Министерство образования и науки Российской Федерации
НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Т.А. АСТАШОВА

ИНФОРМАТИКА

Утверждено
Редакционно-издательским советом университета
в качестве учебного пособия

НОВОСИБИРСК
2017

УДК 004(075.8)
А 91

Рецензенты:

О.В. Нос, д-р техн. наук, доцент;
А.В. Гаврилов, канд. техн. наук, доцент;
Б.В. Малозёмов, канд. техн. наук, доцент

Работа подготовлена на кафедре проектирования технологических машин для студентов I курса МТФ, обучающихся по дисциплине «Информатика» всех направлений

Асташова Т.А.

А 91 Информатика: учебное пособие / Т.А. Асташова. – Новосибирск: Изд-во НГТУ, 2017. – 108 с.

ISBN 978-5-7782-3435-2

В пособии представлены материалы для эффективного изучения дисциплины «Информатика»: темы лекционных занятий, теоретический материал для выполнения лабораторных работ, рекомендуемая литература.

Предназначено для студентов I курса механико-технологического факультета всех направлений, а также при выполнении лабораторных и самостоятельных работ студентами технических специальностей вуза.

УДК 004(075.8)

ISBN 978-5-7782-3435-2

ВВЕДЕНИЕ

Информатика – это основанная на использовании компьютерной техники дисциплина, изучающая структуру и общие свойства информации, а также закономерности и методы ее создания, хранения, поиска, преобразования, передачи и применения в различных сферах деятельности человека.

В последнее десятилетие информатика как фундаментальная наука становится наиболее важной составляющей всей системы научного познания и будет в значительной степени определять пути формирования глобального информационного общества, основанного на знаниях.

В настоящее время без информационных технологий не представляется возможным изучение общенаучных и специальных дисциплин, что делает изучение дисциплины «Информатика» наиболее актуальным. Вместе с этим стремительные темпы развития компьютерных средств и технологий обусловливают постоянное обучение и саморазвитие.

Основная образовательная программа подготовки бакалавров по всем направлениям механико-технологического факультета предусматривает изучение дисциплины «Информатика» как базовый компонент.

Цель дисциплины «Информатика» – изучение основных понятий и определений информатики, способов представления, обработки, передачи и хранения данных, архитектуры компьютера, базовых инструментальных средств, проблемных пакетов программ, офисных технологий.

Особенность данной дисциплины в том, что содержание лекционных курсов и лабораторных работ постоянно меняется. С одной стороны, появление новых технических средств и технологий расширяет содержание курса, с другой – число учебных часов постоянно.

Настоящее учебное пособие призвано помочь студентам I курса механико-технологического факультета всех направлений эффективно

изучить дисциплину «Информатика» обязательного образовательного цикла дисциплин. В учебном пособии представлены материалы для самостоятельного изучения и для использования их на лекционных и лабораторных аудиторных занятиях: темы лекционных занятий, теоретический материал для выполнения лабораторных работ, рекомендуемая литература.

После изучения дисциплины «Информатика» студентам гарантированы следующие результаты.

1. Знание правовых основ информационной безопасности и принципов защиты авторского права на программные продукты.
2. Знание сущности и значения информации в развитии современного общества, опасности и угроз, возникающих в этом процессе.
3. Умение использовать элементарные навыки алгоритмизации и программирования на одном из языков высокого уровня как средство программного моделирования изучаемых объектов и процессов.
4. Умение осуществлять поиск информации в локальных и глобальных сетях.
5. Владение персональным компьютером как средством управления информацией.
6. Умение оценивать состояние и тенденции развития информационных технологий и информатики в современном обществе.
7. Умение проводить библиографическую и информационно-поисковую работу, использовать ее результаты при решении профессиональных задач и оформлении научных трудов.
8. Умение использовать специализированные программные средства при решении профессиональных задач.
9. Умение пользоваться наиболее распространенными офисными и математическими пакетами прикладных программ.
10. Умение применять основные методы, способы и средства получения, хранения и переработки информации с помощью компьютеров и компьютерных средств.

1. СОДЕРЖАНИЕ УЧЕБНОЙ ДИСЦИПЛИНЫ: ТЕМЫ ЛЕКЦИОННЫХ ЗАНЯТИЙ

Тема 1. Основные понятия информатики

Сигналы и данные. Понятие информации. Свойства информации: объективность, полнота, достоверность, адекватность, доступность, актуальность. Информатизация общества. Компьютер как техническое средство информатизации. Информатика как наука и учебная дисциплина.

Тема 2. Меры и единицы измерения информации

Формулы Хартли и Шеннона. Бит, байт и производные от них единицы.

Тема 3. Позиционные системы счисления

Десятичная и двоичная системы. Восьмеричная и шестнадцатеричная системы. Алгоритмы перевода чисел из одной системы в другую.

Тема 4. Логические основы ЭВМ

Двоичный алфавит. Кодирование символьной информации. Представление целых и вещественных чисел. Связь представления чисел с количеством отводимых байтов. Логические величины. Логические операции. Техническая реализация логических величин.

Тема 5. История развития ЭВМ. Понятие и основные виды архитектуры ЭВМ

Докомпьютерный период. Создание первого компьютера. Ламповые и транзисторные ЭВМ. Микроэлектронная база ВТ: интегральные схемы малой степени интеграции, БИС и СБИС. Микропроцессоры.

Принципы фон Неймана. Понятие архитектуры. Процессор, запоминающее устройство (ЗУ). Шина данных, адресная шина и шина команд. Архитектуры с фиксированным набором устройств. Открытые архитектуры.

Тема 6. Состав и назначение основных элементов персонального компьютера

Системный блок и его состав: системная плата, жесткий диск, дисководы, блок питания и другие устройства. Системы ввода-вывода информации: дисплей, клавиатура, мышь, принтер и др. Устройства на системной плате. Характеристики современных микропроцессоров. Системные шины и слоты расширения.

Тема 7. Запоминающие устройства: классификация, принцип работы, основные характеристики

Оперативные и постоянные ЗУ. Адресное пространство ЗУ. Ячейка памяти (ЯП), статические и динамические ЯП. Основные характеристики ЗУ. Техническая реализация модулей памяти. Накопитель на жестком магнитном диске. Принцип работы, основные характеристики. Низкоуровневая структура дисков: дорожки, сектора, цилиндры. Логические диски. Загрузочный сектор, таблицы размещения файлов.

Тема 8. Устройства ввода-вывода данных, их разновидности и основные характеристики

Мониторы. Принципы работы мониторов различных типов. Основные характеристики мониторов. Видеоадаптер: назначение, основные характеристики. Клавиатура, разновидности клавиатур. Манипулятор типа «мышь». Принтеры и сканеры. Мультимедийный проектор.

Тема 9. Понятие системного и служебного (сервисного) программного обеспечения

Системное и прикладное программное обеспечение (ПО). Определение операционной системы (ОС). Функции ОС. Классификация ОС. Эволюция ОС Windows. Концепции графического интерфейса Windows, рабочий стол, окно, объект. Стандартные программы и служебные утилиты: восстановление системы, очистка и дефрагментация дисков, архивация данных. Антивирусные программы. Использование справки.

Тема 10. Файловая структура операционных систем. Операции с файлами

Понятие файловой системы. Функции файловой системы. Примеры файловых систем: FAT, NTFS. Файловая система Windows: файлы и папки. Файловые менеджеры. Копирование, перенос, удаление и переименование файлов средствами Windows и файловыми менеджерами.

Тема 11. Технология обработки текстовой и графической информации

Текст как объект обработки. Культура оформления электронных документов. Редактор документов MS Word. Открытие и сохранение документа. Шаблоны документов. Установка параметров страницы. Настройка шрифта и параметров абзаца. Ввод и редактирование текста. Создание списков. Гиперссылки. Вставка формул. Вставка оглавления, нумерации страниц, колонтитулов, даты. Стили оформления документов. Создание таблиц. Вставка объектов. Проверка орфографии.

Тема 12. Основы баз данных. Системы управления базами данных

Общее понятие о базах данных (БД). Классификация БД. Функции ввода данных, хранения, корректировки, поиска, упорядочения. Защита информации БД. Реляционная таблица как способ хранения данных. Поля и записи. Типы полей. Ключевые поля. Основные и вспомогательные таблицы. Связи таблиц.

