {Горизонтальная ось левого графика}
  Line(5*r Div 4,e Div 2,5*r Div 4,3*e Div 2);
  Arc(5*r Div 4,e,0,90,R);              {Дуга 0–90 градусов}
  OutTextXY(r+60,e+e Div 8,'0–90');
  Line(d,e,d+5*r Div 2,e);              {Правый график}
  Line(d+5*r Div 4,e Div 2, d+5*r Div 4,3*e Div 2);
  Arc(d+5*r Div 4, e,270,540,R);
  OutTextXY(d,e+e Div 8, «270–540»);
  ReadLn;
  CloseGraph;
End.

```

Процедура Circle — вычерчивает окружность.

Формат: Circle(X,Y: Integer; R: Word); .

X, Y — координаты центра;

R — радиус в пикселях в горизонтальном положении.

Окружность выводится текущим цветом, толщина линий — текущим стилем, вид линии — всегда SolidLn (сплошная).

Пример 52. Заполнение окна окружностями различных радиусов.

```

Program 03_52;
Uses Crt, Graph;
Var
  x1,y1,x2,y2,Err: Integer;
Begin
  x1 := Detect;
  InitGraph(x1,x2,"");
  Err := GraphResult;
  If Err <> grOk Then
    WriteLn(GraphErrorMsg(Err))
  Else
    Begin
      {Определение координат окна с учетом разрешения экрана}
      x1 := GetMaxX Div 4;
      y1 := GetMaxY Div 4;
      x2 := 3*x1;
      y2 := 3*y1;
      Rectangle(x1,y1,x2,y2);          {Установка окна}
      Setviewport(x1+1,y1+1,x2-1,y2-1,ClipOn);
      Repeat {Заполнение случайными окружностями}
        Circle(Random(GetMaxX),RanDom(GetMaxX),
          RanDom(GetMaxX Div 5));
      Until KeyPressed;
      ClearViewPort;                   {Очистка окна}
      OutTextXY(0,0,'Press <Enter> . . . ');
      ReadLn;
      CloseGraph;
    End;
  End.

```

Пример 53. Изображение случайных окружностей и заполнение их случайным цветом.

```

Program 03_53;
Uses Crt, Graph;
Var
  d,r,e,x,y: Integer;
Begin
  d := Detect;           {Инициация графики}
  InitGraph(d,r, "");
  x := GetmaxX Div 7;    {Изображение окна}
  y := GetmaxY Div 7;
  Rectangle(x,y,6*x,6*y);
  SetViewport(x+1,y+1,6*x-1,6*y-1,ClipOn);
  Repeat                {Изображение окружностей}
    SetColor(succ(Random(White))); {Цвет линии}
    SetLineStyle(0,0,2*Random(2)+1); {Стиль линии}
    x := Random(GetMaxX); {Координаты центра}
    y := Random(GetMaxY);
    Circle(x,y,Random(GetMaxY Div 4)); {Окружность}
    Delay(50);
  Until KeyPressed;
  CloseGraph;
End.

```

Работа с точками

Процедура PutPixel — выводит заданным цветом точку по указанным координатам.

Формат: PutPixel(X,Y: Integer; Color: Word); .

X, Y — координаты точки, задаются относительно левого верхнего угла окна;

Color — цвет точки.

Пример 54. Изображение звездного неба. Зажигание, а затем погашение случайным образом тысячи «звезд».

```

Program 03_54;
Uses Crt, Graph;
Type
  PixelType = Record
    x,y: Integer;
  End;
Const
  N = 1000;           {Число зажигаемых звезд}
Var
  d,r,e,k,s: Integer;
  x1,y1,x2,y2: Integer;
  a: Array [1..N] Of PixelType; {Координаты звезд}
Begin
  d := Detect;           {Инициация графики}
  InitGraph(d,r,"");
  e := GraphResult;
  If e <> grOk Then
    WriteLn(GraphErrorMsg(e))
  Else
    Begin
      x1 := GetMaxX Div 5;   {Окно в центре экрана}
      y1 := GetMaxY Div 5;
      x2 := 4*x1;
      y2 := 4*y1;
      Rectangle(x1,y1,x2,y2);
      SetViewport(x1+1,y1+1,x2-1,y2-1,ClipOn);
      For k := 1 To N Do     {Массив координат всех звезд}
        With a[k] Do
          Begin

```

```

        x := Random(x2-x1);
        y := Random(y2-y1);
    End;
Repeat
    s := N;                                {ЦИКЛ ВЫВОДА}
    For k := 1 To N Do
        Begin
            With a[k] Do {Возникает звезда}
                Putpixel(x,y,White);
            Delay(10);
            If Not KeyPressed Then
                Begin
                    With a[s] Do          {Гаснет звезда}
                        PutPixel(x,y,Black);
                        Delay(10);
                        s := s-1;
                    End;
                End;
            Delay(100);
        Until KeyPressed;
        While KeyPressed Do k := Ord(Readkey);
    CloseGraph;
End;
End.

```

Функция GetPbcel — возвращает цвет пикселя с указанными координатами.

Формат: GetPixel(X,Y: Integer): Word; .

X, Y — координаты пикселя.

Функции GetMaxX и GetMaxY — возвращают значения типа Word, содержащие максимальные значения координат экрана по горизонтали и вертикали. Их действие можно показать на следующем примере:

Uses Graph;

Var

a,b: Integer;

Begin

a := Detect;

InitGraph(a,b,"");

WriteLn(GetMaxX, GetMaxY:5);

ReadLn;

CloseGraph;

End.

Работа с текстом

Процедура OutText — выводит текстовую строку, начиная с текущего положения указателя.

Формат: OutText(Txt: String); .

Txt — выводимая строка.

Процедура OutTextXY — выводит строку, начиная с заданного места.

Формат: OutTextXY(X,Y: Integer; Txt: String); .

X, Y — координаты точки вывода;

Txt — выводимая строка.

Процедура SetTextStyle — устанавливает стиль текстового вывода на графический экран.

Формат: SetTextStyle(Font,Direct,Size: Word); .

Font — код (номер) шрифта, определяется константой из следующего ряда:

Const

DefaultFont = 0; {Точечный шрифт 8x8}

TriplexFont = 1; {Утроенный шрифт TRIP.CHR}

SmallFont = 2; {Уменьшенный шрифт LITT.CHR}

SansSerifFont = 3; {Прямой шрифт SANS.CHR}

GothicFont = 4. {Готический шрифт GOTH.CHR}

Direct — код направления, определяется константой 0 или 1:

Const

HorizDir = 0; { Слева направо }

VertDir = 1. { Снизу вверх }

Size — код размера шрифта, выбирается из диапазона (1.. 10), минимально различимый шрифт соответствует коду 4.

Пример 55. Демонстрация типов шрифтов, изменения размеров символов и проведения операции масштабирования.

```
Program 03_55;
Uses Graph, Crt;
Var
  Driver, Mode: Integer;
Begin
  Driver := detect;
  InitGraph(Driver, Mode, 'C:\');
  If GraphResult <> grOk Then
    WriteLn(grapherrormsg(GraphResult))
  Else
    Begin
      SetColor(Yellow);
      SetBkColor(Blue);
      SetTextStyle(0,0,3);
      OutTextXY(10,10,'Defaultfont, font-size 3');
      SetTextStyle(1,0,3);
      OutTextXY(10,40,'TriplexFont Pascal, font-size 3');
      SetTextStyle(2,0,3);
      OutTextXY(10,70,'Smallfont Pascal, font-size 3');
      SetTextStyle(3,0,3);
      OutTextXY(10,100,'SansSerifFont Pascal, font-size 3');
      SetTextStyle(4,0,3);
      OutTextXY(10,130,'GothicFont Pascal, font-size 3');
      SetTextStyle(Defaultfont,0,6);
      OutTextXY(10,160,'DefaultFont, font-size 6');
      SetTextStyle(GothicFont,0,6);
      OutTextXY(10,220,'GothicFont, font-size 6');
      SetUserCharSize(3,2,1,2);
      {Ширина символов увеличена в 2 раза}
      OutTextXY(10,250,'Pascal, after varying');
      Repeat Until KeyPressed;
    End;
End.
```

Показ круговых диаграмм

Методика показа круговых диаграмм будет ясна после рассмотрения следующего примера.

Пример 56. Показ круговой диаграммы

```
Program 03_56;
Uses Crt, Graph;
Const n = 8; alfan = 0; xn = 320; yn = 200; r = 80; pi = 3.1415; ns = 5; ms = 2;
Type m=Array[1..n] Of Integer;
      m1=Array[1..n] Of Real;
Var
  y: m1; alfa: m;
  driver,err,k,i,j,bet,z,x: Integer;
  s: Real;
  st: String;
Begin
  ClrScr;
  Write('Укажите размер массива: '); ReadLn(k);
  s := 0; j := 1;
  For i := 1 To k Do
    Begin
```

```

Write('Введите ',i,'-й элемент массива: ');
Read(y[i]);
s:=s+y[i];
End;
driver := Detect;
InitGraph(driver,err,"");
alfa[1] := alfan; alfa[k+1] := alfan+360;
For i := 2 To k+1 Do
  Begin
    alfa[i] := alfa[i-1]+Round(y[i-1]/s*360);
    SetColor(1);
    OutTextXY(5,5,'Для продолжения нажмите клавишу <Enter>');
    ReadLn;
    PiesLice(xn,yn,alfa[i-1],alfa[i],r);
    bet := alfa[i-1]+(alfa[i]-alfa[i-1]) Div 2;
    x := xn+Round(r*Cos(bet*Pi/180));
    z := yn-Round(r*Sin(bet*Pi/180));
    j := j+1;
    SetFillStyle(j,1);
    If((bet>=0)And(bet<=90))Or((bet>=270)And(bet<=360))
      Then x := x+10
    Else x := x-65;
    If ((bet>=0)And(bet<=180)) Then
      Begin
        z := z-10;
        FloodFill(x,z+11,0);
      End
    Else
      Begin
        FloodFill(x,z-3,0);
        z := z+10;
      End;
    Str(y[i-1]:5:2,st);
    OutTextXY(x,z,st);
  End;
ReadLn;
CloseGraph;
End.

```

Обмен с памятью

Процедура GetImage — помещает в память копию фрагмента изображения.

Формат: GetImage(X1,Y1,X2,Y2: Integer; Var: Buf); .

Buf — переменная, куда будет помещена копия фрагмента изображения.

Процедура PutImage — выводит в заданное место экрана изображение, ранее запомненное процедурой GetImage.

Формат: PutImage(X,Y: Integer; Var: Buf; Mode: Word); .

X, Y — координаты левого верхнего угла экрана, где будет помещен фрагмент;

Buf — имя переменной, где хранится фрагмент;

Mode — способ взаимодействия нового изображения с имеющимся на экране изображением, он определяется значением одной из констант:

Const

NormalPut = 0 — замена текущего изображения на копию;

XorPut = 1 — исключаящее ИЛИ; стирает место на экране, откуда была взята копия, при повторе изображение восстанавливается;

OrPut = 2 — объединительное ИЛИ;

AndPut = 3 — логическое И;

NotPut = 4 — инверсия изображения: копия в инверсном виде.

Пример 57. Показ «летающей тарелки» на звездном небе экрана.


```

Program 03_57;
Uses Crt, Graph;
Const
  r = 20;           {Размер НЛО}
  pause = 100;     {Скорость перемещения}
Var
  d,m,e,xm,ym,x,y,lx,ly,rx,ry,i,Size,dx,dy,Width,Height: Integer;
  Saucer: Pointer;
  Label
  Loop;
Begin
  d := Detect;           {Инициация графики}
  InitGraph(d,m,'c:\tp5');
  x := r*5; y := r*2;
  xm := GetMaxX Div 8;   {Единица размера экрана}
  ym := GetMaxY Div 8;
  SetFillStyle(0,Red);
  Ellipse(x,y,0,360,r,r Div 3+2); {Летающая тарелка}
  Ellipse(x,y-4,190,357,r,r Div 3);
  Line(x+7,y-6,x+10,y-12);
  Line(x-7,y-6,x-10,y-12);

  Circle(x+10,y-12,2);
  Circle(x-10,y-12,2);
  FloodFill(x+1,y+4,White);
  lx := x-r-1;          {Габариты тарелки}
  ly := y-14;

  rx := x+r+1;
  ry := y+r Div 3+3;
  Width := rx - lx + 1;
  Height := ry - ly + 1;
  Size := ImageSize(lx,ly,rx,ry);   {Запись изображения}
  GetMem(Saucer,Size);
  GetImage(lx,ly,rx,ry,Saucer^);
  PutImage(lx,ly,Saucer^,XorPut);   {Очистка изображения тарелки}
  Rectangle(xm,ym,7*xm,7*ym); {Звездное небо}
  SetViewport(xm+1,ym+1,7*xm+1,7*ym-1,ClipOn);
  xm := 6*xm;
  ym := 6*ym;
  For i := 1 To 200 Do
    PutPixel(Random(xm),Random(ym),White);
  x := xm Div 2;          {Начальное положение НЛО}
  y := ym Div 2;          {и направление движения}
  dx := 10;
  dy := 10;
  Repeat                  {Цикл движения}
    PutImage(x,y,Saucer^,XorPut); {Изображение на новом месте}
    Delay(pause);           {Пауза}
    PutImage(x,y,Saucer^,XorPut);
    {Стирание изображения на новом месте}
  loop: x := x+dx;
  y := y+dy;
  If (x < 0) Or (x + Width + 1 > xm) Or (y < 0) Or (y + Height + 1 > ym) Then
{Проверка границы экрана}
    Begin                {Изменение направления движения}
      x := x - dx;
      y := y - dy;
      dx := GetMaxX DIV 10 - Random(GetMaxX Div 5);
      dy := GetMaxY DIV 30 - Random(GetMaxY Div 15);
      Goto loop;
    End;
  Until KeyPressed;
  CloseGraph;
End.

```

Функция ImageSize — выдает размер памяти в байтах, необходимый для запоминания прямоугольного фрагмента изображения.

Формат: ImageSize(X1,Y1,X2,Y2: Integer): Word; .

X1, Y1, X2, Y2 — координаты верхнего левого и правого нижнего углов фрагмента изображения.

3.7.3. Библиотечный стандартный модуль System

Модуль System представляет собой основную библиотеку стандартных подпрограмм Турбо Паскаль, без которой не может быть выполнена ни одна программа.

Для модуля System определены следующие процедуры и функции.

Процедура Randomize — иницирует встроенный генератор случайных чисел на заданном интервале целых чисел.

Процедура Exit — осуществляет немедленный выход из программы.

Процедура Halt — прекращает выполнение программы.

Процедура Inc(x [;n]) — увеличивает (инкрементирует) переменную на n. Например:

```
Inc(IntVar);
```

```
Inc(LongintVar, 5); .
```

Процедура Dec(x [;n]) — уменьшает (декрементирует) переменную на n; например:

```
Dec(IntVar);
```

```
Dec(LongintVar, 5); .
```

Функция Abs(x) — возвращает абсолютное значение от X (модуль от X): тип результата соответствует типу аргумента; например:

```
Var
```

```
  r: Real;
```

```
  i: Integer;
```

```
Begin
```

```
  r := Abs(-2.3); {Значение r = 2.3}
```

```
  i := Abs(-157); {Значение r = 157}
```

```
End.
```

Функция ArcTan(x: Real): Real — вычисляет значение арктангенса аргумента.

Функция Cos(x: Real): Real — вычисляет значение косинуса аргумента.

Функция Sin(x: Real): Real — вычисляет значение синуса аргумента.

Функция Chr(x: Byte): Char — возвращает символ, соответствующий коду x; например:

```
Begin
```

```
  WriteLn('Код 75 соответствует знаку ',Chr(75));
```

```
End.
```

Функция Ord(x): Longint — возвращает порядковый номер для перечисляемого типа; например:

```
Var
```

```
  Colors = (Red,Blue,Green);
```

```
Begin
```

```
  WriteLn('Blue имеет порядковый номер ',Ord(Blue));
```

```
  WriteLn('Код ASCII для 'c' ',Ord('c'), '(десятичный)');
```

```
End.
```

Пример 58. Показ значения кода вводимого символа.

```
Program 03_58;
```

```
Var
```

```
  L: Char;
```

```
Begin
```

```
  WriteLn;
```

```
  Write('Введите символ: '); ReadLn(L);
```

```
  WriteLn('Код 'L,' = ',Ord(L));
```

```
End.
```

Функция Exp(x: Real): Real — вычисляет экспоненту аргумента. Тип аргумента и результата — действительное число. Например:

```
y := Exp(1.2); .
```

Функция Ln(x: Real): Real — вычисляет натуральный логарифм аргумента, например:

```
z := Ln(5.6); .
```

Функция Sqr(x): <тип параметра> — вычисляет значение квадрата аргумента. Например:
WriteLn('5 в квадрате будет: ', Sqr(5)); .

Функция Sqrt(x): Real — вычисляет значение квадратного корня аргумента. Например:
WriteLn('КjBaflpaTHbin корень из 8 равен: ', Sqrt(8.0)); .

Функция Frac(x: Real): Real — выделяет дробную часть аргумента; например: Var r: Real; Begin
r := Frac(123.456); {Результат = 0.456}
r := Frac(-123.456); {Результат = -0.456} End.

Результат представляет собой действительное число!

Функция Int(x: Real): Real — выделяет целую часть аргумента; например:
r := Int(123.456); {Результат = 123.0}

r := Int(-123.456); . {Результат = -123.0}

Результат, как и в предыдущем примере, — действительное число!

Функция Round(x: Real): Longint — округляет значение x до целого (Integer), например:

```
Begin
  WriteLn(1.4, ' округляется до ', Round(1.4)); { 1 }
  WriteLn(1.5, ' округляется до ', Round(1.5)); { 2 }
  WriteLn (-1.4, ' округляется до ', Round(-1.4)); { -1 }
  WriteLn (-1.5, ' округляется до ', Round(-1.5)); { -2 }
```

End.

Функция Trunc(x: Real): Longint — отбрасывает дробную часть аргумента; например:

```
Begin
  WriteLn(1.4, ' становится1, Trunc(1.4)); { 1 }
  WriteLn(1.5, ' становится', Trunc(1.5)); { 1 }
  WriteLn(-1.4, ' становится', Trunc(-1.4)); { -1 }
  WriteLn(-1.5, ' становится', Trunc(-1.5)); { -1 }
```

End.

Функция Length(s: String): Integer — возвращает длину строки; например:

```
Var s: String;
Begin
  ReadLn(s);
  WriteLn(7^HHa = ', length(s));
```

End.

Пример 59. Определение числа знаков во вводимой строке.

```
Program 03_59;
Var
  st: String;
Begin
  WriteLn;
  Write('Введите строку: ');
  ReadLn(st);

  WriteLn('Длина строки составляет: ', Length(st), ' знаков.');
```

End.

Функция Odd(x: Longint): Boolean — проверяет аргумент на нечетность; например:

```
If Odd(n) Then
  WriteLn('n — нечетное')
Else
  WriteLn('Что-то четное'); .
```

Пример 60. Ввод в строку нескольких пробелов и числа. Преобразование пробелов строки в ноли.

```

Program 03_60;
Var
  st: String;
Begin
  WriteLn;
  Write('Введите пробелы и число: ');
  ReadLn(st);           {Например, st = '123.5'}
  While Pos(' ', s) > 0 Do
    s[Pos(' ', s)] := '0';
  WriteLn(":25,s)
End.

```

Функция Random(x: Word)] — возвращает случайное число на интервале 0—1.

Пример 61. Вывод строк однотипных знаков, изменяя их цвет на каждой строке случайным образом.

```

Program 03_61;
Uses Crt;
Var
  st: String;
Begin
  st := '*****';
  Randomize;
  Repeat
    TextAttr := Random(256);
    WriteLn(s);
  Until KeyPressed;
  TextAttr := 14;
  WriteLn(s);
  ClrScr;
End.

```

Функция Succ(x): <тип параметра> — возвращает следующее значение параметра.

Идентичная запись: $\text{Ord}(\text{Succ}(x)) = \text{Ord}(x) + 1$.

Функция Pred(x): (Same type as parameter) — получает предшествующее значение параметра.

Идентичная запись: $\text{Ord}(\text{Pred}(x)) = \text{Ord}(x) - 1$.

Например:

```

Var
  Coiors = (Red,Blue,Green);
Begin
  WriteLn('Предшествующее значение 5 ',Pred(5));
  WriteLn ('Последующее значение 10 ',Succ(10));
  If Succ(Red) = Blue Then
    WriteLn('В типе Colors Red является ', 'предшественником Blue');
End.

```

Функция Hi(x: <тип параметра>): Byte — возвращает старший байт аргумента.

Функция Lo(x: <тип параметра>): Byte — возвращает младший байт аргумента.

Функция UpCase(Ch: Char): Char — преобразовывает строчную латинскую букву в прописную.

3.8. Разработка информационных систем для деловой практики

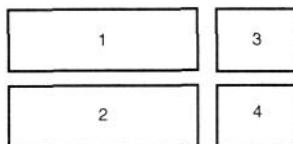
Компьютерная технология позволяет не только собрать и обработать информацию, получаемую в процессе научных исследований и экспериментальных наблюдений, но и, прежде всего, дать наглядное представление о сути явлений. Это можно осуществить с помощью оперативного выполнения различных расчетов, определения параметров математической модели наблюдаемого процесса или наглядного представления движения системы и т. п. — все эти способы компьютерного исследования составляют суть содержания информационной системы.

Этапы проектирования информационной системы:

- ◆ определение массивов исходных данных, представляемых на одном экране;
- ◆ определение формы выдачи результата: текст, статистика, графика;
- ◆ выделение областей экрана (окна) для:
 - ввода исходных данных,

- выдачи результата,
- управления информационной системой (меню);
- ◆ разработка программы ввода, записи в файл и выдачи исходной информации;
- ◆ разработка программы обработки информации по заданному алгоритму;
- ◆ разработка программы представления результатов в графической форме.

В качестве примера размещения информации на экране компьютера можно привести следующее расположение информационных окон (рис. 3.8).



1 — ввод данных; 2 — вывод результата; 3 — меню; 4 — справка

Рисунок 3.8. Размещение информационных окон на экране

При вводе данных о результатах испытаний воздействий на образец изделия можно рекомендовать следующее расположение окон, данных на окне ввода информации: $V[j]$ — тип воздействия (испытания) на изделие (образец), $N = 5$; $P[i]$ — образец, $M = 5$ (рис. 3.9).

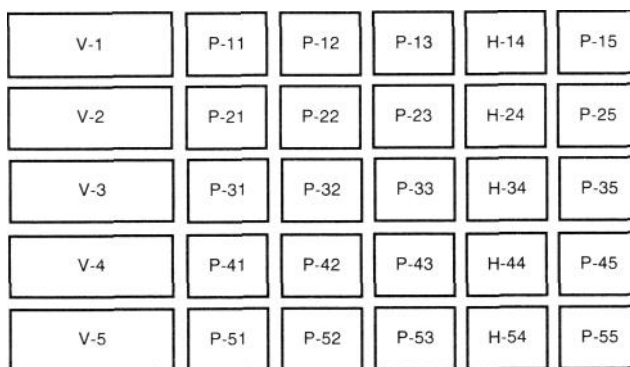


Рисунок 3.9. Размещение окон для записи результатов испытаний изделия

Для хранения информации необходимо предусмотреть запись данных в файл, имя которого должен задать сам пользователь. При выводе данных из файла система должна предусмотреть возможность определения имени файла пользователем.

Обработка данных может предусматривать определение среднего значения и среднеквадратичного отклонения по каждому $V-i$.

Меню информационной системы отражает однозначное соответствие выбранных для управления системой функциональных клавиш клавиатуры компьютера и выполняемых при их нажатии функций системы. В данном примере меню содержит следующие операции: ввод данных, запись в файл, чтение из файла, выдача результата статистического исследования, выход из системы.

Разработка программного обеспечения информационной системы состоит в написании процедуры меню, цикла обращения к этой процедуре и построении программного модуля, содержащего все процедуры, реализующие действие функциональных клавиш меню.

Программа меню может иметь следующий вид:

```
...
Uses Crt, Graph, 03_62_m;
Const
```

...

В этом блоке программы следует показать координаты используемых окон экрана:

```

...
Type
  Menu_key_set = Set Of #59 .. #67;
Var
  driver, mode, error: Integer;
  ch: Char;
  b: Boolean;
  Menu_key: Menu_key_set;
  d: darray; . {Этот массив определен в процедуре модуля системы}
  Теперь покажем содержание процедуры меню системы:
Procedure Menu;
Begin
  ch := ReadKey;
  If ch=#0 Then ch := ReadKey;
  If ch In Menu_key Then
    Frame (x31, y31, x32, y32, False); {Пассивная рамка}

  Case ch Of
    #59: Begin
      {New}
      Frame(x11, y11, x12, y12, True); {Окно 1 активно}
      Place(x11, y11, x12, y12); {Раскраска полей ввода}
      New(d); {Ввод данных}
      Frame(x11, y11, x12, y12, False); {Окно 1 пассивно}
    End;
    #60: Save(d); {Сохранить d на диске}
    #61: Begin
      {Load}
      Clear_d(d); {Чистка d}
      Load(d); {Загрузить данные с диска}
      WriteCol_d(d); {Отобразить d в окне}
    End;
    #62: Begin
      {Clean}
      Window1(x11,y11,x12,y12); {Нарисовать окно 1}
      Window2(x21,y21,x22,y22); {Нарисовать окно 2}
      Clear_d(d); {Чистка массива d}
      Place(x11, y11, x12, y12); {Раскраска полей ввода}
    End;
    #63: Medium(x21,y21,x22,y22,d); {Calculations}
    #64: MyGraph(x21,y21,x22,y22,d); {Graph}
    #67: b := False; {Выход <F9>}
  End;
End; .

```

Текст программы можно построить следующим образом:

```

...
b := True;
While b Do
  Begin
    Menu;
  End;
...

```

При разработке модуля используемых в меню подпрограмм необходимо указать типы используемых переменных и массивов данных, перечень процедур и функций. Приведем фрагменты программного модуля:

```

Unit 03_63_m;
Interface
Uses Crt, Graph;

```

```

Type
  dtype = Record
    s: String[10];
    a: Array[1..5] Of Real;
  End;
  darray = Array[1..5] Of dtype;
Procedure Window1(x1, y1, x2, y2: Integer);      {Окно 1}
...
Procedure Frame(x1, y1, x2, y2: Integer; active: Boolean);
Procedure Place(x1,y1,x2,y2: Integer); {Раскраска полей данных}
Procedure Clear_d(Var d: darray);           {Чистка d}
Procedure WriteCOL_d(Var d: darray); {Отображение массива d}
Procedure New(Var d: darray);               {Ввод данных}
Procedure Save(Var d: darray);   {Сохранение на диске}
Procedure Load(Var d: darray);  {Загрузить данные с диска}
Procedure Medium(x1,y1,x2,y2: Integer; Var d: darray);
Procedure MyGraph(x1,y1,x2,y2: Integer; Var d: darray); .

```

Раздел Implementation включает в себя описания разработанных подпрограмм. Покажем примеры некоторых из них:

```

Procedure Window1;
Begin
  SetFillStyle(1, White);
  Bar(x1, y1, x2, y2);
  SetColor(LightGray);
  SetLineStyle(0, 0, 3);
  Rectangle(x1-3, y1-3, x2+3, y2+3);
  MoveTo(x1+200, y1);
  SetColor(Cyan);
  OutText('Input');
End; .

```

Суть работы активной рамки заключается в том, что при работе в окне с активной рамкой ее цвет изменяется на контрастный. При переходе на другое рабочее окно цвет активной рамки восстанавливается, а активной становится рамка очередного рабочего окна. Механизм использования активной рамки показан на следующем примере:

```

Procedure Frame;
Begin
  If active Then                               {Активная рамка}
    Begin
      SetLineStyle(0, 0, 3);                   {Толстая линия рамки}
      SetColor(Green);                         {Зеленый цвет рамки}
      Rectangle(x1-3, y1-3, x2+3, y2+3);      {Активная рамка}
      Exit;
    End;
  SetLineStyle(0, 0, 3); {Пассивная рамка}
                                {Толстая линия рамки}
  SetColor(LightGray);           {Обычный цвет рамки}
  Rectangle(x1-3, y1-3, x2+3, y2+3); {Пассивная рамка}
  Exit;
End; .

```

Разметку полей для ввода данных в окне ввода целесообразно выполнить в виде отдельной процедуры:

```

Procedure Place;
Var i,j: Integer;
    x,y: Integer;
Begin
    y := y1+20;
    SetFillStyle(1, Cyan);
    For i := 1 To 5 Do
        Begin Bar(x1+16, y, x1+96, y+14);
            y := y+28;
        End;
    x := 120;
    For i := 1 To 5 Do
        Begin
            y := y1+20;
            For j := 1 To 5 Do
                Begin Bar(x1+x, y, x1+x+40, y+14);
                    y := y+28;
                End;
            x := x+64;
        End;
    End; .

```

Для удаления записей с полей окна ввода данных применим процедуру Clear_d:

```

Procedure Clear_d;
Var i,j: Integer;
Begin
    For i := 1 To 5 Do With d[i] Do
        Begin
            s := "";
            For j := 1 To 5 Do a[j] := 0.0;
        End;
    End; .

```

Для отображения на окне ввода ранее запомненных данных используем разработанную процедуру

WriteCol_d:

```

Procedure WriteCol_d;
Var i,j,x,y: Integer;
Begin
    TextColor(Yellow);
    x := 5; y := 3;
    For i := 1 To 5 Do
        Begin
            TextColor(8+i);
            GotoXY(x,y); Write(d[i].s);
            y := y + 2;
        End;
    x := 18;
    y := 3;
    For i := 1 To 5 Do With d[i] Do {Цикл записей dtype}
        Begin
            TextColor(8+i);
            For j := 1 To 5 Do {Цикл чисел в записи}
                Begin
                    GotoXY(x,y); Write(a[j]:5:2); {Вывод числа на экран}
                    x := x + 8; {Следующее поле на экране}
                End;
            y := y + 2; {Следующая строка}
            x := 18; {Начало чисел в строке}
        End;
    End; .

```

Ввод новых данных — результатов испытаний целесообразно организовать в виде отдельной процедуры, как показано на следующем примере:

```

Procedure New;
Var i,j,x,y,z: Integer;
    k: Boolean;

```



```

Begin
  x := 5; y := 3;
  For i := 1 To 5 Do With d[i] Do
    Begin
      GotoXY(x,y);
      ReadLn(s);
      z := 18;
      k := False;
      For j := 1 To 5 Do
        Begin
          GotoXY(z,y);
          ReadLn(a[j]);
          If a[j] = -1.0 Then
            {Условие прекращения ввода данных}
            Begin
              Exit;
              k := True;
            End;
          z := z+8;
          If k Then Exit;
        End;
      If k Then Exit;
      y := y+2;
    End;
  End; .

```

Процедура записи данных на диск (Save) может иметь следующий вид:

```

Procedure Save;
...
Begin
  Assign(f, 'exp.dat');
  Rewrite(f);
  For i := 1 To 5 Do Write(f,d[i]);
  Close(f);
End; .

```

Загрузку данных с диска можно осуществить, используя процедуру Load, фрагмент которой приведен в следующем примере:

```

Procedure Load;
...
Begin
  Assign(f, 'exp.dat');
  Reset(f);
  For i := 1 To 5 Do Read(f,d[i]);
  Close(f);
End; .

```

Вычисление среднего и среднеквадратичного отклонения также следует оформить в виде отдельной процедуры:

Procedure Medium;

```
...
s1 := 0.0; {Среднее значение}
n := 0;
For i := 1 To 5 Do
  For j := 1 To 5 Do With d[1] Do
    Begin
      s1 := s1 + a[i];
      If a[i] <> 0.0 Then n := n+1;
    End;
s1 := s1 / n;
OutTextXY(40,210,'Среднее значение');
GotoXY(25,16);
Write(s1:8:2);
dis1 := 0.0; {Среднеквадратичное отклонение}
For j := 1 To 5 Do With d[1] Do
  dis1 := dis1 + SQR(a[j] - s1);
dis1 := Sqrt(dis1) / n;
OutTextXY(40,240,'Среднеквадратичное отклонение');
GotoXY(40,18);
Write(dis1:8:2); . . .
```

Для отображения данных в виде графиков используем процедуру MyGraph, которая может содержать следующую последовательность операторов:

```
Procedure MyGraph;
Const
  ndx = 4; ndy = 4; n = 5; m = 2;
Var
  ymax, ymin, gx, gy, xmin, xmax, xl, xp, dx, dy: Real;
  i, j, poi, xn, xk, yn, yk, lx, ly, mx, my: Integer;
  ch: Char;
```

```

st: String;
Begin
  WriteCol_d(d);
  SetColor(DarkGray);
  xn := x1 + 60; xk := x2 - 48;    {Координаты поля графика}
  yn := y1 + 15; yk := y2 - 13;
  SetLineStyle(SolidLn,0, NormWidth); {Тонкая линия для}
  Rectangle(xn,yn,xk,yk);          {очерчивания поля графика}
  Line(xn,yn,xn-5,yn+5);
  Line(xn,yn,xn+5,yn+5);          {Стрелки на концах осей}
  Line(xk,yk,xk-5,yk-5);
  Line(xk,yk,xk-5,yk+5);
  ymax := -1e10; ymin := 1e10;
  For i := 1 To 5 Do {Нахождение максимума и минимума}
    {всех значений}
    With d[i] Do
      For j := 1 To 5 Do
        Begin
          If a[j]<ymin Then ymin := a[j];
          If a[j]>ymax Then ymax := a[j];
        End;
      End;
  For i := 1 To 5 Do
    Begin {Рисование графиков тройными}
      xmin := 1; {разноцветными линиями}
      xmax := 5;
      SetColor(8+i);
      With d[i] Do
        Begin
          rx := xmax-xmin;
          ry := ymax-ymin;
          mx := xk-xn;
          my := yk-yn;
          xl := xmin-1;
          SetLineStyle(SolidLn,0,ThickWidth);
          For poi := 1 To 5-1 Do
            Begin
              xl := xl+1;
              xp := xl+1;
            End;
          End;
        End;
    End;
  End;

```

```

        Line(Round((xl-xmin)*mx/rx)+xn,
        Round((ymax-a[poi])*my/ry)+yn,
        Round((xp-xmin)*mx/rx)+xn,
        Round((ymax-a[poi+1])*my/ry)+yn);
    End;
End;
End;
SetColor(DarkGray);
SetLineStyle(SolidLn,0, NormWidth);
Lx := (xk-xn) Div ndx;
dx := (xmax-xmin)/ndx; {Нанесение вертикальной сетки}
For i := 1 To ndx+1 Do {и вывод горизонтальных}
    Begin {надписей}
        Line(xn+lx*(i-1), yn, xn+lx*(i-1), yk);
        Str((xmin+(i-1)*dx):n:m, st);
        OutTextXY(xn+lx*(i-1)-(n-m)*8+4, yk+4, st);
    End;
ly := (yk-yn) Div ndy;
dy := (ymax-ymin)/ndy; {Нанесение горизонтальной}
For i := 1 To ndy + 1 Do {сетки и вывод вертикальных надписей}
    Begin
        Line(xn, yn+(i-1)*ly, xk, yn+(i-1)*ly);
        Str((ymax-(i-1)*dy):n:m, st);
        OutTextXY(xn-(n*8+8), yn+(i-1)*ly-4, st);
    End;
Frame(x1, y1, x2, y2, True); {Активная рамка}
ch := ReadKey; {Приостановить работу}
Frame(x1, y1, x2, y2, False); {Пассивная рамка}
End; .

```

Практикум: Программирование в Турбо Паскале

1. Управление цветом. Ввод и вывод данных

Пример 1. Создание белого окна (X1 = 3, Y1 = 2, X2 = 60, Y2 = 12) на синем экране.

```

Program П_1;
Uses Crt;
Var
    ch: Char;
Begin
    TextBackground(Blue); TextColor(Yellow);
    ClrScr;
    Window(3,2,60,12);
    TextBackground(White);      {Цвет окна — белый}
    TextColor(Black);           {Цвет чернил — черный}
    ClrScr;
    WriteLn('Ввод данных');     {Заголовок окна}
    ch := ReadKey;
    TextBackground(Black);
    TextColor(White);           {Обычная раскраска}
    ClrScr;
    WriteLn('Восстановление цвета экрана');
    ch := ReadKey;
    TextMode(Co80);
End.

```

Пример 2. Изображение трехполосного экрана: белого, синего, красного.

```

Program П_2;
Uses Crt;
Var
  ch: Char;
Begin
  TextBackground(Black); ClrScr;
  Window(3,2,77,8);
  TextBackground(White); ClrScr;
  Window(3,9,77,15);
  TextBackground(LightBlue); ClrScr;
  Window(3,16,77,22);
  TextBackground(LightRed); ClrScr;
  ch := ReadKey;
End.

```

Пример 3. Создание окна заданного цвета с цветом знаков определенного цвета.

```

Program П_3;
Uses Crt;
Var
  k,m: Integer;
  ch: Char;
Begin
  TextBackground(Blue); TextColor(Yellow);
  ClrScr; Window(3,2,60,15);
  Write('Напишите цвет окна'); ReadLn(k);
  Write('Напишите цвет знаков'); ReadLn(m);
  TextBackground(k); TextColor(m);
  ClrScr;
  WriteLn('Цвет окна — ',k,' цвет знаков — ',m);
  ch := ReadKey; TextMode(Co80);
End.

```

Пример 4. Создание белого, желтого и голубого окна на синем экране. Создание надписей на белом окне — «Ввод данных», на желтом — «Вывод результатов», на голубом — «Выполняемая операция».

```

Program П_4;
Uses Crt;
Var
  ch: Char;
Begin
  TextBackground(Blue); TextColor(White);
  ClrScr; GotoXY(32,1);
  WriteLn('Программа П_4');
  Window(3,2,58,12); {Белое окно}
  TextBackground(White); TextColor(Black);
  ClrScr; GotoXY(23,1); Write('Ввод данных');
  Window(3,15,58,23); {Желтое окно}
  TextBackground(Yellow); TextColor(Blue);
  ClrScr; GotoXY(18,1);
  Write('Вывод результатов');
  Window(62,2,76,12); {Голубое окно}
  TextBackground(Сюан); TextColor(Black);
  ClrScr; GotoXY(5,1);
  Write('Выполняемая операция');
  ch := ReadKey; TextMode(Co80);
End.

```

Пример 5. Изображение тени от окон, подготовленных в примере 4.

```

Program П_5;
Uses Crt;
Var
  x1,y1,x2,y2: Integer;
  ch: Char;
Begin
  TextBackground(Blue); TextColor(White);
  ClrScr;
  GotoXY(32,1);
  WriteLn('Программа П_5');
  x1 := 3; y1 := 2; x2 := 58; y2 := 12;
  Window(x2+1,y1+1,x2+2,y2+1); {Вертикальная тень от окна}
  TextBackground(Black); ClrScr;
  Window(x1+2,y2+1,x2+2,y2+1); {Горизонтальная тень от окна}
  TextBackground(Black); ClrScr;
  Window(x1,y1,x2,y2); {Белое окно}
  TextBackGround(White); TextColor(Black);
  ClrScr; GotoXY(23,1); Write('Ввод данных');
  y1:=15; y2 := 23;
  Window(x1+1,y1+1,x2+2,y2+1); {Второй вариант}
  TextBackground(Black); ClrScr;
  Window(x1,y1,x2,y2); {Желтое окно}
  TextBackground(Yellow); TextColor(Blue);
  ClrScr; GotoXY(18,1);
  Write('Результат вычислений');
  x1 := 62; y1 := 2; x2 := 76; y2 := 12;
  Window(x2+1,y1+1,x2+2,y2+1);
  TextBackground(Black); ClrScr;
  Window(x1+2,y2+1,x2+2,y2+1);
  TextBackground(Black); ClrScr;
  Window(x1,y1,x2,y2); {Голубое окно}
  TextBackground(Cyan); TextColor(Black);
  ClrScr; GotoXY(5,1); Write('Операция');
  ch := ReadKey;
  TextMode(Co80);
End.

```

Пример 6. Ввод трех чисел построчно в окно ввода и вывод их на одной строке в окно вывода. Написание заголовка задачи в третьем окне.

```

Program П_6;
Uses Crt;
Var

```

```

a,b,c: Integer;
x1,y1,x2,y2: Integer;
ch: Char;
Begin
  TextBackground(Blue); TextColor(White);
  ClrScr;
  GotoXY(32,1);
  WriteLn('Программа П_6');
  x1 := 3; y1 := 2; x2 := 58; y2 := 12;
  Window(x2+1,y1+1,x2+2,y2+1);      {Вертикальная тень от окна}
  TextBackground(Black); ClrScr;
  Window(x1+2,y2+1,x2+2,y2+1);      {Горизонтальная тень от окна}
  TextBackground(Black); ClrScr;
  Window(x1,y1,x2,y2);              {Белое окно}
  TextBackpround(White); TextColor(Black);
  ClrScr; GotoXY(23,1); Write('Ввод данных');
  y1:=15; y2 := 23;
  Window(x1+1,y1+1,x2+2,y2+1);      {Второй вариант}
  TextBackground(Black); ClrScr;
  Window(x1,y1,x2,y2);              {Желтое окно}
  TextBackground(Yellow); TextColor(Blue);
  ClrScr; GotoXY(18,1);
  Write('Результат вычислений');
  x1 := 62; y1 :=2; x2 := 76; y2 := 12;
  Window(x2+1,y1+1,x2+2,y2+1);
  TextBackground(Black); ClrScr;
  Window(x1+2,y2+1,x2+2,y2+1);
  TextBackground(Black); ClrScr;
  Window(x1,y1,x2,y2);              {Голубое окно}
  TextBackground(Сyan); TextColor(Black);
  ClrScr; GotoXY(5,1); Write('Операция');
  GotoXY(3,4); Write('Ввод-вывод');
  GotoXY(3,5); Write('Данных');
  Window(3,2,58,12);
  GotoXY(2,2); Write('A = '); Read(a);
  GotoXY(2,3); Write('B = '); Read(b);
  GotoXY(2,4); Write('C = '); Read(c);
  Window(2,15,58,23);
  GotoXY(4,4); Write('A = ',a,' B = ',b,' C = ',c);
  ch := ReadKey;
  TextMode(Co80);
End.

```

Пример 7. Ввод символа с клавиатуры и выдача на экран значения его кода.

```

Program П_7;
Uses Crt;
Var
  L: Char;
Begin
  ClrScr;
  WriteLn;
  Write('Введите символ: '); ReadLn(L);
  WriteLn('Код L',Ord(L));
End.

```

Пример 8. Определение кода введенного символа. Определение площади круга, длины окружности, объема шара и площади вписанного в окружность n-угольника с использованием значения этого кода в качестве радиуса.

```

Program П_8;
Uses Crt;
Var
  s,v,l,v: Real;
  n: Integer;
  r: Byte;
  a, ch: Char;
Begin
  ClrScr; WriteLn; Write('Введите символ: '); ReadLn(a);
  Write('Число сторон n-угольника: '); ReadLn(n);
  r := Ord(a); s := Pi*r*r; l := 2*Pi*r;
  v := 4/3*Pi*exp(3*Ln(r));
  su := n*(r*Cos(Pi/n))*(r*Sin(Pi/n));
  WriteLn('Радиус круга = ',r:3);
  WriteLn('Площадь круга = ',s:7:2);
  WriteLn('Длина окружности = ',l:7:2);
  WriteLn('Объем шара = ',v:9:2);
  WiteLn('Площадь ',n,'-угольника ',su:7:2);
  ch := ReadKey;
End

```

Пример 9. Определение площади круга, длины окружности и объема шара с использованием в качестве радиуса суммы знаков трехзначного числа. Изображение исходных данных и результата с использованием окон.

```

Program П_9;
Uses Crt;
Var
  ch: Char;
  a,r,s1,v: Real;
  a1,a2,a3: Integer;
Begin
  TextBackground(1); ClrScr;
  TextColor(0);
  Window(4,4,44,11); TextBackground(0); ClrScr;
  Window(4,14,44,23); TextBackground(0); ClrScr;
  Window(50,4,78,11); TextBackground(0); ClrScr;
  Window(2,2,43,10); TextBackground(15); ClrScr;
  GotoXY(14,1); WriteLn('Ввод данных');
  Write('Введите 3 значное число: ');
  ReadLn(A);
  a1 := Trunc(A/100);
  a2 := Trunc((Frac(A/100))*10);
  a3 := Trunc(Frac(A/10)*10);
  R := a1+a2+a3; S:=PI*Sqr(R); L:=2*PI*R;
  V := (4/3)*PI*exp(3*Ln(R));
  Window(2,13,42,22); TextBackground(Yellow);
  ClrScr; GotoXY(15,1); WriteLn('Результат');
  WriteLn('R =',R:10:2);
  WriteLn('S = ',S:10:2);
  WriteLn('l = ',l:10:2);
  WriteLn('V = ',V:10:2);
  Window(46,2,76,10);
  TextBackground(3); ClrScr;
  GotoXY(10,2); WriteLn('Пример П_9');
  ch := ReadKey;
End.

```

Пример 10. Вычисление значений функции:

$$a = \sqrt[4]{\ln(x4) + 8^x} + (1 + x)^2$$

$$b = x \times y \times z - 3,3 \times (x + \sqrt[2]{z}) / (10^2 + \sqrt{\ln 4})$$

$$c = (n + \sin(e^4 \times \ln \pi)) \times (\sin(e^4 \times \ln \pi) / \cos(2) + |\operatorname{ctg}(x)|).$$

```

Program П_10;
Uses Crt;
Var
  a,b,c: Real;
  x,y,z,n: Integer;
  ch: Char;
Begin
  WriteLn('Введите исходные данные: ');
  Write('X = '); ReadLn(x);
  Write('Y = '); ReadLn(y);
  Write('Z = '); ReadLn(z);
  Write('N = '); ReadLn(n);
  a := exp(1/4*ln(exp(4*ln(x))+exp(x*ln(8))))+sqrt(1+x*x);
  b := (x*y*z-3.3*abs(x+exp(1/4*ln(y))))/(exp(4*ln(10))+sqrt(ln(4)));
  c := (n+sin(exp(4*ln(pi))*sin(exp(4*ln(pi))))/(cos(2)+abs(cos(x)/sin(x)));
  WriteLn('a= ',a:5:2, ' b= ',b:5:2, ' c= ',c:5:2);
  ch := ReadKey;
End.

```

Пример 11. Определение третьей цифры числа.

```

Program П_11;
Var
  c: Real;
  k: Integer;
Begin
  Write('Введите число: '); ReadLn(c);
  k := Trunc(Frac(c/1000)*10);
  WriteLn('3-я цифра числа = ',k);
  ReadLn;
End.

```

Пример 12. Ввод четырехзначного числа. Определение суммы и произведения его знаков. Вычисление квадратных и кубических корней суммы и произведения.

```

Program П_12;
Uses Crt;
Var
  a1,a2,a3,a4,a,i: Integer;
  s,p,s2,s3,p2,p3: Real;
Begin
  ClrScr;
  Write('Введите 4-значное число: ');
  ReadLn(a);
  a1 := Trunc(Frac(a/10)*10);
  a2 := Trunc(Frac(a/100)*10);
  a3 := Trunc(Frac(a/1000)*10);
  a4 := Trunc(a/1000);
  s := a1 + a2 + a3 + a4;
  p := a1 * a2 * a3 * a4 ;

  s2 := Sqrt(s);
  s3 := Exp(1/3(Ln(s)));
  p2 := Sqrt(p);
  p3 := Exp(1/3(Ln(p)));
  WriteLn(a1:3,a2:3,a3:3,a4:3);
End.

```

2. Работа с операторами

Пример 13. Определение равенства суммы двух первых и двух последних цифр целого числа, если они не равны — выдача соответствующего текста.

Вариант 1:

```

Program П_13;
Uses Crt;
Var
  a1,a2,a3,a4: Real;
  a,i: Integer;
Begin
  ClrScr;
  WriteLn('Введите значение a: '); ReadLn(a);
  a1 := Trunc((Frac(a/10)*10);
  a2 := Trunc((Frac(a/100)*10);
  a3 := Trunc((Frac(a/1000))*10);
  a4 := Trunc(a/1000);
  If (a1+a2=a3+a4) Then
    WriteLn('Суммы равны')
  Else
    WriteLn('Суммы не равны');
End.

```

Вариант 2:

```

Program П_13;
Uses Crt;
Var
  a,b,b1,c,c1,d,p,x,s12,s34: Integer;
  m: Char;
Begin
  Repeat
    ClrScr;
    Write('x = '); ReadLn(x);
    a := x Div 1000;
    b1 := x Div 100;
    b := b1 Mod 10;

    c1 := (x Mod 100)*100;
    c := c1 Div 1000;
    d := (x Mod 10);
    WriteLn('a = ',a);
    WriteLn('b = ',b);
    WriteLn('c = ',c); WriteLn('d = ',d);
    p := a*b*c*d; s12 := a+b; s34 := c+d;
    WriteLn('Произведение чисел равно: ',p:5);
    If s12 = s34 Then
      WriteLn('Сумма двух первых чисел равна сумме двух последних');
      WriteLn ('S = ',s12:5)
    Else
      Begin
        WriteLn('Сумма двух первых чисел не равна сумме двух последних');
        WriteLn('чисел');
        WriteLn('s12 = ',s12,' s34 = ',s34);
      End;
    Write('Ввести другое число? (y=да, n=нет) ');
    ReadLn(m);
  Until m = 'n';
End.

```

Пример 14. Вычисление значения функции: $y=\cos^2(x)$, при $x\leq 0$ $y=1/x^2$, при $0<x<2$ $y=x^2-\lg(x)$, при $x\geq 2$.

```

Program П_14;
Uses Crt;
Var
  x,y: Real;
  i: Integer;
  ch: Char;
Begin
  ClrScr; WriteLn;
  Write('Введите число x: '); ReadLn(x);
  If (x<=0) Then
    y := cos(x) * cos(x)
  Else
    If (x<2) Then
      y := 1/(x*x)
    Else y := (exp(ln(2) * x)-ln(x)/ln(10));
  WriteLn('y = ',y);
End.

```

Пример 15. Даны два целых числа. Замена первого нулем, если оно меньше или равно второму.

```

Program П_15;
Uses Crt;
Label q1;
Var
  a,b: Integer;
Begin
  q1: ClrScr;
  Write('Введите число a: '); ReadLn(a);
  Write('Введите число b: '); ReadLn(b);
  If a <= b Then a := 0;
  WriteLn('a = ',a, ' b = ',b); Goto q1;
End.

```

Пример 16. Ввод числа и определение позиций, в которых записан ноль. Если число отрицательное — выдача соответствующего текста.

```

Program П_16;
Uses Crt;
Label q1;
Var
  i,k: Integer;
  a: Real;
  b: Boolean;
Begin
  q1: ClrScr;
  TextBackground(Blue); TextColor(Red);
  b := False;
  WriteLn('Введите число: '); ReadLn(a);
  If k < 0 Then Begin
    WriteLn('Число отрицательное, повторите ввод');
    Goto q1;
  End
  Else Begin
    For i := 1 To 5 Do Begin
      k := Trunk(Frac(a / (10 * i) * 10));
      If k = 0 Then Begin
        b := True;
        WriteLn('В позиции', ' ',i, ' ', 'записан ноль');
      End;
    End;
  End;
  If Not b Then WriteLn('Полей нет');
  ch := ReadKey;
  Goto m;
End.

```

Пример 17. Организация трех окон: для ввода, вывода и меню программы. Использование клавиши <F1>, <F2>, <F9> для управления программой: <F1> — ввод двух чисел, повторение ввода; <F2> — выполнение функции $y = a + b$ и выдача результата; <F9> — выход из программы.

