

Учреждение образования
«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ
УНИВЕРСИТЕТ»

А. П. Лащенко, Т. П. Брусенцова

ИНФОРМАТИКА И КОМПЬЮТЕРНАЯ ГРАФИКА

*Рекомендовано
учебно-методическим объединением
высших учебных заведений Республики Беларусь
по образованию в области природопользования и лесного хозяйства
в качестве учебно-методического пособия
для студентов высших учебных заведений, обучающихся по специальностям
1-36 05 01 «Машины и оборудование лесного комплекса»,
1-46 01 01 «Лесоинженерное дело»,
1-46 01 02 «Технология деревообрабатывающих производств»,
1-08 01 01 «Профессиональное обучение»*

Минск 2008

УДК 00.4(075.8)

ББК 73я7

Л 32

Рецензенты:

кафедра методов оптимального управления БГУ (зав. кафедрой доктор физико-математических наук, профессор *А. И. Калинин*); доцент кафедры информационных технологий автоматизированных систем БГУИР, кандидат технических наук *О. В. Герман*

Все права на данное издание защищены. Воспроизведение всей книги или ее части не может быть осуществлено без разрешения учреждения образования «Белорусский государственный технологический университет».

Лащенко, А. П.

Л 32 Информатика и компьютерная графика : учеб.-метод. пособие для студентов специальностей 1-36 05 01 «Машины и оборудование лесного комплекса», 1-46 01 01 «Лесоинженерное дело», 1-46 01 02 «Технология деревообрабатывающих производств», 1-08 01 01 «Профессиональное обучение» / А. П. Лащенко, Т. П. Брусенцова. – Минск : БГТУ, 2008. – 190 с.

ISBN 978-985-434-827-8

Пособие содержит основные понятия, методические указания для проведения лабораторных работ и подготовки студентов к экзамену по дисциплине «Информатика и компьютерная графика». Оно посвящено базовым вопросам компьютерной обработки информации и включает пять самостоятельных разделов. Все разделы сопровождаются конкретными практическими примерами, которые облегчают усвоение материала студентами.

УДК 00.4(075.8)

ББК 73я7

ISBN 978-985-434-827-8

© УО «Белорусский государственный технологический университет», 2008
© Лащенко А. П., Брусенцова Т. П., 2008

ПРЕДИСЛОВИЕ

Предмет «Информатика и компьютерная графика» является дисциплиной общеобразовательного цикла. Коренное отличие информатики от других технических дисциплин, изучаемых в высшей школе, состоит в том, что ее предмет изучения меняется ускоренными темпами. Сегодня количество компьютеров в мире превышает 800 миллионов единиц и продолжает удваиваться в среднем каждые три года. При этом каждая вычислительная система по своему уникальна. Найти две системы с абсолютно одинаковыми аппаратными и программными конфигурациями весьма сложно, и поэтому для эффективной эксплуатации вычислительной техники, в основе которой находится персональный компьютер (ПК), от специалистов требуется достаточно широкий уровень знаний и практических навыков работы с ПК. Изучение основ современных информационных технологий с использованием компьютерных технологий является одним из важнейших этапов при подготовке современного инженера. Это обусловлено стремительным внедрением новейших информационных технологий во все сферы человеческой деятельности. Квалифицированный инженер должен владеть методами алгоритмизации задач в области своей профессиональной деятельности и уметь реализовывать эти алгоритмы и задачи отрасли в различных интегрированных средах.

Для решения сложных расчетных задач используют программы, написанные специально. В то же время в научной работе встречается широкий спектр задач ограниченной сложности, для решения которых можно использовать универсальные средства.

Целью курса, разработанного в соответствии с образовательными стандартами Республики Беларусь для соответствующих специальностей, является изучение методов подготовки и решения задач на современных ПЭВМ, методов работы с современными пакетами прикладных программ.

Результатом изучения курса для студента должно стать знание современных технических средств взаимодействия с ПЭВМ, основ компьютерных информационных технологий, методов автоматизации обработки документов и представления их в электронном виде, методов решения задач на ПЭВМ в различных режимах, умение создавать Web-документы, работать с основными службами глобальной компьютерной сети, с наиболее распространенными программами MS Office.

1. ОПЕРАЦИОННЫЕ СИСТЕМЫ

Операционная система (ОС) – это комплекс взаимосвязанных системных программ, назначение которого – организация взаимодействия пользователя с компьютером и выполнение всех других программ. В программном обеспечении операционная система занимает основное положение, поскольку осуществляет планирование и контроль всего вычислительного процесса. Любая из компонент программного обеспечения обязательно работает под управлением ОС. Операционная система играет роль связующего звена между аппаратурой компьютера и выполняемыми программами, с одной стороны, а также пользователем, с другой. Операционная система обычно хранится во внешней памяти компьютера – на диске. При включении компьютера она считывается с дисковой памяти и размещается в ОЗУ. Этот процесс называется загрузкой операционной системы.

Каждая ОС состоит как минимум из трех обязательных частей.

Первая – ядро, командный интерпретатор, «переводчик» с программного языка на «железный», язык машинных кодов.

Вторая – специализированные программы для управления различными устройствами, входящими в состав компьютера. Эти программы называются драйверами, т. е. «водителями», управляющими. Сюда же относятся так называемые «системные библиотеки», используемые как самой операционной системой, так и входящими в ее состав программами.

Третья часть – удобная оболочка, с которой общается пользователь – интерфейс. Сегодня графический интерфейс – неизменный атрибут любой операционной системы, будь то Windows 98/ME, Windows NT/2000 или MacOS – операционная система для компьютеров Apple Macintosh. Но операционные системы первых поколений имели не графический, а текстовый интерфейс, т. е. команды компьютеру отдавались не щелчком мышки по рисунку-пиктограмме, а с помощью введения команд с клавиатуры.

Операционные системы можно разделить на группы (классифицировать) по следующим признакам:

1. По количеству пользователей: однопользовательские (обслуживает только одного пользователя), многопользовательские (работает со многими пользователями).

2. По числу процессов: однозадачные (обрабатывают только одну задачу – уже не используются), многозадачные (располагает в опе-

ративной памяти одновременно несколько задач, которые попеременно обрабатывает процессор).

3. По типу средств вычислительной техники: однопроцессорные, многопроцессорные (задачи могут выполняться на разных процессорах; серверы, как правило, многопроцессорные), сетевые (обеспечивают совместное использование ресурсов всеми выполняемыми в сети задачами).

В соответствии с условиями применения различают три режима ОС: пакетной обработки, деления времени и реального времени.

В режиме пакетной обработки ОС последовательно выполняет собранные в пакет задания. В этом режиме пользователь не имеет контакта с ЭВМ, а получает лишь результаты вычислений. В режиме деления времени ОС одновременно выполняет несколько задач, допуская обращение каждого пользователя к ЭВМ. В режиме реального времени ОС обеспечивает управление объектами в соответствии с принимаемыми входными сигналами. Время отклика ЭВМ с ОС реального времени на возмущающее воздействие должно быть минимальным.

По типу интерфейса (способа взаимодействия с пользователем) операционные системы делятся на 2 класса: ОС с интерфейсом командной строки и ОС с графическим интерфейсом.

Первые операционные системы (CP/M, MS-DOS, Unix) вели диалог с пользователем на экране текстового дисплея. Это был в полном смысле слова диалог, в ходе которого человек и компьютер по очереди обменивались сообщениями: человек вводил очередную команду, а компьютер, проверив ее, либо выполнял, либо отвергал по причине ошибки. Такие системы принято называть ОС с интерфейсом командной строки. Очевидно, что подобный способ общения не очень удобен для человека, поскольку требует постоянно держать в голове жесткий синтаксис всех допустимых команд и очень внимательно их вводить. Поэтому почти сразу же стали появляться сервисные системные программы, тем или иным способом облегчающие работу с ОС. Наиболее ярким примером таких программ-оболочек может служить широко известный Norton Commander, который был настолько распространен, что многие пользователи искренне считали его частью операционной системы.

Развитие графических возможностей дисплеев привело к коренному изменению принципов взаимодействия человека и компьютера. Командная строка была безвозвратно вытеснена графическим интерфейсом, когда объекты манипуляций в ОС изображаются в виде

небольших рисунков, а необходимые действия тем или иным образом выбираются из предлагаемого машиной списка – так называемого меню. При подобном методе диалога набор текста полностью отсутствует и вполне достаточно всего нескольких клавиш.

Существенным дополнением к графическому способу ведения диалога явилось появление нового устройства ввода информации в компьютер – манипулятора «мышь», без которого сейчас просто невозможно представить современный компьютер. Примерами операционной системы с графическим интерфейсом служат довольно похожие ОС для компьютеров Macintosh (MAC ОС) и IBM PC – OS/2 и Windows. Последняя система в нашей стране распространена необычайно широко.

В различных моделях компьютеров используют операционные системы с неодинаковой архитектурой и возможностями. Для их работы требуются разные ресурсы, они предоставляют разную степень сервиса для программирования и работы с готовыми программами.

1.1. Операционные системы корпорации Microsoft

1.1.1. Семейство DOS

Двадцатилетняя история операционных систем интересна и поучительна, исполнена драматических событий и героизма, подвигов и предательства. А началась она именно с DOS (аббревиатура словосочетания Disk Operating System), точнее – с первой версией этой ОС, выпущенной корпорацией Microsoft в 1981 г. и предназначенной для поставки с компьютерами IBM PC (хотя сначала IBM отдала предпочтение другой ОС под названием CP/M). Кстати, немногие сегодня помнят, что MS-DOS отнюдь не была оригинальной разработкой самой Microsoft: компания Билла Гейтса лишь доработала «операционку» под названием QDOS, созданную компанией Seattle Computer Products.

Новая 16-разрядная однозадачная операционная система DOS обладала «интерфейсом командной строки», т. е. все команды пользователю приходилось набирать на клавиатуре вручную в командной строке ОС. Никакой графики. Никакого сервиса...

Однако DOS процветала на протяжении 10 лет. У Microsoft даже появились конкуренты в виде фирм Novell, Digital Research и IBM. Каждая из этих компаний выпустила свою версию DOS, которые во многом превосходили продукт Microsoft. В частности, Novell DOS

пользовалась заслуженной популярностью как превосходная сетевая ОС, продукт IBM обладал лучшими сервисными возможностями.

Конечно, со временем DOS совершенствовалась и пополнялась новыми программами. С каждой новой версией она поддерживала все больше типов устройств. Однако главные ее недостатки не были, да и не могли быть устранены.

Основным уязвимым местом DOS оставалась работа с оперативной памятью. Дело в том, что в эпоху создания MS-DOS оперативная память большинства компьютеров не превышала 256 килобайт. DOS могла работать с 640 килобайтами оперативной памяти, и Билл Гейтс утверждал, что никому и никогда не понадобится больший объем.

Но время шло, память на компьютерах потихоньку росла – 1 мегабайт, 2 мегабайта; появились программы, которым требовался для работы весь объем оперативной памяти. Стандартный же сервис DOS этой возможности не предоставлял, поэтому приходилось использовать специальные программы – менеджеры памяти. Но и они не могли заставить «упрямую» DOS размещать загружаемые при включении компьютера программы вне «области 640 килобайт». Возникал парадокс: сколько бы оперативной памяти ни имел ваш компьютер, вы не могли запустить программу, если у вас не имелось достаточно свободного пространства в стандартной памяти – той самой области 640 килобайт...

Вторым недостатком DOS была невозможность работы в полноценном графическом режиме, хотя «железо» тогдашних компьютеров уже могло бы обеспечить его поддержку. Дело в том, что DOS практически не позволяла работать с загружаемыми драйверами для различных видеокарт.

Между тем в конце 1980-х гг. графический режим стал уже стандартным для таких компьютеров, как Apple Macintosh, благодаря чему они превратились в стандарт «издательского» компьютера. MS-DOS могла похвастаться только такой «оболочкой», как знаменитый файловый менеджер Norton Commander.

Наконец, третьим препятствием на пути MS-DOS стала однозадачность. Все больше и больше людей желало запускать на своем компьютере сразу несколько программ с возможностями переключения между ними – а DOS при всем желании этого обеспечить не могла, в отличие от ОС тех же компьютеров Macintosh. В результате с появлением Windows 95 DOS практически сошла со сцены, хотя до сих пор установлена на наших компьютерах в качестве составляющей ядра Windows.

Впрочем, в локальных сетях многих крупных фирм США и Европы до сих пор «трудятся» старенькие компьютеры с процессорами

386 и 486 – и не секрет, что крупные организации очень неохотно обновляют парк своих сетевых ПК, да и многие DOS-программы, установленные на этих компьютерах, до сих пор вполне справляются со своими обязанностями.

Именно эти соображения понудили корпорацию IBM продолжить казалось бы угасшую навек линию DOS. В конце 1998 г. в продаже появился последний представитель этой линии – DOS 2000. Его отличия от предшественников в основном заключаются в корректной работе с 2000-м годом, а также в усовершенствованной системе оптимизации памяти и сжатия дисков.

По понятным причинам на домашних ПК PC DOS 2000 практически не встречается. Да и на компьютерах крупных и мелких фирм уже давно «воцарились» Windows 98/ME/XP. Однако в ряде консервативных стран Западной Европы (как ни странно, их список возглавляет богатая Германия) PC DOS 2000 стал весьма популярен, вытеснив с жестких дисков своего «коллегу» от Microsoft.

1.1.2. Семейство Windows 3.X/9X

Windows 3.1/3.11. Первая версия Windows вышла в свет в конце 1980-х гг. и осталась совершенно незамеченной. Популярность она завоевала далеко не сразу – в 1990 г., когда вышла версия Windows 3.0. Популярность новой версии Windows объяснялась несколькими причинами.

Система Windows 3.1 изначально создавалась так, чтобы полностью взять на себя общение с конкретным типом дисплея или принтера. Как пользователю, так и программисту, создающему приложение под Windows, предоставлены универсальные средства, снимающие проблему обеспечения совместимости с конкретной аппаратурой (аппаратная совместимость) и программным обеспечением (программная совместимость).

Унифицированный единый графический интерфейс с пользователем облегчает изучение новых программных продуктов, позволяет работать с объектами вашего компьютера не с помощью команд, а с помощью наглядных и понятных действий над значками, обозначающими эти объекты.

Одним из средств, обеспечивающих программную совместимость, является механизм обмена данными между различными приложениями. Специальный «почтовый ящик» (clipboard) Windows 3.1 позволяет пользователю переносить информацию из одного приложения

в другое, не заботясь о ее формате и представлении. В отличие от профессиональных операционных систем, где механизм обмена данными между программами доступен только программисту, в Windows 3.1 это делается очень просто и наглядно для пользователя.

Механизм обмена данными между приложениями – жизненно важное свойство многозадачной среды, и в настоящее время производители программного обеспечения пришли уже к выводу, что для переноса данных из одного приложения в другое одного «почтового ящика» явно недостаточно. Появился новый, более универсальный механизм – OLE (Object Linking Embedded – встроенная объектная связь), который позволяет переносить из одного приложения в другое разнородные данные.

Windows не только позволяет работать с привычным программным продуктом, но и предлагает дополнительные возможности (запуск нескольких программ одновременно, быстрое переключение с одной программы на другую, обмен данными между ними и т. п.). Обеспечена возможность работы со всеми прикладными программами MS-DOS (текстовыми процессорами, СУБД, электронными таблицами и пр.). Возможность одновременной работы с несколькими программами значительно повысила удобство и эффективность работы.

Windows 3.1 может работать в одном из трех режимов: Real (реальном), Standart (стандартном), 386 Enhanced (расширенном). В процессе установки Windows анализирует имеющиеся аппаратные ресурсы и автоматически устанавливает режим, наиболее полно использующий возможности имеющейся аппаратуры.

В реальном режиме Windows 3.1 не использует аппаратные возможности, не поддерживаемые MS-DOS (этот режим является единственно возможным для машин с процессором 8086/8088): как и в MS-DOS, пользователь ограничен оперативной памятью в 640 килобайт.

В стандартном режиме (возможном на компьютерах с процессором 80286 или 80386) Windows 3.1 полностью использует имеющуюся на компьютере расширенную память, загружая туда все приложения, написанные специально для Windows. Программы DOS загружаются в обычную память.

В расширенном режиме (возможном на компьютерах с процессором 80386 и выше) при запуске приложений (и Windows, и обычных программ для MS-DOS) Windows 3.1 поддерживает так называемый режим виртуальной машины (программе как бы выделяется свой собственный компьютер со всеми ресурсами), реализуя многозадачную среду.

Windows 3.1 позволяет запускать одновременно несколько программ (в том числе одну и ту же программу несколько раз) с возможностью мгновенного переключения с одной программы на другую. Это позволяет инициировать длительный процесс (печать, сортировку и копирование больших объемов данных) и заняться другой работой, а не ждать, пока он закончится.

Удобство и легкость написания программ для Windows привели к появлению все большего количества разнообразных программ, работающих под управлением этой ОС. Последующие версии Windows были направлены на повышение надежности, а также поддержку средств мультимедиа (версия 3.1) и работу в компьютерных сетях (версия 3.11).

Windows 95 представляет собой универсальную высокопроизводительную многозадачную и многопоточную 32-разрядную ОС нового поколения с графическим интерфейсом и расширенными сетевыми возможностями. Windows 95 – интегрированная среда, обеспечивающая эффективный обмен информацией между отдельными программами и предоставляющая пользователю широкие возможности работы с мультимедиа, обработки текстовой, графической, звуковой и видеоинформации. Интегрированность подразумевает также совместное использование ресурсов компьютера всеми программами.

Эта операционная система обеспечивает работу пользователя в сети, предоставляя встроенные средства поддержки для обмена файлами и меры по их защите, возможность совместного использования принтеров, факсов и других общих ресурсов. Windows 95 позволяет отправлять сообщения электронной почтой, факсимильной связью, поддерживает удаленный доступ. Применяемый в Windows 95 защищенный режим не дает прикладной программе в случае сбоя нарушить работоспособность системы, надежно предохраняет приложения от случайного вмешательства одного процесса в другой, обеспечивает определенную устойчивость к вирусам.

Пользовательский интерфейс Windows 95 прост и удобен. В отличие от оболочки Windows 3 эта операционная система не нуждается в установке на компьютере операционной системы DOS. Она предназначена для установки на настольных ПК и компьютерах блокнотного типа с процессором 486 или Pentium. Рекомендуемый размер оперативной памяти – 32–128 мегабайт. После включения компьютера и выполнения тестовых программ BIOS операционная система Windows 95 автоматически загружается с жесткого диска. После загрузки и инициализации системы на экране появляется рабочий стол,

на котором размещены различные графические объекты. Пользовательский интерфейс спроектирован так, чтобы максимально облегчить усвоение этой операционной системы новичками и создать комфортные условия для пользователя.

1.1.3. Windows 98/98 SE

Windows 98 появилась в 1998 г. и отличается от Windows 95 тем, что в ней операционная система объединена с браузером Internet Explorer посредством интерфейса, выполненного в виде Web-браузера. При сохранившемся интерфейсе внутренняя структура была значительно переработана. Много внимания было уделено работе с Интернетом, а также поддержке современных протоколов передачи информации – стандартов, обеспечивающих обмен информацией между различными устройствами. Кроме того, особенностью Windows 98 является возможность работы с несколькими мониторами. В ней также улучшена совместимость с новыми аппаратными средствами компьютера, она одинаково удобна для использования как на настольных, так и на портативных компьютерах.

В конце 1999 г. в продаже появилась русскоязычная версия нового комплекта *Windows 98 SE*. От предыдущей версии новая Windows отличается тем, что в ее состав включены последняя (пятая) версия браузера Internet Explorer, обновленная система соединения с Internet, а также многочисленные исправления ошибок и новая библиотека драйверов устройств.

1.1.4. Семейство Windows NT/2000/XP

Windows NT (NT – англ. New Technology) – это операционная система, а не просто графическая оболочка. Она использует все возможности новейших моделей персональных компьютеров и работает без DOS. Windows NT – 32-разрядная ОС со встроенной сетевой поддержкой и развитыми многопользовательскими средствами. Она предоставляет пользователям истинную многозадачность, многопроцессорную поддержку, секретность, защиту данных и многое другое. Эта операционная система очень удобна для пользователей, работающих в рамках локальной сети, для коллективных пользователей, особенно для групп, работающих над большими проектами и обменивающимися данными. Следует учитывать, что большая часть достоинств NT проявляется лишь в сетевом режиме работы, т. е. в связке с другими компьютерами.

Windows NT по существу представляет собой операционную систему сервера, приспособленную для использования на рабочей станции. Этим обусловлена архитектура, в которой абсолютная защита прикладных программ и данных берет верх над соображениями скорости и совместимости. Чрезвычайная надежность Windows NT обеспечивается ценой высоких системных затрат, поэтому для получения приемлемой производительности необходимы быстродействующий процессор и по меньшей мере 16 мегабайт ОЗУ. Как и в OS/2 Warp, в системе Windows NT безопасность нижней памяти достигается за счет отказа от совместимости с драйверами устройств реального режима. В среде Windows NT работают собственные 32-разрядные NT-прикладные программы, а также большинство прикладных программ Windows 95. Так же как OS/2 Warp и Windows 95, система Windows NT позволяет выполнять в своей среде 16-разрядные Windows- и DOS-программы.

Windows 2000 – операционная система нового поколения для делового использования на самых разнообразных компьютерах – от портативных до серверов. Система Windows 2000 разработана на основе Windows NT и унаследовала от нее высокую надежность и защищенность информации от постороннего вмешательства. Эта ОС является наилучшей для ведения коммерческой деятельности в Интернете. Она объединяет присущую Windows 98 простоту использования в Интернете, на работе, в пути с присущими Windows NT надежностью, экономичностью и безопасностью. С другой стороны, слабые места той же NT с новой силой проявились в Windows 2000. Высокая требовательность к ресурсам компьютера оттолкнула от новой ОС часть домашних пользователей.

Как и Windows NT, Windows 2000 была выпущена в нескольких вариантах – серверном (Server), для установки на главный, управляющий компьютер сети, и клиентском (Professional) – для рабочих станций. Самая мощная версия – Datacenter, предназначенная для крупных корпораций, была официально представлена в сентябре 2000 г.

Windows XP. В октябре 2001 г. вышла очередная версия ОС Windows – Windows XP, которая существует в нескольких вариантах, в том числе Windows XP Professional Edition (разработана для предприятий и предпринимателей и содержит такие функции, как удаленный доступ к рабочему столу компьютера, шифрование файлов, центральное управление правами доступа и поддержка многопроцессорных систем), Windows XP Home Edition – система для домашнего применения), выпускается как недорогая «урезанная» версия

Professional Editon. Некоторыми из наиболее заметных улучшений в Windows XP по сравнению с Windows 2000 являются:

- новое оформление графического интерфейса, включая более округлые формы и плавные цвета, и дополнительные функциональные улучшения (такие как возможность представления папки в виде слайд-шоу в проводнике Windows);
- возможность быстрого переключения пользователей, позволяющая временно прервать работу одного пользователя и выполнить вход в систему под именем другого пользователя, оставляя при этом приложения, запущенные первым пользователем, включенными;
- функция «удаленный помощник», позволяющая опытным пользователям и техническому персоналу подключаться к компьютеру с системой Windows XP по сети для разрешения проблем. При этом помогающий пользователь может видеть содержимое экрана, вести беседу и (с позволения удаленного пользователя) брать управление в свои руки; программа восстановления системы, предназначенная для возвращения системы в определенное предшествующее состояние;
- улучшенная совместимость со старыми программами и играми. Специальный мастер совместимости позволяет эмулировать для отдельной программы поведение одной из предыдущих версий ОС (начиная с Windows 95);
- возможность удаленного доступа к рабочей станции благодаря включению в систему миниатюрного сервера терминалов (только в издании Professional);
- более развитые функции управления системой из командной строки; поддержка проводником Windows цифровых фотоформатов (например, представление папки в виде слайд-шоу) и аудиофайлов (автоматическое отображение метаданных для аудиофайлов, например, тегов ID3 для MP3-файлов).

1.2. Альтернативные операционные системы

1.2.1. Операционная система Unix

Операционная система Unix была создана в Bell Telephone Laboratories. Unix – многозадачная операционная система, способная обеспечить одновременную работу очень большого количества пользователей. Ядро ОС Unix написано на языке высокого уровня C и

имеет только около 10 процентов кода на ассемблере. Это позволяет за считанные месяцы переносить ОС Unix на другие аппаратные платформы и достаточно легко вносить в нее серьезные изменения и дополнения. Unix является первой действительно переносимой операционной системой. В многочисленные существующие версии Unix постоянно вносятся изменения. С одной стороны, это расширяет возможности системы, делает ее мощнее и надежнее, с другой – ведет к появлению различий между существующими версиями. В связи с этим возникает необходимость стандартизации различных свойств системы. Наличие стандартов облегчает переносимость приложений между различными версиями Unix и защищает как пользователей, так и производителей программного обеспечения. Поэтому в 1980-х гг. разработан ряд стандартов, оказывающих влияние на развитие Unix. Сейчас существуют десятки операционных систем, которые можно объединить под общим названием Unix. В основном это коммерческие версии, выпущенные производителями аппаратных платформ для компьютеров своего производства. Причины популярности Unix:

- код системы написан на языке высокого уровня C, что сделало ее простой для понимания, изменения и переноса на другие платформы. Можно смело сказать, что Unix является одной из наиболее открытых систем;

- Unix – многозадачная многопользовательская система. Один мощный сервер может обслуживать запросы большого количества пользователей, при этом необходимо администрирование только одной системы. Кроме того, система способна выполнять большое количество различных функций, в частности работать как вычислительный сервер, сервер базы данных, сетевой сервер, поддерживающий важнейшие сервисы сети, и т. д.;

- наличие стандартов. Несмотря на разнообразие версий Unix, основой всего семейства являются принципиально одинаковая архитектура и ряд стандартных интерфейсов. Для администратора переход на другую версию системы не составит большого труда, а для пользователей он может и вовсе оказаться незаметным;

- простой, но мощный модульный пользовательский интерфейс. Имея в своем распоряжении набор утилит, каждая из которых решает узкую специализированную задачу, можно конструировать из них сложные комплексы;

- использование единой, легко обслуживаемой иерархической файловой системы. Файловая система Unix – это не только доступ к данным, хранящимся на диске. Через унифицированный интерфейс

файловой системы осуществляется доступ к терминалам, принтерам, сети и т. п.;

- большое количество приложений, в том числе свободно распространяемых, начиная от простейших текстовых редакторов и заканчивая мощными системами управления базами данных.

1.2.2. Операционная система Linux

Начало созданию системы Linux положено в 1991 г. финским студентом Линусом Торвальдсом (Linus Torvalds). В сентябре 1991 г. он распространил по e-mail первый прототип своей операционной системы и призвал откликнуться на его работу всех, кому она нравится или нет. С этого момента многие программисты стали поддерживать Linux, добавляя драйверы устройств, разрабатывая разные продвинутое приложения и др. Атмосфера работы энтузиастов над полезным проектом, а также свободное распространение и использование исходных текстов стали основой феномена Linux. В настоящее время Linux – очень мощная система, но самое замечательное то, что она бесплатная (free).

Линус Торвальдс разработал не саму операционную систему, а только ее ядро, подключив уже имеющиеся компоненты. Сторонние компании, увидев хорошие перспективы для развития своего бизнеса, довольно скоро стали насыщать ОС утилитами и прикладным программным обеспечением. Недостаток такого подхода – отсутствие унифицированной и продуманной процедуры установки системы, и это до сих пор является одним из главных сдерживающих факторов для более широкого распространения Linux.

Феномен Linux вызвал к жизни разговоры о том, что родилась новая философия программирования, принципиально отличающаяся от того, что было раньше. Традиционные стадии жизненного цикла программного продукта таковы: анализ требований, разработка спецификаций, проектирование, макетирование, написание исходного текста, отладка, документирование, тестирование и сопровождение. Главное, что отличает этот подход, – централизация управления разными стадиями и преимущественно «нисходящая» разработка (т. е. постоянная детализация). Однако Linux создавалась по-иному. Готовый работающий макет постоянно совершенствовался и развивался децентрализованной группой энтузиастов, действия которых лишь слегка координировались. Налицо анархичный характер и «восходящая» разработка: сборка все более крупных блоков из ранее созданных мелких. Здесь

можно отметить и другое. При традиционной разработке в основу кладется проектирование и написание текстов, при разработке по методу Linux – макетирование, отладка и тестирование. Иными словами, разработка по методу Linux – это метод проб и ошибок, построенный на интенсивном тестировании. На любом этапе система должна работать, даже если это мини-версия того, к чему стремится разработчик. Естественный отбор оставляет только жизнеспособное. О том, что такое программирование – наука, искусство или ремесло, спорят уже давно. И если в основе традиционной разработки ПО лежит прежде всего ремесло, то при разработке методом компьютерного дарвинизма – несомненно искусство.

Нетрудно заметить, что «восходящая» разработка характеризует так называемое исследовательское программирование, когда система строится вокруг ключевых компонентов и программ, которые создаются на ранних стадиях проекта, а затем постоянно модифицируются. Отсутствие четкого плана, минимальное управление проектом, большое число сторонних территориально удаленных разработчиков, свободный обмен идеями и кодами – все это атрибуты нового программирования. Об особенностях исследовательского программирования написано немало статей. Так, швейцарские профессора А. Киральф, К. Чен и Й. Нивергельт выделили следующие важные моменты:

- разработчик ясно представляет ++направление поиска, но не знает заранее, как далеко он сможет продвинуться к цели;
- нет возможности предвидеть объем ресурсов для достижения того или иного результата;
- разработка не поддается детальному планированию, она ведется методом проб и ошибок;
- такие работы связаны с конкретными исполнителями и отражают их личностные качества.

2. ОФИСНЫЕ ТЕХНОЛОГИИ

2.1. Общие сведения о Microsoft Office

Сегодня Microsoft Office (в дальнейшем – Office) рассматривают как набор инструментов, необходимых для организации работы в офисе, и применяют его не только в качестве комплекта настольных приложений, но и как платформу для создания специализированных решений или средство доступа и обмена данными. Office обеспечивает легкость использования, многофункциональность и интеграцию приложений для ведения делопроизводства предприятий малого и среднего бизнеса.

Как правило, в решении задач, работе над проектами и принятии деловых решений участвуют группы исполнителей. Но для совместной подготовки документа или для управления проектом такой команде необходим надежный фундамент, поддерживающий процесс сотрудничества.

В прошлом попытки совместного использования файлов наталкивались на ряд типичных проблем, в том числе на различие форматов файлов, несогласованность в организации обратной связи с получателем, трудности работы при использовании разных программных средств и отсутствие контроля версий документов с последними изменениями. Поскольку интенсивность совместной работы над проектами продолжает возрастать, то все большее значение приобретают приложения, обеспечивающие комплексность решений и поддержку такого сотрудничества.

Стандартный выпуск Microsoft Office включает следующие приложения:

- **Microsoft Word** – многофункциональный текстовый процессор для операционной системы Microsoft Windows, позволяет создавать документы любой сложности и оформлять их с использованием различных шрифтов, обладает удобным графическим интерфейсом и средствами автоматизации оформления документов. Создаваемые файлы имеют расширение doc.

- **Microsoft Excel** – программа представления и обработки данных в виде электронных таблиц, имеет мощный аппарат математических инструментов для решения задач линейного программирования, оптимизации, статистического моделирования. Создаваемые файлы имеют расширение xls.

- **Microsoft PowerPoint** – программа создания презентаций и слайд-фильмов, предоставляет пользователю возможности оформления текста, рисования, построения диаграмм, а также набор стандартных

иллюстраций, стилевых шаблонов и возможности использования звука и видео. Создаваемые файлы имеют расширение ppt.

- **Microsoft Outlook** – программа управления информацией, которая помогает работать с сообщениями, приходящими по электронной почте, контактными лицами, назначать встречи, ставить задачи, отслеживать деятельность свою и сотрудников, просматривать совместные документы.

Профессиональный выпуск Microsoft Office помимо компонентов стандартной поставки содержит новые программы:

- **Microsoft Access** – программу для создания и редактирования баз данных;

- **Microsoft Publisher** – программу верстки и дизайна текстовых публикаций.

Выпуск Microsoft Office *для разработчиков* (Developer) помимо компонентов профессионального выпуска включает **Microsoft FrontPage** – программу для создания и дизайна страниц Интернет.

Существует еще ряд превосходных офисных программ от Microsoft, которые не входят ни в один комплект поставки Microsoft Office, а как бы примыкают к нему:

- **Microsoft Visio** – пакет деловой графики;
- **Picture-It** – графический пакет;
- **Microsoft Money** – «домашний бухгалтер».

Все программы пакета Microsoft Office содержат встроенные средства программирования, основанные на простом языке VBA (Visual Basic for Applications).

Все офисные приложения устроены одинаково. Общие команды открытия, закрытия и создания файлов, общие принципы работы со справкой, использование шаблонов при создании документов, печать и сохранение документов, настройка панелей инструментов и команд меню – все эти функции не зависят от того, обрабатывает программа текстовый документ, электронную таблицу, презентацию или рисунок.

Рассмотрим общие операции на примере программы Word, как наиболее часто используемой в делопроизводстве.

Создавать документы Office можно разными способами. Самый простой – создание нового документа в открытом приложении.

Создание документа:

1. Откройте нужное приложение, например Word.
2. Щелкните на кнопке **Создать** стандартной панели инструментов или выберите из меню **Файл** ➤ **Создать**.

Будет создан пустой документ, в который можно ввести нужный текст или данные таблицы.

Сохранение документа. Если документ сохраняется впервые или надо вновь сохранить ранее созданный документ под другим именем, необходимо выполнить команду **Сохранить как**:

1. Выберите команду меню **Файл** ➤ **Сохранить как**.
2. Выберите папку, в которой нужно сохранить документ.
3. Введите имя документа в поле **Имя файла**.

С помощью команды **Сохранить** из меню **Файл** можно сохранить текущий документ с тем же именем.

Поле **Имя файла** служит для ввода названия документа. Поле **Тип файла** позволяет осуществить выбор формата сохраняемого файла (по умолчанию предлагается формат открытого приложения).

В левой части окна **Сохранение документа** предлагаются папки для сохранения документа.

Открытие документа:

1. В меню **Файл** выберите команду **Открыть** (открыть файл можно нажатием кнопки **Открыть** или двойным щелчком левой клавиши мыши на его имени).

2. В качестве требуемого типа файла установите, например, **Документы Word**.

3. Укажите нужную папку в окне списка **Папка**.

4. Выберите файл, с которым хотите работать.

Для каждого приложения в нижней части меню **Файл** сохраняется список файлов, которые использовались последними. Любой из этих файлов можно открыть, выбрав его из списка.

Можно открыть любой документ, не открывая приложения, из папки **Мой компьютер** или в окне **Проводник Windows**. Для этого нужно дважды щелкнуть мышью на его имени.

Кроме того, последние документы, которые были открыты на компьютере, можно открыть, выбрав нужный документ в списке **Документы** меню **Пуск**.

2.2. Общие сведения о Microsoft Word

Word – текстовый процессор для операционной системы Microsoft Windows, многофункциональная программа обработки текстов. Это приложение Windows, предназначенное для создания, просмотра, изменения и печати текстовых документов. С помощью Word вы можете не просто набрать текст, но и оформить его по своему вкусу: включить в него таблицы и графики, картинки и даже

звуки и видеоизображения. Word поможет вам составить простое письмо и сложный объемный документ, яркую поздравительную открытку или рекламный блок. По своим функциям Word вплотную приближается к издательским системам и программам верстки. Это значит, что в данном редакторе можно полностью подготовить к печати (или, как говорят специалисты, сверстать) журнал, газету или даже книгу, изготовить WWW-страницу Интернет. «Текстовым процессором номер один» для миллионов пользователей Word делают его возможности, которые он предоставляет для работы с документом:

- возможность создания нового документа с помощью специальных шаблонов (в частности, в Word включены шаблоны стандартных писем, поздравительных записок, отчетов, факсов и ряда других офисных документов);
- возможность одновременного открытия и работы с большим количеством документов;
- автоматическая проверка орфографии, грамматики и даже стилистики при вводе документа;
- автоматическая коррекция наиболее часто повторяющихся ошибок;
- расширенные возможности форматирования документа. В отличие от WordPad, Word допускает выравнивание документа по обоим краям, многоколоночную верстку;
- использование стилей для быстрого форматирования документа;
- возможность автоматизации ввода повторяющихся и стандартных элементов текста;
- удобные механизмы работы с ссылками, сносками, колонтитулами;
- включение в текст элементов, созданных в других программах Microsoft Office, – графических изображений, электронных таблиц и графиков, звуков, видеоизображений и т. д.;
- возможность подготовки простых электронных таблиц и гипертекстовых документов Интернет;
- возможность работы с математическими формулами;
- возможность автоматического создания указателей и оглавления документа;
- возможность отправки готового документа непосредственно из Microsoft Word на факс и по электронной почте (в обоих случаях необходимо, чтобы компьютер пользователя был оснащен модемом) и многое другое.

2.2.1. Окно программы Word

Структура окна программы Word типична для приложений Windows. В нее входят элементы, необходимые для редактирования и форматирования текстов. В верхней части окна располагаются панели команд, к которым относятся строка меню и панели инструментов. После установки программы в ней по умолчанию присутствуют две панели инструментов – **Стандартная** и **Форматирование**. Однако если в ходе работы возникает необходимость в других панелях инструментов, их тоже можно открыть и расположить вдоль любой границы окна или отдельно. Для этого следует выполнить команду **Вид** ➤ **Панели инструментов** и выбрать нужную панель или воспользоваться контекстным меню любой видимой панели инструментов.

Существует возможность изменения панелей инструментов, вставки новых кнопок или замены существующих, а также создания собственных панелей инструментов. Настройка панелей инструментов осуществляется с помощью команды **Сервис** ➤ **Настройка** или команды **Вид** ➤ **Панели инструментов** ➤ **Настройка**.

Координатные линейки. Под панелями инструментов располагается горизонтальная линейка, проградуированная в сантиметрах или дюймах. Она помогает контролировать размещение элементов страницы и управлять операциями форматирования. Изменить единицу измерения линейки можно с помощью команды **Сервис** ➤ **Параметры** ➤ **Общие** ➤ **Единицы измерения**.

Маркеры левого и правого отступов абзаца (нижние треугольники на горизонтальной линейке) позволяют установить необходимый отступ выделенного абзаца от левого и правого полей страницы. Поля страницы обозначены на линейке темно-серым цветом.

Маркер красной строки (верхний треугольник на горизонтальной линейке) позволяет установить отступ красной строки выделенного абзаца. Чтобы изменить стандартный отступ, следует переместить маркер в нужную позицию с помощью мыши.

Скрыть и отобразить координатные линейки можно с помощью команды **Вид** ➤ **Линейка**. Для отображения вертикальной координатной линейки курсор ввода должен находиться на той странице, которая отображается в окне.

Полосы прокрутки. Вертикальная и горизонтальная полосы прокрутки расположены в правой и нижней частях окна программы. Они предназначены для перемещения документа.

Отображение полос прокрутки на экране задается с помощью команды **Сервис** > **Параметры** > **Вид** > **Окно**.

На каждой полосе прокрутки (рис. 2.1) расположен бегунок (2 и 9), позволяющий определить положение документа относительно окна. Он служит также для быстрого перемещения по документу.

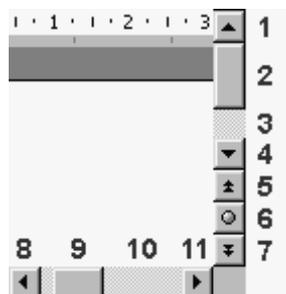


Рис. 2.1. Полосы прокрутки

Вертикальная полоса прокрутки программы Word имеет особенность. Под ней расположены три дополнительные кнопки перехода: на страницу вверх (5) и вниз (7) или к выбранному объекту (6), в качестве которого могут выступать страница, раздел, таблица, сноска, заголовок, рисунок и т. п. При щелчке на кнопке выбора объекта перехода открывается меню с возможными вариантами перехода к нужному месту документа.

В самой нижней части окна располагается **строка состояния**. Она содержит справочную информацию о документе и индикаторы, указывающие на текущий режим работы. Скрыть и показать строку состояния можно с помощью команды **Сервис** > **Настройка** > **Вид**.

Контекстное меню. Некоторые объекты имеют контекстное меню. В состав контекстного меню входят команды, предназначенные для обработки данного объекта.

Для открытия контекстного меню нужно установить указатель мыши на объекте и щелкнуть правой кнопкой.

Режимы просмотра документа. Один и тот же документ может иметь различный вид в зависимости от установленного режима просмотра документа на экране. Таких режимов несколько. Слева от горизонтальной полосы прокрутки располагаются четыре кнопки, позволяющие выбрать вид отображения документа в рабочей области:

1. Обычный режим – используют при простом вводе и редактировании текста. В этом режиме не отображаются специальные элементы страницы, рисунки и столбцы текста. Он предназначен только для работы с текстом.

2. Режим электронного документа – наиболее удобен, если речь идет не о редактировании, а о просмотре готового документа. Слева открывается дополнительная панель с содержанием документа. Она дает наглядное представление о структуре документа и обеспечивает удобный переход к любому разделу. В этом режиме на экране не отображаются кнопки выбора метода представления документа, и чтобы из него выйти, надо воспользоваться пунктом меню **Вид**, в котором имеются нужные элементы управления.

3. Режим разметки страницы – документ представляется на экране точно так, как он будет выглядеть при печати на бумаге. Этот режим наиболее удобен для операций форматирования. Он необходим в случае вставки в документ кадра или графики.

4. Режим структуры – удобен для работы над планом документа (составление, просмотр, редактирование). Он применяется в том случае, когда еще не выбран окончательный вариант документа и необходимо переставлять разделы.

Нужный режим просмотра документа можно включить также с помощью соответствующей команды пункта меню **Вид**.

2.2.2. Создание и редактирование текстовых документов

Параметры страницы документа. По стандарту принято использовать бумагу формата А4. Для данного формата приняты следующие поля: левое – 30 мм, верхнее – 15 мм, правое – 10 мм, нижнее – 20 мм. Этот формат также необходимо установить по умолчанию для используемых принтеров. Параметры страницы документа задаются командой **Файл** ➤ **Параметры страницы** ➤ **Поля** (рис. 2.2).

Установить поля можно также с помощью координатных линеек. Для этого необходимо поставить курсор на координатной линейке на границу между темно-серой и белой областью; когда курсор станет двунаправленной стрелкой, нажать левую клавишу мыши; не отпуская клавишу, протянуть мышь в нужном направлении (в сторону увеличения или уменьшения размера поля).

Набор текста. Окно текущего документа всегда содержит мигающую вертикальную черту – курсор. Ввод текста осуществляется путем набора с клавиатуры. Вводимые символы появляются в месте расположения курсора. Курсор при вводе сдвигается вправо. Чтобы вводимый текст замещал, а не сдвигал текст, имевшийся ранее, включают режим замены. Переключение режима замены осуществляют

нажатием клавиши <Insert> или двойным щелчком на индикаторе <Зам> в строке состояния.

По достижении правого края страницы текст автоматически переносится на новую строку. Клавишу <Enter> нужно нажать только тогда, когда начинается набор нового абзаца. Также необходимо отказаться от использования пробелов для выравнивания текста на странице, например для размещения заголовка по центру или для установки красной строки абзаца.

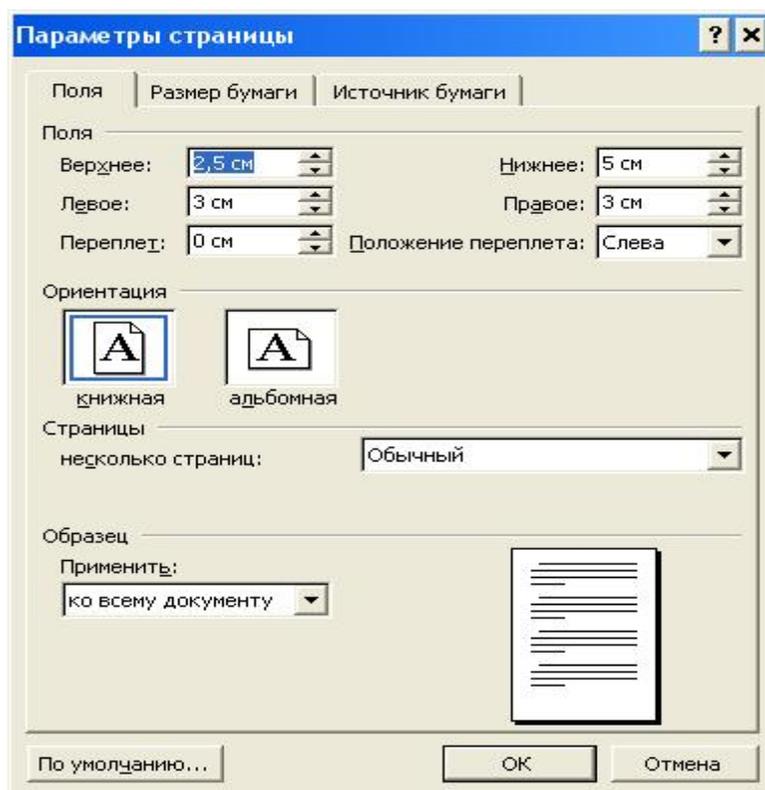


Рис. 2.2. Окно **Параметры страницы**

Отображение специальных символов. Программа Word предусматривает возможность отображения непечатаемых символов. Это символы конца абзаца, табуляции, пробела, мягкого переноса. Отображение этих спецсимволов удобно включать, чтобы видеть все приемы, с помощью которых оформлен текст. Можно быстро включить/выключить их отображение с помощью кнопки **Непечатаемые знаки**  на панели инструментов **Стандартная** или с помощью команды **Сервис** > **Параметры**, перейдя затем на вкладку **Вид**.