Тема 13. Электронные таблицы

Способы и приемы работы с основными элементами MS Excel: таблицами, ячейками, формулами, встроенными функциями. Работа с различными типами данных. Диаграммы и графики. Решение линейных уравнений, нелинейных уравнений и систем уравнений.

Тема 14. Средства электронных презентаций

Основные этапы создания презентаций, структура презентаций. Структура слайда, настройка эффектов анимации. Работа с различными режимами презентаций.

Тема 15. Моделирование как метод познания. Классификация и формы представления моделей

Функциональные и вычислительные задачи. Основные понятия теории моделирования. Суть процесса моделирования объекта. Классификация моделей в зависимости от формы представления объекта оригинала: материальные модели и мысленные. Классификация материальных моделей, абстрактных моделей. Определения и примеры моделей разных видов. Классификация математических моделей. Различные классификационные признаки. Примеры моделей из разных предметных областей.

Тема 16. Методы и технологии моделирования

Общий вид математической модели системы. Классификация методов идентификации математических моделей. Особенности, области использования, примеры задач, достоинства и недостатки, прикладное программное обеспечение. Понятие компьютерного моделирования.

Тема 17. Этапы решения задач на компьютерах

Постановка и формализация задачи. Построение математической и информационной модели. Выбор и обоснование метода решения. Формулировка требований к программе. Разработка структуры входных и выходных данных. Разработка алгоритма. Разработка модульной структуры программы. Разработка алгоритмов отдельных модулей. Разработка текста программы. Тестирование и отладка программы. Исполнение программы и анализ результатов. Сопровождение программы.

Тема 18. Эволюция и классификация языков программирования

Низкоуровневые языки программирования. Ассемблер. Процедурно-ориентированные языки: Фортран, Кобол, Алгол, Бейсик, Паскаль, Си. Языки объектно-ориентированного программирования: C++, Паскаль, Java. Функциональное программирование, язык LISP. Логическое программирование, язык PROLOG.

Тема 19. Понятие алгоритма и его свойства. Блок-схема алгоритма

Понятие алгоритма. Свойства алгоритма. Словесно-формульное представление алгоритма. Блок-схемы алгоритмов. Универсальный

алгоритмический язык (псевдокод). Данные алгоритмов – константы и переменные. Идентификаторы. Сложные типы данных – массивы и структуры.

Тема 20. Программы линейной структуры

Последовательное выполнение действий. Операторы присваивания, выражения, ввод-вывод данных, вызов вспомогательных алгоритмов.

Тема 21. Операторы циклов и ветвления. Базовые алгоритмы

Операторы if, if-else, switch, for, do-while, while или if-then, if-then-else, for-to-do, for-downto-do, case-of, repeat-until, while-do. Последовательное выполнение действий. Вычисление по последовательности формул. Вычисление конечных и бесконечных сумм и произведений. Расчет таблиц функциональных зависимостей. Подсчет числа положительных, отрицательных или нулевых элементов в одномерных и двумерных массивах.

Тема 22. Понятие о структурном программировании

Понятие программного модуля. Входные и выходные данные модуля. Иерархическая структура программы. Библиотеки модулей. Подпрограммы. Подпрограммы-функции и подпрограммы-процедуры. Механизм вызова подпрограмм. Возвращаемое значение. Принципы проектирования программ сверху вниз и снизу вверх. Достоинства и недостатки нисходящего и восходящего программирования. Комбинированный метод.

Тема 23. Объектно-ориентированное программирование

Понятие класса, данных и методов класса. Инкапсуляция. Объекты класса. Наследование, типы наследования. Видимость элементов базового класса. Полиморфизм.

Тема 24. Трансляция, компиляция и интерпретация

Язык программирования высокого уровня и язык машинных команд. Исходный модуль. Режимы компиляции и интерпретации. Объектный модуль. Компоновка объектных и библиотечных модулей. Исполняемый модуль. Переносимость исполняемых модулей. Программы-редакторы исходных текстов, компиляторы и редакторы связей.

Тема 25. Сетевые технологии обработки данных

Компьютерные сети. Серверы и рабочие станции. Узлы и ресурсы. Локальные и глобальные сети. Сетевые базы данных: архитектура «файл-сервер» и «клиент-сервер».

Тема 26. Основы компьютерной коммуникации

Топология сетей: кольцевая, звездообразная, шинная и древовидная конфигурация. Сетевые карты. Сетевые кабели. Концентраторы, коммутаторы и маршрутизаторы. Сетевые протоколы OSI: прикладной, уровень представления, сеансовый, транспортный, сетевой, канальный и физический уровень. Стандарт Ethernet.

Тема 27. Защита информации в локальных и глобальных компьютерных сетях

Основные понятия информационной безопасности. Угрозы безопасности информации и их классификация. Юридические основы информационной безопасности: понятие компьютерного преступления, соответствующие статьи УК. Объекты нападения, виды компьютерных преступлений. Компьютерные вирусы: классификация, каналы распространения, локализация, проявления действий. Критерии защищенности компьютерных систем. Организационные, инженерно-технические и другие меры защиты информации. Брандмауэр. Методы ограничения доступа. Мониторинг несанкционированных действий. Криптографические методы защиты данных. Методы шифрования. Электронная цифровая подпись электронных документов.

Тема 28. Сетевой сервис и сетевые стандарты

Глобальная сеть Интернет. Протоколы TCP/IP. IP-адрес и доменный адрес. Служба WWW. Протокол HTTP. Адрес URL. Протоколы SMTP, POP3 и IMAP4 для электронной почты и FTP для обмена файлами. DNS-сервис. Браузеры. Почтовые программы.

2. ОСНОВЫ РАБОТЫ В СРЕДЕ ОПЕРАЦИОННОЙ СИСТЕМЫ WINDOWS 7 РАБОТА С ФАЙЛОВОЙ СИСТЕМОЙ КОМАНДНАЯ СТРОКА В WINDOWS

Операционная среда Windows 7

Операционная система – это обязательная специальная программа, которая загружается при включении компьютера. Она ведет диалог с пользователем, осуществляет управление компьютером, его ресурсами (памятью, местом на дисках и т. д.), запускает другие прикладные программы на выполнение, обеспечивает корректный выход из программ и выключение компьютера. Операционная система – это удобный способ общения (интерфейс) пользователя с прикладными программами и устройствами компьютера.

Основной рабочей областью операционной системы Windows 7 является Рабочий стол (рис. 1).

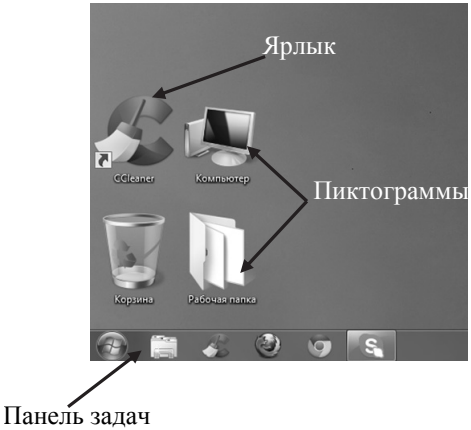


Рис. 1. Рабочий стол ОС Windows 7

Основными понятиями ОС Windows 7 являются: панель задач, папка, файл, приложение, ярлык, контекстное меню, списки переходов.

Файл – это поименованное количество информации, хранящейся в различных видах памяти с различной физической реализацией, однозначно определяемое соответствующей операционной системой.

Папка – поименованная часть раздела или область диска, содержащая некоторое количество файлов или другие папки.

Приложение или прикладная программа – программа, предназначенная для выполнения определенных пользовательских задач.

Ярлык – указатель на месторасположение информации на диске.

Путь к файлу – набор символов, показывающий расположение файла в файловой системе, адрес каталога.

Полный или ***абсолютный путь*** – это путь, который указывает на одно и то же место в файловой системе вне зависимости от текущего рабочего каталога (папки) или других обстоятельств. Полный путь всегда начинается с корневого каталога (папки).

Относительный путь представляет собой путь по отношению к текущему рабочему каталогу (папке) пользователя или активных приложений.