```

Program П_17;
Uses Crt;
Const
  x11= 3; y11= 2; x12=58; y12=12;
  x21= 3; y21=15; x22=58; y22=23;
  x31=62; y31= 2; x32=76; y32=12;
Var
  a,b,c: Integer;
  ch: Char;
  d: Boolean;
Begin
  TextBackground(Blue); TextColor(White); {Синий фон}
  ClrScr; {Раскрасить экран}
  Window(x31,y31,x32,y32); {Окно меню}
  TextBackground(Yellow); TextColor(Black);
  ClrScr; {Раскрасить окно}
  WriteLn('Меню');
  Write('Ввод <F1>');
  Write('Вычисление <F2>');
  Write('Выход <F9>');
  d := True;
  While d Do {Цикл меню}
    Begin
      ch := ReadKey; {Нажать клавишу}
      Case ch Of
        #59: Begin
          Window(x11,y11,x12,y12); {Окно 1}
          TextBackground(Yellow);
          TextColor(Blue);
          ClrScr; {Раскрасить окно}
          Write('Введите a = '); ReadLn(a);
          Write('Введите b = '); ReadLn(b);
        End;
        #60: Begin
          Window(x21,y21,x22,y22); {Окно 2}
          TextBackground(Yellow);
          TextColor(Blue); ClrScr;
          c := a + b;
          Goto(2,2);
          WriteLn('Сумма c = ',c)
        End;
        #67: d := False; {Выход <F9>}
      End;
    End; TextMode(Co80);
  End.

```

3. Работа с массивами

Пример 18. Нахождение в массиве целых чисел минимального элемента.

```

Program П_18;
Uses Crt;
Var
  a: Array[1..50] Of Integer;
  min,n,i: Integer;
Begin
  ClrScr;
  Write('Сколько чисел в массиве?'); ReadLn(n);
  For i := 1 To n Do
    Begin
      Write('Элемент ',i,' : '); ReadLn(a[i]);
    End;
  min := a[1];
  For i := 2 To n Do
    If a[i] < min Then min := a[i];
  WriteLn('Минимальный элемент в массиве = ', min);
  Repeat Until KeyPressed;
End.

```

Пример 19. Определение числа положительных и отрицательных элементов в массиве из нескольких (< 10) чисел. Переписывание положительных чисел в другой массив и его распечатка.

```

Program П_19;
Uses Crt;
Var
  i,k,m: Integer;
  x,y: Array[1..10] Of Real;
Begin
  ClrScr;
  Write('Введите размер массива: '); ReadLn(m);
  For i := 1 To m Do
    Begin
      Write('x[' ,i,']='); Read(x[i]);
    End;
  k := 0;
  m := 0;
  For i := 1 To 10 Do If x[i] > 0 Then Begin
    Inc(k); y[k] := x[i];
  End
  Else
    If x[i] < 0 Then Inc(m);
  WriteLn('В массиве ',k,' положительных и ',m,' отрицательных элементов');
  If k > 0 Then
    For i := 1 To k Do WriteLn('y[' ,i,'] = ',y[i]:7:2)
  Else
    WriteLn('Положительных элементов в массиве нет');
  Repeat Until KeyPressed;
End.

```

Пример 20. Перемена местами элементов массива с номерами m и n.

```

Program П_20;
Uses Crt;
Var
  a: Array[1..20] Of Integer;
  i,m,n,k,l: Integer;
Begin
  ClrScr;
  Write('Введите размер массива: '); ReadLn(l);
  WriteLn('Введите массив: ');
  For i := 1 To l Do
    Begin
      Write('a[' ,i,'] '); ReadLn(a[i]);
    End;
  Write('Введите номера элементов, которые надо поменять местами m,n: ');
  ReadLn(m,n);
  k := a[m]; a[m] := a[n]; a[n] := k;
  WriteLn('Преобразованный массив');
  For i := 1 To l Do Write(' ',a[i]);
End.

```

Пример 21. Перемена местами min и max элементов массива.

```
Program П_21;
Uses Crt;
Var
  a: Array[1..20] Of Integer;
  i,max,min,n,k,m,c: Integer;
Begin
  ClrScr;
  Write('Сколько элементов в массиве?');
  ReadLn(k);
  WriteLn('Введите массив: ');
  For i := 1 To k Do
    Begin
      Write('a[' ,i,'] = '); Read(a[i]);
    End;
  max := a[1]; m := 1; {Определение max элемента}
  For i := 1 To k Do
    Begin
      If a[i] > max Then Begin
        max := a[i]; m := i;
      End;
    End;
  min := a[1]; c := 1; {Определение min элемента}
  For i := 1 To k Do
    Begin
      If a[i] < min Then Begin
        min := a[i]; c := i;
      End;
    End;
  n := a[m]; a[m] := a[c]; a[c] := n;
  WriteLn('max=a[' ,m,']=',max,' min=a[' ,c,']=', min);
  Write('Новый массив: ');
  For i := 1 To k Do Write(a[i], ' ');
  Repeat Until KeyPressed;
End.
```

Самостоятельно: упростите текст предыдущей программы, объединив оба цикла вычислений.

Пример 22. Переписывание элементов, делящихся без остатка на 4, в другой массив и подсчет их суммы.

```
Program П_22;
Uses Crt;
Var
  a,m: Array[1..20] Of Integer;
  i,n,s,k: Integer;
Begin
  ClrScr;
  Write('Введите размер массива: '); ReadLn(k);
  n := 0; s := 0; WriteLn('Введите массив: ');
  For i := 1 To k Do Begin
    Write('m[' ,i,'] = '); ReadLn(m[i]);
    If m[i] Mod 4 = 0 Then
      Begin
        Inc(n); Inc(s,m[i]); a[n] := m[i];
      End;
  End;
  If s = 0 Then
    WriteLn('Элементов, делящихся без остатка на 4, нет')
  Else
    Begin
      WriteLn('Новый массив: ');
      For i := 1 To n Do
        Write('a[' ,i,'] = ',a[i]);
      WriteLn;
      WriteLn('Сумма элементов нового массива = ', s);
    End;
  Repeat Until KeyPressed;
End.
```

Пример 23. Формирование и распечатка массива: $c[i] = b[i] \wedge a > \max$.

```

Program П_23;
Uses Crt;
Var
  b: Array[1..20] Of Integer;
  c: Array[1..20] Of Real;
  max,i,n : Integer;
Begin
  ClrScr;
  Write('Сколько элементов в массиве?'); ReadLn(n);
  For i := 1 To n Do
    Begin
      Write('b[' ,i, ' ] = '); ReadLn(b[i]);
    End;
  max := b[1];
  For i := 1 To n Do
    If b[i] > max Then max := b[i];
  ClrScr;
  WriteLn('Исходный массив');
  For i := 1 To n Do Write(' :5,b[i]);
  WriteLn;
  WriteLn('Максимальный элемент массива = ',max);
  WriteLn('Преобразованный массив');
  For i := 1 To n Do
    Begin
      c[i] := b[i] / max; Write(' :3,c[i]:5:3);
    End;
  Repeat Until KeyPressed;
End.

```

Пример 24. Перемена местами двух элементов матрицы.

```

Program П_24;
Uses Crt;
Var
  a: Array[1..5,1..5] Of Integer;
  f,l,k,m,n,i,j: Integer;
Begin
  ClrScr;
  WriteLn('Число строк в матрице: '); ReadLn(k);
  WriteLn('Число столбцов в матрице: '); ReadLn(l);
  WriteLn('Введите матрицу: ');
  For i := 1 To k Do
    Begin
      For j := 1 To l Do
        Read(a[i,j]);
      ReadLn;
    End;
  WriteLn('Введите индексы одного элемента: ');
  ReadLn(i,j);
  WriteLn('Введите индексы другого элемента: ');
  ReadLn(m,n);
  f := a[i,j]; a[i,j] := a[m,n]; a[m,n] := f;
  WriteLn('Преобразованная матрица');
  For i := 1 To k Do
    Begin
      For j := 1 To l Do Write(' ',a[i,j]);
      WriteLn;
    End;
  Repeat Until KeyPressed;
End.

```

Пример 25. Определение минимального элемента массива с нечетным индексом.

```

Program П_25;
Uses Crt;
Const n=10;
Var
  a: Array[1..n] Of Real;
  min: Real;
  i,m: Integer;
Begin
  ClrScr;
  Write('Введите размер массива: '); ReadLn(m);
  WriteLn('Введите массив: ');
  For i := 1 To m Do
    Begin
      Write('a[' ,i, ' ] = '); ReadLn(a[i]);
    End;
  min := a[1];
  For i := 3 To m Do
    If (i Mod 2 = 1) And (a[i] < min) Then min := a[i];
  WriteLn('Мин. элемент с нечетным индексом = ', min:5:2);
  Repeat Until KeyPressed;
End.

```

Пример 26. Определение максимального по модулю элемента массива задаваемого размера среди элементов, имеющих четный индекс.

```

Program П_26;
Uses Crt;
Label n1;
Var
  a: Array[1..100] Of Real;
  k,i: Integer;
  max,p: Real;
  d,c: Char;
Begin
  ClrScr;
  TextBackground(8); TextColor(15); ClrScr;
  GotoXY(6,3);
  WriteLn('Определить максимальный по модулю элемент массива A');
  WriteLn('задаваемого размера, среди элементов, имеющих четный индекс');
  GotoXY(3,24); Write('Press any key');
  c:= ReadKey; ClrScr; TextBackground(7); ClrScr;
  Window(47,4,78,20); TextBackground(0); ClrScr;
  Window(46,3,77,19); TextBackground(2); ClrScr;
  Write('Результаты'); WriteLn;
  Window(4,4,42,20); TextBackground(0); ClrScr;
  Window(3,3,41,19); TextBackground(1); ClrScr;
  GotoXY(15,1); Write('Ввод данных');
  GotoXY(2,2);
  WriteLn('Введите число элементов массива: ');
  GotoXY(3,3); ReadLn(k);
  For i := 1 To k Do
    Begin
      Write(' ,i, ' элемент равен: '); ReadLn(a[i]);
    End;
  max := a[1];
  For i := 1 To k Do
    Begin
      p := Frac(i/2);
      If p = 0 Then
        If Abs(a[i]) > max Then max := Abs(a[i]);
    End;
  Window(46,3,77,19); TextBackground(2); ClrScr;
  Write('Результаты');
  GotoXY(2,2); WriteLn('Полученный массив: ');
  For i := 1 To k Do WriteLn('a[' ,i, ']= ',a[i]:3:1, ' ');
  WriteLn;
  WriteLn('Максимальный элемент: ',max:0:1);
  d := ReadKey; ClrScr;
End.

```


Пример 27. Ввод и распечатка в виде таблицы массива из целых чисел. Вывод данных в рамке.

```

Program П_27;
Uses Crt;
Label q1, q2, q3;
Var
  i,j,n,k: Integer;
  a: Array[1..80] Of Integer;
Begin
  Repeat
    Window(1,1,80,25);
    TextColor(7); TextBackground(1); ClrScr;
    Window(15,8,67,16);
    TextBackground(2); ClrScr; TextColor(7);
    GotoXY(13,4); Write('Размерность массива: ');
    ReadLn(n);
    TextMode(Co80);
    Window(2,3,78,((n Div 6)+3)*2);
    TextBackground(1); TextColor(14); ClrScr;
    GotoXY(21,1);
    Write('Окно ввода данных');
    i := 1; k := 0;
    While i <= n Do Begin
      k := k + 1;
      For j := 1 To 6 Do
        Begin
          If i > n Then Goto q1;
          GotoXY(j * 8,k+2); Write('y',i,' = '); Read(a[i]);
          Inc(i);
        End;
      End;
    End;
q1: Window(1,1,80,25);
    TextColor(7); TextBackground(1); ClrScr;
    i := 1; k := 0;
    WriteLn('+-----+');
    While i <= n Do Begin
      For j := 1 To 6 Do
        Begin
          If i > n Then Begin
            WriteLn(' '); Goto q2;
          End;
          Write(' | ',a[i]:6);
          Inc(i);
          If i > n Then
            Begin
              WriteLn(' | '); Goto q3;
            End;
        End;
      End;
      WriteLn(' | ');
      If i = n Then Goto q3;
      WriteLn(' |-----+-----+-----+','-----+-----+-----|');
    End;
q2: WriteLn;
q3: WriteLn(' +-----+','-----+');
    Window(15,17,67,20); TextBackground(0); ClrScr;
    TextColor(0); TextBackground(7);
    WriteLn('+-----+');
    WriteLn('| Нажмите <Enter> для повтора или <Esc> для выхода... |');
    WriteLn('+-----+');
    GotoXY(1,1);
  Until ReadKey=#27;
  Window(1,1,80,25); TextBackground(0); ClrScr;
End.

```

Пример 28. Подсчет суммы ряда с точностью до eps.

```

Program П_28;
Uses Crt;
Var
  eps,s,p,a1,a2,da: Real;
  k: Integer;
Begin
  ClrScr; Write('eps = '); ReadLn(eps);
  s := 1.5; a1 := 1/2; k := 3;
  Repeat
    da := 1/k; a2 := a1 * da;
    s := s + a2; p := a1 - a2; a1 := a2;
    Inc(k);
  Until p < eps;
  WriteLn('Сумма ряда = ',s:15:10);
  Repeat Until KeyPressed;
End.

```

Пример 29. Написание программы для предыдущего задания, используя оператор с предусловием.

```

Program П_29;
Uses Crt;
Var
  ds,sn1,sn2,s,xn,x,eps: Real;
  n: Integer;
Begin
  ClrScr; Write('X = '); ReadLn(x);
  Write('eps = '); ReadLn(eps);
  sn1 := sqrt(x); sn2 := sqrt(sn1) / 2;
  s := sn1 + sn2; xn := sn1 - sn2;
  sn1 := sn2; n := 3;
  While xn > eps Do
    Begin
      ds := sqrt(x) / n; sn2 := sn1 * ds;
      xn := sn1 - sn2; s := s + sn2;
      sn1 := sn2; Inc(n);
    End;
  WriteLn('Сумма ряда = ',s:10:5,' ',n);
  Repeat Until KeyPressed;
End.

```

Пример 30. Вычисление значений функций на интервале с заданным шагом.

```

Program П_30;
Uses Crt;
Const pn = 0; pk = 1; hp = 0.05;
Var
  p,t1,t2: Real;
Begin
  ClrScr;
  p := pn;
  WriteLn('+-----+')
  WriteLn('| p | t1 | t2 |');
  WriteLn('+----+-----+-----|');
  Repeat
    t1 := 2.3*p*p*p*p+0.5*p*p*p+5.95*p*p-2.5*p+1.5;
    t2 := 21.7+9.98*exp(p*ln(2));
    WriteLn('| ',p:3:1,' | ',t1:5:0,' | ',t2:5:0,' |');
    p := p + hp;
  Until p > pk;
  WriteLn('+-----+')
  Repeat Until KeyPressed;
End.

```

Пример 31. Составление программы определения наибольшего по абсолютной величине элемента матрицы A размером 3x4 среди элементов, сумма индексов которых четная. Вывод исходной матрицы и результата.

```

Program П_31;
Var
  a: Array [1..3,1..4] Of Real;
  am: Real;
  i,j,im,jm: Integer;
Begin
  For i := 1 To 3 Do
    Begin
      WriteLn('Введите ',i,'-ю строку матрицы: ');
      For j:=1 To 4 Do Read(A[I,J]);
      ReadLn;
    End;
  am := Abs(a[1,1]);
  For i := 1 To 3 Do
    For j := 1 To 4 Do
      If ((i+j) Mod 2=0) And (Abs(a[i,j] > am) Then
        Begin
          am := Abs(a[i,j]); im := i; jm := j;
        End;
    WriteLn(' :10,'Исходная матрица');
  For i := 1 To 3 Do
    Begin
      For j := 1 To 4 Do Write(a[i,j]:6:2,' :2);
      WriteLn;
    End;
  WriteLn('AbsMAX = ',am:8:3,' IMAX = ',im:3,' jm = ',jm:3);
  ReadLn;
End.

```

Пример 32. Нахождение в матрице В размером 4x2 строки с минимальной суммой элементов и ее распечатка. Вывод исходной матрицы.

```

Program П_32;
Var
  B: Array[1..4,1..2] Of Real;
  SM,S: Real;
  I,J,IM: Integer;
Begin
  For I := 1 To 4 Do
    Begin
      WriteLn('Введите ',I,'-ю строку матрицы: ');
      For J := 1 To 2 Do Read(A[I,J]);
      ReadLn;
    End;
  SM := A[1,1];
  For I := 1 To 4 Do
    Begin
      S := 0;
      For J := 1 To 2 Do S := S + A[I,J];
      If S < SM Then
        Begin
          SM := S; IM := I;
        End;
    End;
  WriteLn(' :10,'Исходная матрица');
  For I := 1 To 3 Do
    Begin
      For J := 1 To 4 Do Write(A[I,J]:6:2,' :2);
      WriteLn;
    End;
  WriteLn;
  WriteLn('Строка с максимальной суммой');
  For J := 1 To 2 Do Write(A[IM,J]);
  WriteLn;
  ReadLn;
End.

```

Пример 33. Составление программы упорядочения элементов массива А размером N ($N \leq 20$) элементов по возрастанию значений. Ввод значения N с экрана по запросу (метод «пузырька»).

```

Program П_33;
Var
  A,B: Array [1..20] Of Real;
  AOBM: Real;
  N,N1,I,J: Integer;
Begin
  Write('Введите значение N: '); ReadLn(N);
  WriteLn('Введите ',n,' элементов массива: ');
  For I := 1 To N Do
    Begin Read(A[I]); B[I] := A[I]; End;
  N1 := N-1;
  For I := N1 DownTo 1 Do
    For J := 1 To I Do
      If A[J] > A[J+1] Then
        Begin
          AOBM := A[J]; A[J] := A[J+1]; A[J+1] := AOBM;
        End;
  WriteLn('Исходный массив,' :5, 'Преобразованный');
  For I := 1 To N Do
    WriteLn(B[I]:15:2, ' :7,A[I]:10:2);
  ReadLn;
End.

```

Пример 34. Расстановка элементов массива в порядке возрастания.

```

Program П_34;
Uses Crt;
Const n = 10;
Var
  a: Array[1..n] Of Real;
  f: Real;
  i,j: Byte;
Begin
  ClrScr; WriteLn('Введите массив: ');
  For i := 1 To n Do
    Begin
      Write('a[' ,i, ' ] ');           или
      ReadLn(a[i]);                 min: Real;
    End;
  For i := 1 To n-1 Do               For i := 1 To n - 1 Do
    For j := i+1 To n Do            Begin
      Begin                          min := a[i];
        If a[j] < a[i] Then          For j := i + 1 To n Do
          Begin                      If a[j] > min Then
            f := a[j];                min := a[j];
            a[j] := a[i];             a[i] := min;
            a[i] := f;                End;
          End;
        End;
      End;
    End;
  WriteLn('Упорядоченный массив: ');
  For i := 1 To n Do Write(a[i]:5:0);
End.

```

Пример 35. Ввод матрицы чисел, определение минимального числа в каждой строке и вывод матрицы и столбца минимальных чисел. Оформление ввода и вывода данных в соответствующих окнах экрана.

```

Program П_35;
Uses Crt;
Procedure Window1(x1,y1,x2,y2,col,txtcol: Byte);
Begin
  TextBackground(Black);
  Window(x1+2,y2+1,x2+2,y2+1); ClrScr;
  Window(x2+1,y1+1,x2+2,y2); ClrScr;
  Window(x1,y1,x2,y2);
  TextBackground(col); ClrScr; TextColor(txtcol);
End;

```

```

Var
  a: Array[1..100,1..100] Of Real;
  k: Array[1..100] Of Integer;
  n,m,i,j: Integer;
  min: Real;
  ch: Char;
Label m1;
Begin
  TextBackground(Blue); ClrScr;
m1: Window1(5,3,20,19,15,0);
  Write('Введите разм. массива: '); ReadLn(n,m);
  For i := 1 To n Do
    For j := 1 To m Do
      Begin
        Write('Эл-т[' ,i,' ',j,']= '); ReadLn(a[i,j]);
      End;
  For i := 1 To n Do
    Begin
      min := a[i,1]; k[i] := 1;
      For j := 2 To m Do
        If a[i,j] < min Then
          Begin min := a[i,j]; k[i] := j; End;
      End;
  Window1(24,3,73,19,15,0);
  WriteLn(' :7*m, ' Nmin');
  For i := 1 To n Do
    Begin
      For j := 1 To m Do Write(a[i,j]:7:3);
      WriteLn(' ',k[i]);
    End;
  WriteLn('Хотите продолжить? (y/n)');
  ch := ReadKey; If ch='y' Then Goto m1;
End.

```

Пример 36. Определение в каждом столбце квадратной матрицы максимального элемента и перемена его с диагональным.

```

Program П_36;
Uses Crt;
Var
  a: Array[1..15,1..15] Of Integer;
  c,i,j,max,m,n,f: Integer;
Begin
  ClrScr;
  WriteLn('Введите размер матрицы: '); ReadLn(n);
  WriteLn('Введите матрицу: ');
  For i := 1 To n Do
    Begin
      For j := 1 To n Do
        Read(a[i,j]);
      ReadLn;
    End;
  For j := 1 To n Do
    Begin
      max := a[1,j]; c := 1;
      For i := 1 To n Do
        If a[i,j] > max Then
          Begin
            max := a[i,j]; c := i;
          End;
      WriteLn('max=',a[c,j]);
      f := a[j,j]; a[j,j] := a[c,j]; a[c,j] := f;
    End;
  WriteLn('Полученная матрица');
  For i := 1 To n Do
    Begin
      WriteLn;
      For j := 1 To n Do Write(' ', a[i,j]);
    End;
End.

```

Пример 37. Определение суммы и произведения отрицательных элементов в каждом столбце матрицы.

```

Program П_37;
Uses Crt;
Var
  b: Array[1..20,1..15] Of Integer;
  a1,a2: Array[1..15] Of Integer;
  s,i,j,p,m,n: Integer;
  flag: Boolean;
Begin
  ClrScr;
  WriteLn('Введите размеры матрицы m, n: ');
  ReadLn(m,n);
  WriteLn('Введите матрицу: ');
  For i := 1 To m Do
    Begin
      For j := 1 To n Do Read(b[i,j]);
      ReadLn;
    End;
  flag := True;
  For j := 1 To n Do
    Begin
      s := 0; p := 1;
      For i := 1 To m Do
        If b[i,j] < 0 Then Begin
          s := s+b[i,j]; p := p*b[i,j];
          flag := False;
        End;
      a1[j] := s; a2[j] := p;
    End;
  If flag Then
    WriteLn('В матрице нет отрицат. элементов')
  Else
    Begin
      WriteLn('Суммы отрицат. элементов');
      For i := 1 To n Do Write(' ',a1[i]);
      WriteLn;
      WriteLn('Произведения отрицат. элементов');
      For i := 1 To n Do Write(' ',a2[i]);
      ReadLn;
    End;
End.

```

Пример 38. Ввод массива символов заданного размера, расставление символов в алфавитном порядке.

```

Program П_38;
Uses Crt;
Var
  a: Array[1..80] Of Char;
  i,j,n: Integer;
  m: Char;
Begin
  ClrScr; WriteLn;
  Write('Введите символы, окончание ввода — * : ');
  i := 0;
  Repeat
    Inc(i); a[i] := ReadKey; Write(a[i]);
  Until a[i] = '*';
  n := i - 1; WriteLn;
  Write('Преобразованный массив: ');
  For i := 1 To n-1 Do
    For j := i+1 To n Do
      If a[i] > a[j] Then
        Begin
          m := a[i]; a[i] := a[j]; a[j] := m;
        End;
  For i := 1 To n Do
    Write(a[i]:2);
  Repeat Until KeyPressed;
End.

```

Пример 39. Ввод массива знаков, изображение прямой и обратной последовательности знаков массива.

```
Program П_39;
Uses Crt;
Var
  s1,s2: Array [1..80] Of Char;
  n,i,j: Integer;
Begin
  ClrScr; WriteLn;
  Write('Введите символы, окончание ввода — * : ');
  i := 0;
  Repeat
    i := i+1; s1[i] := ReadKey; Write(a[i]);
  Until s1[i] = '*';
  n := i - 1; WriteLn; j := 1;
  For i := n DownTo 1 Do
    Begin
      s2[j] := s1[i]; j := j + 1;
    End;
  For i := 1 To n Do Write(s1[i]:2, ' ');
  For i := 1 To n Do Write(s2[i]:2);
End.
```

Пример 40. Упорядочение массива символов по алфавиту (по возрастанию цифрового кода символа), используя метод «пузырька».

```
Program П_40;
Uses Crt;
Const nmax = 80;
Var
  m: Array[1..nmax] Of Char;
  i,j: Integer;
  u: Char;
Begin
  ClrScr; WriteLn; i := 0;
  Write('Введите символы, окончание ввода — * : ');
  Repeat
    i := i+1; m[i] := ReadKey; Write(m[i]);
  Until m[i] = '*';
  n := i - 1; WriteLn;
  For i := 1 To nmax - 1 Do
    For j := 1 To nmax - i Do
      If m[j] > m[j+1] Then
        Begin
          u := m[j]; m[j] := m[j+1]; m[j+1] := u;
        End;
    For i:=1 To nmax Do Write (a[i]:2); {Печать упорядоченного массива}
End.
```

Пример 41. Выделение в строке символов и запись в массив слов, разделенных пробелом. Распечатка полученного массива.

```

Program П_41;
Uses Crt;
Const nmax = 5;
Var
  a: Array[1..nmax] Of String;
  c,b: String;           {Стандартная длина}
  k: Integer;
Begin
  ClrScr;
  Write('Введите исходную строку: '); Read (b);
  k := 0; c := '';
  For i := 1 To Length (b) Do
    If b[i] <> ' ' Then
      c := c + b[i]       {Склейка строки}
    Else
      Begin
        Inc(k);
        a[k] := c; {Запись слова в массив A}
        c := '';
      End;
  For i := 1 To k Do
    WriteLn(a[i]);       {Печать полученного массива}
End.

```

Пример 42. Задана строка произвольной длины. Определение количества раз встречаемости используемых букв алфавита в этой строке. Запоминание результатов в массиве и вывод их на экран.

```

Program П_42;
Uses Crt;
Var
  b: String;
  ch: Char;
  l: Array['A'..'Z'] Of Integer;
  ind: Integer;
Begin
  Write('Введите исходную строку: '); ReadLn(b);
  For ch := 'A' To 'Z' Do l[ch] := 0;
  For ind := 1 To Length(b) Do l[b[ind]] := l[b[ind]] + 1;
  For ch := 'A' to 'Z' Do
    If l[ch] > 0 Then WriteLn(ch:3, ' = ', l[ch]);
End.

```

Пример 43. Расстановка слов массива в алфавитном порядке.

```

Program П_43;
Uses Crt;
Var
  a: Array[1..15] Of String;
  i,j,k: Integer;
  n: String;
Begin
  ClrScr; k := 0; WriteLn('Введите список: ');
  Repeat
    Inc(k); Write(k, '. '); ReadLn(a[k]);
    Write('Продолжить — <Enter>, прекратить ввод — <Esc>');
    ch := ReadKey;
  Until ch = #27;
  For i := 1 To k Do
    For j := 1 To k-i Do
      If a[j] > a[j+1] Then
        Begin
          n := a[j]; a[j] := a[j+1]; a[j+1] := n;
        End;
  WriteLn; WriteLn('Список по алфавиту');
  For i := 1 To k Do WriteLn(a[i]);
  Repeat Until KeyPressed;
End.

```


4. Работа со строками

Пример 44. Определение числа вхождений буквы «а» в строку.

```
Program П_44;
Uses Crt;
Var
  st: String;
  i,n: Integer;
Begin
  ClrScr; n := 0;
  WriteLn('Введите строку: '); ReadLn(st);
  For i := 1 To Length(st) Do
    If st[i] = 'a' Then Inc(n);
  WriteLn('Количество букв А = ',n);
End.
```

Пример 45. Ввод строки и определение в ней количества вхождений буквы «а», буквосочетания «гwa» и слов, оканчивающихся на гласную букву.

```
Program П_45;
Uses Crt;
Var
  st: String;
  s,l,n,i: Integer;
  ch: Char;
Begin
  ClrScr; n := 0; l := 0; s := 0; ReadLn(st);
  For i := 1 To Length(st) Do
    If st[i] = 'a' Then n := n+1;
  For i := 1 to Length(st)-2 Do
    If (st[i]='r') And (st[i+1]='w') And (st[i+2]='a') Then Inc(l);
  For i := 1 to Length(st)-1 Do
    If (st[i] In ['a','e','i','o','u','y']) And (st[i+1]=' ') Then Inc(s);
  If (st(Length(st)) In ['a','e','i','o','u','y']) Then Inc(s);
  WriteLn('Текст',st);
  WriteLn('Вхождений буквы «а» — ',n);
  WriteLn('Буквосочетаний «гwa» — ',l);
  WriteLn('Слов, оканчивающихся на гласную, — ',s);
  ch := ReadKey;
End.
```

Пример 46. Ввод строк и определение числа слов, начинающихся с «аЬ».

```
Program П_46;
Uses Crt;
Var
  x: String[20];
  i,j: Integer;
Begin
  WriteLn('Введите текст: '); ReadLn(x);
  j := 0;
  If (x[1] = 'a') AND (x[2] = 'b') Then Inc(j);
  For i := 1 To Length(x) Do
    If (x[i]=' ') And (x[i+1]='a') And (x[i+2]='b') Then Inc(j);
  WriteLn('В тексте ',j,' слов, начинающихся на «ab»'); ReadLn;
End.
```

Пример 47. Замена в строке всех букв «Т» на «***».

```

Program П_47;
Uses Crt;
Var
  a,b: String;
  i: Integer;
Begin
  ClrScr;
  Write('Введите строку: '); ReadLn(a);
  b := '';
  For i := 1 To Length(a) Do
    If a[i] = 't' Then b := b + '***' Else b := b + a[i];
  WriteLn(b);
End.

```

Пример 48. Запись в строке вместо заданного слова соответствующего числа знаков: «*».

```

Program П_48;
Uses Crt;
Var
  a,b,i: Integer;
  text,str: String[80];
  s: Char;
Begin
  ClrScr;
  Write('Введите текст: '); ReadLn(text);
  Write('Введите заменяемое слово: '); ReadLn(str);
  b := Length(str); s := '*'; a := Pos(str,text);
  Repeat
    Delete(text,a,b);
    For i := a To a + b - 1 Do Insert(s,text,a);
    a := Pos(str,text);
  Until a = 0;
  WriteLn ('Преобразованный текст: ',text);
  Repeat Until KeyPressed;
End.

```

Пример 49. Определение числа вхождений подстроки в строку.

```

Program П_49;
Uses Crt;
Label l;
Var
  e1,s1,s2: String;
  a,i,c: Integer;
  ch: Char;
Begin
  Repeat
    ClrScr;
    Write('Введите строку для ее исследования: ');
    ReadLn(s1);
l: Write('Введите подстроку: '); ReadLn(s2);
    If Length(s2) > Length(s1) Then
      Begin
        WriteLn('Ошибка!');
        Goto l;
      End;
    a := Length(s1) - Length(s2) + 1;
    c := 0;
    For i := 1 To a Do
      Begin
        e1 := Copy(s1,i,Length(s2));
        If e1 = s2 Then c := c+1;
      End;
    WriteLn('Элемент ',s2,' включен в строку ',c,'раз');
    Write('Для нового исследования нажмите <Enter>, иначе — <Esc>');
    ch := ReadKey;
  Until ch = #27;
End.

```

Пример 50. Ввод строки, показ ее зеркального отображения справа на экране.

```

Program П_50;
Uses Crt;
Var
  s1,s2: String[20];
  i: Byte;
Begin
  ClrScr; Write('Введите строку: '); ReadLn(s1); s2 := "";
  For i := Length(s1) DownTo 1 Do s2 := s2+s1[i];
  Write(s1, ' — ',s2);
End.

```

Пример 51. Преобразование строки переменной длины путем удаления рядом стоящих повторяющихся букв.

```

Program П_51;
Uses Crt;
Var
  st1,st2: String;
  i: Byte;
Begin
  ClrScr;
  Write('Введите строку: '); ReadLn(st1);
  st2 := "";
  For i := 1 To Length(st1)-1 Do
    If st1[i] <> st1[i+1] Then st2 := Concat(st2,st1[i]);
  WriteLn(' ',st2);
End.

```

Пример 52. Создание нового списка строк путем удаления из списка строк, которые начинаются на букву f.

```

Program П_52;
Uses Crt;
Var
  st1,st2: Array[1..15] Of String;
  i,n,k: Integer;
Begin
  ClrScr;
  Write('Введите количество изделий: '); ReadLn(n);
  WriteLn('Введите список изделий: ');
  For i := 1 To n Do
    Begin
      Write('Изделие',i,' : ');
      ReadLn(st1[i]);
    End;
  k := 1;
  For i := 1 To n Do
    If st1[i][1] <> 'f' Then
      Begin
        st2[k] := st1[i]; Inc(k);
      End;
  WriteLn; WriteLn('Преобразованный список');
  For i := 1 To k Do
    WriteLn(i,' ',st[i]);
End.

```

Пример 53. Определение количества различных букв в строке.

```

Program П_53;
Uses Crt;
Var
  s: String[20];
  a: Array ['A'..'Z'] Of Byte;
  l: Char;
  j: Byte;
Begin
  TextColor(White); TextBackground(Blue);
  ClrScr;
  WriteLn('Введите строку: '); ReadLn(s);
  For l := 'A' to 'Z' Do a[l] := 0;
  For j := 1 To Length(s) Do
    For l := 'A' To 'Z' Do
      If Ucase(s[j])=l Then Inc(a[l]);
  WriteLn('Итрор:');
  For l := 'A' to 'M' Do WriteLn(' ',l,'-',a[l]);
  For l := 'N' To 'Z' Do
    Begin
      GotoXY(20, - Ord('N') + Ord(l)+4);
      WriteLn(' ',l,'-',a[l]);
    End;
  l := ReadKey;
  ClrScr;
End.

```

Пример 54. Построчный ввод слов. Подсчет количества знаков в каждом слове. Составление предложения из СЛОВ.

```

Program П_54;
Uses Crt;
Var
  a: Array[1..10] Of String[15];
  st: String;
  i,n: Byte;
Begin
  ClrScr;
  WriteLn('Укажите количество слов: '); ReadLn(n);
  st := ''; WriteLn('Введите слова: ');
  For i := 1 To n Do
    Begin Write(i,'-e : '); ReadLn(a[i]); End;
  For i := 1 To n Do
    Begin
      st := st + a[i] + ' ';
      WriteLn('В слове ',i,' ',Length(a[i]),' знаков');
    End;
  WriteLn(st);
End.

```

Пример 55. Удаление в строке рядом стоящих повторяющихся символов.

```

Program П_55;
Uses Crt;
Var
  st1,st2: String;
  i: Byte;
  ch: Char;
Begin
  Repeat;
    TextBackground(Blue); ClrScr; TextColor(0);
    Window(2,2,60,10); TextBackground(White);
    ClrScr; GotoXY(23,1); WriteLn('Введите строку: ');
    GotoXY(2,2); ReadLn(st1);
    st2 := '';
    For i :=1 To Length(st1) Do
      If st1[i] <> st1[i+1] Then
        st2 := Concat(st2,st1[i]);
    Window(2,13,60,21);
    TextBackground(White); ClrScr; GotoXY(25,1);
    Write('Результат');
    GotoXY(1,2); WriteLn(' ',st2);
    ch := ReadKey;
    Window(65,2,78,10); TextBackground(Cyan);
    ClrScr; GotoXY(4,1); WriteLn('Меню');
    GotoXY(2,3); WriteLn('<Enter> — продолж');
    GotoXY(2,4); WriteLn('<Esc> — выход');
  Until ReadKey = #27;
End.

```

Пример 56. Ввод строки, состоящей из слов. Построчный показ списка использованных строк. Организация ввода и вывода информации в соответствующих окнах экрана.

```

Program П_56;
Uses Crt;
Type st = String[80];
Var
  str,s: st;
  i: Integer;
  otr: String;
  ch: Char;
Procedure Twin(x,y,xk,yk : Byte);
Const ym = 24;
Begin
  TextColor(Yellow); TextBackground(DarkGray);
  Window(x,y,xk,yk); {1,1,80,12}
  ClrScr;
  TextBackground(LightBlue);
  Window(x-3,y-1,xk-3,yk-1); {1,1,75,11}
  ClrScr;
End;
Begin
  TextMode(Co80); HighVideo;
  TextBackground(LightGray); ClrScr;
  Twin(6,3,75,13);
  Twin(6,15,75,24);
  Twin(6,3,75,13);
  GotoXY(5,3);
  WriteLn('Введите строку, состоящую из слов. (<Enter> конец ввода) ');
  i := 1; str := '';
  Repeat
    ch := ReadKey;
    If (ch >= #65) And (ch <= #122) Or (ch = ' ') Then Write(ch);
    s[i] :=ch; str := str+s[i]; Inc(i);
  Until ch=#13;
  Twin(6,15,75,24);
  GotoXY(10,15); TextColor(Cyan);
  For i:=1 To Ord(str[0]) Do
    Begin
      If (str[i] >=Chr(65)) And (str[i]<=Chr(122))
        Then
          Begin
            GotoXY(10,15); Write(Str[i]);

```

```

        End
    Else
        Begin
            GotoXY(10,15); WriteLn;
        End;
    End;
TextColor(Red+Blink); GotoXY(20,24);
WriteLn('Press any key');
Repeat Until KeyPressed;
TextMode(Co80);
End.

```

5. Работа с записями

Пример 57. Создание массива записей об изданиях: название издания, автор и год издания. По запросу о наличии изданий заданного фактора выдача информации о соответствующих изданиях.

```

Program П_57;
Uses Crt;
Type book = Record
    title: String[40];
    author: String[20];
    year: Integer;
End;
Var
    a: Array [1..20] Of book;
    s: String;
    c: Char;
    l,m: Byte;
Begin
    TextColor(White); TextBackground(Blue);
    ClrScr; WriteLn('Введите данные в формате: ');
    WriteLn(' :5,'Название',' :26,'Фамилия',' :13,'Год издания');
    WriteLn;
    m := 0;
    Repeat
        m := m + 1; GotoXY(1,m + 3); Write(' ',m,' ');
        GotoXY(5,m + 3); ReadLn(a[m].title);
        GotoXY(40,m + 3); ReadLn(a[m].author);
        GotoXY(60,m + 3); ReadLn(a[m].year);
        GotoXY(3,25);
        Write('Продолжить — клавиша <Enter>, иначе — <Esc>');
        c := ReadKey;
    Until c = #27;
    ClrScr; WriteLn;
    Write('Введите фамилию автора для поиска: ');
    ReadLn(s); WriteLn;
    WriteLn('Найденные издания: ');
    WriteLn('Год изд.: Название: ');
    For l := 1 To m Do
        If a[l].author = s Then
            WriteLn(' ',a[l].year:4, ' ',a[l].title,'');
    C := ReadKey;
    ClrScr;
End.

```

Пример 58. Создание базы данных записей о товарах: наименование товара, фирма-изготовитель, дата хранения. Выдача информации о товарах с истекшим сроком хранения. Нахождение убытка.

```

Program П_58;
Uses Crt;
Type
  Data = Record
    day, month: Byte;
  End;
  Ob = Record
    name: String[15];
    firm: String[15];
    sh: Data;
    price: Real;
  End;
Var
  sp: Array[1..80] Of Ob;
  i,n: Integer;
  k,d: Real;
  ch: Char;
  fit: Data;
Begin
  ClrScr; WriteLn; WriteLn('Ввод данных о товарах');
  WriteLn; n := 0;
  Repeat
    n := n + 1;
    With sp[n] Do
      Begin
        GotoXY(1,7 * n);
        WriteLn('Запись ',n,' '); WriteLn;
        Write('Товар — '); ReadLn(name);
        Write('Фирма — '); ReadLn(firm);
        Write('Годен до (мм дд) — ');
        Read(sh.month); ReadLn(sh.day);
        Write('Цена — '); ReadLn(price);
      End;
    GotoXY(31,1);
    Write('Продолжить? — нажмите <Enter>, иначе — <Esc>');
    ch := ReadKey;
  Until ch = #27;
  ClrScr;
  Write('Найти сумму убытка? (y/n) ');
  If ReadKey = 'n' Then Halt(1)
  Else
    Repeat
      ClrScr; WriteLn;
      Write('Введите дату контроля (мм дд): ');
      Read(fit.month); Read(fit.day); WriteLn; k := 0.0;
      For I := 1 To n Do
        With sp[i] Do
          If (fit.month > sh.month) Or (fit.month = sh.month) And (fit.day
          >= sh.day) Then
            Begin
              WriteLn('Фирма ',firm,',',name,' ',price:5:2);
              k := k + price;
            End;
          If k > 0 Then
            Begin
              WriteLn;
              WriteLn('Общая сумма убытка: ',k:6:2);
            End
          Else
            Begin
              WriteLn; WriteLn('Убытка нет');
            End;
          GotoXY(31,25);
          Write('Продолжить контроль? — <Enter>, иначе — <Esc>');
          ch := ReadKey;
        Until ch = #27;
      End.

```

Пример 59. Ввод расписания занятий на неделю, учитывая, что каждый день имеет 4 учебные «пары» и необходимо по запросу показать расписание на день или неделю.

```

Program П_59;
Uses Crt;
Label q1;
Const ned: Array[1..6] Of String[4] = ('поне', 'втор', 'сред', 'четв', 'пятн', 'субб');
Const time: Array[1..4] Of String [11] = ('8.00–9.35', '9.50–11.25', '11.40–
13.15', '13.30–15.05');
Type dis = Array[1..4] Of String[24];
  rasp = Record
    c: String[4];
    a: Dis;
  End;
B = Array[1..7] Of Rasp;
Var
  D: B;
  i, j, x: Integer;
  e: String[15];
  z: String[4];
  c: Char;
Begin
  ClrScr; WriteLn('Введите расписание: ');
  WriteLn;
  For i := 1 To 6 Do
    Begin
      WriteLn(' ', ned[i]); d[i].c := ned[i];
      For j := 1 To 4 Do
        Begin
          Write(time[j], ' '); ReadLn(e);
          d[i].a[j] := time[j] + ' ' + e;
        End;
      End;
  End;
q1: Write('Расписание на 'день', н(неделю) ?');
  ReadLn(z);
  For i := 1 To 6 Do
    Begin
      If z = ned[i] Then
        Begin
          WriteLn(d[i].c);
          For j := 1 To 4 Do
            WriteLn(d[i].a[j]);
          End;
        End;
      End;
  End;
  If z = 'н' Then
    Begin
      j := 4;
      For i := 1 To 3 Do
        Begin
          WriteLn(d[i].c, ' ':27, d[j].c);
          For x := 1 To 4 Do
            WriteLn(d[i].a[x], ' ':18, d[j].a[x]);
          WriteLn; j := j+1;
        End;
      End;
  End;
  Write('Продолжить (Да/Нет)'); ReadLn(c);
  If (c = 'Д') Or (c = 'Y') Then Goto q1
  Else
    Exit;
End.

```

Пример 60. Создание базы данных, отражающих результаты сессии. По запросу выдача фамилий студентов, сдавших экзамены на «4» и «5».


```

Program П_60;
Uses Crt;
Type
  stud = Record
    roup,name: String;
    mark1,mark2,mark3: Byte;
End;
Var
  sp: Array[1..20] Of Stud;
  i,n,m: Byte;
  ch: Char;
Begin
  ClrScr; n := 0
  Repeat
    Inc(n);
    With sp[i] Do
      Begin
        Write('Группа: '); ReadLn(group);
        Write('Фамилия: '); ReadLn(name);
        Write('Экзамен 1: '); ReadLn(mark1);
        Write('Экзамен 2: '); ReadLn(mark2);
        Write('Экзамен 3: '); ReadLn(mark3);
      End;
    Write('Для продолжения нажмите <Enter>, иначе — <Esc>: ');
    ch := ReadKey;
  Until ch = #27;
  ClrScr;
  Write('Список студентов, обучающихся на 4 и 5');
  WriteLn; WriteLn;
  WriteLn('Группа ', ' Фамилия ', 'Оценки: 1 ', ' 2 ', ' 3 ');
  For i := 1 To n Do
    With sp[i] Do
      If (mark1 > 3) Or (mark2 > 3) Or (mark3 > 3) Then
        Begin
          WriteLn(m, ' ', group, ' ', name);
          WriteLn(' ':24, mark1:4, mark2:4, mark3:4);
        End;
    Repeat Until KeyPressed;
  End.

```

Пример 61. Ввод данных о деятельности банка: имя банка, процент годовых и срок хранения. Выдача вкладчику рекомендации о размещении средств.

```

Program П_61;
Uses Crt;
Type
  bank = Record
    name: String[15];
    proc: Integer;
    srok: Integer;
  End;
Var
  a: Array[1..20] Of bank;
  i,k,m: Integer;
  n,r,s: Real;
  ch: Char;
Begin
  Repeat
    ClrScr; m := 0; TextColor(10);
    Write('Введите данные о банках: ');
    Repeat
      Inc(m);
      WriteLn; WriteLn('Запись ',m);
      Write('Имя банка: '); ReadLn(a[m].name);
      Write('Процент годовых: '); ReadLn(a[m].proc);
      Write('Время вклада: '); ReadLn(a[m].srok);
      Write('Продолжить ввод? (y/n) ');
      ch := ReadKey;
    Until (ch = 'n');
    ClrScr; GotoXY(33,3);
    WriteLn('Данные о банках');
  For i := 1 To m Do

```

```

Begin
  GotoXY(15*(i-1)+5,5); Write(a[i].name);
  GotoXY(15*(i-1)+5,6); Write(a[i].proc,'%');
  GotoXY(15*(i-1)+5,7);
  If a[i].srok = 1 Then
    Write(a[i].srok,' год')
  Else
    If a[i].srok < 5 Then
      Write(a[i].srok,' года')
    Else
      Write(a[i].srok,' лет');
  End;
Textcolor(9); GotoXY(2,4);
Write('+-----+');
For i := 1 To m Do
  Begin
    GotoXY(2,i+4);
    Write('|');
    GotoXY(76,i+4);
    Write('|');
  End;
  GotoXY(2,i+5);
  Write('+-----+');
Repeat
  TextColor(13); GotoXY(2,13);
  Write('Введите срок размещения вклада (в годах): ');
  TextColor(14); ReadLn(n);
Until (n < 13) And (n > 0);
s := 0; k := 0;
For i := 1 To m Do
  Begin
    If a[i].proc > s Then
      Begin
        s := a[i].proc; k := i;
      End;
  End;
r := exp(n * ln(1 + s/100));
TextColor(10); GotoXY(2,15);
Write('Наиболее выгодный банк –',a[k].name,');
GotoXY(2,17);
Write('Банк предлагает',a[k].proc,'% годовых.');
```

```

GotoXY(2,19);
Write('Ваш вклад увеличится в ',r:6:2,' раз.');
```

```

  TextColor(21); GotoXY(2,23);
  Write('Нажмите <Enter> для повтора или <Esc> для выхода');
```

```

Until ReadKey <> #13;
End.

```

Пример 62. Создание базы данных, в которой отражены результаты соревнований по тяжелой атлетике. Выдача информации о призерах соревнований, используя метод минимакса.

```

Program П_62;
Uses Crt;
Type
  re = Record
    country: String[20];
    name: String[20];
    ves: Byte;
    res: Byte;
  End;
Var
  sp: Array[1..20] Of re;
  i,j,n: Integer;
  k: re;
  ch: Char;
Begin
  ClrScr;
  For i := 1 To 20 Do
    With sp[i] Do
      Begin
        country := ''; name := ''; res := 0; ves := 0;
      End;
  n := 0;
  Repeat
    Inc(n); WriteLn('Спортсмен: ',n);
    With sp[n] Do
      Begin
        Write('Страна: '); ReadLn(country);
        Write('Имя: '); ReadLn(name);
        Write('Результат: '); ReadLn(res);
        Write('Вес: '); ReadLn(ves);
        WriteLn('Для продолжения нажмите пробел, окончания — *: ');
      End;
  Until ReadKey = '*';
  For i := 1 To n Do
    For j := i To n Do
      If (sp[i].res < sp[j].res) Or ((sp[i].res = sp[j].res) And (sp[i].ves > sp[j].ves))
Then
      Begin
        k := sp[i]; sp[i] := sp[j]; sp[j] := k;
      End;
  ClrScr;
  WriteLn('Призеры соревнований'); WriteLn;
  WriteLn('место,' имя', ' результат', ' страна'); WriteLn;
  If sp[1].res > 0 Then
    WriteLn('1. ',sp[1].name,' ',sp[1].res,' ':10,sp[1].country);
  If sp[2].res > 0 Then
    WriteLn('2. ',sp[2].name,' ',sp[2].res,' ':10,sp[2].country);
  If sp[3].res > 0 Then
    WriteLn('3. ',sp[3].name,' ',sp[2].res,' ':10,sp[3].country);
  ch := ReadKey;
End.

```

Пример 63. Ввод списка студентов, упорядочение его по фамилиям, используя метод «пузырька», нахождение средней оценки по предмету и средней оценки каждого студента.

```

Program П_63;
Uses Crt;
Type
  s = Record
    fam: String;
    o: Array[1..3] Of Integer;
  End;
Var
  x: Array[1..25] Of s;
  y: s;
  j,i,o1,o2,o3,q,n: Integer;
  co1,co2,co3,l: Real;
  ch: Char;
Begin
  ClrScr;
  WriteLn('Введите список группы: '); WriteLn;
  n := 0;
  Repeat
    Inc(n);
    With x[n] Do Begin
      Write(n, ' Фамилия'); ReadLn(fam);
      Write('3-и оценки: '); ReadLn(o[1],o[2],o[3]);
    End;
    WriteLn('Для продолжения нажмите <Enter>, иначе — '*' ');
  Until ReadKey = '*';
  ClrScr;
  For i := 1 To n - 1 Do
    For j := n DownTo 2 Do
      If x[j].fam < x[j-1].fam Then
        Begin
          y := x[j]; x[j] := x[j-1]; x[j-1] := y;
        End;
  WriteLn('Упорядоченный список: '); WriteLn;
  For i := 1 To n Do
    Begin
      Write(x[i].fam);
      For j := 1 To 3 Do
        Write(x[i].o[j]:10);
      WriteLn;
    End;
  WriteLn;
  o1 := 0; o2 := 0; o3 := 0;
  For i := 1 To n Do
    Begin
      o1 := o1+x[i].o[1]; o2 := o2+x[i].o[2]; o3 := o3+ x[i].o[3];
    End;
  co1 := o1/n; co2 := o2/n; co3 := o3/n;
  WriteLn('Средние оценки по предметам: ', co1:10:2,co2:10:2,co3:10:2);
  WriteLn;
  For i := 1 To n Do
    Begin
      q := 0;
      For j := 1 To 3 Do q := q + x[i].o[j];
      l := q/3;
      WriteLn(x[i].fam, ' :5,'средняя оценка',l:20:2);
    End;
  ch := ReadKey;
End.

```

Пример 64. Ввод списка студентов и результатов экзаменационной сессии (3 экзамена). Вычисление средней оценки каждого студента и средней оценки группы.

```

Program П_64;
Uses Crt;
Type

```

```

stud = Record
  group,name: String[25];
  mark: Array[1..4] Of Byte;
  sr: Real;
End;
Var
z: Array[1..100] Of stud;
sub: Array[1..4] Of String[10];
i,j,k: Integer;
x,y: Byte;
srmark: Real;
Begin
  ClrScr; WriteLn('Предметы?');
  For i:=1 To 4 Do
    Begin
      Write(i,' '); ReadLn(sub[i]);
    End;
  Write('Сколько студентов?'); ReadLn(k);
  WriteLn('Введите данные: ');
  y := wherey; x := 2; GotoXY(x,y); Write('Группа');
  x := 20; GotoXY(x,y); Write('Имя');
  x := 40; GotoXY(x,y); Write('Предмет');
  x := 50; GotoXY(x,y); WriteLn('Оценка');
  For i := 1 To k Do
    With z[i] Do
      Begin
        ReadLn(group);
        x := 20; GotoXY(x,WhereY-1); ReadLn(name);
        sr := 0; x := 40; y := WhereY-1;
        For j := 1 To 4 Do
          Begin
            GotoXY(x,y); Write(sub[j]);
            GotoXY(x+10,y); ReadLn(mark[j]);
            sr := sr+mark[j]; y := WhereY;
          End;
        z[i].sr:=z[i].sr/4;
      End;
    srmark := 0;
  For i := 1 To k Do
    With z[i] Do srmark := srmark+sr;
  srmark := srmark/k;
  Write('Средний балл: ',srmark:5:2);
  Repeat Until KeyPressed;
End.

```

6. Работа с файлами

Пример 65. Создание файла со значениями функции $y = \sin(x)$ на интервале $a \leq x \leq b$ с шагом изменения аргумента h . Вычисление среднего арифметического элементов файла с номерами от m до n .