Работа с фрагментами текста. Для удаления, копирования и перемещения фрагментов текста соответствующий фрагмент должен

быть сначала выделен. Выделение фрагмента производится протягиванием мыши с нажатой левой клавишей по нужному тексту или любой командой перемещения курсора при нажатой клавише <Shift>. Двойной щелчок мышью выделяет слово, в котором расположен курсор мыши, тройной – абзац.

Если установить курсор мыши в левом поле страницы, то для выделения строки нужно выполнить одинарный щелчок левой клавишей мыши напротив строки, для выделения абзаца – двойной щелчок.

Выделенный фрагмент удаляют нажатием клавиши или просто путем набора замещающего текста. Перемещение или копирование фрагмента можно осуществить методом перетаскивания (копирование – с нажатой клавишей <Ctrl>), хотя намного удобнее использовать буфер обмена.

Определение вида и начертания шрифта. Один из наиболее простых и, в то же время, наиболее выразительных способов изменения внешнего вида текста состоит в изменении шрифта, которым он набран.

В программе Word по умолчанию все операции изменения шрифта применяются к выделенному фрагменту текста или, при отсутствии выделения, к слову, на котором располагается курсор. Для простейших операций по изменению вида и начертания шрифта используют панель инструментов **Форматирование**.

В раскрывающемся списке **Шрифт** выбирают гарнитуру, в списке **Размер** определяют размер символов, а кнопками **Полужирный** **Ж**, **Курсив** **К** и **Подчеркнутый** **Ч** изменяют их начертание.

Эффекты, недоступные с панели инструментов **Форматирование**, можно создать в диалоговом окне **Шрифт** (рис. 2.3), которое открывают командой **Формат** > **Шрифт** или пунктом **Шрифт** в контекстном меню.

В нижней части всех вкладок этого диалогового окна приводится пример текста, написанного в соответствии с заданными параметрами шрифта.

Элементы управления вкладки **Шрифт** соответствуют элементам панели инструментов **Форматирование**. Раскрывающийся список **Подчеркивание** предоставляет нестандартные варианты подчеркивания текста (например, двойной чертой или пунктиром). Цвет отображения текста изменяют в раскрывающемся списке **Цвет текста**. Эта операция имеет смысл только для электронных документов и документов, которые будут распечатываться на цветном принтере. Нестандартные эффекты оформления текста создают установкой флажков в области

Видоизменение. Здесь можно оформить верхний и нижний индексы, зачеркнутый текст, текст с тенью, большие и малые прописные символы.

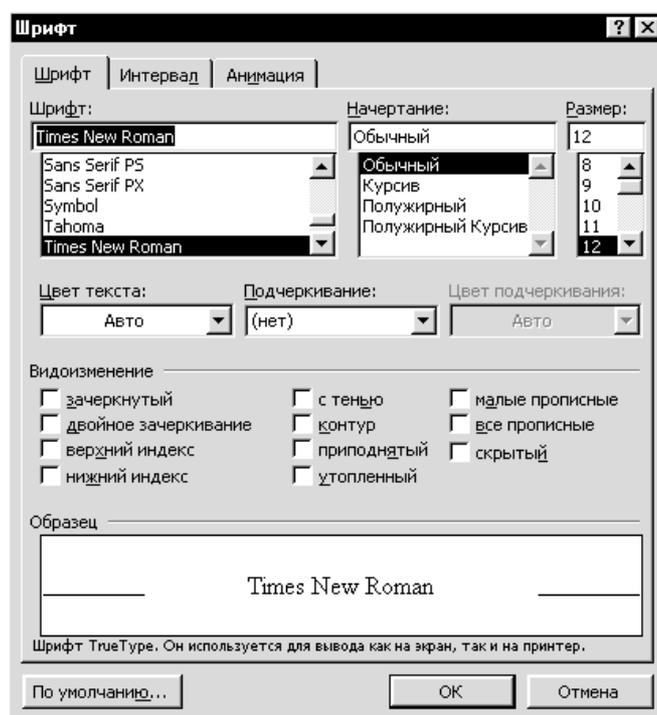


Рис. 2.3. Окно **Шрифт**

Элементы управления вкладки **Интервал** позволяют изменить интервал между символами. Благодаря этому текст может быть уплотнен или разрежен, что нередко используют в заголовках.

Средства вкладки **Анимация** используют для динамического оформления текста, но только в электронных документах.

Выравнивание абзацев. Основной смысловой единицей текста обычно является абзац, поэтому команды выравнивания и операции форматирования предназначены для изменения внешнего вида отдельных абзацев. Выравнивание абзаца – это расположение его текста в соответствии с заданными правилами. Чаще всего речь идет о горизонтальном выравнивании текста, т. е. о его расположении между правым и левым полями страницы.

При выравнивании по левому краю все строки абзаца начинаются с одной и той же позиции – левый край абзаца образует вертикальную линию. При выравнивании по правому краю то же можно сказать о правой границе абзаца. При выравнивании по ширине ровными оказываются и левая, и правая границы. В случае выравнивания по центру строки располагаются симметрично относительно

вертикальной оси, проходящей через середину страницы (такое выравнивание обычно применяют для заголовков).

Книги, журналы и другие печатные издания, документы на русском языке традиционно оформляют с использованием выравнивания по ширине. Такие документы имеют аккуратный вид, хотя в них неминуемо возникает необходимость расстановки переносов.

В программе Word выравнивание задают щелчком на соответствующей кнопке панели инструментов **Форматирование**. Из четырех кнопок (**По левому краю**, **По центру**, **По правому краю** и **По ширине**) может быть включена только одна.

Форматирование абзацев. Для полного форматирования абзаца используют диалоговое окно **Абзац**, которое открывают командой **Формат** > **Абзац** или с помощью пункта **Абзац** в контекстном меню, вызываемом щелчком правой кнопки мыши.

Вкладка **Отступы и интервалы** определяет выравнивание абзаца и его размещение в потоке текста документа. В раскрывающемся списке **Выравнивание** задают способ выравнивания. В области **Отступ** определяют правую и левую границы абзаца относительно правой и левой границ страницы. Раскрывающийся список **Первая строка** позволяет задать наличие и размеры красной строки (абзацного отступа). В области **Интервал** можно задать промежутки между абзацами, а также между строками данного абзаца. Увеличенный интервал между абзацами нередко заменяет абзацный отступ. Вкладка **Положение на странице** предназначена для форматирования абзацев, попадающих на границу между страницами. Здесь можно запретить отрывать от абзаца одну строку, потребовать, чтобы абзац размещался на одной странице целиком, «присоединить» следующий абзац к данному или начать текущим абзацем новую страницу.

Границы и заливка. Создать рамку вокруг текста или страницы, а также заливку документа или его части заданным цветом можно командой **Формат** > **Границы и заливка**. Если граница задается вокруг части текста, а не всей страницы, то его надо предварительно выделить.

Для границ задаются три параметра: **Тип линии**, **Цвет** и **Ширина**. Справа в окне четыре кнопки включают/отключают соответствующую границу (верхнюю, нижнюю, правую, левую). На вкладке **Страница** в списке **Рисунок** выбирается графический (декоративный) стиль рамки для страницы, т. е. она будет состоять из выбранного рисунка. На вкладке **Заливка** задается цвет фона, на котором располагается текст.

Вставка специальных символов. В текст документа можно вставить разные нестандартные символы, которые нельзя непосредственно набрать с клавиатуры, такие как греческие буквы, графические символы. Для этого необходимо выполнить команду **Вставка** ➤ **Символ**. Появится диалоговое окно **Символ** (рис. 2.4), в котором отображены символы выбранного шрифта. На вкладке **Символы** можно найти значки телефона (для визиток), маркеры (для списков, рекламы), символы различных языков (европейских и восточных), на вкладке **Специальные символы** – такие символы, как длинное тире, неразрывный пробел, авторский знак и т. д. Для увеличенного просмотра следует выделить символ, для вставки в текст – нажать кнопку **Вставить**, затем кнопку **Заккрыть**.

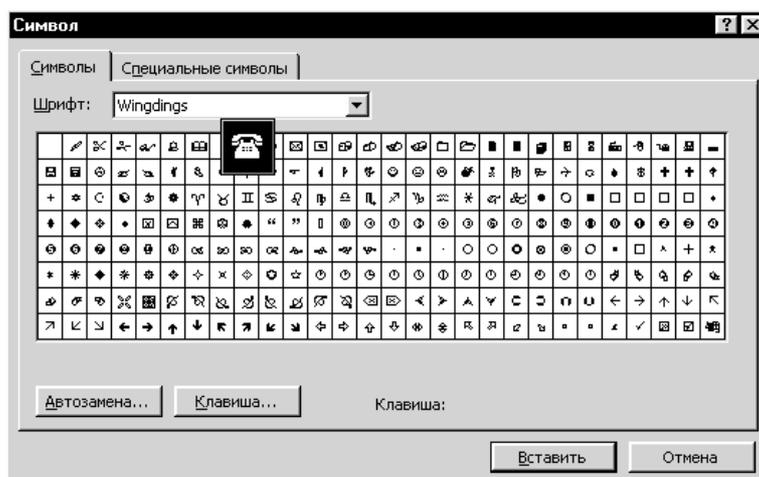


Рис. 2.4. Окно **Символ**

Отмена действия ошибочных команд. Программа Word позволяет отменять действие ошибочных команд и восстанавливать состояние документа, предшествующее неправильным действиям.

Отменить последнюю выполненную операцию можно командой **Правка** ➤ **Отменить**, или кнопкой **Отменить** , или комбинацией клавиш <Ctrl> + <Z>. Можно отменить несколько действий, раскрыв список рядом с кнопкой. Если операция была отменена по ошибке, то ее можно повторить с помощью команды **Правка** ➤ **Повторить**, или кнопкой **Вернуть** , или комбинацией клавиш <Ctrl> + <Y>.

Списки. В виде списков удобно представлять упорядоченную информацию, например перечень предметов или действий. Программа Word поддерживает два вида списков – нумерованные, где пункты последовательно нумеруются, и маркированные, в которых каждый пункт помечается одинаковым маркером.

Для преобразования существующего текста в список надо выделить этот текст и щелкнуть на кнопке **Нумерация**  или **Маркеры**  на панели инструментов **Стандартная**. Программа Word автоматически преобразует новый абзац в элемент нумерованного списка, если он начинается с числа, за которым следует точка. Создание списка заканчивается двукратным нажатием на клавишу **<Enter>** в конце абзаца.

Чтобы изменить или настроить формат списка, необходимо выполнить команду **Формат** ➤ **Список** и в открывшемся диалоговом окне выбрать подходящий вид списка.

Средства поиска и замены. При работе с длинными документами иногда приходится вносить в них повторяющиеся изменения. В программе Word имеются специальные средства поиска и замены, которые позволяют найти в тексте фрагмент, заданный в виде текстовой строки, и заменить его новым текстом. Для этого необходимо командой **Правка** ➤ **Найти** открыть диалоговое окно **Найти и заменить**. На вкладке **Найти** в поле **Найти** надо ввести фрагмент разыскиваемого текста. Задать дополнительные параметры поиска можно, щелкнув на кнопке **Больше**.

Дополнительные кнопки **Формат** и **Специальный** позволяют разыскивать текст, отформатированный указанным образом, и специальные непечатаемые символы. Поиск начинается после щелчка на кнопке **Найти далее**.

Для автоматической замены найденного текста используют элементы управления вкладки **Заменить**. Заменяющую строку вводят в поле **Заменить на**. По щелчку на кнопке **Найти далее** разыскивается очередное место, где заданная строка встречается в документе, затем щелчком на кнопке **Заменить** выполняется замена (если она необходима). Если заранее известно, что замену следует произвести по всему документу и во всех случаях, можно сразу щелкнуть на кнопке **Заменить все**.

Элементы управления вкладки **Перейти** используют для перехода к специфическому тексту или объекту, например к заданной странице, сноске или рисунку.

Автозамена. Программа Word позволяет автоматически исправлять текст в процессе ввода. Например, при вводе **(r)** происходит автозамена этих символов на **®**.

Список слов на автозамену можно пополнять. Для добавления своего элемента автозамены в список надо выполнить команду

Сервис ➤ Параметры автозамены. В поле **заменить:** (рис. 2.5) необходимо ввести заменяемый текст, в поле **на:** – на что он будет заменен.

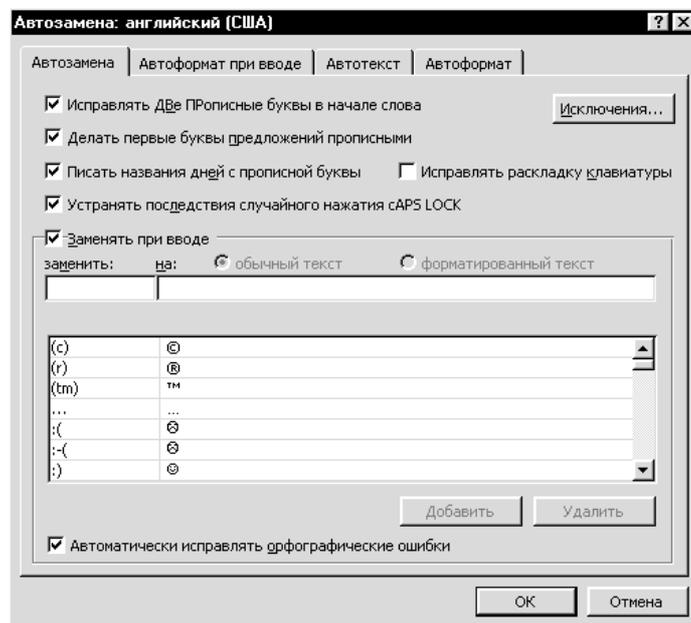


Рис. 2.5. Окно Автозамена

Стилевое оформление. Стиль – это набор параметров форматирования, которые применяются к тексту документа, чтобы быстро изменить его вид. Стили позволяют одним действием применить сразу несколько атрибутов форматирования. Изменив описание стиля, можно за одну операцию поменять оформление документа.

Программа использует два типа стилей: стиль абзаца и стиль символов. Стиль абзаца представляет собой группу форматов, которые влияют на весь абзац в целом: шрифт абзаца, расположение абзаца на полосе (влево, вправо, отступы и др.), табуляцию и т. д. Стиль символов включает в себя оформление, которое может быть применено к символам командами окна **Шрифт**.

Доступные стили перечислены в раскрывающемся списке **Стиль**, расположенном на панели инструментов **Форматирование**. В начале работы с программой Word этот список содержит перечень стилей, заданных по умолчанию. При выборе стиля из списка изменяется формат текущего абзаца или формат выделенного фрагмента.

Формат по образцу – копирование параметров форматирования выделенного объекта или текста. Это форматирование будет затем применено к выбранному объекту или тексту. Чтобы выполнить

форматирование по образцу, надо установить курсор на абзац, имеющий нужный метод форматирования, и щелкнуть на кнопке **Формат по образцу**  на панели инструментов **Стандартная**. Далее следует щелкнуть на абзаце, формат которого требуется изменить, и он будет выглядеть точно так же, как выбранный в качестве образца.

Если требуется изменить формат нескольких абзацев, надо дважды щелкнуть на кнопке **Формат по образцу**. После внесения всех необходимых изменений надо еще раз щелкнуть на кнопке **Формат по образцу** или нажать клавишу <Esc>.

Форматировать по образцу можно не только абзац, но и отдельные слова, символы или объекты.

2.2.3. Подготовка документа к печати

Расстановка переносов. Чтобы текст документа выглядел более компактным, в нем необходимо расставить переносы. Для этого надо выполнить команду **Сервис** > **Язык** > **Расстановка переносов**. В появившемся диалоговом окне следует установить флажок **Автоматическая расстановка переносов**. Word определяет слоги по словарю и правильно делает переносы.

Вставка колонтитула. Колонтитул – это текст, который печатается внизу или вверху каждой страницы документа. Обычно в область колонтитула вносят номер страницы, но можно внести и другие данные, такие как текущая дата, название документа и т. д., а также иллюстрации. Оформить колонтитул можно с помощью панели инструментов, которая появляется при выполнении команды **Вид** > **Колонтитулы**. Для завершения работы с колонтитулом необходимо нажать кнопку **Заккрыть** на панели **Колонтитулы**.

Можно установить разные колонтитулы для первой и последующих страниц, для четных и нечетных страниц. Колонтитулы форматировются так же, как и обычный текст. Для них можно установить границы и заливку.

Автоматическая нумерация страниц. Программа Word позволяет автоматически расставить номера страниц в документе. Для этого необходимо выполнить команду **Вставка** > **Номера страниц**. В открывшемся диалоговом окне (рис. 2.6) следует установить, выбрав из списка, следующие параметры: **Положение**, **Выравнивание**. Нажатие на кнопку **Формат** позволит указать, какими символами будет производиться нумерация (арабские цифры, римские, буквы) и с какого числа начнется нумерация.

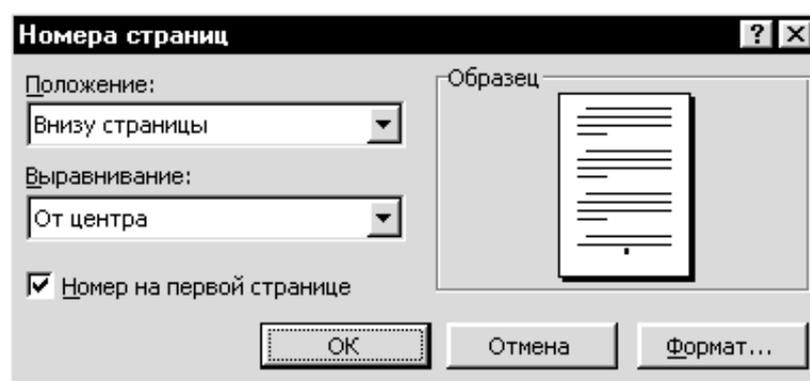


Рис. 2.6. Окно **Номера страниц**

Предварительный просмотр. Чтобы убедиться в том, что текст на бумаге будет выглядеть так, как надо, используют специальный режим предварительного просмотра. Желательно перед печатью любого файла просмотреть его в этом режиме. Во многих случаях это экономит время и бумагу. Для перехода в указанный режим служит кнопка **Предварительный просмотр**  или команда **Файл** > **Предварительный просмотр**. В этом режиме документ редактировать нельзя. Управляющие кнопки на появившейся панели инструментов позволяют только изменять способ отображения.

Пример 2.1. Создание документа.

1. Загрузите программу Word: **Пуск** > **Программы** > **Microsoft Word**.

2. Установите параметры страницы (левое поле – 3 см, правое – 1,5 см, нижнее и верхнее – 2 см) и абзацный отступ 1,25 см: **Файл** > **Параметры страницы**.

3. Наберите три абзаца любого текста, например свою автобиографию. Клавишу <**Enter**> нажимайте только в конце абзаца.

4. Просмотрите документ в четырех разных режимах (обычный, электронного документа, разметки страницы, структуры), используя пункт главного меню **Вид**. Щелкните кнопку **Непечатаемые знаки** и обратите внимание на то, как отображаются на экране символы пробела и конца абзаца.

5. Первый абзац выровняйте по ширине, второй – по левому краю, третий – по правому: выделите абзац, щелкните на соответствующей кнопке панели **Стандартная**.

6. Задайте для каждого абзаца свои параметры. Для первого: шрифт (гарнитура) – Times New Roman, размер (кегель) – 14, междустрочный интервал – одинарный. Для второго: шрифт –Tahoma, размер – 12, междустрочный интервал – полуторный. Для третьего:

шрифт – Courier New, размер – 14, междустрочный интервал – одинарный, интервал перед и после по 6 пт. Для этого выполните следующие действия: выделите абзац, выберите команду **Формат** ➤ **Шрифт**, установите нужный шрифт и размер, далее выполните **Формат** ➤ **Абзац**, выберите нужные интервалы междустрочный, перед и после.

7. Вставьте в начале документа заголовок (свою фамилию). Задайте его параметры: размер – 14 пт, гарнитура – Arial, начертание – полужирный курсив, выравнивание – по центру. Для этого поставьте курсор перед первой буквой набранного текста, нажмите <Enter>, сместите курсор на строку вверх и наберите заголовок, установите для него нужные параметры.

8. После заголовка вставьте символы ➤, ☎, ©: выполните **Вставка** ➤ **Символ**, выберите нужный символ, щелкните на кнопке **Вставка**, затем – **Заккрыть**.

9. Добавьте в конце текста перечень изучаемых дисциплин, оформленный в виде нумерованного списка: наберите **1.пробел** название предмета, нажмите клавишу <Enter>, далее автоматически происходит нумерация предметов. Чтобы закончить перечисление, нажмите <Enter> два раза.

10. Создайте маркированный список следующего вида:

- 8⁰⁵–9⁴⁰ – Математика
- 9⁵⁵–11³⁰ – История
- 11⁴⁵–13²⁰ – Информатика
- 13³⁵–15¹⁰ – Физика
- 15²⁵ – 16³⁰ – Кураторский час

Для этого щелкните на кнопке **Маркеры** панели **Стандартная**, наберите текст. Для указания времени наберите 805, выделите цифры 05, выполните команду **Формат** ➤ **Шрифт**, в окне **Шрифт** установите флажок **Верхний индекс**. Для набора тире нажмите сочетание клавиш <Ctrl> и <-> на дополнительной клавиатуре (расположена справа).

11. Измените вид маркеров в созданном списке: выделите список, выполните команду **Формат** ➤ **Список**, вкладка **Маркированный**, выберите нужный вид маркеров списка.

Сохраните файл в своей рабочей папке.

Пример 2.2. Подготовка документа к печати.

1. Расставьте переносы в тексте документа: **Сервис** ➤ **Язык** ➤ **Расстановка переносов**, включите флажок **Автоматическая расстановка переносов**.

2. Создайте верхний и нижний колонтитулы. В верхнем укажите свои фамилию и инициалы, в нижнем – текущую дату. Для этого выберите **Вид > Колонтитулы**. Появится панель **Колонтитулы**. Наберите фамилию и инициалы, выровняйте по центру, используя соответствующую кнопку. Для перехода в нижний колонтитул щелкните на кнопке **Верхний/Нижний колонтитул** панели **Колонтитулы**. Далее щелкните на кнопке **Дата**, выровняйте дату по правому краю, щелкните на кнопке **Закорыть** для выхода из режима колонтитулов.

3. Пронумеруйте страницы: **Вставка > Номера страниц**, выберите положение – внизу, выравнивание – от центра.

2.2.4. Работа с таблицами

Создание таблиц. Вставить пустую таблицу можно двумя методами: методом вставки или методом рисования.

В первом случае для вставки таблицы в текст необходимо установить курсор в то место, где должна быть таблица, выполнить команду **Таблица > Добавить таблицу** или нажать кнопку **Добавить таблицу**  панели инструментов **Стандартная**.

Во втором случае необходимо выполнить команду **Таблица > Нарисовать таблицу**, на появившейся панели инструментов **Таблицы и границы** выбрать инструмент **Нарисовать таблицу**  и указателем мыши, который приобретет вид карандаша, нарисовать прямоугольник. Затем, проводя горизонтальные и вертикальные линии, нарисовать необходимые столбцы и строки таблицы.

Таблицу можно создать также на основе уже набранного текста путем преобразования его в табличную форму. Для этого нужно определить, какой элемент в тексте можно использовать в качестве разделителей столбцов; главное, чтобы этот разделитель не встречался в качестве элементов основного текста. Обычно применяется знак табуляции: каждая строка таблицы набирается в один абзац, а столбец от столбца отделяется знаком табуляции. Если предполагается пустая ячейка, то на этом месте ставятся два знака табуляции. При преобразовании текста в таблицу число столбцов в таблице определится по максимальному числу столбцов в какой-либо строке. Итак, выбрав разделить, нужно ввести текст (или отформатировать существующий), затем выделить текст и выполнить операцию **Таблица > Преобразовать в таблицу**. В открывшемся диалоговом окне при необходимости уточняются параметры преобразования (тип разделителя, количество столбцов и т. д.).

Заполнение таблиц. Чтобы заполнить таблицу, нужно поместить курсор в ячейку и набрать текст. При наборе текста высота ячейки автоматически будет увеличиваться, чтобы вместить весь набранный текст. Помимо текста в ячейку таблицы можно импортировать графику.

Перемещение по таблице осуществляется с помощью указателя мыши, клавиш перемещения курсора или следующих комбинаций клавиш:

- к следующей ячейке – <Tab>;
- к предыдущей ячейке – <Shift> + <Tab>;
- к первой ячейке в строке – <Alt> + <Home>;
- к последней ячейке строки – <Alt> + <End>;
- к первой ячейке в столбце – <Alt> + <Page Up>;
- к последней ячейке в столбце – <Alt> + <Page Down>.

Редактирование таблиц. В созданной таблице можно добавлять и удалять ячейки, строки, столбцы, можно изменять ширину столбцов, разбивать таблицу на две самостоятельные, форматировать текст в ячейках, используя различные методы форматирования, копировать текст из одной ячейки в другую и т. д.

Выделение элементов таблицы. Для выделения элементов таблицы можно воспользоваться командами **Таблица** ➤ **Выделить строку (столбец)**.

Для выделения строки с помощью курсора необходимо щелкнуть мышью слева от таблицы напротив нужной строки. Для выделения столбца таблицы нужно щелкнуть над нужным столбцом (при этом курсор принимает вид жирной стрелки, направленной вниз). Для выделения ячейки необходимо щелкнуть мышью возле левой границы ячейки, в этом месте курсор приобретает вид стрелки, направленной в правый верхний угол. Выделить несколько ячеек можно, если протянуть по ним курсор мыши при нажатой клавише.

Копирование ячеек, строк и столбцов выполняется, как при работе с обычным текстом документа. Нужно выделить текст, скопировать его в буфер (или вырезать) и вставить в новом месте. Необходимо обратить внимание на следующую особенность: если при копировании выделить содержимое ячейки вместе с символом конца ячейки ☐ (этот символ виден при отображении непечатаемых знаков), то это приведет к замене содержимого всей ячейки на новые значения.

Вставка строк и столбцов. Для добавления в таблицу новой строки (столбца) нужно выделить всю строку (столбец) ниже (правее) места вставки новой строки и выполнить команду **Таблица** ➤ **Добавить строки (столбцы)**. Будет вставлена пустая строка с форматом

абзацев, совпадающим с форматом абзацев выделенной строки. Для добавления в таблицу нескольких строк (столбцов) нужно выделить такое количество строк (столбцов), которое необходимо вставить, и выполнить команду **Таблица ➤ Вставить строки (столбцы)**.

Нажатие клавиши <Tab> или <Enter> при нахождении курсора перед маркером конца строки в таблице приводит к созданию новой строки после этой строки таблицы. Чтобы вставить столбец правее самого последнего столбца таблицы, нужно выделить все маркеры конца строк таблицы (это можно сделать курсором или с помощью команды **Таблица ➤ Выделить столбец**) и выполнить команду **Таблица ➤ Вставить столбцы**.

Удаление строк и столбцов. Для удаления строк (столбцов) необходимо выделить их и выполнить команды **Таблица ➤ Удалить строки (столбцы)**. Если нажать клавишу при выделенном тексте ячеек, то будет удален только текст, а структура таблицы останется без изменений.

Удаление ячеек. Удаление ячеек отличается от удаления полных строк или столбцов тем, что при удалении отдельных ячеек изменяется структура таблицы, поэтому необходимо дополнительно указать, куда сдвигать остающиеся ячейки (вверх или влево). В остальном выполнение команды полностью аналогично удалению строк (столбцов).

Объединение и разделение ячеек. Для объединения нескольких ячеек необходимо выделить их и воспользоваться командой **Таблица ➤ Объединить ячейки**. Для разделения ячеек на несколько к выделенным ячейкам нужно применить команду **Таблица ➤ Разбить ячейки** и в появившемся диалоговом окне указать количество столбцов и строк, на которое нужно разделить ячейки.

Объединить или разделить ячейки можно также при помощи инструментов «карандаш» и «ластик» с панели инструментов **Нарисовать таблицу**.

Изменение параметров ячейки. Для изменения формата текста внутри ячейки необходимо выделить одну или несколько ячеек и воспользоваться командами **Шрифт** и **Абзац** из пункта меню **Формат** или панелью инструментов **Форматирование**. С помощью команды **Формат ➤ Направление текста** можно изменить направление текста в выделенной ячейке.

Для изменения заливки выделенных ячеек необходимо воспользоваться командой **Формат ➤ Границы и заливка**.

Изменение ширины столбца. Установить необходимую ширину столбца таблицы можно, установив курсор в нужный столбец и восполь-

зовавшись командой **Таблица** ➤ **Высота и ширина ячейки**. Чтобы установить одинаковую ширину для нескольких столбцов, нужно выделить их и выбрать команду **Таблица** ➤ **Выровнять ширину столбцов**.

Однако удобнее и быстрее изменять ширину столбцов с помощью мыши. Для этого нужно подвести курсор к границе столбца, ширину которого нужно изменить (при этом указатель мыши примет вид двух параллельных вертикальных линий со стрелками по бокам), и, удерживая левую клавишу мыши, перетащить границу столбца в необходимом направлении.

Для изменения ширины столбцов в зависимости от их содержимого можно выделить необходимые столбцы и воспользоваться командой **Высота и ширина ячейки** ➤ **Столбец** ➤ **Автоподбор** из пункта меню **Таблица**.

При операциях с изменением ширины таблицы необходимо обращать внимание, чтобы ширина ее не превысила ширину полосы набора, иначе это будет выглядеть как неряшливое оформление страницы, к тому же часть таблицы может оказаться вне области печати.

Разбиение таблиц. Если нужно разделить таблицу на две самостоятельные, например, чтобы вставить между частями таблицы какой-либо текст, то выполнить эту операцию можно командой **Таблица** ➤ **Разбить таблицу**. Для этого нужно установить курсор в строку, перед которой должна быть разделена таблица, и выполнить команду разбиения таблицы.

Оформление таблиц. Таблица может быть отформатирована с использованием различных линий и заливок. Для установок параметров форматирования можно воспользоваться командой **Формат** ➤ **Границы и заливка**.

Для быстрого форматирования таблицы можно воспользоваться командой **Таблица** ➤ **Автоформат** и в открывшемся диалоговом окне выбрать тип оформления в списке **Форматы**.

Вычисления в таблицах. Для выполнения вычислений нужно установить курсор в ячейку, где должен быть результат, и вставить формулу, используя команду **Таблица** ➤ **Ссылки на ячейки** ➤ **Формула**. В качестве формулы может быть записана в использованная готовая математическая формула, например $A1 + A2$, функция, которая выбирается из списка **Вставить функцию**, или собственная формула, написанная с использованием математических операторов и ссылок на ячейки.

2	A	B	C
3	A	B	C

Рис 2.7. Ячейки
таблицы

B1, B2, где латинская буква указывает на столбец, а число – на строку (рис. 2.7).

Пример 2.3. Создание, заполнение и редактирование таблиц.

В созданный ранее текстовый документ вставьте таблицу следующего вида:

№ опы- та	Температура			Выводы
	$t_1, ^\circ\text{C}$	$t_2, ^\circ\text{C}$	$t_3, ^\circ\text{C}$	

1. Для создания таблицы воспользуйтесь командой **Таблица** ➤ **Добавить таблицу** и в открывшемся диалоговом окне задайте число строк – 4 и число столбцов – 3:

1	2	3
4	5	6

2. Объедините ячейки под номерами 1 и 4, для чего выделите эти ячейки и выполните команду **Таблица** ➤ **Объединить ячейки**. Аналогичным образом объедините ячейки под номерами 3 и 6.

3. Поставьте курсор в первый столбец и с помощью команды **Таблица** ➤ **Высота и ширина ячейки** задайте ширину ячейки – 1,5 см. Аналогично задайте ширину для второго столбца – 7,5 см, для третьего – 5 см.

4. Просмотрите на горизонтальной линейке численные значения ширины всех столбцов, для этого установите курсор мыши на любую вертикальную линию сетки таблицы, нажмите на кнопку мыши и затем – на клавишу <Alt>.

5. Выделите вторую, третью и четвертую строки второго столбца и выполните команду **Таблица** ➤ **Разбить ячейки**, в появившемся диалоговом окне укажите количество столбцов – 3 (количество строк оставьте без изменения).

В итоге должна получиться таблица следующего вида:

1	2			3
	4	5	6	

--	--	--	--	--

6. Заполните ячейки следующим образом:
 - ячейки под номерами 1, 2, 3 – согласно образцу;
 - в ячейку с номером 4 введите надпись « t_1 , °C». При создании этой надписи используйте верхние и нижние символы;
 - выделите содержимое ячейки с номером 4 и с помощью буфера обмена скопируйте ее содержимое в ячейки 5 и 6;
 - измените содержимое ячеек 5 и 6 в соответствии с образцом;
 - измените направление текста в ячейке с номером 1. Для этого поставьте курсор в ячейку и выполните команду **Формат** ➤ **Направление текста**;
 - измените высоту ячейки 1. Для этого подведите курсор к нижней границе ячейки, при этом указатель мыши примет вид двух параллельных линий со стрелками по бокам, и, удерживая левую клавишу мыши, перетаскивайте границу ячейки вниз до тех пор, пока содержимое этой ячейки не поместится в ячейку целиком.
7. Добавьте в таблицу еще две строки.
8. Заполните все строки таблицы произвольными значениями.
9. Выровняйте содержимое ячеек по центру. Для этого выделите ячейки и примените к ним известные вам команды форматирования абзаца.
10. Выровняйте надпись «Выводы» по вертикали так, чтобы она располагалась по центру ячейки. Для этого выделите ячейку, нажмите правую клавишу мыши и в контекстном меню выберите команду **Выравнивание** ➤ **Центрировать по вертикали** или нажмите соответствующую кнопку на панели инструментов **Таблицы и границы**.
11. Удалите последнюю строку.
12. Выделите первый столбец и выполните команду **Формат** ➤ **Границы и заливка**, во вкладке **Заливка** задайте цвет заливки и тип узора.
13. Примените различные заливки к ячейкам заголовка таблицы.
14. Выделите таблицу и во вкладке **Границы** задайте тип границы – рамка, тип линии – двойной, цвет и толщину линии – по желанию.
15. Добавьте в конец таблицы строку и в первом столбце этой строки напишите «Среднее значение».
16. Вычислите среднее значение для каждого столбца. Для этого в диалоговом окне **Функция** необходимо выбрать функцию **Average (Above)**. Вычисления проводятся в каждой ячейке отдельно.
17. Поменяйте значения температуры в отдельных столбцах и

обновите поле с формулой.

2.2.5. Работа с графическими объектами

Для графического оформления текстового документа используют различные графические элементы. Это иллюстрации, декоративные надписи, графики, диаграммы.

Для грамотной работы с графическими элементами необходимо различать и уметь пользоваться разными видами графики. Выделяют три вида компьютерной графики: растровую, векторную и фрактальную. Они отличаются принципами формирования изображений.

Растровые изображения состоят из массива точек различного цвета. Растровыми являются все сканированные изображения и цифровые фотографии, а также изображения, созданные в графических редакторах, таких как Microsoft Paint. Основным недостатком растровой графики является потеря качества изображения при его увеличении. Данное обстоятельство связано с тем, что увеличение изображения приводит к увеличению точек, из которых оно состоит, а это визуально искажает изображение. Такой эффект называется пикселизацией. Наиболее распространенные форматы растровых изображений: BMP, JPG, GIF, PSD, PCX и TIFF.

Векторные рисунки создаются из элементарных фигур: линий, кривых, прямоугольников и других объектов. При изменении размеров векторного рисунка компьютер прорисовывает линии и фигуры заново, таким образом, не происходит искажения рисунка, характерного для растровых изображений. К векторным рисункам относятся картинки из коллекции Microsoft Office, рисунки, созданные с использованием графических средств пакета Word и специализированными графическими редакторами (CorelDraw, Macromedia Flash и т. д.).

Фрактальная графика основана не на рисовании, а на генерации изображения с использованием программирования. Фрактальная графика получила широкое распространение при создании трехмерной графики (в играх, программах для ландшафтного проектирования и т. д.).

Иллюстрации в текстовом документе можно условно разделить на векторные рисунки и растровые изображения. Векторные рисунки создаются самим пользователем средствами векторной графики, встроенными в Word. Это целесообразно при создании несложных рисунков (схем, простых чертежей). Такие рисунки легко редактируются и к тому же при изменении размеров не теряют качества. Растровые изображения вставляют в документ из готовых файлов. Это

могут быть фотографии, сканированные изображения и другие графические файлы.

Работа с рисунками. Для создания векторного рисунка в Word понадобится панель инструментов **Рисование** (рис. 2.8). Как правило, она расположена в нижней части окна программы, в случае ее отсутствия необходимо выполнить команду **Вид** > **Панели инструментов** > **Рисование**.



Рис. 2.8. Панель **Рисование**

Рисунок можно составить с помощью графических примитивов (линии, прямоугольника, окружности), а можно воспользоваться коллекцией автофигур, которая содержит фигурные стрелки, элементы блок-схем, различные выноски и прочие фигуры. Коллекция автофигур открывается списком **Автофигуры** на панели **Рисование**.

Для редактирования фигуры ее необходимо выделить, щелкнув на ней мышью. Выделенная фигура обозначается по краям белыми маркерами, при подводе курсора к ним он принимает вид двунаправленной стрелки. Если в этот момент нажать левую клавишу мыши и потянуть, то можно изменить размеры фигуры. Если при выделении фигуры появился маркер желтого цвета, то, потянув за этот маркер, можно изменить некоторые параметры фигуры (например, толщину куба, ширину ленты, количество углов в многоугольнике и т. д.). Для выделения нескольких фигур можно нажать на кнопку **Выбор объектов** на панели **Рисование** и выделить прямоугольную область с несколькими фигурами или выделять фигуры, щелкая на них мышью с нажатой клавишей <Shift>.

Для редактирования фигур на панели **Рисование** расположены следующие команды:

 – **цвет заливки** – позволяет выбрать цвета заливки из палитры, создать градиентную заливку, заливку текстурой, узором или рисунком;

 – **цвет линии** – позволяет изменить цвет линий и контуров фигур, выбрать узорные линии;

 – **цвет шрифта** – позволяет изменять цвет шрифта надписей, текста, добавленного в фигуры, а также любого выделенного текста в документе;

 – **тип линии** – позволяет изменить толщину линии, а также выбрать тип линии (к примеру, двойная линия);

 – **тип штриха** – позволяет выбрать тип штриха линии и контура фигуры;

 – **вид стрелки** – позволяет изменить вид стрелок или преобразовать в стрелки различные линии (кривые, дуги и т. д.);

 – **стиль тени** – позволяет назначить фигуре стиль тени, а с помощью команды **Настройка тени** можно сдвигать тень и менять ее цвет;

 – **объем** – позволяет придать фигуре стиль объема, а с помощью встроенной команды **Настройка объема** можно изменить направление перспективы, глубину объема, направление подсветки, цвет объема и даже придать фигуре эффекты различных поверхностей (матовая, металл и т. д.).

Помимо редактирования фигур с ними можно производить различные действия с помощью кнопок панели **Рисование**.

Команда **Действия** ➤ **Порядок** позволяет поместить фигуру перед текстом или за ним. Если фигуры наложены одна на другую, то можно назначить порядок их следования. Команда **Действия** ➤ **Повернуть / отразить** позволяет повернуть фигуру или получить ее зеркальное отображение, отразив вертикально или горизонтально. При вставке рисунка в текст полезно воспользоваться командой **Действия** ➤ **Обтекание текстом**. Для выравнивания нескольких фигур относительно краев или центра рисунка их необходимо выделить и воспользоваться командой **Действия** ➤ **Выровнять/распределить**.

Часто в создаваемые рисунки необходимо вносить различные надписи. Для этого можно воспользоваться кнопкой **Надпись** на панели **Рисование**. Для размещения текста внутри фигуры необходимо щелкнуть правой кнопкой мыши на фигуре и в контекстном меню выбрать команду **Вставить текст**.

После того как рисунок скомпонован из различных фигур, его необходимо сгруппировать. Для этого выделяют все фигуры (кнопка **Выбор объектов**) и выполняют команду **Действия** ➤ **Группировать**. При необходимости рисунок опять может быть разгруппирован.

Картинки из коллекции Microsoft Word представляют собой векторные рисунки, и для некоторых из них действуют команды панели **Рисование**. Такие рисунки можно разгруппировать и редактировать как обычные автофигуры.

Для создания различных текстовых эффектов используют объект **WordArt**. Для создания объекта **WordArt** нужно нажать на кнопку **Добавить объект WordArt** на панели **Рисование** или выполнить команду **Вставка** ➤ **Рисунок** ➤ **Объект WordArt**. Объект **WordArt** можно редактировать (изменять текст, цвет текста, его форму и т. д.) при помощи кнопок на панели инструментов **WordArt** (рис. 2.9).



Рис. 2.9. Панель **WordArt**

Работа с изображениями. Вставить готовое изображение в документ можно с помощью команды **Вставка** ➤ **Рисунок**. В случае вставки рисунка из коллекции Microsoft Office нужно выбрать команду **Картинки**, а для вставки произвольного файла с рисунком – команду **Из файла**.

При щелчке по вставленному изображению открывается панель **Настройка изображения** (рис. 2.10). Если этого не произошло, то нужно выполнить команду **Вид** ➤ **Панели инструментов** ➤ **Настройка изображения**.



Рис. 2.10. Панель **Настройка изображения**

Word не предоставляет возможности детального редактирования готовых растровых изображений, однако с помощью команд панели **Настройка изображения** можно изменить цвет изображения (в серых тонах, черно-белый и т. д.), увеличить (уменьшить) контрастность и яркость, выполнить поворот изображения, назначить обтекание текстом, изменить размеры.

Изображение можно подрезать. Для этого его выделяют, нажимают на кнопку **Обрезка** на панели **Настройка изображения** и, подведя курсор к маркеру со стороны предполагаемой обрезки, уменьшают размеры изображения. Для точного задания параметров обрезки в сантимет-

рах необходимо выполнить команду **Формат рисунка** из контекстного меню и в диалоговом окне **Формат рисунка** выбрать группу **Рисунок**.

Для оптимизации размеров файла текстового документа изображение можно сжать, воспользовавшись кнопкой **Сжатие рисунков** на панели **Настройка изображения**, однако при этом может ухудшиться качество изображения.

Для осуществления форматирования рисунка необходимо его выделить и выбрать соответствующую команду на панели инструментов **Настройка изображения**.

Изображения, созданные в графическом редакторе, имеют формат BMP. Однако файлы этого формата отличаются большими размерами, поэтому целесообразно сохранять файлы в формате JPG. Данный формат позволяет сжимать размер графического файла без видимого ухудшения качества. Этот процесс называется компрессия.

Из-за хорошей компрессии формат JPG наиболее популярен в Интернете, так как при передаче информации по сетям размер файлов имеет решающее значение. Формат GIF, который также применяется в Интернет-технологиях, желательно использовать для компрессии простых картинок, количество цветов в которых не превышает 255.

Для сохранения файла из редактора Paint в формате JPG необходимо в диалоговом окне **Сохранить как** в списке **Тип файла** выбрать jpeg или jpg.

Пример 2.4. Создание и редактирование графических объектов.

1. В графическом редакторе Paint создайте простой рисунок.
2. Сохраните его под именем **ris1** в формате BMP и под именем **ris2** в формате JPG.
3. В проводнике сравните размеры файлов **ris1.bmp** и **ris2.jpg**.
4. Вставьте рисунок в формате JPG в свой текстовый документ.
5. Выделите вставленный рисунок и откройте панель **Настройка изображения**. Командой **Цвет** измените цвет рисунка на оттенки серого, командой **Обтекание текстом** создайте обтекание вокруг рамки. Рассмотрите другие команды панели **Настройка изображения**.
6. Вставьте рисунок из коллекции Microsoft Office. Измените его размеры, обратите внимание на изменение качества изображения при изменении размеров.
7. С помощью панели **Рисование** создайте рисунок. При этом используйте команды **Цвет заливки**, **Цвет линий**, **Объем**. Сгруппи-

руйте все элементы созданного рисунка. Измените размер всего рисунка, перенесите его в другое место документа (для переноса можно воспользоваться буфером обмена или нажать на рисунке левую кнопку мыши и, не отпуская ее, перетянуть рисунок).

2.2.6. Вставка объектов

В текстовый документ можно вставлять различные объекты (рисунки, звуки, видео, диаграммы и т. д.). При этом объекты могут быть внедренными или связанными.

Вставку объекта в документ можно осуществить, используя буфер обмена. Для этого следует открыть импортированный файл (например рисунок, созданный в Paint), выделить информацию и скопировать ее, а затем вставить в текстовый документ.

Если для внедрения объекта воспользоваться командой **Вставка** ➤ **Объект** ➤ **Создание из файла**, то внедренный объект можно будет изменить, не выходя из Word, для этого достаточно дважды щелкнуть по данному объекту. Например, при внедрении рисунка, созданного в Paint, двойной щелчок по рисунку открывает его для редактирования, при этом панель **Стандартная** программы Word заменяется соответствующей панелью программы Paint. Для выхода из режима редактирования достаточно щелкнуть за пределами объекта в любом месте исходного текстового документа. При внедрении объектов размер исходного текстового документа увеличивается на величину внедренного объекта.

Если при внедрении объекта на вкладке **Создание из файла** диалогового окна **Выбор объекта** поставить флажок **Связать с файлом**, то вставляемый объект будет связанным, это значит, что в исходный документ помещается только указатель на местоположение вставляемого объекта. При открытии документа со связанным объектом Word обращается по указанному адресу и отображает внешний вид связанного объекта на момент открытия.