Буфер обмена – оперативная память компьютера, куда помещается информация на время, а затем удаляется. Файл clip.exe отвечает за хранение информации для буфера обмена.

Контекстное меню – вызывает набор дополнительных действий с объектом.

Панель задач – приложение, которое используется для запуска других программ или управления уже запущенными.

Списки переходов – это списки недавно открывавшихся объектов, таких как файлы, папки и веб-сайты, упорядоченные по программам, используемым для их открытия. Список переходов позволяет не только открывать объекты, но и закрепить избранное, обеспечивает быстрый доступ к часто используемым материалам.

Архивация файла – это процесс преобразования информации, хранящейся в файле, к виду, при котором уменьшается избыточность в ее представлении и соответственно требуется меньший объем памяти для хранения.

Сжатие информации в файлах производится за счет устранения избыточности различными способами, например, за счет упрощения кодов, исключения из них постоянных битов или представления повторяющихся символов или повторяющейся последовательности символов в виде коэффициента повторения и соответствующих символов. Применяются различные алгоритмы подобного сжатия информации.

Сжиматься могут как один, так и несколько файлов, которые в сжатом виде помещаются в так называемый архивный файл или архив.

Архивный файл – это специальным образом организованный файл, содержащий в себе один или несколько файлов в сжатом или несжатом виде и служебную информацию об именах файлов, дате и времени их создания или модификации, размерах и т. п.

Программы, осуществляющие упаковку и распаковку файлов, называются ***программами-архиваторами***.

Самораспаковывающийся архивный файл – это загрузочный исполняемый модуль, который способен к самостоятельной разархивации находящихся в нем файлов без использования программы-архиватора. Самораспаковывающийся архив получил название SFX-архив (SelF-eXtracting).

Для создания файла (SFX-архив) необходимо выполнить следующие действия.

1. Выбрать в контекстном меню для выбранного файла команду *Добавить в архив.*
2. Выбрать имя файла.
3. В окне *Имя и параметры архива* на вкладке *Общие* в разделе *Параметры архивации* пометить флажок *Создать SFX-архив.*
4. Нажать на кнопку ОК.

Приложение Проводник

Для работы с файлами и папками в ОС Windows 7 используется приложение Проводник. Программа Проводник является основным инструментом для файловых операций в системе Windows. Она отображает содержимое папок, позволяет копировать, перемещать, удалять, переименовывать папки и файлы, запускать программы.

Рабочая область окна приложения Проводник состоит из двух областей: области дерева папок и устройств (слева) и области активной (открытой) папки (справа). В любой момент времени только одна папка может быть открыта (рис. 2).

Кроме программы **Проводник** в составе операционной среды **Windows** имеется много других служебных и сервисных программ, существенно повышающих удобство работы пользователя. Часть таких программ входит в группу **Стандартные**. Назначение некоторых из них приведено ниже.

Paint – графический редактор для создания несложных рисунков, схем.

Блокнот – простейший текстовый редактор.

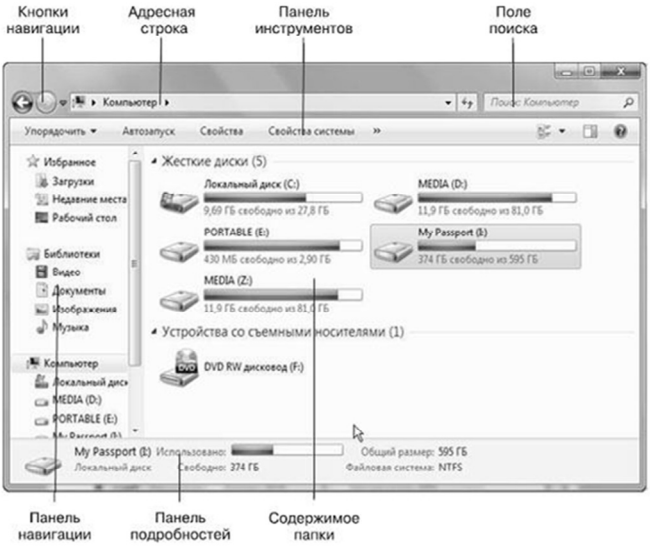


Рис. 2. Главное окно приложения Проводник

WordPad – текстовый редактор для создания документов средней сложности.

Калькулятор – программа для выполнения простейших расчетов.

Поскольку все **Windows**-приложения имеют единый пользовательский интерфейс, общие приемы работы в этих программах аналогичны рассмотренным выше. Конкретную подсказку по работе с той или иной программой можно получить, обратившись к справочной системе программы, вызываемой командой строчного меню ***Справка (?)***.

Основные возможности приложения Проводник представлены в табл. 1.

Таблица 1

Возможности приложения Проводник

Функция	Описание
Кнопки навигации	Перемещение между страницами, которые уже посетили. Эти кнопки работают в сочетании с адресной строкой
Адресная строка	Адресная строка используется для перехода к другой папке или библиотеке либо для возвращения к предыдущей папке или библиотеке

Окончание табл. 1

Функция	Описание
Поле поиска	Позволяет быстро найти любой файл, находящийся в открытой или в одной из вложенных папок
Строка меню	Показывает классические меню папок. По умолчанию строка меню скрыта
Строка состояния	Показывает настройки и статистику для выделенных объектов. Разделена на фрагменты, которые содержат различные сведения.
Заголовки столбцов	Определяет, какая информация должна отображаться в режиме Таблица; изменяет внешний вид и организацию файлов в списке
Панель инструментов	Позволяет выполнять часто используемые типичные задачи
Область содержимого папки	Отображает значки файлов и папок, находящихся в открытой папке
Панель навигации	Позволяет перейти к папке с нужными файлами
Панель подробностей	Используется для просмотра наиболее общих свойств, связанных с выбранным файлом

Командная строка

Командная строка – это интерфейс взаимодействия пользователя с компьютером, в котором команды отдаются путем ввода текстовых строк при помощи клавиатуры.

В современных компьютерных устройствах с графическим интерфейсом командная строка выглядит анахронизмом. Ее использование, на первый взгляд, кажется неоправданно сложным и устаревшим особенно пользователям, которые начали знакомство с компьютерами с графического режима. Тем не менее есть целый ряд случаев, когда работа с командной строкой является желательной и даже необходимой:

1) работа с программами, которые не имеют графического интерфейса;

2) работа с удаленными устройствами при минимальных затратах трафика;

3) работа с большими объемами информации, когда отсутствие элементов графического интерфейса позволяет вместить больше данных на странице;

4) работа одновременно с несколькими файлами, в которых необходимо с высокой скоростью выполнять различные команды.

Диалог пользователя в Командной строке осуществляется в форме команд. Это значит, что для выполнения того или иного действия пользователь должен в командной строке набрать нужную команду и нажать клавишу **Enter**. Когда операционная система к диалогу готова, она выдает приглашение, содержащее, как правило, информацию о текущем диске, текущем каталоге и заканчивающееся символом ">".

Например:

X:\MTF> – текущий диск **X:**, текущий каталог **MTF**

Совокупность команд с учетом правил их записи составляет командный язык операционной системы. Команды в **MS DOS** имеют следующий формат записи:

<команда> [<опции>],

где опции – необязательные параметры, уточняющие команду. К ним относятся диск, маршрут, имя файла или каталога, ключ.

Маршрут – это последовательность из имен каталогов, разделенных символом "\" (обратный слэш). Разделителем команд и опций является пробел.

Синтаксис некоторых наиболее часто используемых команд

1. *Установление текущего диска*

<имя диска>:

Например: A: – объявление текущим диска A:.

2. *Просмотр каталога*

dir [<маршрут\>]<файл> [/p]

В данной команде, если имя файла не задано, то выводится все оглавление каталога. Иначе – только сведения о файле. Ключ /p задает поэкранный вывод.

Примеры записи команды:

dir – вывод оглавления текущего каталога;

dir X:\MTF /p – поэкранный вывод оглавления каталога MTF на диске X:.

3. *Создание каталога*

md [<маршрут>\]<каталог>

Например: md X:\MTF – создание каталога MTF в корневом каталоге диска X:.

4. *Удаление каталога*

rd [<маршрут>\]<каталог>

Например: rd X:\MTF – удаление каталога MTF, находящегося в корневом каталоге диска X:.