```

Program П_65;
Uses Crt;
Type
  TF = File Of Real;
Var
  f: TF;
  x,a,b,hx,sr,y: Real;
  i,k,m,n: Integer;
  l: Boolean;
  ch: Char;
Begin
  l := True;
  Repeat
    ClrScr;
    Write('Интервал изменений аргумента (a, в): ');
    ReadLn(a,b); Write('Шаг: '); ReadLn(hx);
    Assign(f,'file.dat');
    Rewrite(f);
    x := a; k := Trunc((b - a)/hx) + 1;
    For i := 1 To k Do
      Begin
        y := Sin(x);
        Write(f,y);
        x := x+hx;
      End;
    Close(f);
    WriteLn('Файл file.dat содержит элементы с номерами 0..',k-1);
    WriteLn;
    Write('Введите интервал номеров элементов файла: ');
    ReadLn(m,n);
    Reset(f);
    Seek(f,m);
    {Установить файл f на обработку элемента с номером m}
    sr := 0; k := n - m + 1;
    For i := 1 To k Do
      Begin
        Read(f,x);
        sr := sr + x;
      End;
    Close(f);
    sr := sr/k;
    WriteLn('Среднее значение элементов равно: ',sr:6:3);
    WriteLn;
    WriteLn('Для продолжения нажмите любую клавишу, для выхода —');
    WriteLn('<Esc>');
    ch := ReadKey;
    If ch = #27 Then l := False;
  Until l = False;
End.

```

Пример 66. Формирование символьного файла из заглавных и строчных букв латинского алфавита. Вывод на экран всех строчных букв. Завершение ввода данных символом «точка».

```

Program П_66;
Uses Crt;
  t = File Of Char;
Var
  fv: t;
  s: Char;
Begin
  ClrScr; s := '';
  Assign(fv,'simv'); Rewrite(fv);
  Write('Введите последовательность знаков: ');
  While s <> '.' Do Begin Read(s); Write(fv,s); End;
  Close(fv); Reset(fv); WriteLn;
  Write('Последовательность строчных букв: ');
  While Not Eof(fv) Do
    Begin
      Read(fv,s);
      If (s >= 'a') And (s <= 'z') Then Write(s);
    End;
  Close(fv); s := ReadKey;
End.

```

Пример 67. Задан файл целых чисел. Добавление к файлу одного компонента.

```

Program П_67;
Uses Crt;
Label q1;
Const many = 20; {Максимальное количество вводимых чисел}
Type
  t = File Of Integer;
Var
  f1,f2: t;
  number,quant,i: Integer;
  ch: Char;
Begin
  ClrScr;
q1: Write('Укажите количество вводимых чисел: ');
  ReadLn(quant);
  If quant > many Then
    Begin
      WriteLn('Ошибка ввода'); WriteLn;
      Goto 1;
    End;
  Assign(f1,'fil1'); Assign(f2,'fil2'); Rewrite(f1);
  WriteLn('Введите ',quant,' чисел в файл 1: ');
  For i := 1 To quant Do
    Begin
      Read(number); Write(f1,number);
    End;
  WriteLn; Close(f1); Reset(f1); Rewrite(f2);
  WriteLn;
  WriteLn('Переписываем числа в файл 2: ');
  While Not Eof(f1) Do
    Begin
      Read(f1,number); Write(f2,number);
    End;
  Close(f1); Close(f2); WriteLn;
  Write('Содержание файла 2: ');
  Reset(f2);
  While Not Eof(f2) Do
    Begin
      Read(f2,number); Write(number,' ');
    End;
  WriteLn; WriteLn;
  Write('Укажите значение добавляемого к f2 компонента: '); Read(number);
  Write(f2,number); Close(f2);
  WriteLn; Rewrite(f1); Reset(f2);
  WriteLn('Переписываем из f2 в f1');
  While Not Eof(f2) Do
    Begin
      Read(f2,number); Write(f1,number);
    End;
  End.

```



```

    End;
    Close(f1); Close(f2); WriteLn;
    Write('Новое содержимое файла 1: ');
    Reset(f1);
    While Not Eof(f1) Do
        Begin
            Read(f1,number); Write(number,"");
        End;
    Close(f1); WriteLn;
    Write('Новое содержимое файла 2: ');
    Reset(f2);
    While Not Eof(f2) Do
        Begin
            Read(f2,number); Write(number,"");
        End;
    Close(f2); ch := ReadKey;
End.

```

Пример 68. Создание на диске базы данных успеваемости студентов: фамилия студента и его три оценки. Выдача информации о средней успеваемости всех студентов и каждого из группы.

```

Program П_68;
Uses Crt;
Type
    s = Record
        name: String[8];
        a1,a2,a3: Integer;
    End;
text = File Of s;
Var
    xr: Array[1..n] Of s;
    x: s;
    idn: String[8];
    srxr,sa1,sa2,sa3,sa: Real;
    i,n: Integer;
    l: Char;
    f: text;
Begin
    ClrScr; Assign(f,'z.txt'); Reset(f);
    i := 0;
    Repeat
        Inc(i); WriteLn('Запись',i);
        Write('Фамилия: '); Read(f,xr[i].name);
    Until i = n;

```

```

Write('Экзамен 1: '); Read(f,xr[i].a1);
Write('Экзамен 2: '); Read(f,xr[i].a2);
Write('Экзамен 3: '); Read(f,xr[i].a3);
WriteLn('Для продолжения — <Enter>, окончания — <Esc>');
ch := ReadKey;
Until ch <> #27;
n := i; Close(f); Reset(f);
For i := 1 To n Do Read(f,xr[i]);
Close(f);
For i := 1 To n Do
  WriteLn(i,',',xr[i].name,' ',xr[i].a1,' ',xr[i].a2,' ',xr[i].a3);
WriteLn;
sa1 := 0; sa2 := 0; sa3 := 0;
WriteLn('Средние показатели успеваемости группы: ');
WriteLn;
For i := 1 To n Do
  Begin
    sa1 := sa1 + xr[i].a1; sa2 := sa2 + xr[i].a2;
    sa3 := sa3 + xr[i].a3;
  End;
sa1 := sa1/n; sa2 := sa2/n; sa3 := sa3/n;
sa := (sa1+sa2+sa3)/3.0;
WriteLn('Экзамен 1:',sa1:4:2,' Экзамен 2:',sa2:4:2, 'Экзамен 3:',);
WriteLn(,sa3:4:2,'Общий показатель',sa:4:2);
Repeat
  WriteLn;
  Write('Данные о студенте (Введите фамилию: ');
  ReadLn(idn);
  For i := 1 To n Do
    If idn = xr[i].name Then
      srxr := (xr[i].a1 + xr[i].a2 + xr[i].a3) / 3.0;
  WriteLn;
  Write('Данные по ',idn,', ',srxr:5:3); WriteLn;
  WriteLn('Новые данные — <Enter>, выход — <Esc>');
  ch := ReadKey;
Until ch <> #27;
End.

```

Пример 69. Написание программы, позволяющей по выбору создавать, находить, переименовывать, дополнять, читать, удалять файл и записывать информацию в файл.

```

Program П_69;
Uses Crt, DOS;

```

```

Var c,i,u: Char;
t: Integer;
f: Text;
r: Searchrec;
o,p: String;
Procedure Menu;
Begin
  WriteLn(' 1) Записать в новый файл');
  WriteLn(' 2) Найти файл');
  WriteLn(' 3) Прочитать из файла');
  WriteLn(' 4) Дописать в файл');
  WriteLn(' 5) Переименовать файл');
  WriteLn(' 6) Удалить файл');
  WriteLn(' 7) Выход');
End;
Procedure Creation;
Begin
  ClrScr;
  Write('Введите имя файла: '); ReadLn(o);
  WriteLn('Введите текст (макс. кол-во символов — 100), выход — <Enter>: ');
  Write(' '); Assign(f,o+'.txt'); Rewrite(f); t := 1;
  Repeat
    u := ReadKey;
    Write(u);
    t := t + 1;
    Write(f,u);
    If t = 100 Then
      Begin
        Close(f); Halt;
      End;
  Until u = #13;
  Close(f);
  WriteLn; WriteLn('Файл записан. ');
  c := ReadKey;
End;
Procedure App;
Begin
  ClrScr;
  Write('Введите имя файла: '); ReadLn(o);
  WriteLn('Допишите текст(макс. кол-во символов — 100), выход — <Enter>: ');
  Write(' ');
  Assign(f,o+'.txt'); Append(f); t := 1;
  Repeat

```

```

    u := ReadKey; Write(u); t := t+1;
    Write(f,u);
    If t = 100 Then
        Begin
            Close(f); Halt;
        End;
    Until u = #13;
    Close(f); WriteLn;
    WriteLn('Информация добавлена в файл');
    c := ReadKey;
End;
Procedure Find;
Begin
    ClrScr;
    WriteLn(' :25, 'Файлы с расширением. txt:');
    FindFirst(*.txt', archive, r);
    While DosError = 0 Do
        Begin
            Write(r.name, ' ');
            FindNext(r);
        End;
    c := ReadKey;
End;
Procedure Delete;
Begin
    ClrScr;
    Write('Введите имя файла без расширения: ');
    ReadLn(o); Assign(f, o+'.txt'); Reset(f); Erase(f);
    Write('Файл удален. ');
    c := ReadKey;
End;
Procedure Ren;
Begin
    ClrScr;
    Write('Введите старое имя: '); ReadLn(o);
    Write('Введите новое имя: '); ReadLn(p);
    Assign(f, o+'.txt'); Rename(f, p+'.txt');
    WriteLn('Файл переименован. ');
    c := ReadKey;
End;
Procedure Readto;
Begin
    ClrScr;
    Write('Введите имя файла: '); ReadLn(o);
    Assign(f, o+'.txt'); Reset(f);
    Write(' ');
    While Not Eof(f) Do
        Begin
            Read(f, u); Write(u);
        End;
    WriteLn;
    c := ReadKey;
End;
Begin
    Repeat
        ClrScr; TextColor(11);
        menu;
        i := ReadKey;
        Case i Of
            #49: creation;
            #50: find;
            #51: readto;
            #52: app;
            #53: ren;
            #54: delete;
        End;
    Until i = #55;
End.

```

Пример 70. Формирование файла произвольного имени, запись в этот файл последовательности целых чисел и показ последнего компонента файла, минимального из четных, и разности первого и последнего компонента файла.

```

Program П_70;
Uses Crt;
Const
  MaxNameLen = 12; {Максимальная длина полного имени файла}
  ExtNumber = 3;   {Максимальная длина расширения}
  MaxPartNumber = 100; {Максимальное число компонентов файла}
  MaxWordRange = 65534; {Максимальное значение используемых}
  {переменных типа Word}
  StdPointPos = 9; {Максимальная позиция точки в имени файла}
  Esc = #27; {Коды используемых клавиш}
  Enter = #13;
  BackSpace = #8;
  Space = ' '; {Константы типа Char}
  Point = '.';
  NilStr = "";
  AllowedSymbols = ['a'..'z','A'..'Z','_','$','#'];
  {Множество символов, допустимых в имени файла}
  AllowedNumbers = ['0'..'9']; {Множество цифр, допустимых в имени файла}
  Zero = 0; {Идентификаторы ноля и единицы ...}
  One = 1; Two = 2;
Type
  FileName = String[MaxNameLen]; {Тип данных, которые необходимо}
  {получить по условию задачи}
  NeededData = Record
    Summa: LongInt; {Сумма компонентов файла}
    Last: Integer; {Последний компонент}
    MinD2: Integer; {Минимальное из четных файла}
    DecFL: LongInt; {Разность первого и последнего компонента}
  End;
Procedure CreateRandomFile(Name: FileName); {Процедура, создающая}
{случайный файл типа File Of Integer}
Var
  F: File Of Integer; {Идентификатор файла}
  Default: Integer; {Текущий компонент}
  Number: Word; {Его номер}
  MaxNumber: Word; {Число компонента}
Begin
  Assign(F,Name);
  ReWrite(F);
  Randomize;
  MaxNumber := Round(Random(MaxPartNumber)+One);
  For Number := One To MaxNumber Do
    Begin
      Default := Round(Random(MaxWordRange)-MaxInt);
      Write(F,Default);
    End;
  Close(F);
End;
Function ReadFileName: FileName; {Считывает имя файла с проверкой}
{ошибок}
Var
  Key: Char; {Клавиша, нажимаемая пользователем}
  Name: FileName; {Имя файла, вводимое пользователем}
  PointPos: Byte; {Условная позиция точки в имени}
  NoPoint: Boolean; {Есть точка в имени или ее нет}
Begin
  Name := NilStr; PointPos := StdPointPos;
  NoPoint := True;
  Repeat

```

```

If KeyPressed Then
  Begin
    Key := ReadKey;
    If ((Key in AllowedSymbols) Or ((Key In AllowedNumbers) And
    (Length(Name) > Zero))) And (Length(Name)<MaxNameLen) And
    (Length(Name)+One <> PointPos) And ((Not(NoPoint) And (Length(Name)-
    PointPos<ExtNumber)) Or NoPoint) Then
      Begin Name := Name+Key; Write(Key);
      End;
    If (Key = Point) And (NoPoint) Then
      Begin
        Write(Point); Name := Name+Key;
        PointPos := Length(Name); NoPoint := False;
      End;
    If (Key = BackSpace) And (Length(Name)>Zero) Then
      Begin
        If Name[Length(Name)] = Point Then
          Begin
            NoPoint := True;
            PointPos := StdPointPos;
          End;
        Delete(Name,Length(Name),One);
        GotoXY(WhereX-One,WhereY); Write(Space);
        GotoXY(WhereX-One,WhereY);
      End;
    If Key = Esc Then Halt(One); {Выход из программы}
  End;
Until (Key = <Enter>);
WriteLn;
ReadFileName := Name;
End;
Procedure GetNeededData(Name:Filename; Var Data:NeededData); {Процедура}
{возвращает в Data информацию о файле Name}
Var
  F: File Of Integer;           {Идентификатор файла}
  Default: Integer;           {Текущий компонент}
  First: Integer;
Begin
  Assign(F,Name); Reset(F); Read(F,First);
  With Data Do Begin
    MinD2 := First; Summa := Zero;
    While Not(Eof(F)) Do
      Begin
        Read(F,Default);

```

```

        If (Default < MinD2) And (Default Mod Two = Zero) Then
            MinD2 := Default; Inc(Summa, Default);
        End;
        DecFL := First - Default; Last := Default;
    End;
    Close(F);
End;
Var
    Name: FileName; Data: NeededData;
    Key: Char;
Begin
    ClrScr;
    Write('Введите имя файла (???????.??? / <Esc> — Выход)_');
    Name := ReadFileName;           {Вводим имя файла}
    CreateRandomFile(Name);         {Задаем его случайно}
    GetNeededData(Name, Data);      {Подсчитываем нужное}
    WriteLn;                         {Начинаем вывод результатов}
    With Data Do Begin
        WriteLn('Сумма компонентов файла _____', Summa);
        WriteLn('Последний компонент файла _____', Last);
        WriteLn('Минимальное из четных _____', MinD2);
        WriteLn('Разность первого и последнего компонента __', DecFL);
    End;
    WriteLn; WriteLn('Выход — любая клавиша'); {Заканчиваем вывод}
    Repeat Until KeyPressed;
    Key := ReadKey;
End.

```

7. Организация подпрограмм типа «Процедура»

Пример 71. Рисование на экране с помощью одной процедуры трех окон с различными заголовками.

```

Program П_71;
Uses Crt;
Var
    c: Char;
    st: String;
Procedure Win(x1,y1,x2,y2,shadow,paper,inc: Byte; s: String);
Begin
    Window(x2 + 1,y1 + 1,x2 + 2,y2 + 1);
    TextBackground(shadow); ClrScr;
    Window(x1 + 2,y2 + 1,x2 + 2,y2 + 1);
    TextBackground(shadow); ClrScr;
    Window(x1,y1,x2,y2); TextBackground(paper);
    ClrScr; TextColor(inc);
    GotoXY(((x2-x1) Div 2) - (Length(s) Div 2),1);
    Write(s);
End;
Begin
    ClrScr; TextBackground(1); ClrScr; c := ReadKey;
    st := 'Меню';
    Win(52,3,74,12,0,15,2,st); c := ReadKey; st := 'Функция';
    Win(3,3,48,12,0,13,2,st); c := ReadKey; st := 'Результат';
    Win(3,15,48,24,0,14,3,st); c := ReadKey; ClrScr;
End.

```

Пример 72. Умножение элементов четных строк на максимальный элемент, а нечетных — на минимальный элемент матрицы.

```

Program П_72;
Uses Crt;
Var
  a: Array[1..10,1..10] Of Integer;
  n,m,i,j: Integer;
Procedure Art (ni,nj: Integer);
Var
  amin,amax: Integer;
Begin
  amin := a[1,1]; amax := a[1,1];
  For i := 1 To ni Do
    For j := 1 To nj Do
      Begin
        If a[i,j] < amin Then amin := a[i,j];
        If a[i,j] > amax Then amax := a[i,j];
      End;
    For i := 1 To 2 Do
      For j := 1 To nj Do
        a[2*i,j] := amax*a[2*i,j];
    For i := 1 To 3 Do
      For j := 1 To nj Do
        a[2*i-1,j] := amin*a[2*i-1,j];
  TextColor(12); WriteLn;
  WriteLn('Преобразованный массив');
  For i := 1 To ni Do
    For j := 1 To nj Do
      Begin
        GotoXY(10*j-1,i*2+13); TextColor(10); Write (a[i,j]);
      End;
    For j := 1 To nj Do
      Begin
        GotoXY(j*12 - 8,16); Write('S = ',z[j]);
      End;
  zmin := z[1]; d := 1;
  For j := 1 To nj Do
    If z[j] < zmin Then
      Begin
        zmin := z[j]; d := j;
      End;
  For i := 1 To ni Do a[i] := b[i,d];
  For i := 1 To ni Do
    Begin
      TextColor(14); GotoXY(j*12+8,i*2+4); Write(a[i]);
    End;
  GotoXY(1,20); TextColor(12);
  Write('Элементы полученного массива: ');
  For i := 1 To ni Do
    Write(a[i], ' ');
End;
Begin
  Repeat;
  ClrScr;
  Write('Число строк: '); ReadLn(m);
  Write('Число столбцов: '); ReadLn(n);
  asmod(m,n);
  GotoXY(12,25); TextColor(9);
  Write('<Enter> — для продолжения, <Esc> — для выхода');
  Until ReadKey = #27;
End.

```

Пример 75. Написание программы, осуществляющей с помощью меню выполнение следующих операций: ввод данных об итогах сессии (фамилия студента + 3 экзамена), подготовка и вывод списка студентов, имеющих только хорошие и отличные оценки.


```

Program П_75;
Uses Crt;
Type
  student = Record
    fam: String[50];
    gr: String[15];
    e1,e2,e3: Integer;
  End;
Var
  k: Integer;
  f: File Of student;
  s: student;
Procedure Color(f,b: Byte);
Begin
  TextColor(f);
  TextBackground(b);
End;
Procedure Showwin(x0,y0,xk,yk,f,b: Byte; s: String);
Var
  xs,i,j: Integer;
Begin
  color(f,b);
  GotoXY(x0,y0); Write(Chr(201));
  For i := x0 To xk-1 Do Write(Chr(205));
  Write(Chr(187));
  For i := y0+1 To yk-1 Do
    Begin
      GotoXY(x0,i); Write(Chr(186));
      For j := x0 To xk-1 Do Write(' ');
      Write(Chr(186));
    End;
  GotoXY(x0,yk); Write(chr(200));
  For i := x0 To xk-1 Do Write(Chr(205));
  Write(Chr(188));
  color(0,15);
  GotoXY(x0+2,yk+1);
  For i := x0 To xk+1 Do Write(' ');
  For i := y0+1 To yk Do
    Begin
      GotoXY(xk+2,i); Write(' ');
    End;
  xs := x0+(xk-x0) Div 2-Length(s) Div 2;
  color(f,b);
  GotoXY(xs,y0); Write(s);
End;
Function Menu : Integer;
Var
  m: Array[1..4] Of String;
  x,i,ch: Integer;
  c: Char;
Begin
  x := 30;

```

```

Read(f,s);
With s Do Begin
  If (e1>3) And (e2>3) And (e3>3) Then
    Begin
      GotoXY(5,13+a);
      WriteLn(fam:20,gr:10,e1:5,e2:5,e3:5);
    End;
  Inc(a);
End;
End;
Close(f);
End;
Begin
  color(15,1); ClrScr; Assign(f,'student.dat');
  Repeat
    k := menu;
    Case k Of
      1: inputd;
      2: outputd;
      3: work;
    Else
      End;
  Until k = 4;
  color(15,0); ClrScr;
End.

```

Пример 76. Создание базы данных расписания авиарейсов: номер рейса, пункт назначения, время вылета. Реализация поиска в созданной базе данных, т. е. по запросу, содержащему пункт назначения и время вылета. Выдача сведений о номерах рейсов и времени вылета, удовлетворяющих запросу. Если искомым рейсов нет, то выдача соответствующего текста.

```

Program П_76;
Uses Crt;
Type
  avia = Record
    num: Integer;
    city: String [20];
    time: Real;
  End;
Var
  a: avia;
  f: File Of avia;
  k: Byte;
  sl: Boolean;

```

```

ch: Char;
Procedure Inp;           {Процедура ввода данных}
Var
  i: Byte;
  fil: String;
Begin
  ClrScr;
  Write('Введите имя файла: '); ReadLn(fil);
  Assign(f,fil+'.dbf'); Rewrite(f); i := 0;
  Repeat
    Inc(i);
    WriteLn; WriteLn('Запись',i); WriteLn;
    With a Do Begin
      Write('Аэропорт: '); ReadLn(city);
      Write('Номер рейса: '); ReadLn(num);
      Write('Время вылета (ч.м): '); ReadLn(time);
    End;
    Write(f,a);
    WriteLn(' :30,<Enter> — для продолжения, <Esc> — окончания ввода');
  Until ReadKey = #27;
  Close(f);
End;
Procedure Find;         {Процедура поиска}
Var
  c1: String;
  t1: Real;
  n: Byte;
Begin
  ClrScr;
  WriteLn(' :20, Режим поиска информации');
  Write('Укажите аэропорт: '); ReadLn(c1);
  Write('Укажите время вылета: '); ReadLn(t1);
  n := 0; Reset(f);
  While Not Eof(f) Do
    Begin
      Read(f,a); With a Do
        If (c1 = city) And (t1 < time) Then
          Begin
            Inc(n);
            WriteLn(n, ' ',city, ' ',num, ' ',time:5:2);
          End;
        End;
      Close(f);
      If n = 0 Then WriteLn('Таких рейсов нет');
    End;
  End;
Begin                   {Программа}
  inp;
  sl := True;
  While sl Do
    Begin
      find; WriteLn; Write('Продолжить поиск? (y/n): ');
      If ReadKey = 'n' Then sl := False;
    End;
  End;
End.

```

8. Организация подпрограмм типа «Функция»

Пример 77. Замена в строке символов первого вхождения заданной последовательности знаков соответствующим числом пробелов.

```

Program П_77;
Uses Crt;
Var
  a,i,c: Integer;
  text,subtext,ntext: String;
  s: Char;
Function Asm(Var substr,str: String): String;
Begin
  a := Pos(substr,str); c := Length(substr);
  Delete(str,a,c); s := ' ';
  For i := 1 To c Do Insert(s,str,a);
  asm := str;
End;
Begin
  Repeat
    ClrScr;
    Write('Введите текст: '); ReadLn(text);
    Write('Введите подстроку: '); ReadLn(subtext);
    ntext := asm(text,subtext);
    WriteLn('Преобразованный текст: ',ntext);
    Write('<Esc> — для выхода, <Enter> — для продолжения');
  Until ReadKey = #27;
End.

```

Пример 78. Ввод координат нескольких точек на плоскости (≤ 10) и определение наибольшего расстояния между ними.

```

Program П_78;
Uses Crt;

```

```

Type
  a = Array[1..10] Of Integer;
Var
  xt,yt: a;
  answer: Real;
  i,j,n: Integer;
Function Distance(Var xt,yt: a; n: Integer): Real;
Var
  i,j: Integer;
  max,work: Real;
Begin
  max := 0; work := 0;
  For i := 1 To n Do
    For j := 1 To n Do
      Begin
        If i <> j Then
          Begin
            work := sqrt((xt[i]-xt[j])*(xt[i]-xt[j])+(yt[i]-yt[j])*(yt[i]-
yt[j]));
            If work > max Then max := work;
          End;
        End;
      End;
    distance := max;
  End;
Begin
  Repeat
    ClrScr; TextColor(LightMagenta); {Светло-фиолетовый}
    GotoXY(10,8);
    Write('Укажите число точек: '); ReadLn(n);
    For i := 1 To n Do
      For j := 1 To n Do
        Begin
          End;
      ClrScr; TextColor(Yellow);           {Желтый}
      For I := 1 To n Do
        Begin
          GotoXY(10,i); Write('xt[' ,i,']='); ReadLn(xt[i]);
          GotoXY(30,i); Write('yt[' ,i,']='); ReadLn(yt[i]);
        End;
      answer := distance(xt,yt,n);
      ClrScr; TextColor(LightMagenta); GotoXY(10,10);
      Write('Наибольшее расстояние между точками равно: ');
      Write (answer:7:2);
      GotoXY(10,14); TextColor(LightRed);           {Розовый}
      Write('<Esc> — для выхода, <Enter> — для продолжения');
    Until ReadKey=#27;
  End.

```

Пример 79. Ввод строк и расстановка знаков строки по алфавиту.

```

Program П_79;
Uses Crt;
Var
  text1,text2: String;
Function Abc(Var text1: String): String;
Var
  i,j: Integer;
  tx: String;
Begin
  tx := "";
  For i := 97 To 122 Do
    For j:=1 To Length(text1) Do
      If Ord(text1[j]) = i Then tx := tx + text1[j];
    abc := tx;
  End;
Begin
  Repeat
    ClrScr; TextColor(LightCyan); GotoXY(1,3);
    Write('Введите строку: '); ReadLn(text1);
    text2 := abc(text1); GotoXY(1,5); TextColor(White);
    Write('Результат преобразования: ',text2);
    GotoXY(1,15); TextColor(LightRed);
    Write('Press <Esc> to exit, any key to repeat');
  Until ReadKey = #27;
End.

```

Пример 80. Определение максимального элемента матрицы при использовании подпрограммы-функции.

```

Program П_80;
Uses Crt;
Var
  e: Array[1..20,1..20] Of Integer;
  c,n,ni,nj: Integer;
Function Asm(Var fni,fnj: Integer): Integer;
Var
  amax,i,j: Integer;
Begin
  For i := 1 To fni Do
    For j := 1 To fnj Do
      Begin
        GotoXY(j*12-10,i*2);
        Write('E['i','j,'] = '); ReadLn(e[i,j]);
      End;
    amax := 0;
    For i := 1 To fni Do
      For j := 1 To fnj Do
        If e[i,j] > amax Then amax := e[i,j];
      asm := amax;
    End;
  Begin
    Repeat
      ClrScr;
      Write('Число строк матрицы (N): '); ReadLn(ni);
      Write('Число столбцов матрицы (M): '); ReadLn(nj);
      ClrScr; c := 0; c := asm(ni,nj); WriteLn;
      WriteLn('Максимальный элемент: ',c);
      WriteLn('Для продолжения нажмите <Enter>, иначе — <Esc>');
    Until ReadKey = #27;
  End.

```

9. Организация программного модуля

Пример 81. Разработка информационной системы, позволяющей: записывать результаты исследований, представленные в виде пяти действительных чисел, сохранять данные в файле, вызывать их из файла, вычислять среднее значение и среднеквадратичное отклонение для всей совокупности, строить графики для каждой последовательности данных (5), освобождать экран от данных.

Разделим экран на три окна для: ввода информации; выдачи результатов вычислений и графиков; системного меню. На экране ввода покажем 5 строк для ввода информации. На каждой строке выделим по 5 полей для записи результатов соответствующего эксперимента.

Текст модуля, содержащего подпрограммы основных операций системы и организации окон экрана, может быть организован следующим образом*:

* Алгоритм и текст программы подготовлен Алексеем Федоровичем Меняевым.

```
Unit П_81_m;
Interface
Uses Crt, Graph;
Type
  dtype = Record
    s: String[10];
    a: Array [1..5] Of Real;
  End;
  darray = Array[1..5] Of dtype;
Procedure Window1(x1, y1, x2, y2: Integer); {Окно 1}
Procedure Window2(x1, y1, x2, y2: Integer); {Окно 2}
Procedure Window3(x1, y1, x2, y2: Integer); {Окно 3}
Procedure Frame(x1, y1, x2, y2: Integer; active: Boolean);
Procedure Place(x1,y1,x2,y2: Integer); {Раскраска полей данных}
Procedure Clear_d(Var d: darray); {Чистка d}
Procedure Writte_d(Var d: darray); {Отображение массива d}
Procedure New(Var d: darray); {Ввод данных}
Procedure Save(Var d: darray); {Сохранение на диске}
Procedure Load(Var d: darray); {Загрузить данные с диска}
Procedure Medium(x1,y1,x2,y2: Integer; Var d: darray);
Procedure Pict(x1,y1,x2,y2: Integer; Var d: darray);
Implementation
Procedure Window1; {Окно 1}
Begin
  SetFillStyle(1, White); {Белый цвет окна 1}
  Bar(x1, y1, x2, y2); {Изображение окна 1}
  SetColor(LightGray); {Цвет текста}
  SetLineStyle(0, 0, 3); {Тип линии}
  Rectangle(x1-3, y1-3, x2+3, y2+3); {Рамка окна}
  MoveTo(x1+200, y1); {Начало текста в окне 1}
  SetColor(Cyan); {Цвет заголовка}
  OutText('Input'); {Напечатать заголовок}
End;
Procedure Window2; {Окно 2}
Begin
  SetFillStyle(1, White); {Белый цвет окна 2}
  Bar(x1, y1, x2, y2); {Изображение окна 2}
  SetColor(LightGray); {Цвет текста}
  SetLineStyle( 0, 0, 3); {Тип линии}
  Rectangle(x1-3, y1-3, x2+3, y2+3); {Рамка окна}
  MoveTo(x1+200, y1); {Начало текста в окне 2}
  SetColor(Cyan); {Цвет заголовка}
  OutText('Calculations'); {Напечатать заголовок}
End;
Procedure Window3; {Окно 3}
Var
  x: Integer;
Begin
  SetFillStyle(1, Cyan); {Голубой цвет окна 3}
  Bar(x1, y1, x2, y2); {Изображение окна 3}
  SetColor(LightGray); {Цвет текста}
  SetLineStyle(0, 0, 3); {Тип линии}
```

Rectangle(x1-3, y1-3, x2+3, y2+3);	{Рамка окна}
MoveTo(x1+40, y1);	{Начало текста в окне 3}
SetColor(DarkGray);	{Цвет заголовка}
OutText('Menu');	{Напечатать заголовок}
x := x1 + 5;	
OutTextXY(x, y1+10, 'Input F1');	{Пункт Ввод}
OutTextXY(x,y1+20,'Save F2');	{Пункт Сохранить}
OutTextXY(x,y1+30,'Load F3');	{Пункт Загрузить}
OutTextXY(x, y1+40, 'Clean F4');	{Пункт Чистка}
OutTextXY(x,y1+50,'Calculation F5');	{Пункт Вычисление}
OutTextXY(x,y1+60,'Graph F6');	{Пункт График}
OutTextXY(x, y1+70, 'Quit F9');	{Пункт Выход}
End;	
Procedure Frame;	{Рамка активного окна}
Begin	
If active Then	{Активная рамка}
Begin	
SetLineStyle(0, 0, 3);	{Толстая линия рамки}
SetColor(Green);	{Зеленый цвет рамки}
Rectangle(x1-3, y1-3, x2+3, y2+3);	{Активная рамка}
Exit;	
End;	{Пассивная рамка}
SetLineStyle(0, 0, 3);	{Толстая линия рамки}
SetColor(LightGray);	{Обычный цвет рамки}
Rectangle(x1-3, y1-3, x2+3, y2+3);	{Пассивная рамка}
Exit;	
End;	
Procedure Place;	{Раскраска полей данных}
Var i,j: Integer;	
x,y: Integer;	
Begin	
y := y1+20;	{Первая строка данных}
SetFillStyle(1, Cyan);	{Цвет поля данных}
For i := 1 To 5 Do	{Цикл строк}
Begin bar(x1+16, y, x1+96, y+14);	{Поле текста}
y := y+28;	{Через строку}
End;	
x := 120; {Числовые поля}	{Начало чисел в строке}
For i := 1 To 5 Do	{Цикл строк}
Begin	
y := y1+20;	{Первая строка чисел}
For j := 1 To 5 Do	{Цикл полей в строке}
Begin bar(x1+x, y, x1+x+40, y+14);	{Поле числа}


```

        y := y+28;                                {Через строку}
    End;
    x := x+64;                                    {Следующее поле в строке}
End;
Procedure Clear_d;                               {Очистка полей данных}
Var i,j : Integer;
Begin
    For i:=1 To 5 Do With d[i] Do                 {Цикл записей}
        Begin
            s := "";                             {Чистое поле}
            For j := 1 To 5 Do a[j] := 0.0;       {Запись нулей}
        End;
    End;
End;
Procedure Write_d;                             {Отображение записей файла}
Var
    i,j,x,y: Integer;
Begin
    Textcolor(Yellow);                           {Цвет текста полей}
    x := 5; y := 3; {Вывод текстовых полей}       {Начало поля текста s}
    For i := 1 To 5 Do                             {Цикл записей dture}
        Begin
            TextColor(8+i);
            GotoXY(x,y); Write(d[i].s);           {Вывод поля s в окно}
            y := y + 2;                            {Следующее поле}
        End;
    x := 18; y := 3; {Вывод числовых полей} {Начало поля чисел}
    For i := 1 To 5 Do With d[i] Do               {Цикл записей dture}
        Begin
            TextColor(8+i);
            For j := 1 To 5 Do                     {Цикл чисел в записи}
                Begin
                    GotoXY(x,y); Write(a[j]:5:2); {Вывод числа на экран}
                    x := x + 8;                   {Следующее поле на экране}
                End;
            y := y + 2;                             {Следующая строка}
            x := 18;                                {Начало чисел в строке}
        End;
    End;
End;
Procedure New;                                  {Ввод данных с клавиатуры}
Var
    i,j,x,y,z: Integer;
Begin
    x := 5; y := 3;                                {Начало текста}

```

```

For i := 1 To 5 Do With d[i] Do {Цикл записей}
  Begin
    GotoXY(x,y);           {Координата поля текста}
    ReadLn(s);             {Ввод текста}
    z := 18;
    k := False;           {Начало ввода чисел}
    For j := 1 To 5 Do     {Цикл чисел в строке}
      Begin
        GotoXY(z,y);       {Координата числового поля}
        ReadLn(a[j]);      {Ввод числа}
        If a[j] = -1.0 Then {Условие прекращения ввода}
          Begin
            k := True;
            Exit;           {Прекратить ввод}
          End;
        z := z+8;
        If k Then Exit;    {Следующее числовое поле}
      End;
      If k Then Exit;
      y := y+2;           {Следующая строка-запись}
    End;
  End;
Procedure Save;       {Сохранить данные на диске}
  Var
    i: Integer;
    f: File Of dtype;     {Файловая переменная}
  Begin
    Assign(f, 'esp.dat'); Rewrite(f);
    For i := 1 To 5 Do Write(f,d[i]);
    Close(f);
  End;
Procedure Load;      {Загрузить данные с диска}
  Var
    i: Integer;
    f: File Of dtype;     {Файловая переменная}
  Begin
    Assign(f, 'esp.dat'); Reset(f);
    For i := 1 To 5 Do Read(f,d[i]);
    Close(f);
  End;
Procedure Medium;   {Вычисление средних значений}
  Var i,j,n: Integer;
      s1, dis1: Real;

```

```

ch: Char;
Begin
  Frame(x1,y1,x2,y2,True);           {Активная рамка}
  TextColor(Yellow);                 {Цвет текста}
  s1 := 0.0; n := 0;                  {Вычисление среднего значения}
  For i := 1 To 5 Do
    For j := 1 To 5 Do With d[1] Do
      Begin
        s1 := s1 + a[i]; If a[i] <> 0.0 Then Inc(n);
      End;
    s1 := s1 / n;
  OutTextXY(40,210,'Среднее значение');
  GotoXY(25,16); Write (s1:8:2); {Среднеквадратичное отклонение}
  dis1 := 0.0;
  For j:=1 To 5 Do With d[1] Do
    dis1 := dis1 + sqr(a[j] - s1); dis1 := sqrt(dis1) / n;
  OutTextXY(40,240,'Среднеквадратичное отклонение');
  GotoXY(40,18); Write (dis1:8:2);
  ch := ReadKey;                      {Приостановить работу}
  Frame(x1,y1,x2,y2,False);          {Пассивная рамка}
End;
Procedure Pict;
Const
  ndx = 4; ndy = 4; n = 5; m = 2;
Var
  ymax,ymin, rx, ry, xmin, xmax, xl, xp, dx, dy: Real;
  i, j, poi,xn, xk, yn, yk, lx, ly, mx, my: Integer;
  ch: Char;
  st: String;
Begin
  Write_d(d);
  SetColor(DarkGray);                 {Темные линии координат}
  xn := x1+60;xk := x2 - 48;          {Координаты поля графика}
  yn := y1 + 15; yk := y2 - 13;
  SetLineStyle(SolidLn,0, NormWidth); {Тонкая линия}
  Rectangle(xn,yn,xk,yk);             {Очерчивание поля графика}
  Line(xn,yn,xn-5,yn+5); Line (xn,yn,xn+5,yn+5);
  Line(xk,yk,xk-5,yk-5); Line(xk,yk,xk-5,yk+5); {Стрелки}
  ymax := -1e10; ymin := 1e10;
  For i := 1 To 5 Do                  {Нахождение максимума и}
    With d[i] Do                      {минимума всех значений}
      For j := 1 To 5 Do
        Begin
          If a[j]<ymin Then ymin := a[j];

```

```

        If a[j]>ymax Then ymax := a[j];
    End;
For i := 1 To 5 Do {Нанесение графиков разноцветными линиями}
Begin
    xmin := 1;
    xmax := 5;
    SetColor(8+i);
    With d[i] Do
        Begin
            rx := xmax-xmin; ry := ymax-ymin;
            mx := xk-xn; my := yk-yn; xl := xmin-1;
            SetLineStyle(SolidLn,0,ThickWidth);
            For poi := 1 To 5-1 Do
                Begin
                    xl:=xl+1; xp:=xl+1;
                    Line(Round((xl-xmin)*mx/rx)+xn,
                        Round((ymax-a[poi])*my/ry)+yn,
                        Round((xp-xmin)*mx/rx)+xn,
                        Round((ymax-a[poi+1])*my/ry)+yn);
                End;
            End;
        End;
    End;
SetColor(DarkGray); SetLineStyle(SolidLn,0,NormWidth);
lx := (xk-xn) Div ndx;
dx := (xmax-xmin)/ndx; {Нанесение вертикальной сетки}
{и вывод горизонтальных надписей}
For i := 1 To ndx+1 Do
    Begin
        Line(xn+lx*(i-1),yn,xn+lx*(i-1),yk);
        str((xmin+(i-1)*dx):n:m,st);
        OutTextXY(xn+lx*(i-1)-(n-m)*8+4,yk+4,st);
    End;
ly := (yk-yn) Div ndy; dy := (ymax-ymin)/ndy;
For i := 1 To ndy+1 Do {Нанесение горизонтальной сетки и вывод}
{вертикальных надписей}
    Begin
        Line(xn,yn+(i-1)*ly,xk,yn+(i-1)*ly);
        Str((ymax-(i-1)*dy):n:m,st);
        OutTextXY(xn-(n*8+8),yn+(i-1)*ly-4,st);
    End;
Frame(xl, y1, x2, y2, True); {Активная рамка}
ch := ReadKey; {Приостановить работу}
Frame(xl, y1, x2, y2, False); {Пассивная рамка}
End.

```

Текст программы, ориентированной на использование программного модуля n_81_m:

```

Program П_81;
Uses Crt, Graph, П_81;
Const
  x11 = 16; y11 = 8; x12 = 464; y12 = 160;      {Окно 1}
  x21 = 16; y21 = 172; x22 = 464; y22 = 336;   {Окно 2}
  x31 = 488; y31 = 8; x32 = 608; y32 = 160;    {Окно 3}
Type
  Menu_key_set = Set Of #59 .. #67;
Var
  driver, mode, error: Integer;
  ch: Char;
  b: Boolean;
  Menu_key: Menu_key_set;
  d: darray;
Procedure Menu;                                {Меню системы}
Begin
  Frame(x31,y31,x32,y32,True);                 {Рамка активного окна}
  ch := ReadKey;
  If ch=#0 Then ch := ReadKey;                 {Нажать клавишу}
  If ch In Menu_key Then
    Frame (x31,y31,x32,y32,False);             {Пассивная рамка}
  Case ch Of
    #59: Begin                                  {Новые данные}
      Frame(x11,y11,x12,y12,True);             {Окно 1 активно}
      Place(x11, y11, x12,y12);               {Раскраска полей ввода}
      New(d);                                  {Ввод данных}
      Frame(x11,y11,x12,y12, False);          {Окно 1 пассивно}
    End;
    #60: Save(d);                               {Сохранить d на диске}
    #61: Begin                                  {Запись данных с диска}
      Clear_d(d);                               {Чистка d}
      Load(d);                                  {Загрузить данные с диска}
      Write_d(d);                               {Отобразить d в окне}
    End;
    #62: Begin                                  {Очистка полей ввода данных}
      Window1(x11,y11,x12,y12);                {Нарисовать окно 1}
      Window2(x21,y21,x22,y22);                {Нарисовать окно 2}
      Clear_d(d);                               {Чистка массива d}
      Place(x11,y11,x12,y12);                 {Раскраска полей ввода}
    End;
    #63: Medium(x21,y21,x22,y22,d);           {Средние}
    #64: Pict(x21,y21,x22,y22,d);             {Рисование графиков}
    #67: b := False;                           {Выход <F9>}
  End;
End.

```

Пример 82. Разработка информационной системы анализа траектории движения объекта в воздушном пространстве, оперирующей с некоторым набором изменяемых параметров: M , V , α ,

где M — масса объекта;

V — начальная скорость;

α — начальный угол наклона.

Для хранения информации предусмотреть запись данных в файл и чтение данных из файла.

Графическое представление данных реализовать в виде траекторий движения объекта с заданными параметрами.

Меню информационной системы содержит следующие операции:

- меню системы;
- запись в файл;
- чтение из файла;
- изображение графиков;
- очистка окон экрана;
- выход из системы.

Поясним алгоритм определения траектории движения тела в воздушной среде. Основными аргументами для определения траектории являются: скорость тела (U), масса тела (m) и сила, препятствующая движению тела (F). Их взаимодействие можно описать системой дифференциальных уравнений:

$$m \cdot (dU_x/dt) = -F \cdot \cos \alpha; \quad (1)$$

$$m \cdot (dU_y/dt) = -F \cdot \sin \alpha - m \cdot g; \quad (2)$$

$$U = U_x + U_y; \quad (3)$$

$$\operatorname{Tg} \alpha = U_y/U_x; \quad (4)$$

$$F = C \cdot S \cdot R_o \cdot U^2/2. \quad (5)$$

В системе уравнений использованы следующие обозначения:

dU_x/dt — горизонтальная составляющая ускорения тела;

dU_y/dt — вертикальная составляющая ускорения тела;

m — масса тела;

g — ускорение свободного падения;

R_o — плотность воздуха;

α — угол движения тела;

C — коэффициент, учитывающий форму тела;

S — площадь поперечного сечения тела (для шара — это площадь круга).

Решение системы уравнений можно найти следующим образом. Представим весь путь движения тела в виде некоторой ломаной кривой, где точки излома стоят друг от друга на одинаковом расстоянии (dX). Время движения от одной точки излома к другой — dt .

Преобразуем первое из системы дифференциальных уравнений:

$$m \cdot (dU_x/dt) = -F \cdot \cos \alpha;$$

$$m \cdot dU_x = -F \cdot \cos \alpha; \quad (6)$$

Из соотношения становится ясным, что за время dt с учетом горизонтальной составляющей $F \cdot \cos(\alpha)$ изменятся начальная скорость движения тела U_0 и ее горизонтальная составляющая U_{x_0} . Новое значение горизонтальной составляющей (U_{x_1}) станет равно:

$$U_{x_1} = U_{x_0} + dU_x. \quad (7)$$

Аналогичные рассуждения об изменении U_{y_0} позволяют записать:

$$U_{y_1} = U_{y_0} + dU_y \quad (8)$$

Тогда новое значение скорости тела через время dt будет равно:

$$U_1 \sqrt{U_{x_1}^2 + U_{y_1}^2}. \quad (9)$$

Используя новое значение скорости U_1 в пятом соотношении системы дифференциальных уравнений, получим:

$$F_1 = C \cdot R_o/1 \cdot S \cdot U_1^2. \quad (10)$$

Из четвертого уравнения системы получаем новое значение угла α (α_1):

$$\alpha_1 = \operatorname{arctg}(U_{y_1}/U_{x_1}). \quad (11)$$

Теперь определим координаты точки в пространстве, куда переместилось движущееся тело:

$$dX_1 = U_{x_0} \cdot dt \quad (12)$$

$$dY_1 = U_{y_0} \cdot dt \quad (13)$$

Расстояние, на которое переместится тело по горизонтали за все время своего полета, определяется суммой dX_i (равное на первом участке dX_1), за время всех dt :

$$X = \sum dX_i; \quad (14)$$

$$Y = \sum dY_i. \quad (15)$$

Последняя точка моделирования определится координатой $Y=0$, что характеризует момент приземления тела.

Используя предложенную методику определения траектории движения тела, напомним следующие программы.

Текст варианта программного модуля:

```
Unit П_82_m;
Interface
```

```

Uses Crt, Graph;
Type
  typ=Array[1..5,1..3] Of String;
Var
  cir: Pointer;
  i,j,k: Integer;
  buf: Char;
  dat: typ;
Procedure Graf;
Procedure Boll(Var cir: Pointer);
Procedure Fill;
Procedure Wind1;
Procedure Wind2;
Procedure Wind3;
Procedure Menu;
Procedure Prepear;
Procedure Data(Var dat: typ);
Procedure Halp;
Procedure Fulgraph(dat: typ);
Procedure Masht(dat: typ; Var hm,sm: Real);
Procedure Draw(dat: typ; cir: Pointer);
Procedure Save(dat: typ);
Procedure Load(Var dat: typ);
Procedure Select;
Implementation
Procedure Graf;
Var
  driver,mode: Integer;
Begin
  DetectGraph(driver,mode); mode:=1;
  InitGraph(driver,mode,'c:\tp7');
  ClearDevice;
End;
Procedure Boll;
Var size: Integer;
Begin
  Circle(2,2,2);
  size := ImageSize(0,0,4,4); Getmem(cir,size);
  GetImage(0,0,4,4,cir^);
End;
Procedure Fill;
Begin
  SetFillStyle(9,LightGray); Bar(0,0,GetMaxX,GetMaxY);
End;

```

Procedure Wind1;

Begin

```
SetViewPort(0,0,100,100,clipoff);
SetFillStyle(9,DarkGray);
Bar(255,207,GetMaxX-15,215);
Bar(GetMaxX-23,45,GetMaxX-15,215);
SetColor(White); SetLineStyle(0,0,3);
Rectangle(245,25,GetMaxX-25,205);
SetColor(LightGray);
Rectangle(248,28,GetMaxX-28,202);
SetFillStyle(0,DarkGray); Bar(250,30,GetMaxX-30,200);
SetTextStyle(2,0,6); SetColor(LightCyan);
OutTextXY(260,33,'Graphicks'); SetTextStyle(11,0,6);
```

End;

Procedure Wind2;

Begin

```
SetViewPort(0,0,100,100,clipoff);
SetFillStyle(9,DarkGray);
Bar(35,35,200,180);
SetFillStyle(1,Blue);
Bar(28,28,190,170);
```

End;

Procedure Wind3;

Begin

```
SetViewPort(0,0,100,100,clipoff);
SetFillStyle(9,DarkGray);
Bar(100,240,GetMaxX-155,GetMaxY-15);
SetColor(White); SetLineStyle(0,0,3);
Rectangle(95,235,GetMaxX-165,GetMaxY-25);
SetFillStyle(1,7); Bar(97,237,GetMaxX-167,GetMaxY-27);
```

End;

Procedure Menu;

Begin

```
SetColor(15);
SetTextStyle(1,0,1);
OutTextXY(85,30,'Menu');
SetColor(10);
SetTextStyle(11,0,1);
OutTextXY(35,60,'<F1> Help');
OutTextXY(35,70,'<F2> Save');
OutTextXY(35,80,'<F3> Load');
OutTextXY(35,100,'<F5> Data');
OutTextXY(35,110,'<F6> Graphicks');
```



```

    OutTextXY(35,120,'<F8> Clear');
    OutTextXY(35,140,'<F10> Exit');
End;
Procedure Prepear;
Begin
    graf; boll(cir); fill; wind1; wind3; wind2; menu;
End;
Procedure Data;
Var
    i,j,k: Integer;
    m: Real;
    st,stri: String;
    buf: Char;
    lock: Boolean;
Begin
    For i := 1 To 5 Do
        For j := 1 To 3 Do dat[i,j] := "";
    SetViewPort(0,0,1,1,clipoff); SetColor(LightRed);
    Rectangle(95,235,GetMaxX-165,GetMaxY-25);
    SetViewPort(95,250,GetMaxX-30,GetMaxY-30,clipoff);
    SetFillStyle(1,Cyan);
    For i := 0 To 4 Do
        For j := 1 To 3 Do
            Bar(j*65+75,i*15-5,j*65+115,i*15+5);
    SetTextStyle(11,0,4);
    For i := 0 To 4 Do
        Begin
            Str(i+1,st);
            SetViewPort(130,i*15-6+250,135,i*15-1+250,clipoff);
            Case i Of
                0: SetColor(Red);
                2: SetColor(LightGreen);
                4: SetColor(LightCyan);
                1: SetColor(Yellow);
                3: SetColor(LightBlue);
            End;
            OutText(st); OutText('object'); SetColor(Black);
            For j := 1 To 3 Do
                Begin
                    SetFillStyle(1,LightCyan);
                    SetViewPort(j*65+135+35,i*15-5+250,j*65+190+35,i*15+
+5+250,clipon);
                    Bar(0,0,40,20);

```

```

SetViewPort(j*65+135+35,i*15-5+250,j*65+190+35,i*15+
+5+250,clipon);
While Not KeyPressed Do; lock := True;
While lock Do
  Begin
    buf := ReadKey; SetColor(LightBlue); OutText(buf);
    If buf=#13 Then lock := False;
    If lock Then dat[i+1,j]:= dat[i+1,j]+buf;
    If dat[i+1,j]=" Then dat[i+1,j] := '000';
    buf := #0;
  End;
SetFillStyle(1,LightGray);
SetViewPort(j*65+135+35,i*15-5+250,j*65+190+35,i*15+5+
+250,clipon);
Bar(0,0,40,20);
SetViewPort(j*65+135+35,i*15-5+250,j*65+190+35,i*15+
+5+250,clipon);
SetColor(0);
Val(dat[i+1,j],m,k); Str(Round(m),stri); OutText(stri);
End;
End;
SetViewPort(0,0,1,1,clipoff);
SetColor(Cyan);
Rectangle(95,235,GetMaxX-165,GetMaxY-25);
End;
Procedure Halp;
Var
  F: Text;
  st: Array[1..10] Of String;
  i,x: Integer;
Begin
  SetViewPort(0,0,GetMaxX,GetMaxY,clipon);
  ClearDevice;
  SetFillStyle(9,LightGray);
  Bar(0,0,GetMaxX,GetMaxY);
  SetFillStyle(9,DarkGray);
  Bar(45,GetMaxY-35,GetMaxX-25,GetMaxY-25);
  Bar(GetMaxX-35,45,GetMaxX-25,GetMaxY-25);
  SetColor(White); SetLineStyle(0,0,3);
  Rectangle(35,35,GetMaxX-35,GetMaxY-35);
  SetColor(LightCyan);
  Rectangle(38,38,GetMaxX-38,GetMaxY-38);
  SetFillStyle(1,Blue); SetColor(Yellow);
  Bar(40,40,GetMaxX-40,GetMaxY-40);

```

310