Использовать связанные объекты удобно в тех случаях, когда размер импортируемых файлов слишком велик (так как импортируемый файл хранится отдельно, то размер исходного документа не увеличивается) или когда импортируемые файлы используются несколькими пользователями (например, в корпоративных сетях). Однако следует помнить, что, во-первых, связанные объекты должны находиться по указанному адресу (если их переместить в другое место, то произойдет потеря связи), а во-вторых, при переносе документов со связанными объектами (например, на гибких носителях) следует переносить и связанные с ними объекты, иначе они не отобразятся в документе.

Объект MS Equation. С помощью программы MS Equation в текстовом документе можно создавать формулы различной сложности. Для этого необходимо выбрать математические символы с панели **Формула** (рис. 2.11) и ввести необходимые переменные и числа с клавиатуры.

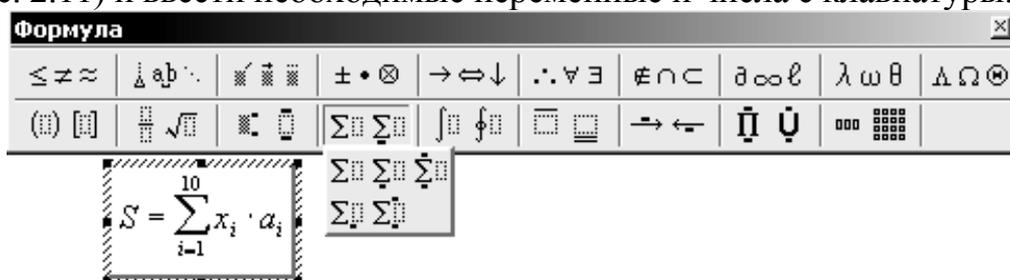


Рис. 2.11. Панель **Формула**

В верхней строке панели инструментов **Формула** содержится более чем 150 математических символов, а в нижней строке можно выбрать различные математические шаблоны, такие как: значки дроби, интеграла, суммы, матрицы и т. д.

Для возврата в текстовый документ нужно щелкнуть в любом месте документа вне формулы. Чтобы изменить формулу, достаточно дважды щелкнуть на ней, при этом откроется панель редактора формул.

Вставка файла. Для вставки готового файла в документ нужно установить курсор на место вставки, открыть диалоговое окно **Вставка** ➤ **Файл**, ввести имя файла или выбрать файл в списке. Редактируется вставленный файл как обычный текст.

Таким же образом в текстовый документ можно вставить текст программы, написанной на языке Pascal. Поскольку файл, созданный в программе TurboPascal, имеет отличное от файлов, созданных в Word, расширение, то в диалоговом окне **Вставка файла** в списке **Тип файлов** необходимо выбрать **Все файлы(*.*)**.

2.2.7. Автоматическое оглавление

При работе с большими документами удобно (а иногда и просто необходимо) применять автоматическое оглавление. Использование автоматического оглавления позволяет быстро переходить к выбранному разделу документа, автоматически изменять номера страниц в оглавлении при вставке или удалении частей документа, а также автоматически изменять наименование заголовков при их изменении в тексте документа.

Для быстрого создания оглавления к заголовкам документа нужно применить стили заголовков (**Заголовок 1**, **Заголовок 2**

и т. д.). При этом необходимо учитывать, что номер заголовка указывает на его положение в иерархии заголовков. Например, стиль **Заголовок 1** может быть использован для заголовка главы, а стиль **Заголовок 2** – для заголовка параграфа и т. д.

После применения к заголовкам документа стилей заголовков необходимо выполнить команду **Вставка** ➤ **Оглавление и указатели**, при этом курсор должен быть установлен в том месте документа, где предполагается разместить оглавление. Далее в диалоговом окне **Оглавление и указатели** нужно выбрать вкладку **Оглавление** и задать внешний вид оглавления и его параметры: число уровней, внешний вид заполнителя, равенные номеров страниц и т. д.

Пример 2.5. Создание комплексного документа.

1. Создайте заголовок для своего текстового документа, используя объект **WordArt**.

2. Откройте панель инструментов **WordArt** и, используя кнопки на этой панели, измените форму надписи, цвет линий и заливки.

3. Вставьте в свой текстовый документ следующие математические формулы:

$$S = \sum_{i=1}^{10} \left(\frac{a_i + 1}{i + 10} \right)^2; \quad I = \int_1^7 \frac{\sqrt{x^2 + 38} - \cos(x)}{x^2 + x + 25} dx;$$

$$\frac{\operatorname{tg}(\alpha + \beta)}{\operatorname{tg}\alpha} = \frac{\sin^2 \alpha \cos \beta}{\sin^2 \beta \cos \alpha} + \sqrt{\cos \alpha}.$$

4. Обрамите каждую формулу рамкой. Для этого выделите формулу, щелкнув на ней мышью, и выполните команду **Формат** ➤ **Границы и заливка**.

5. Вставьте в документ любой файл, содержащий программу на Pascal (файл с расширением pas).

6. Во всем документе расставьте заголовки, например: «Форматирование шрифта», «Форматирование абзаца», «Вставка таблиц», «Вставка рисунка» и т. д.

7. Воспользовавшись списком **Стили** на панели инструментов **Форматирование**, обозначьте стили написанных заголовков как **Заголовки**.

8. В конце документа вставьте автоматическое оглавление.

2.3. Общие сведения о Microsoft Excel

Программа Microsoft Excel предназначена для работы с таблицами данных, преимущественно числовых, с использованием формул и функций. При формировании таблицы выполняют ввод, редактирование и форматирование текстовых и числовых данных, а также формул. Документ Excel называется рабочей книгой. Рабочая книга представляет собой набор рабочих листов, каждый из которых имеет табличную структуру и может содержать одну или несколько страниц. В окне документа в программе Excel отображается только текущий рабочий лист.

Каждый рабочий лист имеет название, которое отображается на ярлычке листа в его левой нижней части. Рабочий лист состоит из строк и столбцов. Столбцы озаглавлены прописными латинскими буквами и, далее, двухбуквенными комбинациями. Строки последовательно нумеруются цифрами. На пересечении столбцов и строк образуются ячейки таблицы. Они являются минимальными элементами для хранения. Обозначение отдельной ячейки сочетает в себе номера столбца и строки, на пересечении которых она расположена, например: A1 или F7. Обозначение ячейки (ее номер) выполняет функции ее адреса. Адреса ячеек используются при записи формул, определяющих взаимосвязь между значениями, расположенными в разных ячейках. Одна из ячеек всегда является активной и выделяется рамкой активной ячейки. Операции ввода и редактирования всегда производятся в активной ячейке.

На данные, расположенные в соседних ячейках, можно ссылаться в формулах как на единое целое. Такую группу ячеек называют диапазоном. Наиболее часто используют прямоугольные диапазоны, образующиеся на пересечении группы последовательно идущих строк и группы последовательно идущих столбцов. Диапазон ячеек обозначают, указывая через двоеточие номера ячеек, расположенных в противоположных углах прямоугольника, например: A2:D15.

Если требуется выделить прямоугольный диапазон ячеек, это можно сделать протягиванием указателя от одной угловой ячейки до противоположной по диагонали. Рамка текущей ячейки при этом расширяется, охватывая весь выбранный диапазон. Чтобы выбрать столбец или строку целиком, следует щелкнуть на заголовке столбца (строки). Протягиванием указателя по заголовкам можно выбрать несколько идущих подряд столбцов или строк.

2.3.1. Окно программы Excel

Структура окна программы Excel типична для приложений Windows. Окно программы Excel состоит из строки заголовка, строки меню, панели инструментов, строки формул, рабочей таблицы (столбцы, строки, рабочее поле), строки состояния.

Под панелями инструментов Microsoft Excel обычно находится строка формул (рис. 2.12), а в нижней части окна – строка состояния.

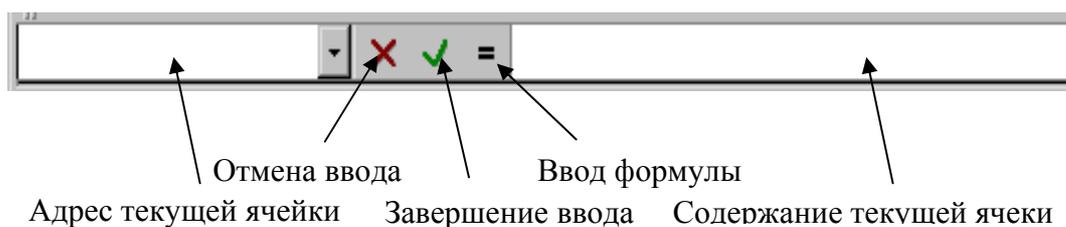


Рис. 2.12. Строка формул

Наличие на экране различных элементов окна Excel (как и в Word) зависит от команд меню **Вид**. Например, чтобы вывести или убрать данные строки, следует в меню **Вид** выбрать соответствующие пункты: **Строка формул** или **Строка состояния**. Чтобы добавить необходимую панель инструментов, используют команду **Вид > Панель инструментов**.

Имена рабочих листов (**Лист1**, **Лист2**, ...) выведены на ярлыках в нижней части окна рабочей книги возле горизонтальной полосы прокрутки. Щелкая на ярлыках, можно переходить от листа к листу внутри рабочей книги. Ярлыки листов можно переименовывать несколькими способами:

- щелкнуть на названии листа правой кнопкой мыши и в контекстном меню выбрать команду **Переименовать**;
- выполнить команду **Формат > Лист > Переименовать**;
- сделать двойной щелчок левой кнопкой мыши на названии листа.

Ввод и редактирование данных. Для ввода данных в ячейку необходимо сделать ее активной и потом ввести данные. Данные появятся в ячейке и в строке формул. Для завершения ввода следует нажать **<Enter>**. Процесс ввода данных закончится, и активной станет соседняя ячейка. В ячейку можно ввести дату, текст, число, формулу, рисунок. При вводе формулы в ячейке будет отображаться результат вычислений по формуле. Сама же формула будет видна на экране в

правой части строки формул, где после щелчка мышью можно будет эту формулу редактировать. Для форматирования информации в ячейке используют функции меню **Формат** > **Ячейки**.

Ввод даты. При вводе даты используется точка или дефис в качестве разделителя, например 09.05.96 или Янв-96. Варианты зависят от формата ячейки. Для отображения времени суток в 12-часовом формате вводится буква а или р, отделенная пробелом от значения времени, например 9:00 р. Чтобы ввести текущую дату, надо нажать клавиши <Ctrl> + <;> (точка с запятой). Чтобы ввести текущее время, надо нажать клавиши <Ctrl> + <Shift> + <:> (двоеточие).

Ввод числа. В Microsoft Excel число может быть записано с помощью следующих символов: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, +, -, (,), ,, /, \$, %, ., E, e. Стоящие перед числом знаки «+» игнорируются, а запятая интерпретируется как разделитель десятичных разрядов. Все другие сочетания клавиш, состоящие из цифр и нецифровых символов, рассматриваются как текст. Перед отрицательным числом необходимо вводить знак «-» или заключать число в круглые скобки. Введенные числа выравниваются в ячейке по правому краю. Если число не помещается в ячейку, то в зависимости от ее ширины число преобразуется к экспоненциальной форме (например 1,7E+05 – это число 170 000) или ячейка заполняется значками #####.

Каждое число в таблице можно представить в разных форматах (с различным количеством десятичных знаков, незначащих нулей и пр.). Для изменения формата содержимого ячейки необходимо:

- выделить ячейки;
- выбрать команду **Ячейки** меню **Формат**;
- в диалоговом окне **Формат ячеек** выбрать вкладку **Число**;
- в списке **Числовые форматы** выбрать тип формата содержимого ячейки, а в полях справа – параметры формата;
- в поле **Образец** будет отображаться пример содержимого ячейки в выбранном формате;
- чтобы ввести новый формат, следует выбрать пункт **Все форматы**, а затем в поле **Тип** ввести новый формат (например: ДД/ММ/ГГ – 01/02/03; 0,00% – 8,00%);
- щелкнуть **ОК**.

Ввод текста. В Microsoft Excel текстом является любая последовательность символов, состоящая из цифр, пробелов и нецифровых символов, например 10AA109. Введенный текст выравнивается в ячейке по левому краю. Если текст не помещается в ячейку, то он показывается поверх следующих

справа пустых ячеек либо показывается его часть, если ячейка справа содержит данные.

Ввод формул. Формула в Excel начинается со знака «=», «+» или «-» и содержит операторы, имена функций, числа, адреса ячеек, соединенные знаками арифметических операций.

После ввода формулы в ячейке выводится результат расчета, а сама формула отображается в строке формул. Для просмотра формул во всех ячейках, а не результата, надо выполнить команду **Сервис** ➤ **Параметры** и во вкладке **Вид** поставить в группе **Параметры окна** флажок **формулы**. Ячейка, содержащая формулу, называется зависимой ячейкой – ее значение зависит от значения другой ячейки.

В Microsoft Excel содержится большое количество стандартных формул, называемых функциями. Функции используются для простых и сложных вычислений. Аргументы функций – числа, текст, адреса ячеек и другие функции – записываются в круглых скобках после их названий и отделяются друг от друга символом «;». Адреса ячеек можно набирать вручную (но это не практично!) в точке ввода, используя латинские буквы и арабские цифры, или получать после щелчков мыши на соответствующих ячейках. Диапазон ячеек можно записать с использованием символа двоеточие «:» и путем выделения смежных ячеек. В текущей формуле адрес ячейки, принадлежащей другому (неактивному) листу, должен содержать имя листа, знак «!» и адрес ячейки на другом листе, например: Лист2!J5.

Ценность формул состоит в том, что, не изменяя однажды созданную формулу, можно неоднократно изменять содержимое ячеек, адреса которых присутствуют в формуле, и получать новые результаты вычислений в ячейке.

Наиболее распространенной, является функция СУММ ()^Σ, суммирующая значения диапазона ячеек. Для быстрого суммирования значений интервала ячеек выделите нужные ячейки, включая и пустую справа или снизу, затем щелкните на кнопке **Автосумма** . Программа Excel подведет итоги.

Ввод последовательностей чисел, дат и текста. Создать последовательность данных можно двумя путями:

- отбуксировав маркер автозаполнения (маленький квадратик в нижнем правом углу активной ячейки);
- выбрав команду **Правка** ➤ **Заполнить** ➤ **Прогрессия**.

Создание текстовой последовательности. Excel распознает простые текстовые данные, такие как дни, месяцы и квартальные аббревиатуры (Январь, Февраль, Март, ...; Пн, Вт, Ср, Чт, ...). Для создания личных текстовых последовательностей предназначена команда **Сервис** ➤ **Параметры**, вкладка **Списки**.

Для заполнения интервала ячеек текстовой последовательностью необходимо выполнить следующие шаги:

1. Выбрать первую ячейку, содержащую данные.

2. Протянуть маркер автозаполнения через соседние ячейки, которые нужно заполнить.

3. Отпустить кнопку мыши.

Создание числовой последовательности. Для заполнения интервала ячеек последовательностью чисел нужно выполнить следующие шаги:

1. Ввести первое число. Если необходимо, чтобы интервал ячеек был заполнен с заданным вами шагом, нужно ввести первые два значения в соседних ячейках.

2. Протянуть маркер автозаполнения через соседние ячейки, которые нужно заполнить.

3. Отпустить кнопку мыши.

Для заполнения нескольких ячеек одной и той же формулой достаточно занести формулу только в первую ячейку диапазона, остальные ячейки заполняются буксировкой за маркер автозаполнения как по строке, так и по столбцу, при этом адрес ячейки в формуле меняется относительно строк или столбцов – так называемая относительная адресация.

Очень важным техническим приемом работы с ячейками или листами является их выделение, так как только над выделенными ячейками или листами можно производить в данный момент одну из множества операций (например: копирование, удаление, вырезание, изменение шрифта, выравнивание горизонтальное или вертикальное, печать).

Способы выделения. Выделение ячейки выполняют щелчком мыши на ней. Для выделения нескольких несмежных групп ячеек следует выделить одну группу, нажать клавишу <Ctrl> и, не отпуская ее, выделить другие необходимые ячейки. Чтобы выделить целый столбец или строку таблицы, необходимо щелкнуть мышью на их имени. Для выделения нескольких столбцов или строк следует щелкнуть на имени первого столбца или строки и растянуть выделение на требуемую область. Для выделения рабочего листа надо щелкнуть на левой верхней (пустой) ячейке пересечения названий столбцов и строк. Для выделения нескольких листов необходимо нажать клавишу <Ctrl> и, не отпуская ее, щелкать на ярлыках листов.

Копирование данных рабочего листа. Копировать данные в программе Excel можно методом «Перетащить и отпустить», с использованием буфера обмена или в соседние ячейки с помощью автозаполнителя.

Копирование данных методом «Перетащить и отпустить»:

1. Выделить интервал ячеек, которые вы хотите скопировать.

2. Поместить курсор мыши на границу выделения.
3. Нажать клавишу <Ctrl>, не отпуская ее, щелкнуть левой кнопкой мыши и, удерживая кнопку и клавишу, переместить курсор в новое место. При этом появится «бегущая» рамка, определяющая размер и положение копируемых данных.
4. Отпустить кнопку мыши, чтобы копируемые данные заняли новое положение.

Копирование данных с помощью буфера обмена:

1. Выделить интервал ячеек, которые нужно скопировать.
2. Выбрать команду **Правка** ➤ **Копировать**. Можно также щелкнуть правой кнопкой мыши и выбрать в контекстном меню команду  **Копировать**. Вокруг выделенной области появится бегущая рамка.
3. Пометить ячейку, в которую нужно вставить копию данных.
4. Выбрать команду **Правка** ➤ **Вставить**. Можно также щелкнуть правой кнопкой мыши и выбрать в контекстном меню команду  **Вставить**. Если нужно вставить только одну копию, то нажать клавишу <Enter>.

Копирование данных с использованием автозаполнителя:

1. Выделить ячейку с данными, которые нужно скопировать.
2. Поместить указатель мыши на маркер автозаполнителя в правом нижнем углу ячейки.
3. Нажать кнопку мыши и, не отпуская ее, протянуть маркер автозаполнителя по соседним ячейкам, в которые необходимо скопировать данные. Отпустить кнопку мыши.

Перемещение данных рабочего листа. Кроме копирования, можно перемещать данные рабочего листа из одной области в другую. При этом можно использовать метод «Перетащить и отпустить» или использовать буфер обмена.

Удаление данных. Для удаления символа слева или справа от указателя ввода используются соответственно клавиши или <Backspace>.

Вставка и удаление столбцов, строк и ячеек. Программа Excel позволяет редактировать данные рабочего листа, вставляя или удаляя столбцы, строки и ячейки. Выполняется это с помощью соответствующей команды из меню **Вставка** или из контекстного меню, открываемого щелчком правой кнопки мыши в момент нахождения курсора на некоторой ячейке (для строк, столбцов, ячеек) или между названиями листов (для листов). Изменения структуры рабочего

листа влекут за собой более серьезные последствия, чем перемещение и копирование данных в новое место.

Чтобы отредактировать данные в ячейке, необходимо:

- сделать ячейку активной и щелкнуть на строке формул, или нажать клавишу <F2>, или дважды щелкнуть на ячейке мышью;
- в ячейке появится текстовый курсор, который можно передвигать клавишами управления курсором;
- отредактировать данные;
- выйти из режима редактирования нажатием клавиши <Enter>.

Перед выполнением любой команды Microsoft Excel следует завершить работу с ячейкой, т. е. выйти из режима ввода или редактирования.

2.3.2. Форматирование таблицы

Выравнивание содержимого ячеек. Содержимое ячеек может быть выровнено по левому краю, по правому краю или по центру. На новом рабочем листе все ячейки имеют формат **Обычный**, в котором числа, даты и время выравниваются по правому краю ячейки, текст – по левому, а логические значения ИСТИНА и ЛОЖЬ – центрируются. Изменение выравнивания не влияет на тип данных. Для выравнивания содержимого ячеек необходимо:

- выделить ячейки, которые следует отформатировать;
- в меню **Формат** выбрать команду **Ячейки**;
- выбрать вкладку **Выравнивание**;
- выбрать тип выравнивания в списках полей **по горизонтали** и **по вертикали**;
- в группе флажков **Отображение** можно включить следующие режимы:

– **переносить по словам** – по достижению правой границы ячейки текст будет переноситься на новую строку;

– **автоподбор ширины** – размер символов уменьшается так, что содержимое ячейки помещается в границах ячейки;

– **объединение ячеек** – выделенные ячейки объединяются в одну;

- в рамке **Ориентация** выбирается направление расположения содержимого в ячейке – текст можно расположить вертикально или под углом.

Для быстрого выравнивания данных в ячейках используются кнопки пиктографического меню – .

Чтобы выровнять текст по центру нескольких столбцов, необходимо:

- выделить ячейку, содержащую данные, которые нужно выровнять по центру нескольких столбцов, и пустые ячейки, находящиеся справа;

- щелкнуть кнопку .

Установка шрифта. Для установления шрифта необходимо в меню **Формат** выбрать команду **Ячейки > Шрифт > Установить необходимые параметры**. Для быстрого форматирования символов используется панель инструментов **Форматирование**.

Оформление таблиц. Таблицы в Microsoft Excel можно обрмить рамкой и заполнить различными цветами. Для обрамления необходимо выделить ячейки, которые нужно обрмить, затем в меню **Формат > Ячейки** выбрать вкладку **Граница** и установить тип и цвет линии, указать внутренние или внешние границы.

Создавать рамки можно также с помощью пиктограммы **Границы** .

Изменение размеров строк и столбцов. По умолчанию ячейки имеют стандартную ширину и высоту. Высота строки определяется размером шрифта. Для изменения высоты строки или ширины столбца можно перетянуть границу заголовка до необходимого значения (на границе заголовка указатель мыши примет вид двунаправленной стрелки) (рис. 2.13). Для изменения размеров сразу нескольких столбцов или строк следует их выделить и перетянуть границу заголовка одного из выделенных элементов. Если на границе заголовков столбцов дважды щелкнуть мышью, то ширина столбца установится по ширине ячейки с самым длинным содержимым.

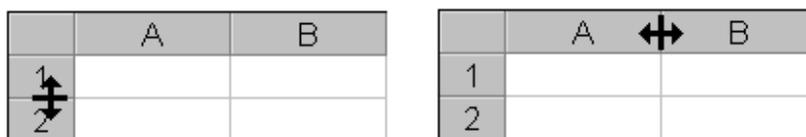


Рис. 2.13. Изменение размеров строк и столбцов

Для точного установления ширины столбцов (строк) необходимо сделать соответствующие установки размеров столбцов и строк с помощью меню **Формат > Столбец (Строка) > Ширина столбца (строки)**. Команда **Автоподбор ширины** устанавливает ширину столбца по ширине ячейки с самым длинным содержимым. Команда **Стандартная ширина** предлагает изменить стандартную ширину для

столбцов рабочего листа. Аналогичные действия можно выполнить и для точной установки высоты столбцов или строк.

Сортировка данных. Сортировка позволяет переупорядочить строки в таблице по любому полю. Для сортировки данных следует выделить одну ячейку таблицы и вызвать команду **Сортировка меню Данные**.

В поле списка **Сортировать по** выбирается поле, по которому будут отсортированы данные, и тип сортировки:

- **по возрастанию** – цифры сортируются по возрастанию, текст – в алфавитном порядке;
- **по убыванию** – сортировка в обратном порядке.

В поле списка **Затем по** указывается поле, по которому будут отсортированы данные, имеющие одинаковые значения в первом ключевом поле. Во втором поле **Затем по** указывается поле, по которому будут отсортированы данные, имеющие одинаковые значения в первых двух ключевых полях.

Для сортировки данных также предназначены кнопки . Перед их использованием следует выделить столбец, по которому необходимо сортировать записи.

При сортировке по одному столбцу строки с одинаковыми значениями в этом столбце сохраняют прежнее упорядочение. Строки с пустыми ячейками в столбце, по которому ведется сортировка, располагаются в конце сортируемого списка. Microsoft Excel позволяет также сортировать не всю таблицу, а только выделенные строки или столбцы.

2.3.3. Работа с формулами

Вычисления в таблицах выполняются с помощью формул. Формула может состоять из математических операторов, констант, ссылок на ячейку и имен функций. Результат выполнения формулы есть некоторое новое значение, содержащееся в ячейке с формулой. Формула в Excel начинается со знака «=», «+» или «-». Чтобы формула отображалась не только в строке формул, но и в ячейке, необходимо выполнить команду **Сервис > Параметры > Вид** и установить флажок **Формула**.

Сообщения об ошибках. Если формула в ячейке не может быть правильно вычислена, Microsoft Excel выводит в ячейку сообщение об ошибке. Если формула содержит ссылку на ячейку, которая содержит значения ошибки, то вместо этой формулы также

будет выводиться сообщение об ошибке. Значение сообщений об ошибках следующее:

#ИМЯ? – Microsoft Excel не смог распознать имя, использованное в формуле;

#ДЕЛ/0! – в формуле делается попытка деления на нуль;

#ЧИСЛО! – нарушены правила задания операторов, принятые в математике;

#Н/Д – такое сообщение может появиться, если в качестве аргумента задана ссылка на пустую ячейку;

#ПУСТО! – неверно указано пересечение двух областей, которые не имеют общих ячеек;

#ССЫЛКА! – в формуле задана ссылка на несуществующую ячейку;

#ЗНАЧ! – использован недопустимый тип аргумента.

Копирование формул. Когда копируется формула, адреса ячеек, содержащиеся в ней, изменяются в зависимости от ее нового расположения. В программе Microsoft Excel такой тип адресации называется относительной ссылкой. Если при копировании необходимо оставить адреса прежними, то в формуле используется абсолютная ссылка. В случае абсолютной ссылки слева от адреса столбца и/или строки появляется знак доллара (\$), например \$A\$1. Если нужно применить абсолютную ссылку к столбцу, а относительную – к строке и наоборот, тогда используется смешенная ссылка, например \$A1 или A\$1.

Необходимо сделать ссылку на ячейку абсолютной перед тем, как копировать формулу. Для этого надо выполнить следующие действия:

1. Выделить ячейку с формулой.
2. Дважды щелкнуть на ячейке.
3. Установить курсор на адрес ячейки и нажать клавишу <F4>.

При этом программа Microsoft Excel вставит знак доллара.

2.3.4. Функции

Функции в MS Excel применяются для выполнения стандартных вычислений. Значения, которые используются для вычисления, называются аргументами. Значения, возвращаемые функциями в качестве ответа, называются результатами.

Аргументы функции записываются в круглых скобках сразу за названием функции и отделяются друг от друга символом точка с запятой «;». Скобки позволяют MS Excel определить, где начинается и где заканчивается список аргументов.

В качестве аргументов можно использовать числа, текст, логические значения, массивы, значения ошибок или ссылки. Аргументы могут быть как константами, так и формулами. В свою очередь эти формулы могут содержать другие функции. Функции, являющиеся аргументом другой функции, называются вложенными. В формулах MS Excel можно использовать до семи уровней вложенности функций.

Задаваемые входные параметры должны иметь допустимые для данного аргумента значения. Некоторые функции могут иметь необязательные аргументы, которые могут отсутствовать при вычислении значения функции.

Рассмотрим случай, когда у функции нет аргументов. Примерами таких функций являются ПИ, которая возвращает число 3,14, или функция СЕГОДНЯ, возвращающая текущую дату. При вводе таких функций нужно сразу после названия функции поставить круглые скобки. Если вы хотите получить в ячейке число π (пи) или текущую дату, то введите в ячейки формулы следующего вида:

=ПИ()

=СЕГОДНЯ()

Для того чтобы вычислить значение функции, введите в ячейку равенство «=», а затем название функции и список ее аргументов. Нельзя вставлять пробелы между названием функции и скобками, в которых записаны аргументы. В противном случае MS Excel выдаст сообщение об ошибке #ИМЯ.

Статистические функции используются для статистического анализа данных. К ним относятся такие функции, например, как: МИН(), МАКС(), СРЗНАЧ() и т. д.

Математические функции образуют одну из многочисленных категорий среди функций. С помощью них значительно облегчается процесс вычисления данных. К ним относятся такие функции, как: COS(), SIN(), КОРЕНЬ() и т. д.

Ввод в формулы дат и времени. Программа Excel преобразует значения даты и времени в так называемые сериальные числа и использует их при вычислениях. В программе число 1 соответствует значению даты 01.01.1900, максимальное значение 65 380 – дате 31.12.2078. Когда используется дата или время в формуле, их вводят в формате, принятом в программе Excel, и заключают их в двойные кавычки. Затем программа сама выполнит необходимые преобразования. Например, чтобы найти количество дней между двумя числами,

необходимо ввести следующую формулу: ="2.4.97"->"27.3.97"(результат равен шести дням).

Эту задачу можно решить также, используя формулу =A1-A2, если в ячейке A1 будет введена дата 2.4.97, а в A2 – 27.3.97. Результат будет равен 6, если установить формат результирующей ячейки **Числовой**.

Логические функции предназначены для проверки выполнения условия или для проверки нескольких условий. Так, функция ЕСЛИ позволяет определить, выполняется ли указанное условие, и возвращает одно значение, если условие истинно, и другое, если оно ложно.

Функция И возвращает значение ИСТИНА, если все аргументы имеют значение ИСТИНА; возвращает значение ЛОЖЬ, если хотя бы один аргумент имеет значение ЛОЖЬ.

И(логическое_значение1;логическое_значение2;...)

Логическое_значение1, логическое_значение2, ... – это от 1 до 30 проверяемых условий, которые могут иметь значение либо ИСТИНА, либо ЛОЖЬ.

Аргументы должны быть логическими значениями, массивами или ссылками, которые содержат логические значения. Если аргумент, который является ссылкой или массивом, содержит тексты или пустые ячейки, то такие значения игнорируются. Если указанный интервал не содержит логических значений, то И возвращает значение ошибки #ЗНАЧ!.

Примеры:

И(ИСТИНА;ИСТИНА) равняется ИСТИНА;

И(ИСТИНА;ЛОЖЬ) равняется ЛОЖЬ;

И(2+2=4;2+3=5) равняется ИСТИНА.

Функция ЕСЛИ используется для условной проверки значений и формул.

ЕСЛИ(лог_выражение;значение_если_ИСТИНА;значение_если_ЛОЖЬ)

Лог_выражение – это любое значение или выражение, которое при вычислении дает значение ИСТИНА или ЛОЖЬ. Значение_если_ИСТИНА – это значение, которое возвращается, если лог_выражение – ИСТИНА. Если лог_выражение – ИСТИНА и значение_если_ИСТИНА опущено, то возвращается значение ИСТИНА. Значение_если_ИСТИНА может быть другой формулой.

Значение_если_ЛОЖЬ – это значение, которое возвращается, если лог_выражение – ЛОЖЬ. Если лог_выражение – ЛОЖЬ и

значение_если_ЛОЖЬ опущено, то возвращается значение ЛОЖЬ. Значение_если_ЛОЖЬ может быть другой формулой.

До 7 функций ЕСЛИ могут быть вложены друг в друга в качестве значений аргументов значение_если_ИСТИНА и значение_если_ЛОЖЬ. Функция ЕСЛИ всегда возвращает значение, возвращаемое вычисленным аргументом.

Если какой-либо аргумент функции ЕСЛИ является массивом, то при выполнении этой функции вычисляется каждый элемент массива. Если какой-либо из аргументов значение_если_ИСТИНА или значение_если_ЛОЖЬ является действием, то все действия выполняются.

В формуле

ЕСЛИ(A10=100;СУММ(B5:B15); " ")

если значение ячейки A10 – 100, то лог_выражение имеет значение ИСТИНА и вычисляется сумма для ячеек B5:B15. В противном случае лог_выражение имеет значение ЛОЖЬ и возвращается пустой текст (" "), очищающий ячейку, которая содержит функцию ЕСЛИ.

Предположим, что рабочий лист по расходам содержит в ячейках B2:B4 фактические расходы за январь, февраль, март: 1500, 500 и 500 соответственно; в ячейки C2:C4 помещены данные по предполагаемым расходам за те же периоды: 900, 900 и 925:

	A	B	C	D
1		Практические расходы	Предполагаемые расходы	
2	Январь	1500	900	Превышение бюджета
3	Февраль	500	900	ОК
4	Март	500	925	ОК

Можно написать формулу для проверки соответствия бюджету расходов определенного месяца, генерирующую тексты сообщений:

ЕСЛИ(B2>C2; "Превышение бюджета";"ОК")

Функция НЕ меняет на противоположное логическое значение своего аргумента. Функция НЕ используется в тех случаях, когда необходимо быть уверенным в том, что значение не равно некоторой конкретной величине.

НЕ(логическое_значение)

Логическое_значение – это значение или выражение, которое при вычислении дает ИСТИНА или ЛОЖЬ. Если логическое_значение имеет значение ЛОЖЬ, то функция НЕ возвращает значение ИСТИНА; если логическое_значение имеет значение ИСТИНА, то функция НЕ возвращает значение ЛОЖЬ.

Функция ИЛИ возвращает ИСТИНА, если хотя бы один из аргументов имеет значение ИСТИНА; возвращает ЛОЖЬ, если все аргументы имеют значение ЛОЖЬ.

ИЛИ(логическое значение1;логическое значение2;...)

Логическое_значение1, логическое_значение2, ... – это от 1 до 30 проверяемых условий, которые могут иметь значение либо ИСТИНА, либо ЛОЖЬ.

Аргументы должны быть выражены логическими значениями, такими как ИСТИНА или ЛОЖЬ, массивами или ссылками, которые содержат логические значения.

Если аргумент, который является массивом или ссылкой, содержит тексты, пустые значения или значения ошибок, то эти значения игнорируются.

Если заданный интервал не содержит логических значений, то функция ИЛИ возвращает значение ошибки #ЗНАЧ!.

Можно использовать функцию ИЛИ как формулу массива, чтобы проверить, имеются ли значения в массиве.

2.3.5. Создание диаграмм

Диаграмма – это представление данных таблицы в графическом виде, который используется для их анализа и сравнения. На диаграмме числовые данные ячеек изображаются в виде точек, линий, полос, столбиков, секторов и в другой форме. Группы элементов данных, отражающих содержимое ячеек одной строки или столбца на рабочем листе, составляют *ряд данных*.

Для создания диаграммы необходимо:

- на рабочем листе выделить данные, по которым следует построить диаграмму, включая ячейки, содержащие имена категорий или рядов, которые будут использоваться в диаграмме;
- выбрать команду **Диаграмма** из меню **Вставка** или щелкнуть на кнопке ;
- в диалоговых окнах **Мастера диаграмм** выбрать тип, формат и другие параметры диаграммы;

- для перехода к следующему шагу использовать кнопку <Далее>;

- для построения диаграммы на любом шаге можно щелкнуть кнопку **Готово**, тогда **Мастер диаграмм** самостоятельно закончит построение диаграммы;

- в последнем окне щелкнуть кнопку **Готово**.

Диаграмму можно перетянуть мышью в любое место. Для изменения размера диаграммы необходимо щелкнуть на ней мышью и перетянуть маркеры выделения. Для изменения типа и параметров построенной диаграммы следует щелкнуть на диаграмме правой кнопкой мыши и в контекстном меню выбрать подходящую команду. Для удаления диаграммы следует щелкнуть на ней мышью, чтобы появились маркеры выделения, и нажать клавишу .

Пример 2.6. Создайте базу данных, автоматизирующую принятие решения о поступлении абитуриента на основании полученных им оценок. Проходной балл – 9. Проанализируйте, сколько человек поступило, не поступило и получило оценку «2».

1. Создайте таблицу следующего вида:

Фамилия	Адрес	Дата рождения	Пол	Оценки		Сумма баллов
				Математика	Физика	

2. Сохраните таблицу под именем «Абитуриент» в своей папке.

3. Введите данные для 10 абитуриентов.

4. Отформатируйте заголовки таблицы следующим образом: разместите данные по центру столбцов, установите полужирный шрифт, голубой фон заливки. Для этого:

- объедините ячейки «Фамилия», «Адрес», «Дата рождения», «Пол», «Сумма баллов» – по вертикали; «Оценки» – по горизонтали. Для этого сначала мышкой выделите необходимые ячейки, затем воспользуйтесь функцией меню **Формат > Ячейки > Выравнивание**, установите флажок **Объединение ячеек** или воспользуйтесь пиктограммой ;

- выберите необходимый цвет, нажав на панели инструментов кнопку **Цвет заливки**;

• установите оформление ячеек – на кнопке **Внешние границы**  панели инструментов щелкните на стрелке, выберите **Все границы**.

5. Подсчитайте суммарный балл по каждому абитуриенту: установите курсор в ячейку столбца «Сумма баллов» для первого студента, на панели инструментов выберите кнопку **Автосумма** , выделите мышью ячейки, в которых надо подсчитать сумму, нажмите **<Enter>**.

6. Скопируйте введенную формулу в ячейки для остальных студентов: выделите мышью ячейку с формулой и протяните указателем мыши за правый нижний угол до необходимого места, отпустите кнопку мыши.

7. Введите в конец таблицы столбец «Анализ» и отформатируйте его, как все заголовки.

8. Введите формулы, обеспечивающие анализ следующей информации: если абитуриент получил «2» по какому-либо предмету – вывести слово «Завалил»; если не набрал проходного балла – вывести «Не прошел»; если абитуриент набрал нужное количество баллов – вывести «Поступил». Для этого в ячейке столбца «Анализ» для первого студента введите формулу `=Если(Е3<3;"Завалил";Если(Ф3<3;"Завалил";Если(Г3>20;"Поступил"; "Не прошел")))`, где Е3, Ф3, Г3 – адреса ячеек, где находятся оценки и сумма баллов первого абитуриента. Формулу скопируйте протяжкой мыши на ячейки столбца «Анализ» для всех абитуриентов.

9. Отсортируйте таблицу по фамилиям в алфавитном порядке. Для этого выделите ячейку с первой фамилией и нажмите на панели инструментов кнопку  или выполните команду меню **Данные > Сортировка**.

10. Введите в строки ниже таблицы итоговые значения: «Всего поступавших», «Сколько поступило», «Сколько не прошло по конкурсу», «Сколько завалило экзамены» и выполните вычисления. Для этого воспользуйтесь соответствующими формулами:

- для строки «Всего поступавших» – формула `СЧЕТ(знач1;знач2;...)`;

- для строки «Сколько поступило» – формула `СЧЕТЕСЛИ(диапазон; "Поступил")`;

- для строки «Сколько не прошло по конкурсу» – формула `СЧЕТЕСЛИ(диапазон; "Не прошел")`;

- для строки «Сколько завалило экзамены» – формула `СЧЕТЕСЛИ(диапазон; "Завалил")`.

Диапазон – это имена ячеек столбца «Анализ», например I3:I13.

11. По полученным в результате расчетов данным постройте на новом листе диаграмму. Для этого выделите с помощью мыши диапазон ячеек для построения диаграммы. Запустите **Мастер диаграмм**, выбрав его значок на панели инструментов, и в диалоговом режиме укажите:

- тип диаграммы – гистограмма, кнопка **Далее**;
- место расположения данных – по строкам или по столбцам, **Далее**;
- введите заголовок диаграммы – «Абитуриенты», при необходимости можно выбрать различные варианты оформления диаграммы и нажать кнопку **Далее**;
- где построить диаграмму, нажмите кнопку **Готово**.

2.4. MS Power Point

MS Power Point – одна из лучших программ подготовки и проведения презентаций. Она является компонентом MS Office и предназначена для создания презентационных материалов в виде слайдов и их вывода на бумагу, экран.

Power Point позволяет планировать, создавать и демонстрировать презентацию. Пользователю предоставляется модифицируемый набор шаблонов, редактор слайдов, средство построения схемы, поддерживается множество форматов импорта/экспорта, включая импорт из Excel, Lotus 1-2-3 и файлов ASCII. Предусмотрены функции работы с текстом, построения графиков, проектирования схемы презентации и механизм буксировки.

Презентация – это совокупность слайдов по определенной тематике, обычно оформленных в едином стиле, и соответствующих сопроводительных материалов (плана презентации, замечаний докладчика, материалов для раздачи слушателям). Power Point позволяет выводить на экран или печатать все компоненты презентации, а также сохранять презентации в виде Web-страниц для последующей публикации на Web-сервере.

Для удобства разработки презентаций в Power Point имеется ряд специальных средств: мастеров, шаблонов (образцов, заготовок), которые содержат разнообразные элементы форматирования текста, цветового оформления, графические объекты и библиотеки клипов.

Слайд представляет собой сложный объект, который может включать заголовки, текст, таблицы, графические объекты, схемы организации, звуковые фрагменты, видеоклипы и гиперссылки.

Каждый слайд сопровождается страницей заметок, на которую можно заносить поясняющий текст как во время создания, так и при его демонстрации. Этот текст используется для фиксации основных моментов презентации и сопутствующих им обстоятельств. Можно сформировать страницы заметок для любого числа слайдов и, например, занести на них сценарий, сопровождающий презентацию.

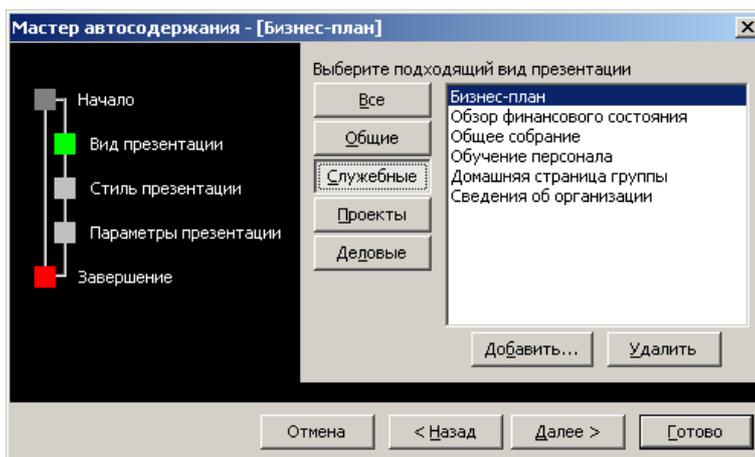
Демонстрация презентации представляет собой процесс показа слайдов в некотором порядке (не обязательно в порядке возрастания номеров слайдов), при этом смена слайдов осуществляется вручную или автоматически. Пользователь управляет процессом с помощью панели управления презентацией и навигатора слайдов. Он может также создавать интерактивную презентацию с помощью задания на слайдах гиперссылок и управляющих кнопок.

Если аудитория велика или удалена от докладчика, то демонстрация презентации, включая видео и звук, может осуществляться по Интернету, а также с использованием проектора, подключенного к компьютеру (для подключения используется мастер проекторов).

2.4.1. Создание презентации

После запуска Power Point предлагает определить, каким образом будет создаваться презентация: новая пустая презентация, с помощью **Мастера автосодержания** (рис. 2.14), с помощью выбора шаблона презентации.

Мастер автосодержания обеспечивает поэтапное создание новой презентации пользователем и предлагает выбрать в качестве исходного материала презентацию с определенным типовым содержанием и оформлением.



1. Рис 2.14. Окно Мастер автосодержания

Мастер автосодержания предлагает несколько образцов презентаций на различные темы. Среди его достоинств – панель, позволяющая в любой момент перейти к любому из окон мастера, классификация шаблонов по темам и возможность предварительно настроить параметры показа.

По указанным пользователем параметрам **Мастер автосодержания** построит предварительную схему презентации с размеченными слайдами. Разметка слайда представляет собой схему размещения меток-заполнителей (заголовка, текста, таблицы, графического объекта). Пользователю необходимо изменить содержимое этих слайдов в соответствии со своими потребностями.

Соответствующие метки-заполнители после щелчка (область для ввода текста) или двойного щелчка мышью (область для графического объекта) открываются для редактирования, и пользователь вводит в них свой текст или вставляет нужный ему объект.

Постепенно заменяя содержимое меток-заполнителей на всех слайдах сформированной последовательности, получаем готовую презентацию, которую затем можно демонстрировать. Можно создавать и свой шаблон презентаций, если не устраивают предлагаемые пакетом варианты.

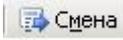
Шаблон презентации дает возможность выбрать вид оформления слайда и содержит набор цветовых схем слайдов.

Пустая презентация формирует презентацию «с нуля» и предназначена для работы профессионального пользователя. После щелчка на значке **Новая презентация**, расположенном с правой стороны рабочего окна, будет предложено выбрать разметку для слайда. В новой презентации используются цветовая схема, стиль заголовка и стили текста презентации, принимаемой по умолчанию. Нужно ввести на титульном слайде заголовки презентации и прочие сведения и нажать на кнопку **Создать слайд** для создания следующего кадра.