5. *Смена текущего каталога*

cd [<маршрут>\]<каталог>

Например: cd X:\MTF – переход в подкаталог MTF корневого каталога диска X:.

6. *Просмотр дерева каталогов на заданном диске*

tree [<имя диска>]

Например: tree X: – вывод дерева каталогов диска X:.

7. *Создание файлов*

copy con [<диск>][<маршрут\>]<файл-приемник>

con – логическое устройство консоль (клавиатура + монитор).

После выполнения команды и записи информации в файл необходимо завершить создание файла – комбинация **Ctrl+z,** после чего обязательно нажать клавишу **Enter.**

8. *Копирование файлов*

copy [<диск>][<маршрут\>]<файл-источник>
[<диск>][<маршрут\>]<файл-приемник>

Примеры записи команды copy.

copy X:\MTF\КМ-11\abc.txt X:\MTF\def.txt – копирование файла abc.txt из каталога КМ-11 в каталог MTF под именем def.txt.

copy *.* X:\ – копирование всех файлов текущего каталога в корневой каталог диска X:.

9. *Удаление файлов*

del [<диск>:][<маршрут\>]<файл>

В именах файлов можно употреблять символы "*****" и "**?**".

Например:

del X:\MTF\def.txt – удаление файла def.txt из каталога MTF диска X:

del X:\MTF*.txt – удаление всех файлов, имеющих расширение.txt из каталога MTF на диске X:

10. *Переименование файлов*

ren [<диск>:][<маршрут\>]<файл1> <файл2>

Например: ren X:\MTF\def.txt abc.txt – переименование файла def.txt из каталога MTF диска X: в файл abc.txt, который будет расположен в текущем каталоге.

Процесс взаимодействия пользователя посредством командного языка достаточно сложен и неудобен. Для его упрощения используются специальные программы (операционные оболочки), которые создают качественно новую среду, предоставляющую пользователю более наглядные и более удобные средства. Большую популярность получила оболочка FAR (рис. 3).

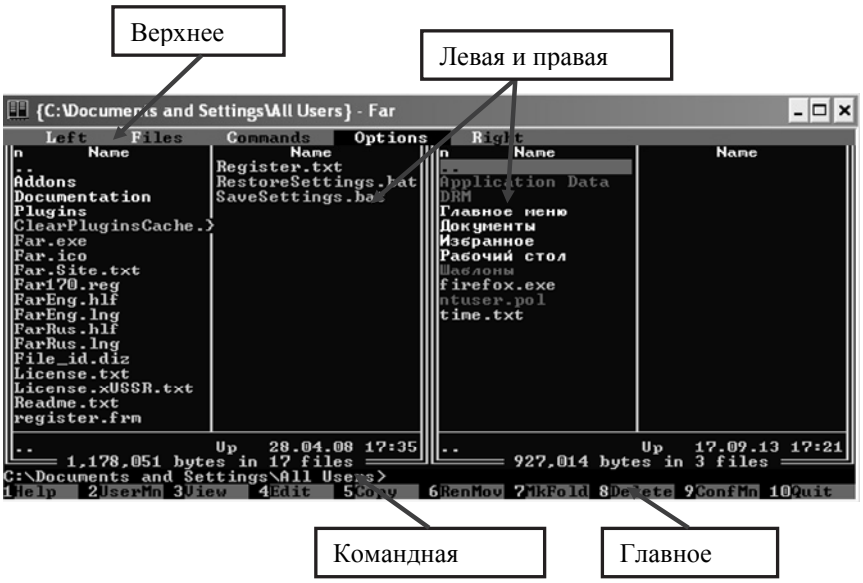


Рис. 3. Внешний вид интерфейса программной оболочки FAR

Действия **FAR** при нажатии функциональных клавиш **F1–F10** определены в главном меню оболочки.

F1 – вызов на экран справки по **FAR** (**Help**);

F2 – вызов на экран пользовательского меню;

F3 – просмотр файла на экране;

F4 – редактирование выделенного файла. При одновременном нажатии клавиш **Shift +F4** создается новый файл с возможностью его последующего редактирования;

F5 – копирование файлов и каталогов;

F6 – переименование (перемещение) файлов и каталогов;

F7 – создание каталога;

F8 – удаление файлов и каталогов;

F9 – вызов верхнего меню;

F10 – выход из оболочки.

Нажатие клавиши **Enter** при активном верхнем меню и кнопкой мыши приводит к появлению на экране дисплея еще одного, вертикального, меню, состав команд которого зависит от того, какой пункт был выделен в верхнем меню. Содержанием вертикальных меню являются команды оболочки **FAR**. Назначение пунктов верхнего меню следующее.

Левая – содержит команды управления левой панелью, позволяющие менять вид и форму вывода информации в панели, производить смену диска.

Файл – содержит команды манипулирования файлами и каталогами. Некоторые из этих команд продублированы в главном меню, другие предназначены для установки атрибутов файлов и выделения групп файлов.

Диск – содержит команды работы с дисками.

Команды – содержит разнообразные команды, с помощью которых пользователь может осуществлять поиск файлов на диске, сравнивать каталоги, включать/отключать панели и др. Кроме того, здесь же содержатся команды конфигурирования оболочки, с помощью которых устанавливаются внешний вид экрана и режимы работы **FAR**.

Правая – содержит команды управления правой панелью.

Более подробно о назначении команд можно узнать, обратившись к функции Помощь (нажав клавишу **F1**).

Контрольные вопросы

1. Назовите определение и назначение операционной системы.
2. Для чего необходимы папки файловой системы ОС Windows?

3. Что такое ярлык приложения и документа? Назовите назначение ярлыка.

4. В чем сходство и в чем различие структуры окон различных стандартных программ Windows 7?

5. Что такое список переходов и его назначение?

6. Что называют Контекстным меню? Что содержит Контекстное меню объекта операционной системы?

7. Для чего необходима Панель задач? Каковы ее основные возможности?

8. Для чего используется приложение Проводник? Назовите основные возможности приложения.

9. Для чего необходим буфер обмена?

10. Назначение приложения Командная строка. Назовите наиболее часто применяемые команды приложения.

11. Что такое путь к файлу? Чем отличается абсолютный и относительный путь к файлу?

12. В каких ситуациях необходимо приложение Командная строка?

13. Что такое архивация файлов?

14. Для чего используют самораспаковывающийся архив?

3. РАБОТА С ПАКЕТОМ MS OFFICE 2007

3.1. Текстовый редактор MS Word Создание сложного документа Специальные возможности

Программа Microsoft Word предназначена для подготовки и редакторской правки больших по объему и сложных по структуре документов. Структура окна приложения Microsoft Word представлена на рис. 4. Кроме текста в состав документа могут входить схемы, рисунки, таблицы, формулы.

Некоторые понятия редактора Microsoft Word

Абзац – фрагмент текста от одного нажатия клавиши ENTER до следующего. В ячейках таблицы абзацем является фрагмент от начала ячейки до ближайшего нажатия клавиши ENTER или знака конца ячейки.

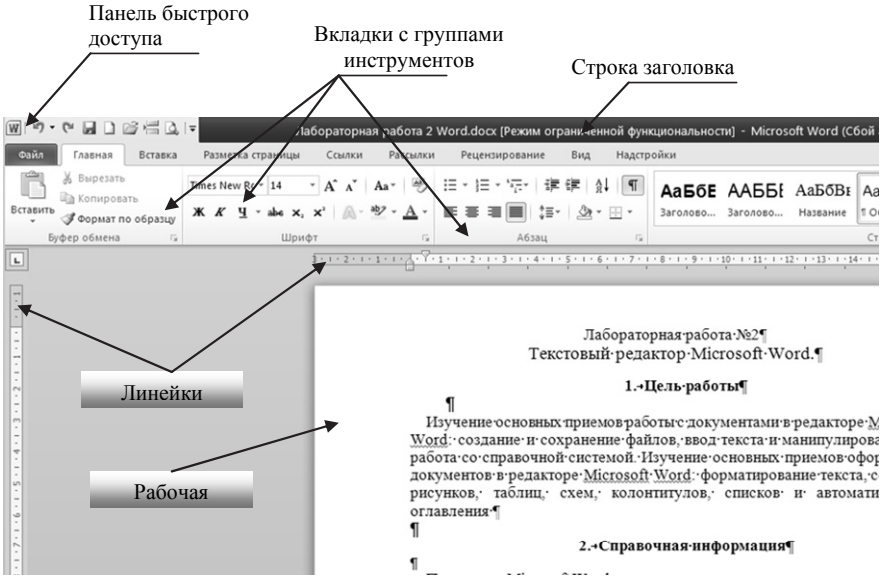


Рис. 4. Структура окна Microsoft Word

При установке параметров для одного абзаца выделять его необязательно. Достаточно, если в этом абзаце будет находиться курсор. Если же оформляется сразу несколько абзацев, их необходимо выделить.