```

SetColor(LightCyan); SetTextStyle(1,0,3);
OutTextXY(280,40,'Halp'); SetTextStyle(1,0,2);
OutTextXY(50,75,' This program allow to make a graphicks models');
OutTextXY(50,100,' of flying subjects with constant mass. ');
OutTextXY(50,125,' You can enter results of five shooting in field' );
OutTextXY(50,150,' «data» and see results graphicks output in');
OutTextXY(50,175,' field «graphicks».);
While Not KeyPressed Do
  While KeyPressed Do buf := ReadKey;
  Prepear;
End;
Procedure Fulgraph;
Var
  i,j,k,ix,iy,t,color: Integer;
  x,y,dy,dx: Real;
  lock: Boolean;
  st: String;
  buf: Char;
Begin
  ClearDevice;
  SetViewPort(0,0,600,250,clipoff); SetFillStyle(1,LightGray);
  Bar(2,2,GetMaxX-2,31); SetTextStyle(1,0,2); SetColor(Blue);
  OutTextXY(220,5,'Full information. ');
  SetBkColor(0); SetColor(LightCyan); SetLineStyle(0,0,3);
  Rectangle(1,1,GetMaxX-1,GetMaxY-1);
  SetColor(LightCyan); Rectangle(1,32,GetMaxX-1,GetMaxY-1);
  SetViewPort(4,34,600,250,clipoff);
  SetTextStyle(2,0,5); SetColor(White);
  OutTextXY(10,5,'Graphicks. ');
  SetTextStyle(1,0,2); SetLineStyle(0,0,1);
  Line(5,5,5,170); Line(5,5,2,15); Line(5,5,8,15);
  Line(1,165,330,165); Line(330,165,320,162);
  Line(330,165,320,168);
  For i:=1 To 5 Do
    Begin
      y:=0; x:=5;
      Val(dat[i,1],ix,k); Val(dat[i,2],iy,k);
      dx:=ix/100; dy:=iy/100;
      SetLineStyle(0,1,1);
      SetColor(0); LineTo(5,165);
      lock:=True;
      While Lock Do
        Begin

```

```

x := x+dx; y := y-dy; dy := dy-0.1; dx := dx-0.003;
Case i Of
  1: color:=LightRed;
  2: color:=Yellow;
  3: color:=LightGreen;
  4: color:=LightBlue;
  5: color:=LightCyan;
End;
SetColor(color); LineTo(Round(x),Round(y)+165);
PutPixel(Round(x),Round(y)+165,color);
If y>0 Then lock:=False;
End;
SetColor(White); SetLineStyle(0,0,1); Line(5,5,5,170);
Line(1,165,330,165);
End;
SetTextStyle(2,0,5); OutTextXY(7,187,'Data:');
For i := 0 To 4 Do
  Begin
    Case i Of
      0: SetColor(Red);
      1: SetColor(Yellow);
      2: SetColor(LightGreen);
      3: SetColor(LightBlue);
      4: SetColor(LightCyan);
    End;
    SetTextStyle(11,0,4);
    OutTextXY(270,i*15-3+215,'Color ');
    SetFillStyle(1,Cyan);
  End;
SetTextStyle(11,0,4); SetColor(LightCyan);
For i := 0 To 4 Do
  Begin
    str(i+1,st);
    SetViewPort(10,i*15-6+250,55,i*15-1+250,clipoff);
    OutText(st); OutText(' object');
    For j:=1 To 3 Do
      Begin
        SetViewPort(j*45+55,i*15-
          5+250,j*65+115+35,i*15+5+250,clipon);
        OutText(dat[i+1,j]);
      End;
    End;
  End;
While Not KeyPressed Do
  Prepear;

```

```

End;
Procedure Masht;
Var
  hx, hy, x, y, dx, dy, O, Ux, Uy, dUx, dUy, dt, F, Cx, S, r, h: Real;
  gx, gy, count, m, u, ci, color, speed: Integer;
  ch: Char;
Begin
  sm := 0; hm := 0;
  For i := 1 To 5 Do
    Begin
      Val(dat[i,1],m,k); Val(dat[i,2],U,k); Val(dat[i,3],O,k);
      x := 0; y := 0; gx := 0; gy := 0; {Set zero point}
      Cx := 0.8; S := 25E-6; r := 100; dt := 2;
      Ux := U*cos(O); Uy:=U*sin(O);
      If (m<>0) And (u<>0) And (o<>90) Then
        Begin
          O := o*pi/180;
          speed := u; count := 0;
          While (y>=0) Do
            Begin
              F:=cx*r*u*u/2*s;
              dUx:=-F*Cos(O)/m*dt;
              dUy:=((-F*Sin(O))-m*9.81)/m*dt;
              Ux:=abs(Ux+dUx); Uy:=Uy+dUy;
              dx:=Ux*dt; dy:=Uy*dt;
              x:=x+dx; y:=y+dy;
              O:=ArcTan(Uy/Ux); Inc(count);
              If y>hm Then hm:=y;
            End;
            If sm<x Then sm:=x;
          End;
        End;
      End;
    End;
  End;
Procedure Draw;
Var
  hx, hy, x, y, dx, dy, O, Ux, Uy, dUx, dUy, dt, F, Cx, S, r, hm, sm: Real;
  gx, gy, count, m, u, ci, color, speed: Integer;
  ch: Char;
  hmst, smst: String;
Begin
  masht(dat, hm, sm);
  For ci := 1 To 5 Do
    Begin

```

```

x :=0; y :=0; gx :=0; gy :=0; {Set zero point}
Cx := 0.8; S := 25E-6; r := 100; dt := 0.1;
i := ci;
Val(dat[i,1],m,k); Val(dat[i,2],U,k); Val(dat[i,3],O,k);
If (m<>0) And (u<>0) And (o<>90) Then
  Begin
    O:=o*pi/180;
    speed:=u; count:=0;
    SetViewPort(250,30,GetMaxX-30,200,clipon);
    SetColor(White); SetLineStyle(0,0,1);
    Line(5,5,5,170); Line(5,5,2,15); Line(5,5,8,15);
    Line(1,165,330,165); Line(330,165,320,162);
    Line(330,165,320,168);
    SetViewPort(0,0,GetMaxX,200,clipoff);
    U x:= U*Cos(O); Uy := U*Sin(O);
    MoveTo(255,195);
    While (y>=0) And (gx<1000) Do
      Begin
        PutImage(gx-2,gy-2,cir^,1);
        Case ci Of
          1 : color := LightRed;
          2: color := Yellow;
          3: color := LightGreen;
          4: color := LightBlue;
          5: color := LightCyan;
        End;
        SetColor(color); PutImage(gx-2,gy-2,cir^,1);
        gx := Round((200/Round(sm))*x) +255;
        gy := -Round((130/Round(hm))*y) +195;
        LineTo(gx,gy);
        F := cx*r*u*u/2*s; dUx:=-F*Cos(O)/m*dt;
        dUy :=((-F*Sin(O))-m*9.81)/m*dt;
        Ux := Abs(Ux+dUx); Uy:=Uy+dUy;
        dx := Ux*dt; dy := Uy*dt;
        x := x+dx; y := y+dy;
        O := ArcTan(Uy/Ux); Inc(count);
      End;
    End;
  End;
End;
SetViewPort(0,0,1,1,clipoff); SetTextStyle(11,0,1);
str(Round(hm),hmst); str(Round(sm),smst);
OutTextXY(450,30,hmst); OutTextXY(450,45,smst);
SetLineStyle(0,0,3); SetColor(LightGray);

```

```

    Rectangle(247,27,GetMaxX-27,203);
    While KeyPressed Do buf := ReadKey;
    buf := #0;
End;
Procedure Save;
Var
    fl: File Of String;
Begin
    Assign(fl,'parametr.dat'); Rewrite(fl);
    For i := 1 To 5 Do
        For j := 1 To 3 Do Write(fl,dat[i,j]);
    Close(fl);
    SetFillStyle(1,0); Bar(210,260,410,300);
    SetFillStyle(1,LightGreen); Bar(200,250,400,290);
    SetColor(yellow); SetLineStyle(0,0,3);
    Rectangle(203,253,397,287);
    SetTextStyle(11,0,1); SetColor(Blue);
    OutTextXY(230,270,'Saving parameters');
    Delay(400);
    Wind3;
End;
Procedure Load;
Var
    f: File Of String;
    st,stri: String;
    m: Real;
    k: Integer;
Begin
    Assign(f,'parametr.dat'); Reset(f);
    For i:=1 To 5 Do
        For j:=1 To 3 Do Read(f,dat[i,j]);
    Close(f);
    SetFillStyle(1,0); Bar(210,260,410,300);
    SetFillStyle(1,Red); Bar(200,250,400,290)
    SetColor(Yellow); SetLineStyle(0,0,3);
    Rectangle(203,253,397,287);
    SetTextStyle(11,0,1); SetColor(Blue);
    OutTextXY(230,270,'Loading secesfull');
    Delay(400); Wind3; SetViewPort(0,0,1,1,clipoff);
    SetColor(LightRed);
    Rectangle(95,235,GetMaxX-165,GetMaxY-25);
    SetViewPort(35,250,GetMaxX-30,GetMaxY-30,clipoff);
    SetTextStyle(11,0,4); SetColor(Black);

```

```

For i := 0 To 4 Do
  Begin
    str(i+1,st);
    SetViewPort(130,i*15-6+250,135,i*15-1+250,clipoff);
    Case i Of
      0: SetColor(Red);
      1: SetColor(Yellow);
      2: SetColor(LightGreen);
      3: SetColor(LightBlue);
      4: SetColor(LightCyan);
    End;
    OutText(st); OutText('object'); SetColor(Black);
    For j := 1 To 3 Do
      Begin
        SetFillStyle(1,LightCyan);
        SetViewPort(j*65+135+35,i*15-5+250,j*65+190+35,i*15+5+
+250,clipon);
        Bar(0,0,40,20);
        SetViewPort(j*65+135+35,i*15-
5+250,j*65+190+35,i*15+5+250,clipon);
        SetFillStyle(1,LightGray);
        SetViewPort(j*65+135+35,i*15-5+250,j*65+190+35,i*15+5+
+250,clipon);
        Bar(0,0,40,20);
        SetViewPort(j*65+135+35,i*15-5+250,j*65+190+35,i*15+5+
+250,clipon);
        SetColor(0);
        Val(dat[i+1,j],m,k); Str(round(m),stri); Outtext(stri);
      End;
    End;
    SetViewPort(0,0,1,1,clipoff); SetColor(Cyan);
    Rectangle(95,235,GetMaxX-165,GetMaxY-25);
  End;
Procedure Select;
  Var
    buf: Char;
    lock,exit: Boolean;
  Begin
    While lock Do
      Begin
        SetColor(LightRed); Rectangle(28,28,190,170);
        buf:=#0; buf:=ReadKey;
        SetColor(LightCyan); Rectangle(28,28,190,170);
        Case buf Of
          #059: halp;
          #060: save(dat);
          #061: load(dat);
          #063: data(dat);
          #064: draw(dat,cir);
          #066: Begin
            wind3; wind1;
            For i := 1 To 5 Do
              For j := 1 To 3 Do dat[i,j] := '000';
            End;
          #27: lock := False;
          #068: lock := False;
        End;
      End;
    End;
  End.

```

Текст основной программы можно представить в следующем виде:

```

Program П_82;
Uses n_82m;
Begin
  Prepear;
  Select;
End.

```


ГЛАВА 4. ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

4.1. Работа в программе Norton Commander

Цель работы: отработать навыки работы с среде программы-оболочки Norton Commander и овладеть методами использования текстового редактора (NE) Norton Commander.

1. Установка программы-оболочки NC.

1. Наберите на командной строке ps.exe и запустите программу.
2. На правом (или левом) поле экрана посмотрите содержание диска C: (или E:), а на соседнем — содержание диска D:.
3. Поменяйте местами правую и левую панели.
4. Отключите панель с каталогом C: (или E:).
5. Вызовите на свободное место панель с рабочим диском (например, D:).
6. Создайте на рабочем диске каталог с Вашим именем и войдите в него.
7. Создайте в каталоге новый файл (например, Work.txt) и запишите в нем тему выполняемой работы.
8. Перепишите с корневого каталога в Ваш каталог файл Work.txt.
9. Переименуйте файл Work.txt в Вашем каталоге, присвоив ему новое имя с расширением .bac.
10. Просмотрите содержимое файлаbac.
11. Добавьте в файл сведения об используемых файлах и каталогах.
12. Перепишите файлbac из подкаталога в корневой каталогдис-KaD:.
13. Удалите файлы из корневого каталога.
14. Удалите файлtxt из Вашего каталога, а затем сам каталог.
15. Перейдите с диска D: на диск C: или (E:).
16. Выйдите из программы Norton Commander.

2. Запись информации в редакторе NC.

1. Вызовите текстовый редактор NC с помощью меню NC (клавиша <F2>).
2. Выберите удобный цвет экрана (<Shift>+<F9>).
3. Запишите форму для расписания учебных занятий на конкретный день недели: в поле 1 запишите время начала занятий, а в строки -номер аудитории и название дисциплины. Например:
8.00
9.50
11.40
13.30.
4. Используя знаки псевдографики, расчертите поле расписания:

Время	Аудитория	Дисциплина
1	2	3

Для нанесения линий следует одновременно нажимать клавишу <Alt> и одну из цифр цифрового блока клавиатуры.

5. Выделите текст с помощью клавиши <F3>.
6. Поместите выделенный участок текста в карман: <Ctrl>+<F3>.
7. Восстановите текст из кармана: <Ctrl>+<F4>.
8. Переведите курсор на нижнюю строку и повторите восстановление текста 5 раз.
9. Отмените выделение текста с помощью клавиши <F4>.
10. Замените слово «понедельник» во 2—5 фрагментах на соответствующий день недели.
11. Запишите в поле 2 аудитории, а в поле 3 — учебные курсы на каждый день недели.
12. Запишите текст из окна 1 в файл ord.txt. 13. Используя режим поиска и замены по образцу, замените комбинацию символов «математика» на «mathematics» и перепишите текст в файл ord_a.txt.
14. Проведите разметку страниц текста с интервалом 1.1.
15. Сохраните полученный текст.

4.2. Работа в графической среде Paint

В качестве содержания самостоятельной работы станет изображение товарного знака на изделия фирмы. Такая фирма может производить компьютеры, радиолокационное оборудование, литье, холодильные установки, двигатели, спортивные товары и т. д.

Текст задания ориентирован на подготовку товарного знака автомобильной фирмы, однако следует придерживаться его последовательности при выполнении товарных знаков других фирм.

1. Используя графический редактор Paint, изобразите подробный контур профиля автомобиля.
2. Раскрасьте рисунок, используя различные краски для кузова, колес и аксессуаров автомобиля.
3. Поместите на рисунке имя фирмы и знаки: защиты от копирования и регистрации товарного знака.
4. Уменьшите рисунок объекта на 30%.
5. Скопируйте рисунок в файл и буфер обмена.
6. Перейдите в текстовый редактор Wndows и создайте текстовый файл с названием лабораторной работы и заданием к ней.
7. Перенесите в этот файл подготовленный рисунок и закрепите его в тексте заявки на регистрацию товарного знака.
8. Сохраните файл с текстом заявки.

4.3. Основные команды интегрированной среды Турбо Паскаля

Цель работы: отработать навыки отладки и выполнения программ в Турбо Паскале.

1. Перейдите на рабочий диск компьютера (например, D:).
2. Вызовите Турбо Паскаль, набрав на командной строке: Turbo.
3. В активное окно запросите новый файл, определите его имя: нажмите клавишу <F2>, подведите курсор к началу строки и напишите новое имя (например, Bril.pas), нажмите клавишу <Enter>.
4. Перепишите текст следующей программы:

Program 04_1;

(Вывести результат выполнения операций над двумя целыми числами)

```
Var
n,m : Integer;
Begin
  Write('Введите два целых числа: '); ReadLn(n,m);
  WriteLn('n + m = ',n + m);
  WriteLn('n*m = ',n*m,);
  WriteLn('n/m = ',n/m,' ',n/m:5:2);
  WriteLn('n Div m = ',n Div m);
  WriteLn('n Mod n = ',n Mod m);
  WriteLn;
  WriteLn('Not n = ',Not n);
  WriteLn('Not n = ',Not n);
  WriteLn('n And m = ',n And m);
  WriteLn('n Or m = ',n Or m);
  WriteLn('n Xor m = ',n Xor m);
  WriteLn;
  WriteLn('n Shl m = ',n Shl m);
  WriteLn('n Shr m = ',n Shr m);
End.
```

5. Проведите компиляцию программы, используя опцию **Compile|Compile** или клавишу <F9>.
6. Исправьте обнаруженные ошибки, после исправления снова выполните п. 5. После окончания успешной компиляции нажмите любую клавишу и переходите к выполнению п. 7.
7. Выполните пошаговое исполнение программы. Для этого выберите режим **Run|Step over** или постоянно нажимайте клавишу <F7>, следите с помощью контрастной полосы за процессом выполнения программы.
8. Повторите процесс выполнения программы в автоматическом режиме, одновременно нажав клавиши <Ctrl>+<F9>.

9. Введите текст одной из следующих программ и самостоятельно добейтесь успешного запуска программы.

```
Program 04_2;
Uses Graph, Crt;
Var
  d, r, e: Integer;
Begin
  d := Detect;
  Initgraph(d, r, "");
  e := GraphResult;
  If e <> grOk Then
    WriteLn(GraphErrorMsg(e))
  Else
    Begin
      d := GetMaxX Div 4;
      r := GetMaxY Div 4;
      Rectangle(d,r,3*d,3*r);
      SetViewPort(d+1,r+1,3*d-1,3*r-1,ClipOn);
      Repeat
        SetFillStyle(Random(12),
          Random(succ(GetMaxColor)));
        Bar(Random(GetMaxX),
          Random(GetMaxY),
          Random(GetMaxX),
          Random(GetMaxY));
      Until KeyPressed;
      If ReadKey=#0 Then d := Ord(ReadKey);
      CloseGraph;
    End;
End.
```

```
Program 04_3;
Uses Crt, Graph;
Var
  d,r,e,k,j,x,y: Integer;
Begin
  d := Detect;
  InitGraph(d,r,"");
  e := GraphResult;
  If e <> grOk Then
    WriteLn(GraphErrorMsg(e))
  Else
    Begin
```

```

x := GetMaxX Div 6;
y := GetMaxY Div 5;
For j := 0 To 2 Do
  For k := 0 To 3 Do
    Begin
      Rectangle((k+1)*x,(j+1)*y,(k+2)*x,(j+2)*y);
      SetfillStyle(k+j*4,j+1);
      Bar((k+1)*x+1,(j+1)*y+1,(k+2)*x-1,(j+2)*y-1);
    End;
  If ReadKey = #0 Then k := Ord(ReadKey);
  CloseGraph;
End;
End.

```