Презентацию можно просматривать в различных режимах. Переход от одного режима в другой осуществляется с помощью команд меню **Вид** или ряда кнопок (слева внизу окна презентации). Режимы работы Power Point следующие:

-  **обычный** – послайдный режим редактирования (команда **Вид > Слайды**). В нем создают и редактируют отдельные элементы слайдов. С одного слайда на другой переключаются при помощи вертикальной полосы прокрутки, а также клавиш **<Page Up>** и **<Page Down>**;

-  **структуры** – показ общей структуры, уменьшенного изображения слайда – миниатюры и заметок к нему (команда **Вид > Структура**);

-  **сортировщик слайдов** – просмотр всей последовательности слайдов и изменение порядка их расположения в презентации. В режиме сортировщика слайдов маленькие изображения слайдов выстраиваются на экране одно за другим в том порядке, в каком их будут показывать во время выступления (команда **Вид > Сортировщик слайдов**). В этом режиме можно создавать и копировать слайды по одному и группами, а также менять их последовательность. Под стандартной панелью инструментов появляется панель **Сортировщик слайдов**, в которой кнопка  **Смена** позволяет выбрать способ перехода от одного слайда к другому. То же самое можно сделать, выбрав в меню **Показ слайдов** или из контекстного меню слайда команду **Смена слайдов**. Справа от слайда появится диалоговое окно выбора функций смены;

-  **страницы заметок** – для заполнения или просмотра страницы заметок для каждого слайда. Этот режим напоминает режим просмотра документов Word перед печатью (команда **Вид > Страницы заметок**). Страница делится надвое. В верхней части изображен слайд, а в нижней – поле для заметок докладчика. Заметки вносят при создании презентации. Затем их можно вывести на печать;

-  **показ слайдов** – для демонстрации презентации во время выступления (команда **Вид > Показ слайдов**). Обычно вид полноэкранный, а слайды меняются либо автоматически, либо по щелчку мыши. Но эти параметры можно и изменить. Для этого следует выбрать команду **Показ слайдов > Настройка презентации** и внести нужные коррективы в появившемся диалоговом окне.

Power Point позволяет создавать презентации в определенном стиле. Существует три способа управления стилем презентации: шаблоны оформления, цветовые схемы и образцы.

Шаблон оформления содержит цветовое оформление, образцы слайдов и заголовков с настраиваемыми форматами и стилизованные шрифты, определяющие вид презентации. Если не устраивают предлагаемые пакетом варианты, можно создавать и свой шаблон презентаций.

Цветовая схема состоит из восьми цветов, используемых в качестве основных для текста, фона, заливки, акцентов и т. п.

Каждый цвет схемы используется автоматически для различных элементов слайда. При применении шаблонов оформления можно выбрать цветовую схему из набора, хранящегося в каждом шаблоне. Это облегчает подбор цветовых схем, гарантируя цветовую совместимость с другими слайдами презентации.

В презентации можно использовать образец слайдов, который задает расположение и некоторые параметры текста (шрифт, его размер и цвет, цвет фона, заливку и стиль маркера списка). Он содержит пустые рамки для текста и колонтитулов, включающих дату, время и номер слайда, и управляет оформлением всех слайдов. Любое изменение образца отражается на каждом слайде презентации, например при добавлении графического объекта в образец он появится на всех слайдах презентации. С другой стороны, если созданы уникальные слайды (например, использованы цвета или заливка, отличные от цветовой схемы образца слайдов), они сохраняют свое оформление, несмотря на изменения образца слайдов.

2.4.2. Работа со слайдами и объектами

Для того чтобы создать новый слайд, необходимо выполнить команду **Вставка** > **Создать слайд** или воспользоваться соответствующей кнопкой на панели инструментов. В диалоговом окне надо выбрать требуемую структуру слайда (**Разметка слайда**) и заполнить появившийся на экране шаблон необходимой информацией (рис. 2.15). Также в Power Point слайды можно создавать следующими способами:

1. При помощи команды **Вставка** > **Дублировать слайд**. Выделите уже имеющийся слайд презентации, выберите эту команду – и получите слайд, расположенный сразу после его прототипа.

2. При помощи команды **Вставка** > **Слайды из файлов**. Эта команда позволяет копировать слайды из одной презентации в другую. Если воспользоваться ею, откроется диалоговое окно **Поиск слайдов**. Щелкните на кнопке **Обзор** и выберите презентацию, в которой содержится нужный вам слайд. Чтобы его найти, щелкните на кнопке **Показать**. Затем отметьте нужные слайды и щелкните на кнопке **Вставить**.

3. При помощи команды **Вставка** > **Слайд из структуры**. Под структурой в данном случае подразумевается любой текстовый файл, абзацы которого Power Point преобразует в слайды презентации.

Чтобы в презентацию, настроенную на альбомную ориентацию (книжную), включить серию слайдов книжной ориентации (альбомной), составьте из этих слайдов новую презентацию, создайте с ней

связь через гиперссылку (см. ниже), затем откройте эту презентацию в ходе показа слайдов.

Чтобы создать связь с отдельным слайдом, имеющим иную ориентацию, откройте презентацию, в которую он входит, и переведите этот слайд в новую презентацию.

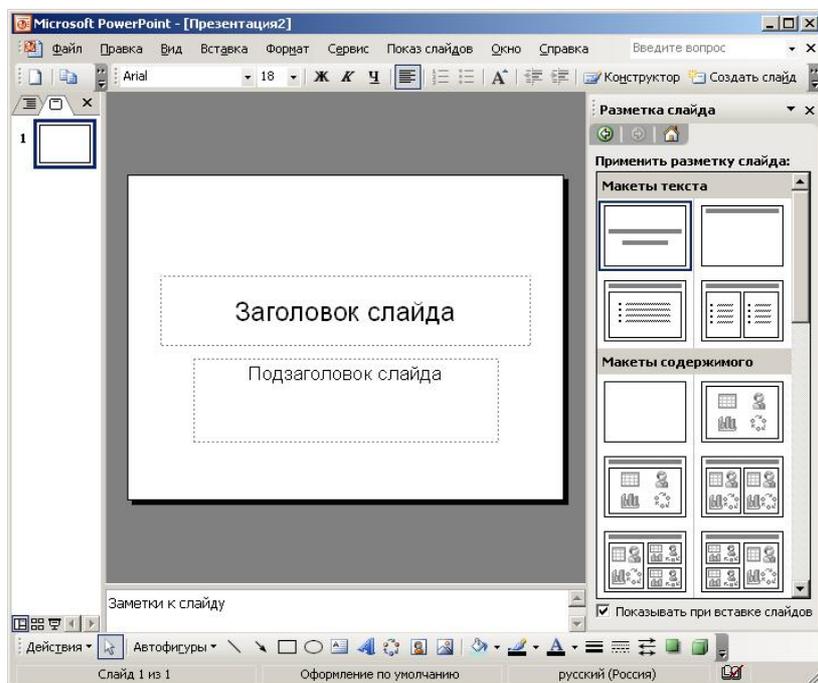


Рис. 2.15. Окно создания нового слайда

Слайды Power Point состоят из объектов (текст, график, рисунок, схема организации, видеоклип; при этом электронная таблица Excel, таблица Word и др. становятся объектами при добавлении к слайду). К объекту применимы все известные средства форматирования. При работе с объектом его необходимо предварительно выделить, щелкнув на нем мышью, а затем можно изменять его размеры, расположение, цвет, границы и т. д.

Разметка слайда определяет расположение, количество и виды объектов на слайде. Положение и тип объекта указаны в виде соответствующей *метки-заполнителя* (рамки). Например, существует разметка, содержащая метки-заполнители для заголовка, текста и диаграммы, а также разметка с метками-заполнителями для текста и графики.

Метки-заполнители заголовка и текста выполнены в соответствии с форматами, заданными в образце слайда выбранной презентации. При необходимости пользователь может перемещать метки, изменять размеры и переформатировать, если его не устраивает образец слайда.

Для художественного оформления презентации применяется дизайн, цветовая схема, фон и разметка слайдов. Чтобы настроить эти параметры, используют команды **Разметка слайда**, **Фон**, **Оформление слайда** из меню **Формат** или контекстного меню слайда в режиме послайдного редактирования. Дизайн распространяется на всю презентацию. Разметка же, цветовая схема и фон могут изменяться от слайда к слайду. Тем не менее следует стремиться к единству формы, а также к тому, чтобы она соответствовала содержанию.

По команде **Фон** открывается диалоговое окно, где определяется не только цвет, но и узор, градиентная заливка, текстура или изображение, которое послужит фоном для текущего слайда (кнопка **Применить** или для всей презентации кнопка **Применить ко всем**).

Для того чтобы изменить оформление группы слайдов, нужно перейти в режим сортировки, выделить эту группу и воспользоваться командами меню **Формат**.

Для удаления слайда выделите его и выберите команду **Правка** > **Удалить слайд**. Чтобы удалить несколько слайдов одновременно, переключитесь в режим сортировщика слайдов или режим структуры, нажмите клавишу <Shift> и, удерживая ее, щелкните поочередно на всех слайдах, затем выберите команду **Удалить слайд**.

Чтобы вставить текст вне разметки или фигуры (например, снабдить рисунки надписями или выносками), можно воспользоваться инструментом **Надпись** , расположенным на панели инструментов **Рисование**, или командой **Вставка** > **Надпись**. В месте, которое вы затем укажете мышью, появится маленькая рамка, в какие обычно заключаются надписи на слайдах Power Point. Остается только ввести и отформатировать требуемый текст.

Для вставки рисунка используется команда **Вставка** > **Рисунок**. В слайд можно включить диаграмму (**Вставка** > **Диаграмма**) или другие объекты (**Вставка** > **Объект** > **Тип объекта**).

При размещении на слайде нескольких объектов Power Point автоматически формирует порядок их расположения, помещая каждый на своем слое по мере добавления в слайд. Порядок расположения объектов определяет способ их перекрывания: верхний объект закрывает части объектов, лежащих под ним. Если объект «потерян», необходимо нажимать клавиши <Tab> или <Shift> + <Tab> для циклического перемещения по слоям до тех пор, пока нужный объект не выделится.

Отдельные объекты или группы объектов можно перемещать по слоям. Например, можно переместить объект на уровень выше или ниже, на верхний или нижний слой с помощью с помощью команды

Порядок из контекстного меню слайда. Наложение объектов часто применяется для создания различных эффектов.

Для текста и любого объекта можно задать способ появления на экране, например вылет слева, появление текста по буквам, словам, абзацам. Можно также задать поведение объектов при добавлении нового элемента – затемнение или изменение цвета. Для этих целей применяются команды **Показ слайдов > Эффекты** или **Показ слайдов > Настройка анимации**.

2.4.3. Демонстрация презентации

Управление сменой слайдов и выбором специальных эффектов при их появлении во время демонстрации (например, жалюзи горизонтальное, шашки, которые помогают выделить те или иные моменты презентации) осуществляется с помощью команды **Показ слайдов > Смена слайдов** (рис. 2.16).

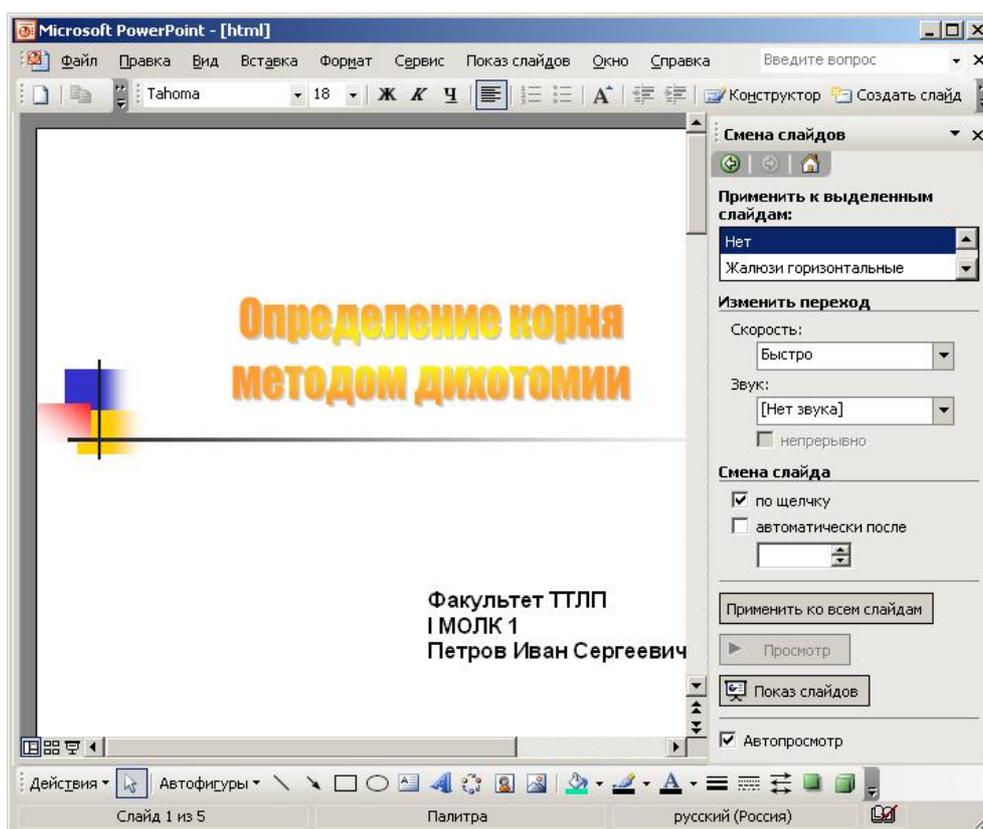


Рис. 2.16. Окно смены слайдов

В появившемся справа диалоговом окне можно определить способ перехода к следующему слайду: автоматически через заданное

время или по щелчку мыши, а также эффект перехода слайда, например: наплыв вниз, растворить, жалюзи и т. д.

Для запуска презентации надо встать на первый слайд и выбрать в меню **Показ слайдов** > **Начать показ**.

Для смены порядка показа слайдов надо перейти в режим сортировки слайдов, выбрав в меню **Вид команду Сортировщик слайдов**, и путем перетаскивания слайдов определить их порядок показа.

Каждый слайд презентации присутствует на экране определенное время, длительностью которого можно управлять, вручную переходя к другому слайду, или установить предварительно, задав смену слайда через определенный промежуток времени (команда **Показ слайдов** > **Настройка времени**). Для получения информации о временных интервалах следует открыть презентацию в режиме сортировщика слайдов, в котором под каждым слайдом указана продолжительность его присутствия на экране.

Существует три способа показа слайдов:

1. Управляемый докладчиком обычный способ проведения показа, когда слайды отображаются в полноэкранном режиме. Презентацию можно проводить вручную или в автоматическом режиме, останавливать ее для записи замечаний или действий и даже записывать во время презентации речевое сопровождение. Этот режим удобен для показа презентации на большом экране, проведения собрания по сети или трансляции презентации.

2. Управляемая пользователем презентация, которая отображается в небольшом окне (при просмотре одним пользователем по сети или через Интернет), имеются команды смены слайдов, редактирования, копирования и печати слайдов. В этом режиме переход к другому слайду осуществляется с помощью полосы прокрутки или клавиш <**Page Up**> и <**Page Down**>. Для удобства работы можно вывести панель инструментов Web.

3. Автоматический способ, когда презентация проводится полностью автоматически на весь экран. Этот режим можно использовать на выставочном стенде или собрании. При этом можно запретить использование большинства команд меню и включить режим циклического показа.

Чтобы выбрать способ, необходимо установить в соответствующее положение переключатель в диалоговом окне **Настройка презентации** (пункт меню **Показ слайдов**).

Для управления ходом показа слайдов можно использовать и управляющие кнопки. Если на каждый слайд требуется разместить одни и те же кнопки, рекомендуется расположить их на образце слайдов.

Для более гибкого управления можно создать ответвления к другим слайдам или к другим презентациям. При подготовке ответвления обычно используют управляющие кнопки. Чтобы поместить кнопку на слайд, необходимо:

- Выполнить команду **Показ слайдов** > **Управляющие кнопки** и указать требуемую кнопку (например: **Домой**, **Назад**, **Далее**, **В начало** или **Возврат**).

- В окне **Настройка действия** принять гиперссылку, предложенную в списке **Перейти по гиперссылке**, или задать другую. В качестве гиперссылки можно определить как слайд в текущей презентации, так и переход в другую презентацию, и даже URL-адрес или запуск любого приложения, причем выполнение выбранного действия можно задать по щелчку на управляющей кнопке или при наведении на нее указателя мыши. Если осуществлено ответвление к презентации, то после ее демонстрации происходит автоматический возврат к исходному слайду. При переходе к слайду внутри презентации для возвращения к исходному необходимо использовать **Навигатор слайдов** или создать соответствующую кнопку. Power Point позволяет приостановить или возобновить демонстрацию слайдов. Для этого во время демонстрации слайдов необходимо щелкнуть правой кнопкой мыши, выбрать **Экран** > **Пауза** > **Завершить показ слайдов** или нажать клавишу <Esc>.

Чтобы демонстрация была сохранена на диске в виде показа слайдов, необходимо при сохранении выполнить команду **Файл** > **Сохранить как** и в списке **Тип файла** выбрать **Демонстрация Power Point**.

Файл, сохраненный в виде демонстрации, имеет расширение pps. При его открытии с рабочего стола автоматически загружается показ слайдов, а по его завершении Power Point закрывается и восстанавливается рабочий стол. Если показ слайдов запускается из Power Point, то по его завершении презентация остается открытой и доступной для редактирования.

Пример 2.7. Создание презентации с использованием **Мастера автосодержания**.

1. Загрузите MS Power Point: **Пуск** > **Программы** > **Microsoft Power Point** или двойной щелчок мышкой по ярлыку MS Power Point на рабочем столе MS Windows.

2. Создайте презентацию с использованием **Мастера автосодержания**.

3. В стартовом диалоговом окне в разделе **Создать презентацию** выберите **Мастер автосодержания** > **ОК** > **Далее**.

4. В разделе **Выберите подходящий вид презентации** щелчком мыши выделите **Предлагаем стратегию** > **Далее**.

5. В разделе **Предполагаемый способ вывода презентации** установите переключатель **Презентация на экране** > **Далее**.

6. В поле **Заголовок презентации** введите «Учебная презентация», в поле **Нижний колонтитул** – свою фамилию и инициалы, установите флажки **Дата последнего изменения**, **Номер слайда**, щелкните **Далее** > **Готово**.

7. Воспользовавшись вертикальной полосой прокрутки, совершите следующие перемещения по слайдам презентации:

- перейдите к пятому слайду;
- вернитесь к последнему слайду;
- перейдите к первому слайду.

При этом слева от заголовка слайда отображается его номер.

8. Для увеличения размера области структуры переключитесь в режим структуры: **Вид** > **Структура** или соответствующий значок в левом нижнем углу окна MS Power Point.

9. Для увеличения размера области слайда переключитесь в режим слайдов: **Вид** > **Слайды** или соответствующий значок в левом нижнем углу окна MS Power Point.

10. Для получения общего представления обо всей презентации в целом переключитесь в режим сортировщика слайдов: **Вид** > **Сортировщик слайдов** или соответствующий значок в левом нижнем углу окна MS Power Point (под правым нижним углом слайда – его номер, текущий слайд выделен прямоугольной рамкой).

11. Переключитесь в режим страниц заметок: **Вид** > **Страницы заметок** (на листе – уменьшенная копия слайда и поле для ввода текста заметок к этому слайду).

12. Запустите экранную презентацию (показ слайдов): **Вид** > **Показ слайдов**, или **Показ слайдов** > **Начать показ**, или соответствующий значок в левом нижнем углу окна MS Power Point (в последнем случае показ начинается с текущего слайда). Переход к следующему/предыдущему слайду – <**Page Down**>/<**Page Up**> (для смены слайдов в прямом направлении – щелчок мыши в любом месте экрана).

13. Завершите показ слайдов: <**Esc**> или перейдите к последнему слайду с помощью клавиши <**Page Down**>.

14. Переключитесь в обычный режим отображения презентации: **Вид** > **Слайды** или соответствующий значок в левом нижнем углу окна MS Power Point.

15. Сохраните презентацию в своей рабочей папке: **Файл** > **Сохранить**, введите имя файла в соответствующее поле, в поле **Тип файла** оставьте **Презентация (*.ppt)** (MS Power Point автоматически добавит к имени файла расширение .ppt), кнопка **Сохранить**.

16. Завершите работу с презентацией: **Файл** > **Заккрыть**.

17. Завершите работу с MS Power Point: **Файл** > **Выход**.

Пример 2.8. Работа с объектами, размещаемыми на слайде.

1. Создайте новую презентацию. Переключитесь в режим отображения слайдов: **Вид** > **Образец** > **Образец слайдов**.

2. Введите текст заголовка – «Интернет-магазин».

3. Установите с помощью меню **Формат** > **Шрифт** для образца заголовка слайда шрифт Times New Roman, размер шрифта – 60 пт, цвет шрифта – красный.

4. Установите для заголовка желтую тень с помощью кнопки **Тень** на панели рисования.

5. Переключитесь в обычный режим просмотра презентации: **Вид** > **Слайды** или соответствующая кнопка в левом нижнем углу окна.

6. Установить фон слайда – белый мрамор с помощью меню **Формат** > **Фон** (или контекстного меню слайда): в диалоговом окне в раскрывающемся списке **Заливка фона** > **Способы заливки** > **Текстура** выберите **Белый мрамор**, нажмите **ОК** > **Применить**.

7. Установите автоматическую анимацию для объектов слайда Интернет-магазин: **Показ слайдов** > **Настройка анимации**; на вкладке **Время** в списке **Объекты для анимации** установите опцию в требуемом поле, в списке **Анимация** укажите **автоматически через 1 с**.

8. Создайте второй слайд: **Вставка** > **Создать слайд**. Выберите разметку слайда «заголовок и текст».

9. Введите текст заголовка «Пакеты Microsoft 2007». Измените цвет заголовка второго слайда презентации.

10. Введите в разметку – Word, Excel, Power Point.

11. Установить для заголовка размер шрифта – 40 пт.

12. Примените для заголовка голубую тень.

13. Внизу слайда добавьте объект **Надпись** и ведите в нее текст «Наши цены».

14. Создайте новый слайд: **Вставка** > **Создать слайд**. Выберите автомакет, содержащий заголовок, текст и графику.

15. Введите заголовок слайда – «Word». Добавьте на слайд рисунок – копию экрана окна пакета Word. Для получения копии экрана воспользуйтесь клавишами <Alt> + <Print Screen>. Введите текст, подберите фон и цветовое оформление слайда.

16. Создайте слайды для описания пакетов Excel и Power Point аналогичным образом.

17. Создайте новый слайд, содержащий текст и таблицу. Введите заголовок и заполните таблицу информацией о стоимости пакетов Word, Excel, Power Point, выполните цветовое оформление.

18. На каждый слайд добавьте управляющую кнопку перехода на слайд «Пакеты Microsoft 2007»: в меню **Показ слайдов** > **Управляющие кнопки** выберите вид кнопки и разместите ее на слайде; в появившемся окне **Настройка действия** установите флажок **Перейти по гиперссылке**, в предложенном списке выберите **Слайд** и укажите слайд «Пакеты Microsoft 2007».

19. Перейдите на слайд «Пакеты Microsoft 2007». Названия пакетов Word, Excel, Power Point сделайте гиперссылками. Для этого соответствующее слово, например «Word», выделите мышкой, воспользуйтесь командой меню **Вставка** > **Гиперссылка**, в диалоговом окне выберите **Связать с местом в документе** и укажите нужный слайд.

20. Установите анимацию для объектов созданных слайдов: **Показ слайдов** > **Настройка анимации**.

21. Запустите экранную презентацию (показ слайдов): **Вид** > **Показ слайдов** или **Показ слайдов** > **Начать показ**.

22. Сохраните презентацию в своей папке.

3. ОСНОВЫ ПРОГРАММИРОВАНИЯ

Основой компьютерной грамотности любого специалиста является создание программы для ПК. Компьютерные программы создают программисты – люди, обученные процессу их составления (программированию). Управлять ПК можно по заданному алгоритму. Алгоритм – это точно определенное описание способа решения задачи в виде конечной (по времени) последовательности действий. Такое описание еще называется формальным. Для представления алгоритма в виде, понятном ПК, служат языки программирования. Сначала всегда разрабатывается алгоритм действия, а затем он записывается на одном из таких языков. В итоге получается текст программы – полное, законченное и детальное описание алгоритма на языке программирования. Затем этот текст программы специальными служебными приложениями, которые называются трансляторами, либо переводится в машинный код, либо исполняется.

Самому написать программу в машинном коде весьма сложно, причем эта сложность резко возрастает с увеличением размера программы и трудоемкости решения нужной задачи соответствующей отрасли. Условно можно считать, что машинный код приемлем, если размер программы не превышает нескольких десятков байтов (байт – наименьшая единица измерения информации) и нет потребности в операциях ручного ввода-вывода данных (данные для ПК – это зарегистрированные сигналы).

Поэтому сегодня практически все программы создаются с помощью языков программирования. Теоретически программу можно написать и средствами обычного (естественного) языка – это называется программированием на метаязыке (подобный подход обычно используется на этапе составления алгоритма), но автоматически перевести такую программу в машинный код пока невозможно из-за высокой неоднозначности естественного языка.

Языки программирования – искусственные языки. От естественных они отличаются ограниченным числом «слов», значение которых понятно транслятору, и очень строгими правилами записи команд (операторов). Совокупность подобных требований образует синтаксис языка программирования, а смысл каждой команды и других конструкций языка – его семантику. Нарушение формы записи программы приводит к тому, что транслятор не может понять назначение оператора и выдает сообщение о синтаксической ошибке,

а правильно написанное, но не отвечающее алгоритму использование команд языка приводит к семантическим ошибкам (называемым еще логическими ошибками или ошибками времени выполнения).

3.1. Система программирования TurboPascal

Язык программирования Pascal до настоящего времени, наряду с Бейсиком и СИ, входит в состав наиболее известных. Будучи задуманным профессором Цюрихского технологического института Никлаусом Виртом как средство обучения систематическому программированию, он изучается в курсах информатики большинства школ и высших учебных заведений. Одной из популярных систем программирования, поддерживающих этот язык, является TurboPascal различных версий.

Разработанная в 1984 г. корпорацией Borland International система программирования TurboPascal, объединяющая транслятор, экранный редактор, редактор связей и средства отладки, оказалась настолько удобной, что сразу приобрела широкую популярность и имеет практическое применение как многооконная интегрированная система с пунктами головного меню. К настоящему времени создано несколько версий этой системы.

Интегрированная среда (Турбо-среда) позволяет набирать тексты программ с использованием встроенного редактора текстов, компилировать их, выполнять, проводить отладку программ.

Интегрированность среды проявляется не только в единой идеологии построения компонентов, но и в их связи друг с другом. Так, при возникновении ошибки трансляции система автоматически переходит в режим экранного редактирования и ставит курсор в точку возникновения ошибки. Аналогичные действия выполняются и отладчиком при возникновении ошибки во время выполнения программы.

Запуск системы TurboPascal осуществляется командой **turbo.exe**, после выполнения, которой на экране появляется главное меню системы. Для выхода из системы следует нажать клавиши **<Alt> + <X>** или использовать команду **Quit** из пункта головного меню **File**.

После входа в систему TurboPascal на экране дисплея появится основной информационный кадр. Он состоит из трех полей:

- 1) главное меню;
- 2) окно редактора;
- 3) строка состояния.

Главное меню предназначено для выбора режима работы системы: ввод программы, компиляция, выполнение, отладка и т. д.

Окно редактора содержит текст программы. Строка в окне редактора указывает местоположение курсора (**Line** – номер строки, **Col** – номер колонки текста), а также включенные режимы редактирования, например **Insert** – включен режим вставки, **Indent** – установлен режим автоматического отступа; **C:NONAME.PAS** – текущая директория, имя и тип расширения файла, находящегося в активном окне (выделено двойной линией). В строке состояния расшифровывается назначение функциональных клавиш в текущем режиме работы системы.

Для выбора одной из команд, наименования которых сведены в главное меню, необходимо перейти в режим работы с главным меню. Переход в режим меню осуществляется нажатием клавиши <**F10**>. Повторное нажатие этой клавиши приводит к переходу в режим редактирования. Дальнейшие нажатия вызывают поочередную смену режимов.

В главном меню содержатся следующие команды: **File** (файл), **Edit** (редактор), **Run** (выполнение), **Compile** (компилирование), **Options** (опции), **Debug** (отладка), **Break/watch** (прерывание/просмотр). Все они, кроме **Edit**, имеют собственные подменю, некоторые несколько вложенных подменю (ниспадающие).

Запуск соответствующих команд (режимов) осуществляется либо посредством выделенного прямоугольника – с использованием клавиш перемещения курсора подведите выделенный прямоугольник к нужной команде и нажмите <**Enter**>, либо нажатием соответствующих функциональных клавиш, в дальнейшем называемых синонимами. Синоним, если он существует, записан в соответствующем подменю рядом с именем команды пункта меню.

Есть еще один способ запуска команд. Если вы находитесь в нужном вам меню, то не обязательно перемещать выделенный прямоугольник к требуемой команде, достаточно нажать клавишу с заглавной буквой этой команды (например, **R** – команда **Run** в главном меню).

Таким образом, выбирайте один из следующих способов запуска команд:

1. Нажимая последовательно клавиши <**F10**> (вход в главное меню), перемещения курсора (выбор команды меню и вход в подменю), вы находите нужное подменю, и далее у вас два альтернативных варианта действий:

а) подвести курсор к требуемой команде и активизировать клавишу **<Enter>**;

б) нажать клавишу, соответствующую заглавной букве команды.

2. Вы набираете синоним команды, не утруждая себя поиском соответствующей команды из пунктов головного меню. Но в этом случае надо быть уверенным, что вы правильно запомнили синоним команды. Например: нажатием функциональной клавиши-синонима **<F3>** будет произведена активизация команды **Load** из пункта головного меню **File**.

Управление работой системы может выполняться с помощью стрелок, т. е. управляющих клавиш, и клавиши **<Enter>**. Для выбора команды из пунктов главного меню с помощью стрелок надо, нажимая клавиши управления курсором, переместить выделение на имя этого пункта и нажать **<Enter>**.

Для того чтобы откомпилировать и выполнить введенную программу, необходимо перейти в режим **Run** и, выбрав соответствующую команду **Run**, нажать клавишу **<Enter>** или, находясь в редакторе, нажать клавишу **<Ctrl>** и, не отпуская ее, **<F9>**. Далее такую последовательность действий будем обозначать **<Ctrl> + <F9>**.

При наличии синтаксических ошибок на экран выводится диагностическое сообщение. Курсор при этом будет находиться в строке, содержащей ошибку. Для продолжения отладки, после исправления, следует снова ввести **<Ctrl> + <F9>**. Подсказку о работе в текущем режиме можно получить, нажав клавишу **<F1>**. Для получения информации об операторах, процедурах, функциях и остальных элементах программы надо переместить курсор к интересующему элементу и нажать клавиши **<Ctrl> + <F1>**. Для перелистывания выводимой информации используются клавиши **<Page Up>** и **<Page Down>**. Выход из режима и снятие экранов с текстами происходят при нажатии клавиши **<Esc>**.

После завершения выполнения программы на экране монитора появляется окно редактора с текстом программы. Для просмотра результатов надо нажать комбинацию клавиш **<Alt> + <F5>**. Повторное нажатие этих клавиш приводит к переходу в окно редактора системы TurboPascal.

При редактировании программы можно перемещать курсор по экрану, удалять символы и строки, вносить добавления, выделять части текста в отдельные блоки и выполнять с ними такие же операции, как и с отдельными символами языка программирования Паскаль (см. таблицу).

Команды	Клавиши	Выполняемые действия
Отметка начала блока	Ctrl + K + B	Маркер начала блока помещается в позицию курсора
Отметка конца блока	Ctrl + K + K	Маркер конца блока помещается в позицию курсора
Копирование блока	Ctrl + K + C	Копия выделенного блока помещается перед курсором
Перемещение блока	Ctrl + K + V	Выделенный блок перемещается в позицию перед курсором
Удаление блока	Ctrl + K + Y	Выделенный блок удаляется
Смена разметки	Ctrl + K + H	Отмена/установка разметки выделенного блока
Печать блока	Ctrl + K + P	Печать выделенного блока
Запись блока в файл	Ctrl + K + W	Выделенный блок записывается в файл по указанному маршруту

3.2. Начальные сведения для подготовки и выполнения программ

Система TurboPascal обладает развитыми средствами для подготовки, редактирования, отладки и выполнения программ. Перед началом надо убедиться, что у вас имеется на жестком диске система программирования TurboPascal. Затем следует перейти в тот директорию, в котором находятся файлы системы TurboPascal, запустить систему командой **turbo.exe**. На экране появится главное меню, а курсор установится в первой позиции окна встроенного текстового редактора (**Edit**). В этом режиме можно набирать текст программы или корректировать уже набранный текст. Вход в текстовый редактор из главного меню осуществляется командой пункта меню **Edit**, выход из текстового редактора в главное меню – клавишей **<F10>**.

Находясь во встроенном текстовом редакторе, можно передвигать курсор вверх и вниз, вправо и влево. Для этого используйте клавиши перемещения курсора. Каждую строку программы заканчивайте нажатием клавиши ввода **<Enter>**. При этом курсор сам перейдет на начало следующей строки.

Если вы случайно пропустили некоторую строку программы, то подведите курсор к концу той строки, после которой должна быть вставка, и нажмите клавишу **<Enter>**. Все строки, расположенные ниже курсора, переместятся вниз, оставив пустую строку. Курсор автоматически переместится в первую позицию образовавшейся пустой строки.

Если вам необходимо удалить некоторую строку, то подведите к ней курсор и нажмите одновременно комбинацию клавиш **<Ctrl> + <Y>**.

Вставить символ в строку тоже очень просто. Подведите курсор к тому символу, перед которым необходимо осуществить вставку, убедитесь в том, что режим вставки включен (включается и выключается клавишей **<Ins>**), и наберите на клавиатуре вставляемый символ (или последовательность символов). Если режим **<Ins>** не включен, то вы запишите вставляемые символы вместо уже имеющихся в программной строке. Индикатором того, что режим вставки включен, является наличие слова **Insert** во второй строке экрана Турбо-среды.

Если вам надо удалить символ в строке, то подведите к нему курсор и нажмите на клавишу ****. Символ будет удален, а все оставшиеся после курсора символы сдвинутся влево.

После того как вы набрали текст программы и убедились визуально, что в нем нет ошибок, надо отправить его на компиляцию и выполнение. Для этих целей служит команда главного меню **Run**. У этой команды есть синоним – **<Ctrl> + <F9>**.

Команда **Run** выполняет два действия:

- 1) компилирует программу, находящуюся в редакторе и, если в ней не обнаружено синтаксических ошибок, посылает ее на выполнение;
- 2) если программа уже откомпилирована (для выполнения только компиляции используется команда **Compile**), то команда **Run** посылает программу на выполнение, не повторяя этап компиляции.

Если в программе используются операторы вывода на экран (**write** или **writeln**), то вы сможете просмотреть результаты счета, так как они выводятся в окно **Edit**. Для просмотра результатов в окне **Edit** нажмите одновременно клавиши **<Alt> + <F5>**. Чтобы вернуться обратно в окно редактирования, нажмите любую клавишу.

Скорее всего, вам не удастся сразу написать правильную программу. За исключением очень простых программ это не получается даже у опытных программистов. В поиске ошибок вам поможет система TurboPascal.

Различают три типа ошибок: синтаксические ошибки (ошибки компиляции), ошибки выполнения и ошибки в алгоритме программы.

Синтаксические ошибки возникают при нарушении правил записи языка программирования Паскаль – их обнаруживает компилятор. При этом курсор будет указывать на тот оператор в программе, где возможна ошибка, а в первой строке окна редактирования (выделенной другим цветом) появится сообщение об ошибке. Сообщение об ошибке содержит ее номер и поясняющий текст. Нажмите клавишу <F1> (**Help**), и система укажет поясняющим текстом причину возникновения ошибки.

Ошибки выполнения – это такие ошибки, которые не нарушают синтаксис языка Паскаль, но приводят к ошибочным операциям в процессе выполнения программ (например, попытка деления на ноль). В этом случае система также выдаст сообщение об ошибке с указанием оператора, но уже на этапе выполнения программы.

Ошибки в алгоритме программы – это такие ошибки, которые при верных исходных данных и безошибочной работе программы в системе TurboPascal приводят к неверным результатам. Такие ошибки должен обнаруживать сам программист. Система TurboPascal помогает ему в этом, предоставляя услуги встроенного отладчика (пошаговое выполнение и др.).

Система TurboPascal представляет еще одну важную услугу пользователю. Если у вас возникли затруднения, нажмите клавишу <F1>, и на экране дисплея появится подсказка из Help-файла (**turbo.hlp**). Для выхода из подсказки требуется нажать клавишу <Esc>.

3.3. Структура программы Паскаль

Программа реализует алгоритмы решения задачи. Она объединяет последовательность действий, выполняемых над определенными данными с помощью определенных операций для реализации определенной цели. Основные характеристики программы – точность полученного результата, время выполнения и объем требуемой памяти. Хорошо написанная программа должна соответствовать принципам структурного программирования. Целью структурного программирования является попытка упростить написание ясных по структуре программ и снизить количество логических ошибок.

Программа на языке Паскаль состоит из строк. Максимальная длина строк не должна превышать 127 символов. Если в строке более 127 символов, то все лишние символы компилятором игнорируются. Набор текста программы осуществляется с помощью встроенного

редактора текстов системы программирования TurboPascal или любого другого редактора. В первом случае программа может после выхода из редактора компилироваться и выполняться; во втором случае программу необходимо записать в файл на магнитный носитель и вызывать для компиляции и выполнения из среды системы TurboPascal.

Программист, набирая текст программы, имеет право произвольно располагать строки на экране. Строка может начинаться с любой колонки, т. е. величина отступа от левой границы экрана для каждой строки устанавливается самим программистом с целью получения наиболее читабельного, по его мнению, текста программы. Количество операторов в строке произвольно.

Существуют много схем написания программ на Паскале, все они отличаются количеством отступов слева в каждой строке и различным использованием прописных букв. Единого мнения по этому вопросу нет. Схема, которая использовалась в настоящем пособии при записи примеров программ, имеет следующие черты: зарезервированные слова Program, Procedure, Function в любом случае начинаются с прописной буквы. Имена процедур, функций начинаются с прописных букв. Операторы, имена констант, переменных записываются только строчными буквами; логически подчиненные структуры записываются на одну строку ниже и на одну позицию правее по отношению к более старшим. Такая схема, возможно, не является лучшей, но вполне оправдала себя в практической работе со студентами.

Размер программы имеет предел. Редактор текстов и компилятор позволяют обрабатывать программы объемом до 64 килобайт. Если программа требует большего количества памяти, следует воспользоваться средствами включения файлов или оверлейным методом построения программ.

Синтаксически программа состоит из необязательного заголовка и блока. Блок может содержать в себе другие блоки. Блок состоит из двух частей: описательной и исполнительной. Первая часть может отсутствовать, без второй блок не имеет смысла. Блок, который не входит ни в какой другой, называется глобальным. Если в глобальном блоке находятся другие блоки, они называются локальными. Глобальный блок – это основная программа, он должен присутствовать в любом случае. Локальные блоки – это процедуры и функции, их присутствие необязательно. Объекты программы (типы, переменные, константы и т. д.) соответственно называются глобаль-

ными и локальными. Область действия объектов – блок, где они описаны, и все вложенные в него блоки. Существуют разные мнения об использовании глобальных и локальных объектов. Блочная структура обеспечивает структуризацию программ на уровне исходных текстов. В идеальном случае программа на языке Паскаль состоит из процедур и функций, которые вызываются для выполнения из раздела операторов основной программы. В начале программы находится заголовок, состоящий в общем случае из зарезервированного слова Program и имени программы. Заголовок программы несет чисто смысловую нагрузку и может отсутствовать, однако рекомендуется всегда его записывать для быстрого распознавания нужной программы среди листингов других программ:

```
Program Zarplata;  
Program SpisokGrupp;
```

После заголовка следует программный блок, состоящий в общем случае из раздела модулей (Uses) и разделов: описание меток (Label), описание констант (Const), определение типов данных (Type), описание переменных (Var), описание процедур и функций (Procedure, Function), операторов (Begin...End.).

Любой раздел, кроме раздела операторов, может отсутствовать. Например, самая короткая программа, написанная на языке Паскаль:

```
Begin  
  writeln('Самая короткая программа')  
End.
```

Разделы описаний могут встречаться в программе любое количество раз и следовать в любом порядке. Главное, чтобы все описания объектов программы были сделаны до того, как они будут использованы. Исключением является раздел модулей (Uses), который обязательно должен быть в начале программного блока.

3.4. Программирование алгоритмов линейной структуры

Линейной структурой называется алгоритм, в котором результат получается путем однократного выполнения последовательности действий при любых значениях исходных данных. Согласно линейному алгоритму, прогон программы начинается с ее первого выполняемого оператора. Операторы будут задействованы последовательно, один за другим, в соответствии с их расположением в тексте программы.

Перед реализацией алгоритма этого типа необходимо ознакомиться со структурой программы на языке Паскаль, правилами записи выражений, операторами присваивания и ввода-вывода, стандартными арифметическими функциями (см. прил. 1).

3.4.1. Оператор присваивания

Операторы, не содержащие в себе никаких других операторов, называются простыми, к ним относится оператор присваивания ($:=$).

Оператор присваивания является самым распространенным оператором в любой программе вычислительного характера. Он предназначен для вычисления значения некоторого арифметического выражения и присваивания этого значения переменной.

Синтаксис:

$y:=a,$

где $:=$ – знак операции присваивания; y – идентификатор переменной; a – арифметическое выражение.

Семантика:

1) действие оператора присваивания заключается в вычислении арифметического выражения, стоящего в правой части, и присваивания полученного значения переменной, идентификатор которой стоит в левой части оператора;

2) к моменту выполнения оператора все величины, входящие в арифметическое выражение, должны быть определены, т. е. получены конкретные значения.

Итак, можно сказать, что оператор присваивания предназначен для программирования вычислений по формулам.

3.4.2. Операторы ввода-вывода

Операторы ввода-вывода позволяют обмениваться информацией во время выполнения программы между программистом-пользователем и ПК. В языке программирования Паскаль имеются операторы ввода данных `read`, `readln` и операторы вывода `write`, `writeln`.

Оператор ввода данных `read` позволяет вводить данные в ПК во время выполнения программы и присваивать их значения переменным и элементам массивов, используемым в программе.

Синтаксис:

`read(a1,a2,...,an),`

где `read` – ключевое слово, которое означает «читать»; `a1, a2, ..., an` – список переменных допустимых типов данных, разделенных запятой, например:

```
read(a1,b,c);  
read(a);
```

Семантика:

1) при выполнении оператора `read` вычисления по программе прерываются, и на экране будет расположена последняя информация;

2) программист набирает на клавиатуре соответствующие списку ввода данные (значение данных) минимум через один пробел, набираемая информация высвечивается на экране. После набора данных для одного оператора `read` нажимается клавиша ввода **<Enter>**. Значения данных должны вводиться в строгом соответствии с синтаксисом языка Паскаль.

Если в программе имеется несколько операторов `read`, данные для них вводятся потоком, т. е. после считывания значений переменных для одного оператора `read` данные для следующего в программе оператора `read` набираются в той же строке, что и для предыдущего, до окончания строки, затем происходит переход на следующую строку:

```
Program Vvod;  
  Var a,b,s1:integer;  
      c,d,s2:real;  
  Begin  
    read(a,b);  
    s1:=a+b;  
    read(c,d);  
    s2:=c-d;  
    ...  
  End.
```

Набираем на клавиатуре:

```
2 15 1.5 1.54E+01 <Enter>
```

Оператор чтения `readln` аналогичен оператору `read`, единственное отличие заключается в том, что после считывания последнего в списке значения для одного оператора `readln` данные для следующего оператора `readln` будут считываться с начала новой строки. Если в предыдущем примере заменить операторы `read` на `readln`, то после

набора на клавиатуре значений для *a* и *b* курсор перейдет на новую строку, где будут набираться данные для *c* и *d*:

```
2 15 <Enter>  
1.5 1.54E+01 <Enter>
```

Оператор вывода данных позволяет выводить в процессе вычислений на экран дисплея сообщения, результаты счета программы в виде цифровой или текстовой информации в текстовом режиме.

Синтаксис:

```
write(a1,a2,...,an),
```

где *write* – ключевое слово, означает «запись»; *a1*, *a2*, ..., *an* – список объектов вывода, разделенных запятой, например:

```
write('r=',x:5:3);  
write(10);  
write(a+b-2);
```

В первом варианте значение *x* выводится на экран дисплея с тремя значащими цифрами после запятой, во втором – выражение представлено значением, и в третьем случае выводится результат арифметического выражения.

Обобщим изложенные правила выполнения оператора *write*.

Семантика:

- 1) действие оператора состоит в последовательном выводе на экран дисплея (или устройство печати) значений объектов вывода;
- 2) если объект списка – числовая величина (константа или переменная), то выводится ее численное значение;
- 3) если объект списка в кавычках (константа или символьная переменная), то на экран выводится ее значение в виде последовательности символов без кавычек.

Оператор записи *writeln* аналогичен оператору *write*, но после вывода последнего в списке значения для текущего оператора *writeln* происходит перевод курсора к началу следующей строки. Оператор *writeln*, записанный без параметров, вызывает пропуск строки. Для пояснения работы оператора *writeln* приведем программу для вычисления площади прямоугольника:

```
Program Primer;  
Var a,b,s:integer;  
Begin
```

```

a:=8;b:=4;s:=a*b;
writeln('_____');
writeln(' | Сторона a | Сторона b | Площадь | ');
writeln('_____');
writeln(' |',a:11,' |',b:11,' |',s:9,' | ');
writeln('_____');
End.

```

В результате работы программы получим таблицу:

Сторона a	Сторона b	Площадь
8	4	32

Каждая строка на экране дисплея будет начинаться с первой позиции новой строки.

Рассматриваемые операторы присваивания, ввода и вывода позволяют писать законченные Паскаль-программы простейшей линейной структуры.

Следует отметить, что удобно непосредственно перед вводом выводить на экран сообщение, поясняющее пользователю программы, какую информацию необходимо ввести. Это можно сделать с помощью процедуры вывода, помещенной в программу непосредственно перед процедурой ввода. Ниже приведена программа, предназначенная для ввода двух целых чисел, их сложения и вывода на экран результата. Перед вводом на экран будет выведено сообщение о том, что надо ввести два числа:

```

Program Vvod1;
Var a,b,e:integer;
Begin
  writeln('Введите два числа');
  readln(a,b);
  c:=a+b;
  writeln ('сумма = ',c);
End.