Раздел – часть документа, в которой применяется определенный формат страницы. Например, для изменения количества колонок необходимо вставить в документ новый раздел с помощью команды вкладки Вставка/Разрыв.

Списки – это фрагменты текста, пункты которого отмечены специальными знаками. Списки могут быть маркированными, нумерованными и многоуровневыми.

Колонтитулы – это небольшие идентификаторы, которые указываются вверху и внизу во всем документе и содержат важные сведения о нем. В них содержатся такие сведения, как номера страниц, даты, название книги или главы, а также имя автора.

Стили – это наборы различных вариантов форматирования, которые отображаются в виде эскизов в коллекции экспресс-стилей. Коллекция экспресс-стилей используется для быстрого применения стилей к абзацам или фрагментам текста документа.

Форматирование – оформление текста, связанное с изменением его внешнего вида. Различают три основные операции форматирования:

- форматирование абзацев – изменение полей абзацев, изменение интервалов между строками и выравнивание абзацев;
- форматирование символов – изменение шрифта отдельных букв, слов, предложений и абзацев;
- форматирование страниц – выбор размера, ориентации и полей страниц.

Использование формул в таблицах

Вычисления и логические сравнения в таблицах, созданных в текстовом редакторе, можно выполнять с помощью формул. Команда ***Формула*** находится в разделе Работа с таблицами на вкладке Макет в группе Данные. Основные функции для работы с данными представлены в табл. 2.

Таблица 2

Основные функции для работы с данными

Основные функции	Описание
=AVERAGE()	Вычисление среднего арифметического
=COUNT()	Подсчет количества
=MAX(), =MIN()	Нахождение максимума, минимума
=SUM()	Подсчет суммы
Ключевое слово	***Местонахождение данных***
BELOW, ABOVE	Под ячейкой и над ячейкой
LEFT, RIGHT	Слева и справа от ячейки

Сноски предназначены для добавления к тексту комментариев, объяснений, указания источника информации. Сноски бывают обычными (в конце страницы) и концевыми (в конце всего текста).

Оглавление – это список заголовков документа. Для того чтобы быстро сделать оглавление, документ должен быть отформатирован согласно встроенным форматам уровней структуры или стилей заголовков.

Гиперссылка – это выделенный цветом и подчеркнутый текст или графический элемент, на который можно нажать для открытия одного из следующих элементов:

- файла;
- местоположения в файле;

- веб-страницы в Интернете;
- веб-страницы в интрасети;
- узла Gopher, Telnet или FTP.

Гиперссылки создают для удобного использования объемных документов и дают возможность пользователям быстро переходить к нужному месту в документе (раздел, глава, пункт или же специально отмеченная страница, сноска).

Контрольные вопросы

1. Для чего необходим текстовый редактор MSWord?
2. Что может содержать сложный документ?
3. Объясните основные понятия документа: абзац, раздел, форматирование.
4. Для чего используют гиперссылки в текстовом документе?
5. Каковы возможности использования формул в таблицах текстового редактора?
6. В чем отличие понятий редактирования и форматирования?
7. Назначение колонтитулов в текстовом редакторе.

3.2. Изучение возможностей редактора создания презентаций Power Point

Презентация – мультимедийный инструмент, используемый в ходе докладов или сообщений для повышения выразительности выступления, более убедительной и наглядной иллюстрации описываемых фактов и явлений. Microsoft Office PowerPoint – программа для создания и проведения презентаций.

MS PowerPoint позволяет создавать наглядные презентации, интегрируя текст, графику, видео и другие элементы на отдельных страницах, называемых «слайдами». MS PowerPoint дает возможность создавать слайды, содержащие перемещаемые таблицы и обтекающий текст, а также редактировать, демонстрировать и распечатывать слайды.

Инструменты и возможности

Тема презентации (Вкладка Дизайн – Тема)

Тема – набор унифицированных элементов, определяющих внешний вид документа с помощью цвета, шрифтов и графических объек-

тов. Внешний вид презентации можно легко изменить в любой момент, применив другую тему (рис. 5).

Рис. 5. Темы

Макет слайда (вкладка Главная – Макет)

Макеты слайдов определяют форматирование, размещение и заполнители для всего содержимого на слайде (рис. 6).

Рис. 6. Макеты слайдов

Добавление гиперссылок (Вкладка Вставка – Гиперссылка)

Инструмент Гиперссылка предназначен для перехода с одного слайда на другой, к ресурсу в локальной сети или в Интернете либо даже к другому файлу или программе.

Для создания ссылки необходимо:

- выделить текст, по которому нужно нажать для активации гиперссылки либо можно выделить объект (например, клип или рисунок SmartArt);
- в группе Ссылки вкладки Вставка щелкнуть по элементу Гиперссылка;
- в диалоговом окне Вставка гиперссылки нажать соответствующую кнопку в поле Мои адреса, чтобы задать назначение ссылки (т. е. место, на которое указывает ссылка).

Эффекты анимации слайдов (Вкладка Анимация)

Анимационная схема – предопределенный тип смены слайдов и библиотека анимационных эффектов, применяемых к объектам слайда (рис. 7).

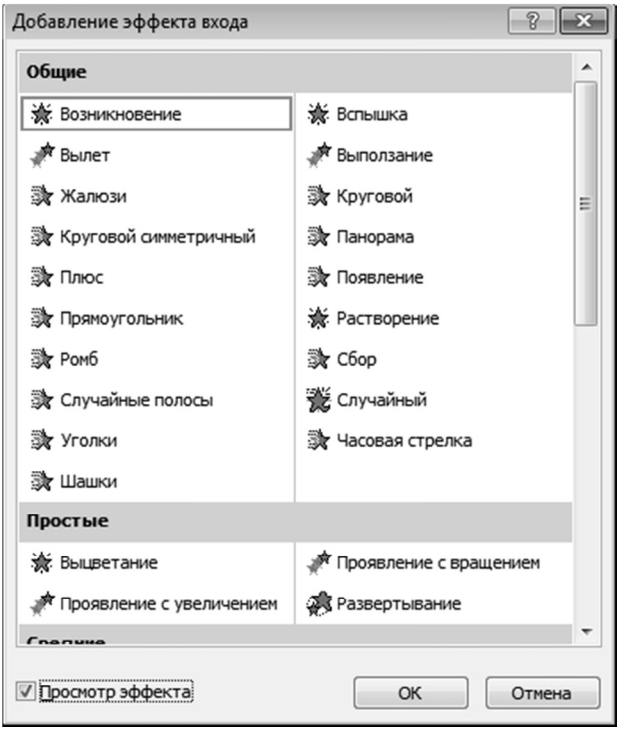


Рис. 7. Эффекты анимации

Использование технологии триггеров

«Триггер» (англ.) – спусковой крючок. С помощью триггеров в программе Power Point можно пользоваться технологией «горячих зон», когда, не меняя слайд, мы изменяем какой-то его отдельный фрагмент.

Триггер – это интерактивное средство анимации, позволяющее задать действие выделенному элементу, анимация запускается по щелчку. Через триггер происходит запуск анимационного эффекта или группы эффектов. Триггер можно применить к любому объекту на слайде. Он, как и управляющая кнопка, срабатывает при наведении курсора по щелчку левой кнопки мышки, при этом в момент наведения сам курсор меняет внешний вид на «ладошку».

Пример использования триггера

Задача. На слайде находятся два объекта (прямоугольник и треугольник), и необходимо появление одного объекта (прямоугольника) при нажатии на другой объект (треугольник).