```

Program 04_4;
Uses Crt,Graph;
Var
  x1,y1,x2,y2,Err: Integer;
Begin
  x1 := Detect;
  InitGraph(x1,x2,"");
  Err := GraphResult;
  If Err <>grOk Then
    WriteLn(GraphErrorMsg(Err))
  Else
    Begin
      x1 := GetMaxX Div 4;
      y1 := GetMaxY Div 4;
      x2 := 3*x1;
      y2 := 3*y1;
      Rectangle(x1,y1,x2,y2);
      Setviewport(x1+1,y1+1,x2-1,y2-1,ClipOn);
      Repeat
        Circle(Random(GetMaxX),
          RanDom(GetMaxX),
          RanDom(GetMaxX Div 5));
        Until KeyPressed;
      Clearviewport;
      OutTextXY(0,0,'Press <Enter> . . . ');
      ReadLn;
      CloseGraph;
    End;
  End.

```

4.4. Оконный интерфейс в программах пользователя

1. Изобразите на экране три окна: для ввода данных, вывода данных, записи варианта задания. Покажите тень от окон. Введите 5 вещественных чисел в поле первого окна, указывая их имена; выведите эти числа на поле второго окна, указывая имя и значение числа, соблюдая постоянную форму вывода.
2. Введите данные 5 чисел, сопровождая ввод звуковым сигналом, выведите данные, сопровождая их вывод другим звуковым сигналом.
3. Изобразите на экране три окна: для ввода данных, вывода данных, записи варианта задания. Покажите тень от окон. Введите 6 вещественных чисел в поле первого окна, указывая их имена; выведите эти числа на поле второго окна, указывая имя и значение числа, соблюдая постоянную форму вывода.
4. Введите данные 6 чисел, сопровождая ввод звуковым сигналом, выведите данные, сопровождая их вывод другим звуковым сигналом и изменением цвета тени соответствующего окна.
5. Изобразите на экране три окна: для ввода данных, вывода данных, записи варианта задания. Покажите тень от окон. Введите 8 целых чисел в поле первого окна, указывая их имена; выведите эти числа на поле второго окна, указывая имя и значение числа, соблюдая постоянную форму вывода.
6. Введите данные 8 чисел, сопровождая ввод звуковым сигналом, выведите данные, сопровождая их вывод другим звуковым сигналом.

4.5. Программы линейной структуры

1. Введите действительное число, содержащее 4 знака до запятой и 5 — после. Определите сумму знаков целой и дробной частей числа. Найдите произведение, десятичные логарифмы корня 7-й степени этих сумм.

2. Оформите оконный интерфейс для выполнения задачи 1, предусмотрев окно ввода числа, окно выдачи информации и окно-указатель варианта выполняемой работы.

3. Выдайте данные по выплате месячных процентов (m) по вкладам и общей прибыли ($12mn$). В качестве исходных данных возьмите: объем вклада (s), срок выплаты (n), процент (p). Месячная выплата вычисляется по формуле:

$$sr(1+r)^n m / (12((1+r)^n - 1)),$$

где $r = p/100$.

4. Оформите оконный интерфейс для выполнения предыдущей задачи, предусмотрев окно ввода данных, окно выдачи информации и окно-указатель варианта выполняемой работы.

5. Введите символ с клавиатуры. Найдите сумму, произведение, средние арифметическое и геометрическое цифр его кода.

6. Оформите оконный интерфейс для выполнения предыдущей задачи, предусмотрев окно ввода данных, окно выдачи информации и окно-указатель варианта выполняемой работы.

4.6. Программы разветвляющейся структуры

1.

$$y = \begin{cases} 1 + \sin(x), & \text{если } x < 0,5 \\ 0,5(1 + \cos x), & \text{если } 0,5 \leq x \leq 1,5 \\ 1/(1+x), & \text{если } x > 1,5 \end{cases}$$

2. Напишите программу для предыдущей задачи, используя оператор выбора.

3.

$$S = \begin{cases} 0,35t + \sin t + 0,1, & \text{если } t < 0,4 \\ t/(t+0,1), & \text{если } t > 0,4 \\ 3/(t + \cos t), & \text{если } t = 0,4 \end{cases}$$

4. Напишите программу для предыдущей задачи, используя оператор выбора.

5.

$$P = \begin{cases} 10,4x + \arccos x + 9x, & \text{если } 0,6 \leq x < 0,8 \\ 0, & \text{если } x < 0,6 \\ \ln x + \operatorname{tg} x, & \text{если } x \geq 0,8 \end{cases}$$

6. Напишите программу для предыдущей задачи, используя оператор выбора.

4.7. Организация циклов

1. Вычислите сумму положительных и сумму отрицательных элементов массива A задаваемого размера.

2. Найдите в массиве A первый элемент, кратный 5.

3. Организуйте оконный интерфейс для решения двух предыдущих задач.

4. Определите максимальный по модулю элемент массива A задаваемого размера среди элементов, имеющих четный индекс.

5. Разделите массив A на два массива P и Q так, чтобы элементами P были положительные элементы L , а элементами Q — отрицательные элементы L .

6. Организуйте оконный интерфейс для решения двух предыдущих задач.

7. Вычислите сумму максимального и минимального элементов массива C задаваемого размера.

8. Перенесите в массив D все элементы, превышающие C_{\min} по абсолютной величине. Выведите исходный массив и результат.

9. Организуйте оконный интерфейс для решения двух предыдущих задач.

4.8. Обработка многомерных массивов данных

1. В матрице $G(L,M)$, где $L \leq 10$, $M \leq 9$, найдите столбец с минимальным количеством положительных элементов. Напечатайте исходную матрицу в виде матрицы и найденный номер столбца.

2. Организуйте оконный интерфейс для отображения входной информации, результат вычисления и указания варианта лабораторной работы.

3. Из положительных элементов целочисленной матрицы $B(L,K)$, где $L \leq 12$, $K \leq 10$, сформируйте вектор G . В полученном векторе G поменяйте местами минимальный элемент с первым по порядку элементом. Напечатайте исходную матрицу в виде матрицы и вектор G до и после преобразования.

4. Организуйте оконный интерфейс для отображения входной информации, результаты вычисления и указания варианта лабораторной работы.

5. Задана матрица $A(L,K)$, где $L \leq 12$, $K \leq 12$. В каждом столбце поменяйте местами максимальный элемент с диагональным. Распечатайте в виде матриц исходную и преобразованную матрицы A .

6. Организуйте оконный интерфейс для отображения входной информации, результаты вычисления и указания варианта лабораторной работы.

4.9. Обработка символьных данных

1. Введите строку из нескольких слов. Определите количество слов в строке, начинающихся с буквы «а» и заканчивающихся буквой «с».

2. Введите строку произвольной длины. Определите, сколько раз и какие согласные алфавита встречаются в этой строке. Результаты запомните в массиве и выведите их на экран.

3. Введите строку из нескольких слов. Определите количество слов в строке, оканчивающихся на гласную.

4. Введите строку из нескольких слов. Сформируйте массив из букв этих слов, расставленных по алфавиту.

5. Введите строку из нескольких слов. Определите количество слов в строке, содержащих сочетание букв qwe .

6. Введите строку произвольной длины. Определите, сколько раз и какие согласные алфавита встречаются в этой строке. Результаты запомните в массиве и выведите их на экран.

4.10. Обработка строковых данных

1. Введите несколько слов в строке, определите состав этого текста: число слов, число букв в каждом слове и соотношение гласных и согласных в тексте.

2. Создайте экранный интерфейс для оформления процессов ввода и вывода информации в программе предыдущего задания.

3. Введите несколько слов в строке, подчеркните отдельно каждое слово в строке, используя знак «-» на следующей строке, а также определите, сколько раз повторяются используемые в тексте буквы.

4. Создайте экранный интерфейс для оформления процессов ввода и вывода информации в программе предыдущего задания.

5. Введите строку, состоящую из нескольких слов, и проведите анализ введенного текста: число слов в строке и знаков в каждом слове, количество гласных и их распределение (число вхождений).

6. Создайте экранный интерфейс для оформления процессов ввода и вывода информации в программе предыдущего задания.

4.11. Обработка данных типа «Запись»

1. Создайте базу данных из задаваемого числа записей, отражающих результаты сессии: фамилия, группа, три предмета, три оценки. По запросу об отличниках сессии выдайте список по алфавиту фамилий студентов по каждой группе, сдавших все экзамены на «отлично».

2. Разработайте экранный интерфейс к предыдущему заданию и возможность многократного обращения к базе данных.

3. Создайте базу записей об учебной литературе: наименование, автор, учебная дисциплина, число экземпляров, цена. По запросу о наличии учебной литературы по заданной дисциплине выдайте список изданий (по алфавиту), указывая число их экземпляров и общую стоимость издания.

4. Разработайте экранный интерфейс к предыдущему заданию и возможность многократного обращения к базе данных.

5. Создайте базу данных из задаваемого числа записей, отражающих результаты сессии: фамилия, группа, три предмета, три оценки. По запросу о лучшей группе выдайте перечень тех групп, которые имеют наивысший суммарный средний балл по всем экзаменам.

6. Разработайте экранный интерфейс к предыдущему заданию и возможность многократного обращения к базе данных.

4.12. Типизированные и текстовые файлы

1. Создайте файл произвольного имени, в который запишите целые числа. Организуйте возможность дополнения файла, переписи в другой файл четных чисел и в третий — нечетных, сохраняя порядок следования чисел. Предусмотрите возможность удаления данных из базы данных, а также самого файла.

2. Для предыдущего задания организуйте окно записи информации, окно выдачи данных и поле меню программы.

3. Организуйте файл произвольного имени с расширением .db, в который запишите различные знаки. Определите, являются ли два первых символа файла цифрами. Если да, то установите, является ли число, образованное этими цифрами, четным. Предусмотрите возможность удаления данных из базы данных, а также самого файла.

4. Для предыдущего задания организуйте окно записи информации, окно выдачи данных и поле меню программы.

5. Организуйте файл произвольного имени с расширением .db, в котором создайте базу данных из различных символов. Добавьте литеру «С» в начало и конец файла. Удалите из файла все символы «+» и «-». Предусмотрите возможность удаления данных из базы данных, а также самого файла.

6. Для предыдущего задания организуйте окно записи информации, окно выдачи данных и поле меню программы.

4.13. Процедуры и функции

1. Разработайте программу, позволяющую создавать базы данных соревнований по лыжному двоеборью (бег + прыжки): спортивное общество, фамилия спортсмена, вид упражнения, результат (в баллах); по запросу о призерах соревнования по отдельным видам выдать на экран: список призеров с указанием фамилии, результата и спортивного общества; определять результат победителя соревнований. Программу подготовьте, используя обращение к подпрограммам.

2. Разработайте экранный интерфейс к предыдущему заданию и возможность многократного обращения к базе данных.

3. Разработайте программу, позволяющую создавать базы данных из задаваемого числа записей, отражающих результаты сессии: фамилия, группа, три предмета, три оценки; определять отличников и неуспевающих студентов. Программу подготовьте, используя обращение к подпрограммам.

4. Разработайте экранный интерфейс к предыдущему заданию и возможность многократного обращения к меню программы.

5. Разработайте программу, позволяющую создавать базы записей об учебной литературе: наименование, автор, учебная дисциплина, число экземпляров; определять по запросу о наличии учебной литературы по заданной дисциплине список изданий; выдавать число экземпляров определенного издания.

Программу подготовьте, используя обращение к подпрограммам.

6. Разработайте экранный интерфейс к предыдущему заданию и возможность многократного обращения к базе данных.

4.14. Обработка массивов данных

1. Введите элементы целочисленного массива $C(9)$. Нечетные по значению элементы этого массива перепишите в новый массив подряд без пропусков. Покажите на экране исходный и полученный массивы.

2. Определите количество нулевых элементов в каждом столбце матрицы $B(5,4)$. Результаты запомните и выведите на экран.

3. Введите элементы целочисленного массива $A(8)$. Найдите и выдайте на экран среднее геометрическое элементов этого массива, без остатка делящихся на 4. Если в массиве нет подобных элементов — выдайте соответствующий текст.

4. Сформируйте массив из той строки матрицы $C(4,4)$, в которой находится минимальный элемент главной диагонали матрицы. Выведите на экран сформированный массив.

5. Введите целочисленный массив $E(9)$. Переставьте местами максимальный и нулевой элементы массива. Выведите на экран номера и значения максимального и нулевого элементов, а также полученный массив.

6. Введите матрицу $A(5,6)$, определите номера минимальных по модулю элементов всех строк матрицы и сформируйте новый массив из номеров столбцов, в которых располагаются эти элементы. Полученный массив выдайте на экран.

4.15. Строки и записи

1. Напишите программу определения числа каждой из букв строки.

2. Составьте массив записей, содержащих сведения о жильцах дома: фамилия, номер дома, номер квартиры, площадь квартиры. По запросу «Общая жилая площадь дома N?» введите соответствующий номер дома и выдайте на экран информацию.

3. Напишите программу определения числа цифровых знаков в строке.

4. Составьте массив записей, содержащих сведения о рынке изделий: название, стандарт и цена изделия. Обеспечьте ответ на вопрос о числе и сумме изделий заданного названия.

5. Напишите программу расстановки в спецификации изделий в алфавитном порядке.

6. Составьте массив записей, содержащих сведения о результатах эксперимента: номер, температура, давление, влажность, результат. Обеспечьте ответ на вопрос об условиях экспериментов, имеющих высокий и низкий результат.

ГЛАВА 5. ИНФОРМАЦИОННЫЙ СЕРВИС ИНТЕРНЕТА

5.1. Структура глобальной сети

Интернет (Internet) является глобальной информационной сетью, объединяющей множество сетей (индивидуальных, корпоративных, региональных и т. п.), для свободной передачи информации любому пользователю сети. Поток документов, войдя в Интернет через индивидуальную сеть, проходит через ряд корпоративных и региональных сетей и достигает адресата.

5.1.1. Информационные службы сети

Интернет предоставляет доступ к набору информационных служб (сервисов), основными среди которых являются:

- ♦ *электронная почта* (e-mail) — служба, обеспечивающая передачу письма (сообщения) на любой компьютер, находящийся в сети. Для доставки электронных сообщений используется метод последовательной передачи от узла к узлу сети, причем маршрут следования скрыт от получателя. Если сообщение не достигает адресата в установленное время или адресат указан неверно, то оно возвращается отправителю;
- ♦ *список рассылки* — служба, обеспечивающая рассылку сообщений по заранее установленным адресам. Обычно список рассылки хранится на отдельном сервере, куда поступают сообщения и откуда они передаются адресатам по списку. Этот вид службы позволяет организовать обсуждение отдельных вопросов, не предназначенных для широкого круга лиц;

- ◆ **система телеконференций** — это служба сети, которая реализует хранение сообщений на выделенных серверах (News-серверы) и обеспечивает доступ к этим сообщениям с компьютеров подписчиков данной телеконференции;
- ◆ **протокол передачи файлов** (File Transfer Protocol, FTP) — служба которая используется для доступа к файловым архивам, расположенным на серверах, входящих в различные глобальные сети, включая и Интернет. Она позволяет просмотреть файлы, доступные на одном из FTP-компьютеров, и скопировать необходимые;
- ◆ **Gopher** — информационная система, обеспечивающая интерактивный интерфейс для организации поиска информации, основанный на системе меню. Пункты таких меню могут ссылаться на информационные ресурсы, расположенные на различных Gopher-серверах;
- ◆ **World Wide Web** — служба доступа к информации с помощью гипертекстовых ссылок. Ее суть заключается в том, что в текст документа включают ссылки на другие документы, которые могут располагаться и на других серверах, что позволяет объединить в одном документе информацию, представленную в различных формах (текст, графика, звук и т. п.) и распределенную по множеству компьютеров. Это создает условия для обращения к мультимедийным документам, организованным в форме различных учебников, монографий, справочников и т. п. WWW обеспечивает доступ практически ко всем другим службам Интернета.

На рис. 5.1 показана упрощенная структура передачи сообщения в Интернете. В ней можно выделить следующие основные компоненты: абонент (пользователь) (1), провайдер (3 и 5), модем (2), система телекоммуникации (4).



Рисунок 5.1. Структура передачи сообщения в Интернете

В качестве посредника в сети выступает провайдер службы (сервисный центр Интернета — ISP).

Компьютер абонента сети связан с компьютером провайдера (хост-машина, файловый сервер и др.) с помощью телефонной линии и модема (в ряде случаев используется выделенный канал связи).

Модем — устройство, преобразующее знаки сообщения (документа) в последовательность сигналов. С помощью проводной связи модем подключается к последовательному порту компьютера (обычно это COM1). Применяют также и встроенный в компьютер модем (модемная карта, факс-модемная карта). В этом случае телефонная сеть подключается непосредственно к системному блоку компьютера.

Компьютер провайдера размещает на своем жестком диске информацию, полученную на адрес пользователя, и передает информацию по запросу самого пользователя постранично (по файлам).

5.1.2. Сетевые протоколы

Документы в глобальных информационных сетях (далее просто сетях) передаются при соблюдении соглашения о формате данных, соответствующей системы адресации и синхронизации, которые определяются как сетевые протоколы. Так как многие сети имеют свои сетевые протоколы, то непосредственное взаимодействие между сетями невозможно.

Объединение глобальных информационных сетей осуществляется через шлюзы (соответствующие компьютерные системы). Шлюз способен принять сообщение из одной сети и передать его в другую или доставить его по указанному адресу. На практике информация передается из одной сети в другую через множество последовательных шлюзов, которые обеспечивают сквозную маршрутизацию сообщений по всей сети.

Набор сетевых протоколов, которые использует Интернет (семейство протоколов межсетевое обмена), определяется как **Протокол управления передачей/протокол Интернета** (Transmission Control Protocol/Internet Protocol, TCP/IP). Эти протоколы — по существу машинный язык сети. Базовые

протоколы семейства (спецификации) TCP/IP содержат схемы адресации, которые опознают каждый компьютер, подключенный к сети, используя для этого IP-адрес компьютера, и прикладные программы.

Маршрутизаторы используют IP-адрес для перемещения сообщений по сети. Сообщение разделяется на пакеты, однако каждый пакет информации имеет IP-адреса компьютера-отправителя и компьютера-получателя.

IP-адрес — это уникальное имя компьютера, под которым он известен в сети. Это имя представляет собой 4-байтную последовательность десятичных чисел, разделенных точками. Например: 144.206.160.32 или 206.246.150.10 и др.

TCP определяет способ разделения сообщений на пакеты, их пересылку по сети и сборку сообщения перед его получением адресатом.

Числовая адресация удобна (для машинной обработки таблиц маршрутов, но неприемлема для ее использования человеком. Для решения проблемы применяют **систему доменных имен** (Domain Name System, DNS), в которой определено однозначное соответствие доменного имени компьютера и его IP-адреса.

DNS строится по иерархическому принципу. На верхнем уровне размещаются национальные домены (например, ru, jp, uk) и специальные домены: *gov* — правительственные учреждения; *mil* — военные организации; *edu* — учебные заведения; *com* — коммерческие организации; *net* — сервисные центры Интернета; *org* — прочие организации.

Домены первого уровня составляют правую часть имени. Левее следуют домены, определяющие регионы, например: *msk* — для Москвы или организации, *bmstu* — для Московского государственного технического университета им. Н.Э. Баумана, который имеет свою информационную сеть. Между именами доменов первого и второго уровня ставится разделительная точка. Например, *bmstu.ru*, *fcde.ru* — домен Межвузовского центра дистанционного обучения.

Левее следуют домены подразделений организаций, например: *ibm.bmstu.ru* — домен факультета ИБМ МГТУ им. Н.Э. Баумана.

Кроме DNS-адреса, в Интернете применяют и другую систему адресов. В частности, служба World Wide Web использует **универсальный локатор ресурсов** (Uniform Resource Locator, URL) для адресации файлов, расположенных на информационных серверах Интернета. URL-адрес состоит из названия протокола, по которому осуществляется доступ к ресурсу (например, *http*, *gopher*, *ftp*), адреса сервера и адреса файла.

URL-адрес в Интернете имеет следующий формат (в квадратных скобках указаны необязательные элементы адреса):

`protocol://[user[:password]@]host.domain[:port]/path/filename`, где **protocol**: // — название протокола, используемого для доступа к соответствующему ресурсу. Первый элемент отделяется от остальных двоеточием. В качестве примера обозначения первого элемента URL-адреса можно привести:

- ◆ *http* — используется для ресурсов, расположенных на WWW-серверах;
- ◆ *file* — применяется для доступа к файлам, расположенным на винчестере пользователя;
- ◆ *ftp* — используется для доступа к файловым архивам Интернета;
- ◆ *mailto* — необходим для адресов электронной почты.

После двоеточия следует двойная косая черта (правый слеш), за которой указывается адрес компьютера с искомым ресурсом;

user — необязательное имя пользователя, которое может потребоваться или для доступа к FTP-серверам, или для организации индивидуального адреса электронной почты, или для индивидуального адреса веб-страницы;

:password — пароль, ставится после имени сервера через двоеточие. Имя пользователя ресурса и его пароль отделяются от остальных частей адреса символом @, который раньше называли обезьянкой, а сейчас — собачкой;

host.domain — имя домена компьютера в символьном формате;

:port — номер порта, если указанный номер отличается от стандартного, используемого данным протоколом по умолчанию. Номер указывается через двоеточие;

/path — заключительная часть адреса локатора ресурсов (путь). Здесь указывается путь к файлу, в котором хранится запрашиваемый ресурс.

/filename — имя файла (искомый ресурс).

В заключительной части URL-адреса может размещаться служебная информация, определяющая условия взаимодействия клиент-программы с сервером.

Например: <http://www.fcde.ru/de> — URL-адрес указывает последовательно: протокол доступа к ресурсу сети Интернета, расположенному на WWW-сервере ([http-ресурс](http://)), имя домена (www.fcde.ru), имя домашней страницы журнала «Дистанционное образование». Здесь также может располагаться имя файла, имеющего расширение .html (гипертекстовый документ).

5.2. Программы просмотра документов информационных сетей



Программное обеспечение Интернета состоит из множества программ: веб-браузеры, клиент-программы для электронной почты и др.

Браузер* (browser, от англ. *browse* — просматривать) — программа для просмотра документов. Такая программа по инициативе абонента направляет запрос на сервер провайдера. Между компьютерами абонента и провайдера устанавливается соединение, и сервер направляет клиенту ответ на запрос, после чего соединение разрывается.

* В литературе нередко используют термин «броузер», что более точно передает фонетику слова browser.

Браузер содержит набор инструментов, позволяющих просматривать информационные материалы, размещенные в информационных сетях, выделять и копировать необходимую информацию и т. п. Такая программа является следствием развития WWW-службы, применяющей для передачи информации протокол передачи гипертекста (HyperText Transfer Protocol, HTTP).




Документы, которые браузер получает от WWW-сервера, обычно представляют собой документы, написанные на специальном языке, называемом гипертекстовым языком меток (HyperText Markup Language, HTML). Этот язык состоит из набора соглашений, в соответствии с которым в текстовый файл вставляются метки, определяющие размещение и вид документа в окне браузера.

В качестве наиболее активно используемых браузеров, можно указать на  Netscape Communicator и  Microsoft Internet Explorer. С их помощью осуществляется доступ ко всем службам Интернета: электронной почте, телеконференциям, WWW и др.

Обе упомянутые программы-браузеры имеют идентичную структуру, отличаясь лишь небольшими особенностями. О них можно судить после знакомства с программами. Для этого сначала ознакомимся с основными приемами работы в среде Netscape Communicator при рассмотрении методов использования электронной почты, а затем при изучении методов использования WWW-пространства — в среде Microsoft Internet Explorer.

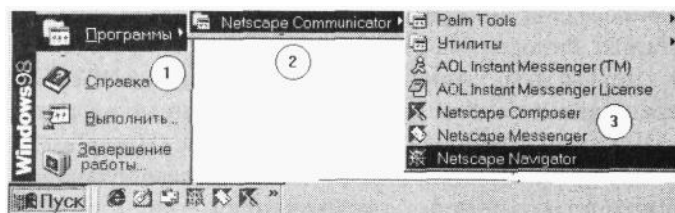
5.2.1. Оконный интерфейс браузера Netscape Communicator

Комплекс программ Netscape Communicator (NSC) — это набор из интегрированных друг с другом приложений, предназначенных для выполнения различных задач обмена в глобальных информационных сетях. В число приложений входят:

-  ◆ Netscape Navigator — веб-браузер, обладающий свойствами, облегчающими работу в информационных сетях;
-  ◆ Netscape Messenger — универсальная почтовая программа для работы с электронной почтой и почтовыми конференциями;
-  ◆ Netscape Composer — текстовый редактор для подготовки документов.

Программа NSC может работать на различных платформах.

Netscape Communicator (NSC) может использоваться как приложение Windows, и поэтому для работы с ним следует предварительно ознакомиться с приемами организации информации на рабочем столе Windows.

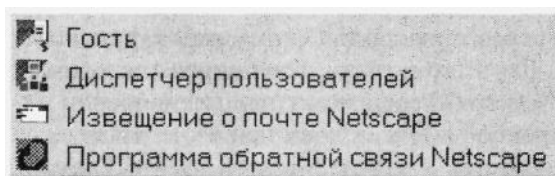


1 — панель первого уровня; 2 — панель второго уровня; 3 — панель третьего уровня

Рисунок 5.2. Вызов программ Netscape Communicator с помощью кнопки **Пуск**

На рис. 5.2 показан фрагмент рабочего стола, на котором раскрыто меню **Пуск**. Используя это меню, следует щелкнуть мышью по строке **Программы** (1), перейти на второй уровень меню (2) и щелкнуть по строке **Netscape**

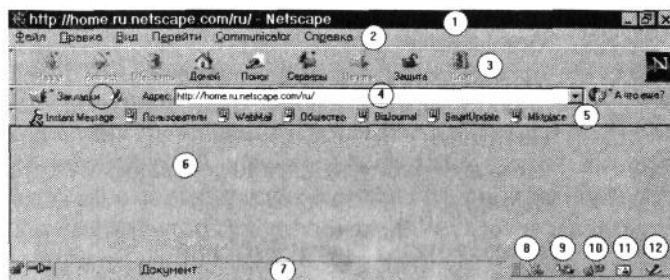
Communicator, затем на третьем уровне меню (3) щелкнуть по строке **Netscape Navigator**, чтобы перейти на рабочее окно браузера.



Строка **Утилиты**, размещенная на панели третьего уровня, позволяет раскрыть панель четвертого уровня, на которой размещена строка **Диспетчер пользователей**, обращение к которой используется для настройки программы электронной почты (установка опций идентификации — профиля пользователя).

Для начала работы в сети Интернет следует щелкнуть мышью по значку NSC или вызвать в меню **Пуск** программу **Netscape Navigator**. В процессе установки программы она начнет самостоятельно устанавливать связь с домашней страницей NSC. Следует оборвать этот процесс, щелкнув мышью по кнопке **Стоп**. Теперь можно активизировать меню **Файл** и выбрать опцию **Новый**, где щелкнуть мышью по опции

Navigator Windows. На рабочий стол будет вызвано окно Netscape Navigator, показанное на рис. 5.3.




1 — строка заголовка; 2 — строка меню; 3 — панель инструментов; 4 — панель закладок; 5 — панель каталогов; 6 — область просмотра документа; 7 — строка состояния; 8 — кнопка вызова **Netscape Navigator**; 9 — почтовый ящик (Mail box); 10 — электронные конференции (Discussions); 11 — адресная книга; 12 — вызов текстового редактора (Composer)

Рисунок 5.3. Окно Netscape Navigator

Окно Netscape Navigator, или WWW-браузер, обеспечивает полноценный доступ к информационным ресурсам Интернета.

Строка заголовка содержит стандартные элементы окна приложений Windows:

 — кнопка системного меню браузера;


 — кнопки свертки, раскрытия рабочего окна и закрытия приложения;









Официальная Информация - Netscape — заголовок, который состоит из названия просматриваемого документа и названия приложения.


Строка меню обеспечивает доступ к функциям Netscape: загрузка документов для просмотра и печати, настройка внешнего вида окна, перемещение между документами, аннотирование документов и др.

Панель инструментов — вызывает наиболее часто используемые команды NSC. Кнопки панели содержат надписи и пиктограммы, определяющие их функции. Те из кнопок, которые не могут выполнить предназначенные для них действия, на рис. 5.3 показаны затененными.

Кнопки панели управления имеют следующие назначения:

 — возврат к предыдущему документу. Дублирует строку меню **Go | Back**;

-  — переход к следующему документу из списка, просмотренных в текущем сеансе работы. Дублирует строку меню **Go | Forward**;
-  — перезагрузка текущего документа. Дублирует строку меню **View | Reload**;
-  — переход к документу, установленному в качестве домашней страницы при настройке программы. Реализует строку меню **Go | Home**;
-  — Поиск файла или каталога. Реализует строку меню **Edit | Search**.
-  — путеводитель по сети Интернет;
-  — печать текущего документа. Реализует строку меню **File | Print**;
-  — установка защиты на документы пользователя. Реализует строку меню **Communicator | Security Info**;
-  — останавливает процесс загрузки документа. Дублирует строку меню **Go | Stop Loading**.

 **Панель закладок** содержит URL-адрес документа, который можно записать в строке **Адрес** или выбрать из раскрывающегося списка после щелчка мышью по кнопке раскрытия окна (см. рис. 5.3, 4).

На панели закладок находится многоуровневый раскрывающийся список **Закладки** (рис. 5.4).

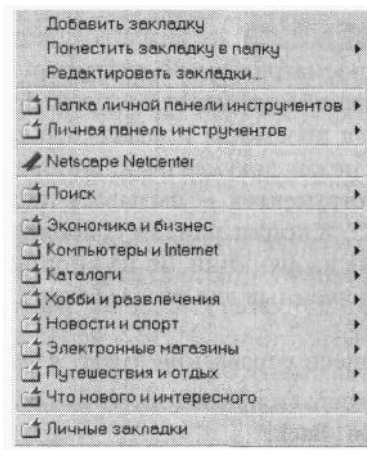



Рисунок 5.4. Список панели закладок

Панель персональных каталогов — содержит кнопку регистрации абонента электронной почты (Instant Message) и раскрывающиеся списки URL-адресов Интернета, сгруппированные в несколько областей: **Пользователи**, **WebMail**, **Общество** и др. Щелчок по каждому из значков области позволяет раскрыть список адресов сети, которые занесены пользователем NSC и отражают его информационное пространство в сети. Каждый список обеспечивает быстрый доступ к различным ресурсам WWW-пространства: документам и каталогам различных веб-серверов, включая и серверы компании Netscape Communications.

Показанные каталоги можно переименовать и настроить на каталоги пользователя браузера таким образом, что вызов необходимого документа можно осуществить с помощью обращения к одному из установленных пользователем каталогов.

 Все три панели (панель каталогов, панель закладок и панель инструментов) можно менять местами. С помощью щелчка мыши их можно убрать со стола, оставляя на их месте небольшие закладки.

Каталоги панели закладок можно переименовывать, а также можно заменить содержание самих каталогов.

Область просмотра документа — это главная область рабочего окна, на которой размещается текст документа со встроенными изображениями.

Строка состояния — отображает текущее действие программы, полученный объем документа и объем графических изображений: в левой части строки отражено число принятых байт и общий объем

загружаемого объекта, в правой — графический индикатор, указывающий объем полученной части общего объема передаваемых данных.

<http://www.bmstu.ru/general.html>

Во время просмотра документа на статусной строке размещен URL-адрес гипертекстовых ссылок, на которые позиционируется курсор мыши.


Здесь иногда авторы документов размещают бегущую строку, что не следует делать, так как бегущая информация затрудняет чтение URL-адреса.

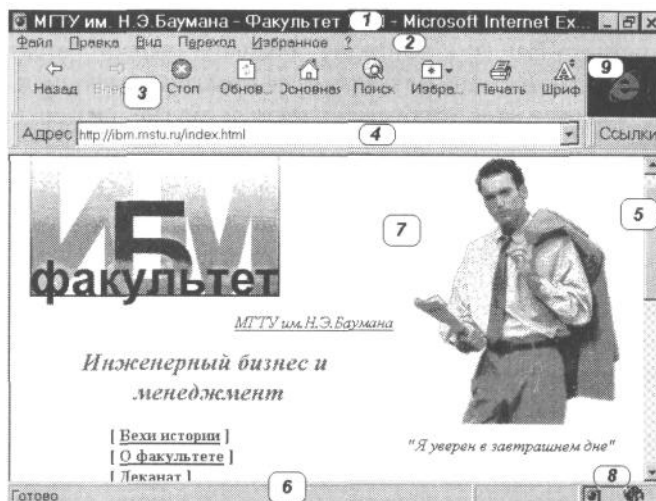
В правой части статусной строки устанавливается группа кнопок, позволяющая вызвать Netscape Navigator, — почтовый ящик, телеконференции, адресную книгу, текстовый редактор NSC (см. рис. 5.3, 12).

 В левом углу статусной строки размещен значок защиты информации NSC.

5.2.2. Оконный интерфейс браузера MS Internet Explorer

Среда Microsoft Internet Explorer нередко рассматривается как браузер для работы с веб-документами, а остальные службы Интернета – как дополнение к этой функции.

 Для запуска браузера необходимо щелкнуть по соответствующему значку на рабочем столе Windows или использовать программное приложение MS Explorer (Проводник). На рис. 5.5 показано рабочее окно программы. Оно содержит традиционные для программ Microsoft элементы: строка заголовка, строка меню, панель инструментов, линейка прокрутки, рабочее (главное) окно, строка состояния. Кроме указанных элементов, на экране установлены две дополнительные панели, на первой из которых помещена строка URL-адреса, а на второй — кнопки обращения к наиболее популярным ресурсам Интернета.



- 1 — строка заголовка; 2 — строка меню; 3 — панель инструментов;
- 4 — строка URL-адреса; 5 — линейка прокрутки; 6 — строка состояния;
- 7 — рабочее окно браузера; 8 — кнопка изменения шрифта (глобус);
- 9 — индикатор активности

Рисунок 5.5. Окно программы Internet Explorer

Назначение и возможности применения элементов экрана Internet Explorer следующие.

Строка заголовка содержит имя текущей веб-страницы или другого файла, изображаемого в окне, кнопки управления размером экрана (кнопки минимизации и максимизации) и кнопку окончания сеанса работы с программой.

Строка меню содержит наборы команд, сгруппированные по функциональному признаку:

- ◆ **Файл** — команды, управляющие работой программы при обработке HTML-документов и других файлов;
- ◆ **Правка** — команды, дающие возможность редактировать документ: вырезать, вставлять, переносить текст документа и другие операции;
- ◆ **Вид** — команды, определяющие облик окна Internet Explorer;

- ◆ **Переход** — команды, позволяющие перемещаться между различными файлами и веб-страницами;
- ◆ **Избранное** — команды перемещения ярлыка текущей веб-страницы в личную папку и пунктов из личного списка в подпапки, здесь также находится обращение к списку ярлыков личной папки **Favorites**;
- ◆ **Справка** — команды вывода информации о возможностях и методах работы программы. В меню этот набор команд обозначен знаком вопроса.

Панель инструментов содержит три части: набор командных кнопок, реализующих наиболее часто повторяющиеся команды меню, поле URL-адреса и группа быстрых связей (ссылки) к веб-узлам, которые рекомендуются авторами программы или назначаются самим пользователем.

Для раскрытия группы быстрых связей следует щелкнуть по кнопке **Ссылки**. Группа связей, на которой позиционируется курсор мыши, изменяет свой цвет (рисунок значка становится цветным) и показывается в форме кнопки (рис. 5.6). Подстрочная подпись определяет назначение выделенной кнопки (для данного примера URL-адрес WWW-сервера фирмы Microsoft). Для возврата к окну адреса необходимо щелкнуть по кнопке **Адрес**.

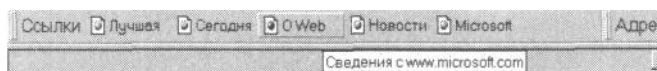


Рисунок 5.6. Панель быстрых связей

Линейка прокрутки позволяет перемещать поле документа как по вертикали, так и по горизонтали. Если документ помещается на экране, то линейки прокрутки не изображаются.

Строка состояния может содержать адрес текущего веб-узла, имя конкретного файла, к которому обращена ссылка, бегущую строку. Здесь также даются данные о процессе передачи информации.

Рабочее окно отображает текст и другие элементы документа веб-страницы или файла.

Глобус осуществляет переход от одной кодировки к другой в случае, если документ использует несколько алфавитов.

Индикатор активности содержит символ программы, который находится в движении, когда осуществляются процессы приема или передачи документов.

Панели экрана можно перемещать и объединять в одной строке. Для выполнения этих операций следует:

1. Поместить курсор на название раздела. Курсор примет вид указывающего пальца.
2. Перетащить панель в нужное место (вверх или вниз).

Для изменения конструкции самой строки следует установить курсор на левой части перемещаемой панели и перетащить ее влево или вправо. Результат такого перемещения представлен на рис. 5.7.



1 — панель кнопок управления; 2 — панель быстрых связей; 3 — адресная строка; 4 — ярлычки панели инструментов

Рисунок 5.7. Реорганизация оформления панели инструментов

Дублирование активных окон. Для удерживания текущего документа в отдельном окне и осуществления различных операций над текстом других документов применяют метод дублирования активных окон. Метод позволяет работать над текущим документом в тех случаях, когда идет длительная загрузка нового документа, осуществляется продолжительный диалог, обновление экрана (в игровых документах) и т. п.

Для выполнения операции дублирования следует в меню **Файл** активизировать опцию **Создать окно**. В этом окне необходимо осуществить вызов нового документа.

Панель быстрых связей. На рис. 5.7 показана панель быстрых связей с конкретными веб-узлами. Она состоит из 5 групп ссылок: Лучшая, Сегодня, O Web, Новости, Microsoft. При инсталляции MS Internet Explorer формирует список узлов фирмы Microsoft. Однако каждой связи (ссылке) можно

назначить любые другие группы адресов, которые может установить сам пользователь. По умолчанию быстрые связи выполняют следующие функции:

- ◆ **Лучшая** — выводит на страницы сервера, содержащего, по мнению фирмы, самые важные сообщения;
- ◆ **Сегодня** — обеспечивает переход на страницу центральной MSN-связи (MSN's Link Central). Она содержит связи с новыми и временными веб-узлами, включающими страницы новостей, спортивных событий, новых кинофильмов и телепрограмм;
- ◆ **О Web** — предлагает страницы, содержащие руководство по использованию Интернета;
- ◆ **Новости** — содержит адреса страниц, специализирующихся на последних новостях в мире Internet Explorer;
- ◆ **Microsoft** — подключает к страницам, содержащим информацию о последних разработках фирмы Microsoft, о бесплатном копировании программ, о новых фрагментах программного обеспечения.

Для изменения содержания групп быстрых связей необходимо выполнить следующую последовательность действий:

1. Перейти на веб-страницу, к которой следует установить быстрое подключение.
2. Открыть меню Вид и активизировать опцию **Параметры**. В результате на экран будет вызвана вкладка **Параметры** (рис. 5.8).

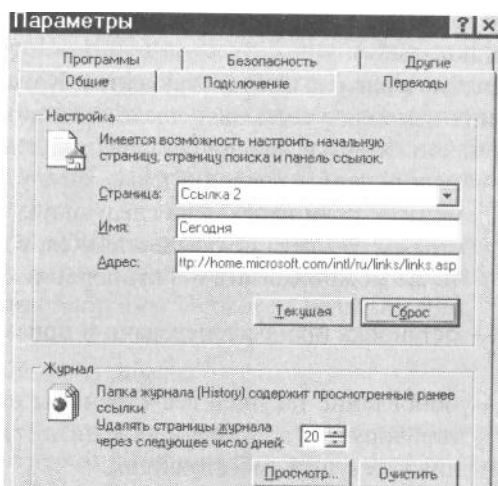


Рисунок 5.8. Вкладка **Параметры**

3. Щелкнуть по листу **Переходы**.
4. Открыть раскрывающийся список **Страница** (рис. 5.9) и щелкнуть мышью по той строке, которая обозначает группу связей, где показана ранее установленная быстрая связь, например **Ссылка 2**.

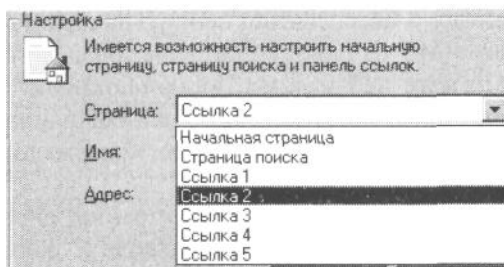


Рисунок 5.9. Раскрывающийся список номеров ссылок


5. Для установки группы связей и ее переименования следует в поле **Имя** записать новое имя группы. Это имя будет установлено на соответствующей кнопке панели быстрых связей. Следует ограничить название группы 9 знаками.


6. Необходимо проконтролировать корректность записи URL-адреса подключаемой страницы в строке **Адрес**.

7. Для подключения быстрой связи к текущей веб-странице следует щелкнуть по кнопке **Текущая**, расположенной под строкой **Адрес**.


8. Щелкнуть последовательно по кнопкам **Применить** и **ОК** управляющего меню вкладки, расположенного внизу поля вкладки (на рисунке не показаны).


Командные кнопки панели инструментов. Командные кнопки панели управления изменяют свой цвет после позиционирования на их поле курсора мыши. Кнопки панели составляют следующий набор:


 — перемещение вперед и назад между веб-страницами документов. Если переход на следующую страницу по гипертекстовым ссылкам не осуществляется, то кнопка не высвечена.

 То же можно сказать и в отношении кнопки **Назад**;

 — остановка процесса передачи и приема информации;


 — обновление текущей веб-страницы на экране браузера. Активизируется в случае необходимости повторения обращения к текущей веб-странице;

 — переход на домашнюю страницу;

 — режим поиска файла или веб-страницы;

 — активизация команд меню **Избранное**;

 — вывод на печать текущей веб-страницы или файла;

 — изменение размера шрифта, используемого для показа текущей веб-страницы или файла. Применяется пользователем с ослабленным зрением.

5.3. Электронная почта

Схема работы электронной почты аналогична обычной почте. На каждого абонента заводится электронный адрес и выделяется некоторое пространство на жестком диске компьютера почтового сервера провайдера (каждому почтовому адресу — почтовый ящик). Доступ к электронному почтовому ящику закрывается паролем (ключ от ящика). Пользователь отправляет письма, помещая их в почтовый ящик. Пользователь получает письма, забирая корреспонденцию из почтового ящика.

Если адрес в электронном послании указан неверно, то оно возвращается адресату с соответствующей пометкой.

Механизм доставки электронной почты (как и в обычной почте) скрыт от адресатов. Отличие от традиционной почты определяется лишь временем доставки письма. Для электронной почты оно составляет, как правило, несколько минут.

Рассмотрим процесс работы в сервисе электронной почты, используя в качестве программного приложения почтовую программу Netscape Mail, входящую в состав программ Netscape Communicator.

5.3.1. Установка почтового ящика

Почтовый ящик организуется после заключения соответствующего договора между провайдером и пользователем. Программное обеспечение, предназначенное для автоматизации процесса обмена документами с компьютером провайдера (передача исходящих и прием входящих документов), принято называть **почтовой программой**. Ее суть заключается в обеспечении необходимого интерфейса для приема, просмотра, подготовки, обработки и рассылки почты конкретного пользователя компьютера (если их несколько).

Для работы в режиме электронной почты необходимо установить свой адрес, определить настройки в почтовой программе и идентифицировать пользователя. Последняя операция определяется как **установка профайла почты на компьютере пользователя***.

* В последнее время широкое распространение получило жаргонное определение профайла как профиля абонента.

Настройка почтовой программы для ее использования конкретным пользователем (или установка профиля пользователя на его компьютере) осуществляется в следующей последовательности.

Вначале следует вызвать программу (организация данных пользователя) с помощью **Проводника** или используя меню кнопки **Пуск**. Далее необходимо перейти на третий уровень меню и активизировать опцию **Утилиты**, а затем щелкнуть по надписи **Диспетчер пользователей**.

Диалог **Диспетчер пользователей**, изображенный на рис. 5.10, содержит область информации о ранее установленных пользователях и область кнопок управления, записи и редактирования списка пользователей: **Создать**, **Переименовать**, **Удалить**, **Назад**.

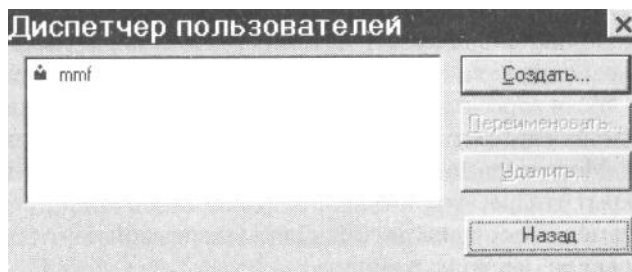


Рисунок 5.10. Диалог Диспетчер пользователей

Для идентификации нового пользователя следует нажать на кнопку **Создать**.

В процессе выполнения всех операций требуется внимательно рассмотреть каждый диалог, записать необходимые данные и щелкнуть по кнопке **Далее**. Если обнаружена ошибка, то можно вернуться на предыдущий диалог установки и внести необходимые исправления.

Новый диалог содержит текстовое послание, предупреждающее о необходимости установки файла идентификации для каждого почтового абонента, работающего на этом компьютере. Следует снова щелкнуть по кнопке **Далее**.

В следующем диалоге установки данных (рис. 5.11) следует в поле **Полное имя** записать полное имя отправителя почтовых сообщений, например Сергей Петров.

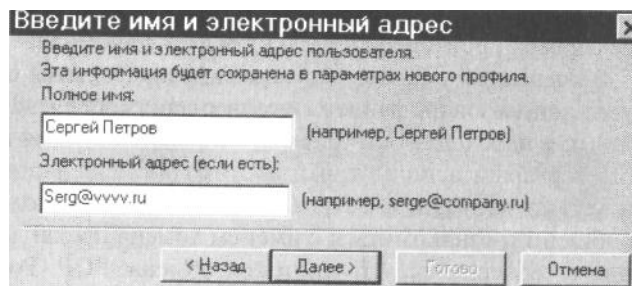


Рисунок 5.11. Диалог установки имени отправителя почты и его почтового адреса

В поле **Электронный адрес** необходимо записать почтовый адрес абонента, например Serg@ww.ru; здесь следует указать полный адрес, который состоит из имени пользователя (буквенного кода), символа @ и адреса почтового сервера. Этот адрес выделяется провайдером. В имени адреса следует использовать только знаки латинского алфавита (пробелы недопустимы). Затем необходимо щелкнуть по кнопке **Далее**.

Диалог, приведенный на рис. 5.12, позволяет записать полное имя файла настроек получателя почты (Profile name). Этот шаг позволяет разделить файлы настроек различных пользователей почтовой программы.

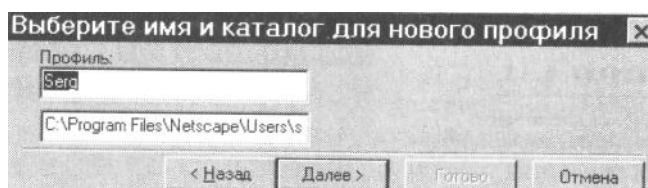


Рисунок 5.12. Фрагмент диалога установки полного имени профиля

Здесь следует записать в поле окна **Профиль** имя файла, в котором программа сохранит установленные пользователем настройки, закладки, установки, и каталог, в который будут направлены новости и текущая корреспонденция.

В поле второго окна нужно записать имя каталога сервера провайдера, на котором будет расположен почтовый ящик пользователя, например: C:\Student\Group4\ivanov. Однако лучше не вносить изменения в предложенный программой адрес и щелкнуть по кнопке **Далее**.

Последний диалог переводит абонента в режим контроля ранее введенных данных. Здесь необходимо проконтролировать правильность записи имени отправителя почты, его почтового адреса и почтовый адрес сервера провайдера и щелкнуть по кнопке **Далее**.

В очередном диалоге необходимо ознакомиться с именем, которое будет использовать файловый сервер для распределения входящей почты (при необходимости, внести исправления), имя входящего файлового сервера и используемый протокол обмена данными. Здесь следует: установить (или внести исправления) в имя получателя почтовых сообщений, ознакомиться с именем домена провайдера (адресом файлового сервера), указать протокол обмена: POP (Post-Office Protocol) или IMAP (Internet Mail Protocol). Тип протокола определяет провайдер, и поэтому не рекомендуется его изменять. Затем следует щелкнуть по кнопке **Далее**.

В последнем диалоге можно показать адрес ввода документов для сервера телеконференций, однако целесообразно эту операцию выполнить позже, а в данном случае диалог следует пропустить и нажать на кнопку **Готово**.

На этом заканчивается процесс идентификации пользователя почтового ящика. NSC запускает программу Netscape Navigator с установкой домашней страницы. Для останова процесса следует щелкнуть по кнопке **Стоп**, а затем — по кнопке **Домой**.

При новом обращении к браузеру на экране появляется диалог диспетчера пользователя, в окне которого показан список зарегистрированных абонентов (рис. 5.13).

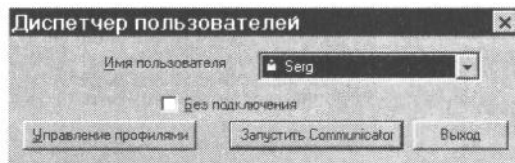


Рисунок 5.13. Диалог диспетчера абонентов электронной почты

Для работы с приложениями NSC следует выбрать свою строку и щелкнуть по кнопке **Запустить Communicator**.

5.3.2. Редактирование файла настроек

Для внесения изменений в файл настроек почтового ящика следует воспользоваться командой **Параметры** системного меню **Правка**, диалоговое окно которой показано на рис. 5.14, а затем перейти на строку **Почта и конференции** в окне **Категория**.

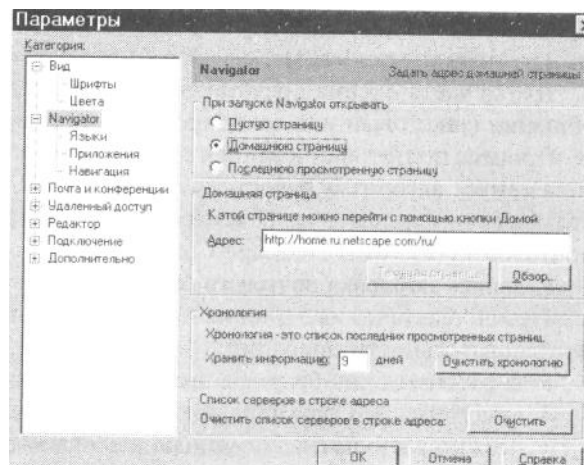


Рисунок 5.14. Диалоговое окно **Параметры** меню **Правка**

Диалог позволяет не только настроить внешний вид браузера, но и внести изменения в настройку домашней страницы, определить приложения, способы навигации и др.

5.3.3. Структура электронного письма

Электронное письмо состоит из следующих основных элементов:

1. Заголовок письма.
2. Характеристика письма.
3. Текст письма.

Первые два элемента письма образуют «конверт с адресом». Информация конверта используется различными компьютерными системами, и поэтому для правильной ее обработки применяют международный формат электронных сообщений, который определен документом Standart for the Format of ARPA — Internet Text messages или RFC822 (Request for Comment), его также называют форматом Интернета.

В соответствии с описанием этого формата строки в начале письма содержат описание маршрута движения сообщения по цепочке хост-машин и сведения о местоположении (IP-адрес) компьютера провайдера (хост-машины) в сети, об используемом программном обеспечении, времени получения входящего и времени отправления исходящего письма и другую информацию.

Заголовок письма формируется автоматически с учетом тех адресов, которые запишет отправитель в процессе подготовки сообщения.

Вторая часть письма содержит характеристики самого письма. Они также генерируются автоматически в процессе отправки письма.

Третья часть письма представляет собой непосредственно текст сообщения (текстовый или закодированный бинарный файл).

Процесс подготовки и записи письма осуществляется по вполне устоявшемуся алгоритму и состоит в заполнении соответствующих текстовых окон приложения. Покажем этот процесс, используя в качестве почтовой программы Netscape Communicator.

1. Запись заголовка почтового сообщения.

Запись текста нового письма предполагает выполнение следующей последовательности операций:

- ♦ в рабочем окне браузера вызвать интерфейс почтовой программы Netscape Communicator, который показан на рис. 5.15. Это можно выполнить с помощью щелчка мышью по соответствующей пиктограмме, расположенной на статусной строке;

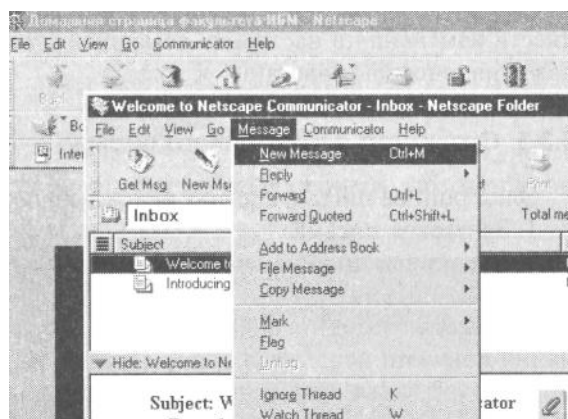
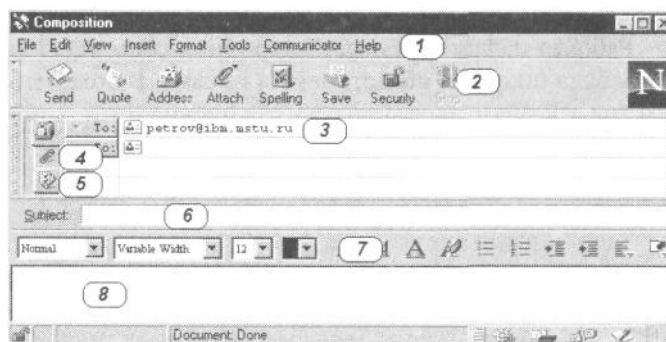


Рисунок 5.15. Окно диалога почтовой программы

- ♦ раскрыть меню **Message**; 
- ♦ щелкнуть мышью на опции **Новое письмо** или нажать клавиши **<Ctrl>+<M>**.

Последние две операции можно заменить одним щелчком по кнопке **Создать сообщение**, расположенной на панели инструментов.









В результате выполнения команды на рабочий стол устанавливается окно **Message Composition**. Оно содержит область команд, панель инструментов и рабочую область (рис. 5.16).



1 — операционное меню; 2 — панель инструментов; 3 — строки адреса получателя; 4 — строка имени файла, присоединяемого к письму; 5 — строка установки условий доставки письма; 6 — строка темы письма; 7 — панель форматирования текста письма; 8 — поле сообщения


Рисунок 5.16. Окно **Message Composition**

Панель инструментов окна **Message Composition** включает в себя 8 кнопок:

-  — отправляет составленное сообщение. Дублирует пункт меню **Файл | Send now** (немедленная отправка) или **Файл | Send later** (задержка отправки) в зависимости от установленного режима отправки письма;
-  — включает в редактируемое сообщение текст выбранного письма (на которое пишется ответ) в виде цитаты. Рекомендует пункт меню **File | Include original text**;
-  — вызывает окно **Attachments** для выбора присоединяемого файла. Дублирует пункт меню **File Attach File**;
-  — вызывает окно **Select Addresses** для выбора одного из адресов, записанных в адресную книгу;
-  — проверяет орфографию текста письма;
-  — вызывает окно сохранения подготовленного письма;
-  — устанавливает защиту на отправленное сообщение;
-  — останавливает процесс отправки сообщения. Кнопка используется при отсутствии доступа к почтовому серверу.

Рабочая область **Message Composition** состоит из двух частей: поле заголовка письма и область текста письма. В поле заголовка размещены три закладки: адреса получателя письма и тех, кому предназначены копии сообщения, имя прикрепляемого к письму файла, условия доставки письма. В поле заголовка размещается строка с указанием темы письма.

Запись данных для заголовка (обязательных реквизитов) письма осуществляется в следующей последовательности:

-  — щелкнуть по первой закладке адресной панели (см. рис. 5.16, 3). Ввести в поле **To:** адрес получателя письма. Если необходимо послать копию послания по другим адресам, то после нажатия клавиши **<Enter>** на следующей строке следует записать второй адрес **Сс** (Carbon сору, копирование копии, под копирку). В этом случае получатель оригинала будет знать, кому послана копия письма. Если записать адрес на третьей строке **ВСс** (Blind Сс, слепое копирование копии), то получатель оригинала не узнает, кому еще послана копия письма;

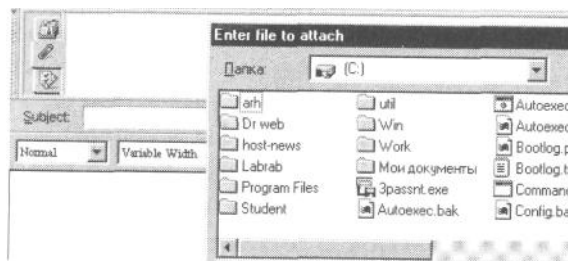




Рисунок 5.17. Окно выбора файла, прикрепляемого к письму

-  – щелкнуть по второй закладке с изображением скрепки (см. рис 5.16, 4) (Attachment, прикрепление). Если к тексту письма следует прикрепить файл: дважды щелкнуть по полю окна закладки, выбрать на схеме появившегося каталога требуемый файл и щелкнуть по его имени мышью (рис. 5.17);
-  – перейти на третью закладку-строку адресной панели и активизировать ее (рис. 5.16,5). С помощью этой строки задаются различные установки условий доставки письма: получить уведомление о получении письма, задать приоритет доставки и др., например, необходимость извещения о получении письма, установить степень важности (приоритет), необходимость подать сигнал в случае получения письма и др. (рис. 5.18);

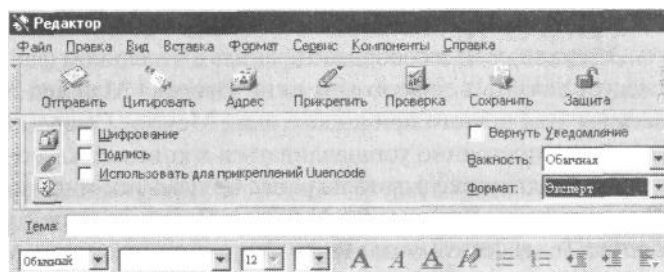



Рисунок 5.18. Установка условий доставки письма

- записать тему письма в поле **Subject** (см. рис. 5.16,6). При отправлении письма зарубежным партнерам следует заполнить эту строку либо соответствующим текстом на английском языке, либо используя латинские буквы, например Noviy vzglyad na peremeni;
- установить кегль и начертание шрифта текста письма, размещение текста по полю листа с помощью кнопок панели форматирования (см. рис. 5.16, 7).

2. Запись текста письма:

-  – написать текст письма на поле **Сообщение** (см. рис. 5.16,8);
- для отсылки сообщения нажать на кнопку **Send**.

При написании текста письма необходимо придерживаться следующих правил:

1. На первой строке следует написать имя адресата, например:

Г-н Иванов С.И., или, если адресат Вам хорошо известен, можно вставить приветствие:

Здравствуй, Сергей.

Если имя адресата неизвестно или письмо адресовано коллективному адресату (организации, предприятию и т. п.), то можно обойтись только приветствием, например:

Здравствуйте, или Hello.

Первая строка заканчивается запятой!

2. Пропустить одну строку и написать текст письма.

3. Пропустить строку письма и завершить письмо вежливыми пожеланиями типа:

Наилучшие пожелания, или Best regards,.

4. Пропустить еще одну строку и написать Ваше имя и дату отправки письма.

5. В конце письма можно указать некоторые данные об отправителе письма: организация, должность, телефон и т. п.

Эти данные целесообразно хранить в отдельном файле, имя которого следует указать в диалоговом окне **Options Mail and News Preferences | Identity**. После этого при вызове окна **Message Composition** эта информация будет постоянно устанавливаться в конце каждого нового письма.

При подготовке ответа на ранее полученное письмо следует использовать кнопку **Reply** (окна Netscape Communicator, см. рис. 5.16) или опцию **To sender** команды **Reply** операционного меню **Message**, или нажать горячие клавиши **<Ctrl>+<R>**. На рабочий стол будет установлено окно **Message Composition**, на котором все обязательные поля будут заполнены и на поле письма показана копия ранее полученного послания. Остается выполнить следующие операции:

а) после знака «>», которым отделяется текст ранее полученного письма, записать текст ответа на письмо;

б) установить при необходимости дополнительные адреса рассылки;

в) щелкнуть по кнопке **Send** для отправки письма.

При подготовке ответа на письмо можно изменить обозначение темы сообщения, включить целиком или частично текст присланного письма.

3. Отправка сообщений.



Почтовая программа предлагает два режима отправки сообщений (меню **Options** окна **Message Composition**):

- ◆ Immediate Delivery (немедленная доставка) — режим срочной отправки писем: сообщение отправляется сразу после нажатия кнопки **Send**;
- ◆ Deferred Delivery (отложенная доставка) — режим отложенной отправки писем: после нажатия на кнопку **Send** письмо помещается в почтовый ящик **Outbox**, а для отправки сообщения активизируется команда **Send Mail in Outbox** из меню **File**.

После успешной отправки письма окно **Message Composition** удаляется с рабочего стола. Если процесс отправки письма затягивается, следует щелкнуть мышью по кнопке **Stop** на панели инструментов, чтобы прервать процесс соединения с почтовым сервером и проверить его настройку.

По умолчанию в окне диалога установлены опции, обеспечивающие немедленную отровку письма и пересылку копии отправленных писем в ящик **Sent**.

5.3.4. Обработка почтовых сообщений



Для работы с поступившей почтой необходимо запустить программу **Netscape Messenger**, используя меню кнопки **Пуск** рабочего стола или пиктограмму почтовой программы. Она располагается на статусной строке рабочего окна Netscape Communicator.

Вызвать программу можно с помощью команды меню **Communicator Messenger mailbox**.

В результате действия одной из перечисленных операций программа запросит пароль, а затем автоматически выполнит обращение к почтовому ящику (команда **Get New Mail**), расположенному на компьютере провайдера для получения самой свежей почты.

Затем на рабочий стол будет вызвано рабочее окно программы **Messenger Mailbox**, состоящее из двух частей: первая — список заголовков писем, вторая — содержание письма, выделенного в списке заголовков (рис. 5.19).

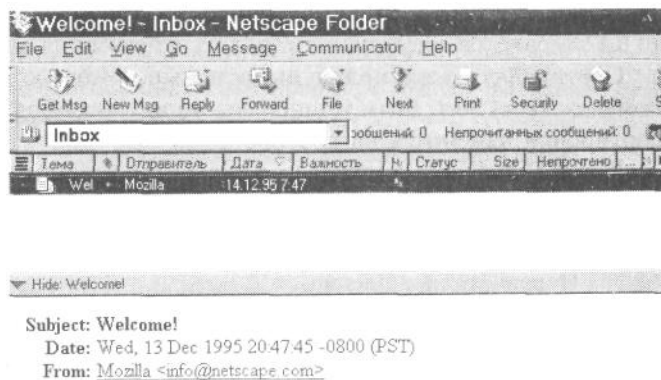
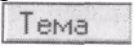
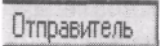
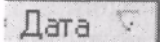
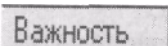
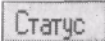

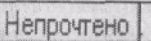


Рисунок 5.19. Интерфейс программы **Messenger Mailbox**

Между двумя областями рабочего окна установлена строка темы раскрытого письма.

Область письма можно увеличить за счет удаления списка писем, находящихся в почтовом ящике. Для этого следует щелкнуть по левой стрелке разделительной строки: первая область рабочего окна исчезнет, а текст окна займет все поле окна. Для восстановления изображения следует щелкнуть по той же стрелке, которая к этому времени переместится на левый край статусной строки.

Информация о письмах почтового ящика размещена в форме таблицы, на каждой строке которой размещены столбцы — данные о реквизитах письма:


	– тема сообщения, указанная его автором при отправке письма;
	– отправитель сообщения, в качестве которого используются или имя корреспондента, или адресное имя;
	– дата и время отправки письма;
	– приоритет доставки письма;
   ...	— другие поля-характеристики письма.

Заголовки полей выполнены в виде кнопок. Кнопки можно использовать в целях сортировки списка по соответствующему признаку. Для этого следует щелкнуть по одной из кнопок строки-заголовка таблицы списка писем.

Ширину и порядок следования кнопок заголовка таблицы можно изменять, захватывая их границы и перетаскивая их в нужное место с помощью мыши. Если текст операции не помещается на поле кнопки, то он заменяется сокращением, вплоть до указания знака многоточия.

Сортировку писем можно выполнить с помощью меню, используя команду View | Sort. Если установить опцию Ascending, то направление сортировки изменится на противоположное.

Каждое сообщение может иметь свой статус. В качестве статуса используются признаки:

 **Особо важное письмо;**

 **Непрочитанное письмо.**

Непрочитанные письма отмечаются в списке писем полужирным текстом и зелеными ромбиками в специальном поле. Сообщение считается прочитанным, после того как оно было отображено в области просмотра писем. При этом зеленый ромбик исчезает, а сообщение о письме отображается без выделения.

Для восстановления статуса непрочитанного письма необходимо щелкнуть мышью на том месте, где раньше стоял ромбик, или выполнить следующую последовательность операций:

- ◆ выделить строку с указанием восстанавливаемого письма;
- ◆ открыть меню **Message**;
- ◆ активизировать опцию **Mark as Unread**.

Аналогично можно отмечать как прочитанные письма, так и те, которые не читались. Для этого следует щелкнуть мышью по зеленому ромбику или активизировать команду **Message | Mark as Read**.

Красные флажки используют для пометки особо важных сообщений. Для пометки письма красным флажком следует в соответствующей строке в поле флажка щелкнуть мышью или активизировать команду **Message | Flag**. Для снятия флажка следует щелкнуть по нему или вызвать команду **Message | Unflag Message**.

Каждый пользователь почтовой программы использует систему почтовых ящиков. По умолчанию Netscape Mail устанавливает четыре почтовых ящика для каждого абонента:

- ◆ **Sent** — содержит копии всех отправленных писем;
- ◆ **Unsent messages** — письма, при отправке которых была установлена опция отложенной отправки;
- ◆ **Trash** — корзина (мусорка), в которую помещают сообщения, удаленные из всех почтовых ящиков;
- ◆ **Samples** — примеры;
- ◆ **Drafts** — заготовки, черновики, наброски и т.п.

Можно создать и использовать другие почтовые ящики (папки).

Над списком писем, размещенных в почтовом ящике, установлено окно выбора почтового ящика. По умолчанию открыт почтовый ящик **Inbox** — входящая почта (рис. 5.20).

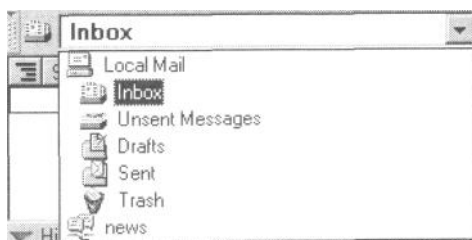


Рисунок 5.20. Список почтовых ящиков (папок) для писем почтовой программы

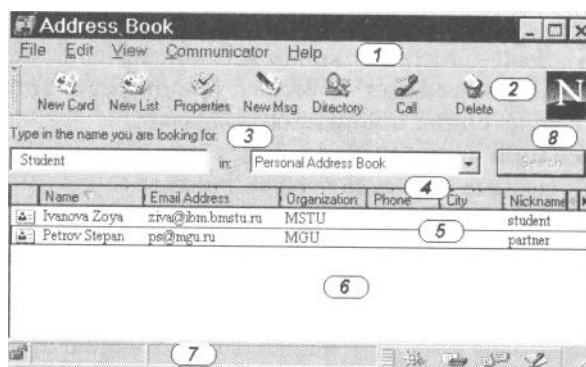
Для просмотра других почтовых ящиков пользователя писем следует раскрыть окно и щелкнуть по имени одного из списка.

5.3.5. Работа с адресной книгой

Адресная книга **Netscape Mail** предназначена для записи имен, краткой характеристики, адреса электронной почты и псевдонима адресата. Эту информацию можно использовать в адресных полях заголовка отправляемых писем.

Используя псевдонимы (клички) адресатов, можно ускорить процесс вызова и установки в письмо необходимых данных адресата.

Для вызова адресной книги следует воспользоваться командой меню **Communicator | Address Book**. После активизации этой команды на экране появляется окно адресной книги, показанное на рис. 5.21.



1 — строка системного меню; 2 — панель инструментов; 3 — текстовая строка имен; 4 — строки адресатов; 5 — запись данных адресата; 6 — рабочее поле окна; 7 — статусная строка; 8 — кнопка поиска информации

Рисунок 5.21. Окно адресной книги

Информация адресной книги хранится в файле **address.htm**, расположенном в том каталоге, где установлен Netscape Communicator. Для изменения имени и расположения адресного файла следует воспользоваться командой меню **File | Save As**, а при необходимости ее использовать активизировать команду **File | Import**.



Для записи данных о новом адресате следует щелкнуть по первой кнопке панели инструментов **New Card**. На рабочий стол будет установлена вкладка, показанная на рис. 5.22. Вкладка состоит из трех листов: **Name**, **Contact**, **Netscape Conference**.

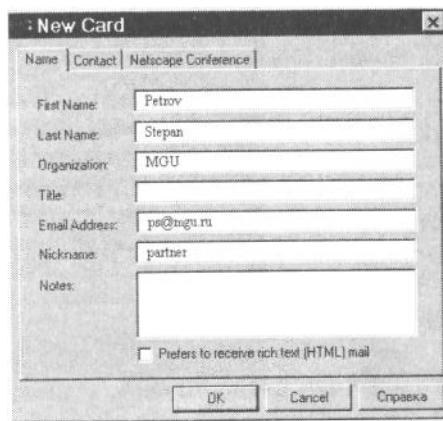


Рисунок 5.22. Лист **Name** вкладки **New Card**

Содержание первого листа **Name** имеет следующие окна для ввода данных:

- ◆ **First Name:** — имя адресата;
- ◆ **Last Name:** — фамилия адресата;
- ◆ **Organization:** — место работы (учебы, службы и т.п.) адресата;
- ◆ **Title:** — звание (форма обращения);
- ◆ **E-mail Address:** — адрес электронной почты, который устанавливается в поле **Mail To**;
- ◆ **Nick name** — псевдоним адресата (здесь следует говорить о кратком обращении к адресату, в некоторых случаях такое обращение определяют как «кличка», «прозвище» и т.п.). На рис. 5.22 показано использование слова «Partner» в качестве псевдонима адресата;
- ◆ **Notes** — краткое описание адресата. Информация из этого поля не используется при отправлении письма, но позволяет напомнить о некоторых особенностях корреспондента (дата рождения, вид деятельности, финансовое положение, наличие детей и т.п.).

После заполнения первого листа вкладки следует записать информацию для контактов с адресатом с помощью других средств коммуникации. Общий вид листа **Contact** показан на рис. 5.23.

На третьем листе вкладки — **Netscape Conference** — устанавливаются формы взаимодействия с электронными конференциями фирмы Netscape Communication.

Работа с адресной книгой осуществляется с помощью кнопок управления, расположенных на панели инструментов.

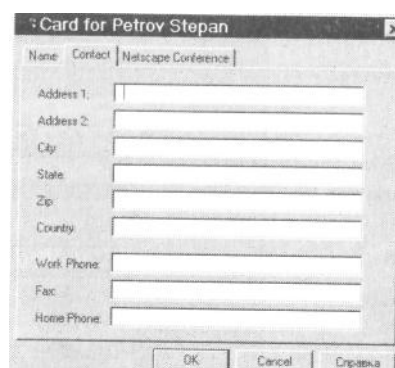


Рисунок 5.23. Лист **Contact** вкладки **New Card**

ПРАКТИКУМ: ТЕХНОЛОГИЯ ПРИМЕНЕНИЯ ЭЛЕКТРОННОЙ ПОЧТЫ

Цель работы: овладеть навыками применения специализированных программ (браузеров) для обращения к различным ресурсам Интернета; изучить основные приемы регистрации, настройки и поиска информации при использовании браузера Netscape Communicator; применить методы передачи информации в Интернете с помощью сервиса Электронная почта и др.

1. Подключение к Интернету с помощью браузера Netscape Communicator.

1.1. Вызовите программу регистрации пользователя браузера:

а) в меню **Пуск** выберите опцию **Программы**, из меню второго уровня выберите строку **Netscape Communicator**;

б) на следующей панели меню перейдите на строку **Утилиты**, далее на новой панели меню — на строку **Диспетчер пользователей**;

в) в диалоге диспетчера щелкните по кнопке **Создать**.

Диалог, который появляется на экране после нажатия кнопки **Создать**, позволяет записать полное имя файла настроек пользователя (Profile name), что дает возможность разделить файлы настроек различных пользователей браузера.

1.2. Запишите информацию в профайл пользователя (определите профиль пользователя):

а) перейдите к диалогу **Профиль пользователя** к окну записи имени пользователя и в поле **Полное имя** запишите свою фамилию и инициалы или условное имя, например Student№ (№ — номер компьютера в аудитории), а в поле **Адрес** укажите, например: USER№@ibm.bmstu.ru (где № — номер компьютера, остальное — адрес сервера факультета-провайдера)*, щелкните по кнопке **Далее**;

* Название профиля устанавливается провайдером. Здесь и далее используются имена, установленные для компьютерных классов ф-та ИБМ МГТУ им. Н.Э. Баумана.

б) в новом диалоге в поле **Название профиля** запишите USER№, на нижней строке указан адрес, куда будет помещен профайл пользователя. В каталог будут направлены новости и текущая корреспонденция (этот адрес следует запомнить), щелкните по кнопке **Далее**;

в) в следующих диалогах ознакомьтесь с ранее введенной информацией и покажите адрес сервера исходящей почты: ibm.bmstu.ru, укажите протокол обмена: POP (Post-Office Protocol), пролистайте все диалоги и нажмите на кнопку **Готово**.

*Система запустит браузер и автоматически обратится к домашней странице. Щелкните по кнопке **Стоп**, чтобы остановить браузер!*

2. Изучение методов работы информационной службы Интернета Электронная почта.

2.1. Вызовите почтовую программу **Messenger Mailbox** для записи нового письма:

а) в правом нижнем углу экрана под рабочим окном находится статусная строка, где следует щелкнуть по соответствующей пиктограмме;

б) в диалоге контроля входа запишите пароль: pdd№ (номер определяет преподаватель)*;

* Пароль установлен для компьютеров, размещенных в компьютерных классах ф-та ИБМ МГТУ им. Н.Э. Баумана.

в) ознакомьтесь с системой сформированных браузером почтовых ящиков.

г) на панели инструментов выберите кнопку **Новое письмо**;

д) ознакомьтесь с основными кнопками панели управления текстовым редактором почтовой программы.

2.2. Подготовьте текст письма:

а) запишите данные для заголовка (обязательных реквизитов) письма: введите в поле **To:** адрес получателя письма: USER(№+1). Если необходимо послать копию послания по другим адресам, то используйте адрес USER(№+2);

б) запишите тему письма в поле **Тема**, например test_;

в) установите кегль и начертание шрифта текста письма, размещение текста по полю листа с помощью кнопок панели форматирования;

г) подготовьте текст сообщения о товарах и услугах, предлагаемых Вашей фирмой.

2.3. Передайте и обработайте почтовые сообщения:

а) отправьте сообщение, щелкнув по кнопке **Отправить**;

б) в текстовом редакторе Microsoft Word подготовьте небольшой прайс-лист о предлагаемых товарах и услугах и направьте новое письмо с прикрепленным файлом по нескольким адресам;

в) отправьте по первому адресу новое письмо о планах на вечер текущего дня, запросив необходимость подтверждения факта доставки почтового сообщения;