```

Допустим, мы хотим найти сумму чисел 5 и 10, тогда в процессе выполнения программы на экран будет выведено следующее сообщение:

Введите два числа

После выполнения процедуры ввода в диалоговом режиме 5 <Enter> 10 <Enter> на экране монитора будет сообщение вида

сумма = 15

Пример 3.1. Составить программу вычисления объема хлыста, имеющего форму конуса, по заданному радиусу основания и высоте.

Программа:

```
Program Vvod2;
  Var r,h,v:real;
  Begin
    writeln('Введите радиус r, м, и длину h, м, хлыста');
    readln(r,h);
    v:=pi*h*r*r/3;
    writeln('Объем хлыста v = ',v:4:2, 'куб. м');
  End.
```

В процессе выполнения программы на экране монитора будет следующее сообщение:

Введите радиус r, м, и длину h, м, хлыста

Выполнив ввод данных 0.21 10 <Enter>, будем иметь информацию экрана монитора в текстовом режиме:

Объем хлыста v = 0.46 куб. м

В данном случае идентификатор pi является встроенной константой.

3.5. Программирование алгоритмов разветвляющейся структуры

Очередность записи программных строк, состоящих из операторов на языке Паскаль, определяет естественный порядок их выполнения. Однако реализовать алгоритм с помощью последовательно выполняющихся операторов удастся далеко не всегда. Поэтому в языке программирования Паскаль введены операторы управления, которые дают возможность перехода из одного места программы в другое. К ним относятся:

- оператор безусловного перехода goto;
- оператор условного перехода if;
- оператор выбора case.

1. *Оператор безусловного перехода goto.*

Синтаксис:

```
goto <метка> ,
```

где `goto` – ключевое слово, означает «перейти к»; `метка` – идентификатор Паскаля или число целого типа из диапазона 0–9999, описанные в разделе `Label`.

Пример:

```
goto M1;
```

Семантика:

1) осуществляется переход к выполнению оператора, имеющего метку, указанную справа от ключевого слова `goto`;

2) оператор безусловного перехода `goto` изменяет естественный порядок выполнения операторов в программе, но передает управление только по одному направлению.

2. *Оператор условного перехода if.*

Синтаксис:

```
if <условие> then <оператор1> else <оператор2> ,
```

где ключевые слова `if`, `then`, `else` означают соответственно «если», «то», «иначе». Условие – это логическое выражение (булевское). Оно может быть простым, или сложным. Сложные условия образуются с помощью логических операций `and`, `or`, `not`. При записи условий могут использоваться все возможные операции отношения. Результат логического выражения – `true` (истинно) или `false` (ложь).

Семантика:

1) вычисляются выражения, соединенные знаком отношения, и осуществляется проверка условия;

2) если результат выражения – `true`, выполняется `<оператор1>`, если `false` – `<оператор2>`, далее управление передается оператору, непосредственно следующему за данным оператором условного перехода.

В языке Паскаль определено два типа операторов условного перехода. Рассмотренный оператор называется полным оператором. Сокращенный оператор условия имеет вид

```
if <условие> then <оператор>
```

Оператор, расположенный после слова `then`, будет выполнен только при результате условия `true`.

Пример 3.2. Рассмотрим задачу выбора максимального из двух чисел. Для ее решения необходимо сравнить эти числа и в зависимости

от результата сравнения вывести на экран первое или второе число. Выбрать необходимое действие можно с помощью оператора условия if.

Под условием понимается любое выражение логического типа. Если условие выполняется (значение выражения – true), то выбирается оператор1, иначе – оператор2. Используем оператор условия в программе, определяющей максимальное число из двух введенных целых чисел:

```
Program Max;  
  Var n1,n2:integer;  
  Begin  
    writeln('Введите два числа');  
    readln(n1,n2);  
    if n1>n2 then writeln('максимум = ',n1)  
      else writeln('максимум = ',n2);  
  End.
```

Проверяемое в операторе условие

```
if n1>n2 then writeln('максимум = ',n1)  
  else writeln('максимум = ',n2);
```

состоит в том, истинно ли утверждение, что значение переменной n1 больше, чем значение переменной n2. Если это так, то выполнится первый оператор – writeln('максимум = ',n1) и на экран будет выведено значение переменной n1. Если же условие не выполняется, т. е. n1 меньше или равно n2, то выполняется второй оператор – writeln('максимум = ',n2) и на экран выводится значение переменной n2. Обратите внимание, что перед else точка с запятой не ставится.

В языке программирования Паскаль последовательность операторов может быть объединена в группу, рассматриваемую как один оператор или блок. Это выполняется с помощью зарезервированных слов языка Паскаль begin и end.

```
begin  
  оператор1;  
  оператор2;  
  ...  
  операторN;  
end;
```

Составной оператор используется в тех случаях, когда по правилам языка допускается использовать только один оператор, а требуется выполнить несколько действий.

3. *Оператор выбора case.*

Данный оператор управления является обобщением оператора if и позволяет сделать выбор из произвольного числа имеющихся вариантов. Он состоит из выражения, называемого селектором, и списка параметров, каждому из которых предшествует список констант выбора (список может состоять и из одной константы). Как и в операторе if, здесь может присутствовать слово else, имеющее тот же смысл.

Синтаксис:

```
case <выражение-селектор> of
<список1>:<оператор1>;
<список2>:<оператор2>;
...
<списокN>:<операторN>
    else <оператор>
end;
```

Оператор case работает следующим образом. Сначала вычисляется значение выражения-селектора, затем обеспечивается реализация того оператора, константа выбора которого равна текущему значению селектора. Если ни одна из констант не равна текущему значению селектора, выполняется оператор, стоящий за словом else.

Если слово else отсутствует, активизируется оператор, находящийся за словом end, т. е. первый оператор за границей case. Список констант выбора состоит из произвольного количества значений или диапазонов, отделенных друг от друга запятыми. Границы диапазона записываются двумя константами через разделитель '..'.

3.6. Программирование алгоритмов циклической структуры

Операторы повтора (цикла) используются в программе при организации циклов. Цикл – это последовательность операторов, которая может выполняться более одного раза. Если количество повторов известно заранее, используется оператор for, если количество повторов неизвестно, применяются операторы repeat или while.

1. Оператор повтора for состоит из заголовка и тела цикла. Он может быть представлен в двух форматах:

```
for <параметр цикла>:=<s1> to <s2> do <оператор>;
for <параметр цикла>:=<s2> downto <s1> <оператор>;
```

где `for ... do` – заголовок цикла; `s1` и `s2` – выражения, определяющие соответственно начальное и конечное значения параметра цикла; `<оператор>` – тело цикла. Тело цикла может быть простым или составным оператором. Оператор `for` обеспечивает выполнение тела цикла до тех пор, пока не будут перебраны все значения параметра цикла от начального до конечного. Например, оператор `for i:=1 to 10 do write('*');` десять раз выведет на экран в одной строке символ «*», а оператор `for i:=1 to 5 do writeln(Sqrt(i));` выведет пять результатов извлечения квадратного корня из `i`, причем каждый результат будет находиться в отдельной строке. После нормального завершения оператора `for` значение параметра цикла равно конечному значению. Если оператор `for` не выполнялся, значение параметра цикла не определено.

В теле оператора `for` могут находиться другие операторы `for`. Это позволяет строить циклы, содержащие внутренние циклы. Такие внутренние циклы называются вложенными. В операторе `for` Паскаль не допускает изменения параметра цикла на величину, отличную от единицы. Однако это не является большим недостатком, так как любой шаг можно задать при организации циклов в операторах `repeat` и `while`.

2. Оператор повтора `repeat` состоит из заголовка (`repeat`), тела цикла и условия окончания – `until`.

Синтаксис:

```
repeat
  <оператор>;
  <оператор>;
  ...
  <оператор>
until <условие>.
```

Условие – выражение булевского типа. При написании условия допустимы булевские операции и операции отношения. Операторы, заключенные между словами `repeat` и `until`, являются телом цикла. Вначале выполняется тело цикла, затем проверяется условие выхода из цикла. Если результат булевского выражения `false`, тело цикла активизируется еще раз, если результат `true` – происходит выход из цикла.

Оператор `repeat` имеет три характерные особенности: выполняется, по крайней мере, один раз; тело цикла выполняется, пока результат условия равен `false`; в теле цикла может находиться произвольное число операторов без операторных скобок `begin...end`. В отличие от оператора `for`, в этом операторе параметр цикла должен влиять на значение условия, иначе цикл будет выполняться бесконечно.

3. Оператор while аналогичен оператору repeat, но проверка условия выполнения тела цикла производится в самом начале оператора.

Синтаксис:

```
while <условие> do <тело цикла>
```

Условие – булевское выражение, а тело цикла – пустой, простой или составной оператор. Перед каждым выполнением тела цикла вычисляется значение выражения условия. Если результат равен true, тело цикла выполняется и снова вычисляется выражение условия. Если результат равен false, происходит выход из цикла и переход к первому после while оператору. Если перед первым выполнением цикла значение выражения было false, тело цикла вообще не выполняется и происходит переход управления к следующему оператору.

Пример 3.3. Составим программу для вычисления и вывода на экран монитора таблицы значений $y = ax^2$, при $x = 3, 5, 7, \dots, 15$; $a = 16,2$.

С использованием оператора повтора for программа имеет вид:

```
Program Tab1;  
Const a=16.2;  
Var i,x:integer;  
    y:real;  
Begin  
    writeln('Таблица значений');  
    writeln;  
    x:=3;  
    for i:=1 to 7 do  
        begin  
            y:=a*x*x;  
            writeln('x = ',x:3, 'y = ',y:5:2);  
            x:=x+2;  
        end;  
End.
```

С использованием repeat (постусловием) программа выглядит так:

```
Program Tab2;  
Const a=16.2;  
Var x:integer;  
    y:real;  
Begin  
    x:=3;
```

```

repeat
  y:=a*x*x;
  writeln('x = ',x:3, 'y = ',y:5:2);
  x:=x+2;
until x>=16;
End.

```

С использованием оператора while (предусловием) программа имеет вид:

```

Program Tab3;
Const a=16.2;
Var x:integer;
    y:real;
Begin
  x:=3;
  while x<=16 do
  begin
    y:=a*x*x;
    writeln('x = ',x:3, 'y = ',y:5:2);
    x:=x+2;
  end;
End.

```

3.7. Программирование с использованием массивов

3.7.1. Описание массива

Довольно часто возникают задачи, для решения которых необходимо ввести большое количество данных одинакового типа, к тому же при обработке этих данных необходимо выполнить одни и те же операции, например задача вычисления суммы объема хлыстов, имеющих форму конуса.

В этом алгоритме важен порядок расположения переменных, а не их имена, поэтому имя у всех введенных чисел может быть одно, а различаться они должны по номеру в последовательности. Такой способ обращения к данным и обеспечивают массивы.

Под массивом понимается упорядоченная совокупность конечного числа данных одного типа, объединенных под общим именем. Имена массивов образуются так же, как и имена простых перемен-

ных. Массив – это структурированный тип данных, состоящий из фиксированного числа элементов. Использованию в программе массивов должно предшествовать их описание. В нем указывается размер массива, т. е. определяется объем памяти, который необходимо отвести под тот или иной массив.

Синтаксис:

```
Var <список идентификаторов>:array [тип индекса] of <тип компонент>;
```

где Var, array, of – ключевые слова; список идентификаторов – имена массивов; тип индекса – начальное и конечное значение индексов; тип компонент – тип данных объявленных массивов, например:

```
Var x,y:array[1..5] of real;
```

В разделе объявления переменных Var зарезервировано пять ячеек памяти для элементов массива x и пять ячеек для элементов массива y.

Одномерные массивы обычно используются для представления векторов. Массивы могут иметь несколько индексов, тогда элементы массива являются соответственно элементами матрицы (двумерный массив) и т. д.

Для описания массива можно использовать предварительно определенные константы, например:

```
Const a=1;b=5;  
Var x:array[a..b] of real;
```

3.7.2. Действия над элементами массива

После объявления массива каждый его элемент можно обработать, указав идентификатор (имя) массива и индекс элемента в квадратных скобках. Например, записи `mas[2]`, `vector[10]` позволяют обратиться ко второму элементу массива `mas` и десятому элементу массива `vector`.

Индексированные элементы массива называются индексированными переменными и могут быть использованы так же, как и простые переменные. Например, они могут находиться в выражениях в качестве операндов, использоваться в операторах `for`, `while`, `repeat`, входить в качестве параметров в операторы `read`, `readln`, `write`, `writeln`; им можно присваивать любые значения, соответствующие их типу. Рассмотрим типичные ситуации, возникающие при работе с данными типа `array`.

Паскаль не имеет средств ввода-вывода элементов массива сразу, поэтому ввод и вывод значений производится поэлементно.

Значения элементам массива можно присвоить с помощью оператора присваивания, однако чаще всего они вводятся в диалоговом режиме: с использованием клавиатуры, операторов read или readln и оператора повтора for:

```
Program Massiv1;  
  Var x:array[1..5] of real; i:integer;  
  Begin  
    for i:=1 to 5 do  
      begin  
        writeln('Введите x (' ,i, ')');  
        readln(x[i]);  
      end;  
    End.
```

Однако при большом количестве элементов такой способ (диалоговый) ввода не рационален. Гораздо удобнее получить тот же результат, используя оператор присваивания:

```
Program Massiv2;  
  Const n=5;  
  Var x:array[1..n] of real;  
  Begin  
    writeln('Ввод пяти элементов массива x');  
    x[1]:=2;x[2]:=6;x[3]:=14.1;x[4]:=12;x[5]:=55;  
  End.
```

Вывод значений элементов массива выполняется аналогичным образом, но используются операторы write или writeln:

```
Program Massiv3;  
  Const n=5;  
  Var x:array[1..n] of real;  
      i:integer;  
  Begin  
    for i:=1 to n do  
      begin  
        writeln('Введите x(' ,i, ')');  
        readln(x[i]);  
      end;  
    writeln ('Вывод элементов массива x в одну строку');  
    for i:=1 to 5 do  
      write('x(' ,i, ')=' ,x[i]:5:2);
```

```
    readln;  
End.
```

Для задания значений элементов массива их можно объявлять как типизированные константы:

```
Program Massiv4;  
Const x:array[1..5] of real=(-2,2,14.1,12.3,55);  
Var i:integer;  
Begin  
    writeln('Вывод элементов массива x в столбец');  
    for i:=1 to 5 do  
        writeln(x[i]:5:2);  
    writeln('Вывод элементов массива x в строку');  
    for i:=1 to 5 do  
        write(x[i]:5:2);  
    readln;  
End.
```

3.8. Модули

Понятие модуля или, в общем случае, модульного программирования возникло на определенном этапе развития вычислительного дела и было обусловлено, в первую очередь, возрастающими объемами программ, их увеличивающейся внутренней сложностью и коллективным характером разработок.

С этой точки зрения, введение понятия модуля в TurboPascal, которое было проведено начиная с 4-й версии системы, явилось решающим шагом на пути его превращения в язык, пригодный для крупных разработок производственного и коммерческого назначения на современном уровне технологии программирования.

Модульные средства в интегрированной системе TurboPascal заметно слабее аналогичных возможностей тех языков, для которых модульный принцип был положен в основу их проектирования. Однако следует признать, что разработчики системы TurboPascal нашли удачный компромисс, достаточно органично встроив принципиально новое понятие в считающийся уже классическим язык Pascal, не нарушив при этом его целостности, элегантности и простоты и одновременно значительно расширив его возможности.

Все программные ресурсы модуля можно разбить на две части: объекты, прямо предназначенные для использования другими программами или модулями, и объекты рабочего характера. Если модуль содержит некоторую подпрограмму универсального назначения, пригодную для использования другими программами, то вызываемые этой подпрограммой процедуры и функции, содержащиеся в модуле, и используемые ею переменные имеют сугубо внутренний характер. В соответствии с этим модуль, кроме заголовка, имеет две основные части, называемые интерфейсом и реализацией.

TurboPascal имеет восемь стандартных модулей, из них наиболее распространенные – это Crt, Printer, Graph.

Программные ресурсы, сосредоточенные в стандартных модулях, используют мощные пакеты системных средств, которые обеспечивают высокую эффективность и широкий спектр применений системы TurboPascal.

Каждый стандартный модуль содержит логически связанную совокупность типов, констант, переменных и подпрограмм, относящихся к определенной области применений. Для того чтобы воспользоваться ресурсами стандартного модуля, необходимо указать его имя в спецификации использования по обычным правилам.

Рассмотрим назначение основных модулей системы TurboPascal.

Модуль Crt обеспечивает практически полный спектр возможностей для доступа к экрану дисплея в текстовом режиме. Кроме того, в данный модуль включены средства чтения информации с клавиатуры (включая расширенные коды клавиш) и простейшего управления звуком.

Модуль Printer содержит единственный интерфейсный элемент – переменную 1st стандартного типа text, системно связанную с логическим устройством PRN (т. е. с печатающим устройством, если оно имеется в конфигурации). Использование этой переменной в операторах write и writeln приводит к выводу информации на печать, например:

```
writeln(1st,'Информатика и компьютерная графика');
```

Модуль Graph объединяет многочисленные программные средства управления графическим режимом работы дисплея. Данный модуль обеспечивает использование всех возможностей наиболее распространенных типов дисплея – CGA, EGA, VGA, Hercules и т. п., позволяет создавать разнообразные и эффективные графические программы.

3.8.1. Текстовый режим работы. Модуль Crt

Текстовый режим работы ПК характеризуется двумя параметрами: максимальным числом символов в строке и количеством строк на экране. Установка значений этих параметров производится с помощью стандартной процедуры `TextMode`.

Синтаксис:

`TextMode(mode word);`

где `mode` – константа целого типа. Выполнение этой процедуры приводит к очистке экрана и активизации указанного константой `mode` режима. Режим указывают с помощью константы или эквивалентного ей стандартного идентификатора. Например: `TextMode(1)` – активизация цветного текстового режима с экраном в 25 строк по 40 символов в строке; `TextMode(259)` – активизация цветного текстового режима в 50 строк по 80 символов в строке. По умолчанию (при отсутствии процедуры `TextMode`) устанавливается стандартный режим экрана (25 строк по 80 символов в строке).

При работе в текстовом режиме полезными являются следующие процедуры и функции (реализуемые модулем `Crt`):

- `Window(x1,y1,x2,y2:integer)` – процедура, которая определяет на экране дисплея новое активное текстовое окно. Окно – это ограниченная область экрана, выполняющая те же функции, что и полный экран. Однако следует учитывать, что после определения окна все координаты задаются относительно активного окна (начиная с 1 от его левого верхнего угла), а не полного экрана. При определении окна `x1`, `y1` являются координатами верхнего левого угла окна, а `x2`, `y2` – координатами нижнего правого угла с учетом того, что левый угол полного экрана имеет координаты (1; 1), а минимальный размер активного окна включает один столбец и одну строку. При неправильном задании параметров окна `x1`, `y1` или `x2`, `y2` (например, когда `x1 > 80` или `y2 = 30`) процедура `Window` игнорируется;

- `ClrScr` – процедура, которая очищает активное окно и устанавливает курсор в левый верхний угол;

- `GoToXY(x,y:integer)` – процедура, которая перемещает курсор в позицию с координатами (`x`; `y`) в рамках активного окна;

- `Delay(time:integer)` – процедура, которая вызывает задержку выполнения программы на заданное число миллисекунд `time` (0,001 с);

- `Sound(hz:integer)` – процедура, которая включает внутренний динамик, `hz` задает частоту генерируемого динамиком сигнала

в герцах. Звуковой сигнал звучит до тех пор, пока не будет выключен с помощью процедуры NoSound; NoSound – процедура, которая отключает внутренний динамик;

- TextBackGround(color:integer) – устанавливает цвет фона (т. е. цвет области, которая окружает выводимый символ). Здесь color – выражение целого типа в диапазоне от 0 до 7, соответствующее одной из восьми констант цветов, определенное в модуле Crt;

- TextColor(color:integer) – устанавливает цвет выводимых символов, здесь color – выражение целого типа из диапазона от 0 до 15.

Рассмотрим несложную программу, выполняющую вывод на экран дисплея окна с информацией:

```
Program Okno;  
Uses Crt; {Подсоединение модуля Crt}  
Var i:integer;  
Begin  
  TextBackGround(2); {Задание фона, зеленый}  
  TextColor(15); {Задание цвета выводимых символов, белые}  
  Window(20,8,60,18); {Организация окна}  
  ClrScr; {Очистка экрана}  
  GoToXY(18,3); {Установка курсора в образовавшемся окне} wri-  
  teln('Лашенко');  
  GoToXY(13,5);  
  writeln('Анатолий Павлович');  
  readln; {Задержка информации на экране монитора}  
End.
```

Рассмотрим работу этой программы.

Для вызова стандартных программ, обеспечивающих работу с экраном монитора в текстовом режиме, используется Uses Crt. Затем организовывается окно с определенным фоном (TextBackGround) и цветом выводимых символов в этом окне (TextColor). Процедуры GoToXY устанавливают курсор для вывода информации операторами writeln в относительных координатах нового окна.

Наряду с процедурами и функциями организации работы с экраном модуль Crt включает средства, управляемые звуком. В персональном компьютере имеется возможность генерировать с помощью встроенного динамика звуковые сигналы частотой из диапазона 37–32 767 Гц. Воспроизводятся только чистые тона без каких-либо искажений. Сила (громкость) звука не регулируется.

С помощью процедур Sound, NoSound, Delay и операторов цикла for, while, repeat можно создать самые разнообразные эффекты: звучание сирены, метронома, будильника, пение птиц, фрагменты музыкальных произведений и т. д.

Одним из способов построения мелодичных звуковых рядов является использование частот, соответствующих нотам. Частоты загружаются в один массив, продолжительность звучания каждой ноты – в другой.

Программа:

```
Program Nota;  
Uses CRT;  
Const n:array[1..7] of integer=(262,294,330,349,392,440,494);  
      t:array[1..7] of integer=(10,11,12,13,14,15,16);  
Var i:integer;  
Begin  
  for i:=1 to 7 do {Организация цикла}  
  begin  
    Sound(n[i]); {Включение динамика, задание частоты}  
    Delay(t[i]*1000); {Задание продолжительности}  
  end;  
  NoSound; {Выключение динамика}  
  readln; {Задержка информации на экране}  
End.
```

В данной программе, используя оператор цикла for как счетчик и элементы двух массивов (n, t), в которых задается частота нот и продолжительность звучания определенной ноты, имеем музыкальную гамму.

3.8.2. Графический режим работы. Модуль Graph

Графический режим реализуется с помощью модуля Graph и используется для отображения на экране графической информации. При этом экран дисплея разделяется на прямоугольную сетку, состоящую из множества мельчайших элементов изображения, называемых пикселями. Каждый пиксел обладает свойствами светимости. Таким образом, любое изображение может быть синтезировано из множества отдельных точек (пикселов). Реализация графического режима в ПК обеспечивается благодаря наличию специальной схемы, называемой графическим адаптером.

Работу графического адаптера поддерживает специальная программа, называемая драйвером. Загрузочный модуль драйвера хра-

нится в специальном файле с расширением bgi. Если в вашей программе используются какие-либо шрифты, то кроме bgi-файла необходимо наличие одного или более файлов шрифтов (chr-файлов). Загрузка драйвера осуществляется с помощью процедуры InitGraph, имеющей следующий формат:

```
InitGraph(var gd:integer, var gm:integer, pathD:string),
```

где gd, gm и pathD являются параметрами-переменными. Gd задает номер графического драйвера: 0 (или Detect) – автоматический выбор драйвера. Gm задает номер графического режима, допустимого для заданного драйвера (передается процедуре InitGraph только в случае, если gd не равен 0). PathD задает путь к каталогу, в котором находится графический драйвер (bgi-файл).

Если gd равен Detect или 0, то процедура InitGraph автоматически проверяет наличие графической аппаратуры и загружает соответствующий графический драйвер, находящийся в директории pathD. Если при этом имеется выбор из нескольких типов драйверов (или их режимов), то загружается тот из них, который обеспечивает более высокое качество синтезированного изображения в графическом режиме. Если pathD = " (без пробела), то подразумевается, что файлы драйверов содержатся в текущем каталоге.

В случае если gd не равен 0, то значение этой переменной рассматривается как номер драйвера, для которого необходимо определить соответствующий графический режим gm. Режим задается с помощью константы или эквивалентного ей стандартного идентификатора. Каждый графический режим, в свою очередь, характеризуется двумя параметрами: разрешающей способностью экрана и количеством одновременно отображаемых цветов.

Для выхода из графического режима используется процедура CloseGraph. Эта процедура восстанавливает режим, существовавший до инициализации графики. Для работы в графическом режиме на ПК используется свыше 80 стандартных процедур и функций.

В качестве примера рассмотрим технику прямого обращения к процедуре Circle(x,y,r), которая синтезирует и отображает на экране видеотерминала в графическом режиме окружность с центром O(x, y) и радиусом r. Для доступа к этой процедуре воспользуемся командой <Ctrl> + <F1>. Выбрав в окне **Help** команду **Процедуры > Circle > Пример** и нажав клавишу <Enter>, в окне **Edit** загрузим файл по демонстрации стандартной процедуры Circle. Сделав активным курсор (клавиши <C>, <Enter>, , <Enter>) и используя клавиши управления

курсором, выделим тело программы. Затем, после снятия выделения блока (<Ctrl> + <K> + <H>) и дополнения программы соответствующими операторами, на экране монитора будут представлены шесть разноцветных концентрических окружностей с центрами в точке O(320, 240), так как по умолчанию разрешающая способность экрана дисплея – 640×480 пикселей.

Программа:

```
Program Graphik1;
Uses Graph;
Var gd,gm,r:integer;
Begin
  gd:=Detect; {Автоматическая загрузка оптимального драйвера}
  InitGraph(gd,gm,""); {Установка графического режима}
  for r:=1 to 6 do
    begin
      Setcolor(r); {Установка цвета линии}
      Circle(320,240,r* 30); {Вычерчивает окружность}
    end;
  readln;
  CloseGraph; {Закрытие графического режима}
End.
```

Разработка программ в системе TurboPascal, ориентированных на непосредственное использование эффектов с физическими устройствами персонального компьютера, позволяет использовать в теле программы несколько стандартных модулей. Рассмотрим на примере простейшего вычисления по формуле $y = (a + b) / c$ использование модулей Crt и Graph для оформления результатов выполнения программы:

```
Program Graphik2;
Uses CRT,Graph;
Const c=2;
Var a,b,y:real;
    gm,gd:integer;
Begin
  a:=2;
  b:=3;
  y:=(a+b)/c;
  gd:=0;
  InitGraph(gd,gm,"");
  ClrScr; {Очистка экрана монитора}
```

```

SetColor(12); {Задание цвета символов, алые}
SetTextStyle(0,0,14); {Задание шрифта текста}
OutTextXY(180,80,'ответ'); {Выводит текст в точке (180, 80)}
SetTextStyle(0,0,12);
SetColor(8);
SetBkColor(1); {Задание фона, синий}
OutTextXY (100, 200, 'ответ');
Sound(500);
Delay(2000);
NoSound;
readln; {Задержка информации на экране}
CloseGraph; {Закрытие графического режима}
TextMode(1); {Установка текстового режима}
TextColor(15); {Установка цвета символов, белые}
GoToXY(14,12);
writeln('y=',y:5:2);
readln;
End.

```

3.9. Программирование в системе TurboPascal

Пример 3.4. Имеется круг с центром в точке $O(0; 0)$ и произвольным радиусом r . Составить программу для вывода на экран монитора, зная координаты точки $M(x; y)$, следующей информации: если точка находится в круге или на окружности, – «Попал», в противном случае – «Мимо».

Решение.

Математическая формулировка данной задачи имеет следующий вид:
$$z = \begin{cases} \text{Попал, если } x^2 + y^2 \leq r^2, \\ \text{Мимо, если } x^2 + y^2 > r^2. \end{cases}$$

Программа:

```

Program Krug;
{Программа алгоритма разветвляющейся структуры}
Uses CRT;
Const r=5;
Var x,y:real;
Begin
  ClrScr;

```

```

TextMode(2);
GoToXY(30,12);
TextBackGround(2);
writeln('Введите координаты точки M(x, y)');
Readln(x,y);
ClrScr;
TextMode(1);
if x*x+y*y<=r*r then
    begin
        TextColor(3);
        GoToXY(18,12);
        writeln('Попал');
    end
else
    begin
        TextColor(5);
        GoToXY(18, 12);
        writeln('Мимо');
    end;
    readln;
End.

```

Пояснение.

В данной программе после заголовка программы стоит пустой оператор, заключенный в фигурные скобки, который на вычислительный процесс не влияет (комментарий). Процедуры ClrScr, TextMode и GoToXY обеспечивают наглядность информации (интерфейс пользователя), представленной на экране монитора, на результат выполнения программы это не влияет. Оператор writeln('Введите координаты точки M(x, y)') выводит подсказку на экран монитора для ввода чисел, фактических координат точки M(x, y), так как необходимо вычислить нахождение точки относительно заданного круга с радиусом r. Оператор readln присваивает численное значение переменным x и y в диалоговом режиме. Оператор if, реализующий алгоритм разветвляющейся структуры, в зависимости от результата логического выражения (истина или ложь) передаст управление на выполнение составного оператора (begin...end), стоящего после зарезервированных слов then или else. Процедура readln произведет задержку информации на экране до нажатия клавиши <Enter>.

Пример 3.5. Вычислить сумму и произведение положительных элементов массива $x\{-2; 2; -3; 3; -4; 4; -5; 5, -6; 6\}$.

Программа:

```
Program Summa;  
Uses CRT;  
Const n=10;  
Var x:array[1..n] of real;  
    s,p:real;  
    i:integer;  
Begin  
  s:=0;  
  p:=1;  
  for i:=1 to n do  
    begin  
      writeln('Введите x(' ,i,') элемент массива');  
      readln(x[i]);  
      if x[i]>0 then  
        begin  
          s:=s+x[i];  
          p:=p*x[i];  
        end;  
    end;  
  ClrScr;  
  TextMode(1);  
  TextColor(18);  
  GoToXY(3,11);  
  writeln('Сумма = ',s:5:2,'Произведение = ',p:6:2);  
  Delay(5000);  
End.
```

Пояснение.

В этой программе оператор `array` обеспечивает резервирование в памяти компьютера необходимого количества ячеек памяти (n) для массива данных x . Элементы массива будут введены в диалоговом режиме при использовании оператора повтора `for`, причем при использовании параметра цикла i (`integer`) каждому введенному элементу будет присвоено имя $x[i]$, $i = 1, \dots, 10$. По условию задачи необходимо определить сумму (s) и произведение (p) положительных чисел массива x . Реализацию этого процесса производит оператор `if`, находящийся в теле цикла (составной оператор). Язык программирования

Паскаль требует, чтобы до выполнения обращения к ячейке памяти она была определена, это обеспечивает оператор присваивания (s:=0; r:=1). После запуска программы на выполнение (<Ctrl> + <F9>) и ввода данных элементов массива x на экране дисплея будет находиться информация в течение пяти секунд (процедура Delay(5000)):

Сумма = 20.00 Произведение = 720.00

Пример 3.6. Составить программу по вычислению объема четырех бревен, имеющих форму усеченного конуса, длиной пять метров и разными радиусами: нижним – m, см {24; 21; 18; 16}, верхним – rv, см {20; 18; 14; 13}.

Программа:

```
Program Brevno;
Uses Crt;
Const n=4;l=5;
Var m,rv:array[1..n] of real;
    i:integer;
    v:real;
Begin
  m[1]:=0.24;m[2]:=0.21;m[3]:=0.18;m[4]:=0.16;
  rv[1]:=0.20;rv[2]:=0.18;rv[3]:=0.14;rv[4]:=0.13;
  v:=0;
  for i:=1 to n do
    v:=v+pi*l*(rv[i]*rv[i]+rv[i]*m[i]+m[i]*m[i])/3;
  TextBackground(2);
  TextColor(15);
  Window(20,8,60,18);
  ClrScr;
  GoToXY(6,3);
  writeln('Объем четырех бревен равен');
  GoToXY(15,5);
  writeln(v:5:2,'куб. м');
  repeat
    for i:=1 to 10 do
      begin
        Sound(1200);
        Delay(100);
        NoSound;
      end;
    Delay(180);
```

```
Until KeyPressed;  
End.
```

Пояснение.

В разделе описания Var оператором array зарезервировано восемь ячеек памяти для характеристик каждого бревна (rn[i], rv[i]). В разделе Const задана длина бревен (l=5). Применяв операторы присваивания (:=) и команды редактора системы программирования TurboPascal, введем радиусы бревен в автоматическом режиме. Оператор цикла for вычислит сумму объемов четырех бревен различных диаметров. При использовании оператора вывода writeln в текстовом режиме на экране монитора появится следующая информация:

```
Объем четырех бревен равен  
2.10 куб. м
```

Она будет выведена в созданном окне с определенным фоном (TextBackGround) и цветом выводимых символов в этом окне (TextColor). Процедуры GoToXY устанавливают курсор для вывода информации в относительных координатах нового окна.

В процессе выполнения программы возможны самые различные аварийные ситуации: ошибки при вводе или выводе данных, выход из допустимых диапазонов и многое другое. Звуковое сопровождение таких ситуаций служит для экстренного привлечения внимания пользователя и является необходимым требованием при разработке хороших программ. Звучание может быть разовым, бесконечным (до прерывания пользователем) или дискретным (через определенные промежутки времени). В качестве примера в теле программы с использованием оператора цикла for создана процедура, имитирующая телефонный звонок. Оператор repeat устанавливает продолжительность работы оператора повтора до нажатия любой клавиши.

Пример 3.7. Написать программу вычисления корня трансцендентного уравнения $\cos(x) - x = 0$, используя метод дихотомии (деление отрезка пополам).

Программа:

```
Program dixotom;  
{Метод дихотомии}  
Uses Crt;  
Var a,b,x,e:real;  
Label 1;  
{Задание функции пользователя}
```

```

Function F(t:real):real;
begin
  F:=cos(t)-t;
end;
Begin
{Организация окна для ввода информации}
  Window(20,3,60,5);
  TextBackGround(3);
  TextColor(9);
  ClrScr;
  writeln('Введите границы отрезка и точность');
{Ввод информации}
  readln(a,b,e);
{Организация цикла, заданного неявно}
  while abs(b-a)>e do
    begin
      x:=(a+b)/2;
      if F(x)=0 then goto 1;
      if F(x)*F(a)<0 then b:=x
        else a:=x;
    end;
{Организация окна для вывода информации}
  TextBackGround(5);
  TextColor(12);
  Window(20,10,60,12);
  ClrScr;
{Вывод информации}
  writeln('корень x=',x:8:2);
{Задержка информации на экране монитора}
  readln;
{Задание стандартного текстового режима}
  TextMode(2);
End.

```

Пояснение.

В разделе описаний (Function) задается функция, для которой находится корень. Для этого определяется отрезок, как правило, графическим способом (см. рис. 4.3). Используя оператор цикла с предусловием (while), реализуем итерационный процесс до выполнения условия $(abs(b-a)>e)$. Для оформления ввода-вывода

информации используем процедуру Window. Так как в программе организовывается два окна, необходимо включить процедуру Text-Mode(2).

Пример 3.8. Пусть имеется ряд точек $A_1(1; 2)$, $A_2(2; 2.9)$, $A_3(3; 4.05)$, $A_4(4; 5)$, $A_5(5; 6.1)$. Надо написать программу определения коэффициентов a_0 и a_1 эмпирической линейной зависимости $y(x) = a_0 + a_1 \cdot x$, таких чтобы прямая прошла в «облаке» точек с наименьшим общим среднеквадратическим отклонением от них. Для решения этой задачи (линейной регрессии) используется метод наименьших квадратов.

Программа:

```
Program mnk;
{Метод наименьших квадратов}
Const n=5
Var x,y:array[1..n] of real;
    sx,sy,sx2,sxy,a,a0,a1:real;
    i:integer;
Begin
{Ввод данных}
x(1):=1;x(2):=2;x(3):=3;x(4):=4;x(5):=5;
y(1):=2;y(2):=2.9;y(3):=4.05;y(4):=5; y(5):=27.6;
{Накопление сумм}
sx:=0;sy:=0;sx2:=0;sxy:=0;
for i:=1 to n do
begin
sx:=sx+x(i);
sy:=sy+y(i);
sx2:=sx2+x(i)*x(i);
sxy:=sxy+x(i)*y(i);
end;
{Определение эмпирических коэффициентов}
a:=n*sx2-sx*sx;
a0:=(sy*sx2-sxy*sx)/a;
a1:=(n*sxy-sx*sy)/a;
{Вывод значений коэффициентов}
writeln('a0=',a0:5:2,'a1=',a1:5:2);
End.
```

Пояснение.

Для определения коэффициентов уравнения регрессии a_0 и a_1 , представленного линейной зависимостью, на основании метода наименьших квадратов (МНК) следует решить систему уравнений относительно неизвестных a_0 и a_1 :

$$\begin{cases} a_0 * n + a_1 * sx = sy, \\ a_0 * sx + a_1 * sx2 = sxy, \end{cases}$$

где n – количество экспериментальных точек;

$$sx = \sum_{i=1}^n x_i; \quad sy = \sum_{i=1}^n y_i; \quad sx2 = \sum_{i=1}^n x_i^2; \quad sxy = \sum_{i=1}^n x_i * y_i.$$

Используя правило Крамера, имеем:

$$\Delta = \begin{vmatrix} n & sx \\ sx & sx2 \end{vmatrix}; \quad \Delta_{a_0} = \begin{vmatrix} sy & sx \\ sxy & sx2 \end{vmatrix}; \quad \Delta_{a_1} = \begin{vmatrix} n & sy \\ sx & sxy \end{vmatrix},$$

откуда находим $a_0 = \Delta_{a_0} / \Delta$; $a_1 = \Delta_{a_1} / \Delta$.

Ввод элементов массива производится в автоматическом режиме. С помощью оператора цикла `for` накапливаются суммы, необходимые для реализации МНК.

4. ПРИКЛАДНОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ

4.1. Общие сведения о программе Mathcad

Для решения сложных расчетных задач используют программы, написанные специально. В то же время в научной работе встречается широкий спектр задач ограниченной сложности, для решения которых можно использовать универсальные средства.

К универсальным программам, пригодным для решения таких задач, относится, например, программа Mathcad, которая представляет собой автоматизированную систему, позволяющую динамически обрабатывать данные в числовом и аналитическом (формульном) виде. Программа Mathcad сочетает в себе возможности проведения расчетов и подготовки форматированных научных и технических документов. Следует хорошо представлять себе, что в состав Mathcad входят несколько интегрированных между собой компонентов:

- мощный текстовый редактор, позволяющий вводить, редактировать и форматировать как текст, так и математические выражения;
- вычислительный процессор, умеющий проводить расчеты по введенным формулам, используя встроенные численные методы;
- символьный процессор, являющийся, фактически, системой искусственного интеллекта.

Объединение текстового, формульного и графического редакторов с вычислительным ядром позволяет готовить активные электронные документы с высоким качеством оформления, способные выполнять расчеты с наглядной демонстрацией результатов. Итоговые документы могут трансформироваться в файлы форматов RTF и HTML и использоваться в пакете MS Office и в сетях Internet, Intranet.

Научно-технические документы обычно содержат формулы, результаты расчетов в виде таблиц данных или графиков, текстовые комментарии или описания, другие иллюстрации. В программе Mathcad им соответствуют два вида объектов: формулы и текстовые блоки. Формулы вычисляются с использованием числовых констант, переменных, функций (стандартных и определенных пользователем), а также общепринятых обозначений математических операций.

Включенные в документ Mathcad формулы автоматически приводятся к стандартной научно-технической форме записи. Графики,

которые автоматически строятся на основе результатов расчетов, также рассматриваются как формулы. Комментарии, описания и иллюстрации размещаются в текстовых блоках, которые игнорируются при проведении расчетов.

Чтобы буквенные обозначения можно было использовать при расчетах по формулам, им должны быть присвоены числовые значения. В программе Mathcad буквенные обозначения рассматриваются как переменные и их значения задаются при помощи оператора присваивания. Таким же образом можно задать числовые последовательности, аналитически определенные функции, матрицы и векторы.

Если все значения переменных известны, то для вычисления числового значения выражения (скалярного, векторного или матричного) надо подставить все числовые значения и произвести все заданные действия. В программе Mathcad для этого применяют оператор вычисления. В ходе вычисления автоматически используются значения переменных и определения функций, заданные в документе ранее. Удобно задать значения известных параметров, провести вычисления с использованием аналитических формул, результат присвоить некоторой переменной, а затем использовать оператор вычисления для вывода значения этой переменной. Изменение значения любой переменной, коррекция любой формулы означает, что все расчеты, зависящие от этой величины, нужно проделать заново. Такая необходимость возникает при выборе подходящих значений параметров или условий, поиске оптимального варианта, исследовании зависимости результата от начальных условий. Электронный документ, разработанный в программе Mathcad, готов к подобной ситуации. При изменении какой-либо формулы Mathcad автоматически производит необходимые вычисления, обновляя изменившиеся значения.

4.2. Входной язык системы Mathcad

При решении задач система Mathcad требует от пользователя описания алгоритма решения задачи на входном языке.

Алфавит входного языка пакета Mathcad – совокупность специальных знаков и слов, которые используются при задании команд, необходимых для решения задачи.

Алфавит содержит:

- прописные и строчные буквы (латинские и греческие);
- цифры от 0 до 9;

- системные переменные;
- операторы;
- имена встроенных функций;
- специальные знаки;
- типы данных: константы, переменные, массивы, данные файлового типа;
 - операторы: элементы языка, с помощью которых создаются математические выражения;
 - функции: встроенные и определяемые пользователем.

4.2.1. Операторы

Набор основных арифметических операторов системы (сложение «+», вычитание «-», умножение «*», деление «/», возведение в степень «^») возможен с клавиатуры или с использованием кнопок панели инструментов **Арифметика**, которые появляются при выборе из меню **Вид** > **Панели инструментов** > **Арифметика** или при щелчке на пиктограмме панели инструментов **Арифметика** на панели инструментов **Математика** (рис. 4.1).

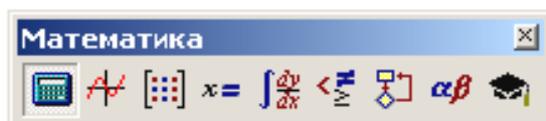


Рис. 4.1. Панель инструментов **Математика**

Элементы формул можно вводить с клавиатуры или с помощью панели инструментов **Математика**, которая вызывается командой меню **Вид** > **Панели инструментов** > **Математика**.

Назначение элементов этой панели следующее:

 – панель инструментов **Арифметика** (арифметические инструменты) – ввод чисел, знаков математических операций, наиболее часто используемых стандартных функций;

 – панель инструментов **График** (инструменты графиков) – построение графиков;

 – панель инструментов **Матрицы** (векторные и матричные инструменты) – ввод векторов и матриц и задание матричных операций;



– панель инструментов **Вычисления** (инструменты некоторых знаков) – ввод операторов вычисления и знаков логических операций;



– панель инструментов **Исчисление** (операторы математического анализа) – задание операций, относящихся к математическому анализу;



– панель инструментов **Булево** (символы логических операций) – ввод знаков логических выражений;



– панель инструментов **Программирование** (операторы и символы программирования) – ввод операторов программирования в системе Mathcad;



– панель инструментов **Греческий алфавит** (символы греческого алфавита) – ввод греческих букв;



– панель инструментов **Символы** (символические операторы) – осуществление символьных вычислений.

Основные операторы:

- оператор присваивания «:=» (двоеточие и знак равенства);
- оператор вывода «=>» (знак равенства). Например, после ввода выражения $4 + 19^2 - \log(3)$ надо ввести с клавиатуры символ «=>», в итоге отображается результат вычисления;
- оператор сравнения (равно) «=>» (символический знак равенства). Например, при записи $\sin(x) = 0$ необходимо использовать символический знак равенства, ввод которого осуществляется с помощью пиктограммы **Булево равенство =**, находящейся на панели инструментов **Вычисления**, или нажатием клавиш <Ctrl> + <=>.

4.2.2. Числовые и размерные константы

Константы – именованные объекты, хранящие некоторые значения, которые не могут быть изменены.

Числовые константы задаются с помощью арабских цифр, десятичной точки, знака «минус».

Например: 567 – целочисленная десятичная константа; 45.6 – десятичная константа с дробной частью; $3.5 \cdot 10^{-5}$ – десятичная константа с мантиссой 3.5 и порядком –5.

Размерные константы задаются в виде

<числовая константа> <знак умножения> <единица измерения>

Например, 5 секунд в Mathcad выглядит как $5 \cdot \text{sec}$.

Последовательность действий при вводе размерных констант:

- введите константу или переменную;
- введите знак умножения * (звездочка);
- выберите команду меню **Вставка** > **Единицы измерения**;
- выберите в окне **Измерения** нужный тип;
- выберите в окне **Единицы** нужную величину;
- щелкните на кнопке **ОК**.

4.2.3. Переменные

Переменные в системе Mathcad – именованные объекты, имеющие некоторое значение, способное изменяться по ходу выполнения программы.

Имена переменных могут иметь произвольную длину, но начинаться должны с буквы. Они могут состоять из букв (латинских и греческих), цифр от 0 до 9, символа бесконечности, символа подчеркивания, апострофа, символа процента (%), нижних индексов.

Для того чтобы можно было вычислить выражение, зависящее от каких-либо переменных, значения этих переменных должны быть определены.