Порядок действий

1. Поместить на слайд две фигуры (прямоугольник и треугольник).

2. Выбрать для первого объекта необходимый эффект анимации на вкладке Настройка анимации (рис. 8).

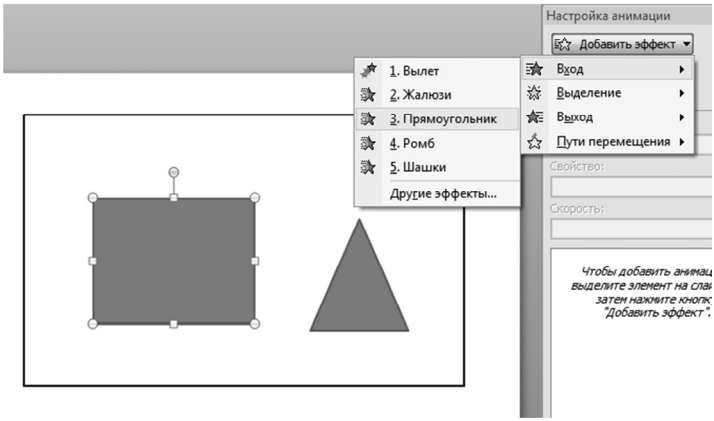


Рис. 8. Настройка анимации

3. Настроить параметры эффекта для прямоугольника, выбрав команду Параметры эффектов из контекстного меню. В появившемся

окне Прямоугольник на вкладке Время нажать кнопку Переключатели. Для переключателя Начать выполнение эффекта при щелчке выбрать из списка нужный объект (треугольник) и нажать ОК (рис. 9).

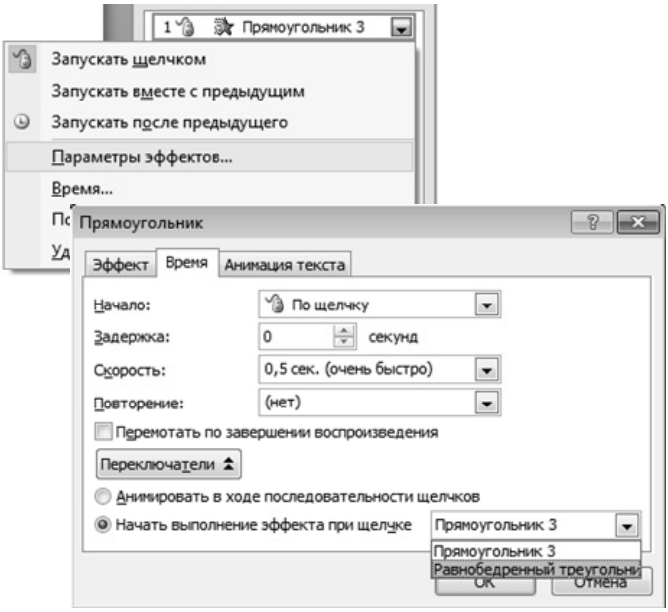


Рис. 9. Настройка эффектов анимации

4. При запуске презентации на слайде будет изображен треугольник, при нажатии на который (курсор мыши при наведении на объект выглядит в форме ладошки) появится нужный объект (прямоугольник).

Инструмент WordArt (Вкладка Вставка – WordArt)

WordArt – это коллекция стилей текста, которые можно добавлять в документы Office для создания декоративных эффектов (см. рис. 9).

Создание специальных эффектов при помощи инструмента WordArt осуществляется с панели Вставка – Текст, для этого необходимо:

- отобразить нужный слайд в режиме слайдов, а затем щелкнуть на вкладке Вставить кнопку WordArt и выбрать стиль текста;
- создать необходимый текст в появившемся на слайде текстовом блоке;

• настроить текст надписи, используя стили WordArt: тень, отражение, рельеф и т. д.

Инструмент SmartArt (Вкладка Вставка – SmartArt)

Рисунок SmartArt – это настраиваемое средство, позволяющее передать информацию зрительным образом (рис. 10).

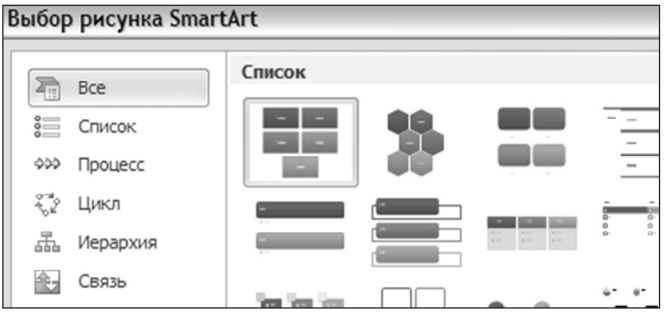


Рис. 10. Выбор рисунка SmartArt

Использование кнопок управления

Для создания кнопок управления необходимо:

1) создать управляющую кнопку, используя Вкладку Вставка – Фигуры – Управляющие кнопки;

2) в диалоговом окне Настройка действия выбрать переключатель Перейти по гиперссылке – номер слайда (рис. 11).

Примечание: если кнопка создана без настройки действия, то изменения можно сделать, выбрав из контекстного меню пункт Изменить гиперссылку.

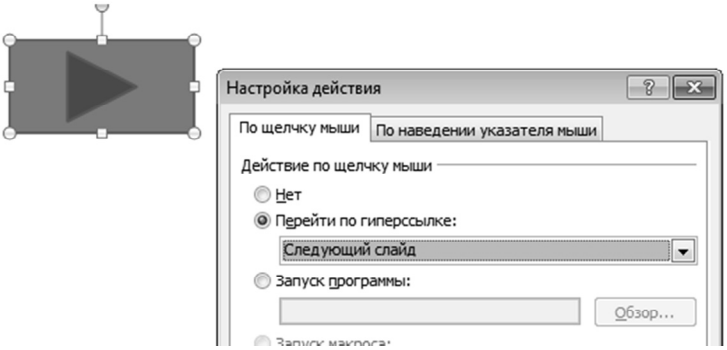


Рис. 11. Работа с управляющей кнопкой

Некоторые правила оформления презентации

Правила шрифтового оформления:

- размер шрифта: 24–54 пункта (заголовок), 18–36 пунктов (обычный текст);
- цвет шрифта и цвет фона должны контрастировать (текст должен хорошо читаться), но не резать глаза;
- тип шрифта: для основного текста гладкий шрифт без засечек (Arial, Tahoma, Verdana), для заголовка можно использовать декоративный шрифт, если он хорошо читаем;
- курсив, подчеркивание, жирный шрифт, прописные буквы рекомендуется использовать только для смыслового выделения фрагмента текста.

Графическая информация:

- рисунки, фотографии, диаграммы призваны дополнить текстовую информацию или передать ее в более наглядном виде;
- желательно избегать в презентации рисунков, не несущих смысловой нагрузки, если они не являются частью стилевого оформления;
- цвет графических изображений не должен резко контрастировать с общим стилевым оформлением слайда;
- иллюстрации рекомендуется сопровождать пояснительным текстом;
- если графическое изображение используется в качестве фона, то текст на этом фоне должен быть хорошо читаем.

Правила выбора цветовой гаммы:

- цветовая гамма должна состоять не более чем из двух-трех цветов;
- существуют несочетаемые комбинации цветов;
- черный цвет имеет негативный (мрачный) подтекст;
- белый текст на черном фоне читается плохо (инверсия плохо читается).

Контрольные вопросы

1. Для чего используется редактор создания презентаций MS Power Point?
2. Какие режимы просмотра презентации вы знаете?
3. Какие операции можно выполнять на вкладке Структура в левой панели экрана?

4. Как можно добавить колонтитул в нижнюю часть слайда и какие поля он содержит?

5. Как изменить цветовую схему слайдов презентации?

6. Как добавить в слайд звуковые и видеофайлы?

7. Как добавить в слайд графические объекты?

8. Как установить анимацию для перехода от одного слайда к другому?

9. Как установить анимацию для всех объектов слайда?

10. Какое назначение триггеров в MS Power Point?