г) просмотрите список новых писем, поступивших на Ваш почтовый ящик, с помощью кнопки **Просмотр** на панели почтовой программы;

д) одно из писем пометьте как непрочитанное, т.е. измените его статус;

е) отметьте одно из писем как наиболее важное — щелкните по полю, где изображен красный флажок.

- 2.4. Сформируйте адресную книгу. Для этого выполните следующую последовательность действий:
- а) в меню **Компоненты** щелкните по опции **Адресная книга**;
 - б) щелкните по кнопке **Новая запись** и внесите данные о Ваших корреспондентах (три записи). Заполните первые два листа вкладки для каждого нового адреса;
 - в) выберите одного из корреспондентов в адресной книге и пошлите ему ответ. Обратите внимание на автоматическое заполнение адресных строк письма!
 - г) дополните записи адресной книги новыми сведениями об адресате из полученной почты (из поля редактора перенесите запись e-mail адреса в адресную книгу).

2.5. Удалите содержимое всех Ваших почтовых ящиков.

3. Использование сервиса электронной почты при обращении к специализированным почтовым системам, например mailto.ru, или к серверам больших поисковых систем.

3.1. Перейдите на почтовый веб-сервер (например, mail.ru или найдите соответствующую услугу в поисковых системах типа Rambler.ru, yandex.ru и т. п.) и организуйте свой личный почтовый адрес (самостоятельно).

3.2. Передайте контрольное сообщение своим коллегам, используя подготовленный личный почтовый адрес, и получите ответ.

3.3. Снабдите новое письмо цветной подложкой.

4. Закройте Netscape Communicator и снова с помощью меню Пуск войдите в диспетчер пользователей для того, чтобы удалить свой файл идентификации.

5. Скопируйте текстовый файл на дискету.

6. Проверьте дискету на вирусы.

5.4. Работа с веб-документами

Активное использование информационной сети Интернет в различных областях деятельности общества привело к развитию одной из основных ее служб — World Wide Web (WWW). Так, нередко, говоря о работе в Интернете, предполагают использование только ее WWW-службы. Важно отметить, что возможности Интернета более широкие.

Здесь находят применение такие службы (сервисы), как электронная почта, телеконференции, системы поиска и доставки файлов, объединенные единым программным обеспечением.

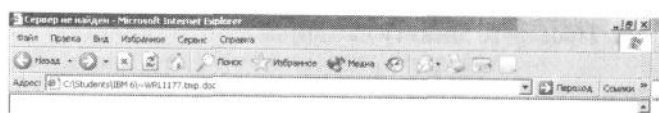
В основе всемирной паутины (WWW) лежит идея формирования документа в форме гипертекстовой структуры — это средство поддержки системы гипертекста в Интернете.

Гипертекст сегодня обозначает форму организации документа, текст которого может содержать ссылки к отдельным словам, фразам, рисункам и другим элементам текста, что позволяет получить дополнительную информацию об этих элементах текста. С помощью ссылок на отдельные страницы текста связываются между собой многие документы, размещенные на различных сайтах, образуя гипертекстовый документ. Таким образом, гипертекстовый документ состоит из набора текстовых фрагментов, графики, выделенных элементов документа и системы связи выделенных элементов с другими документами.

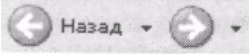
Благодаря своему свойству создавать документы с ассоциативными связями, отражающими индивидуальность каждого конкретного автора, WWW-сервис стал определять лицо Интернета и, более того, стал доминантой при использовании других служб Интернета: электронная почта, телеконференции и др.

5.4.1. Методы доступа к веб-документам

Для знакомства с методами доступа к информации, размещенной на серверах Интернета, используем браузер **Internet Explorer**. На рис. 5.24 показаны основные элементы рабочего окна обозревателя: строка заголовка окна, системное меню, кнопочная панель инструментов, адресная строка, рабочее поле браузера и др.



На кнопочной панели инструментов находятся следующие команды:



— перемещение **Назад и Вперед** по документам, вызванным на поле браузера в текущем сеансе работы. Кнопки **Назад и Вперед** позволяют постранично переходить от одной страницы к другой в обратном и прямом направлениях соответственно. Кнопки находятся на панели инструментов браузера. Они связаны со списком Журнал для идентификации связей назад и вперед, их можно реализовать, воспользовавшись соответствующими опциями меню **Избранное**.



— остановка процесса доступа к документам Интернета;



— обновление содержания ранее прочитанного документа;



— домашняя страница браузера (**Домой**);



— переход к поисковой странице браузера (**Поиск**);



— активизация списка избранных адресов Интернета;



— медиа-адреса, содержащие медиа-документы;



— список посещавшихся веб-адресов Интернета;



— список команд, использующих электронную почту: электронные письма, телеконференции и др.;



— печать веб-документа;



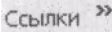
— правка веб-документа;



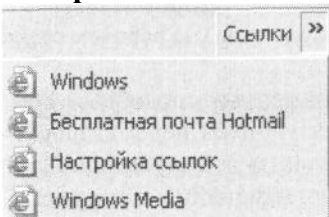
— вызов сервера обсуждения (**Обсудить**);



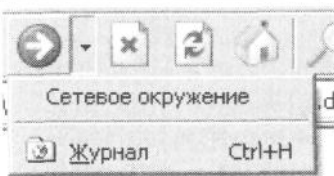
Рядом с адресной строкой размещена кнопка **Переход**, позволяющая осуществить переход по адресу, установленному в поле адресной строки.



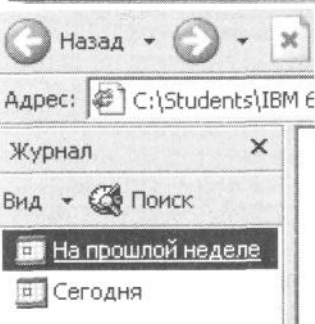
Кнопка **Ссылки** открывает следующий список: **Windows, Бесплатная почта Hotmail, Настройка сbsuioKnWindows Media.**



Ссылки (быстрые связи) помогают оперативно обратиться к веб-страницам сети. Для их использования следует сначала указать необходимые адреса на панели быстрых связей, а затем выбрать одну из них, раскрывая необходимые списки быстрых связей.

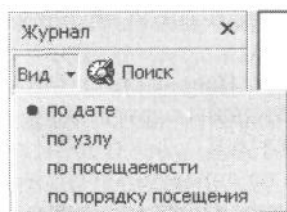


В левой части рабочей области браузера размещена область задач, на которой, в частности, с помощью команд можно обратиться к сетевому окружению или посмотреть на список недавних веб-контактов с помощью команды **Журнал**. Эту папку можно также вызвать комбинацией клавиш **<Ctrl>+<H>**.



Журнал позволяет формировать записи об Интернет-контактах на определенном временном участке, например **На прошлой неделе**, или текущие контакты с помощью гипертекстовой ссылки **Сегодня**.

Записи журнала можно ранжировать по дате, по узлу, по посещаемости и по порядку посещения, используя раскрывающийся список кнопки Вид.

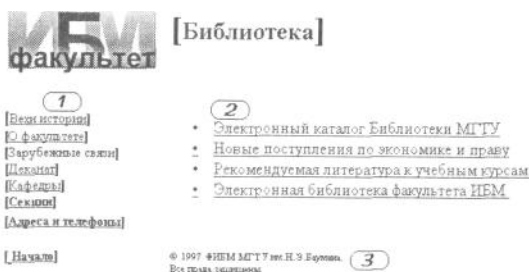


Журнал также содержит возможность активизации инструментов поиска документов в веб-пространстве с помощью кнопки **Поиск**.

Браузер позволяет реализовать доступ к веб-документам несколькими способами:

1. Щелкнуть мышью по связи в поле текущего отображаемого документа.
 2. В адресной строке записать URL-адрес нового узла (документа).
 3. Выбрать URL из списка узлов, которые посещались ранее.
 4. Выбрать URL из личного списка.
 5. Щелкать по кнопкам перемещения **Вперед и Назад** для перехода к документам, которые изучались в текущем сеансе.
 6. Щелкнуть по одной из быстрых связей.
 7. Дважды щелкнуть по ярлыку веб-страницы (узла) на рабочем столе или в меню **Пуск** и др.
- Рассмотрим методику реализации перечисленных способов.

Щелчок по связи текущего документа обеспечивает переход на следующую страницу гипертекста. Элемент связи на веб-странице выделен с помощью подчеркивания (рис. 5.25) или изменения цвета текста. Если графический элемент документа связан с другим веб-документом, то он имеет цветную окантовку.



- 1 — меню страницы, реализованное с помощью связей; 2 — список в тексте с использованием связей, позволяющий перейти на другие страницы документа; 3 — текст документа без связей

Рисунок 5.25. Фрагмент веб-страницы библиотеки факультета ИБММГТУ им. Н.Э. Баумана

Состояние активизации связи определяет вид курсора. В активном состоянии связи курсор принимает вид указывающего пальца. После активизации элемента связи (отдельное слово, фраза и т.п.) цвет текста связи может измениться, что при возвращении на текущую страницу позволяет определить ранее просмотренные связи.

Связи документа могут находиться в поле меню (рис. 5.25, 1), быть установлены в тексте документа (рис. 5.25,2), соединены с другими элементами документа (например, рисунок, фото и т. п.).

На рис. 5.25 показана возможность перехода на справочную систему **Электронная библиотека ф-та ИБМ**, где размещена рекомендуемая литература к учебным курсам (если щелкнуть по соответствующей строке). При необходимости перейти на другие страницы документа, которые перечислены в левом краю листа, следует щелкнуть по соответствующему заголовку. Например, для перехода в начало документа надо щелкнуть по связи **Начало**.

Переход по связи отражен в строке состояния (статусная строка), где появляется адрес нового документа.

Набор адреса документа (URL-узла), сопровождаемый щелчком по строке адреса, позволяет посетить любой узел Интернета, не покидая текущий документ. Для этого необходимо в адресной строке набрать URL-адрес узла, файла или службы и щелкнуть мышью.

Согласно протоколу HTTP каждый файл на любом компьютере, подключенном к Интернету, идентифицируется с помощью уникального кода URL (Uniform Resource Locator), который определяет тип файла и его местонахождение в файловом пространстве. Формат записи URL-адреса следующий:

тип ресурса://адрес/путь/имя_файла.расширение.

Поле тип ресурса показывает тип службы;

адрес/путь — каталог компьютера, в котором размещен искомый документ;

имя_файла.расширение — имя искомого файла.

Если путь и имя файла не указаны в строке адреса, то браузер перейдет на домашнюю страницу.

Если протокол URL не задан, то браузер автоматически добавляет http://, что определяет необходимость поиска веб-страницы. Если необходимо найти документ другой службы Интернета, следует ввести полный URL-адрес.

Для вызова документов, расположенных в текущем компьютере, на дискете, CD-ROM или в локальной сети, надо записать полное имя файла, обращая внимание на то, что при записи адреса в файловой системе используется левый слеш, вместо правого слеша, который применяется для записи URL-адреса. Например, если документ находится на диске компьютера, следует записать:

c:\example\student\zoya.htm ,

но если обращение направлено к веб-документу студента с другого файлового сервера, адресная строка должна содержать URL-адрес документа. В этом случае необходимо записать адрес, используя правый слеш, например:

http://www.ibm.mstu.ru/example/student/zoya.html

Для вызова документа, размещенного на дисковом пространстве компьютера, можно также использовать диалог **Открытие документа**, который вызывается с помощью опции **Открыть** меню **Файл** (рис. 5.26).

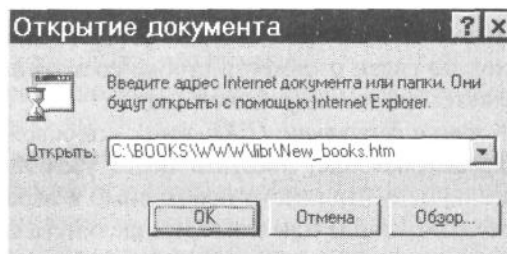


Рисунок 5.26. Диалог **Открытие Документа**

В диалоге можно сразу записать полное имя файла или воспользоваться кнопкой **Обзор** для перехода в режим программы проводника.

Вызов ранее посещавшихся веб-страниц информационной сети можно также осуществить, раскрыв список адресной строки и щелкнув по соответствующей строке списка адресов, так как браузер сохраняет копии веб-страниц в специальной папке.

Этот способ эффективен в тех случаях, когда есть уверенность, что оригиналы веб-страниц не изменялись во времени, а на компьютере есть место для хранения достаточно большого числа веб-страниц.

Управление обозревателем Интернета осуществляется с помощью опций, размещенных на листах вкладки **Свойства обозревателя**, которая вызывается с помощью меню **Сервис**.

Вкладка **Свойства обозревателя** содержит следующие листы: **Общие**, **Безопасность**, **Конфиденциальность**, **Содержание**, **Подключения**, **Программы** и **Дополнительно**.

Для исключения негативных явлений следует вызвать листы **Безопасность** и **Конфиденциальность** и на их полях определить содержание соответствующих настроек.

Для настройки функциональных возможностей обозревателя следует использовать лист **Общие** и лист **Программы**.

С помощью опций листа **Общие** (рис. 5.27) можно установить адрес домашней страницы. Для этого следует перейти на соответствующую область листа и в поле окна текстовой строки **Адрес** записать URL-адрес веб-сервера. Домашний адрес можно установить другим способом: вызвать на рабочее поле обозревателя соответствующий документ, перейти на лист **Общие** вкладки **Свойства обозревателя** и щелкнуть по кнопке **С текущей**. Для установки домашней страницы можно использовать и кнопку **С исходной**. Если необходимо всегда начинать с пустой страницы, то следует щелкнуть по кнопке **С пустой**.

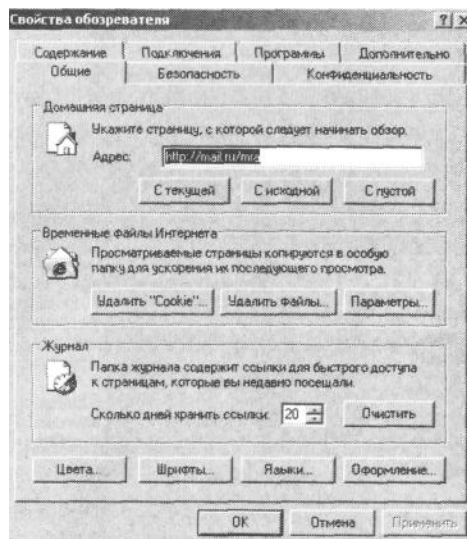


Рисунок 5.27. Лист **Общие** вкладки **Свойства обозревателя**

Кроме того, лист **Общие** дает возможность проводить обновление страниц, находящихся в особой папке. Для этого следует перейти в область **Временные файлы Интернета**, размещенную в центре листа **Общие**. Здесь просматриваемые файлы копируются в особую папку для ускорения их последующего просмотра. При активизации параметра обновления в процессе каждого сеанса браузер будет обновлять копию ранее посещавшихся веб-страниц. Область имеет три кнопки управления: **Удалить «Cookie»**, **Удалить файлы** и **Параметры**.

Третья область листа **Общие** позволяет определить объем папки журнала, который содержит ссылки к ранее посещавшимся веб-адресам. Так, на рис. 5.27 установлен показатель, позволяющий хранить информацию в течение 20 дней. Щелчком по кнопке **Очистить** все записи удаляются из папки **Журнал**.

Лист **Программы** вкладки **Свойства обозревателя** позволяет установить приложения, которые автоматически (по умолчанию) будут подсоединены к браузеру для работы с соответствующей службой Интернета (рис. 5.28).

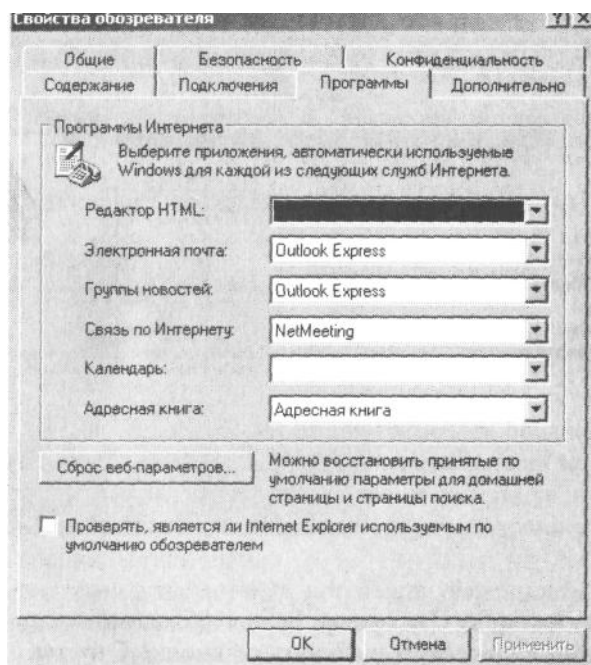


Рисунок 5.28. Лист **Программы** вкладки **Свойства обозревателя**

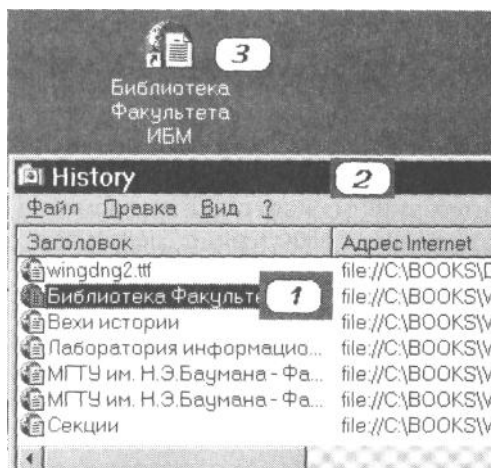
В области **Программы Интернета** следует указать наименования программных приложений, используемых для установки: редактора HTML, электронной почты, группы новостей, связи по Интернету, календаря, адресной книги. При необходимости восстановления принятых по умолчанию

параметров домашней страницы и страницы поиска следует щелкнуть по кнопке **Сброс веб-параметров**.

Выбор документа из списка ранее полученных документов можно выполнить с помощью списка Журнал (в Windows 9x – это History), который вызывается с помощью соответствующей кнопки, размещенной на панели инструментов (рис. 5.29).

Щелчком по выделенной строке (см. рис. 5.29, 1) можно вызвать соответствующий документ.

Для вызова документов, к которым необходимо систематическое обращение, на рабочем столе или в поле пускового меню следует установить соответствующие ярлыки. Для этого необходимо захватить ярлык, находящийся в левой части поля заголовка URL-адреса, и перетащить его на поле рабочего стола (см. рис. 5.29, 3).



1 — строка URL-адреса; 2 — строка заголовка; 3 — ярлык адреса на рабочем столе

Рисунок 5.29. Фрагмент диалогового окна **History**

Вызов документа с помощью ярлыка осуществляется двойным щелчком по соответствующему значку, ранее установленному, например, с помощью вкладки Журнал

5.4.2. Установка страницы поиска

Использование средств поиска предполагает обращение к специализированным страницам Интернета и других сетей. На этих страницах, как правило, установлены специальные окна для ввода ключевых слов, которые могут встретиться в URL-адресах, и кнопки запуска операции поиска. В случае успешного поиска можно перейти на найденные веб-страницы, для возврата — использовать кнопку **Назад** панели инструментов браузера.

Каждый пользователь может установить свою исходную систему поиска информации. Такая операция определяется как установка страницы поиска. Эта страница будет вызвана в окно браузера при нажатии на кнопку **Поиск** панели инструментов браузера. Страницу поиска можно заменить.

Для установки или изменения страницы поиска следует использовать возможности вкладки **Свойства обозревателя**.

5.4.3. Копирование веб-страниц

В целях сокращения времени сеанса связи веб-страницы целесообразно их копировать на диск и читать после отключения от Интернета. Такой метод работы с веб-документами достаточно эффективен, если заранее известны искомые документы: объявления, газеты, журналы и другие интерактивные источники информации.

Для копирования веб-страниц следует выполнить следующие операции:

- ◆ предусмотреть в файловой системе компьютера установку рабочего каталога (папки), куда будут направлены копии документов. Эту операцию следует выполнить заранее, установив папку в личном каталоге, например: student\new_web_1;
- ◆ перейти на искомую веб-страницу;
- ◆ открыть меню **Файл** и активизировать опцию **Сохранить как файл**;

- ◆ с помощью диалогового окна отыскать необходимую папку, например student\new_web_1, записать в строке имени файла имя файла, в котором будет храниться текущая веб-страница, установить расширение файла HTML(*.htm, *.html) с помощью раскрывающегося списка расширений файла; » щелкнуть по кнопке **Сохранить**.

Следует учесть, что при копировании веб-страницы запоминается лишь текст и информация о компоновке html-страницы. Графический материал, аудиовизуальные фрагменты и другие шаблоны находятся в других файлах и поэтому не копируются.

Для запоминания, удаления, копирования и поиска отдельных фрагментов веб-страницы следует воспользоваться командами контекстного меню **Правка**: Вырезать, Копировать, Вставить, Выделить все, Найти. Для выполнения этих команд следует выделить с помощью мыши соответствующий фрагмент документа. Если необходимо выделить весь документ, то следует воспользоваться командой Выделить все. После выделения участка или всего документа необходимо щелкнуть правой кнопкой мыши и использовать одну из перечисленных команд:

- ◆ **Вырезать** — удаляет выделенный фрагмент документа с экрана и помещает его в буфер обмена. Следует учесть, что сам документ не изменяется;
- ◆ **Копировать** — помещает копию выделенного фрагмента документа в буфер обмена, откуда его можно скопировать в любой другой документ, подготовленный с помощью программ Microsoft Office;
- ◆ **Вставить** — перемещает содержимое буфера обмена в текстовое поле документа;
- ◆ **Найти** — используется для поиска слов, фраз, символов в текущей веб-странице. Применяется, как правило, в тех случаях, когда текст документа не помещается на экране.

Для просмотра ранее скопированных веб-страниц необходимо с помощью **Проводника** найти соответствующий файл в файловом пространстве диска и двойным щелчком вызвать страницу. Одновременно будет вызван на рабочий стол и браузер.

Для перехода на другую, ранее скопированную страницу следует в меню **Файл** активизировать опцию **Открыть**. С помощью диалогового окна указать новое имя открываемого файла и вызвать соответствующую страницу на рабочий стол.

5.5. Участие в телеконференциях

Телеконференции в Интернете реализуются с помощью службы **Internet News**. Они направлены на быстрое распространение информации о событиях в мире, бизнесе, науке, культуре и др. Каждая тема обсуждается в отдельном потоке сообщений, называемом телеконференцией или группой новостей.

В большинстве телеконференций можно опубликовать сообщение, которое автоматически рассылается другим подписчикам. Таким образом реализуется интерактивное сообщество пользователей. Последнее накладывает определенные нравственные и этические обязанности на участников телеконференций. И так как эти требования для достаточно большого числа пользователей весьма относительно, то уровень «шума» во многих телеконференциях очень высок.

5.5.1. Структура службы

Каждая телеконференция имеет уникальное имя, которое определяет тему обсуждения и общую категорию, или домен, который включает тему (например, телеконференция comp.hard).

Темы могут быть разделены на подтемы и т. д.

Наиболее распространенными телеконференциями являются:

comp — компьютерные проблемы; news — новости конференции Usenet;

sci — наука и техника;

talk — обсуждения;

misc — разные проблемы;

biz — бизнес, реклама.

Процедура участия в телеконференции состоит в подготовке статьи для телеконференции (или ответа на прочитанную информацию), отправке ее на News-сервер, где статье присваивается уникальный номер и где она рассылается по всем адресам участников телеконференции.

Как правило, участие в телеконференциях платное и требует внесения абонентской платы, однако в отдельных сетях имеют место бесплатные конференции. Чаще это относится к корпоративным информационным сетям.

Для чтения статей телеконференции необходимо с помощью программы чтения новостей выполнить следующие процедуры:

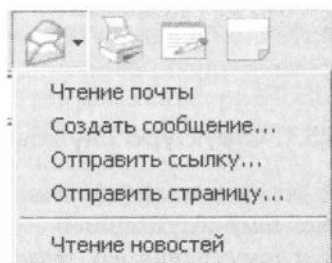
- ◆ загрузить список телеконференций;
- ◆ выбрать необходимую телеконференцию;
- ◆ загрузить список доступных в телеконференции статей;
- ◆ выбрать и загрузить отдельные статьи.

Практически все News-серверы используют стандартный сетевой протокол передачи новостей (Network News Transfer Protocol, NNTP) для обмена статьями с клиент-программой.

5.5.2. Чтение новостей

Программа **Internet News** интегрирована в браузер, и ее можно запускать с помощью команды **Чтение новостей**, размещенной на раскрывающемся списке кнопки **Почта**.

При первом обращении к службе на экране будет установлено окно **Обзор папок**, с помощью которого необходимо установить имя папки и ее местонахождение в файловой системе компьютера. Здесь следует установить папку типа **Mu_news** в личном каталоге, например **Student**. Затем программа переходит к настройке **Internet News**.



В первой строке на поле первого окна настройки следует указать имя пользователя, которое впоследствии будет подставляться в конец всех его сообщений (рис. 5.30).

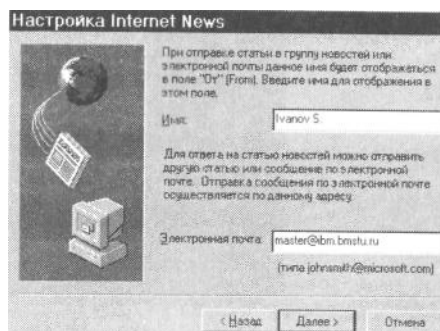


Рисунок 5.30. Первый диалог настройки **Internet News**

На второй строке необходимо записать электронный адрес пользователя (группы пользователей). Адрес задается заранее провайдером службы.

В строке **Сервер новостей**, расположенной во втором диалоге настройки (рис. 5.31) следует записать точное имя News-сервера. Впоследствии его можно заменить на другое имя. Если провайдер требует послать на News-сервер имя пользователя и пароль, то следует щелкнуть по кнопке **Требуется вход на сервер**. В этом случае открываются две строки: **Учетная запись** и **Пароль**, которые необходимо заполнить.

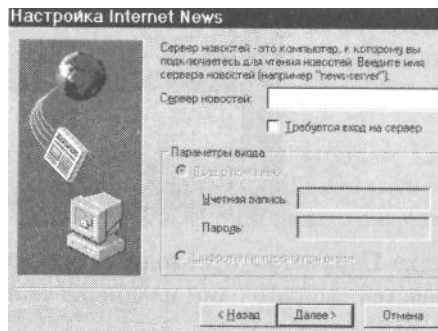


Рисунок 5.31. Второй диалог настройки **Internet News**

После установки имени News-сервера на экран вызывается третий диалог, в котором необходимо указать способ подключения к серверу новостей: с помощью локальной сети, вручную, с помощью модема. Если работа осуществляется в компьютерном классе, то следует воспользоваться первым способом — **С помощью локальной сети** (рис. 5.32).

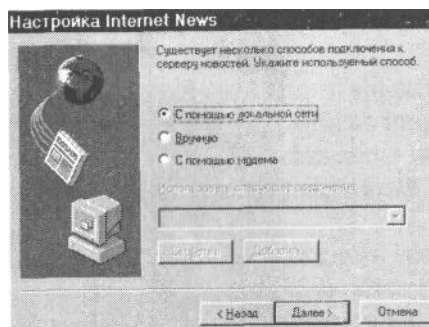


Рисунок 5.32. Третий диалог настройки **Internet News**

Для изменения параметров сервера новостей можно использовать вкладку **Параметры** (рис. 5.33), которая вызывается с помощью опции **Параметры** меню **Новости**, размещенном на окне **Internet News** (рис. 5.34).

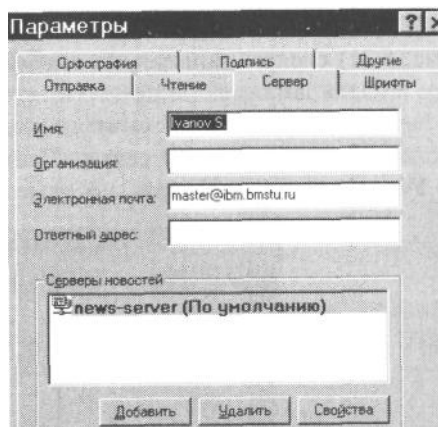


Рисунок 5.33. Лист **Сервер** вкладки **Параметры**

На вкладке **Параметры** необходимо вызвать лист **Сервер**, а в окне **Серверы новостей** выбрать один из списка серверов.

5.5.3. Чтение статей телеконференций

Если щелкнуть по кнопке **Новости** браузера, то на экране будет установлена страница сервера новостей фирмы Microsoft.

Рациональнее для перехода к программе чтения новостей щелкнуть по кнопке **Почта** панели инструментов браузера. В раскрывшемся списке выбрать строку **Чтение новостей**.

В этом случае на экране устанавливается окно **Internet News**, показанное на рис. 5.34. Окно имеет стандартные органы управления: системная строка, командное меню, панель инструментов, адресная строка, рабочее окно и др.

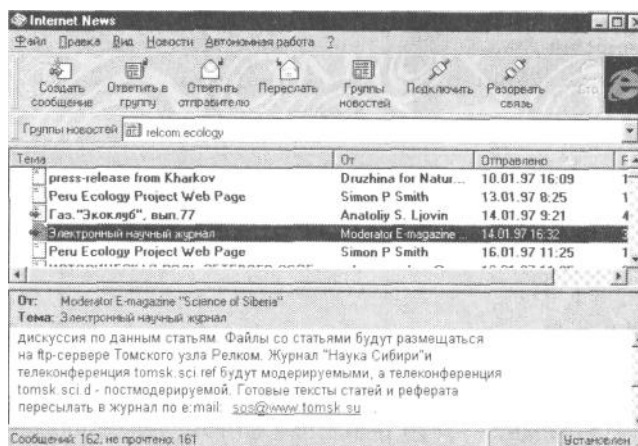


Рисунок 5.34. Окно **Internet News**

Если обращение к телеконференциям осуществляется впервые, то верхняя и нижняя части рабочего окна будут пустыми. Однако на практике в рабочем поле окна новостей показывается список документов той конференции, к которой было осуществлено обращение в последнем сеансе. Так, на рис. 5.34 показан список журналов (телеконференций) *relcom.ecology*.

Для изменения группы новостей следует щелкнуть по кнопке **Группы новостей** панели инструментов программы. В этом случае на экране размещается окно **Группы новостей**, к которым можно обратиться с компьютера пользователя (рис. 5.35).

В верхней части окна находится строка, позволяющая ускорить поиск группы новостей (телеконференции). Если достаточно просмотреть список с помощью линеек прокрутки, то эту строку можно не заполнять.

После установки курсора на необходимом адресе — теме конференции следует щелкнуть по кнопке **Переход**. На экране компьютера снова будет установлено рабочее окно **Internet News**.

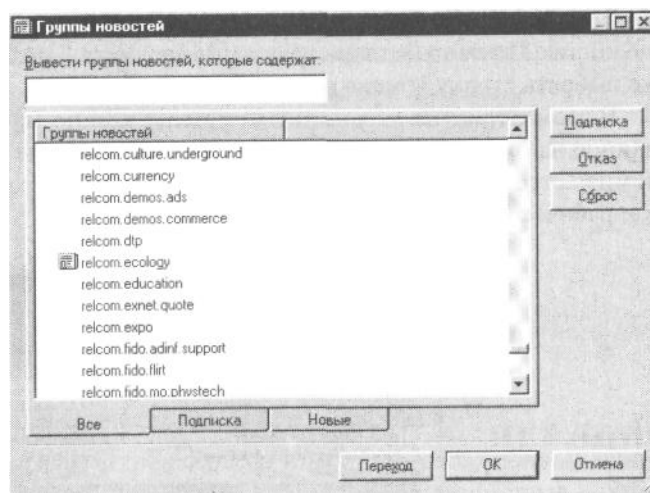


Рисунок 5.35. Окно **Группы новостей**

5.5.4. Просмотр документов телеконференций

Рабочее окно **Internet News** состоит из двух частей: верхней и нижней.

В верхней части размещен список телеконференций по именам в алфавитном порядке.

В нижней части показан текст документа. Под рабочим окном находится строка, отражающая общее число документов и число прочтенных среди них.

Для доступа к документам в адресной строке должно быть установлено имя конференции, в верхней части выделено имя документа, а в нижней части после щелчка мышью по полю заголовка документа – непосредственно текст документа.

В ряде случаев, например при использовании русского алфавита, текст документов трудно прочитать. Для выхода из этого положения следует воспользоваться опцией **Набор символов** меню **Вид**. На открываемом листе щелкнуть по строке **Кириллица (Windows)**.

Если необходима установка шрифта для текста заглавий конференций, то следует воспользоваться меню **Новости**, в котором активизировать опцию **Параметры**. На вкладке вызвать лист **Шрифты** и перейти к изменению шрифта (кнопка **Изменить**). Установить набор символов (кириллица) и щелкнуть по кнопке **ОК**, а затем по кнопке **Применить** и снова **ОК**.

5.5.5. Отправка документов на телеконференцию



Для отправки документов на телеконференцию следует щелкнуть по кнопке **Создать сообщение**, расположенной на панели инструментов **Internet News**. На рабочий стол устанавливается окно, позволяющее послать документ на телеконференцию. Бланк письма показан на рис. 5.36.

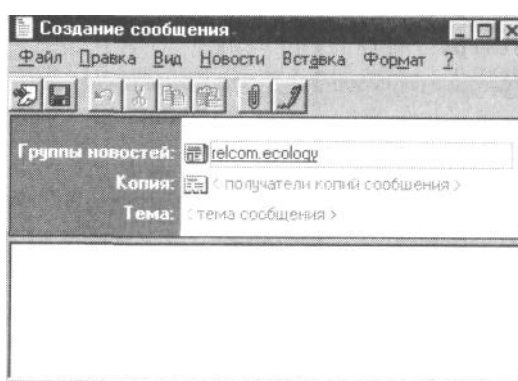


Рисунок 5.36. Окно Создание сообщения

Бланк состоит из поля заголовка и поля текста сообщения. В поле заголовка на первой строке следует записать адрес новостей. Если сообщение направляется на конференцию, с документами которой осуществлялась работа, то это имя устанавливается автоматически. На второй строке указываются адреса получателей копии документа. Эта строка может остаться пустой. На третьей строке записывается тема сообщения (1—3 слова). В поле письма записывается текст сообщения.

При составлении сообщения следует придерживаться правил работы с электронной почтой.

Если необходимо послать ответ на опубликованный документ (сообщение, возражение, мнение и т. п.), то следует щелкнуть по кнопке **Ответить отправителю**. В этом случае на рабочий стол устанавливается бланк ответа, приведенный на рис. 5.37.

В строке заголовка записывается имя документа, автору которого пишется сообщение. Заголовок ответа формируется на основе данных рецензируемого документа.

В поле послания устанавливаются первые строки документа (в прямоугольных скобках).

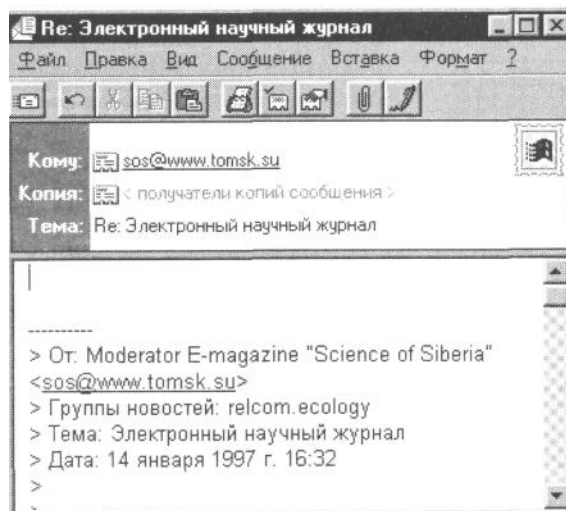


Рисунок 5.37. Бланк сообщения на адрес автора прочитанного документа

Первая строка предназначена для записи формы послания (ответ, сообщение и т. п.).

В подготовленном бланке находится весь текст документа. В процессе записи сообщения следует удалить лишние строки, оставляя лишь напоминание о том документе, на который пишется реплика.



При пересылке сообщения можно воспользоваться услугами электронной почты. Для этого необходимо щелкнуть по кнопке **Переслать**. В результате появится поле для нового заголовка с пустыми строками адреса назначения послания. Следует заполнить пустые строки и отослать сообщение.

5.6. Поисковые системы и каталоги ресурсов

В настоящее время существует несколько мощных отечественных поисковых систем, ориентированных на работу с русским текстом и индексирование российского веб-пространства, а также выполняющих поиск в архивах русскоязычных телеконференций. К числу ведущих российских систем относят: Rambler (www.rambler.ru), Russian Internet Search (www.search.ru), Русская машина поиска (search.interrussia.com), TELA (tela.dux.ru), система поиска в телеконференциях relcom (www.dux.ru/news.html) и др.

В русском секторе Интернета имеются и свои каталоги. Их структура состоит из тематических разделов и в ряде случаев — подразделов, записи которых содержат обязательные сведения о наименовании веб-сайта, его URL-адресе и присутствующие в ряде каталогов сведения о городе и стране. Некоторые каталоги имеют краткие описания каждого информационного ресурса и возможности поиска сведений по ключевым словам в названиях, адресах и описаниях, что повышает ценность таких тематических подборок ресурсов.

5.6.1. Желтые страницы российского сегмента Интернета

В качестве примера функционирующих российских WWW-серверов приведем информацию об отдельных организациях, обслуживающих соответствующий сервер и специализирующихся в области обработки и передачи информации в сфере производства и торговли. Их общее число достаточно велико, поэтому сначала укажем на возможную их классификацию:

- ◆ Базы данных и информационные системы;
- ◆ Предметы потребления и бытовая техника;
- ◆ Продовольственные товары;
- ◆ Недвижимость и строительство;
- ◆ Транспорт и транспортные услуги;
- ◆ Промышленное оборудование;
- ◆ Машиностроение;
- ◆ Приборостроение;
- ◆ Энергетика и энергоресурсы;

- ◆ Маркетинг и реклама;
- ◆ Рынок труда.

Базы данных и информационные системы

Базы данных на сервере компании «Демос». Предоставляет он-лайнный доступ к большому числу баз данных различного тематического содержания. Имеет разделы: Коммерция (московские цены на компьютерную и бытовую технику), Товары и цены, Услуги и цены (оба каталога являются электронными версиями соответствующих печатных изданий), Информационные бюллетени агентства «Полином» (коммерческие предложения фирм по текущим прайс-листам). Часть информации имеет свободный доступ.

<http://dbs.demos.su>

Бесплатные коммерческие объявления на сервере Simplex. Имеет два раздела: коллекции ссылок по темам литературы и прессы и электронная доска бесплатных коммерческих объявлений. Объявления сгруппированы по темам в следующие разделы: Продукты питания, напитки, табачные изделия; Металлы, сырье химические товары; Машины, транспортные услуги, экспедиции; Одежда, потребительские товары; Лекарства, медицинское оборудование; Электротовары, электроника; Деловые контакты, совместные предприятия; Международные поиски и предложения работы; Разное.

<http://www.simplex.ru>.

Бюллетень «Прайс-лист». Позволяет получить обширную информацию по разным группам товаров и услуг, а также различным фирмам. Имеется гостевой вход, обеспечивающий ограниченный доступ к информации.

<http://www.relis.ru:8080/datumb/PLineR?LOGIN>

Международный информационный рынок. Содержит базы данных по различным коммерческим предложениям, правовым вопросам, фондовым рынкам и др. Бесплатно можно просмотреть рубрикаторы баз данных, а для получения полноправного доступа к информации необходимо пройти процедуру подписки и оплаты. На сервере имеется информационно-поисковая система «Артефакт», обеспечивающая полнотекстовый поиск.

<http://mir.glasnet.ru>

Сетевой торговый дом. Представляет собой тематический сайт, посвященный электронным методам организации торговли и объединяющий ряд виртуальных магазинов. Сервер реализован АО «Релком».

<http://www.com.ru>

Синие страницы России. Содержат электронную версию национального справочно-энциклопедического ежегодника. В разделе «Государственные, предпринимательские и финансовые структуры Российской Федерации» содержатся сведения о крупнейших предприятиях России.

<http://www.luepages.dux.ru>

Транспорт и транспортные услуги

Transport on Line. На сервере создана информационная система, охватывающая все аспекты перевозок: от оперативной информации по заявкам по сети электронного фрахта EFNet до периодических изданий (Морской флот, Море, Воздушный транспорт, Автомобильный транспорт) и справочной информации. Сервер имеет следующие разделы: Сеть электронного фрахта; Периодические издания; Реклама фирм; Официальная информация; Аналитические обзоры и статистика; Выставки; Конференции; презентации; Программное обеспечение; Страхование; Сети и системы связи и др. Каждый раздел имеет подразделы: Общий, Море, Авиа, Авто, Железнодорожный, <http://www.transport.ru>

5.6.2. Наука и техника

Развитию научных ресурсов Интернета в России уделяется большое внимание. По содержанию информационные системы, поддерживающие это направление, можно разделить следующим образом:

- ◆ Научно-исследовательские организации, фонды и информационные системы;
- ◆ Астрономия, авиация, космонавтика;

- ◆ Биология;
- ◆ Информатика и вычислительная техника;
- ◆ Математика и механика;
- ◆ Науки о земле;
- ◆ Приборостроение и электроника;
- ◆ Физика;
- ◆ Химия;
- ◆ Гуманитарные науки и экономика.

Большинство веб-серверов этого направления имеет следующее содержание: общая характеристика организации и направлений научной деятельности; структура института и страницы отделов и лабораторий; описание наиболее крупных проектов, перечень публикаций и др.

Следует заметить, что большое число научных ресурсов находится на серверах университетов, поэтому можно порекомендовать обратиться также к соответствующим серверам.

Научно-исследовательские организации, фонды и информационные системы

Государственная публичная научно-техническая библиотека. Сервер содержит электронный каталог библиотеки с возможностью поиска по ключевым словам, раздел «Каталоги и ресурсы библиотеки» содержит подборки ссылок на серверы Интернета и электронный вариант журнала «Научные и технические библиотеки», издаваемого в ГПНТБ.

<http://gpntb.ippi.ras.ru>

Служба Инфомаг. Информационная служба, созданная с целью распространения по информационным сетям библиографической информации, содержащей сведения об оглавлениях научных и технических журналов, зарубежных научных электронных бюллетеней, рефераты статей, электронные версии журналов и книг.

<http://www.ripn.net/infomag>

Астрономия, авиация и космонавтика

Российская научная космическая сеть (RSSI). Некоммерческая компьютерная сеть, объединяющая на добровольной основе имеющие мировую известность академические организации, научно-исследовательские центры и институты, медицинские учреждения, учебные и образовательные организации. На сервере представлены сведения о проекте RSSI, перечень организаций, сведения о загрузке канала ИКИ РАН -NASA и информация по профилю исследований.

<http://www.rssi.ru>

Институт космических исследований РАН. Сервер содержит информацию о российских и международных научных непилотируемых программах исследования космоса и программах мониторинга окружающей среды. Имеет отдел истории непилотируемой космонавтики, архивы космических и метеорологических данных.

<http://www.iki.rssi.ru>

Информатика и вычислительная техника

Вычислительный центр РАН. На сервере представлена структура института и направления исследования, схема компьютерной сети, FTP-архив с WWW-интерфейсом, с книгами, документацией, программным обеспечением для разных операционных систем. Имеются пособия для пользователей Интернетом.

<http://www.ccas.ru>

Институт автоматизированных систем. Институт является оператором сети ИАСНЕТ — сети передачи данных общего пользования на территории России и стран ближнего зарубежья. На его сервере представлены оборудование и программы типовых решений корпоративных сетей.

<http://www.iasnet.ru>

5.6.3. Образование

В российском разделе Интернета собрано большое число информационных ресурсов в области образования, ссылки на которые можно сгруппировать в следующих основных разделах:

- ◆ Информационные системы и справочники;
- ◆ Высшие учебные заведения;
- ◆ Школьное образование;
- ◆ Образовательные центры и проекты;
- ◆ Учебные материалы, компьютерные обучающие программы и дистанционное обучение.

Раздел, посвященный университетскому образованию, является наиболее обширным и содержит сведения почти о всех веб-серверах российских вузов. В качестве примера организации доступа к серверам в области образования покажем возможность обращения к справочной системе вузов России и к серверу МГТУ им. Н.Э. Баумана.

Информационные системы и справочники

База данных и справочник по вузам России. База данных «Вузы России» содержит по каждому вузу следующую информацию: адрес, сведения о руководстве, телефоны руководителей, численность студентов, ученые специальности, специальности кандидатских и докторских советов.

В электронном справочнике «Высшие учебные заведения Российской Федерации» собраны данные о государственных и негосударственных высших учебных заведениях, включая краткую характеристику вуза, направления подготовки кадров и научных исследований, его международные связи.

<http://www.informika.ru/windows/goscom/basevuz/global.html>

МГТУ им. Н.Э. Баумана. Сервер содержит сведения о факультетах и системе обучения в техническом университете.

<http://www.bmstu.ru>

Центры дистанционного образования

Институт дистанционного обучения. На этом сервере находятся ответы на вопросы о дистанционном обучении. Разъясняются его концепция, цели, задачи, нормативно-правовая база. На его страницах помещен перечень специальностей, по которым проводится подготовка, планы занятий с указанием количества часов и список экзаменов. На сервере публикуется журнал «Дистанционное образование», предлагающий статьи об организации учебного процесса, официальные документы, новости, материалы конференций и семинаров.

e-mail: www.ido.ru

Институт экономического развития. Содержит информацию о программах по информационным технологиям, разработывавшихся с участием Института экономического развития. Здесь можно ознакомиться с материалами семинара «Использование Internet в подготовке учебных материалов и в научных исследованиях», посмотреть содержание курса, презентации страниц участников семинара. Страница «Интерактивные программы для дистанционного обучения» предлагает самотестирование по персональным компьютерам, программному обеспечению, Интернету и WWW с наглядным представлением результатов, тесты по микро- и макроэкономике.

e-mail: www.edimo.ru

Центр «Информика». Сервер Центра информатизации Министерства образования РФ предлагает большой список ассоциаций и институтов, центров дистанционного образования в высших учебных заведениях, поисковую систему по свободно распространяемому программному обеспечению для дистанционного образования, а также содержит информацию по методическому обеспечению учебного процесса и сведения о различных проектах в сфере дистанционного обучения. Знакомит с двумя электронными курсами — по экономике и праву в России.

e-mail: www.informika.ru

Евразийская ассоциация дистанционного образования (ЕАДО). Создана для координации деятельности различных организаций, заинтересованных в развитии дистанционного образования. На сервере публикуются материалы конференций по дистанционному образованию, помещен устав организации. Кроме того, здесь собрана богатая коллекция ссылок на ресурсы по дистанционному

обучению в Интернете, а также список организаций, связанных с развитием дистанционного образования.

e-mail: www.dist-edu.ru

Центр дистанционного обучения МИЭМ. Проводит подготовку и переподготовку специалистов в системе дистанционного обучения по различным программам. Это и получение второго высшего образования, и профессиональная переподготовка, и обучение по отдельным курсам с выдачей сертификата МИЭМ, и обучение за рубежом в ведущих университетах Америки и Австралии. По каждому из направлений обучения на страницах сервера предлагается аннотация, краткое содержание, учебный план, условия приема и информация о стоимости обучения.

e-mail: dlc.miem.edu.ru

Проекты по дистанционному обучению группы «Махаон». Включает в себя компьютерный курс «Атомная энергетика и ее безопасность», разработанный группой «Махаон» совместно с несколькими другими научными учреждениями. Те, кто интересуется теоретическим аспектом проблемы дистанционного обучения, могут ознакомиться с материалами научно-методических конференций, посвященных проблемам образования и развития телекоммуникаций. Кроме того, на сервере собрана коллекция ссылок на ресурсы по дистанционному обучению и помещено несколько электронных тестов из различных областей знаний (английский язык, литература).

e-mail: Дистанционное обучение: www.machaon.ru/educ.html

Право и менеджмент. Интерактивные учебники и дистанционное обучение при подготовке специалистов в области права и управления бизнесом. На сервере размещены перечень специальностей, условия поступления и оплаты обучения. Курс по программе дистанционного обучения предполагает самостоятельную работу в течение семестра и посещение раз в год очных сессий длительностью 7—10 дней. После сдачи выпускных экзаменов и защиты дипломной работы студенты получают диплом.

e-mail: www.aha.ru/~hsk

Центр «Линк». Распространяет профессиональные образовательные программы, подготовленные Школой бизнеса Открытого университета Великобритании. В настоящее время предлагаются следующие программы, переведенные на русский язык и адаптированные к российским условиям: «Мастер бизнес-администрирования», «Профессиональный сертификат в области менеджмента» и «Профессиональный диплом в области менеджмента». Программы ориентированы на последовательное повышение профессионального и образовательного уровня менеджеров соответствующей квалификации. Обучение происходит по ступеням, и к изучению каждой следующей ступени программы допускаются только те, кто сдал экзамен за предыдущую. На сервере можно более подробно ознакомиться с программами, узнать их перечень, условия обучения. Здесь же помещена карта региональных учебных центров и представительств центра «Линк».

e-mail: www.link.msk.ru

«Основы работы в Интернете». Курс дистанционного обучения, предлагаемого Центром открытых систем КГТУ им. А.Н. Туполева. На страницах сайта помещена информация о преподавателях, которые будут работать с каждым из подписчиков индивидуально, отвечая на все вопросы по электронной почте.

e-mail: www.dlab.kiev.ua/dori/r100.htm, www.ospu.odessa.ua/dori/r100.htm

5.6.4. Интернет-магазин

Реализация интерактивного диалога, осуществленная в сети Интернет, привела к развитию новых форм информационного общения. Одной из них является новая форма обслуживания покупателей в магазинах. Ее основное отличие от традиционной заключается в том, что перечень предлагаемых для покупки товаров показывается на экране потенциального покупателя, выделенные товары из списка можно рассмотреть, вызывая их изображения. Далее, если товар необходим покупателю, он помечает их, на основе этих пометок формируется заказ, который передается в диалоговом режиме и хранится на сервере магазина в течение нескольких суток. Окончание операции традиционно, так как товар следует получить традиционным способом, осуществив вывоз его со склада магазина.

Интернет-магазин позволяет клиенту, вошедшему через Интернет на соответствующую страницу веб-сервера торговой компании, осуществить выбор товаров из перечня товаров, находящихся в наличии на складе, и их помещение в заказ. Товарное пространство для удобства ориентации клиента организовано в виде иерархического дерева.

При помещении товара в заказ происходит его автоматическое резервирование на некоторый фиксированный срок, продолжительность которого может зависеть от суммы заказа, категории клиента и предыдущей истории обслуживания клиента, которая хранится в информационном пространстве торговой системы компании.

После формирования заказа клиент либо авторизуется, если он уже пользовался ранее услугами Интернет-магазина компании, либо вводит о себе необходимую информацию, которая регистрируется в базе данных торговой системы.

После регистрации клиент может либо распечатать счет, либо, при отсутствии у него принтера, просто записать номер счета, который одновременно является номером заказа клиента.

Далее клиент имеет возможность приехать в магазин компании и, предварительно оплатив его, получить заказ, который зарезервирован на его имя.

При получении заказа клиенту сообщается его идентификационный код, который в дальнейшем должен им использоваться при авторизации в системе.

Другие формы обслуживания клиента (например, по кредитной карте) в настоящее время не всегда доступны не столько по техническим причинам, сколько по причине отсутствия юридической и финансовой базы для их реализации в России.

Чтобы просмотреть товары, представленные в Интернет-магазине, обычно следует в меню, расположенном на первой странице магазина, щелкнуть по кнопке **Добавить в заказ**. В отдельном окне появится список товаров, организованный в виде дерева. Группа товаров обозначается кружком, а сам товар нередко обозначается специальным значком. Товары, имеющиеся в данный момент на складе, выделены курсором одного цвета, а отсутствующие на складе — другого. Изображение товара передается на экран после нажатия кнопки **Осмотр....** Чтобы поместить товар в корзину покупателя, необходимо дважды щелкнуть мышью по его надписи или воспользоваться кнопкой **ОК**. Выбранный товар появляется в окне корзины.

Для изменения содержания товарной корзины необходимо щелкнуть по соответствующей строке мышью и набрать новую цифру в окошке **Количество**. Можно удалить строку заказа, нажав кнопку **Удалить позицию**.

После формирования товарной корзины следует нажать кнопку **Выполнить заказ**. В отдельном окошке появится бланк регистрации. Если клиент впервые пользуется услугами Интернет-магазина, то необходимо включить переключатель **Новый клиент** и заполнить поля левого столбца бланка, кроме поля Код. После заполнения полей надо нажать на кнопку **ОК**.

Если клиент не первый раз посещает магазин, он вводит только свой личный код Интернет-магазина.

Счет для оплаты заказа появляется в основном окне браузера. Его можно распечатать либо только записать его номер.

5.6.5. Виртуальные библиотеки

Многие библиотеки мира обеспечивают доступ к своим информационным ресурсам с помощью Интернета. Для этого создаются домашние веб-страницы библиотек, на которых размещаются списки различных форм информационного обслуживания абонентов библиотеки. Приведем основные из них:

- ◆ поиск библиографических описаний изданий, хранящихся в фондах библиотеки;
- ◆ текстовые файлы изданий библиотеки;
- ◆ списки рекомендуемой литературы;
- ◆ правила пользования библиотекой и режимы ее работы;
- ◆ информация о различных мероприятиях библиотеки: конференции, совещания и т.п.
- ◆ другая информация.

Такие веб-страницы характеризуют пассивный диалог между библиотекой и ее абонентом. Примером такой организации веб-страницы служит сайт Государственной публичной научно-технической библиотеки России (<http://www.gpntb.ru>) — ГПНТБ (рис. 5.38).



Рисунок 5.38. Фрагмент Web-страницы ГПНТБ России

Для получения информации следует щелкнуть мышью по одной из надписей или по выделенной связи рабочего поля страницы. Основным недостатком этой и подобных ей страниц крупных отечественных библиотек является то, что они закрывают доступ к поисковой системе, требуя предварительной регистрации (что означает, как правило, необходимость оплаты сеансов поиска информации).

Основным содержанием веб-страницы является ее поисковая система. В ГПНТБ переход на ее поле осуществляется после щелчка по строке **Каталоги и базы данных**. Она содержит следующие каталоги и базы данных:

- ◆ электронный каталог ГПНТБ России;
- ◆ новые поступления;
- ◆ авторефераты диссертаций;
- ◆ российский сводный каталог по научно-технической литературе;
- ◆ фонд алгоритмов и программ;
- ◆ кто есть кто в библиотечной сфере России.

Интерфейс поисковой системы ГПНТБ сформирован по примеру подобных систем, реализованных на компьютерах предыдущих поколений, когда простейший дизайн был продиктован слабыми изобразительными возможностями компьютерной технологии: на экран выведены поля основных поисковых признаков, что превращает форму в трудночитаемый документ, не способствующий доверительному диалогу.

5.7. Интернет-объединение информационных сетей России

Информационные сети России, использующие единый протокол семейства TCP/IP (семейство IP-сетей) и единое адресное пространство, образуют важную составляющую сети Интернет. Они обеспечивают обмен информацией с использованием соответствующего прикладного сервера (службы) Интернета (WWW, E-mail, FTP и др.).

Структура подключения компьютера к Интернету предполагает реализацию следующего последовательного иерархического доступа к сети: сначала обращение к локальной сети компьютера (этот элемент структуры может отсутствовать), далее — к компьютеру регионального сервис-провайдера (Internet service provider, ISP), затем — к национальному ISP, откуда к международному ISP — сетям, входящим в мировую магистральную инфраструктуру Интернета.

ISP — организация, обеспечивающая подключение компьютеров своей сети к Интернет-сети более высокого уровня иерархии.

Региональные и национальные ISP могут устанавливать одноуровневые соединения, организуя обмен трафиком между своими сетями, позволяя снизить загрузку внешних каналов.

Национальная инфраструктура IP-сетей определяется как число магистральных каналов внутри страны, внешних каналов для связи с зарубежными сетями, числом каналов в региональных узлах, так и характеристиками каналов передачи данных, количеством местных сервис-провайдеров.

5.7.1. Национальные провайдеры Интернета

Структура российских сетей начала активно формироваться в 1991—1992 гг., организуя передачу информации в рамках электронной почты (Eunet/Relcom, Sovam Teleport, Glasnet, Freenet), к которым в 1993 г. присоединились научно-образовательные сети RUNNet и Radio-MSU. В 1996 г. к перечисленным сетям добавились сети Demos/Internet, Global One Russia и Rosnet, которые вместе с первыми сформировали российский сегмент Интернета.

Число национальных провайдеров Интернета постоянно изменяется и поэтому трудно указать их точное число. Также постоянно растет и суммарная пропускная способность каналов связи российских сетей. К концу 1996 г. она составила 15 Мбит/с и продолжает ежемесячно возрастать. Приведем сведения о некоторых основных фирмах — участниках российского Интернета (более полную информацию о российской части Интернета публикуют в обзоре Internet Domain Survey — www.nw.com).

EUnet/Relcom (от англ. *Reliable Communications* — надежные связи) (www.relcom.ru) является частью сети EUnet (www.eu.net) — крупнейшего в Европе объединения коммерческих сетей, предоставляющих доступ к услугам Интернета.

Суммарная пропускная способность каналов сети с мировыми сетями Интернета достигает 6 Мбит/с.

Доступ к сети осуществляют более 100 организаций — провайдеров сетевых услуг в 11 странах постсоветского пространства.

Demos biternet (диалоговая единая мобильная операционная система) (www.demos.su) сформировалась на базе сети Relcom, а в 1996 г. выделилась в самостоятельную организацию. Она имеет наземный канал в США (Москва — Бостон) производительностью 2 Мбит/с, связанный с сервис-провайдером MCI Telecommunications Corporation (www.mci.com).

Sovam Teleport (www.sovam.com) является совместным предприятием, организованным в 1990 г. тремя учредителями: Институтом автоматизированных систем (Россия), Cable&Wireless (Великобритания), Global Telesystem Group (США).

Организация имеет три независимых международных канала общей производительностью 4 Мбит/с.

Global One Russia (www.global-one.ru), или «Глобал Один» представляет собой российскую часть группы компаний Global One (www.global-one.net), которая была образована в 1996 г. тремя крупнейшими мировыми телекоммуникационными компаниями: Deutsche Telecom (www.dtag.de), France Telecom (www.francetelecom.fr) и Sprint (www.sprint.com).

«Глобал Один» возник на базе группы компаний: Спринт и Центральный телеграф, действовавших на отечественном рынке с 1990 г.

GlasNet (Гласнет, www.glasnet.ru) входит в международную организацию «Ассоциация поддержки прогрессивных коммуникаций» (www.apc.org) — Association for Progressive Communications (APC), объединяющую информационные сети мира и оказывающую им поддержку.

Сеть Гласнет имеет два внешних канала в США: 512 Кбит/с спутниковый канал в сеть CRL (www.crl.net) и 256 Кбит/с наземный канал в сеть BBNPlanet (www.bbn.com).

Rosnet (Роснет, www/rosnet.ra) — коммерческая телекоммуникационная сеть общего назначения, на базе которой предоставляется комплекс интегрированных коммуникационных и информационных услуг. С 1996 г. Роснет имеет собственный высокоскоростной канал в магистральную часть американского Интернета.

RUNNet (Russian University Network, www.runnet.ru) является IP-сетью, объединяющей региональные сети и сети крупных университетов и других научно-образовательных учреждений. Управление и сопровождение сети осуществляется Научным центром компьютерных телекоммуникационных сетей высшей школы (Вузтелекомцентр), находящимся в Санкт-Петербурге.

Основной вход в глобальный Интернет осуществляется по наземному каналу в скандинавскую академическую сеть, производительность которого составляет 2 Мбит/с и по каналу Москва — Париж в сеть Ebone (www.ebone.net) из сети MSUnet.

FREEnet (The network For Research, Education and Engineering, www.free.net) объединяет на добровольной основе региональные академические компьютерные сети, организации Российской академии наук, университеты, учебные и научно-исследовательские организации.

Основной канал доступа к глобальной сети Интернет поддерживается каналом DTAG (Германия, www.dtag.de).

RELARN-IP. Ассоциация научных и учебных организаций — пользователей сетей передачи данных (RELARN) координирует поддержку организаций, организующих развитие научно-общественных компьютерных сетей. Ее деятельность поддерживается сервером научно-исследовательского института развития общественных сетей.

5.7.2. Информационные ресурсы Интернета

Общее представление об информационных ресурсах, доступных в российской части Интернета, дает краткая характеристика российского WWW и средств поиска информации.

Первые российские WWW-серверы появились в начале 1994 г. Сначала они создавались в университетских и академических сетях, коммерческих фирмах и других организациях, являющихся сервис-провайдерами Интернета. Сегодня спектр WWW-серверов охватывает практически все области науки, производственной и коммерческой деятельности, здравоохранения, образования и культуры России, хотя и отстает от масштабов американского на два порядка.

Доступ к информации, размещенной на российских WWW-серверах, в большинстве случаев свободен и бесплатен. Однако существуют серверы, на которых функционируют и коммерческие информационные системы. В этих случаях для получения доступа к информации необходимо указать имя и пароль, которые задаются заранее при внесении соответствующей суммы. Условия и схема оплаты определены на страницах тех же серверов. Ряд серверов размещает корпоративные информационные системы, доступ к которым ограничен.

ПРАКТИКУМ: ПОИСК И ОБРАБОТКА ИНФОРМАЦИИ В ИНТЕРНЕТЕ

Цель работы: овладеть навыками: поиска и копирования информации в Интернете при помощи специализированных поисковых (справочных) систем (машин), поиска и обращения к веб-сайтам различных организаций, использования диалоговых форм в Интернете и т.п. при помощи браузера Microsoft Explorer.

1. Подготовка браузера к работе.

1.1. Активизируйте на рабочем столе программу **Internet Explorer**:

- а) на рабочем столе щелкните по значку **Internet Explorer**;
- б) щелкнуть по кнопке **Стоп** на панели инструментов браузера,
- в) установите на адресной строке адрес веб-сайта факультета и перейдите на него.

1.2. Ознакомьтесь с основными элементами рабочего окна браузера: строка заголовка, строка меню, панель инструментов, панель быстрых связей, адресная панель, состоящая из кнопок наиболее популярных ресурсов и строки, линейка прокрутки, рабочее окно, строка состояния.

1.3. Оформите элементы управления браузера:

а) соедините на одной панели группу быстрых связей, панель инструментов, адресную строку. Поместите курсор на начало строки вверх или вниз. Измените размер строки. Для этого, захватив и перемещая вправо или влево вертикальную разделительную черту, откройте или закройте необходимую часть панели строки;

б) разместите на объединенной панели группу быстрых связей;

в) откройте на объединенной панели адресную строку;

г) восстановите исходное положение панели инструментов, группы быстрых связей и адресной строки;

д) создайте дополнительное окно для работы с Web-документами. Удержите текущий документ в отдельном окне:

◆ в меню **Файл** активизируйте опцию **Создать окно**;

◆ в новом окне осуществите вызов необходимого документа;

е) создайте группу быстрых связей, например **Сегодня**, объявив ее как Stud_new, и установите связь с веб-сайтом факультета:

◆ перейдите на страницу, к которой следует установить быстрое подключение;

◆ откройте меню Вид и активизируйте опцию **Параметры**;

- ◆ выберите лист **Переходы**;
 - ◆ в списке **Страница** выберите одну из группы связей и нажмите на кнопку **Текущая**;
 - ◆ затем закройте вкладку, щелкнув последовательно по кнопкам **Применить** и **ОК**;
- ж) проверьте эффективность выполненной операции;
- з) восстановите исходное состояние панели быстрых связей.

1.4. Проведите работу со списком ранее посещавшихся страниц. Выберите документ из списка ранее полученных документов с помощью списка **Журнал (History)**:

- а) посетите несколько веб-страниц, размещенных на веб-сайте университета;
- б) вызовите список **History** и выделите одну из строк списка;
- в) захватите строку в поле ярлыка (первая колонка) мышью и перетащите ее на свободное место рабочего стола;
- г) закройте браузер;
- д) щелкнув дважды по ярлыку веб-страницы, вызовите браузер на рабочий стол;
- е) установите многооконный режим чтения документов из папки Журнал. Заполните окна документами из папки **Журнал (History)**;
- ж) восстановите однооконный режим и перейдите на домашнюю страницу факультета университета.

1.5. Осуществите быстрый переход KWeb-странице, размещенной в рабочем каталоге:

- а) перейдите на веб-узел факультета и копируйте его страницу в рабочую папку, установив для файла расширение .html;
- б) в меню **Файл** выберите опцию **Открыть**;
- в) в диалоге **Открытие документа** щелкните по кнопке **Обзор** и найдите файл с ранее копированной страницей, активизируйте эту страницу;
- г) вернитесь на страницу факультета.

2. Поиск информации на веб-серверах Интернета.

2.1. Осуществите поиск текущей информации:

- а) перейдите на веб-сайт информационного издания gazeta.ru или lenta.ru;
- б) ознакомьтесь с содержанием веб-сайта и обратите внимание на структуру размещения информации.

2.2. Осуществите поиск периодических веб-изданий:

- а) подключитесь к странице, содержащей информацию о журнале **Дистанционное обучение**: www.dist-edu.ru;
- б) найдите оглавление одного из номеров журнала;
- в) копируйте оглавление издания в текстовый файл;
- г) вернитесь на домашнюю страницу.

2.3. Ознакомьтесь с содержанием и возможностями поисковых веб-серверов:

- а) используя веб-сервер **rambler.ru**:
 - ◆ ознакомьтесь с продукцией предприятия «ЗИЛ» или «Москвич»;
 - ◆ найдите сведения о погоде на завтра;
 - ◆ узнайте о состоянии курса валюты;
- б) используя веб-сервер **Aport.ru**:
 - ◆ найдите список научных учреждений в области экономики, расположенных в Москве;
 - ◆ определите адреса предприятий, работающих в области компьютерной технологии в Санкт-Петербурге;
 - ◆ скопируйте содержание последнего номера электронного журнала **Бизнес и безопасность в России**;
- в) используя веб-сервер **Yandex.ru**:
 - ◆ подберите сведения о товарах и услугах по профилю Вашей фирмы;
 - ◆ найдите список периодических изданий в области экономики, издаваемых в России;
 - ◆ перейдите на страницу службы **narod.ru**, ознакомьтесь с содержанием страницы службы.

2.4. Посетите электронный каталог ГПНТБ России:

- а) воспользуйтесь веб-страницей ГПНТБ России (следует найти ее);
- б) подберите библиографию по профилю научных интересов.

3. Работа со справочными системами.

3.1. Выберите несколько адресов российских университетов, ведущих подготовку по вашей специальности.

3.2. С помощью поисковой системы найдите информацию об учебных заведениях, ведущих подготовку специалистов в выбранных Вами направлениях (например, обратившись к поисковой системе по адресу <http://weblist.ru/>, перейдите на страницу **Образование**), и перенесите информацию об этих учебных заведениях на поле текстового документа.

3.3. Перейдите на домашнюю страницу факультета Вашего университета, найдите справочную систему **Расписание занятий** и скопируйте на текстовый документ расписание занятий Вашей группы.

3.4. Скопируйте в файл отчета о проделанной работе графический знак или интересную фотографию, обнаруженную на поле одного из веб-документов.

4. Работа в службе телеконференций.

4.1. Установите режим работы с телеконференциями.

4.2. Обратитесь к группе новостей и выберите одну из списка (tomato).

4.3. Ознакомьтесь со списком документов телеконференции и выберите наиболее интересный.

4.4. Скопируйте фрагмент найденного документа в текстовый файл.

4.5. Подготовьте ответ на сообщение и отошлите его автору документа.

5. Восстановите исходные установки браузера.

6. Закройте все приложения.

ГЛАВА 6. HTML — ЯЗЫК ГИПЕРТЕКСТОВОЙ РАЗМЕТКИ

Одно из основных преимуществ современной информационной технологии проявляется в том, что она позволяет работать «одновременно» с несколькими документами. Эти документы могут находиться в различных географических точках, расстояние между которыми не играет никакой роли, иметь различные формы представления информации: текст, графика, таблицы, видео и др. Таким образом, возникает ситуация, позволяющая обращаться к большому многообразию документов или к их элементам, не отходя от компьютера.

Такая ситуация возможна благодаря механизму гиперссылок, который может использовать компьютерный документ. Компьютер, анализируя щелчок мыши на поле документа, отличает простой элемент текста (слово, рисунок и др.) от элемента текста, имеющего связь (ссылку) с другим удаленным документом. На практике гиперссылка имеет цвет текста, отличный от обычного элемента текста, также он чаще всего подчеркнут. Такой элемент определяют как элемент с гиперссылкой (гиперсвязью), а документ, имеющий элементы с гиперссылкой, как гипердокумент.

Документы с гиперссылками определили основное содержание веб-документов, используемых в WWW-сервисе Интернета, так как они формируются в виде совокупности отдельных файлов, ссылки к которым установлены на поле основного веб-документа, что позволяет значительно уменьшить не только объем документа, но и прежде всего время его загрузки на поле браузера. Таким образом, документы с гиперссылками в Интернете чаще всего определяются как веб-документы или веб-страницы.

Веб-страницы имеют особое значение в системе обучения, так как позволяют обращаться к дополнительным источникам информации, активизируя процесс обучения и давая возможность учащемуся постоянно совершенствоваться в предметной области.

В рамках веб-страниц отдельные элементы изображаемой страницы могут иметь горячие связи (hot links) или гиперсвязи, как с элементами текущей страницы, так и с элементами других веб-страниц. Последние могут составлять текущий документ или находиться в составе других документов, которые, в свою очередь, могут располагаться как на текущем, так и на других серверах Интернета, и, более того, горячие связи могут подключать другие ресурсы Интернета, в частности адресовать текущий документ по электронной почте.

Для записи веб-документа используется специализированный язык гипертекстовой разметки — HTML (HyperText Markup Language), который прост в своем содержании и овладение которым позволяет миллионам авторов публиковать свои веб-страницы, содержащие различные информационные материалы, результаты исследований, деловую, учебную и другую информацию.

Первая версия HTML была создана в 1990 г. на базе языка стандарта обобщенной разметки (Standard Generalized Markup Language, SGML). Стандарт HTML 2.0 определяет построение большинства современных страниц WWW. В 1996 г. была выпущена новая версия языка HTML 3.2, позволяющая

более широко представлять таблицы в веб-документах, использовать новые типы форматирования, включать апплеты Java и др.

HTML — это текстовый язык, позволяющий создать WWW-документ (веб-документ) в форме электронной книги, которую можно пролистать, ознакомиться с оглавлением, просмотреть сноски, сделать заметки на полях, перейти на страницы другой книги (справочника и т. п.), получить там дополнительную информацию и снова вернуться к исходному тексту.

Подготовку гипертекстовых документов можно осуществить с помощью простых текстовых редакторов, однако наиболее эффективно использовать специальные программы составления гипертекстовых документов, например Microsoft FrontPage, или встроенные в браузер специальные вкладки редактирования подобных документов, например Netscape Composer. Многие приложения группы офисных приложений также имеют специальные программы для подготовки веб-документов, например система MS Word. Для овладения методами подготовки гипертекстовых документов с целью их публикации в Интернете необходимо:

- ◆ изучить основные правила написания документов на языке HTML;
- ◆ следовать основным правилам соблюдения стиля подготовки документов с помощью языка HTML;
- ◆ научиться представлять в Интернете гипертекстовый документ через текущий сервер WWW.

6.1. Структура HTML-языка

HTML — это язык описания структуры страниц, который позволяет использовать традиционный текст, форматировав его в абзацы, таблицы, заголовки, списки и другие структуры, применять графические объекты и устанавливать ссылки на другие страницы текущего документа и к другим документам и элементы их структур.

Как и любой язык программирования, HTML-язык имеет свой синтаксис и свою структуру. Синтаксис языка определяется набором команд и правилами их использования. Структура языка определяет последовательность размещения команд языка и элементов гипертекстового документа в теговой модели.

В основе гипертекстового языка лежит понятие «тег» (от англ. *tag* — метка, ярлык, этикетка, бирка), обозначающее инструкцию по форматированию следующего за тегом текстового (или иного) фрагмента документа.

Формирование структуры записи поддерживается следующими тегами: HTML, заголовок (Head) и тело документа (Body). Приведенные теги определяются как технические и в общем случае могут не использоваться, однако так поступать не рекомендуется.

Тег сообщает браузеру, каким образом следует представить на экране текущий фрагмент документа или откуда взять дополнительную информацию.

Теги представляют собой последовательность символов, заключенную между знаками «<» и «>» (угловые скобки). Теги располагаются построчно, внутри неформатированного текста документа. Например:

```
<TITLE>Лаборатория информационных технологий</TITLE> .
```

Здесь текст «Лаборатория информационных технологий» окружен тегами <TITLE> и </TITLE>, которые заставят отобразить на верхней строке окна редактора заголовок документа. Первый тег определяет действие разметки для следующего за ним текста. Второй тег, начинающийся с правого слеша, показывает на окончание зоны действия первого тега, в данном случае: <TITLE>.

Тег может иметь атрибуты, определяющие особенности действия текущего тега: расположение текста, цвет букв, рисунок подложки и т. п.

Общая схема построения строки HTML-документа следующая:

```
<имя тега [атрибут(=значение атрибута) ...]>текст</имя тега>, где имя тега — команда, выполняемая браузером. Совокупность доступных тегов определяет набор команд HTML-языка;
```

атрибут — параметры выполняемой команды. Атрибутов может быть несколько и каждый из них может иметь свои параметры. Для некоторых команд атрибуты не устанавливаются, и эта часть конструкции может отсутствовать;

```
<имя тега [список атрибутов]> — определяет начало действия команды;
```

```
текст — неформатированный текст документа;
```

```
</имя тега> — тег, закрывающий область действия текущей команды языка.
```

Команды языка (теги) могут определять различные действия над текстом документа: изменение гарнитуры и шрифта знаков текста, выделение участков текста цветом, встраивание графических элементов и др.

Совокупность строк программы, написанной в формате HTML, определяется как теговая модель.

При записи текста HTML-документа следует учесть следующие *правила*:

1. Пробелы в тексте программы игнорируются.

2. Теги форматирования могут быть написаны строчными и/или прописными буквами. В тех случаях, когда текст расположен в двойных кавычках, следует соблюдать однозначное написание, т. е. записи: `` и `` не эквивалентны.

3. Теги следует писать парами. Открывающий тег активизирует текущую команду, а закрывающий — выключает ее.

Среди базовых тегов HTML исключением из последнего правила являются `<BASE>` (основная информация), `
` (конец строки), `<HR>` (горизонтальная линейка), `` (изображение). Эти теги определяются как непарные, они пишутся только в начале команды.

Пара тегов называется контейнером, поскольку эффект действия команды осуществляется в тексте, размещенном внутри этой пары. Например, для выделения текущей строки полужирным шрифтом следует написать контейнер: `Расписание занятий`, а выделение курсивом потребует пару тегов: `<K>` и `</K>`.

6.1.1. Обозначение HTML-документа

Для обозначения содержания файла HTML-документа следует весь текст документа взять в теги:

```
<HTML>
```

```
...
```

```
</HTML>
```

Пара тегов `<HTML>` и `</HTML>` включает в себе все другие тэги текста программы. Тег `<HTML>` должен размещаться на первой строке, а тег `</HTML>` — на последней строке программы. Эти теги определяют код HTML. Формат обозначения:

```
<HTML>КОд HTML</HTML>
```

6.1.2. Заголовок документа

Заголовок документа оформляется с помощью тега `<Head>`, в поле которого устанавливают два тега: `<Title>`, который определяет название документа, и тег «Базовый URL» `<Base тег="базовый URL">`. В поле заголовка может располагаться и другая информация (как правило, комментарий).

Строка заголовка документа следует сразу после тега `<HTML>` и начинается с тега `<HEAD>`, а заканчивается тегом `</HEAD>`.

Формат заголовка:

```
<HEAD>область заголовка</HEAD>
```

Название документа `<TITLE>` — строка заголовка, которая устанавливается в верхней строке окна браузера и может использоваться браузерами для того, чтобы показать документ в списке **Журнала** и **Избранные страницы**. Имя документа должно достаточно ясно идентифицировать документ, но не должно быть особенно длинным. Не рекомендуется превышать размер имени в 40 знаков.

Формат названия документа:

```
<TITLE>ИМЯ документа</TITLE>
```

Базовый URL определяет абсолютный URL документа. Все другие URL, встречающиеся в тексте программы, могут указываться по отношению к нему. Формат:

```
<BASE HREF="базовый URL">
```

Общая конструкция заголовка HTML-программы будет следующей:

```
<HTML>
```

```
<HEAD>
```

```
<TITLE>Практическая работа №1</TITLE>
```

```
<BASE HREF="http://www.ibm.bmstu.ru/student">
```

```
</HEAD>
```

```
</HTML>
```

В заголовке имя документа и его URL-адрес показаны в качестве примера.

6.1.3. Тело документа

Тело документа HTML содержит непосредственно текст документа, теги, используемые для форматирования текста, встраивания других элементов документа и установки связей с другими гипертекстовыми документами.

Тело документа начинается сразу после его заголовка и ограничивается тегами <BODY> и </BODY>. Общая структура документа при включении пары <BODY> выглядит следующим образом:

```
<HTML>
<HEAD>
<TITLE>Практическая работа №1</TITLE>
<BASE HREF=>http://www.ibm.bmstu.ru/student<>
</HEAD>
<BODY>
</BODY>
</HTML>
```

6.2. Запись текста веб-документа

Размещение текста на поле гипертекстового документа выполняется с помощью специальных команд форматирования. Этот процесс определяется как разметка HTML-документа. Основные приемы разметки текста формируются в следующие группы:

- ◆ начало абзаца и конец строки;
- ◆ стили заголовков;
- ◆ физические стили;
- ◆ логические стили;
- ◆ преформатированный текст;
- ◆ списки;
- ◆ специальные символы.

6.2.1. Начало абзаца и конец строки

Начало абзаца определяется тегом <P>, а его конец — </P>. Тег может иметь атрибут ALIGN, позволяющий установить текст внутри абзаца по левой стороне, по центру, по правой стороне документа, по ширине: ALIGN = LEFT|CENTER|RIGHT|JUSTIFY. Пример записи тега:

```
<P ALIGN = LEFT> ... </P>.
```

Конец строки — абзацы разделяются пустой строкой. Если необходимо начать новый абзац без пропуска строки, следует использовать непарный тег
. Команда определяет обрыв строки (Line Break). Например:

```
<P>
Первая строка инструкции <BR>
Вторая строка инструкции <BR>
Третья строка инструкции <BR>
</P>.
```

Так как по умолчанию текст выравнивается по левому краю, то в показанном примере атрибут тега опущен. В результате выполнения тега текст инструкции в документе будет расположен построчно и выровнен по левому полю документа.

Если текст не имеет элементов (команд) форматирования, то он рассматривается как набор строк максимальной длины и при выводе на экран автоматически разбивается браузером на строки в зависимости от настройки браузера и текущего графического режима.

Тег <NOBR> </NOBR> запрещает перевод строк. Если строка превысит размер экрана, то в рабочем окне браузера появится линейка прокрутки.

6.2.2. Стили заголовков

Гипертекстовый документ, как и любой документ, снабжается заголовком. Для выделения заголовка в тексте HTML-язык предлагает шесть стилей заголовков, обозначаемых как H1, H2, ... H6.

Стиль предполагает использование самого крупного кегля. Для установки стиля заголовка следует воспользоваться следующим форматом записи:

<H*i*>Заголовок</H*i*> ,

где *i* — номер стиля заголовка.

Запишем в качестве примера начальные строки веб-страницы студента, используя различные стили заголовка:

<H1>Веб-страница студента</H1> .

<H3>Университет: МГТУ</H3>

<H5>Курс: 2</H5>

<H6>Специальность: Менеджмент</H6> .

Результат форматирования документа показан на рис. 6.1.

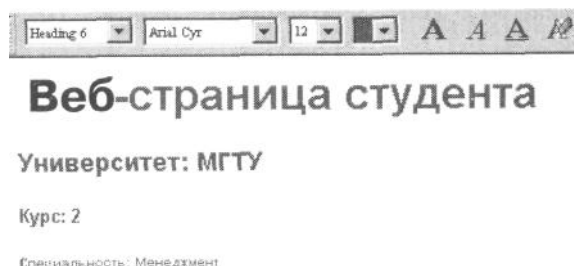


Рисунок 6.1. Примеры стилей заголовков

6.2.3. Физические стили

Физическим стилем текста называют способ выделения знаков текста: полужирный (жирный), курсив (наклон), подчеркивание, машинопись (фиксированная ширина). Изменение стиля текста выполняется с помощью тегов, список которых приведен в табл. 6.1.

Таблица 6.1

Стиль выделения текста	Тег
Полужирный	...
Курсив	<I>...</I>
Подчеркивание	<U>...</U>
Машинопись	<TT>...</TT>

6.2.4. Логические стили

Логические стили выделяют помеченный фрагмент текста. Однако реализация такого стиля не всегда эффективна. Они могут иметь различное изображение в различных браузерах.

В табл. 6.2 приведены теги логических стилей HTML. Следует учесть, что эффект от применения стиля зависит от используемого браузера и поэтому показанные стили полностью соответствуют своему действию только при просмотре документа в браузере Microsoft Explorer.

Таблица 6.2

Имя стиля	Тег	Представление
Адрес	<ADDRESS>...</ADDRESS>	Курсив
Цитируемый абзац	<BLOCKQUOTE>...</BLOCKQUOTE>	Отступы справа и слева
Цитата	<CITE>...</CITE>	Курсив
Код	<CODE>...</CODE>	Фиксированная ширина шрифта
Определение	<DFN>...</DFN>	Полужирный шрифт
Выделение в тексте	...	Курсив

Клавиатура	<KBD>...</KBD>	Фиксированная ширина шрифта
Пример	<SAMP>...</SAMP>	— «—
Выделение	...	Полужирный шрифт
Переменная	<VAR>...</VAR>	Курсив

6.2.5. Преформатированный текст

Текст, использующий шрифт с фиксированной шириной знаков, определяется как преформатированный. Он реализуется с помощью пары тегов <PRE> и </PRE>. Применяется для выравнивания текста в колонки и для подготовки таблиц. Например, если нужно разместить выровненный текст, то его следует записать так, как это показано на рис. 6.2а):

а)

Вид издания	Получено	Списано
Учебники	125	36
Монографии	36	4
Справочники	15	0

б)

```
<PRE>
Вид  == издания ==  Получено ==  Списано
Учебники  == 125 == 36
Монографии == 36 == 4
Справочники == 15 == 0
</PRE>
```

Рисунок 6.2. Примеры использования преформатированного текста

Отличие оформления при использовании преформатированного текста от обычного шрифта заключается в том, что длина текста или чисел определяется размером букв или цифр и размером пропусков между знаками, а для преформатированного текста — все буквы и пропуски имеют одинаковый размер (например, установив один пропуск перед числом 36, можно отодвинуть все число вправо на один знак по сравнению с числом 125, как показано на рис. 6.2б).

Применение тегов <PRE> и </PRE> позволяет активно использовать дополнительные пробелы, знаки табуляции, возврат каретки. Тег <PRE> предполагает запись текста с нового абзаца.

6.2.6. Списки

Простой список состоит из нескольких строк, каждая из которых имеет маркер. Если маркер постоянный, то список определяют как неупорядоченный (маркированный), если в качестве маркера используют цифровую или буквенную последовательность, то список определяют как нумерованный.

 — тег-контейнер, внутри которого располагаются все строки списка;

 — тег, определяющий каждую строку списка (закрывать этот тег необязательно);

TYPE — атрибут тега , определяющий тип маркера: диск, окружность, квадрат — disk, circle, square (следует использовать только строчные буквы). Например, <UL TYPE=circle> (по умолчанию — диск).

 — тег-контейнер, внутри которого располагаются элементы списка;

 — тег строки списка;

TYPE — параметр вида нумерации списка: A — прописные латинские буквы, a — строчные латинские буквы, I — большие римские цифры, i — маленькие римские цифры, 1 — арабские цифры (используется по умолчанию). Пример: <OL TYPE=A>;

START — параметр списка, определяющий начальное значение нумерации (по умолчанию равен 1). В качестве параметра можно использовать только натуральное число, например <OL START=2>;

СОМПАКТ — указывает на вывод списка в компактном виде: уменьшенный шрифт и расстояние между строчками.

Язык гипертекстовой разметки позволяет организовать список описаний (определений), напоминающий страницу толкового словаря: слева устанавливается термин, а в правой части — его определение (толкование). Для такой формы организации списка необходимо использовать следующую конструкцию:

`<DL>` `</DL>` — тег списка описаний, который состоит из строк немаркированного списка и определений к этим строкам, которые записываются после соответствующей строки в виде текста (описание строки списка);

`<DT>` `</DT>` — тег строки списка;

`<DD>` `</DD>` — тег описания строки списка.

6.2.7. Специальные символы

В тексте HTML-документа можно показать специальные символы: зарезервированные символы, символы ASCII кода, комментарии, неразрывный пробел.

Зарезервированные символы используются при записи тегов. Однако, если они необходимы для записи непосредственно текста документа, то они записываются специальным образом, используя знак `&`. Например:

`<` — для записи знака `<`;

`>` — для записи знака `>`;

`&` — для записи знака `&`;

`"` — для записи знака `"`.

Буквы иностранных алфавитов — символы, отсутствующие на традиционной клавиатуре, но используемые для записи HTML-документа, записываются после знака `&` (амперсant) и заканчиваются знаком `«;»`.

Символ ASCII-кода записывается по тем же правилам: начинается последовательность знаком амперсant, затем следует запись цифрового кода, заканчивается запись знаком `«;»`. Например, для записи знака защиты авторского права следует написать: `Copyright ©`, 2003, что позволит изобразить:

Copyright У, 2003

Таблица символов (Character map), позволяющая узнать код любого символа ASCII, поставляется вместе с Windows.

Комментарии устанавливаются в текст HTML-кода, но не выводятся на экран. Тег комментария начинается с восклицательного знака и имеет следующий формат: `<!-- Текст комментария -->`.

Неразрывный пробел используется в тех случаях, когда нельзя разрывать рядом стоящие слова. Такой пробел (non-breaking space) записывается с помощью последовательности ` `.

6.3. Организация гипертекстовых ссылок

Гипертекстовые ссылки позволяют связать отдельные элементы текста (слова, группы слов, элементы графики и т.д.) с другими HTML-документами, что дает возможность осуществить быстрый переход от одного документа к другому по смыслу без использования процедур поиска, а также эффективное оглавление документа.

В результате установка гипертекстовых ссылок формирует сложный документ, имеющий нелинейную структуру. Благодаря этим ссылкам можно знакомиться с содержанием документа в любой последовательности, привлекая для пояснений фрагменты других документов.

Для установки гиперссылок следует использовать следующую теговую конструкцию:

тег `...` — указатель ссылки, между тегами которого устанавливается текст указателя. В поле тега устанавливается адрес гиперссылки `HREF=URL` (`HREF` — Hypertext REFerence), т. е. адрес того документа, который будет вызван после щелчка мышью по гиперссылке. Между парой тегов можно записать текст, установить рисунок, дать адрес веб-страницы и т. д., который и будет оформлен как гиперссылка: подчеркнут, выделен цветом и т. п.

Таким образом, гипертекстовая ссылка состоит из двух частей: указателя и адреса (URL-адреса).

Указатель — это текст (слово, рисунок, группа слов и т. п.), по которому необходимо щелкнуть мышью с целью перехода на связанный с этим текстом другой документ.

Указатель оформляется в виде подчеркивания и выделения цветом соответствующего фрагмента текста. Например, в тексте «Новые информационные технологии» указатель связывает слово «технологии» с другим документом, раскрывающим содержание подчеркнутого слова. Связывание осуществляется с помощью URL-адреса.

URL-адрес указывает на адрес документа, к которому обратится браузер при активизации связи.

6.3.1. Формирование указателей

В качестве указателя может выступать любой элемент текста вне зависимости от его размеров или способа форматирования. Формат адресного указателя следующий: `ТСКСТ указателя`, где А — указатель (anchor, якорь);

HREF — гипертекстовая ссылка (Hypertext REFerence);

URL — адрес гипертекстовой ссылки;

текст указателя — элемент документа, который выделяется на экране подчеркиванием или другим цветом. Вид указателя может иметь иной формат, что следует оформить соответствующей парой тегов, например:

`<I>Т6КСТ указателя</I>`.

Например, если записать: `Мой университет` и в поле тега заменить знаки URL на веб-адрес университета (оставляя кавычки), то после вызова на рабочее поле браузера подготовленной страницы на экране слова «Мой университет» будут выделены (цветом и/или подчеркиванием), а после щелчка мышью по этим словам браузер загрузит первую (главную, индексную) страницу университета.

6.3.2. Создание оглавления с помощью имен указателей

При подготовке достаточно больших гипертекстовых документов (пособий, инструкций, рекомендаций и т. п.) следует в начале документа поместить его оглавление. Оглавление является примером составления имен указателей, определяющим переход на соответствующий фрагмент документа.

Для реализации документа, содержащего оглавление, необходимо выполнить следующие операции:

- ◆ разделить текст на отдельные фрагменты документа (главы, разделы и т. п.) и оформить эти разделы в виде самостоятельных файлов;
- ◆ создать имена указателей для каждого заголовка (главы, раздела и т. п.), например: `<H3>Глава 1</H3>`;
- ◆ установить в тексте оглавления и в других текстовых фрагментах ссылки на установленные указатели, используя знак # и имя указателя в конце URL документа, например: `Смотри первую главу`.

6.3.3. Применение гиперссылок для формирования оглавления текущего веб-документа

Гиперссылки можно использовать для формирования оглавления документа. Такие гиперссылки определяются как внутренние ссылки.

Оглавление документа выполняется в следующей последовательности: сначала следует разделить текст документа на отдельные разделы, отделяя их несколькими пустыми строками, затем необходимо озаглавить каждый раздел и установить перед каждым из них указатели, используя конструкцию гиперссылки, в заключение в начале текста документа устанавливаются оглавление, которое состоит из соответствующего числа гиперссылок.

В конце каждого раздела документа можно поставить гиперссылку на начало оглавления документа, например «Вернуться в оглавление».

Указатель, который необходимо установить в начале каждой главы (части, раздела) текста (перед заголовком главы), имеет следующую конструкцию: ` ` (в качестве «имя_части» можно записать «Label_1» и т. д.). Имя главы должно состоять из одного слова, для чего можно использовать знак «_».

В оглавлении для создания ссылки на соответствующий указатель, размещенный в текущем документе, в параметре HREF следует указать имя ссылки с префиксом #: ` Подготовка документа`. Например, ` Подготовка документа <^>`. В этом случае после щелчка по первой строке оглавления текст документа будет прокручен до раздела, который начнется с указателя `...`

ПРАКТИКУМ: ФОРМИРОВАНИЕ ТЕКСТА ВЕБ-ДОКУМЕНТА

Цель работы: овладеть навыками разметки текста веб-документа с использованием языка гипертекстовой разметки HTML: организация заголовка и содержательной части документа, методы

логического и физического форматирования строк, заголовков, преформатированного текста, списков, разделение на абзацы, перевод строки, применение специальных символов, организация гиперссылок, формирование оглавления документа и др.

1. Подготовка HTML-документа.

1.1. Вызовите на поле экрана стандартный текстовый редактор **WordPad** или **Блокнот**:

1.2. а) сохраните текущий документ в рабочей папке под именем 4_1_ФИО (первые буквы фамилии, имени и отчества обучаемого) с расширением .html, используя опцию **Сохранить как...**;

б) разместите рабочее поле текстового редактора на правой половине экрана;

в) вызовите браузер **Internet Explorer** и установите его на левой половине экрана.

1.2. Вызовите на рабочее поле браузера HTML-документ из своей рабочей папки. Для этого в меню **Файл** найдите опцию **Из файла**, нажмите кнопку **Обзор** и найдите ранее подготовленный файл в рабочей папке.

1.3. Запишите заголовок:

а) в поле заголовка установите пару тегов <TITLE> и запишите между ними строку «Практическое занятие 4.1_ФИО»;

б) сохраните измененный текст в текстовом редакторе, обращая внимание на расширение .html;

в) активизируйте браузер, щелкнув по кнопке **Обновить**. Обратите внимание на содержание верхней строки окна браузера. Она должна отобразить название документа. Запомните URL-адрес документа, указанный в поле адресной строки.

1.4. Оформите содержательную часть документа:

а) перейдите на поле текстового редактора и установите пару тегов <BODY> после тега </HEAD>.

б) установите курсор между парой <BODY> и несколько раз нажмите на клавишу <Enter>, чтобы расширить поле для строк разметки текста документа.

2. Запись текста.

2.1. Установите заголовок на поле документа:

а) запишите в начале содержательной части четыре строки заголовка, увеличивая шрифт заголовка каждой следующей строки: наименование университета, наименование факультета, наименование специальности, фамилия, имя, отчество студента.

б) перейдите на окно браузера, нажмите на кнопку **Обновить**.

2.2. Запишите отдельные строки текста.

а) разделите следующий текст на пять строк, используя тег
 (в качестве разделителя строк следует использовать запятую): Нам не дано предугадать, Как слово наше отзовется, — И нам сочувствие дается, Как нам дается благодать... Ф.И. Тютчев;

б) перейдите на окно браузера, нажмите на кнопку **Обновить** и проверьте результат разметки текста.

2.3. Запишите текст по абзацам:

а) установите заголовок «Запись текста по абзацам»;

б) на четырех следующих строках разместите фразу: «Форматирование абзаца», размещая ее по правому краю, по левому краю, по центру и по ширине соответственно. В последнем случае, чтобы понять работу атрибута, следует повторить фразу 10 раз;

в) перейдите на окно браузера, нажмите на кнопку **Обновить** и проверьте результат.

2.4. Выделите текст:

а) запишите заголовок текущего задания и построчно название первых четырех физических стилей, используя на каждой строчке соответствующую форму выделения текста;

б) начните новый абзац и перенесите копию текста (п. 2.2), представьте его с помощью шрифта большего размера;

в) перенесите строку автора текста (п. 2.2) и представьте ее с помощью шрифта меньшего размера;

г) в новом абзаце запишите математическую формулу: $(X+Y)*Z^{X,Y}$;

д) на новой строке запишите химическую формулу $C_6H_{12}O_5$;

е) перейдите на окно браузера, нажмите на кнопку **Обновить** и проконтролируйте результат разметки;

ж) установите параметры шрифта по умолчанию в заголовке программы и оцените результат;

з) уберите строку с установкой параметров шрифта по умолчанию.

2.5. Примените логические стили для организации фрагментов текста:

а) в режиме текстового редактора на первой строке содержательной части HTML-документа запишите сведения об авторе документа (фамилия, инициалы, время записи документа), используя логический стиль. Перейдите на окно браузера, нажмите на кнопку **Обновить** и просмотрите результат;

б) найдите веб-страницу кафедры и скопируйте фрагмент сопроводительного текста. Перенесите этот фрагмент на поле формируемого HTML-документа и оформите этот фрагмент как цитируемый абзац. Перейдите на окно браузера, нажмите на кнопку **Обновить**.

2.6. Осуществите выделение с помощью преформатированного текста:

а) используя машинописный текст, сформируйте табл. 6.3;

б) перейдите на окно браузера, нажмите на кнопку **Обновить** и проверьте результат.

Таблица 6.3

Вид издания	Получено	Списано
Учебники	150	46
Монографии	20	2
Справочники	5	

2.7. Используя в тексте символы, которые применяются для разметки текста, осуществите следующие операции:

а) по центру строки запишите: «Тег
 определяет обрыв строки (Line Break)»;

б) на последней строке документа запишите: ©УФИО, год (ASCII-код знака «©» соответствует числу 169, можно также использовать комбинацию «©»). Перейдите на окно браузера, нажмите на кнопку **Обновить** и проверьте результат;

в) установите курсор на начало строки об авторских правах и несколько раз нажмите клавишу <Enter>. Далее упражнения записывайте в образовавшемся разрыве текста так, чтобы строка с упоминанием об авторских правах всегда оставалась последней в содержательной части документа.

2.8. Выделите произвольный фрагмент текста с помощью изменения его параметров:

а) выделите цветом ранее подготовленную таблицу с преформатированным текстом;

б) перейдите на окно браузера, нажмите на кнопку **Обновить**.

3. Организация гипертекстовых ссылок.

3.1. Установите гиперссылки:

а) запишите фразу: «Студент ФИО учится в университете ... на факультете... на кафедре...». Вместо точек укажите реальный текст, оформив его в виде гиперссылок на соответствующие веб-страницы университета, факультета, кафедры;

б) перейдите на окно браузера, нажмите на кнопку **Обновить**. Проверьте эффективность действия гиперсвязей.

3.2. Сформируйте оглавление для текущего документа:

а) установите указатели в начале упражнений, выполненных по следующим пунктам заданий: 1, 2.1, 2.4, 2.7, 3, присвоив им соответствующие имена (имя_части);

б) запишите построчно в начале содержательной части документа оглавление со ссылками на соответствующие указатели;

в) перейдите на окно браузера, нажмите на кнопку **Обновить**. Проверьте эффективность действия гиперссылок;

г) организуйте текст задания с подробным оглавлением в форме HTML-документа.

4. Копируйте текст текущего задания в рабочую папку, присвоив ему имя 4_ФИО.

5. Сохраните HTML-коды документов, проверьте их на вирус и запишите на дискету.

6. Закройте все приложения на рабочем столе.

6.4. Цвет текста и фоновые изображения

Для разработки страниц веб-документов используют различные цветовые решения. Они устанавливаются либо при записи параметров тега <BODY> и тогда распространяются на весь документ либо при записи отдельных элементов страницы и тогда они записываются в поле соответствующего тега. В этих случаях говорят о разработке графических страниц.

TEXT, LINK, ALINK, VLINK — параметры, характеризующие цвета текста и указателей ссылок (неиспользованные обозначаются параметром **LINK**, использованные — **VLINK**, а активизированные в настоящий момент — **ALINK**). Формат параметра следующий:

```
<BODY TEXT=код цвета LINK=код1 цвета>
```

текст документа

```
</BODY>
```

Для указания кода цвета используется шестнадцатиричный код: **#RRGGBB**, значение которого также определяется соответствующими кодами.

По умолчанию для текста и ссылок устанавливаются следующие цвета: **TEXT=black, LINK=blue, ALINK=red, VLINK=purple**.

6.5. Установка табличных элементов

Гипертекстовые документы могут использовать не только текст, но и другие формы представления информации, например таблицы. Они используются в тех случаях, когда суть документа (или его фрагмента) раскрывается через определенный набор числовых данных. Последнее нередко позволяет принимать быстрое и правильное решение.

При использовании только текстовых тегов для создания таблиц следует применять тег преформатированного текста. В этом случае все поле документа можно разделить по числу размещаемых знаков (включая и пробелы).

Для упрощения процедур организации таблиц в HTML-документах применяют специальные табличные теги. В основе их реализации заложено понятие «ячейка», которая может содержать как элементы данных таблицы, так и ее заголовок. Логически связанные ячейки составляют строку таблицы, а строки, в свою очередь, образуют саму таблицу.

6.5.1. Основные табличные теги

Тег «таблица» содержит пару: `<TABLE>теги создания таблицы</TABLE>`.

Рамка вокруг таблицы определяется атрибутом **BORDER=n**, с помощью которого создается рамка шириной **n** пикселей. По умолчанию рамка не изображается.

Теги `<TR>` и `</TR>` определяют содержание текущей строки таблицы.

Атрибутами тега являются:

- ◆ **ALIGN** — контролирует горизонтальное расположение содержимого в ячейках строки. Он принимает значения: **LEFT|RIGHT|CENTER** (выравнивание по левому краю, по правому краю и по центру);
- ◆ **VALIGN** — определяет вертикальное расположение элементов строки. Он принимает значения: **TOP|BOTTOM|MIDDLE** (выравнивание по верхнему краю, по нижнему краю, по середине). По умолчанию устанавливаются атрибуты горизонтального и вертикального выравнивания.

Теги `<TD>` и `</TD>` устанавливают содержимое ячейки таблицы. В качестве содержания можно использовать данные или текст. При использовании этой пары тегов можно осуществить следующие конструкции:

- ◆ если отсутствует текст или записано ` `, то ячейка остается пустой;
- ◆ если использовать тег ``, то можно установить рисунок на поле ячейки таблицы;
- ◆ установить на поле ячейки таблицы новую (вложенную) таблицу.

Содержимое обоих типов ячеек имеет вертикальное выравнивание **MIDDLE**, а горизонтальное: по левому краю — для данных, и по середине — для текста.

Теги `<TH>` и `<DH>` устанавливают текст в ячейке таблицы. Текст ячейки автоматически выравнивается по центру и выделяется полужирным шрифтом.

Тег `<CAPTION>` и `</CAPTION>` позволяют сформировать заголовок таблицы, устанавливая его вне ее поля. По умолчанию тег имеет параметр **TOP** (для заголовка), но его можно заменить на **BOTTOM** для организации подписи к таблице. Тег заголовка должен следовать либо непосредственно после тега `<CAPTION>`, если заголовок находится над таблицей, или непосредственно перед `<CAPTION>`, если заголовок находится под таблицей.

В качестве примера покажем текст HTML-документа, позволяющего выводить в форме таблицы распределение часов учебных занятий в учебном курсе.

```
<TABLE BORDER=2>
<CAPTION>Учебная программа</CAPTION>
<TR>
<TH>Виды занятий</TH>
<TH>Лекции</TH> <TH>Семинары</TH> <TH>Лабораторные</TH>
</TR>
<TR ALIGN=LEFT>
<TH>Учебные часы</TH>
<TDALIGN=CENTER>17</TD><TDALIGN=CENTER>17</TD>
<TD ALIGN=CENTER>34</TD>
</TR> </TABLE>
```

Результат действия программы позволит получить на экране документ, показанный на рис. 6.4.

Виды занятий	Лекции	Семинары	Лабораторные
Учебные часы	17	17	34

Рисунок 6.4. Простая таблица на поле веб-страницы

При подготовке таблицы следует учесть следующее замечание: параметры выравнивания в тегах `<TD>`, `<TR>`, `<TH>` имеют более высокий приоритет перед другими параметрами, распространяемыми на данную ячейку.

Дополнительные атрибуты табличного тега `<TABLE>`:

CELLSPACING — задает расстояние между отдельными ячейками таблицы. По умолчанию атрибут имеет значение 2. Например, если необходимо увеличить расстояние между ячейками до 4, следует записать:

```
<TABLE CELLSPACING=4> .
```

CELLPADDING — определяет расстояние между текстом (рисунком и т. п.) отдельной ячейки и разделительной линией. По умолчанию атрибут принимает значение 1. Например, для более экономного использования площади документа можно записать:

```
<TABLE BORDER=0 CELLSPACING=0 CELLPADDING=0> .
```

WIDTH=значение или параметр — определяет ширину ячейки:

- ♦ при использовании с тегом `<TABLE>` устанавливает ширину таблицы в пикселях или процентах от общей ширины рабочего окна браузера;
- ♦ совместно с тегами `<TH>` и `<TD>` определяет ширину ячейки таблицы в пикселях или процентах от ширины таблицы.

HEIGHT=значение или параметр — определяет высоту ячейки или таблицы. Действие атрибута аналогично атрибуту **WIDTH**.

BGCOLOR и **BORDERCOLOR** — устанавливают цвет фона и разделяющих линий таблицы. Атрибуты используются также с тегами `<TH>`, `<TD>` и `<TR>`.

BACKGROUND — поддерживает цветовой режим фона таблицы, когда в качестве фона используется рисунок.

BORDERCOLORLIGHT — перекрашивает правый и нижний края ячейки,

BORDERCOLORDARK — перекрашивает левый и верхний края ячейки.

6.5.2. Размещение сложных таблиц

Таблицы, в которых ячейки могут занимать несколько строк, можно определить как сложные. Для организации сложных таблиц в тегах `<TD>` и `<TH>` применяют следующие атрибуты: **ROWSPAN**, **COLSPAN**, **NOWRAP**.

Эти атрибуты определяют то число строк или столбцов, которые должна занимать текущая ячейка. При этом таблица размечается построчно.

ROWSPAN — определяет число строк, которое будет занимать текущая ячейка. По умолчанию параметр принимает значение 1.

COLSPAN — устанавливает число колонок таблицы, которые займет текущая ячейка. По умолчанию параметр равен 1.

NOWRAP — запрещает разбивку текущей строки, помещая ячейку на всю ширину таблицы.

В качестве примера текста HTML-документа рассмотрим текст, формирующий таблицу учебной нагрузки студентов в сессию по семестрам обучения. Ее вид в текстовом документе показан в табл. 6.5.

Рассмотрим основные этапы разработки таблицы.

Таблица 6.5

		Год обучения			
		1 курс	2 курс	3 курс	4 курс
1 семестр	Зачет	5	5	4	3
	Экзамен	4	4	3	4
2 семестр	Зачет	4	4	5	3
	Экзамен	4	4	4	4

1. Открыть табличный тег и установить ширину разделительных линий таблицы: `<TABLE BORDER=1>`.

2. Для шапки (головки) таблицы открыть теги строки и текста, указав в последнем параметры: `ROWSPAN=2 COLSPAN=2` (угловая ячейка занимает две строки и два столбца), закрыть тег текста, так как первая ячейка пуста. В теге следующей текстовой ячейки первой строки указать, что она займет четыре столбца с помощью параметра `COLSPAN=4`, и записать текст ячейки «Год обучения»: `<TH COLSPAN=4>Год обучения</TH>`

3. Записать вторую строку шапки таблицы, открыв тег строки, ячейки, и тегов текстового содержания ячеек: `<TR><TH>1 курс</TH><TH>2 Курс</TH><TH>3 Курс</TH><TH>4 Курс</TH></TR>`.

4. Самостоятельно разместить цифровые данные посередине ячейки.

5. Перейти на окно браузера, нажать на кнопку **Обновить**. Проверить организацию шапки таблицы.

6. Перейти в поле текстового редактора. Открыть третью строку и указать, что первая ячейка занимает две строки и на своем поле содержит текст «1 семестр», что в поле второй ячейки следует написать «Зачеты», а в следующих четырех ячейках — их количественные значения:

```
<TR><TH ROWSPAN=2>1 семестр</TH>  
<TH>Зачеты</TH><TD>5</TD><TD>5</TD><TD>3</TD><TD>4</TD>  
</TR>
```

7. Открыть четвертую строку таблицы (она начинается с описания второй ячейки). Здесь следует указать текст «Экзамен», а в следующих ячейках строки — их количественные величины:

```
<TR>  
<TH>Экзамен</TH>  
<TD>4</TD><TD>4</TD><TD>4</TD><TD>4</TD>  
</TR>
```

8. Перейти на окно браузера, нажать на кнопку **Обновить**. Проверить организацию таблицы для первого семестра.

9. Перейти на поле документа и записать разметку пятой и шестой строк таблицы, повторяя ранее показанные действия. Перейти на окно браузера, нажать на кнопку **Обновить**. Проверить организацию таблицы.

10. Обратит внимание на то, что данные в таблице размещены по левому краю, изменить их положение, расположив по центру.

11. Выделить шапку сложной таблицы цветом, выделить остальные строки таблицы другим цветом и изменить цвет разделительных линий.

После разметки на поле веб-страницы появится таблица, изображенная на рис. 6.5.

		Год обучения			
		1 курс	2 курс	3 курс	4 курс
1 семестр	зачет	5	5	4	3
	экзамен	4	4	3	4
2 семестр	зачет	4	4	5	3
	экзамен	4	4	4	4

Рисунок 6.5. Фрагмент рабочего поля веб-документа, содержащего сложную таблицу

6.5.3. Совмещение текста и таблицы на поле документа

В тех случаях, когда в поле текста необходимо установить небольшую таблицу, используют специальный прием, основанный на форматировании таблицы по одному из краев поля документа и размещении текста вокруг поля таблицы.

Суть метода заключается в реализации следующих тегов:

- ◆ смещения поля таблицы (например, влево): `<TABLE ALIGN =LEFT>`;
- ◆ формирования строк таблицы;
- ◆ записи текста документа.

В качестве примера дополним страницу с таблицей числа экзаменов и зачетов, представленной на рис. 6.6, и рассмотрим результат работы новой программы, который показан на том же рисунке.

		Год обучения			
		1 курс	2 курс	3 курс	4 курс
1 семестр	зачет	5	5	4	3
	экзамен	4	4	3	4
2 семестр	зачет	4	4	5	3
	экзамен	4	4	4	4

Из таблицы следует, что общее число экзаменов и зачетов за период обучения студента в университете равно 54.

Рисунок 6.7. Совмещение текста и таблицы на поле документа

Для совмещения текста и таблицы в строках программы следует внести следующие дополнения:

```
<HTML>
<TABLE BORDER=2 ALIGN=LEFT>
описание таблицы
</TABLE>
```

Из таблицы следует... (далее по тексту, представленному на рис. 6.6).