Для присвоения значения переменной необходимо:

- ввести имя переменной;
- набрать двоеточие (:), что приведет к появлению знака присваивания (:=) и следующего за ним поля ввода, или щелкнуть на кнопке **Присвоить значение** на панели инструментов **Арифметика**;
- набрать в поле ввода число или выражение.

Mathcad вычислит соответствующее значение и присвоит его переменной.

Все переменные и функции, присутствующие во введенном выражении, должны быть определены заранее. В противном случае переменные, значения которых не определены к моменту вычисления выражения, будут отмечены на экране дисплея негативным изображением.

Если переменной присваивается начальное значение с помощью оператора «:=», то присваивание называется локальным (при-

сваивание должно производиться перед использованием переменной в выражениях).

Если переменной присваивается начальное значение с помощью оператора знака « \equiv » из панели инструментов **Вычисление**, то присваивание называется глобальным (присваивание может проводиться в любом месте документа).

Размерные переменные характеризуются не только значением, но и указанием физической величины, значение которой они хранят.

Системные переменные имеют значения, которые определены системой Mathcad.

К системным переменным относятся:

- число «пи» – $\pi = 3.14$;
- основание натурального логарифма – $e = 2.71$;
- системная бесконечность – $\infty = 10^{37}$;
- процент – $\% = 0.01$;
- погрешность численных методов – **tol** = 0.001;
- нижняя граница индекса массивов – **origin** = 0.

4.2.4. Массивы

Массивами в системе Mathcad (arrays) называют упорядоченные последовательности чисел, или элементов массива. Доступ к любому элементу массива возможен по его индексу, т. е. номеру в последовательности чисел (в листинге 1 a – это массив, a_1 – его элемент). Применение массивов чрезвычайно эффективно в математических расчетах.

Листинг 1. Одномерный массив (вектор):

$$a := \begin{pmatrix} 14 \\ 1.4 \\ 4.7 \end{pmatrix}$$

$$a_0=14 \quad a_1=1.4 \quad a_2=4.7$$

В Mathcad условно выделяются два типа массивов:

- векторы, или одноиндексные массивы (листинг 1), матрицы, или двухиндексные (листинг 2), и тензоры, или многоиндексные;
- ранжированные переменные (range variables) – векторы, элементы которых определенным образом зависят от их индекса.

Листинг 2. Двумерный массив (матрица):

$$a := \begin{pmatrix} 0.1 & 2.8 \\ 3.7 & 0 \end{pmatrix}$$

$$a_{0,0}=0.1 \quad a_{0,1}=2.8 \quad a_{1,0}=3.7 \quad a_{1,1}=0$$

Доступ ко всему массиву осуществляется по имени векторной переменной. Например, последовательность символов «a», «=>» в листингах 1 и 2 приведет к выводу соответствующего вектора или матрицы. В Mathcad имеются операторы и встроенные функции, которые действуют на векторы и матрицы целиком, например транспонирование, матричное умножение и т. д.

Над элементами массива можно совершать действия, как над обычными числами. Нужно только правильно задать соответствующий индекс или сочетание индексов массива. Например, чтобы получить доступ к нулевому элементу вектора a из листинга 1:

- введите имя переменной массива (a);
- нажмите кнопку **Subscript (Нижний индекс)** со значком x_n на панели **Matrix (Матрица)** либо введите <[>;
- в появившийся справа снизу от имени массива местозаполнитель введите желаемый индекс (0).

Если после этого ввести знак численного вывода, то справа от него появится значение нулевого элемента вектора, как показано во второй строке листинга 1.

Чтобы получить доступ к элементу многоиндексного массива (например, элементу $a_{1,0}$ матрицы a из листинга 2):

- введите имя переменной массива (a);
- перейдите к вводу нижнего индекса, щелкнув <[>;
- введите в местозаполнитель индекса первый индекс (1), запятую (,) и в появившийся после запятой местозаполнитель введите второй индекс (0).

В результате будет получен доступ к элементу последней строке листинга 2.

В рассмотренных листингах нумерация индексов массивов начинается с нуля, иными словами, первый элемент массива имеет индекс 0. Стартовый индекс массива задается системной переменной **ORIGIN**, которая по умолчанию равна нулю. Если необходимо нумеровать элементы векторов и матриц с единицы, присвойте этой переменной значение 1 (листинг 3).

Листинг 3. Изменение нумерации индексов массивов:

```
ORIGIN:=1
a:=

$$\begin{pmatrix} 14 \\ 1.4 \\ 4.7 \end{pmatrix}$$

a1=14 a2=1.4 a3=4.7
```

Помимо доступа к отдельным элементам массива имеется возможность совершать действия над его подмассивами (например, векторами-столбцами, образующими матрицу). Делается это с помощью оператора со значком $x^{<>}$ на панели **Matrix**.

Ранжированные переменные – это особый класс переменных, который в Mathcad заменяет управляющие структуры, называемые циклами. Эти переменные имеют ряд фиксированных значений, меняющихся от начального до конечного с определенным шагом. Ранжированные переменные имеют имя и индекс (порядковый номер) каждого элемента.

Для создания ранжированной переменной целочисленного типа используется выражение

$$\text{Name}:=N_{\text{begin}}..N_{\text{end}}$$

Например, если в вычислениях используется ряд натуральных чисел $\{1, 2, 3, 4, 5\}$, в Mathcad это выглядит как $M:=1..5$ (имя переменной может быть любым, русские символы не допускаются).

Для создания ранжированной переменной общего вида используется выражение

$$\text{Name}:=N_{\text{begin}},(N_{\text{begin}}+\text{Step})..N_{\text{end}}$$

где Name – имя переменной; N_{begin} – начальное значение переменной; N_{end} – конечное значение переменной; Step – шаг изменения переменной.

Например:

- если в вычислениях используется ряд чисел $\{1, 3, 5, 7, 9\}$, в Mathcad это выглядит как $M:=1,(1+2)..9$ (имя переменной может быть любым) или $M:=1,3..9$;
- если в вычислениях используется ряд чисел $\{9, 7, 5, 3, 1\}$, в Mathcad это выглядит как $M:=9,7..1$ (имя переменной может быть любым).

Вывод значений ранжированной переменной:

$\text{Name} =$

Существует несколько способов создания массива. Самый простой и наглядный способ создания вектора или матрицы заключается в следующем:

- нажмите кнопку **Matrix or Vector (Матрица или вектор)** на панели **Matrix (Матрица)** (рис. 4.2), либо клавиши $\langle \text{Ctrl} \rangle + \langle \text{M} \rangle$, либо выберите пункт меню **Insert > Matrix (Вставка > Матрица)**;
- в диалоговом окне **Insert Matrix (Вставка матрицы)** задайте целое число столбцов и строк матрицы, которую хотите создать.

Например, для создания вектора 3×1 введите показанные на рис. 4.2 значения;

- нажмите кнопку **OK** или **Insert (Вставить)** – в результате в документ будет вставлена заготовка матрицы с определенным числом строк и столбцов;

- введите значения в местозаполнители элементов матрицы. Переходить от одного элемента матрицы к другому можно с помощью указателя мыши, клавиш со стрелками либо клавиши **<Tab>**.

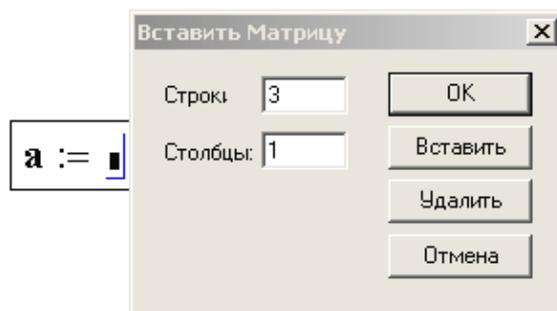


Рис. 4.2. Вставка матрицы

Добавление в уже созданную матрицу строк или столбцов производится точно так же.

В местозаполнители элементов матрицы можно вставлять не только числа (действительные или комплексные), но и любые математические выражения, состоящие из переменных, операторов, встроенных и пользовательских функций.

Другой способ создания массива заключается в определении любого количества его элементов. Это можно сделать, присваивая значения непосредственно отдельным элементам массива, применяя ранжированные переменные (листинг 4).

Листинг 4. Создание матрицы определением ее элементов:

$$A_{0,0}:=3 \quad A_{3,2}:=9$$
$$A = \begin{pmatrix} 3 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 9 \end{pmatrix}$$

Любой из этих способов позволяет присвоить нужное значение как всем элементам массива, так и части из них либо даже одному-единственному элементу. В последнем случае создается массив, размерность которого задается индексами введенного элемента

(листинг 4), а неопределенным элементам по умолчанию присваиваются нулевые значения.

В Mathcad легко создать матрицы определенного вида с помощью одной из встроенных функций:

- $\text{identity}(N)$ – единичная матрица размера $N \times N$;
- $\text{diag}(v)$ – диагональная матрица, на диагонали которой находятся элементы вектора v ;
- $\text{geninv}(A)$ – создание матрицы, обратной (слева) матрице A ;
- $\text{rref}(A)$ – преобразование матрицы или вектора A в ступенчатый вид, где N – целое число; v – вектор; A – матрица из действительных чисел.

Примеры использования этих функций приведены в листинге 5.

Листинг 5. Создание матриц специального вида:

$$\text{identity}(3) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\text{diag} = \left(\begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} \right) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{pmatrix}$$

$$A := \begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{pmatrix}$$

$$\text{geninv}(A) = \begin{pmatrix} -1.333 & -0.333 & 0.667 \\ 1.083 & 0.333 & -0.417 \end{pmatrix}$$

$$\text{geninv}(A) \cdot A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

$$\text{rref}(A) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix}$$

$$\text{rref} \left(\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix} \right) = \begin{pmatrix} 1 & 0 & -1 \\ 0 & 1 & 2 \end{pmatrix}$$

В любом месте документа допускается как переопределение каждого из элементов массива, так и изменение его размерности. Чтобы поменять размерность всего массива, необходимо присвоить любое значение новому элементу, индексы которого выходят за границы прежней размерности.

4.2.5. Стандартные и пользовательские функции

Произвольные зависимости между входными и выходными параметрами задаются при помощи функций. Функции принимают набор параметров и возвращают значение, скалярное или векторное (матричное). В формулах можно применять стандартные встроенные функции, а также функции, определенные пользователем.

Чтобы использовать функцию в выражении, надо определить значения входных параметров в скобках после имени функции. Имена простейших математических функций можно ввести с панели инструментов **Arithmetic (Счет)**.

Информация о других функциях содержится в справочной системе. Вставить в выражение стандартную функцию можно командой **Insert > Function (Вставка > Функция)**. В диалоговом окне **Insert Function (Вставка функции)** слева выбирается категория, к которой относится функция, а справа – конкретная функция. Пользовательские функции должны быть сначала определены. Определение задается при помощи оператора присваивания. В левой части указывается имя пользовательской функции и в скобках формальные параметры – переменные, от которых она зависит. Справа от знака присваивания эти переменные должны использоваться в выражении.

При применении пользовательской функции в последующих формулах ее имя вводят вручную.

4.3. Ввод формул и текста

Документ программы Mathcad называется рабочим листом. Он содержит следующие объекты: формулы и текстовые блоки. В ходе расчетов формулы обрабатываются последовательно, слева направо и сверху вниз, а текстовые блоки игнорируются.

Ввод информации осуществляется в месте расположения курсора. Программа Mathcad использует три вида курсоров. Если ни один объект не выбран, применяется крестообразный курсор, определяющий место создания следующего объекта. При вводе формул используется уголко-вый курсор, указывающий текущий элемент выражения, при вводе данных в текстовый блок – текстовый курсор в виде вертикальной черты.

Формулы – основные объекты рабочего листа. Новый объект по умолчанию является формулой. Чтобы начать ввод формулы, надо

установить крестообразный курсор в нужное место и приступить к вводу букв, цифр, знаков операций. При этом создается область формулы, в которой появляется уголковый курсор, охватывающий текущий элемент формулы, например имя переменной (функции) или число. При вводе бинарного оператора по другую сторону знака операции автоматически появляется заполнитель в виде черного прямоугольника. В это место вводят очередной операнд.

Для управления порядком операций используют скобки, которые можно вводить вручную. Уголковый курсор позволяет автоматизировать такие действия. Чтобы выделить элементы формулы, которые в рамках операции должны рассматриваться как единое целое, используют клавишу <Пробел>. При каждом ее нажатии уголковый курсор «расширяется», охватывая элементы формулы, примыкающие к данному фрагменту. После ввода знака операции элементы в пределах уголкового курсора автоматически заключаются в скобки.

Элементы формул можно вводить с клавиатуры или с помощью специальных панелей управления. Панели управления открывают с использованием меню **View (Вид)** или кнопками панели управления **Math (Математика)**. Для ввода элементов формул предназначены следующие панели управления:

- **Arithmetic (Счет)** – для ввода чисел, знаков типичных математических операций и наиболее часто употребляемых стандартных функций;
- **Evaluation (Вычисление)** – для ввода операторов вычисления и знаков логических операций;
- **Graph (График)** – для построения графиков;
- **Matrix (Матрица)** – для ввода векторов и матриц и задания матричных операций;
- **Calculus (Исчисление)** – для задания операций, относящихся к математическому анализу;
- **Greek (Греческий алфавит)** – для ввода греческих букв;
- **Symbolic (Аналитические вычисления)** – для управления аналитическими преобразованиями.

Текст, помещенный в рабочий лист, содержит комментарии и описания и предназначен для ознакомления, а не для использования в расчетах. Программа Mathcad определяет назначение текущего блока автоматически при первом нажатии клавиши <Пробел>. Если введенный текст не может быть интерпретирован как формула, блок преобразуется в текстовый и последующие данные рассматриваются как текст. Создать текстовый блок позволяет команда **Insert >**

Text Region (Вставка > Текстовый блок). Иногда требуется встроить формулу внутрь текстового блока. Для этого служит команда **Insert > Math Region (Вставка > Формула)**.

4.4. Форматирование формул и текста

Для форматирования формул и текста в программе Mathcad используется панель инструментов **Formatting (Форматирование)**. С ее помощью можно индивидуально отформатировать любую формулу или текстовый блок, задав гарнитуру и размер шрифта, а также полужирное, курсивное или подчеркнутое начертание символов. В текстовых блоках можно также задавать тип выравнивания и применять маркированные и нумерованные списки.

В качестве средств автоматизации используются стили оформления. Выбрать стиль оформления текстового блока или элемента формулы можно из списка **Style (Стиль)** на панели инструментов **Formatting (Форматирование)**. Для формул и текстовых блоков применяются разные наборы стилей.

Для изменения стиля оформления формулы или создания нового стиля используется команда **Format > Equation (Формат > Выражение)**. Изменение стандартных стилей **Variables (Переменные)** и **Constants (Константы)** влияет на отображение формул по всему документу. Стиль оформления имени переменной учитывается при ее определении. Так, переменные x и x рассматриваются как различные и невзаимозаменяемые.

При оформлении текстовых блоков можно использовать более обширный набор стилей. Настройка стилей текстовых блоков производится командой **Format > Style (Формат > Стиль)**.

4.5. Решение уравнений и систем

Для численного поиска корней уравнения в программе Mathcad используется встроенная функция **root**. Она служит для решения уравнений вида $f(x) = 0$, где $f(x)$ – выражение, корни которого нужно найти, а x – неизвестное. Чтобы найти корни с помощью функции **root**, надо присвоить искомой переменной начальное значение, а затем вычислить корень при помощи вызова функции: **root(f(x), x)**. Здесь $f(x)$ – функция переменной x , используемой в качестве второго параметра. Функция **root** возвращает значение независимой переменной, обращающее функцию $f(x)$ в 0. Например:

```
x:=1  
root(2·sin(x)-x,x)=1.895
```

Если уравнение имеет несколько корней (как в данном примере), то результат, выдаваемый функцией `root`, зависит от выбранного начального приближения:

```
x:=0  
root(2·sin(x)-x,x)=0
```

Для решения уравнения или системы уравнений (неравенств) численным методом в системе Mathcad можно использовать так называемый блок решения, который начинается с ключевого слова `given` (дано) и заканчивается вызовом функции `find` (найти). Между ними располагают «логические утверждения», задающие ограничения на значения искомых величин, иными словами, уравнения и неравенства. Всем переменным, используемым для обозначения неизвестных величин, должны быть заранее присвоены начальные значения. Например:

```
x:=2  
Given  
2·sin(x)-x=0  
Find(x)=1.895
```

4.6. Построение графиков

Если нужно построить двухмерный график в прямоугольных декартовых координатных осях $X - Y$, дают команду **Insert > Graph > X – Y Plot (Вставка > График > Декартовы координаты)**. В области размещения графика находятся заполнители для указания отображаемых выражений и диапазона изменения величин. Заполнитель у середины оси координат предназначен для переменной или выражения, отображаемого по этой оси. Обычно используют диапазон или вектор значений. Граничные значения по осям выбираются автоматически в соответствии с диапазоном изменения величины, но их можно задать и вручную.

В одной графической области можно построить несколько графиков (рис. 4.3). Для этого надо у соответствующей оси перечислить несколько выражений через запятую. Например:

$z1(x) := \cos(x)$
 $z2(x) := x$
 $z(x) := z1(x) \cdot z2(x)$
 $x := -4, -3..4$

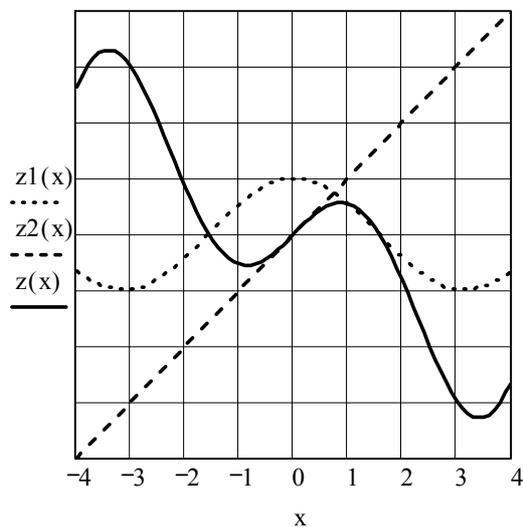


Рис. 4.3. Графики Mathcad

Каждая кривая изображается своим цветом и начертанием, а для форматирования графика надо дважды щелкнуть на области графика для вызова окна форматирования (рис. 4.4).

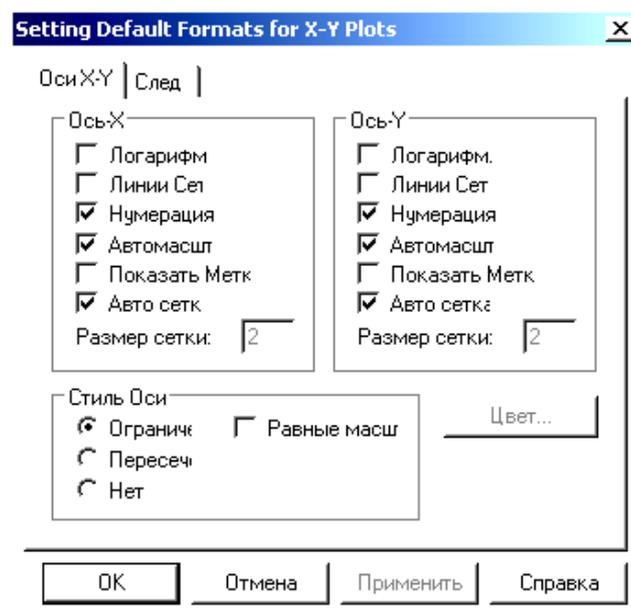


Рис. 4.4. Окно для форматирования графика

Для управления отображением построенных линий служит вкладка **Traces (Линии)** в открывшемся диалоговом окне. Текущий формат каждой линии приведен в списке, а под списком расположены элементы управления, позволяющие изменять формат. Поле **Legend Label (Описание)** задает описание линии, которое отображается только при сбросе флажка **Hide Legend (Скрыть описание)**.

Список **Symbol (Символ)** позволяет выбрать маркеры для отдельных точек, список **Line (Тип линии)** задает тип линии, список **Color (Цвет)** – цвет. Список **Type (Тип)** определяет способ связи отдельных точек, а список **Width (Толщина)** – толщину линии.

4.7. Аналитические вычисления

С помощью аналитических вычислений находят аналитические или полные решения уравнений и систем, а также проводят преобразования сложных выражений (например, упрощение). Иначе говоря, при таком подходе можно получить нечисловой результат.

В программе Mathcad конкретные значения, присвоенные переменным, при этом игнорируются – переменные рассматриваются как неопределенные параметры. Команды для выполнения аналитических вычислений в основном сосредоточены в меню **Symbolics (Аналитические вычисления)**.

Чтобы упростить выражение (или часть выражения), надо выбрать его при помощи уголкового курсора и дать команду **Symbolics > Simplify (Аналитические вычисления > Упростить)**. При этом выполняются арифметические действия, сокращаются общие множители и приводятся подобные члены, применяются тригонометрические тождества, упрощаются выражения с радикалами, а также выражения, содержащие прямую и обратную функции. Некоторые действия по раскрытию скобок и упрощению сложных тригонометрических выражений требуют применения команды **Symbolics > Expand (Аналитические вычисления > Раскрыть)**.

Команду **Symbolics > Simplify (Аналитические вычисления > Упростить)** применяют и в более сложных случаях. Например, с ее помощью можно:

- вычислить предел числовой последовательности, заданной общим членом;
- найти общую формулу для суммы членов числовой последовательности, заданной общим членом;

- вычислить производную данной функции;
- найти первообразную данной функции или значение неопределенного интеграла.

Другие возможности меню **Symbolics** (**Аналитические вычисления**) состоят в выполнении аналитических операций, ориентированных на переменную, использованную в выражении. Для этого надо выделить в выражении переменную и выбрать команду из меню **Symbolics > Variable** (**Аналитические вычисления > Переменная**). Команда **Solve** (**Решить**) ищет корни функции, заданной данным выражением, например, если выделить уголковым курсором переменную x в выражении $ax^2 + bx + c = 0$, то в результате применения команды **Symbolics > Variable > Solve** (**Аналитические вычисления > Переменная > Решить**) будут найдены все корни.

4.8. Использование системы Mathcad для решения инженерно-экономических задач

4.8.1. Решение линейных и нелинейных уравнений и систем

Решение уравнений – алгебраических или трансцендентных – представляет собой одну из существенных задач прикладного анализа, потребность, которая возникает в многочисленных и самых разнообразных разделах физики, механики, техники и естествознания в широком смысле этого слова.

Методы решения уравнений делятся на две группы. К первой группе относятся такие методы, которые позволяют путем различных преобразований упростить уравнение и записать решение в виде формулы. Вторую группу составляют численные методы. Решение в этом случае ищется с помощью операций (при желании только арифметических) над числами, причем ответ получается сразу в виде числа (обычно приближенного). К первой группе относится, например, метод Кардано для решения уравнений третьей степени, способ понижения степени возвратных уравнений и т. д. Эти методы могут быть иногда полезны, однако практическая их ценность в общем невелика, так как между формулой и численным ответом, необходимым, как правило, в практических задачах, лежит еще большое количество вычислений. Кроме того, эти методы применимы лишь к сравнительно узкому кругу задач.

В настоящее время существует и применяется большое количество численных методов решения уравнений и их систем. Это методы

итераций, Ньютона, Гаусса, Зейделя и др. Разнообразие методов свидетельствует о нетривиальности решения данной задачи и множестве практических сфер ее применения и, кроме того, связано со следующими обстоятельствами:

1) решение уравнения или системы уравнений редко осуществляется от начала до конца одним способом. Обычно вначале корни уравнений определяются каким-нибудь методом со сравнительно небольшой точностью, а затем «уточняются» с помощью других методов, достигая заданной точности;

2) не существует методов, одинаково эффективных для всех уравнений или систем уравнений. Естественно, что для одних уравнений более удобны одни методы, для других – другие. Это различие не всегда можно определить заранее. Поэтому в процессе решения приходится иногда отказываться от одного метода и переходить к другому;

3) решение уравнений численными методами требует довольно значительной вычислительной работы, что вызывает необходимость разработки алгоритма и написания программы на одном из языков программирования.

Система Mathcad решает линейные уравнения и системы уравнений (кстати, как и нелинейные) итерационным методом, при этом может использоваться подстановка одних переменных в другие, нередко сводящая задачу к точному решению.

Главное достоинство системы Mathcad в том, что при решении таких задач не требуется никаких записей в программах, кроме записи исходного уравнения. Естественно, это единственная «неприятность», от которой уже нельзя избавиться, – начальное приближение в случае использования численного метода решения и оператора для решения уравнения.

Пример 4.1. Найти корни уравнения $e^x - \cos x = 0$ численным методом.

В окне редактора Mathcad программа имеет вид

```
x:=0  
root(ex-cos(x),x)=0
```

Зная, что это уравнение имеет не один корень, зададим другое начальное приближение, тогда имеем

```
x:=-2  
root(ex-cos(x),x)=-1.292
```

Этим самым мы определим следующий корень исходного уравнения на интервале $(-2; -1)$.

Используя блок given – find, в окне редактора будем иметь

Начальное приближение

$x:=0$

Булевское выражение

given

$e^x - \cos(x) = 0$

Получение решения

find(x)=0

Пример 4.2. Найти корни уравнения $e^x - \cos x = 0$ аналитическим методом.

Для решения этой задачи в Mathcad имеются три дублирующих друг друга способа.

1. Выделив уголковым курсором переменную x в выражении $\exp(x) - \cos(x) = 0$ и применив команду **Symbolics** \triangleright **Variable** \triangleright **Solve** (**Символы** \triangleright **Переменная** \triangleright **Вычислить**), можно найти все корни.

2. Используя блок given – find, в окне редактора будем иметь

given

$e^x - \cos(x) = 0$

find(x) \rightarrow 0

3. Используя оператор root, в окне редактора будем иметь

root($e^x - \cos(x), x$) \rightarrow 0

В Mathcad есть эффективный способ вычисления корней уравнения для частного случая, когда функция $f(x)$ представляет собой полином (многочлен) n -го порядка:

$$f(x) = a_0 + a_1x + a_2x^2 + \dots + a_nx^n.$$

Известно, что уравнение, в котором левая часть представляет собой полином n -го порядка, имеет n корней. Среди них могут быть как вещественные, так и комплексные корни. Для решения уравнения $f(x) = 0$ с полиномиальной правой частью такого вида используется функция polyroots(v), где v – вектор коэффициентов полинома, записанных в последовательности от низшего (нулевого) к высшему.

Пример 4.3. Найти все корни полинома $f(x) = -6 + 11x - 6x^2 + x^3$.

С использованием функции polyroots имеем запись в окне редактора системы Mathcad:

$f(x) := -6 + 11 \cdot x - 6 \cdot x^2 + x^3$

$v := (-6 \ 11 \ -6 \ 1)^T$

`x:=polyroots(v)`

$$x = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$$

При решении уравнения получен один вещественный и два комплексно-сопряженных корня. Применение функции `polyroots` по сравнению с функцией `root` для уравнений с полиномиальной правой частью предпочтительнее по двум причинам:

- не требуется выполнения этапа отделения корней;
- функция позволяет найти как вещественные, так и комплексные корни.

При решении систем линейных и нелинейных уравнений Mathcad использует итерационные методы, причем часто решение систем линейных уравнений относят к некорректным задачам. Так случается, если задана система, у которой не хватает переменных (т. е. переопределена матрица A коэффициентов a_{ij}). Итерационный метод, реализованный в Mathcad, позволяет легко решать такие задачи, обычно требующие специальных методов. Однако попытки решения другой ненормальной системы (с вырожденной матрицей, $\det|A| = 0$) к успеху обычно не приводят.

На практике нет общих методов решения систем нелинейных уравнений, которые давали бы гарантию правильного решения или даже решения вообще. Нередки случаи, когда решение зацикливается и с заданной точностью получить его просто невозможно. Встречаются и случаи, когда из-за неудачного и произвольного выбора начальных значений переменных решение обречено на неудачу. Кроме того, заведомо неизвестна область существования корней, и некоторые корни при решении можно попросту проскочить. Система Mathcad это исключает, причем она позволяет решать системы нелинейных уравнений, содержащие ограничения, обычно выражаемые неравенствами.

Ограничения – еще одно мощное средство, позволяющее отбросить нереальные или неинтересующие нас решения.

Пример 4.4. Определить точку пересечения окружности с центром в точке $O(0; 0)$ и радиусом $R = 4$ с параболой $y = x^2$ в первой четверти декартовой системы координат.

Решение.

Эти кривые в силу симметрии параболы относительно оси Oy пересекаются в двух точках. Для определения одной из них нам необходимо решить систему уравнений

$$\begin{cases} x^2 + y^2 = 16, \\ y = x^2, \end{cases}$$

используя начальные приближения $x = 0$, $y = 0$, что удовлетворяет требованиям данной задачи. Листинг программы данной задачи имеет следующий вид:

```
x:=0
y:=0
Given
x^2+y^2=16
y=x^2
Find(x,y)=(1.879
           3.531)
```

К системам линейных уравнений сводится множество, если не сказать большинство, задач вычислительной математики, поэтому одним из основных вопросов вычислительной линейной алгебры является решение систем линейных алгебраических уравнений (СЛАУ), т. е. систем уравнений вида

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n &= b_1, \\ a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n &= b_2, \\ \dots \\ a_{m1}x_1 + a_{m2}x_2 + \dots + a_{mn}x_n &= b_m. \end{aligned}$$

В матричной форме СЛАУ записывается в эквивалентном виде:

$$A \cdot x = b,$$

где A – матрица коэффициентов СЛАУ размерности $m \times n$; x – вектор неизвестных; b – вектор правых частей уравнений.

СЛАУ имеет единственное решение, если матрица A является невырожденной, или, по-другому, несингулярной, т. е. ее определитель не равен нулю.

С вычислительной точки зрения, решение СЛАУ не представляет трудностей, если матрица A не очень велика. С большой матрицей проблем также не возникнет, если она не очень плохо обусловлена. В Mathcad СЛАУ можно решить как в более наглядной форме, используя вычислительный блок given – find, так и в более удобной для записи форме, используя встроенную функцию lsolve или матричную запись для решения системы линейных уравнений.

Пример 4.5. Решить систему уравнений

$$\begin{cases} 2x + 5y + 3z = 14, \\ 7x + 4y + 9z = 24, \\ 3x + 8y + 6z = 34. \end{cases}$$

Решение.

Решим эту СЛАУ, используя различные способы, и представим листинги решений в системе Mathcad.

Листинг 1. Решение СЛАУ с помощью блока given – find:

$$x:=0 \quad y:=0 \quad z:=0$$

Given

$$2 \cdot x + 5 \cdot y + 3 \cdot z = 14$$

$$7 \cdot x + 4 \cdot y + 9 \cdot z = 24$$

$$3 \cdot x + 8 \cdot y + 6 \cdot z = 34$$

$$\text{Find}(x,y,z) = \begin{pmatrix} 1.977 \\ 1.816 \\ 0.322 \end{pmatrix}$$

Листинг 2. Решение СЛАУ с помощью функции lsolve:

$$A := \begin{pmatrix} 2 & 5 & 3 \\ 7 & 4 & 9 \\ 3 & 8 & 6 \end{pmatrix} \quad b := \begin{pmatrix} 14 \\ 24 \\ 34 \end{pmatrix}$$

$$\text{lsolve}(A,b) = \begin{pmatrix} 1.977 \\ 1.816 \\ 0.322 \end{pmatrix}$$

Листинг 3. Решение СЛАУ матричным методом:

$$A := \begin{pmatrix} 2 & 5 & 3 \\ 7 & 4 & 9 \\ 3 & 8 & 6 \end{pmatrix} \quad b := \begin{pmatrix} 14 \\ 24 \\ 34 \end{pmatrix}$$

$$x := A^{-1} \cdot b$$

$$x = \begin{pmatrix} 1.977 \\ 1.816 \\ 0.322 \end{pmatrix}$$

Встроенную функцию lsolve допускается применять и при символьном решении СЛАУ (листинг 4).

Листинг 4. Символьное решение СЛАУ:

$$A := \begin{pmatrix} 2 & 5 & 3 \\ 7 & 4 & 9 \\ 3 & 8 & 6 \end{pmatrix} \quad b := \begin{pmatrix} 14 \\ 24 \\ 34 \end{pmatrix}$$

$$\text{Lsolve}(A,b) \rightarrow \begin{pmatrix} 1.977 \\ 1.816 \\ 0.322 \end{pmatrix}$$

4.8.2. Поиск экстремумов функций

Функция $y = f(x)$ называется возрастающей (убывающей) в некотором промежутке, если для любых точек x_1 и x_2 , принадлежащих данному промежутку, из неравенства $x_1 < x_2$ следует неравенство $f(x_1) < f(x_2)$. Знак первой производной указывает на возрастание или убывание функции на заданном промежутке. Если в некотором промежутке производная данной функции положительна, то функция возрастает в этом промежутке, если отрицательна – убывает на данном промежутке.

Максимумом (минимумом) функции $y = f(x)$ называется такое ее значение $y_i = f(x_i)$, которое больше (меньше) всех других ее значений, принимаемых в точках x , достаточно близких к точке x_i и отличных от нее.

Максимум и минимум называются экстремумом функции. Значения аргумента и функции, при которых достигается экстремум, – точки экстремума.

Достаточным условием экстремума являются следующие условия. Если в точке $x = x_0$ первая производная функции $y = f(x)$ равна нулю, а вторая производная отлична от нуля, то x_0 – точка экстремума, причем:

- 1) x_0 – точка максимума, если $f''(x_0) < 0$;
- 2) x_0 – точка минимума, если $f''(x_0) > 0$.

Чтобы найти наибольшее значение функции $y = f(x)$ на отрезке $[a; b]$, необходимо вычислить значения ее максимумов на этом отрезке, значения функции на ее концах, т. е. $f(a), f(b)$, и из полученных чисел выбрать самое большое. Аналогично находится наименьшее значение функции.

На практике часто необходимо найти экстремум (или экстремумы) некоторой целевой функции $F(x_1, x_2, \dots, x_n)$ n переменных x_i (проектных параметров). Такая функция описывает $(n + 1)$ -мерную поверхность. Соответственно функция $F(x)$ одного параметра $x_i = x$ описывает некоторую кривую на плоскости.

Поиск экстремумов функции одной переменной является самостоятельной и часто встречаемой задачей. Кроме того, к нему сво-

дится гораздо более сложная задача поиска экстремумов функций многих переменных.

В общем случае функция $F(x)$ может иметь несколько экстремумов (максимумов или минимумов). Из них главный (оптимальное решение для пространства проектирования) называется глобальным. Задача поиска экстремумов сводится к их локализации и уточнению значений x и $F(x)$ в точке экстремума. Будем также полагать, что на значения параметра x (если это особо не оговорено) накладываются ограничения в виде неравенств $a \leq x \leq b$, где a и b – границы интервала поиска. В пределах отрезка $[a; b]$ функцию считаем унимодальной, т. е. содержащей один экстремум.

Поиск экстремумов численными методами, как правило, разделяется на два этапа.

На первом этапе выделяются интервалы аргумента x , в которых существует единственная точка x^* , где функция $F(x)$ принимает экстремальное значение. Первый этап поиска решения задачи нахождения экстремума близок по идеологии к задаче отделения корней уравнений и не поддается строгой алгоритмизации. Обычно интервалы унимодальности находят на основе анализа упрощенных математических моделей исследуемых процессов. Графическое представление исследуемой функции также помогает на первом этапе, хотя и требует значительных временных затрат, поскольку шаг изменения аргумента должен быть достаточно малым, чтобы не пропустить возможные экстремумы.

На втором этапе осуществляется уточнение местоположений экстремумов на интервалах унимодальности функций с использованием различных методов (метод Фибоначчи, дихотомии и др.).

Для организации решения поиска максимума (минимума) функции одной переменной, используя систему Mathcad, достаточно задать саму функцию и определить так называемую целевую функцию с конкретным именем-параметром. Кроме того, необходимо задать установку maximize (имя-параметр) на поиск максимума целевой функции. Это указание запускает встроенную в систему стандартную программу поиска одного из экстремумов численным методом.

Для указания отрезка $[a; b]$ используется зарезервированное слово given.

Пример 4.6. Определить максимум функции $y(x) = e^{0.4x} \sin x$. Для реализации данной задачи программа на рабочем листе имеет вид:

$$\begin{aligned} f(x) &:= e^{(0.4 \cdot x)} \cdot \sin(x) \\ x &:= 2 \end{aligned}$$

$z := \text{maximize}(f, x)$
 $z = 1.951$

4.8.3. Решение задач линейного программирования

Математическое программирование представляет собой математическую дисциплину, занимающуюся изучением экстремальных задач и разработкой методов их решения.

В общем виде математическая постановка экстремальной задачи состоит в определении наибольшего или наименьшего значения целевой функции $f(x_1, x_2, \dots, x_n)$ при условиях $g_i(x_1, x_2, \dots, x_n) \leq b_i$ ($i = 1, \dots, m$), где f и g_i – заданные функции, а b_i – некоторые действительные числа.

Задачи математического программирования делятся на задачи линейного и нелинейного программирования. При этом если все функции f и g_i линейные, то соответствующая задача является задачей линейного программирования. Если же хотя бы одна из указанных функций нелинейная, то соответствующая задача является задачей нелинейного программирования.

Использование системы Mathcad позволяет последовательно находить решение задач, получающихся из исходной с помощью внесения изменений в исходные данные, а также производить построения различных целевых функций. Наряду с этим использование указанного пакета дает возможность получать отчеты, необходимые для проведения широкого послеоптимизационного анализа решения задач линейного и нелинейного программирования, а также осуществлять другие различные исследования, обусловленные рациональным подходом к решению задач математического программирования.

Задачи оптимизации. Найдем решение задачи, состоящей в определении максимального значения функции

$$F = c_1x_1 + c_2x_2 \quad (1)$$

при условиях

$$a_{i1}x_1 + a_{i2}x_2 \leq b_i, i = 1, \dots, m, \quad (2)$$

$$x_j \geq 0, j = (1; 2). \quad (3)$$

Каждое из неравенств (2), (3) системы ограничений задачи геометрически определяет полуплоскость соответственно с граничными прямыми $a_{i1}x_1 + a_{i2}x_2 = b_i$ ($i = 1, m$), $x_1 = 0$, $x_2 = 0$. Так как множество точек пересечения данных полуплоскостей выпуклое, то областью

допустимых решений задачи (1)–(3) является выпуклое множество, которое называется многоугольником решений.

Таким образом, исходная задача линейного программирования состоит в нахождении такой точки многоугольника решений, в которой целевая функция F принимает максимальное значение. Эта точка существует тогда, когда многоугольник решений не пуст и на нем целевая функция ограничена сверху.

При указанных условиях в одной из вершин многоугольника решений целевая функция принимает максимальное значение. Для определения данной вершины построим линию уровня $c_1x_1 + c_2x_2 = h$, где h – некоторая постоянная, проходящая через многоугольник решений, и будем передвигать ее в направлении вектора $C = (c_1; c_2)$ до тех пор, пока она не пройдет через последнюю ее общую точку с многоугольником решений. Координаты указанной точки и определяют оптимальный план данной задачи.

Итак, нахождение решения задачи линейного программирования (1)–(3) на основе ее геометрической интерпретации включает следующие этапы:

- строят прямые, уравнения которых получаются в результате замены в ограничениях (2) и (3) знаков неравенств на знаки точных равенств;
- находят полуплоскости, определяемые каждым из ограничений задачи;
- находят многоугольник решений;
- строят вектор $C = (c_1; c_2)$;
- строят прямую $c_1x_1 + c_2x_2 = h$, проходящую через многоугольник решений;
- передвигают прямую $c_1x_1 + c_2x_2 = h$ в направлении вектора C , в результате чего либо находят точку (точки), в которой целевая функция принимает максимальное значение, либо устанавливают неограниченность сверху функции на множестве планов;
- определяют координаты точки максимума функции и вычисляют значение целевой функции в этой точке.

Пример 4.7. Для производства двух видов изделий А и В лесхоз использует три вида сырья (доска, брус, бревно). Нормы расхода сырья каждого вида на изготовление единицы продукции данного вида, а также общее количество сырья каждого вида, которое может быть использовано лесхозом, приведены в табл. 4.1.

Учитывая, что изделия А и В могут производиться в любых соотношениях (сбыт обеспечен), требуется составить такой план их вы-

пуска, при котором прибыль лесхоза от реализации всех изделий является максимальной.

Таблица 4.1

Виды сырья	Нормы расхода сырья, м ³ , на одно изделие		Общее количество сырья, м ³
	А	В	
Доска	12	4	300
Брус	4	4	120
Бревно	3	12	252

Прибыль от реализации одного изделия А и В равна соответственно 30 и 40 усл. ед.

Решение.

Предположим, что лесхоз изготовит x_1 изделий вида А и x_2 изделий вида В. Поскольку производство продукции ограничено имеющимся в распоряжении лесхоза сырьем каждого вида и количество изготавливаемых изделий не может быть отрицательным, должны выполняться неравенства

$$12x_1 + 4x_2 \leq 300,$$

$$4x_1 + 4x_2 \leq 120,$$

$$3x_1 + 12x_2 \leq 252,$$

$$x_1, x_2 > 0.$$

Общая прибыль от реализации x_1 изделий вида А и x_2 изделий вида В составит $F = 30x_1 + 40x_2$.

Таким образом, мы приходим к следующей математической задаче: среди всех неотрицательных решений данной системы линейных неравенств требуется найти такое, при котором функция F принимает максимальное значение.

Найдем решение сформулированной задачи, используя ее геометрическую интерпретацию. Сначала определим многоугольник решений. Для этого в неравенствах системы ограничений и условиях неотрицательности переменных знаки неравенств заменим на знаки точных равенств и найдем соответствующие прямые:

$$12x_1 + 4x_2 = 300,$$

$$4x_1 + 4x_2 = 120,$$

$$3x_1 + 12x_2 = 252,$$

$$x_1 = 0,$$

$$x_2 = 0.$$

Эти прямые изображены на рис. 4.5. Каждая из построенных прямых делит плоскость на две полуплоскости. Координаты точек одной полуплоскости удовлетворяют исходному неравенству, а другой – нет.

Чтобы определить искомую полуплоскость, нужно взять какую-нибудь точку, принадлежащую одной из полуплоскостей, и проверить, удовлетворяют ли ее координаты данному неравенству. Если координаты взятой точки удовлетворяют данному неравенству, то искомой является та полуплоскость, которой принадлежит эта точка, в противном случае – другая полуплоскость.

Найдем, например, полуплоскость, определяемую неравенством $12x_1 + 4x_2 \leq 300$. Для этого построим прямую $12x_1 + 4x_2 = 300$, возьмем какую-нибудь точку, принадлежащую одной из двух полученных полуплоскостей, например точку $O(0; 0)$. Координаты этой точки удовлетворяют неравенству $12 \cdot 0 + 4 \cdot 0 < 300$; значит, полуплоскость, которой принадлежит точка $O(0; 0)$, определяется неравенством $12x_1 + 4x_2 \leq 300$. Это и показано стрелками на рис. 4.5.

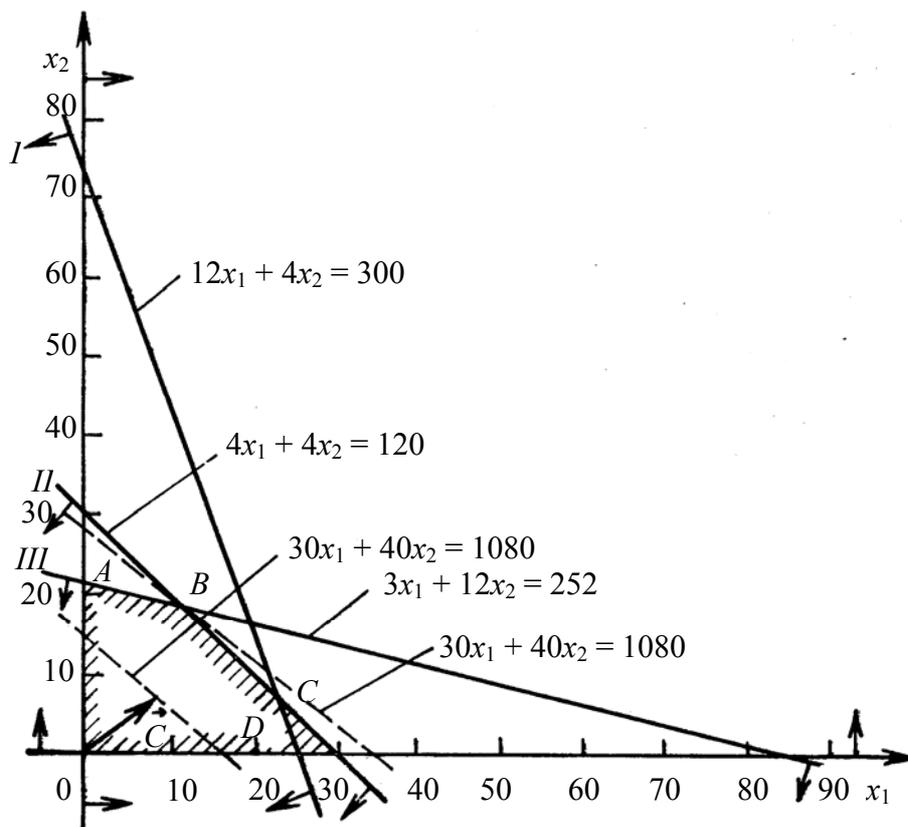


Рис. 4.5. Многоугольник решений

Пересечение полученных полуплоскостей и определяет многоугольник решений данной задачи

Как видно на рис. 4.5, многоугольником решений является пятиугольник $OABCD$. Координаты любой точки, принадлежащей этому

пятиугольнику, удовлетворяют данной системе неравенств и условию неотрицательности переменных. Поэтому сформулированная задача будет решена, если мы сможем найти точку, принадлежащую пятиугольнику $OABCD$, в которой функция F принимает максимальное значение. Чтобы найти указанную точку, построим вектор $C = (30; 40)$ и прямую $30x_1 + 40x_2 = h$, где h – некоторая постоянная, такая что прямая $30x_1 + 40x_2 = h$ имеет общие точки с построенным многоугольником решений. Положим, например, $h = 480$ и построим прямую $30x_1 + 40x_2 = 480$ (рис. 4.5).