3.3. MS Excel. Основные приемы работы с книгами, ячейками, листами, формулами Работа со встроенными функциями

Microsoft Excel – программа, предназначенная для работы с электронными таблицами. Позволяет вести экономические расчеты, строить отчеты, графики, диаграммы.

Основными элементами Microsoft Excel являются: ячейка, диапазон ячеек, строка, столбец, адрес ячейки, формула, абсолютная и относительная адресация (рис. 12).

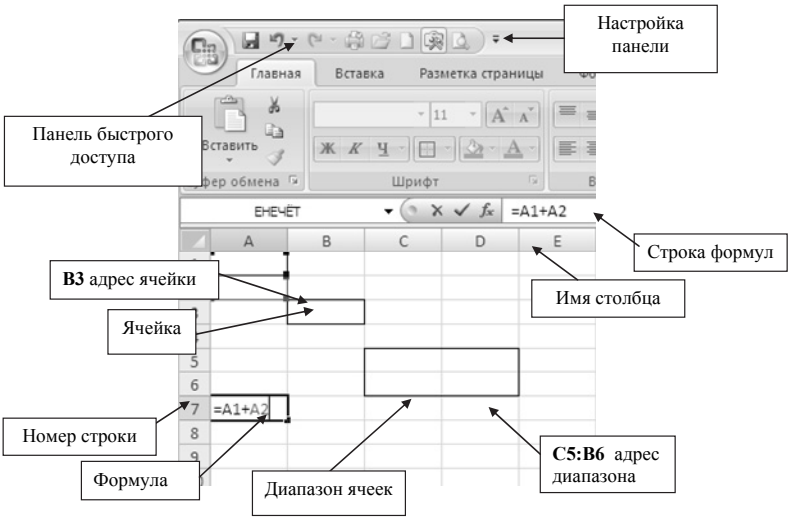


Рис. 12. Основные элементы MS Excel

Ячейка в MS Excel – это основной элемент электронной таблицы, образованный пересечением столбца и строки.

Адрес ячейки – имя столбца и номер строки, на пересечении которых находится ячейка, представляют собой **координаты**, определяющие расположение этой ячейки на листе.

Диапазон ячеек – это группа рядом стоящих ячеек, образующая прямоугольник.

Абсолютная адресация или абсолютные ссылки – ссылки, которые при копировании в составе формулы в другую ячейку не изменяют, например C3.

Абсолютные ссылки используются в формулах, когда нежелательно автоматическое изменение ссылки при копировании.

Относительная адресация или относительные ссылки – ссылки, которые при копировании в составе формулы в другую ячейку автоматически изменяются.

Формулой Excel считается все, что начинается со знака "=".

В формуле можно использовать различные типы операторов (арифметические и т. п.), текст, ссылки на ячейку или диапазон ячеек, круглые скобки, именованные диапазоны. В формулах соблюдается приоритет выполнения операций (умножение выполняется раньше сложения и т. п.). Для изменения порядка выполнения операций используются круглые скобки.

Использование текста в формулах

Если в формуле используется текст, то он **обязательно** должен быть заключен в двойные кавычки.

Если написать формулу «=мама», Excel выдаст ошибку, а если написать «="мама"» – все ОК – корректная формула.

Использование ссылок в формулах

Для того чтобы вставить в формулу адрес ячейки (ссылку на ячейку), не обязательно писать его вручную. Проще поставить знак «=», затем левой кнопкой щелкнуть на нужной ячейке или выделить нужный диапазон ячеек.

Контрольные вопросы

1. Дайте понятие основным группам элементов главного окна MS Excel: строка заголовка, строка меню, панели инструментов, строка состояния, окно документа, полосы прокрутки (удалите полосы прокрутки, добавьте полосы прокрутки).

2. Дайте понятие основным элементам структуры документа MS Excel: рабочая книга, рабочий лист, ячейка, диапазон ячеек, адрес ячейки.

3. Что такое форматирование ячеек?

4. Что такое относительная и абсолютная адресация?

5. Назовите правила выполнения простых вычислений.

6. Перечислите этапы использования встроенных функций.

3.4. MS Excel. Построение диаграмм и графиков Работа со списками

Диаграммы (понятие, назначение). Объекты диаграмм

Диаграмма – объект электронной таблицы, наглядно показывающий соотношение каких-либо величин.

Назначение диаграммы: графическое отображение данных для анализа и сравнения. Объекты диаграммы в MS Excel представлены в табл. 3.

Таблица 3

Объекты диаграммы в MS Excel

№ п/п	Объект диаграммы	Описание
1	Область диаграммы	Прямоугольная область, на которой отображаются элементы диаграммы
2	Область построения диаграммы	Прямоугольная область, ограниченная осями
3	Линии сетки	Линии, которые начинаются с делений шкалы. Как правило, их прибавляют к оси значений, чтобы идентифицировать значение точек данных
4	Заголовки	Название диаграммы, название оси значений, название оси категорий
5	Ось категорий	Ось, на которой отображаются заголовки из листа с таблицей, как правило, горизонтальная
6	Ось значений	Ось, на которой располагаются значения данных из листа, по обыкновению вертикальная
7	Метки данных	Добавляются в диаграмму для отображения конкретного значения точки данных
8	Точка данных	Элемент ряда данных, который соответствует значению одной ячейки в листе

Окончание табл. 3

№ п/п	Объект диаграммы	Описание
9	Шкала	Цифровые метки на оси значений. Минимальное значение равняется нулю
10	Метки делений	Маленькие линии, которые отображают разделение шкалы на осях
11	Легенда	Текстовое поле с описанием рядов данных
12	Ряд данных	Строка или столбец данных из листа. Названия всех рядов приводятся в легенде
13	Таблица данных	Таблица, в которой отображаются входные данные диаграммы
14	Маркеры выделения	Если щелкнуть по объекту, возле него появляется несколько черных квадратов, которые показывают, что объект выбран. Объект выбирают для перемещения или редактирования

Типы диаграмм

В MS Excel имеется возможность выбора из нескольких типов диаграмм, причем каждый тип имеет несколько разновидностей (видов). MS Excel включает в себя следующие основные типы диаграмм.

Гистограмма – отображает значения различных категорий, позволяет показать несколько рядов данных.

Линейчатая – отображает также значения для различных категорий, отличается ориентацией осей *Ox* и *Oy*.

График – представлен в виде точек данных, соединенных тонкой линией, позволяет проследить изменения предложенных данных, показать несколько рядов данных.

Круговая и кольцевая диаграммы – можно показать только один ряд данных. Выбор данного типа определяется соображением целесообразности и наглядности.

Диаграмма с областями – отображает изменение значений ряда с течением времени, а также изменение вклада отдельных значений.

Понятие списка в Excel

Списком называется таблица Excel, которая состоит из одного и более столбцов. Столбцам списка присваиваются уникальные имена полей, которые заносятся в первую строку списка. Все ячейки в столбце имеют один и тот же формат данных, поэтому все строки, или, как

их еще называют, записи, однотипны. Пример списка представлен на рис. 13.

Фамилия	Возраст	Пол
Петухова	17	ж
Петров	17	м
Зайцева	17	ж

Рис. 13. Пример списка

Работа со списками в Excel

Большинство операций, предназначенных для работы со списками, сосредоточено на вкладке ***Данные***. Если список создан правильно, то достаточно выделить одну из ячеек внутри списка и нажать нужную команду в меню Данные. Excel автоматически определит границы вашего списка.

Инструмент ***Сортировка*** позволяет осуществить сортировку по выбранному критерию по одному или, в порядке приоритета, по двум или даже трем полям списка.

Инструмент ***Фильтр*** дает возможность показывать только те записи в списке, которые удовлетворяют некоторому критерию.

Инструмент ***Консолидация*** используется для подведения итогов и составления отчета по результатам нескольких листов, можно консолидировать данные из отдельных листов в основном листе. Листы могут находиться в той же книге, что и основной лист, или в других книгах. При консолидации данных они компонуются так, что их становится проще обновлять и обобщать на регулярной основе или по требованию.

Сводные таблицы Excel

Сводные таблицы позволяют осуществлять групповые операции над данными, находящимися либо в списках, либо в нескольких диапазонах консолидации, либо во внешних базах данных.