```
</HTML>
```

Если текст занимает большее число строк на поле документа, чем таблица, то текст огибает поле таблицы и занимает всю ширину страницы.

Прием совмещения текста и таблицы используется также для организации пояснений к отдельным фразам (рисункам и т. п.) документа. Например, если вместо таблицы использовать выделенные слова «Экзамены и зачеты» и оставить комментарий об их числе, то текст документа может выглядеть так, как показано на рис. 6.7, а текст разметки – следующим образом:

```
<HTML>
<TABLE ALIGN=LEFT>
<TR>
  <TD ALIGN=CENTER><HR>
  <B>Экзамены и зачеты</B>
  <HR>
</TD>
</TR>
```

```
</TABLE>
<BLOCKQUOTE>
Текст пояснения к выделенным словам.
</BLOCKQUOTE>
```

ЭКЗАМЕНЫ И ЗАЧЕТЫ	Из таблицы следует, что общее число экзаменов и зачетов за период обучения студента в университете равно 54.
-------------------	--

Рисунок 6.7. Выделение текста с помощью таблицы

В показанном примере использованы теги <BLOCKQUOTE> и </BLOCKQUOTE>, с помощью которых поясняющий текст размещается посередине относительно поясняемых слов «Экзамены и зачеты», так, чтобы текст пояснения начинался раньше этих слов и заканчивался после них.

ПРАКТИКУМ: СПИСКИ И ТАБЛИЦЫ НА ПОЛЕ ВЕБ-СТРАНИЦ

Цель работы: овладеть навыками установки списков и таблиц на страницах веб-документа: маркированный список, графические маркеры, нумерованный список, списки определений, вложенные списки, простые таблицы, форматирование данных внутри таблицы, цвет в таблицах, сложные таблицы, вложенные таблицы, особенности построения таблиц и др.

1. Подготовка документа для ввода текста.

1.1. Вызовите новый документ в текстовом редакторе **WordPad**, сохраните его под новым именем с расширением .html:

а) сохраните новый документ в личной папке под именем 4_2_ФИО: меню **Файл**, опция **Сохранить как...**, укажите в списке **Тип файла** документ HTML;

б) запишите заголовок HTML-документа: HTML-списки и таблицы: ФИО;

в) установите рабочее окно редактора на правой стороне экрана.

1.2. Вызовите браузер и перейдите на поле веб-документа:

а) в меню **Файл** активизируйте опцию **Открыть страницу**. В новом диалоге щелкните по кнопке **Обзор**;

б) выберите в рабочей папке необходимый файл. Вернувшись к диалогу, щелкните по кнопке **Открыть**;

в) установите рабочее окно редактора на левой стороне экрана.

2. Разметка простого списка.

2.1. Организуйте маркированный (неупорядоченный) список:

а) создайте список учебных дисциплин текущего семестра, используя один из дополнительных маркеров;

б) сформируйте вложенный маркированный список, используя в качестве первого уровня список учебных дисциплин, а в качестве второго — дополнительные подстроки: «лекции и семинары» с использованием другого маркера, указав число учебных часов в семестре по каждому виду занятий;

в) перейдите на окно браузера, нажмите на кнопку **Обновить**. Проверьте организацию списка.

2.2. Сформируйте список с использованием рисунка в качестве маркера:

а) подготовьте самостоятельно рисунок маркера (например, шарик) или найдите на веб-страницах подходящий маркер, скопируйте его в рабочую папку в формате .gif или .jpg;

б) разместите заголовок для нового списка учебных занятий на текущий день;

в) запишите строки списка, начиная каждую из них с установки URL-адреса рисунка маркера, используя конструкцию сТроКа сnncKa
;

г) закройте список. Проверьте результат.

2.3. Организуйте нумерованный (упорядоченный) список:

а) приведите нумерованный список сотрудников фирмы, состоящий из пяти строк;

б) разместите на веб-странице список, пример организации которого показан на рис. 6.8;

- I Глава 1
 - A. Проблема 1
 - B. Проблема 2
- II Глава 2
 - A. Задача 1
 - 1. Цель 1
 - 2. Цель 2
 - а) Средство 1
 - б) Средство 2
 - 3. Цель 3
 - B. Задача 2
 - C. Задача 3

Рисунок 6.8.

в) разработайте упорядоченный список — оглавление учебного курса, состоящий из частей, тем, глав, параграфов, разделов; г) составьте нумерованный список Ваших основных задач, начиная нумерацию с буквы «С»;

д) перейдите на окно браузера, нажмите на кнопку **Обновить**. Проверьте организацию списка.

2.4. Организуйте список описаний:

а) на веб-странице повторите список товаров фирмы с описанием их свойств, используя формат списка описаний;

б) перейдите на окно браузера, нажмите на кнопку **Обновить**. Проверьте организацию списка.

3. Установка табличных объектов.

3.1. Разместите простую таблицу на поле документа:

а) сформируйте простую таблицу «Учебная программа»;

Учебная программа

Виды занятий	Лекции	Семинары	Лабораторные
Учебные часы	10	14	20

б) сформируйте и разместите на поле веб-документа таблицу «Состояние фонда литературы», состоящую из четырех столбцов: издания, получено, списано и трех строк: монографии, учебники, справочники. В ячейки таблицы установите ориентировочные значения;

в) перейдите на окно браузера, нажмите на кнопку **Обновить**. Проверьте организацию таблицы;

г) повторите таблицу «Учебная программа», изменив расстояние между ячейками до 10 пикселей и отступ между рамкой и данными до 5 пикселей. Проверьте организацию таблицы;

д) повторите таблицу «Учебная программа», используя размеры 50x50 пикселей и 200x200 пикселей. Объясните полученные реальные размеры таблицы;

е) разместите таблицу «Состояние фонда литературы» на левой половине экрана, а на правой половине покажите текст, объясняющий причины списания книг. Для этого следует использовать конструкцию: `<TABLE BORDER=2 ALIGN=LEFT>`. Текст необходимо писать после закрывающего тега таблицы.

3.2. Оформите таблицу с использованием цвета:

а) выделите шапку таблицы «Учебная программа» неярким цветом (например, #FFFF00);

б) выделите вторую строку таблицы другим цветом (например, #F5F5F5) и измените цвет разделительных линий;

в) перейдите на окно браузера, нажмите на кнопку **Обновить**. Проверьте организацию таблицы.

3.3. Используйте метод совмещения таблицы и текста для организации пояснений к отдельным фразам (рисункам) текста:

а) записывая разметку новой таблицы, сместите ее влево;

б) организуйте таблицу из одной ячейки (по центру).

в) после табличного тега организуйте текст пояснения с помощью контейнера `<BLOCKQUOTE>`;

г) перейдите на окно браузера, нажмите на кнопку **Обновить**. Проверьте организацию таблицы.

3.4. Сформируйте сложную таблицу «Итоги деятельности предприятия» и внесите необходимые данные:

Итоги деятельности предприятия

		1 полугодие	2 полугодие	Год
Цех 1	Доходы	565	647	1183
	Расходы	105	134	
Цех 2	Доходы	1200	1250	2447
	Расходы	187	190	

3.5. Разработайте таблицу, в которой покажите число учебных часов по видам занятий: лекции, семинары, лабораторные для трех учебных дисциплин:

а) организуйте таблицу из трех строк без разделительных линий. На первой строке разместите одну общую ячейку с текстом «Учебная программа текущего семестра»;

б) вторую и третью строки таблицы разбейте на три. На второй строке в каждой из этих ячеек запишите соответствующие заголовки — учебные курсы, а в ячейках третьей строки установите таблицы (в две строки) с указанием вида учебных занятий — в первой строке и число учебных часов в семестре по каждому виду занятий — во второй строке;

в) перейдите на окно браузера, нажмите на кнопку **Обновить**. Проверьте организацию таблицы.

4. Формирование галереи изображений использовании таблицы.

4.1. Организуйте таблицу с необходимым числом ячеек и строк (например, 3x3).

В данном примере использование тега <P> продиктовано необходимостью добавления нескольких пустых строк сверху и снизу разделительной линии. Для раздела двух частей число пропусков можно увеличить, а для подчеркивания — уменьшить.

4.2. Выберите на веб-страницах подходящие изображения и запомните их URL-адреса.

4.3. Установите на поле каждой ячейки таблицы ссылки на соответствующий URL-адрес:

```
<IMG SRC="URL"X
```

4.4. Перейдите на окно браузера, нажмите на кнопку **Обновить**. Проверьте организацию таблицы.

5. Сохраните HTML-коды документов, проверьте их на вирусы и запишите на дискету.

6. Закройте все приложения на рабочем столе.

6.6. Размещение графических элементов

Графические элементы являются важной составляющей практически любого HTML-документа. Диапазон графических элементов — от прямой линии до сложных изображений. Основным критерием, который используется для определения необходимости установки графического элемента, служит критерий повышения эффективности. Его суть состоит в том, что следует использовать соответствующую графику там и в тех случаях, когда и где необходимо повысить информативное восприятие передаваемого документа.

К графическим элементам относят: горизонтальные и вертикальные линии, графические изображения, указатели ссылок.

6.6.1. Горизонтальные линии

Для разделения фрагментов документа, а также для выделения части документа используют горизонтальные линии. Для установки горизонтальной линии служит тег <HR> (Horizontal Rule, горизонтальная линейка, шпон). Этот тег является исключением и не требует закрывающего тега. В тексте он записывается следующим образом:

```
... <P>
```

```
<HR>
```

```
</P> ..
```

Тег <HR> имеет атрибуты, позволяющие управлять размером, толщиной, выравниванием, цветом и видом тени линеек:

- ◆ **WIDTH=pixels|percent** — изменяет размер линейки, задается в пикселях или в процентах от ширины окна браузера;
- ◆ **ALIGN=LEFT|RIGHT|CENTER** — выравнивает фрагмент линейки;
- ◆ **SIZE=n** — определяет толщину линейки (по умолчанию n = 1);

- ◆ **COLOR=RGB** (триплет или название цвета) — задает цвет;
 - NOSHADA** — отменяет тени от линеек, при этом выводятся сплошные полосы.
- Текст, показанный на рис. 6.9, позволит оценить воздействие рассмотренных атрибутов:
- ```
<HR>
```
- Линия в 1 пиксель<?>
- ```
<HR SIZE=10 WIDTH=50% ALIGN=RIGHT>
```
- Линия с параметрами: SIZE=10 WIDTH=50% ALIGN=RIGHT<P>
- ```
<HR SIZE=14 NOSHADE>
```
- Линия с параметрами: SIZE=14 NOSHADE<P>
- ```
<HR SIZE=20 NOSHADE WIDTH=40% ALIGN=LEFT>
```
- Линия с параметрами: SIZE=20 NOSHADE WIDTH=40% ALIGN=LEFT<P>
- ```
<HR>
```

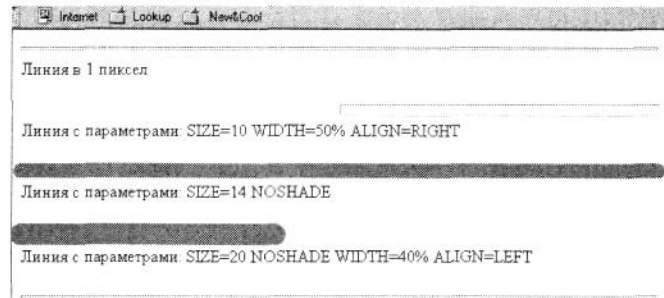


Рисунок 6.9. Управление формой и размещением разделительной линии

### 6.6.2. Графические изображения

Графические изображения в веб-документах формируются в процессе реализации комплекса операций. Такие операции становятся более ясными, если показать содержание таких понятий, как *черезстрочный показ*, *прозрачность*, *анимация*. *Черезстрочный показ* (Interlacing) означает, что изображения разделяются на строки, а само изображение хранится в виде нескольких фрагментов, содержащих информацию о кратных строках. По мере считывания изображения браузер прорисовывает его за несколько проходов (кратных числу фрагментов).

*Прозрачность* (Transparency) определяет состояние документа, при котором изображение фона документа распространяется и на поле установленного рисунка. В непрозрачном формате (GIF) окно рисунка выделено на поле документа.

*Анимация* создает условия для хранения в одном файле нескольких кадров и воспроизведения каждого из них на поле рисунка через заданный временной промежуток.

В World Wide Web применяют два формата для хранения графических данных: GIF и JPEG.

Формат **GIF** (Graphics Interchange Format, формат обмена графическими данными) является стандартом для хранения файлов с изображениями. Графика в этом стандарте ограничивается 256 цветами (вернее, оттенками). Это число говорит о том, что стандарт целесообразно применять для искусственно созданных изображений: эмблем, значков и т. п., но не следует его использовать для хранения данных цветных фотографий.

Формат **JPEG** (Joint Photographic Experts Group, Объединенная группа экспертов по изображениям) определяет набор стандартов, используемых для поддержки полноцветных изображений, сохраняя их в сжатой форме.

Для помещения изображения на поле веб-страницы следует использовать тег **<IMG>**. Его формат: **<IMG SRC="URL">**, где URL — адрес файла, содержащего графические данные;

**SRC** — атрибут, указывающий URL-адрес файла с изображением.

Атрибуты габаритов рисунка **WIDTH** и **HEIGHT** определяют ширину и высоту рисунка в пикселях. Они позволяют браузеру еще в процессе загрузки файла рисунка оставить на поле необходимое место.

Атрибут размещения рисунка по полю текста **ALIGN** позволяет разместить текст относительно поля рисунка, выравнивая его по верху, по середине и по низу изображения (рис. 6.10). Для реализации этого



### 6.6.3. Указатели ссылок

Для создания гиперссылки на поле рисунка следует установить URL-адрес ссылки непосредственно перед тегом графического элемента. Формат графической гиперссылки:

```

```

```

```

```

```

Для примера использования различных рисунков в качестве кнопок меню покажем фрагмент веб-страницы Центра дистанционного обучения (рис. 6.12).

### 6.6.4. Графические элементы на полях таблиц

В целях организации более информативных документов, отличающихся ярко выраженной индивидуальностью, активно используют индивидуальные графические элементы при составлении списков или таблиц.

Графический элемент можно вставить в ячейку таблицы или установить перед соответствующей строкой списка. В последнем случае рисунок определяется как графический маркер.



Рисунок 6.12. Пиктограммы в качестве кнопок меню веб-страницы

В качестве примера приведем листинг HTML-документа, который реализует документ в форме списка, представленного на рис. 6.13:

```
<HTML>
```

```
<HEAD></HEAD>
```

```
<BODY>
```

```
 <U>Отделение "Библиотекосведение"
Международной академии информатизации </U>
```

```
<BLOCKQUOTE>Основная задача — координация фундаментальных научных исследований для
решения проблем функционирования библиотек в современных условиях</BLOCKQUOTE>
```

```
 <U> Научное направление "Документные
коммуникационные системы"</ U>
```

```
<BLOCKQUOTE>Проведение научных исследований в области разработки методов организации
информационного пространства современных организаций</BLOCKQUOTE>
```

```
</BODY>
```

```
</HTML>
```

При установке рисунков в качестве маркеров могут возникнуть проблемы, некоторые из которых хорошо видны на рис. 6.13. Основные из них:

- ◆ текст элемента списка не табулируется относительно маркера;
- ◆ центр графического изображения не выровнен по центру текстовой строки;
- ◆ при превышении длины текстового элемента списка он выравнивается так, что вторая строка начинается под графическим изображением.



**Отделение «Библиоковедение»  
Международной академии информатизации**

Основная задача — координация фундаментальных научных исследований для решения проблем функционирования библиотек в современных условиях



**Научное направление «Документные коммуникационные системы»**

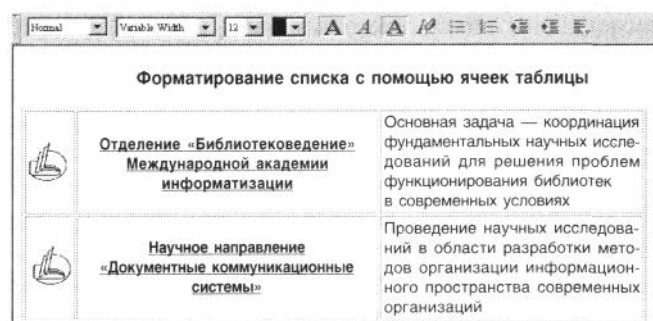
Проведение научных исследований в области разработки методов организации информационного пространства современных организаций

**Рисунок 6.13.** Организация списка с использованием рисунка в качестве маркера

Для исключения указанных недостатков используют специальный прием, позволяющий форматировать список с помощью ячеек таблицы. Покажем метод реализации этого приема на примере переработанного текста предыдущей задачи:

```
<HTML>
<HEAD></HEAD>
<BODY>
<CENTER>
<H3>Форматирование списка с помощью ячеек таблицы</H3> </ CENTER>
<TABLE BORDER >
<TR>
<TD></TD>
<TD><CENTER><U>Отделение "Библиоковедение" Международной академии информатизации</U></CENTER></TD>
<TD>Основная задача — координация фундаментальных научных исследований для решения проблем функционирования библиотек в современных условиях<Д"О>
</TR>
<TR>
<TD> </TD>
<TD><CENTER><U>Научное направление</U> </ CENTER>
<CENTER><U>"Документные коммуникационные системы"</U> </CENTER>
</TD>
<TD>Проведение научных исследований в области разработки методов организации информационного пространства современных организаций
</TD>
</TR>
</TABLE>
</BODY>
</HTML>
```

Результат предложенного метода форматирования показан на рис. 6.14. Здесь линии таблицы приведены только для того, чтобы яснее понять действие ее разделительных линий и внутри ячейечного форматирования на размещение списка. В реальных условиях линии таблицы следует опустить.



**Рисунок 6.14.** Пример форматирования списка с помощью ячеек таблицы

## 6.7. Карты-изображения на поле веб-документа

**Карты-изображения** представляет собой понятие, характеризующее возможность перехода по связям на другие веб-страницы с помощью щелчка по изображению, установленному на поле документа. Применение карт-изображений предусматривает замену текстовых связей на графические. При этом поле документа организуется в виде нескольких графических элементов (последнее и определило использование термина «карта» как совокупности географического изображения государств на плоскости и возможности перехода к соответствующей информации с помощью щелчка по соответствующему фрагменту изображения).

При упрощении понятия рисунка до изображения совокупности клавиш или кнопок с соответствующими надписями с помощью карт-изображений можно организовать различные панели переключателей. Эта идея определяется различными терминами, наиболее распространенным среди которых является **Imagemap** (изображение карты).

Мар-концепция (Imagemap) предполагает использование встроенного в HTML-изображения, для которого определены активные (горячие) точки и области, имеющие ссылки на различные URL-адреса.

Для реализации Мар-концепции используется рисунок (графический файл) и формируется Мар-элемент, который описывает разметку рисунка (указывает его части) и назначает ссылки для соответствующих частей рисунка.

Файл конфигурации карты-изображения (файл описания карты) представляет собой текстовый файл, который содержит информацию об активных областях текущего изображения. Форма области определяется его моделью (типом). В качестве типов (моделей) изображения могут использоваться только: прямоугольник, круг, многоугольник и точка.

Каждая модель определяет соответствующую геометрическую фигуру, координаты которой задаются относительно левого верхнего угла модели:

- ◆ Rest — прямоугольник. Активная область внутри прямоугольника, задаются координаты верхнего левого и правого нижнего углов;
- ◆ Circle — круг. Активная область внутри круга, задаются координаты центра круга и одной из граничных точек;
- ◆ Poly — многоугольник. Активная область внутри многоугольника, задаются координаты вершин многоугольника (до 100 вершин);
- ◆ Point — точка. Активная область вблизи от изображения, но не внутри другой активной области, координаты задаются относительно верхнего левого угла изображения;
- ◆ Default — все неактивные области изображения. Эта область задается по умолчанию.

*Одновременное использование областей Point и Default недопустимо.*

Рассмотрим методику создания графического меню с помощью метода Imagemap, карта которого приведена на рис. 6.15.

Рисунок разработанного поля меню и файл его описания устанавливаются с помощью строчки в теле HTML-документа, как показано на следующем примере:

```
<IMG SRC="primer.jpg" USEMAP="#BUT" ALT="Рабочее меню" HSPACE=80 VSPACE=1
BORDER=0 HEIGHT=75 WIDTH=235>
```

В этой строке приведено имя графического файла: primer.jpg и имя файла разметки активных зон: USEMAP="#BUT" (Элемент MAP). Знак # означает, что текст MAP-файла находится в текущем HTML-документе.

Атрибут карты-изображения ISMAP указывает, что изображение будет использоваться в качестве карты-изображения, располагающейся на сервере.

Элемент MAP имеет следующий формат:

```
<MAP NAME="имя">
<AREA[SHAPE="модель"] CORDS="x,y,...">[HREF="ссылка"] [NOHREF]>
</MAP>
```

Атрибут NAME содержит имя, которое впоследствии идентифицирует элемент в тексте HTML-документа.

Тег <AREA> содержит сведения о форме модели (типа), используемой в качестве кнопки, клавиши и т. п. Моделей может быть несколько.

Атрибут COORDS устанавливает значение координат для размещения модели на поле документа. Если используется модель-прямоугольник («Rest»), то указываются координаты верхнего левого и

правого нижнего углов прямоугольника в виде: «X1,Y1,X2,Y2». Например, для описания области 50x50 следует задать координаты «0,0,49,49».

Атрибут HREF указывает на необходимость вызова из этой области документа, имя которого (ссылка) и задается этим атрибутом.

Атрибут NOHREF определяет, что данная область не является ссылкой.

В Map-элементе можно указать несколько тегов <AREA>, что определит соответствующие области документа. Если указанные области будут иметь пересечения, то та из них, что указана ранее, будет иметь более высокий приоритет.

Обработка MAP-файла состоит в определении типа области и адреса ресурса (URL), который будет возвращен пользователю после щелчка внутри данной области. Продолжая работу над примером, следует записать map-файл (файл описания карты) кнопочного переключателя (см. рис. 6.15):

```
<MAP NAME="BUT">
<AREA SHAPE="RECT" COORDS="1,1,117,37" HREF="PR_1.htm">
<AREA SHAPE="RECT" COORDS="1,38,117,75" HREF="PR_2.htm">
<AREA SHAPE="RECT" COORDS="117,1,37,235" HREF="PR_5.htm">
<AREA SHAPE="RECT" COORDS="117,38,235,75" HREF="PR_6.htm"> </MAP>
```

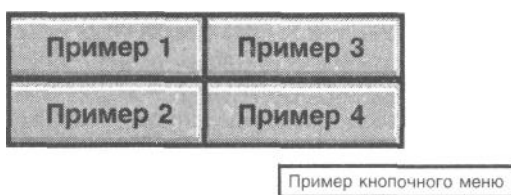


Рисунок 6.15. Карта кнопочного меню

Для использования кнопочного переключателя в поле HTML документа следует щелкнуть мышью по соответствующей области рисунка.

Карты-изображения могут иметь разнообразный вид. В качестве одного из примеров покажем веб-страницу библиотеки Масачусетского технологического института (MIT), приведенную на рис. 6.16.

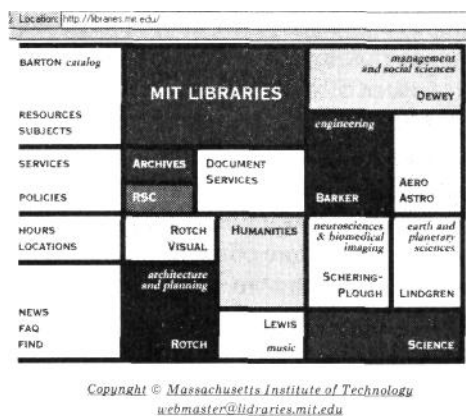


Рисунок 6.16. Фрагмент веб-страницы библиотеки MIT

## 6.8. Организация веб-страниц с использованием фреймов

Веб-страницу документа можно представить в виде совокупности независимых областей, которые определяются как окна или фреймы (frame). Каждый фрейм имеет свои собственные границы, параметры и описывается в собственном HTML-документе.

Фреймы обладают рядом свойств:

- ◆ загружают документ со своим URL-адресом;
- ◆ имеют свое имя;
- ◆ динамически изменяют свой размер в соответствии с изменениями рабочего окна документа.



Применение фрейма на поле документа может быть продиктовано необходимостью решения следующих задач:

1. Установка панели кнопок управления, окна сообщений или заголовка, других элементов, которые могут оставаться на экране, даже если текст прокручивается в других окнах документа.
2. Размещение постоянного оглавления документа, подсказки и т.п. на правой или левой стороне документа.

Фреймы создаются с помощью тегов `<FRAMESET>` и `<FRAME>`, а реализуются посредством `FRAME`-документов.

На первом этапе создания фрейм-документа следует разделить экран на несколько частей.

Фрейм-документ имеет базовую структуру, аналогичную структуре обычного HTML-документа, за исключением того, что теги `<BODY>` и `</BODY>` заменяются на пару тегов:

`<FRAMESET>` тело `</FRAMESET>`.

Тег `<FRAMESET>` может иметь атрибут `ROWS` — для деления окна на несколько полос или `COLS` — для деления окна на несколько колонок. Каждый атрибут указывает список значений, определяющих размеры полос. Размеры задаются в пикселях, в процентах от размера рабочего окна браузера или в виде «\*», что соответствует использованию оставшегося пространства.

Установка содержания фреймов осуществляется с помощью тега `<FRAME>`, который имеет атрибут `SRC`, сообщающий браузеру на адрес загружаемого в окно документа. Кроме этого атрибута, тег `<FRAME>` может использовать другие атрибуты:

- ◆ `MARGINHEIGHT=n` — задает размер пустого пространства над и под фреймом;
- ◆ `MARGINWIDTH=n` — задает размер пустого пространства слева и справа от фрейма;
- ◆ `NAME="name"` — присваивает фрейму уникальное имя, на которое можно ссылаться из других документов;
- ◆ `NORESIZE` — не позволяет пользователю изменить размеры фрейма;
- ◆ `SCROLLING=YES|NO|AUTO` — управляет появлением в фрейме горизонтальных и вертикальных линеек (scrollbars);
- ◆ `SRC="URL"` — задает URL-адрес документа, загружаемого во фрейм;
- ◆ `FRAMEBORDER=n` — задает ширину границы между фреймами.

В качестве примера, позволяющего оценить преимущества установки фреймов, а также овладеть этим методом, приведем текст HTML-документа, устанавливающий четыре равных окна на поле браузера. На поле двух окон выведены документы, содержащие ранее рассмотренные примеры, а на поле другой пары окон — фрагменты справочной системы университета о библиотеке и учебных секциях:

```
<FRAMESET ROWS="50%,50%">
 <FRAMESET COLS="50%,50%">
 <FRAME SRC="PR_7.htm">
 <FRAME SRC="PR_5.htm">
 </FRAMESET>
 <FRAMESET COLS="50%,50%">
 <FRAME SRC="sections.htm">
 <FRAME SRC="library.htm">
 </FRAMESET>
</FRAMESET>
```

Результат организации работы с веб-документами при использовании четырех окон показан на рис. 6.17.

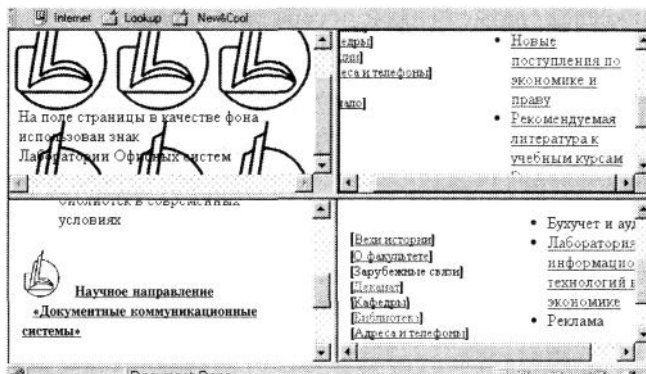


Рисунок 6.17. Установка фреймов на рабочем окне браузера

На нижнем левом окне отсутствует горизонтальная линейка, так как информация соответствующего документа полностью помещается в выделенной области.

С помощью мыши можно перетаскивать по экрану разделительные линии, изменяя габариты окон, увеличивая область окна того документа, с которым необходимо ознакомиться.

В процессе работы с документами, размещенными на полях фреймов, возникает потребность перехода по гиперссылке на поле других веб-документов. В этом случае следует определить размеры нового рабочего окна. Для этого следует в теге гиперссылки <A> использовать параметр TARGET, определяющий имя фрейма или окна браузера. В него будет загружаться текущий документ, на который указывает ссылка. По умолчанию документ загружается в текущий фрейм или окно. Имеются также четыре зарезервированных имени, при задании которых выполняются следующие действия:

TARGET="\_blank" — обеспечивает загрузку документа в новое окно (без имени);

TARGET="\_self" - обеспечивает загрузку документа в текущий фрейм (окно);

TARGET="\_top" — вызывает загрузку документа в распахнутое (полное) окно;

TARGET="\_parent" – загружает документ в область, занимаемую фреймом-родителем текущего фрейма. При отсутствии фрейма-родителя данное значение аналогично действию «\_top».

## ПРАКТИКУМ: ПРИМЕНЕНИЕ ГРАФИКИ И ФРЕЙМОВ ДЛЯ ОРГАНИЗАЦИИ ВЕБ-ДОКУМЕНТОВ

*Цель работы: овладеть навыками размещения графических элементов, применения графических элементов в качестве указателей ссылок, использования карт-изображений на полях HTML-документа, разработка фрейм-документов и др.*

### 1. Подготовка документа для ввода текста.

1.1. Установите режим записи веб-документа и сохраните новый документ в личной папке под именем 4\_3\_ФИО в рабочей папке. 1.2. Запишите в заголовок HTML-кода: HTML-графика и фреймы: ФИО.

1.3. Установите информацию об авторе и дате обновления документа, используя тег ADDRESS.

1.4. Запишите текст по центру первой строки «Страница <Фамилия Имя Отчество>».

1.5. Вызовите браузер и перейдите на поле веб-документа.

### 2. Размещение графических элементов.

2.1. Установите горизонтальную линию:

а) на поле нового документа установите заголовок темы и проведите четыре линии: линия в 1 пиксель, линия толщиной в 10 пикселей в правой половине экрана, линия в 14 пикселей в виде сплошной полосы, сплошная линия толщиной в 20 пикселей на левой половине экрана размером в 40%; под каждой линией запишите ее параметры;

б) отделите выполненное задание зеленой сплошной полосой в 200 пикселей, размещенной посередине экрана;

в) перейдите на окно браузера и проверьте правильность подготовки нового документа.

2.2. Установите изображение на поле страницы:

а) установите товарный знак фирмы, а рядом с ним — имя фирмы и направление ее деятельности: по верхнему краю, по середине, по нижнему краю (в качестве рисунка можно использовать рисунок,

разработанный при изучении методов работы в графическом редакторе Paint или взятый из любой веб-страницы);

б) отделите пример горизонтальной цветной линией;

в) перейдите на окно браузера и проверьте правильность выполнения задания.

2.3. Установите ссылки в форме рисунков:

а) запомните адреса трех веб-страниц (например, университета, факультета, кафедры);

б) установите таблицу размером 2x3 без разделительных линий и разместите в ячейках первой строки краткое наименование веб-страниц;

в) на второй строке в соответствующие ячейки введите рисунки соответствующих веб-страниц;

г) свяжите изображения с гиперссылкой, используя конструкцию: <AHREF="URL-адрес"> <IMG SRC="имя картинки"X/A>.

2.4. Установите рамку вокруг изображения товарного знака (п. 2.2), используя следующую конструкцию: <IMG SRC=label.gif BORDER= n>. Проверьте правильность выполненной разметки.

### 3. Применение фреймов на веб-странице.

3.1. Разработайте фрейм-документ:

а) откройте новый HTML-документ, в котором замените теги <BODY> на теги <FRAMESET>;

б) сформируйте оконную веб-страницу, разместив на ней четыре различные веб-страницы, используя следующую конструкцию:

```
<FRAMESET ROWS="50%,50%">
```

```
<FRAMESET COLS="50%,50%">
```

```
<FRAME SRC="3_1_OHO.htm">
```

```
<FRAME SRC="4_1_OMO.htm">
```

```
</FRAMESET>
```

```
<FRAMESET COLS="50%,50%">
```

```
<FRAME SRC="5_1_0>HO.htm">
```

```
<FRAME SRC="Label.gif.htm"> — рисунок товарного знака фирмы
```

```
</FRAMESET>
```

```
</FRAMESET>
```

3.2. Подготовьте окно для личной страницы:

а) откройте новый документ и установите на его поле три фрейма: первый — во всю ширину листа (1), второй (2) и третий (3) — рядом друг с другом под первым;

б) на поле первого фрейма установите логотип Вашей фирмы\*, на поле второго — список продуктов фирмы, а на поле третьего — выбранный графический файл;

---

\* Для организации первого фрейма следует использовать конструкцию <FRAMESET COLS=100%,0%>.

в) перейдите на окно браузера и проверьте правильность подготовки нового документа.

### 4. Ссылка на ресурсы Интернета.

4.1. Установите на поле третьего окна (п. 3.2) ссылку на электронную почту, используя следующую конструкцию: <A HREF="mailto:адрес почты">Адрес для отзывов</A>.

4.2. Перейдите на окно браузера, нажмите на кнопку **Обновить**. Отправьте электронное сообщение.

### 5. Переход на новые веб-страницы из окон фреймов.

5.1. Подготовьте четыре HTML-документа, на каждом из которых опишите соответствующие основные параметры выпускаемой продукции.

5.2. Установите гиперссылки на подготовленные HTML-документы в списке продукции, установленном на поле второго фрейма в упражнении 3.26, выбирая для каждой ссылки новое значение параметра TARGET.

5.3. Проверьте результат использования параметра TARGET.

6. Сохраните HTML-коды документов, проверьте их на вирусы и запишите на дискету.

7. Закройте все приложения на рабочем столе.

## СЛОВАРЬ ТЕРМИНОВ

### А

Адрес

Номер байта информации, размещенного в оперативной памяти

Адресная книга	Книга адресов электронной почты
Алгоритм	Система формальных правил, четко и однозначно определяющая порядок выполнения операций (команд)
Ассемблер	Язык программирования на уровне описания команд машинного языка

<b>Б</b>	
База данных	Организованная последовательность записей, состоящих из постоянного набора данных
Байт	Единица для записи информации. Один байт размещается в 8 битах
Бит	Единица двоичной информации
Блокнот	Приложение Windows, предназначенное для записи текста
Браузер	Программа, предназначенная для просмотра веб-документов
Буфер обмена	Область памяти, куда помещается информация для временного хранения
Бэйсик	Язык программирования общего назначения, характеризующийся относительно простой структурой

<b>В</b>	
Веб-страница	Совокупность текстовых, графических, табличных и др. объектов, организованная на рабочем окне браузера и предназначенная для отображения информации в сервисе WWW Интернета
Ветвление	Команда выбора продолжения выполнения программы, принимающая решение в зависимости от выполнения некоторого условия
Видеоадаптер	Электронная схема, управляющая размещением пикселей на экране дисплея и размещенная, как правило, на отдельной видеокарте
Видеокарта	Электронная схема видеоадаптера, размещенная на пластике и имеющая разъем для установки на системную плату компьютера
Винчестер	Устройство хранения информации (памяти) на жестких магнитных дисках
Внешняя память	Устройство компьютера, позволяющее хранить информацию при его отключении (обычно оно характеризуется значительно большим объемом памяти)
Время системное	Приложение, устанавливаемое на панели задач и обеспечивающее выдачу текущего времени: часы и минуты. При раскрытии кнопки времени можно просмотреть дату, день и год
Выражение	Правая часть оператора, содержащая последовательность операндов и выполняемой(ых) операции(й). Различают арифметические, строковые, логические и др.

<b>Г</b>	
Гипертекст	Документ, снабженный гиперссылками
Гипертекстовая ссылка	Средство связи текущего документа с другими документами. Состоит из указателя и адреса
Главное меню	Меню приложений, раскрываемое после щелчка мышью по кнопке Пуск

Группа новостей	Сервис Интернета
<b>Д</b>	
Данные	Информация, которая используется программой в процессе ее выполнения
Дерево	Структура отношений между объектами, при которой установлена процедура однозначного перехода от одного уровня иерархии к другому, и обратно
Джава (Java)	Язык интерактивного веб-программирования
Диапазон	Область применимых значений
Диск	Устройство внешней памяти, размещенной на тонкой пластине с полупроводниковым напылением для записи и считывания информации. Совокупность жестких дисков образует винчестер
Дискета	Диск на виниловой основе с нанесенным на него магнитным слоем и встроенный в жесткий пластмассовый корпус, предназначенный для физического переноса информации. Иногда ее определяют как флоппи-диск
Дисковод	Устройство для считывания информации с дискет или дисков
Дисплей	Устройство отображения информации компьютера, реализованное на основе ЭЛТ, ЛСД или плазменного экрана
Документ	Файлы, содержащие информацию в форме текста, таблиц, графики, и т. п.
Домашняя страница	Страница, устанавливаемая на окне браузера в момент его загрузки
<b>З</b>	
Загрузка	Процесс обнаружения, размещения в оперативной памяти и запуск на выполнение текущей программы
Звуковая карта	Электронная схема, размещенная на твердой основе, устанавливаемая на системной плате компьютера и предназначенная для преобразования цифрового кода в звук
Значок	Графическое обозначение документа
<b>И</b>	
Идентификатор	Обозначение переменной (имя переменной)
Имя (в Паскале)	Любая последовательность цифр, латинских знаков и знака подчеркивания, не начинающаяся с цифры
Индекс	Цифровое обозначение переменной
Инициализация	Переключение режимов работы компьютера, например из текстового в графический, и наоборот
Интернет	Глобальная компьютерная сеть, обеспечивающая доступ своих пользователей к основным сервисам: доступ к веб-страницам, электронной почте, телеконференциям и др. на бесплатной основе
Интерфейс (техническое средство)	Технические средства, обеспечивающие взаимосвязь различных устройств и компонентов компьютера

Интерфейс пользователя	Средства обмена информацией между программой и оператором в процессе выполнения программы
<b>К</b>	
Карты-изображения	Средство графического отображения гиперсвязей
Каталог	Упорядоченная группа файлов и вложенных каталогов в файловом пространстве компьютера, снабженная именем
Клавиатура	Устройство ввода информации в компьютер с помощью символов, цифр и клавиш управления
Код	Совокупность цифр, определяющая функциональное действие (клавиши, группы клавиш)
Код машинный	Последовательность строк машинных команд и адресов ячеек, над которыми выполняется текущая операция
Команда	Инструкция, воспринимаемая компьютером для реализации определенных действий
Комментарий	Любые пояснительные тексты в программе, предназначенные для понимания процесса, но не выполняемые компьютером
Компакт-диск	Средство для организации внешней памяти компьютера, представляющее собой диск с металлизированной поверхностью, на которой лазерным лучом записывается и считывается информация
Компиляция	Процесс построчного преобразования текста программы, написанного на языке программирования в машинный код, выполняемый компьютером
Компьютер	Электронно-вычислительное устройство, предназначенное для поиска, хранения и преобразования информации под управлением программного обеспечения
Константа	Постоянная величина
Контекстное меню	Меню группы команд, вызываемое с помощью щелчка правой кнопки мыши по полю окна приложения или его вкладки
Копирование	Процесс перемещения фрагмента программы (данных, текста и т. п.) в тексте программы; перемещение файлов и каталогов в файловом пространстве компьютера
Курсор	Символ, отражающий на экране местоположение указателя мыши
<b>Л</b>	
Локальная сеть	Компьютеры, соединенные специальным кабелем и поддерживаемые соответствующей операционной системой для высокоскоростного обмена информацией
<b>М</b>	
Массив	Упорядоченная совокупность данных, переменных и т. п.
Машинный код	См. <i>Код машинный</i>
Метка	Произвольное имя, определенное в поле переменных и используемое для указания места, к которому должна перейти программа

Модем	Устройство модуляции-демодуляции аналогового сигнала (из цифры в код, и наоборот) для передачи цифрового сообщения, которое поступает от компьютера по аналоговым (телефонным) сетям, и обратно
Модуль	Отдельные программы, позволяющие реализовать определенные функции. Стандартные модули Паскаля реализуют функции Паскаля
Монитор	Устройство отображения видеoinформации компьютера
Мышь	Устройство перемещения и активизации курсора по экрану монитора компьютера
<b>Н</b>	
Неразрывный пробел	Тег языка HTML, запрещающий запись информации в текущую область таблицы
<b>О</b>	
Объект	Совокупность данных и процедур, которые эти объекты обрабатывают, например указание на геометрию и цвет рабочего окна на экране, процессы его перемещения по экрану, изменения размеров и др. процедуры и функции, необходимые для работы с ним
Окно	Область экрана, выделенная границами и имеющая строку имени и кнопки управления
Оператор	Команды, записанные на соответствующем языке программирования
Операционная система	Комплекс программ, обеспечивающий взаимосвязь всех устройств компьютера
Операция логическая	Операция, использующая в своей структуре логический оператор условия
Отладка	Процесс редактирования хода выполнения программы, заключающийся во внесении исправлений и дополнений в текст программы
<b>П</b>	
Память	Средство для хранения информации; разделяют на оперативную и внешнюю
Панель быстрых связей	Панель, раскрываемая кнопкой Ссылки, размещенной на панели инструментов, и содержащая список заранее установленных веб-адресов
Панель закладок	См. <i>Панель быстрых связей</i> (используется в NSC)
Папка	Каталог в структуре файлового пространства компьютера
Параметр	Совокупность данных, определяющая условия выполнения процедуры, функции отдельной программы и т. п.
Паскаль	Универсальный язык программирования, ориентированный на разработку программ для решения научных и технических задач
Переменная	Обозначение аргумента или функции, принимающее различные значения
Пиксель	Светящаяся точка на экране

Полоса прокрутки	Средство перемещения поля документа в рабочем окне приложения, снабженное стрелками построчного перемещения, кнопкой свободного перемещения и кнопками перехода к началу и концу документа
Почтовое сообщение	Краткое текстовое или графическое сообщение, переданное по электронной почте
Почтовый ящик	Файловое пространство, размещенное на компьютере провайдера, предназначенное для хранения поступающих в адрес абонента почтовых сообщений
Принтер	Устройство вывода информации на бумагу
Провайдер	Организация-посредник, обеспечивающая доступ абонента к Интернету
Программа	Упорядоченная совокупность команд, написанная на языке программирования
Процедура	Отдельная программа, выполняемая при ее вызове из программы
Процессор	Устройство компьютера, обеспечивающее выполнение арифметико-логических операций
<b>Р</b>	
Рабочий стол Windows	Экран компьютера, снабженный набором значков и ярлыками программ или документов, а также панель программ с кнопкой Пуск
Раздел	Часть программы, содержащая или описание переменных, меток, процедур и т. п., или операторы программы
Редактор	Специальная программа, предназначенная для работы с текстом программы или документа. Различают также табличные, графические и др. редакторы
Режим работы компьютера	Режим, предусматривающий обработку графической информации или текстовой информации
Режим работы редактора	Организация работы редактора для реализации соответствующих операций, например режим вставки, режим замещения и т. п.
<b>С</b>	
Сетевой протокол	Машинный язык сети, определяющий процессы межсетевого обмена информацией. Он содержит, например, IP-адрес, способ разделения сообщения на пакеты (TCP)
Сеть	Группа компьютеров, объединенная между собой для обмена информацией
Си	Объектно-ориентированный язык программирования, предназначенный для разработки сложных информационных систем
Синтаксис	Часть грамматики, описывающая правила записи текста
Системный блок	Совокупность устройств компьютера, оформленная в виде самостоятельной конструктивной единицы (размещенная в отдельном металлическом корпусе)
Сканер	Прибор, предназначенный для считывания информации с листа бумаги



Сортировка	Упорядочение информации в соответствии с указанным принципом сортировки, например по возрастанию, убыванию, в произвольном порядке и др.
Специальные символы	Символы клавиатуры, используемые в языке HTML для разметки веб-документа
Списки рассылки	Почтовые адреса абонентов, по которым рассылаются корпоративные новости и др. сообщения
Страница поиска	Адрес поисковой системы, за которым закрепляется специальная кнопка, размещенная на панели инструментов браузера
Сумматор	Блок процессора, обеспечивающий выполнение арифметических операций в компьютере
Счетчик	Электронная схема, обеспечивающая подсчет сигналов. Переменная величина, в которой хранится результат счета, например число прохождения цикла
<b>Т</b>	
Табличный тег	Инструкция в языке HTML, определяющая начало разметки таблицы
Тег	Метка, позволяющая отделить команды разметки от элементов текста документа. Тег сообщает браузеру, каким образом следует представить на экране следующий фрагмент документа
Телеконференция	Сервис Интернета, позволяющий обмениваться информацией отдельной группе абонентов
Тип данных	Характеристика переменных, определяющая множество допустимых операций над ними и формат их представления в компьютере
Трансляция	Комплексное преобразование текста программы, написанного на языке программирования, в машинный код
<b>У</b>	
Указатель ссылки	Текст, слово, рисунок и т. п., предназначенные для перехода на другой документ
Устройство ввода	Устройство ввода информации в компьютер: с магнитных лент — стример, с кассовых аппаратов и т. п.
Устройство вывода	Устройство вывода на различные носители информации: принтер, плоттер, кассовые аппараты, экраны и т. п.
Устройство запоминающее	Устройство хранения информации: оперативное, внешнее
Устройство периферийное	Общее обозначение для любых типов внешних устройств
<b>Ф</b>	
Файл	Область дискового пространства, обозначенная именем и предназначенная для размещения программ, данных и др. информации
Фрейм	Рамка, область рабочего окна браузера, предоставленная для вывода веб-документа



11. *Фаронов В.В.* Турбо Паскаль 7.0. Практика программирования. Учеб. пособие. — 7-е изд., перераб. — М.: Нолидис, 2001. — 416 с.
12. *Фигурнов В.В.* IBM для пользователя. Краткий курс. — Сокращенная версия 7-го изд. — М.: ИНФРА-М, 2001. — 779 с.
13. MS DOS: Справочное руководство для пользователей компьютеров IBM PC. — М.: ВА ПРИНТ, 1994. — 363 с.

## Содержание

Введение.....	3
От автора .....	6
Глава 1. ОСНОВНЫЕ КОМПОНЕНТЫ КОМПЬЮТЕРНОЙ ТЕХНОЛОГИИ .....	8
1.1. Компьютерные технологии и информационное пространство .....	8
1.2. Компьютер: структура и параметры.....	9
1.2.1. Структурная схема компьютера.....	9
1.2.2. Дополнительные устройства ввода-вывода информации .....	10
1.2.3. Классификация компьютеров.....	12
1.2.4. Сети передачи данных.....	12
1.3. Программное обеспечение компьютера .....	13
1.3.1. Классификация программного обеспечения .....	13
1.3.2. Файловая система хранения информации .....	14
1.4. Дисковая операционная система MS DOS .....	16
1.4.1. Состав и назначение MS DOS .....	16
1.4.2. Распределение памяти в MS DOS .....	16
1.4.3. Файловая структура диска.....	16
1.4.4. Утилиты MS DOS .....	17
1.4.5. Драйверы устройств в MS DOS .....	20
1.5. Выполнение команд MS DOS .....	21
1.6. Программа-оболочка Norton Commander .....	23
Практикум: работа в среде MS DOS.....	26
Глава 2. ГРАФИЧЕСКАЯ ОПЕРАЦИОННАЯ СИСТЕМА WINDOWS XP .....	27
2.1. Рабочий стол операционной с и с т е м ы .....	28
2.1.1. Основные элементы рабочего стола .....	29
2.1.2. Оформление рабочего стола.....	36
2.1.3. Управление объектами на рабочем столе .....	38
2.2. Панель управления .....	39
2.3. Построение файловой системы.....	41
2.3.1. Форматирование диска.....	41
2.3.2. Управление папками.....	42
2.4. Настройка панели задач .....	44
2.5. Настройка меню кнопки Пуск .....	45
2.6. Приложение Проводник.....	46
2.7. Приложение Корзина .....	48
2.8. Приложение Портфель.....	49
2.9. Приложение Блокнот.....	50
2.10. Приложения Назначенные задания и Дата и время .....	50
2.11. Приложение Paint .....	51
2.11.1. Основные элементы графического редактора Paint .....	52
2.11.2. Системное меню графического редактора Paint.....	53
2.11.3. Инструменты графического редактора Paint .....	55
2.11.4. Вставка графических изображений.....	57
Практикум: Работа в графической операционной системе семейства WINDOWS.....	58
Глава 3. ОСНОВЫ ПРОГРАММИРОВАНИЯ НА ТУРБО ПАСКАЛЬ 7.0.....	66
3.1. Процесс программирования.....	66
3.2. Интегрированная инструментальная оболочка Турбо Паскаля.....	68

3.2.1. Опции главного меню.....	69
3.2.2. Назначение функциональных клавиш .....	72
3.2.3. Текстовый редактор.....	73
3.2.4. Операции.....	74
3.3. Язык программирования Турбо Паскаль 7.0.....	74
3.3.1. Лексика языка .....	75
3.3.2. Грамматика языка .....	77
3.4. Операторы.....	78
3.4.1. Операторы ввода и вывода данных.....	78
3.4.2. Простой оператор перехода Goto.....	79
3.4.3. Структурированные операторы .....	80
3.6. Типы данных.....	85
3.5.1. Простые типы данных .....	85
3.5.2. Структурированные типы данных Тип-массив .....	87
3.6. Процедуры и функции.....	98
3.7. Модуль в Турбо Паскале.....	104
3.7.1. Библиотечный стандартный модуль Crt .....	106
3.7.2. Библиотечный стандартный модуль Graph .....	111
3.7.3. Библиотечный стандартный модуль System.....	130
3.8. Разработка информационных систем для деловой практики .....	132
Практикум: Программирование в Турбо Паскале .....	140
Глава 4. ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ .....	209
4.1. Работа в программе Norton Commander.....	209
4.2. Работа в графической среде Paint .....	210
4.3. Основные команды интегрированной среды Турбо Паскаля .....	210
4.4. Оконный интерфейс в программах пользователя .....	212
4.5. Программы линейной структуры.....	213
4.6. Программы разветвляющейся структуры.....	213
4.7. Организация циклов .....	213
4.8. Обработка многомерных массивов данных.....	214
4.9. Обработка символьных данных .....	214
4.10. Обработка строковых данных .....	214
4.11. Обработка данных типа «Запись».....	214
4.12. Типизированные и текстовые файлы.....	215
4.13. Процедуры и функции.....	215
4.14. Обработка массивов данных .....	216
4.15. Строки и записи.....	216
Глава 5. ИНФОРМАЦИОННЫЙ СЕРВИС ИНТЕРНЕТА .....	216
5.1. Структура глобальной сети.....	216
5.1.1. Информационные службы сети.....	216
5.1.2. Сетевые протоколы.....	217
5.2. Программы просмотра документов информационных сетей .....	219
5.2.1. Оконный интерфейс браузера Netscape Communicator .....	219
5.2.2. Оконный интерфейс браузера MS Internet Explorer .....	222
5.3. Электронная почта .....	225
5.3.1. Установка почтового ящика.....	225
5.3.2. Редактирование файла настроек .....	227
5.3.3. Структура электронного письма .....	228
5.3.4. Обработка почтовых сообщений.....	231
5.3.5. Работа с адресной книгой.....	233
ПРАКТИКУМ: ТЕХНОЛОГИЯ ПРИМЕНЕНИЯ ЭЛЕКТРОННОЙ ПОЧТЫ.....	234
5.4. Работа с веб-документами.....	236
5.4.1. Методы доступа к веб-документам.....	236
5.4.2. Установка страницы поиска.....	241
5.4.3. Копирование веб-страниц .....	241

5.5. Участие в телеконференциях .....	242
5.5.1. Структура службы .....	242
5.5.2. Чтение новостей.....	243
5.5.3. Чтение статей телеконференций .....	244
5.5.4. Просмотр документов телеконференций.....	245
5.5.5. Отправка документов на телеконференцию .....	246
5.6. Поисковые системы и каталоги ресурсов.....	247
5.6.1. Желтые страницы российского сегмента Интернета .....	247
5.6.2. Наука и техника .....	248
5.6.3. Образование .....	250
5.6.4. Интернет-магазин .....	251
5.6.5. Виртуальные библиотеки .....	252
5.7. Интернет-объединение информационных сетей России .....	253
5.7.1. Национальные провайдеры Интернета.....	254
5.7.2. Информационные ресурсы Интернета.....	255
ПРАКТИКУМ: ПОИСК И ОБРАБОТКА ИНФОРМАЦИИ В ИНТЕРНЕТЕ .....	255
Глава 6. HTML — ЯЗЫК ГИПЕРТЕКСТОВОЙ РАЗМЕТКИ.....	257
6.1. Структура HTML-языка .....	258
6.1.1. Обозначение HTML-документа .....	259
6.1.2. Заголовок документа .....	259
6.1.3. Тело документа .....	260
6.2. Запись текста веб-документа .....	260
6.2.1. Начало абзаца и конец строки.....	260
6.2.2. Стили заголовков.....	260
6.2.3. Физические стили .....	261
6.2.4. Логические стили.....	261
6.2.5. Преформатированный текст.....	262
6.2.6. Списки.....	262
6.2.7. Специальные символы.....	263
6.3. Организация гипертекстовых ссылок.....	263
6.3.1. Формирование указателей.....	264
6.3.2. Создание оглавления с помощью имен указателей.....	264
6.3.3. Применение гиперссылок для формирования оглавления текущего веб-документа .....	264
ПРАКТИКУМ: ФОРМИРОВАНИЕ ТЕКСТА ВЕБ-ДОКУМЕНТА.....	264
6.5. Установка табличных элементов .....	268
6.5.1. Основные табличные теги.....	268
6.5.2. Размещение сложных таблиц .....	269
6.5.3. Совмещение текста и таблицы на поле документа .....	271
ПРАКТИКУМ: СПИСКИ И ТАБЛИЦЫ НА ПОЛЕ ВЕБ-СТРАНИЦ .....	272
6.6. Размещение графических элементов .....	274
6.6.1. Горизонтальные линии.....	274
6.6.2. Графические изображения .....	275
6.6.3. Указатели ссылок.....	277
6.6.4. Графические элементы на полях таблиц .....	277
6.7. Карты-изображения на поле веб-документа.....	279
6.8. Организация веб-страниц с использованием фреймов .....	280
ПРАКТИКУМ: ПРИМЕНЕНИЕ ГРАФИКИ И ФРЕЙМОВ ДЛЯ ОРГАНИЗАЦИИ ВЕБ-ДОКУМЕНТОВ.....	282
Словарь терминов .....	283
Рекомендуемая литература.....	290

*Учебное издание*

**Меняев Михаил Федорович**

## ИНФОРМАТИКА

Учебное пособие

Главный редактор *В.П. Соколова* Ведущий редактор *И.Л. Августин*

Редактор *Ю.А. Серова*

Корректор *А.В. Бенецкая*

Компьютерная верстка *М.В. Сенотрусовой*

Подписано в печать 16.05.03. Формат 60 х 90 Печать офсетная. Гарнитура «Times» Печ. л. 29. Тираж 6000 экз. Заказ я-34

ООО «Омега-Л». Почтовый адрес:

123022, г. Москва, Столярный пер., д. 14, п.2

Тел. (095) 253-46-82

ФГУП «Издательство «Высшая школа», 127994, Москва, ГСП-4, Неглинная ул., 29/14. Тел.: (095)

200-04-56 E-mail: [info@v-schkola.ru](mailto:info@v-schkola.ru) <http://www.v-schkola.ru>

Отдел продаж: (095) 200-07-69. 200-59-39, факс: (095) 200-03-01. Отдел «Книга — почтой»: (095) 200-33-36 E-mail: [bookpost@v-schkola.ru](mailto:bookpost@v-schkola.ru)

Отпечатано в типографии ГУП ПИК «Идел-Пресс» в полном соответствии с качеством предоставленных диапозитивов.

420066, г. Казань, ул. Декабристов, 2.