Если теперь взять какую-нибудь точку, принадлежащую построенной прямой и многоугольнику решений, то ее координаты определяют такой план производства изделий А и В, при котором прибыль от их реализации равна 480 усл. ед. Далее, полагая h равным некоторому числу, большему 480, мы будем получать различные параллельные прямые. Если они имеют общие точки с многоугольником решений, то эти точки определяют планы производства изделий А и В, при которых прибыль от их реализации превзойдет 480 усл. ед.

Перемещая построенную прямую $30x_1 + 40x_2 = 480$ в направлении вектора C , видим, что последней общей точкой ее с многоугольником решений задачи служит точка B . Координаты этой точки и определяют план выпуска изделий А и В, при котором прибыль от их реализации является максимальной.

Найдем координаты точки B как точки пересечения двух прямых. Следовательно, ее координаты удовлетворяют уравнениям этих прямых:

$$\begin{aligned} 4x_1^* + 4x_2^* &= 120, \\ 3x_1^* + 12x_2^* &= 252. \end{aligned}$$

Решив эту систему уравнений, получим $x_1^* = 12$, $x_2^* = 18$.

Следовательно, если лесхоз изготовит 12 изделий вида А и 18 изделий вида В, то он получит максимальную прибыль, равную $F_{\max} = 30 \cdot 12 + 40 \cdot 18 = 1080$ усл. ед.

В настоящее время решение этих задач становится актуальным для признания рентабельности производства при изготовлении тех или иных изделий. Поэтому если целевая функция зависит от двух переменных, то, как видно из примера, эту задачу можно решить графическим путем, в противном случае задача линейного программирования, когда целевая функция F зависит от многих факторов и имеется множество ограничений, требует применения трудоемких вычислений.

Покажем на данном примере, сколь просто справляется Mathcad с решением этой довольно головоломной задачи. В рабочей области окна системы Mathcad программа имеет вид:

```

f(x1,x2):=30·x1+40·x2
x1:=10
x2:=15
Given
x1≥0
x2≥0
12·x1+4·x2≤300
4·x1+4·x2≤120
3·x1+12·x2≤252
R:=Maximize(f,x1,x2)
R= $\begin{pmatrix} 12 \\ 18 \end{pmatrix}$ 

```

Рассмотрим другие задачи линейного программирования, используемые в экономических и производственных расчетах.

Пример 4.8. Цех предприятия должен изготовить 100 изделий трех типов. Каждого изделия нужно не менее 20 шт. На изделия уходит соответственно 4, 5, и 2 кг однородного металла при его общем запасе 340 кг, а также по 5, 9 и 2 кг пластмассы при ее общем запасе 700 кг.

Сколько изделий каждого типа надо выпустить для получения максимального объема выпуска в денежном выражении, если цена изделий составляет по калькуляции 4, 3 и 2 усл. ед.?

Для наглядности листинга программы в окне редактора Mathcad введем комментарий в виде объекта текст. Для ввода текста следует выполнить команду **Вставка** **Текстовая область** или нажать клавишу двойной кавычки "<"> и установить вид и размер шрифта так же, как и в текстовом редакторе Word.

Данная производственная задача явно сводится к задаче вычисления максимума функции с ограничениями. С использованием функции maximize программа в окне редактора Mathcad имеет следующий вид:

```

Обозначим x1, x2 и x3 – количество изделий 1, 2 и 3-го вида.
Целевая функция:
f(x1,x2,x3):=4·x1+3·x2+2·x3
Произвольные начальные приближения:
x1:=1 x2:=1 x3:=1
Блок решений и ограничений:
Given
x1≥0
x2≥0

```

$$\begin{aligned}
&x_3 \geq 0 \\
&4 \cdot x_1 + 5 \cdot x_2 + 2 \cdot x_3 \leq 340 \\
&5 \cdot x_1 + 9 \cdot x_2 + 2 \cdot x_3 \leq 700 \\
&x_1 + x_2 + x_3 = 100 \\
&R := \text{maximize}(f, x_1, x_2, x_3)
\end{aligned}$$

$$R = \begin{pmatrix} 70 \\ 0 \\ 30 \end{pmatrix}$$

Максимальная прибыль:
 $f(R_0, R_1, R_2) = 340$

Данное решение $x_1 = 70$, $x_2 = 0$, $x_3 = 30$ позволит получить максимальную прибыль в денежном эквиваленте $f = 340$ усл. ед.

Пример 4.9. Лесхозу требуется не более 10 трехтонных и не более 8 пятитонных автомашин. Отпускная цена автомашин: первой марки – 3000 усл. ед., второй марки – 6000 усл. ед. Лесхоз может выделить для приобретения автомашин от 18 до 72 тыс. усл. ед.

Сколько следует приобрести автомашин каждой марки в отдельности, чтобы их суммарная грузоподъемность была максимальной? Определить максимальную суммарную грузоподъемность.

Данная производственная задача также сводится к задаче вычисления максимума функции с ограничениями. С использованием функции `maximize` программа в окне редактора Mathcad имеет следующий вид:

Обозначим x_1 , x_2 – количество трехтонных и пятитонных автомашин.

Целевая функция:

$$f(x_1, x_2) = 3 \cdot x_1 + 5 \cdot x_2$$

Произвольные начальные приближения:

$$x_1 := 1 \quad x_2 := 1$$

Блок решений и ограничений:

Given

$$10 \geq x_1 \geq 0$$

$$8 \geq x_2 \geq 0$$

$$3000 \cdot x_1 + 6000 \cdot x_2 \geq 18\ 000$$

$$3000 \cdot x_1 + 6000 \cdot x_2 \leq 72\ 000$$

$$R := \text{maximize}(f, x_1, x_2)$$

$$R = \begin{pmatrix} 10 \\ 7 \end{pmatrix}$$

Максимальная грузоподъемность:

$$f(R_0, R_1) = 65$$

Системе совершенно безразличны новые трудности, возникающие вследствие нелинейности целевой функции или ограничений.

Итерационные процедуры оптимизации системы Mathcad универсальны и позволяют решать как линейные, так и нелинейные задачи оптимизации.

Транспортная задача. Общая постановка транспортной задачи состоит в определении оптимального плана перевозок некоторого однородного груза из m пунктов отправления A_1, A_2, \dots, A_m в n пунктов назначения B_1, B_2, \dots, B_n . При этом в качестве критерия оптимальности обычно берется либо минимальная стоимость перевозок всего груза, либо минимальное время его доставки. Рассмотрим транспортную задачу, в качестве критерия оптимальности которой взята минимальная стоимость перевозок всего груза. Обозначим через c_{ij} тарифы перевозки единицы груза из i -го пункта отправления в j -й пункт назначения, через a_i – запасы груза в i -м пункте отправления, через b_j – потребности в грузе в j -м пункте назначения, а через x_{ij} – количество единиц груза, перевозимого из i -го пункта отправления в j -й пункт назначения. Тогда математическая постановка задачи состоит в определении минимального значения функции:

$$F = \sum_{i=1}^m \sum_{j=1}^n c_{ij} x_{ij}$$

при условиях

$$\sum_{i=1}^m x_{ij} = b_j \quad (j = 1 \dots n),$$

$$\sum_{j=1}^n x_{ij} = a_i \quad (i = 1 \dots m),$$

$$x_{ij} \geq 0.$$

Поскольку переменные x_{ij} удовлетворяют записанным системам линейных уравнений и условию неотрицательности, обеспечиваются доставка необходимого количества груза в каждый из пунктов назначения, вывоз имеющегося груза из всех пунктов отправления, а также исключаются обратные перевозки. Всякое неотрицательное решение этих систем линейных уравнений, определяемое матрицей $X = x_{ij}$, называется планом транспортной задачи.

План $X^* = x_{ij}^*$, при котором целевая функция F принимает свое минимальное значение, называется оптимальным планом транспортной задачи.

Обычно исходные данные транспортной задачи записывают в виде табл. 4.2.

Таблица 4.2

Пункты от- правления	Пункты назначения					Запасы
	B_1	...	B_j	...	B_n	
A_1	c_{11}	...	c_{1j}	...	c_{1n}	a_1
...
A_i	c_{i1}	...	c_{ij}	...	c_{in}	a_i
...
A_m	c_{m1}	...	c_{mj}	...	c_{mn}	a_m
Потребности	b_1	...	b_j	...	b_n	–

Очевидно, общее наличие груза у поставщиков равно $\sum_{i=1}^m a_i$, а общая потребность в грузе в пунктах назначения составляет $\sum_{j=1}^n b_j$.

Если общая потребность в грузе в пунктах назначения равна запасу груза в пунктах отправления, т. е.

$$\sum_{i=1}^m a_i = \sum_{j=1}^n b_j,$$

то модель такой транспортной задачи называется закрытой. Если же указанное условие не выполняется, то модель транспортной задачи называется открытой.

Число переменных x_{ij} в транспортной задаче с m пунктами отправления и n пунктами назначения равно $n \cdot m$, а число уравнений в ограничениях составляет $n + m$. Следовательно, опорный план транспортной задачи может иметь не более $n + m - 1$ отличных от нуля неизвестных.

Если в опорном плане число отличных от нуля компонент равно в точности $n + m - 1$, то план является невырожденным, а если меньше – то вырожденным.

Рассмотрим простую закрытую транспортную задачу с небольшим числом неизвестных: $m = 2$, $n = 3$. Исходные данные закрытой транспортной задачи представим в виде табл. 4.3.

Таблица 4.3

Пункты отправления	Пункты назначения			Запасы
	B_1	B_2	B_3	
A_1	45	46	76	400
A_2	43	57	54	500
Потребности	400	300	200	–

Для набора листинга решения исходной задачи необходимы две палитры – **Булево** и **Вычисления**, показанные на рис. 4.6.

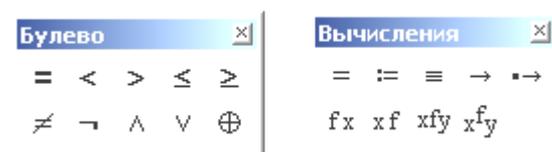


Рис. 4.6. Палитры **Булево** и **Вычисления**

Запись транспортной задачи в окне редактора Mathcad имеет следующий вид:

Тарифы – стоимости перевозки единицы продукта от первого поставщика (A_1) потребителям 1, 2, 3 (B_1, B_2, B_3):

$$c_{11}:=45 \quad c_{12}:=46 \quad c_{13}:=76$$

Тарифы – стоимости перевозки единицы продукта от второго поставщика (A_2) потребителям 1, 2, 3 (B_1, B_2, B_3):

$$c_{21}:=43 \quad c_{22}:=57 \quad c_{23}:=54$$

Количество продукта у 1-го и 2-го поставщика:

$$a_1:=400 \quad a_2:=500$$

Потребности потребителей 1, 2, 3-го:

$$b_1:=400 \quad b_2:=300 \quad b_3:=200$$

Общие затраты на перевозку (целевая функция):

$$f(x_{11}, x_{12}, x_{13}, x_{21}, x_{22}, x_{23}) := c_{11} \cdot x_{11} + c_{12} \cdot x_{12} + c_{13} \cdot x_{13} + c_{21} \cdot x_{21} + c_{22} \cdot x_{22} + c_{23} \cdot x_{23}$$

Начальные приближения:

$$x_{11}:=400 \quad x_{12}:=0 \quad x_{13}:=0 \quad x_{21}:=0 \quad x_{22}:=300 \quad x_{23}:=200$$

Блок ограничений:

Given

Условие равенства вывозимого продукта запасам у 1-го и 2-го поставщиков:

$$x_{11} + x_{12} + x_{13} = a_1$$

$$x_{21} + x_{22} + x_{23} = a_2$$

Условие равенства доставленного продукта потребностям 1, 2 и 3-го потребителей:

$$x_{11} + x_{21} = b_1$$

$$x_{12} + x_{22} = b_2$$

$$x_{13} + x_{23} = b_3$$

Физический смысл транспортной задачи:

$$x_{11} \geq 0 \quad x_{12} \geq 0 \quad x_{13} \geq 0$$

$$x_{21} \geq 0 \quad x_{22} \geq 0 \quad x_{23} \geq 0$$

Минимизация:

$$c := \text{Minimize}(f, x_{11}, x_{12}, x_{13}, x_{21}, x_{22}, x_{23})$$

$$x_{11}:=c_0 \quad x_{12}:=c_1 \quad x_{13}:=c_2 \quad x_{21}:=c_3 \quad x_{22}:=c_4 \quad x_{23}:=c_5$$

Минимальные затраты на перевозку:

$$D:=(c_0, c_1, c_2, c_3, c_4, c_5)$$

$$D=42000$$

Имеющихся текстовых пояснений в документах Mathcad достаточно, чтобы можно было легко решить и другие подобные задачи. Для решения необходимо лишь изменить числовые значения в данных документах.

При решении транспортных задач с большим числом переменных целесообразно использовать матрицы. В них с помощью палитры матричных операций, показанной на рис. 4.7, размещаются тарифы перевозок, начальные приближения для объемов перевозок, запасы поставщиков и потребности потребителей.

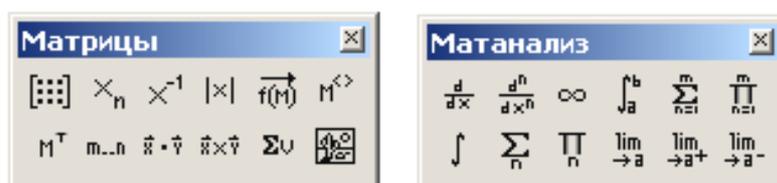


Рис. 4.7. Панели **Матрицы** и **Матанализ**

Целевая функция записывается с помощью шаблона суммы (панель **Матанализ**). В качестве примера рассмотрим задачу, исходные данные которой представлены в виде табл. 4.4.

Таблица 4.4

Пункты отправления	Пункты назначения				Запасы
	B ₁	B ₂	B ₃	B ₄	
A ₁	2	3	5	4	30
A ₂	3	2	4	1	40
A ₃	4	3	2	6	20
Потребности	20	25	35	10	90

Решение закрытой транспортной задачи с тремя поставщиками и четырьмя потребителями имеет следующий вид:

Тарифы – стоимости перевозки единицы продукта от трех поставщиков (A_i) четырем потребителям (B_j):

$$c := \begin{pmatrix} 2 & 3 & 5 & 4 \\ 3 & 2 & 4 & 1 \\ 4 & 3 & 2 & 6 \end{pmatrix}$$

Количество продукта у поставщиков:

$$a := \begin{pmatrix} 30 \\ 40 \\ 20 \end{pmatrix}$$

Потребности потребителей:

$$b := \begin{pmatrix} 20 \\ 25 \\ 35 \\ 10 \end{pmatrix}$$

Общие затраты на перевозку (целевая функция):

$$f(x) := \sum_{i=0}^2 \sum_{k=0}^3 c_{i,k} \cdot x_{i,k}$$

Начальное приближение:

$$x := \begin{pmatrix} 20 & 10 & 0 & 0 \\ 0 & 15 & 25 & 0 \\ 0 & 0 & 10 & 10 \end{pmatrix}$$

Вспомогательная матрица:

$$i:=0..2 \quad k:=0..3 \quad e_{k,i}:=1$$

$$e = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

Given

Ограничения:

равенство вывозимого продукта запасам поставщиков:

$$(x \cdot e)^{<0>} = a$$

равенство вывозимого продукта равно потребностям потребителей:

$$((e \cdot x)^T)^{<0>} = b$$

Условие неотрицательности объема поставок:

$$x \geq 0$$

Минимизация:

$$y := \text{Minimize}(f, x)$$

Ответ:

$$y = \begin{pmatrix} 20 & 0 & 10 & 0 \\ 0 & 25 & 5 & 10 \\ 0 & 0 & 20 & 0 \end{pmatrix}$$

Минимальные затраты на перевозку:
 $f(y)=210$

Для перехода к решению открытых транспортных задач с объемом запасов у поставщиков, не равным потребностям потребителей, в документе делаются соответствующие изменения, как показано на примерах в прил. 2.

Пример 4.10. Для строительства четырех дорог используется гравий из трех карьеров. Запасы гравия в каждом из карьеров соответственно равны 120, 280 и 160 усл. ед. Потребности в гравии для строительства каждой из дорог соответственно равны 130, 220, 60 и 70 усл. ед. Известны также тарифы перевозок 1 усл. ед. гравия из каждого карьера к каждой строящейся дороге, которые задаются матрицей. Составить такой план перевозок гравия, при котором потребности в нем каждой из строящихся дорог были бы удовлетворены при наименьшей общей стоимости перевозок.

Решение.

Исходные данные задачи сведем в таблицу (табл. 4.5).

Таблица 4.5

Пункты от- правления	Пункты назначения				Запасы
	B_1	B_2	B_3	B_4	
A_1	1	7	9	5	120
A_2	4	2	6	8	280
A_3	3	8	1	2	160
Потребности	130	220	60	70	–

Как видно из табл. 4.5, запасы гравия в карьерах A_1 , A_2 и A_3 ($120 + 280 + 160 = 560$) больше, чем потребности в нем на строящихся дорогах ($130 + 220 + 60 + 70 = 480$). Следовательно, модель исходной транспортной задачи является открытой.

Решение открытой транспортной задачи с тремя поставщиками и четырьмя потребителями имеет следующий вид:

Тарифы – стоимости перевозки единицы продукта от трех поставщиков (A_i) четырем потребителям (B_j):

$$c := \begin{pmatrix} 1 & 7 & 9 & 5 \\ 4 & 2 & 6 & 8 \\ 3 & 8 & 1 & 2 \end{pmatrix}$$

Количество продукта у поставщиков:

$$a := \begin{pmatrix} 120 \\ 280 \\ 160 \end{pmatrix}$$

Потребности потребителей:

$$b := \begin{pmatrix} 130 \\ 220 \\ 60 \\ 70 \end{pmatrix}$$

Общие затраты на перевозку (целевая функция):

$$f(x) := \sum_{i=0}^2 \sum_{k=0}^3 c_{i,k} \cdot x_{i,k}$$

Начальное приближение:

$$x := \begin{pmatrix} 100 & 0 & 0 & 0 \\ 0 & 200 & 0 & 0 \\ 0 & 0 & 40 & 20 \end{pmatrix}$$

Вспомогательная матрица:

$$i:=0..2 \quad k:=0..3 \quad e_{i,k}:=1$$

$$e = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

Given

Ограничения:

Неравенство вывозимого продукта запасам поставщиков:

$$(x \cdot e)^{<0>} \leq a$$

Неравенство вывозимого продукта потребностям потребителей:

$$((e \cdot x)^T)^{<0>} \geq b$$

Условие неотрицательности объема поставок:

$$x \geq 0$$

Минимизация:

$$y := \text{Minimize}(f, x)$$

Ответ:

$$y = \begin{pmatrix} 120 & 0 & 0 & 0 \\ 0 & 220 & 0 & 0 \\ 10 & 0 & 60 & 70 \end{pmatrix}$$

Минимальные затраты на перевозку:

$$f(y) = 790$$

Как видно из листинга программы, исходная задача имеет оптимальный план, при котором остается неиспользованным 60 усл. ед. гравия во втором карьере и 20 усл. ед. в третьем, а общая минимальная стоимость перевозок составляет 790 усл. ед.

4.8.4. Решение задач интерполирования и регрессионного анализа

Одной из важнейших задач в процессе математического моделирования является вычисление значений функций, входящих в математическое описание модели. Для сложных моделей подобные вычисления могут быть трудоемкими даже при использовании ПК.

Используемые в математических моделях функции задаются как аналитическим способом, так и табличным, при котором функции известны только при дискретных значениях аргументов.

Пусть известные значения некоторой функции $f(x)$ образуют табл. 4.6.

Таблица 4.6

x	x_0	x_1	...	x_n
$f(x)$	y_0	y_1	...	y_n

При этом требуется получить значение функции $f(x)$ для такого значения аргумента x , которое входит в отрезок $[x_0; x_n]$, но не совпадает ни с одним из значений x_i ($i = 0, 1, \dots, n$).

Классический подход к решению задачи построения аппроксимирующей (от латинского *approximo* – приближаюсь) функции $\varphi(x)$ основывается на требовании строгого совпадения значений $f(x)$ и $\varphi(x)$ в точках x_i . В этом случае нахождение приближенной функции называют интерполяцией, а точки x_i – узлами интерполяции.

Линейная интерполяция. Самый простой вид интерполяции – линейная, которая представляет искомую зависимость $\varphi(x)$ в виде ломаной линии. Интерполирующая функция $\varphi(x)$ состоит из отрезков прямых, соединяющих точки (рис. 4.8).

Для построения линейной интерполяции служит встроенная функция `interp`.

Пример 4.11. Пусть имеются точки $A_1(1; 2)$, $A_2(2; 2.5)$, $A_3(3; 4.5)$, $A_4(4; 5)$, $A_5(5; 6.8)$. Надо вычислить значения этой функции в требуемых точках, например $\varphi(2.5)$ и $\varphi(4.58)$.

Листинг программы по решению этой задачи следующий:

```
x:=(1 2 3 4 5)T  
y:=(2 2.5 4.5 5 6.8)T  
f(t):=linterp(x,y,t)
```

Как видно из листинга, чтобы осуществить линейную интерполяцию, надо выполнить следующие действия:

- ввести векторы данных x и y (первые две строки листинга);
- определить функцию $\text{linterp}(x, y, t)$;
- вычислить значения этой функции в требуемых точках $f(2.5) = 3.5$ и $f(4.58) = 6.044$;
- построить график, как показано на рис. 4.8.

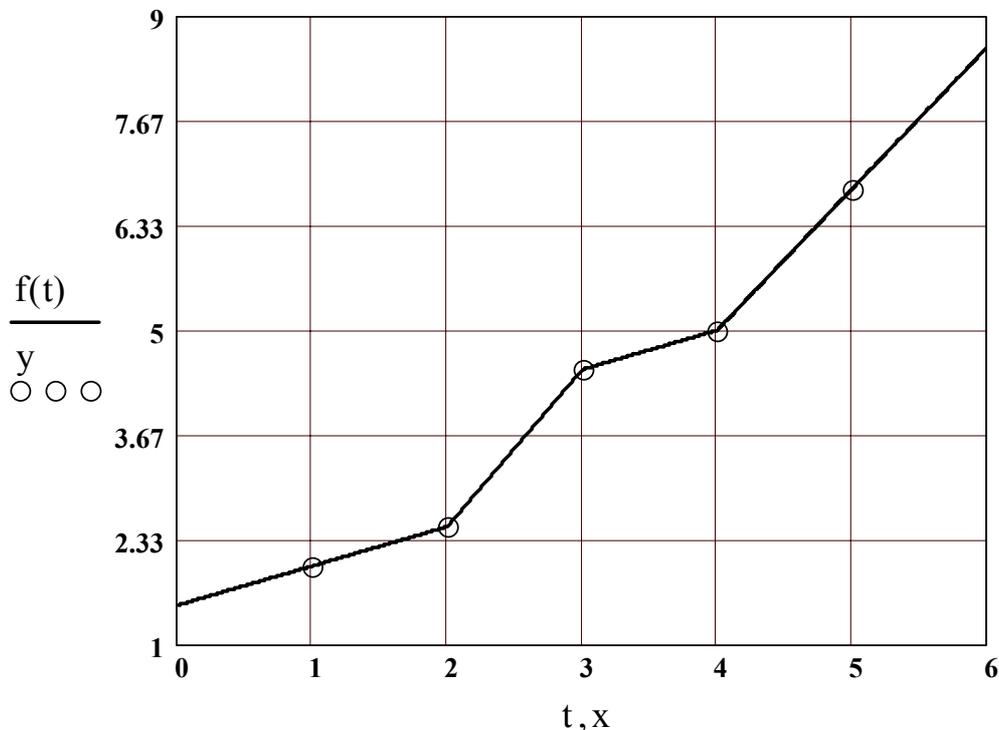


Рис. 4.8. Линейная интерполяция

Линейная регрессия. Совпадение значений в узлах иногда может вовсе не означать совпадения характеров поведения исходной и интерполирующей функций. Требование неукоснительного совпадения значений в узлах выглядит тем более не оправданным, если значения функции $f(x)$ получены в результате измерений и являются сомнительными.

Поставим сейчас задачу так, чтобы с самого начала обязательно учитывался характер исходной функции: найти функцию заданного

вида $y = \varphi(x)$, которая в точках x_1, x_2, \dots, x_n принимает значения, как можно более близкие к табличным значениям y_1, y_2, \dots, y_n .

Практически вид приближенной функции $\varphi(x)$ можно определить следующим образом. По табл. 4.6 строится точечный график функции $f(x)$, а затем проводится плавная кривая, по возможности, наилучшим образом отражающая характер расположения точек. По полученной таким образом кривой устанавливается вид приближающей функции.

Следует заметить, что строгая функциональная зависимость для экспериментально полученной табл. 4.6 наблюдается редко, ибо каждая из участвующих в ней величин $y = \varphi(x)$ (ее называют эмпирической зависимостью или уравнением регрессии y на x) интересна тем, что позволяет находить значение функции $f(x)$ для нетабличных значений x , «сглаживая» результаты измерений величины y . Воспользовавшись метрикой евклидова пространства, приходим к требованию, что сумма

$$s = (f(x_1) - \varphi(x_1))^2 + (f(x_2) - \varphi(x_2))^2 + \dots + (f(x_n) - \varphi(x_n))^2$$

должна быть наименьшей.

Итак, задача приближения функции f теперь формулируется следующим образом: для функции f , заданной табл. 4.6, найти функцию $\varphi(x)$ определенного вида так, чтобы сумма квадратов была наименьшей. Эта задача носит название аппроксимации функции методом наименьших квадратов (МНК). Не вдаваясь в его тонкости (см. [7]), отметим главное – этот метод позволяет так подобрать некоторые параметры аппроксимирующей зависимости, что она описывает исходные данные с минимальной среднеквадратической погрешностью. Система Mathcad хороша тем, что при работе с ней вполне достаточно помнить лишь об этом фундаментальном определении МНК, а все остальное Mathcad автоматически сделает сама, используя соответствующие процедуры.

Рассмотрим простую задачу на линейную регрессию.

Пример 4.12. Пусть имеется ряд точек $A_1(1; 2)$, $A_2(2; 2.9)$, $A_3(3; 4.05)$, $A_4(4; 5)$, $A_5(5; 6.1)$. Надо подобрать коэффициенты a_0 и a_1 линейной зависимости $y(x) = a_0 + a_1 \cdot x$ такими, чтобы прямая прошла в «облаке» точек с наименьшим общим среднеквадратическим отклонением от них.

Для решения этой задачи (линейной регрессии) в Mathcad имеются два дублирующих друг друга способа. Правила их применения представлены в следующих листингах.

Листинг 1. Линейная регрессия.

$$x := \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{pmatrix} \quad y := \begin{pmatrix} 2 \\ 2.9 \\ 4.05 \\ 5 \\ 6.1 \end{pmatrix}$$

a0:=intercept(x,y)

a1:=slope(x,y)

a0=0.92

a1=1.03

z(x):=a0+a1·x

Листинг 2. Линейная регрессия.

x:=(1 2 3 4 5)^T

y:=(2 2.9 4.05 5 6.1)^T

Line(x,y)= $\begin{pmatrix} 0.92 \\ 1.03 \end{pmatrix}$

z(x):=Line(x,y)₀+Line(x,y)₁·x

Представим эту задачу графическим способом, задав полученную эмпирическую зависимость $z(x)$ в виде линии, а координаты вектора y соответствующими точками (рис. 4.9).

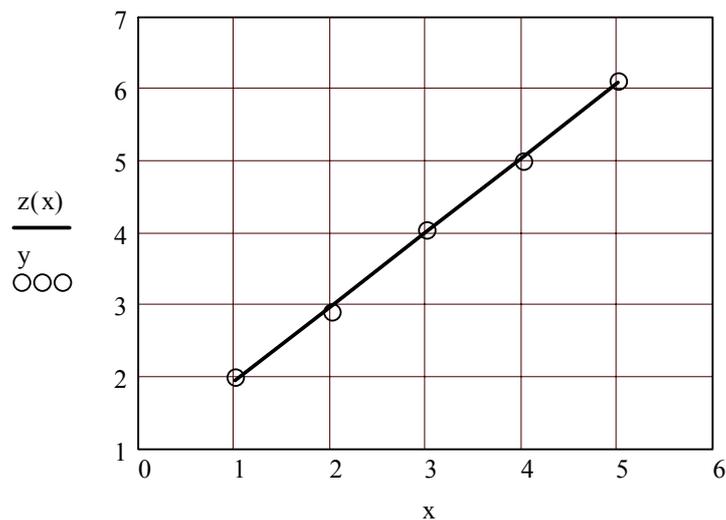


Рис. 4.9. Графики зависимости $z(x)$ и вектора y

Анализируя представленные графики, можно утверждать о правильно выбранной математической модели (уравнения регрессии) исследуемого процесса.

5. СЕТЕВЫЕ ТЕХНОЛОГИИ

5.1. Компьютерные сети

Локальная вычислительная сеть (ЛВС, LAN – Local Area Network) – это совокупность аппаратного и программного обеспечения, позволяющего объединить компьютеры в единую распределенную систему обработки и хранения информации. К аппаратному обеспечению можно отнести компьютеры с установленными на них сетевыми адаптерами, повторители, концентраторы, коммутаторы, мосты, маршрутизаторы и др., соединенные между собой сетевыми кабелями. Программное обеспечение – это сетевые операционные системы и протоколы передачи информации. Расстояние между компьютерами, объединяемыми в ЛВС, обычно не превышает нескольких километров (термин «локальные сети»), что связано с затуханием электрического сигнала в кабелях.

Протокол – это набор правил, в соответствии с которым компьютеры обмениваются информацией. Эти правила включают формат, время и последовательность передачи данных, способы контроля и коррекции ошибок.

5.1.1. История Internet

Глобальная компьютерная сеть Internet начиналась с сети ARPAnet – оборонного проекта, который финансировался Агентством перспективных исследований Министерства обороны США (Advanced Research Projects Agency, ARPA). Целью проекта являлась разработка компьютерной сети, призванной обеспечить устойчивое функционирование системы управления страной в условиях ядерной войны. В модели ARPAnet предполагалось, что любая часть сети может исчезнуть в любой момент. Несмотря на это сеть должна продолжать работать (насколько это возможно). Первые документы, описывающие технические требования к системе, появились в 1964 г., а в 1969 г. четыре компьютера были объединены в реально действующую сеть ARPAnet. В 1971 г. сеть насчитывала уже 14 компьютеров, а в 1972 г. – 37.

Непосредственно сама сеть Internet появилась как результат большой компьютерной программы Национального научного фонда США (National Science Foundation, NSF). Для проведения науч-

ных исследований NSF организовал по всей стране несколько центров вычислений и оснастил их суперкомпьютерами, подключив к центрам вычислений американские университеты и объединив все компьютеры в единую глобальную сеть. Первоначально планировалось использовать для этих целей ARPAnet, но администрация министерства обороны не разрешила подключение американских университетов к оборонной сети. В результате NSF создал свою собственную сеть NSFnet. В качестве основы этой сети были выбраны протоколы TCP/IP, разработанные в рамках проекта ARPAnet. Впоследствии к NSFnet присоединились еще порядка нескольких сотен различных сетей. Общим для всех этих сетей являлся тот факт, что для обмена информацией между собой они использовали единый механизм – семейство протоколов TCP/IP. Таким образом и зародился Internet – глобальная сеть, объединяющая локальные сети на основании протокола TCP/IP.

Первоначально Internet существовал как некоммерческая сеть, которая использовалась для обмена результатами научных исследований. Практически все лаборатории мира, имеющие доступ к сети, стали размещать свои публикации в электронном виде в архивах Internet, а уже только после этого выпускать печатные копии работ. До 1989 г. Internet оставалась некоммерческой сетью, и к ней подключались исключительно государственные и академические сети. В 1989 г. к Internet подключилась первая коммерческая сеть – MCI mail. Тогда же в мире Internet произошла еще одна революция: Тим Бернерс-Ли (Tim Berners-Lee) разработал язык гипертекстовой разметки (HTML), что привело к созданию в Internet нового сервиса – сети World Wide Web («всемирная паутина», WWW). Фактически большинство начинающих пользователей, перемещаясь по Internet (serfing), редко пользуются чем-то большим, чем гипертекстовые HTML-страницы, поэтому для них сеть WWW и есть Internet.

5.1.2. Протоколы прикладного уровня

В соответствии с архитектурой «клиент – сервер» программа делится на две части (одна работает на сервере, вторая – на компьютере пользователя), функционирующие как единое целое. Протоколы прикладного уровня описывают взаимодействие клиентской и серверной частей программы. Выделяют следующие наиболее известные прикладные протоколы:

1. HTTP (Hyper Text Transfer Protocol) – протокол передачи гипертекста. Используется в WWW для передачи гипертекстовых HTML-страниц. При работе по этому протоколу каждый элемент HTML-страницы загружается отдельно, причем соединение между загрузками прерывается и никакой информации о соединении не сохраняется. Это сделано для того, чтобы каждый пользователь Web-страниц получал «по чуть-чуть, в порядке общей очереди». В противном случае могла бы создаться ситуация, когда один человек качает страницу с большим количеством рисунков высокого разрешения, а все остальные ждут пока он это закончит.

2. FTP (File Transfer Protocol) – протокол передачи файлов. Предназначен для копирования файлов между компьютерами. Полностью занимает канал, пока не будет получен файл, сохраняет информацию о соединении. При сбое возможна докачка с того места, где произошел сбой.

3. SMTP, IMAP-4, POP3 – почтовые протоколы (электронная почта). Отличие: SMTP – протокол, рассчитанный на доставку почты до конкретного получателя, POP3 и IMAP-4 – протоколы взаимодействия пользователя со своим почтовым ящиком на сервере. При использовании SMTP предполагается, что почтовый адрес указывает на компьютер конечного получателя и что на этом компьютере запущена специальная программа, которая принимает и обрабатывает почту. Однако чаще всего бывает, что почта не доставляется на компьютер каждого отдельного пользователя, а обрабатывается централизованно, на отдельном почтовом сервере. В таком случае каждый пользователь имеет на почтовом сервере свой почтовый ящик. Почта доставляется до сервера по протоколу SMTP (конечный получатель – сервер) и помещается в почтовые ящики пользователей. Затем пользователи подключаются к своим почтовым ящикам по протоколу POP3 или IMAP-4 и забирают почту. Протокол POP3 требует полностью скачать себе всю почту, а затем разбираться, нужна она вам была или нет. Протокол IMAP-4 позволяет просматривать на сервере заголовки писем (указывается статус письма: новое, отвеченное и т. п.) и скачивать с сервера только необходимые письма или даже часть некоторого письма. Также можно на стороне сервера проводить поиск по сообщениям, создавать иерархию каталогов для хранения полученных писем (копии скачанных писем остаются на сервере, пока вы их не удалите). Фактически IMAP-4 дублирует функции почтовых программ пользователя (например, Microsoft Outlook), однако существенным

отличием здесь является то, что если Microsoft Outlook работает на компьютере пользователя, то команды протокола IMAP-4 выполняются на сервере, а значит каталоги с письмами хранятся в одном месте (на сервере), что очень удобно если вы часто подключаетесь к серверу с разных компьютеров и не хотите на каждом компьютере иметь полную копию всех писем.

5.1.3. Web-браузеры

Как уже говорилось выше, большую часть пользователей в Internet интересуют гипертекстовые HTML-страницы, которые позволяют представить информацию в виде документов с перекрестными гиперссылками и привлекательным графическим оформлением. Для просмотра гипертекстовых страниц применяются специальные программы – Web-браузеры. На сегодняшний день существует большое количество Web-браузеров, но самыми популярными являются Internet Explorer (входит в состав ОС Windows) и Netscape Navigator. Распространение также получил браузер Opera. Остальные браузеры занимают незначительную долю рынка (менее 1%). Несмотря на то что все браузеры предназначены для одного и того же – просмотра HTML-страниц, между ними имеются различия: страницы в Internet Explorer и Netscape Navigator выглядят по-разному, хотя общая структура страницы сохраняется. Особенно большие проблемы возникают при использовании в HTML-страницах программ, написанных на языке JavaScript, так как объектные модели Internet Explorer и Netscape Navigator различаются. Фактически приходится создавать два варианта страниц: один – для Internet Explorer, второй – для Netscape Navigator. Программа Internet Explorer предназначена для просмотра Web-страниц. Она поддерживает все новые возможности, закладываемые разработчиками Web-страниц, включая звуковое и видеосопровождение отображаемой информации. В комплект Internet Explorer кроме обозревателя Internet Explorer входят также другие программы, в частности Outlook Express, предназначенная для работы с электронной почтой.

После запуска программы на экране появится окно навигатора Internet Explorer (рис. 5.1).

Заголовок окна – стандартный заголовок Windows, в котором кроме названия программы отображается еще и название открытой Web-страницы.

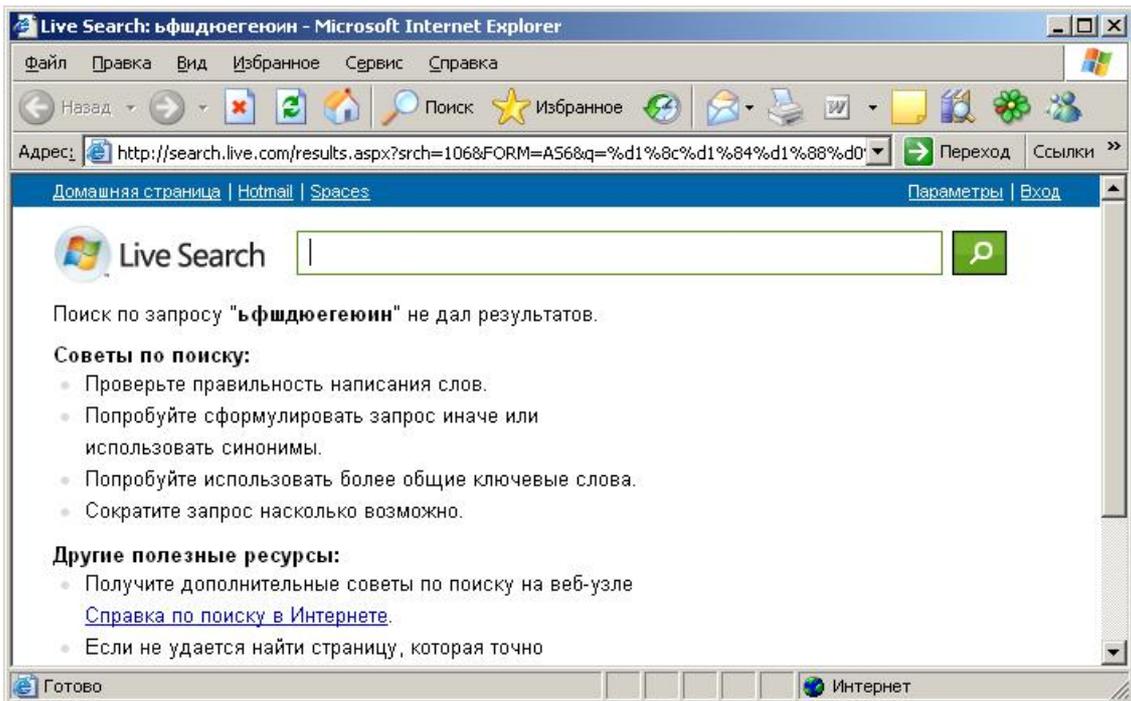


Рис. 5.1. Окно навигатора Internet Explorer

Под заголовком располагается меню. С его помощью вы можете выбрать любую команду Internet Explorer. На панели инструментов расположены значки, обозначающие различные действия, которые можно выполнить в процессе работы:

 ,  – **Назад** и **Вперед** – позволяют перемещаться по просмотренным документам;

 – **Остановить** – прерывает загрузку документа. Остановить загрузку документа можно также, нажав клавишу <Esc>;

 – **Обновить** – дает возможность пользователю повторить получение открытого документа;

 – **Домой** – возвращает пользователя на Web-страницу, зарегистрированную как стартовая при запуске навигатора;

 – **Поиск** – открывает панель для поиска нужной информации в Интернет;

 – **Избранное** – дает возможность перейти к списку адресов, созданных пользователем;

 – **Журнал** – позволяет открыть список ссылок на те страницы, которые были просмотрены ранее, и быстро перейти на одну из них.

Под панелью инструментов расположено поле для ввода адреса страницы, которую вы хотите просмотреть. Вместо поля для ввода адреса можно отобразить панель с несколькими ссылками на Web-страницы. Ссылка – это значок, за которым закреплен адрес какой-либо Web-страницы в сети Интернет. Для того чтобы отобразить панель ссылок, щелкните указателем мыши на поле **Ссылки**. Вновь отобразить панель с полем для ввода адреса можно, щелкнув указателем мыши на поле **Адрес**.

Строка состояния предназначена для индикации тех действий, которые в данный момент выполняет программа Internet Explorer. Профессиональная работа с Internet Explorer обязательно включает в себя умение разбираться в надписях, появляющихся на этой строке. В процессе работы с Интернет в строку состояния периодически выводятся сообщения об адресах источников информации, режиме ожидания, готовности запрошенного документа и ряд других полезных сведений.

5.1.4. Адресация в Интернет

Интернет-адрес, или так называемый указатель **URL** (Uniform Resource Locator), – это адрес сетевого ресурса. В общем случае URL содержит:

- информацию о сетевом протоколе;
- адрес хоста (доменное имя);
- адрес файла (имя файла).

URL хранит всю информацию, необходимую для нахождения файла, в следующем порядке: информация о сетевом протоколе, Интернет-адрес компьютера (доменное имя), нужная папка и затем название документа. Информацию о сетевом протоколе чаще всего опускают, автоматически подразумевая HTTP.

Доменные имена удобны для запоминания, их в основном используют, сообщая друг другу адрес того или иного сайта. Однако все компьютеры, подключенные к глобальной сети, идентифицируются по собственному уникальному номеру, называемому **IP-адрес** (например: 234.208.12.129).

Для преобразования доменного имени в числовой IP-адрес существует **Служба доменных имен** (Domain Name Service, DNS). Компьютеры, выполняющие такое преобразование, называются **DNS-серверами**.

Так, например, когда вы вводите в поле адрес www.bstu.unibel.by, DNS-сервер преобразует его в 213.59.0.214. Можно использовать числовые IP-адреса, но это менее удобно.

Уникальность доменного имени гарантирует Всемирная служба доменных имен, которая, в свою очередь, делегирует свои полномочия соответствующим национальным службам. В Беларуси это Координационный центр национального домена сети Интернет – www.tld.by.

Электронная почта. Наиболее полезный ресурс Интернет – это электронная почта, или e-mail. Электронная почта была создана для того, чтобы позволить пользователям Интернет обмениваться короткими текстовыми сообщениями. Однако, как и с другими ресурсами Интернет, возможности почтовых программ существенно расширились. Сейчас по электронной почте возможно пересылать любые документы. Так, электронная почта позволяет не только обмениваться письмами, но и приложить (attach) к письму любой файл: графический, программу и т. д. При этом к одному письму может быть приложено несколько файлов (attachment), благодаря использованию стандарта MIME (Multipurpose Internet Mail Extension), который позволяет приложить к письму произвольное количество attachment-ов, разделяя разные файлы между собой при помощи специальной строки-разделителя (произвольный набор символов, который не встречается в файлах данных и служит для указания границ файлов).

Адрес электронного почтового ящика вида `vasya@server.ru` можно получить двумя путями: первый – завести себе платный почтовый ящик на каком-либо сервере (в частности, у своего провайдера), второй – получить бесплатный почтовый ящик на одном из серверов в Internet. Существует большое количество серверов, которые позволяют создать (sign in) собственный бесплатный почтовый ящик ограниченного объема, просто заполнив несколько простых форм (не обязательно указывать реальные данные). Примеры адресов: www.hotmail.com, www.yahoo.com, www.mail.ru, www.tut.by и др. Работать с такими почтовыми ящиками можно по протоколу HTTP при помощи обычного Web-браузера (например, Internet Explorer) или, если сервер предоставляет конкретный вид сервиса, по протоколам SMTP или POP3 при помощи специальных программ почтовых клиентов Outlook Express, Microsoft Outlook, Netscape Communicator, The Bat.

5.2. Язык гипертекстовой разметки HTML

Язык гипертекстовой разметки HTML (HyperText Markup Language) был предложен Тимом Бернерсом-Ли в 1989 г. и в настоя-

щее время является стандартом для представления гипертекстовых документов в сети World Wide Web (WWW). Поскольку большинство людей, перемещаясь по сайтам при помощи гиперссылок, никогда не покидают WWW, то можно сказать, что HTML – это один из основных языков в Internet.

HTML предназначен для написания гипертекстовых документов, публикуемых в World Wide Web. Документ на языке HTML может включать следующие компоненты:

- стилизованный и форматированный текст;
- команды включения графических и звуковых файлов;
- гиперсвязи с различными ресурсами Internet;
- скрипты на языке JavaScript и VBScript;
- различные объекты, например Flash-анимацию.

5.2.1. Создание Web-документа

Документы HTML являются обычными текстовыми файлами, содержащими специальные теги (или управляющие элементы) разметки. Теги разметки указывают браузеру Web (программе пользователя для отображения web-страниц, например Internet Explorer, Mozilla, Netscape или Opera), как надо вывести страницу.

Файлы HTML обычно имеют расширения htm или html. Их можно создавать при помощи любого текстового редактора.

В документе HTML можно выделить два основных блока: головная часть и тело документа. Содержимое головной части не выводится на экран пользователя, за исключением заголовка, в ней, как правило, указывают ключевые слова, авторов и другую служебную информацию, а также подключают внешние таблицы стилей и скрипты. В теле документа размещают ту информацию, которая будет выведена пользователю.

Документ HTML представляет собой обычный ASCII-файл с добавленными в него управляющими HTML-кодами – тегами. Поскольку теги распознаются и выполняются web-браузером, то язык HTML не зависит от типа компьютера. Существует множество различных тегов, позволяющих включать в HTML-страницу таблицы, рисунки, гиперссылки, задавать шрифт и цвет фона, и даже встраивать в страницу программы, написанные на языках Java, Java-script и VBScript. Однако для того чтобы теги работали, они должны быть правильно организованы. Браузеры не выполняют неизвестные или неправильно записанные теги, благодаря чему достигается

надежность HTML: даже если вы все сделаете не правильно – ничего страшного не произойдет, просто правильные команды будут выполнены, а неправильные или неизвестные браузер проигнорирует.

Тег состоит из имени, за которым может следовать необязательный список атрибутов. Теги HTML используются парами и заключаются между двумя символами угловых скобок <(начальный тег)> и </(конечный тег)>, конечный тег всегда содержит косую черту. Запись тегов осуществляется как прописными буквами, так и строчными.

Текст между начальным и конечным тегами является содержимым элемента. Некоторые теги не имеют конечного, например тег принудительного переноса строки
. Регистр символов для отображения тегов не важен, например, <p> и <P> означает одно и то же.

Каждый тег имеет свой набор допустимых атрибутов. Атрибуты (параметры) уточняют действие тегов, располагаются внутри начального тега и отделяются друг от друга пробелами.

Атрибуты всегда используются в виде пары «имя/значение». Общий формат задания атрибутов имеет вид:

```
<имя_тега имя_атрибута="значение">
```

Например, тег

```
<body bgcolor="red">
```

означает, что цвет фона страницы должен быть красным, а тег

```
<p align="center">
```

задает выравнивание параграфа по центру страницы отображения браузера.

Атрибуты всегда помещаются в начальном теге элемента HTML. Значения атрибутов должны заключаться в кавычки. Наиболее широко используются двойные кавычки, но одиночные кавычки также допустимы.

Все теги HTML по их назначению и области действия можно разделить на следующие основные группы:

- определяющие структуру документа;
- оформляющие блоки гипертекста (параграфы, списки, таблицы, картинки);
- гипертекстовые ссылки и закладки;
- формы для организации диалога;
- вызывающие программы.

Каждый документ начинается с тега `<html>` и содержит заголовок `<head>` и тело `<body>`.

Теги `<head>` и `</head>` ставятся в начале документа и обрамляют теги `<title>`.

Основная часть страницы расположена между тегами `<body>` и `</body>`.

Базовая структура HTML-документа представлена ниже:

```
<html>
<head>
<title> Это заголовок HTML-документа! </title>
</head>
<body> А это содержимое HTML-документа!!! </body>
</html>
```

Теги структуры HTML-документа:

- `<html></html>` – указывают на начало и конец HTML-документа;

- `<head></head>` – начало и конец заголовка документа;

- `<title></title>` – текст, заключенный между этими тегами, отображается в заголовке окна Internet Explorer при просмотре HTML-документа;

- `<body></body>` – содержимое HTML-документа.

Тег `<body>` может иметь следующие параметры:

`text` – цвет текста документа. Название цвета указывается английским словом или в RGB. Пример: `<body text="yellow">` или `text="#FFFF00"`;

`background` – фоновая картинка документа. Пример: `<body background="fon.gif">`;

`bgproperties` – при задании этому параметру значения `fixed` фоновая картинка не будет прокручиваться вместе с документом, т. е. будет неподвижна. Пример: `<body bgproperties="fixed">`;

`bgcolor` – цвет фона документа. Если указать одновременно и параметр `background`, и параметр `bgcolor`, то цвет фона документа будет отображаться, только если не удастся загрузить фоновую картинку;

`nowrap` – при задании этого параметра строчка, не помещающаяся в окне, не будет переноситься на новую строку (появятся полосы прокрутки). Пример: `<body nowrap>`;

`link` – цвет гиперссылок в документе;

`alink` – цвет активных (выделенных) гиперссылок в документе;

`vlink` – цвет посещенных гиперссылок в документе.

В рамках тега `<body></body>` пишется содержимое всего остального документа в виде обычного текста, заключенного в теги форматирования, а также другие специальные теги.

5.2.2. Форматирование текста

Заголовки разного уровня: `<h1></h1>`, `<h2></h2>`, ..., `<h6></h6>`.

Заголовок `<h1>` определяет заголовок самого главного уровня, а `<h6>` – заголовок самого низкого уровня. Пример:

```
<h1>Это заголовок первого уровня</h1>
```

```
<h2>Это заголовок второго уровня</h2>
```

Параграф (абзац) `<p></p>`. Текст каждого параграфа выводится в виде отдельного блока, который отделяется от предыдущих и последующих блоков страницы пустой строкой. Однако отображение параграфа браузером может быть легко изменено посредством таблицы стилей.

Параграфы можно записывать без закрывающего тега `</p>`, однако лучше этого не делать, тем более что в следующей версии HTML не позволит пропустить ни один незакрывающийся тег.

Пример отображения параграфов:

```
<html>
```

```
<body>
```

```
<p>Это параграф 1.</p>
```

```
<p>Это параграф 2.</p>
```

```
<p>Это параграф 3.</p>
```

```
</body>
```

```
</html>
```

Атрибуты выравнивания текста: `align="left"` – по левому краю экрана; `align="center"` – по центру; `align="right"` – по правому краю экрана.

Определение типа, размера и цвета шрифта – тег ``. С данным тегом могут использоваться атрибуты установки размера шрифта: `size=3` – абсолютный размер шрифта (возможные значения от 1 до 7); цвета шрифта: `color="red"`; гарнитуры шрифта: `face="arial"`. Все атрибуты могут быть использованы совместно внутри данного тега. Например:

```
<font size=3 color="red" face="arial"></font>
```

Начертание символов. Для выделения текста используются теги:

```
<b></b>
```

 – выделение текста полужирным шрифтом;

`<i></i>` – выделение текста курсивом;

`<u></u>` – подчеркивание текста.

Переносы строк. Для переноса внутри параграфа используется тег `
`, который выполняет принудительный перенос строки:

```
<html>
<body>
<p>Это <br>пара<br>граф с переносами строк</p>
</body>
</html>
```

Тег `
` не имеет закрывающего тега. Поэтому для совместимости с будущими версиями стандарта рекомендуется следующее написание тега: `
`.

Горизонтальная линейка. Разделять различные элементы можно при помощи горизонтальной линейки, для этого используйте тег `<hr>`:

```
<html>
<body>
<p> Этот параграф отобразится сверху горизонтальной полосы.</p>
<hr>
<p> Этот параграф отобразится снизу горизонтальной полосы.</p>
</body>
</html>
```

Тег `<hr>` не имеет закрывающего тега. Поэтому для совместимости с будущими версиями стандарта рекомендуется следующее написание тега: `<hr/>`. Для этого тега определен ряд атрибутов, но они являются устаревшими. И хотя их применение возможно, но консорциум W3C их использовать не рекомендует. Вместо них следует использовать таблицы стилей.

Комментарии в HTML. Тег комментария используется для вставки комментариев в исходный код HTML. Комментарии будут проигнорированы браузером. Их используют для пояснения кода, что может помочь при редактировании исходного кода в будущем.

Пример:

```
<html>
<body>
Этот текст будет показан в окне браузера.
<!-- Этот текст не будет показан, это комментарий. -->
</body>
</html>
```

Фон. Тег <body> имеет два атрибута, которые позволяют определить фон. Фон можно задавать с помощью цвета или изображения.

Атрибут bgcolor определяет цвет фона для страницы HTML. Значение этого атрибута может быть шестнадцатеричным числом, значением RGB или названием цвета:

```
<body bgcolor="#000000">  
<body bgcolor="rgb(0,0,0)">  
<body bgcolor="black">
```

Все приведенные выше строки задают черный цвет фона. Наиболее часто используется первый способ задания цвета.

Атрибут background определяет изображение для фона страницы HTML. Значением этого атрибута является адрес URL изображения, которое желательно использовать. Если изображение меньше окна браузера, то изображение будет циклически повторяться, пока не заполнит все окно:

```
<body background="clouds.gif">.
```

Цвета в HTML. Цвета выводятся с помощью смешения источников Red (красного), Green (зеленого) и Blue (синего) цвета. Цвета определяют с помощью шестнадцатеричной записи комбинации значений красного, зеленого и синего цветов (RGB). Наименьшее значение, которое можно задать одному из источников, равно 0 (#00). Максимальное значение – 255 (#FF).

Цвет шрифта можно задать с помощью атрибута color в теге . Например:

```
<font color= "f25805">Это цветной текст1</font>  
<font color= "red">Это цветной текст2</font>
```

Списки. HTML поддерживает нумерованные списки, маркированные списки и списки определений. Отличаются эти разновидности списков лишь способом оформления. Перед пунктами маркированных списков обычно ставятся символы-буллеты (bullets), например точки, ромбики и т. п., в то время как пунктам нумерованных списков предшествуют их номера.

Маркированный список является списком элементов. Элементы списка маркируются с помощью специальных знаков (обычно небольшой черный круг). Маркированный список начинается с тега . Каждый элемент списка начинается с тега .

Пример:

```
<html>
<body>
<h4>Маркированный список:</h4>
<ul>
  <li>элемент 1</li>
  <li>элемент 2</li>
  <li>элемент 3</li>
</ul>
</body>
</html>
```

Внутри элемента списка можно помещать параграфы, переносы строк, изображения, ссылки, другие списки и т. д. У тега `` есть параметр `type`, который устанавливает вид маркера:

```
<li type="disk"> – закрашенный кружок (●);
<li type="circle"> – полый кружок (○);
<li type="square"> – закрашенный квадрат (■).
```

Табулирование осуществляется тегом ``. Чтобы отступ (табуляция) был больше, надо вкладывать тег `` в самого себя.

Нумерованный список также является списком элементов. Элементы списка маркируются с помощью чисел или букв. Нумерованный список начинается с тега ``. Каждый элемент списка начинается с тега ``. У тега `` может быть два атрибута: `start` (определяет первое число, с которого начинается нумерация пунктов) и `type` (определяет стиль нумерации пунктов). Последний может иметь значения:

```
"A" – заглавные буквы A, B, C...
"a" – строчные буквы a, b, c...
"I" – большие римские числа I, II, III...
"i" – маленькие римские числа i, ii, iii...
"1" – арабские числа 1, 2, 3...
```

Пример:

```
<html>
<body>
<h4>Нумерованный список:</h4>
<ol>
  <li>элемент 1</li>
  <li>элемент 2</li>
  <li>элемент 3</li>
  <li>элемент 4</li>
```

```
<li>элемент 5</li>
</ol>
</body>
</html>
```

Список определений не является списком элементов. Это список терминов и определений терминов. Список определений начинается с тега <dl>. Каждый термин списка определений начинается с тега <dt>. Каждое определение списка начинается с тега <dd>:

```
<html>
<body>
<dl>
<dt>элемент 1</dt>
<dd>описание элемента 1</dd>
<dt>элемент 2</dt>
<dd>описание элемента 2</dd>
</dl>
</body>
</html>
```

Внутри определения списка определений (тег <dd>) можно помещать параграфы, переносы строк, изображения, ссылки и т. д.

5.2.3. Таблицы

Основным тегом для обозначения таблицы является <table>. Построение таблицы осуществляется по строкам, для обозначения которых применяется тег <tr>. Каждый элемент данных таблицы начинается с тега <td>. Стандарт HTML определяет два типа тегов для обозначения ячеек: <th> и <td>. Первый предназначен для обозначения заголовков, а второй – данных в ячейках. Отличие этих двух элементов заключается лишь в том, как их содержимое отображается браузером. Заголовки большинство браузеров выделяют полужирным шрифтом и центрируют в своих ячейках. Ячейка данных может содержать текст, изображения, списки, параграфы, формы, горизонтальные линейки, таблицы и т. д.

Пример кода программы создания таблицы, содержащей две строки и три столбца:

```
<table border="1">
<tr>
```

```

    <td>1.1</td>
    <td>1.2</td>
    <td>1.3</td>
</tr>
<tr>
    <td>2.1</td>
    <td>2.2</td>
    <td>2.3</td>
</tr>
</table>

```

Теги создания таблицы:

- `<table></table>` – начало и конец таблицы;
- `<tbody></tbody>` – тело таблицы;
- `<tr></tr>` – начало и конец строки таблицы;
- `<th></th>` – начало и конец ячейки шапки таблицы.

Для тега `<table>` определены следующие атрибуты:

- `align` – определяет способ горизонтального выравнивания таблицы на странице. Возможные значения: `left`, `center`, `right`. Значение по умолчанию – `left`;
- `valign` – определяет способ вертикального выравнивания для содержимого таблицы. Возможные значения: `top`, `bottom`, `middle`;
- `border` – определяет ширину внешней рамки таблицы (в пикселах). При `border="0"` или при отсутствии этого параметра рамка отображаться не будет;
- `cellpadding` – определяет расстояние (в пикселах) между рамкой каждой ячейки таблицы и содержащимся в ней материалом;
- `cellspacing` – определяет расстояние (в пикселах) между границами соседних ячеек;
- `width` – определяет ширину таблицы. Ширина задается либо в пикселах, либо в процентном отношении к ширине окна браузера. По умолчанию этот параметр определяется автоматически в зависимости от объема содержащегося в таблице материала;
- `height` – определяет высоту таблицы. Высота задается либо в пикселах, либо в процентном отношении к высоте окна браузера. По умолчанию этот параметр определяется автоматически в зависимости от объема содержащегося в таблице материала;
- `bgcolor` – определяет цвет фона ячеек таблицы. Задается либо RGB-значением в шестнадцатиричной системе, либо одним из 16 базовых цветов;

- background – позволяет заполнить фон таблицы рисунком. В качестве значения необходимо указать URL рисунка.

5.2.4. Гиперссылки

Гипертекстовые ссылки необходимы для соединения с другими документами в Web. Для их записи используется тег `<a href>`. Здесь `a` – открывающий тег, называемый якорем, а `href` – атрибут, определяющий гипертекстовую ссылку.

Пример создания ссылки в документе HTML:

```
<html>
<body>
<p>
<a href="page1.htm">
Этот текст является ссылкой на страницу page1.html</a>
</p>
</body>
</html>
```

5.2.5. Вставка изображения в HTML

Для размещения изображений в HTML используется тег `` с использованием обязательного атрибута `src` (от source – источник), который указывает адрес загружаемого файла с изображением.

Синтаксис определения изображения:

```

```

Пример:

```
<img src "logo.gif" alt="Это логотип">
```

Формат изображения, его расположение и ряд других параметров задается при помощи атрибутов тега ``.

Для выравнивания изображений используется атрибут `align`:

- align=top – изображение выравнивается по верхнему краю текущей текстовой строки без изменения позиции по горизонтали. При этом речь идет как о тексте, так и о графике;
- align=middle – изображение центрируется по вертикали на базовой линии текущей текстовой строки без изменения позиции по горизонтали;
- align=bottom – нижний край изображения выравнивается по горизонтали на базовой линии текущей текстовой строки;

- align=left – изображение смещается к левому краю рабочей зоны, последующий текст сразу же начинает «обтекать» графику;
- align=right – то же, что и для left, только изображение смещается к правой части рабочей зоны.

Пример вариантов выравнивания изображений в тексте:

```
<html>
<body>
<p>Изображение выравнено по верху
при помощи align="top".</p>
<p>Значение "middle" атрибута align

выравнивает изображение по вертикали.</p>
<p>Выравнивание изображения по нижнему краю.

</p>
<p>В этом примере изображение

```

смещено влево, и начиная со следующей строки текст как бы обтекает вставленный рисунок. При таком способе выравнивания желательно помещать изображение в начале параграфа.</p>


```
<p>
```

Изображение прижато к правому краю рабочей области. Текст занимает все свободное пространство слева от изображения, обтекая его.</p>

```
</body>
</html>
```

Для форматирования изображений внутри текста используются атрибуты hspace и vspace, которые определяют интервал между графическим изображением и обтекающим текстом по горизонтали и вертикали соответственно. Размер отступов задается в пикселах:

```
<html>
<body>
<p>
```

В данном варианте используются установленные по умолчанию значения hspace и vspace. Изображение выравнено по левому краю при помощи align="left"</p>.

```
<p>
```

В данном варианте значения hspace и vspace установлены равными 40. Обратите внимание, что эти параметры влияют не столько на отступ от текста до изображения, сколько на величину свободной зоны вокруг изображения вообще. В данном случае изображение сместилось вправо вниз.

```
</p>
```

<p>Примечание: Для лучшего понимания работы атрибутов hspace и vspace в данном примере уменьшите размер окна вашего браузера.

```
</p>
</body>
</html>
```

Изображение в HTML может быть обрамлено прямоугольной рамкой. Для этого используется атрибут `border`, который задает ширину рамки вокруг изображения в пикселах:

```
<html>
<body>
<p><img src = "logo.gif" align = "left">
Изображение без рамки (по умолчанию для простых графических
изображений).</p>
```

```
<p><img src = "logo.gif" align = "left" border = "5">
```

В данном варианте толщина рамки установлена в 5 пикселей.

```
</p>
```

При помощи атрибутов `height` и `width` (высота и ширина соответственно) можно задать размеры отображения графического файла на Web-странице:

```
<html>
<body>
```

В первом случае изображение выводится с сохранением своих исходных размеров (253*190 пикселей).

```
<br>
```

```

```

```
<br>
```

Во втором варианте высота изображения уменьшена до 100 пикселей. Обратите внимание, что хотя явно задан только атрибут `height`, браузер автоматически отмасштабировал изображение и по ширине с сохранением исходных пропорций.

```
<br>
```

```

```

```
<br>
```

В третьем варианте принудительно заданы оба атрибута, при этом высота уменьшена до 100 пикселей, а ширина оставлена на прежнем уровне. В этом случае автоматического масштабирования не производится, и изображение сжимается по вертикали.

```
<br>
```

```

```

```
</body>
```

```
</html>
```

5.2.6. Бегущая строка

Бегущую строку можно создавать при помощи тега `<marquee>`, например так:

```
<marquee> Приветик </marquee>
```

или в более полном варианте:

```
<marquee width = "75%" height = "20px" bgcolor = "yellow" direction =  
"left" behaviour = "scroll" loop = "10" scrollamount = "30px" scrolldelay = "30"  
align = "top"> Приветик </marquee>
```

Атрибуты тега <marquee>:

- width, height, bgcolor – соответственно ширина, длина и цвет фона окна бегущей строки;
- direction – направление движения: left – слева, right – справа;
- behaviour – поведение: scroll – прокрутить текст *n* раз и исчезнуть, slide – прокрутить текст *n* раз и остаться, alternate – отскакивать от краев экрана;
- loop – число раз прокрутки. Если loop не указано или указано число –1, то прокрутка будет бесконечной;
- scrollamount – количество пикселей, на которое строка смещается за 1 шаг;
- scrolldelay – задержка в миллисекундах перед каждым шагом прокрутки;
- align – выравнивание строки в своем окне: top – верх, middle – центр, bottom – низ.

5.3. Редактор FrontPage

Microsoft FrontPage – это обширная прикладная программа, которую можно использовать для разработки сайтов. Она содержит все, что вам требуется для создания сайтов, от простой веб-сводки до сложного веб-магазина розничной торговли.

FrontPage позволяет создавать страницы профессионального качества, при этом не требуя знания тонкостей HTML. FrontPage сам сгенерирует соответствующий код HTML. При желании вы сможете прямо в редакторе писать код HTML и тут же, не запуская браузер, видеть результаты своей деятельности.

Работа в редакторе FrontPage весьма похожа на работу с текстовым процессором, подобным Microsoft Word. FrontPage позволяет одновременно открывать несколько файлов, а так как редактор оперирует страницами, то смело можно сказать, что он позволяет открывать одновременно несколько страниц. Эта возможность очень

полезна, например, в тех случаях, когда, чтобы соблюсти согласованность и точность информации, требуется быстро переходить от одной страницы к другой. Редактор также позволяет скопировать страницу (вместе со всем кодом HTML) из Internet и отредактировать ее. Это может быть удобно, когда надо быстро получить информацию с других своих сайтов.

Интерфейс пакета существенно отличается от других приложений Microsoft Office, хотя выдержан единый стиль его оформления. Как показано на рис. 5.2, верхние четыре строки похожи на интерфейсы других приложений: в верхней строке содержится заголовок документа, однако с указанием абсолютного пути к файлу, затем следует строка меню, еще ниже размещаются две панели инструментов и основное место занимает рабочее окно программы. Панели инструментов схожи с аналогичными в пакете Word в режиме HTML-редактора.

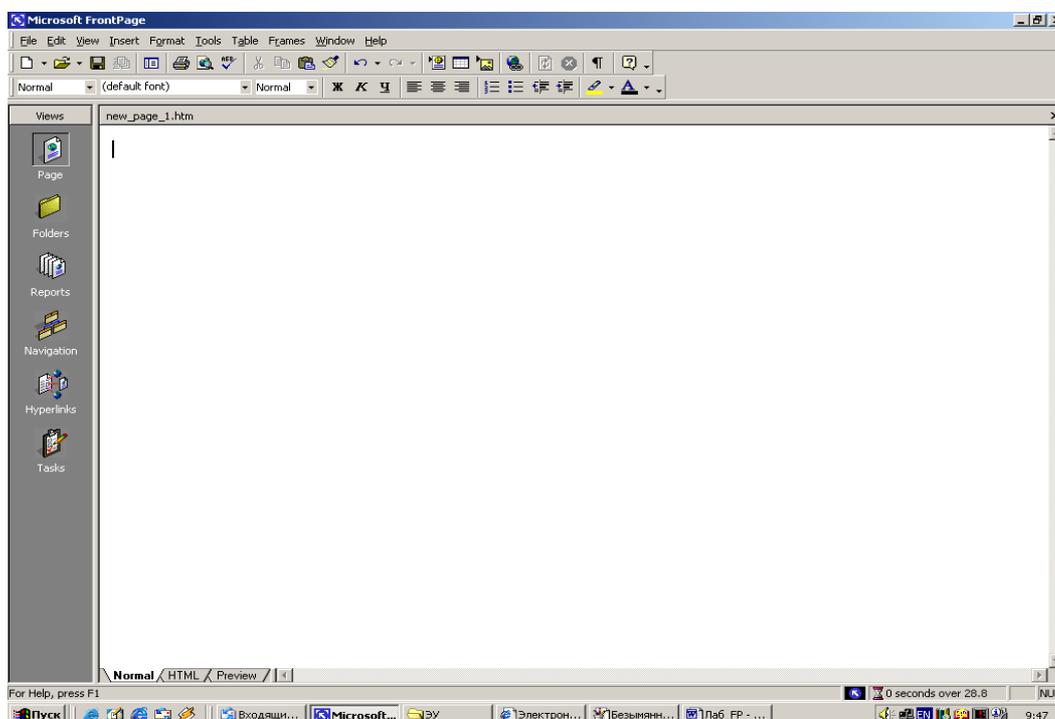


Рис. 5.2. Окно редактора FrontPage

Окно редактора делится по вертикали на три части, размер которых можно регулировать стандартными средствами оболочки Windows. Слева размещается панель переключателя режимов. С помощью ее кнопок можно выбрать один из вариантов представления Web-узла (электронного издания). В панели **Folder List (Список папок)** показана иерархическая структура папок и файлов, входящих

в состав издания. Основную часть рабочего окна занимает **Панель представления**, в которой отображается выбранный элемент структуры Web-узла – обычно одна из Web-страниц, которая входит в состав данного узла.

Варианты представления Web-узла:

- Режим страницы (**Page**). В таком режиме можно редактировать любую выбранную страницу электронного издания, изменять размещенный на ней текст и рисунки, вводить новые HTML-теги, задавать новые и редактировать имеющиеся гиперссылки. Для переключения режимов просмотра страницы в нижней части панели представления имеются три вкладки: **Normal**, **HTML** и **Preview**.

- Кнопка **Folders (Папки)** включает режим анализа структуры папок и файлов, входящих в Web-узел. Этот режим подобен просмотру структуры локальных дисков, определяемому Проводником.

- В режиме **Reports (Отчеты)** в рабочем окне представлен отчет о текущем состоянии Web-узла.

- Кнопка **Navigation (Навигация)** отображает в рабочем окне программы структуру издания со всем разнообразием гиперссылок, позволяющих перемещаться как внутри данного Web-узла, так и на страницы других узлов.

- Кнопка **Hyperlinks (Гиперссылки)** в графической форме представляет все гиперссылки, переводящие на текущую страницу и с текущей страницы на другие страницы узла.

- Кнопка **Tasks (Задачи)** переводит рабочее окно в режим, напоминающий записную книжку. В нем в рабочем окне отображается список задач, намеченный пользователем для выполнения в будущем.

FrontPage включает в себя все необходимое для работы с Web-узлом: программу FrontPage Explorer для навигации по Web-узлу, мощный редактор Web-страниц FrontPage Editor, средства для работы с графикой, средства для публикации документов и пр.

Важной составной частью интерфейса пакета являются панели инструментов. Первые две из них – **Стандартная** и **Форматирование** – используются по умолчанию и в основном подобны аналогичным панелям инструментов в других пакетах, входящих в состав MS Office. Это относится к инструментам **Создать**, **Открыть**, **Сохранить**, **Печать**, **Проверка орфографии**, **Вырезать**, **Скопировать**, **Вклеить**, **Отменить**, **Повторить**, **Отобразить непечатные символы**, **Помощь** панели **Стандартная**. Действие всех перечисленных инструментов дублируется командами секций меню.

Из специализированных инструментов этой панели следует отметить кнопку **Web-публикация**, расположенную правее кнопки **Сохранить**. Этот инструмент осуществляет публикацию разработанного Web-узла в сети по определенному адресу. Перед стандартным инструментом **Таблица** находится кнопка **Вставить компонент**, расположенный рядом с ней черный треугольник показывает наличие нескольких вариантов этого инструмента, которые становятся доступными при его активизации. Инструмент **Таблица**, предназначен для размещения соответствующего элемента на Web-странице. Другие панели инструментов могут быть выведены на экран с помощью команды **Toolbar (Панели инструментов)** меню **View (Вид)**. Их назначение:

- **DHTML Effects (Эффекты динамического HTML)**. Эта панель обеспечивает добавление соответствующих эффектов с помощью своих инструментов. Сам термин используется для обозначения HTML-страниц с динамически изменяемым содержимым;
- **Pictures (Рисование)**. Панель служит для создания и редактирования графических изображений;
- **Positioning (Размещение)**. Служит для изменения положения элементов на проектируемой Web-странице;
- **Style (Стиль)**. Служит для стилового оформления элементов Web-страницы;
- **Reporting (Отчеты)**. Эта панель инструментов служит для управления отчетами о состоянии Web-узла;
- **Navigation (Навигация)**. Инструменты этой панели обеспечивают работу в режиме навигации;
- **Tables (Таблицы)**. Эта панель объединяет инструменты, предназначенные для создания таблиц и работы с ними.

Основное достоинство редактора FrontPage для начинающих разработчиков – интегрированная среда создания Web-страниц, объединяющая возможности текстового редактора, компилятора HTML и браузера.

Создание Web-страницы в редакторе FrontPage:

1. Откройте окно редактора: **Пуск** > **Программы** > **FrontPage**.
Откройте окно нового документа: **File** > **New Page** или нажмите на клавиши <Ctrl> + <N>.

2. Наберите в окне текст:
Это моя первая страница.
Я учусь в БГТУ
Моя фамилия ПЕТРОВ.
НОМЕР ГРУППЫ 123.

3. Выделите строку «Это моя первая страница.». Откройте пункт меню **Format** ➤ **Font**. Измените стиль (**Style**), размер (**Size**), цвет (**Color**).

4. Откройте закладку **HTML** и просмотрите код созданной страницы.

5. Откройте закладку **Preview**. Просмотрите, как будет выглядеть ваша страница в окне браузера.

6. Перейдите на закладку **Normal**.

7. Измените размер, цвет, начертание введенного текста.

8. Добавьте таблицу: **Table** ➤ **Insert** ➤ **Table**. Создайте таблицу из трех столбцов и четырех строк. Заполните их фамилиями и именами своих друзей.

9. Просмотрите, как выглядит текст HTML-документа (переключитесь на закладку **HTML**). Перейдите на закладку **Normal**. Выделите вторую строку текста. Сделайте ее «бегущей». Для этого выполните следующие действия: **Insert** ➤ **Component** ➤ **Marquee**.

10. Вставьте картинку из стандартной библиотеки рисунков: **Insert** ➤ **Picture**.

11. Вставьте дату и время. Дата и время могут быть «статическими» и «динамическими», т. е. обновляемыми каждый раз при открытии вашей страницы. Для этого выполните действия **Insert** ➤ **Data and Time**.

12. За строкой «Я учусь в БГТУ.» закрепите ссылку на сайт БГТУ. Для этого выполните следующие действия: выделите строку, выберите **Insert** ➤ **Hyperlink**. В строке **URL** введите адрес сайта БГТУ: www.bstu.unibel.by. Нажмите на кнопку **OK**.

КРАТКИЙ СПРАВОЧНИК ПО TURBOPASCAL

1. Алфавит языка

Алфавит языка – это набор зарезервированных символов и ключевых слов, с помощью которых пользователь создает свои прикладные программы.

Основными символами языка в системе программирования **TurboPascal** являются:

- латинские буквы A ... Z, a ... z;
- цифры 0 ... 9;
- спецсимволы + - * / = < > () [] { } . , ; # \$.

Нет различий между большими (прописными) и малыми (строчными) буквами при их использовании для определения имен переменных, процедур, функций и меток. Максимальная длина программной строки – 127 символов. В символьных константах и комментариях могут использоваться любые другие знаки (например, буквы русского алфавита).

Зарезервированными словами в языке программирования Паскаль (они не могут быть переопределены пользователем) являются:

ABSOLUTE	FILE	MOD	SHR
AND	FOR	ML	STRING
ARRAY	FORWARD	NOT	THEN
BEGIN	FUNCTION	OF	TO
CASE	GOTO	OR	TYPE
CONST	IF	PACKED	UNIT
BTV	IMPLEMENTATION	PROCEDURE	UNTIL
DO	IN	PROGRAM	USES
DOWNTO	INLINE	RECORD	VAR
ELSE	INTERFACE	REPEAT	WHILE
END	INTERRUPT	SET	WITH
EXTERNAL	LABEL	SHL	XOR

2. Элементы языка, определяемые пользователем

Идентификаторы используются в качестве имен переменных, констант, программ, процедур и т. п., начинаются с латинской буквы или знака подчеркивания и состоят из латинских букв, цифр и симво-

ла «_». Длина идентификатора ограничена длиной программной строки (т. е. 127 символами), но при этом для компилятора значимыми являются только первые 64 символа. Прописные и строчные буквы не различаются, т. е. `_index` и `_INDEX` – это один и тот же идентификатор. Запрещено в качестве идентификаторов использовать зарезервированные ключевые слова.

Комментарий – любой текст, ограниченный `(*...*)` или `{...}`. Вложенность комментариев не допускается, но можно `(*...{...}...*)` и `{...(*...*)...}`. Комментарий может быть помещен в любом месте программы.

3. Стандартные функции

Стандартные основные арифметические функции TurboPascal приведены в таблице:

Функция	Назначение
Abs(x)	Вычисление абсолютного значения x
Sqr(x)	Вычисление квадрата x (x^2)
Sin(x)	Вычисление синуса x
Cos(x)	Вычисление косинуса x
Arctan(x)	Вычисление арктангенса x
Exp(x)	Вычисление экспоненты x
Ln(x)	Вычисление натурального логарифма x
Sqrt(x)	Вычисление квадратного корня x

Отметим, что в тригонометрических функциях синуса или косинуса аргумент должен быть задан только в радианах. Если аргумент x дан в градусах, то для перевода значения аргумента в радианы используется формула $y = x \cdot \pi / 180$.

Видно, что в TurboPascal определены только три тригонометрические функции (Sin, Cos, Arctan). Для вычисления остальных тригонометрических функций необходимо использовать известные соотношения:

$$\text{Tg}(x) = \text{Sin}(x) / \text{Cos}(x);$$

$$\text{Ctg}(x) = \text{Cos}(x) / \text{Sin}(x);$$

$$\text{Csc}(x) = 1 / \text{Sin}(x);$$

$$\text{Sc}(x) = 1 / \text{Cos}(x);$$

$$\text{Log}_a(x) = \ln(x) / \ln(a).$$

ПРИЛОЖЕНИЕ 2

ЛИСТИНГИ РЕШЕНИЯ ТРАНСПОРТНЫХ ЗАДАЧ

Закрытая задача	Открытые задачи	
<p>Запасы равны потребностям</p> $\sum_{i=1}^m a_i = \sum_{j=1}^n b_j$	<p>Запасы больше потребностей</p> $\sum_{i=1}^m a_i > \sum_{j=1}^n b_j$	<p>Запасы меньше потребностей</p> $\sum_{i=1}^m a_i < \sum_{j=1}^n b_j$
<p>Запасы продукта у поставщиков 1, 2 и 3</p> $a \equiv \begin{bmatrix} 30 \\ 40 \\ 20 \end{bmatrix}$ <p>в сумме равны потребностям потребителей</p> $b \equiv \begin{bmatrix} 20 \\ 25 \\ 35 \\ 10 \end{bmatrix}$	<p>Запасы продукта у поставщиков</p> $a \equiv \begin{bmatrix} 35 \\ 40 \\ 25 \end{bmatrix}$ <p>в сумме превышают потребности потребителей</p> $b \equiv \begin{bmatrix} 20 \\ 25 \\ 35 \\ 10 \end{bmatrix}$	<p>Запасы продукта у поставщиков</p> $a \equiv \begin{bmatrix} 30 \\ 40 \\ 20 \end{bmatrix}$ <p>в сумме меньше потребности потребителей</p> $b \equiv \begin{bmatrix} 25 \\ 30 \\ 40 \\ 10 \end{bmatrix}$
<p>Ограничения: равенство вывозимого продукта запасам у поставщиков: $(x \cdot e)^{<0>} = a;$ количество вывозимого продукта равно потребностям потребителей: $((e \cdot x)^T)^{<0>} = b$</p>	<p>Ограничения: количество вывозимого продукта меньше запасов у поставщиков: $(x \cdot e)^{<0>} \leq a;$ количество вывозимого продукта больше потребностей потребителей: $((e \cdot x)^T)^{<0>} \geq b$</p>	<p>Ограничения: количество вывозимого продукта больше запасов у поставщиков: $(x \cdot e)^{<0>} \geq a;$ количество вывозимого продукта меньше потребностей потребителей: $((e \cdot x)^T)^{<0>} \leq b$</p>
<p>Ответ:</p> $y = \begin{bmatrix} 20 & 0 & 10 & 0 \\ 0 & 25 & 5 & 10 \\ 0 & 0 & 20 & 0 \end{bmatrix}.$ <p>Минимальные затраты на перевозку $f(y) = 210$</p>	<p>Ответ:</p> $y = \begin{bmatrix} 20 & 5 & 0 & 0 \\ 0 & 20 & 10 & 10 \\ 0 & 0 & 25 & 0 \end{bmatrix}.$ <p>Минимальные затраты на перевозку $f(y) = 195$</p>	<p>Ответ:</p> $y = \begin{bmatrix} 25 & 0 & 5 & 0 \\ 0 & 30 & 0 & 10 \\ 0 & 0 & 20 & 0 \end{bmatrix}.$ <p>Минимальные затраты на перевозку $f(y) = 185$</p>

КРАТКИЙ АНГЛО-РУССКИЙ СЛОВАРЬ

Abort – отказать(ся)	Extent – протяжение
Absent – отсутствовать	External – внешний
Access – доступ	Fail – пропускать
Account – отчет, счет	Field – поле
Adept – знаток	File – реестр, дело
Advenced – улучшение	Found – обнаруживать
Again – опять	Free – свобода
Another – другой	General – общий
Available – доступный	Greate – создавать
Big – большой	Halted – не работает
Bond – связь	Head – головка
Boorkmark – закладка	High – высокий
Break – прекращение	Host – хозяин; узел
Brouse – поиск, обзор	Ignore – игнорировать
Cancel – аннулировать	Insert – вставлять
Can't – невозможно	Job – работа
Chat – разговор	Land – земля
Choice – выбор	Landscape – ландшафт
Choose – выбирать	Law – закон
Chum – приятель	List – печать
Circle – окружность	Local – местный
Cite – вызывать	Longer – ждать
Complete – нормальный	Memory – память
Device – устройство	Name – имя
Domain – владение	Net – сеть
Echange – обмен	New – новый
Empty – пустой	None – неизвестный
Enouth – достаточный	Note – примечание
Enter – ввод	Out look – смотреть
Excel – превосходный	Overflow – переполнение
Excite – возбуждать	Path – путь
Execute – выполнять	Point – точка
Exist – быть	Press – нажимать
Exit – выход	Protect – защита
Exlude – исключать	Provide – снабжение
Explorer – исследовать	Provider – снабженец

Rank – ряд
Read – читать
Replace – заменять
Retry – повторять
Room – место
Serve – служить
Side – поверхность
Site – участок
Soft – мягкий
Source – источник
Space – область
Strike – нажимать
Sure – уверенный
Target – приемник
Too – слишком
Try – попытаться

Unable – невозможный
Us – нам
Use – применение
Valid – действительно
Vary – изменять
Very – очень
Voice – выражать, голос
Volume – том
Wait – ждать
Ware – изделия
Web – паутина
Wid – широкий
Word – слово
World – мир
Write – писать
Yes – да

ЛИТЕРАТУРА

1. Леонтьев, В. П. Новейшая энциклопедия персонального компьютера / В. П. Леонтьев. – 3-е изд., перераб. и доп. – М.: ОЛМА-ПРЕСС, 2002. – 847 с.
2. Дьяконов, В. П. Mathcad: учеб. курс / В. П. Дьяконов. – СПб.: Питер, 2001. – 640 с.
3. Акулич, И. Л. Математическое программирование в примерах и задачах / И. Л. Акулич. – М.: Высшая школа, 1986. – 320 с.
4. Лащенко, А. П. Информатика и компьютерная графика: учеб.-метод. пособие для студентов специальности 1-75 01 01 «Лесное хозяйство» / А. П. Лащенко. – Минск: БГТУ, 2004. – 66 с.
5. Лащенко, А. П. Компьютерные информационные технологии: учеб. пособие / А. П. Лащенко, Т. П. Брусенцова, Н. И. Потапенко. – Минск: БГТУ, 2004. – 70 с.
6. Кирьянов, Д. В. Самоучитель Mathcad 2001 / Д. В. Кирьянов. – СПб.: БХВ-Петербург, 2002. – 544 с.
7. Вержбицкий, В. М. Основы численных методов / В. М. Вержбицкий. – М.: Высшая школа, 2002. – 840 с.
8. Ракитин, В. И. Практическое руководство по методам вычислений с приложением программ для персональных компьютеров: учеб. пособие / В. И. Ракитин. – М.: Высшая школа, 1998. – 383 с.
9. Вирт, Н. Алгоритмы и структуры данных / Н. Вирт. – М.: Мир, 1989. – 360 с.
10. Информатика. Базовый курс / С. В. Симонович [и др.]. – СПб.: Питер, 2001. – 640 с.
11. Кульгин, Н. Б. TurboPascal в задачах и примерах / Н. Б. Кульгин. – СПб.: БХВ-Петербург, 2000. – 256 с.
12. Рудаков, П. Основы языка Pascal / П. И. Рудаков, М. А. Федотов. – М.: Радио и связь, горячая линия – Телеком, 1999. – 208 с.
13. Меженный, О. А. TurboPascal: самоучитель / О. А. Меженный. – М.: Вильямс, 2003. – 336 с.
14. Басалыга, В. И. Основы компьютерной грамотности: произв.-практ. пособие / В. И. Басалыга. – Минск: НТЦ «АТЦ», 2001 – 332 с.

ОГЛАВЛЕНИЕ

Предисловие	3
1. ОПЕРАЦИОННЫЕ СИСТЕМЫ.....	4
1.1. Операционные системы корпорации Microsoft	6
1.1.1. Семейство DOS	6
1.1.2. Семейство Windows 3.X/9X.....	8
1.1.3. Windows 98/98 SE.....	11
1.1.4. Семейство Windows NT/2000/XP	11
1.2. Альтернативные операционные системы	13
1.2.1. Операционная система Unix	13
1.2.2. Операционная система Linux.....	15
2. ОФИСНЫЕ ТЕХНОЛОГИИ	17
2.1. Общие сведения о Microsoft Office	17
2.2. Общие сведения о Microsoft Word.....	19
2.2.1. Окно программы Word	21
2.2.2. Создание и редактирование текстовых документов.....	23
2.2.3. Подготовка документа к печати	31
2.2.4. Работа с таблицами	34
2.2.5. Работа с графическими объектами.....	40
2.2.6. Вставка объектов.....	45
2.2.7. Автоматическое оглавление	46
2.3. Общие сведения о Microsoft Excel.....	47
2.3.1. Окно программы Excel	48
2.3.2. Форматирование таблицы	54
2.3.3. Работа с формулами.....	56
2.3.4. Функции	57
2.3.5. Создание диаграмм	61
2.4. MS Power Point	64
2.4.1. Создание презентации	65
2.4.2. Работа со слайдами и объектами.....	68
2.4.3. Демонстрация презентации.....	71
3. ОСНОВЫ ПРОГРАММИРОВАНИЯ.....	77
3.1. Система программирования TurboPascal.....	78
3.2. Начальные сведения для подготовки и выполнения программ.....	81
3.3. Структура программы Пакаль	83

3.4. Программирование алгоритмов линейной структуры	85
3.4.1. Оператор присваивания.....	86
3.4.2. Операторы ввода-вывода	86
3.5. Программирование алгоритмов разветвляющейся структуры.....	90
3.6. Программирование алгоритмов циклической структуры.....	93
3.7. Программирование с использованием массивов	96
3.7.1. Описание массива	96
3.7.2. Действия над элементами массива.....	97
3.8. Модули	99
3.8.1. Текстовый режим работы. Модуль Crt	101
3.8.2. Графический режим работы. Модуль Graph.....	103
3.9. Программирование в системе TurboPascal	106
4. ПРИКЛАДНОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ.....	114
4.1. Общие сведения о программе Mathcad	114
4.2. Входной язык системы Mathcad	115
4.2.1. Операторы.....	116
4.2.2. Числовые и размерные константы	117
4.2.3. Переменные	118
4.2.4. Массивы	119
4.2.5. Стандартные и пользовательские функции	124
4.3. Ввод формул и текста	124
4.4. Форматирование формул и текста.....	126
4.5. Решение уравнений и систем.....	126
4.6. Построение графиков.....	127
4.7. Аналитические вычисления	129
4.8. Использование системы Mathcad для решения инженерно- экономических задач	130
4.8.1. Решение линейных и нелинейных уравнений и систем	130
4.8.2. Поиск экстремумов функций.....	136
4.8.3. Решение задач линейного программирования.....	138
4.8.4. Решение задач интерполирования и регрессионного анализа	152
5. СЕТЕВЫЕ ТЕХНОЛОГИИ.....	156
5.1. Компьютерные сети	156
5.1.1. История Internet.....	156
5.1.2. Протоколы прикладного уровня.....	157
5.1.3. Web-браузеры	159
5.1.4. Адресация в Интернет	161

5.2. Язык гипертекстовой разметки HTML	162
5.2.1. Создание Web-документа.....	163
5.2.2. Форматирование текста.....	166
5.2.3. Таблицы.....	170
5.2.4. Гиперссылки	172
5.2.5. Вставка изображения в HTML.....	172
5.2.6. Бегущая строка.....	174
5.3. Редактор FrontPage.....	175
Приложение 1	180
Приложение 2	182
Приложение 3	183
Литература	185

Учебное издание

Лашенко Анатолий Павлович
Брусенцова Татьяна Палладьевна

ИНФОРМАТИКА И КОМПЬЮТЕРНАЯ ГРАФИКА

Учебно-методическое пособие

Редактор *И. О. Гордейчик*
Компьютерная верстка *Д. С. Семижен*

Подписано в печать 22.09.2008. Формат 60×84¹/₁₆.
Бумага офсетная. Гарнитура Таймс. Печать офсетная.
Усл. печ. л. 11,0. Уч.-изд. л. 11,4.
Тираж 300 экз. Заказ .

Учреждение образования
«Белорусский государственный технологический университет».
220006. Минск, Свердлова, 13а.
ЛИ № 02330/0133255 от 30.04.2004.

Отпечатано в лаборатории полиграфии учреждения образования
«Белорусский государственный технологический университет».
220006. Минск, Свердлова, 13.
ЛП № 02330/0056739 от 22.01.2004.

Переплетно-брошюровочные процессы
произведены в ОАО «Полиграфкомбинат им. Я. Коласа».
220600. Минск, Красная, 23. Заказ .