Контрольные вопросы

1. Для чего нужны диаграммы и графики в Microsoft Excel, с помощью чего можно создать диаграммы и графики?

2. Каковы основные объекты диаграммы?

3. Расскажите основные шаги построения диаграммы.

4. Что такое список? Для какой информации используют список?

5. Назовите основные правила создания списков.

6. Для каких задач используют команду «Сводная таблица» для списков?

3.5. MS Excel. Решение линейных уравнений, нелинейных уравнений и систем уравнений Работа с матрицами

Алгоритм нахождения корней полинома на примере полинома $x^3 - 0,01x^2 - 0,7044x + 0,139104 = 0$

1. Решить уравнение графически.

а) Провести табулирование (введение значений переменной x в ячейки A2:A12 полинома на интервале от –1 до 1 с шагом 0,2.

б) В ячейку B2 поместить формулу для вычисления функции $f(x)$: = A2^3 – 0,01 · A2^2 – 0,7044 · A2 + 0,139104 и скопировать для всех переменных x (рис. 14).

2. Построить график функции $f(x)$.

3. Проанализировать результаты построенного графика: определены интервалы, на которых находятся корни полинома: [–1, –0.8], [0.2,0.4] и [0.6, 0.8] или приближенные начальные значения.

4. Ввести приближенные начальные значения, взятые из полученных интервалов выше в ячейки A14:A16.

5. Найти корни полинома методом последовательных приближений с помощью команды ***Сервис → Подбор параметра*** (Настройка панели быстрого доступа → Другие команды → Подбор параметра). Относительная погрешность вычислений и предельное число итераций (например, 0,00001 и 1000) задаются на вкладке ***Сервис → Параметры*** (Параметры Excel → Формулы).

6. Заполнить диалоговое окно Подбор параметра, следуя рис. 15.

В поле **Установить в ячейке** дается ссылка на ячейку, в которую введена формула, вычисляющая значение левой части уравнения для приближенного значения корня полинома ячейки A14.

	A	B
1	**x**	**y(x)**
2	-1	-0,166
3	-0,8	0,184
4	-0,6	0,342
5	-0,4	0,355
6	-0,2	0,272
7	0	0,139
8	0,2	0,006
9	0,4	-0,080
10	0,6	-0,071
11	0,8	0,081
12	1	0,425
13	**Корни полинома**	**Значение функции**
14	-0,9	0,035964
15	0,3	-0,046116
16	0,7	-0,015876
17		

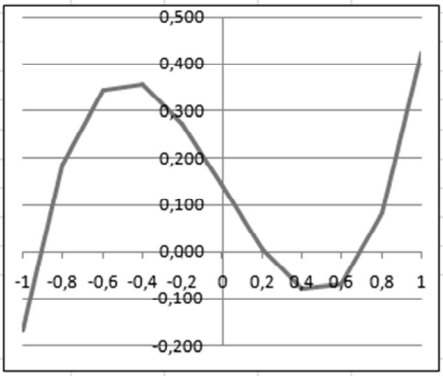


Рис. 14. Решение уравнения

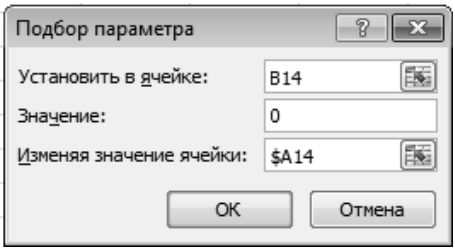


Рис. 15. Подбор параметра

В поле **Значение** вводим правую часть уравнения, а в поле **Изменяя значение ячейки** дается ссылка на ячейку, отведенную под переменную.

После нажатия кнопки ОК появится диалоговое окно **Результат подбора параметра** (рис. 16) с сообщением об успешном завершении поиска решения, приближенное значение корня будет помещено в ячейку A14.

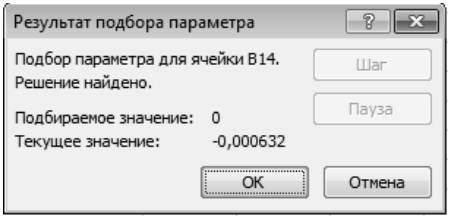


Рис. 16. Результат подбора параметров

Два оставшихся корня находим аналогично. Результаты вычислений будут помещены в ячейки строк 15 и 16 (рис. 17).

	Корни полинома	Значение функции
13		
14	-0,92034081	-0,000632
15	0,21021354	-0,000123
16	0,7207183	0,000602

Рис. 17. Корни полинома

Решение систем линейных алгебраических уравнений (СЛАУ)

Пусть задана СЛАУ следующего вида:

$$a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1,$$

$$a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2,$$

$$\ldots$$

$$a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n = b_n.$$

Эту систему можно представить в матричном виде: $AX = b$, где

$$A = \begin{pmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ & & \ldots & \\ a_{n1} & a_{n2} & \ldots & a_{nn} \end{pmatrix}$$ – матрица коэффициентов системы уравнений;

$$X = \begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{pmatrix}$$ – вектор неизвестных; $$b = \begin{pmatrix} b_1 \\ b_2 \\ \dots \\ b_n \end{pmatrix}$$ – вектор правых частей.

Метод обратной матрицы

Систему линейных алгебраических уравнений $A \cdot X = b$ умножим слева на матрицу, обратную к A. Система уравнений примет вид

$$A^{-1} \cdot A \cdot X = A^{-1} \cdot b, \quad E \cdot X = A^{-1} \cdot b, \ (E - \text{единичная матрица})$$

Таким образом, вектор неизвестных вычисляется по формуле $X = A^{-1} \cdot b$.

Метод Крамера

В этом случае неизвестные $x_1, x_2, \dots, x_n$ вычисляются по формуле

$$x_i = \frac{\Delta_i}{\Delta}, \ i = 1, ..., n,$$

где x_1 определяется как произведение определителя матрицы A на определитель матрицы, получаемой из матрицы A путем замены i-го столбца вектором b.

Алгоритм решения системы уравнений методом обратной матрицы на примере системы

$$\begin{cases} x_2 - 13x_3 + 4x_4 = -5, \\ x_1 - 2x_3 + 3x_4 = -4, \\ 3x_1 + 21x_2 - 5x_4 = 2, \\ 4x_1 + 3x_2 - 5x_3 = 5. \end{cases}$$

В этом случае матрица коэффициентов A и вектор свободных коэффициентов b имеют вид

$$A = \begin{pmatrix} 0 & 1 & -13 & 4 \\ 1 & 0 & -2 & 3 \\ 3 & 21 & 0 & -5 \\ 4 & 3 & -5 & 0 \end{pmatrix}, \quad b = \begin{pmatrix} -5 \\ -4 \\ 2 \\ 3 \end{pmatrix}.$$

1. Введем матрицу A и вектор b в рабочий лист MS Excel (рис. 18). В нашем случае матрица A находится в ячейках B1:E4, а вектор b в диапазоне G1:G4.

	A	B	C	D	E	F	G	H
1	A=	0	1	-13	4	b=	-5	
2		1	0	-2	3		-4	
3		3	21	0	-5		2	
4		4	3	-5	0		3	
5								

Рис. 18. Матрица A

2. Для решения системы методом обратной матрицы необходимо вычислить матрицу, обратную A:

а) выделить ячейки для хранения обратной матрицы; пусть это будут ячейки B6:E9;

б) использовать функцию МОБР (массив) из категории Математические, в поле ввода **Массив** ввести диапазон ячеек, содержащий матрицу (рис. 19);

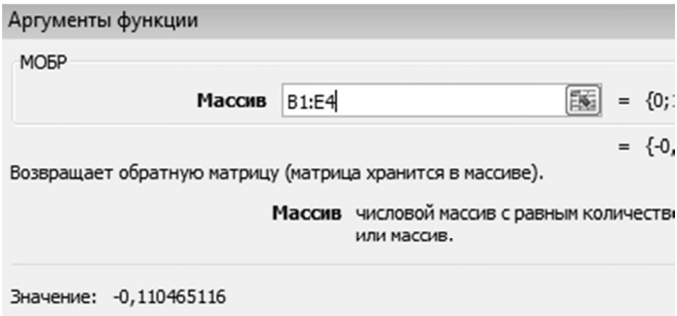
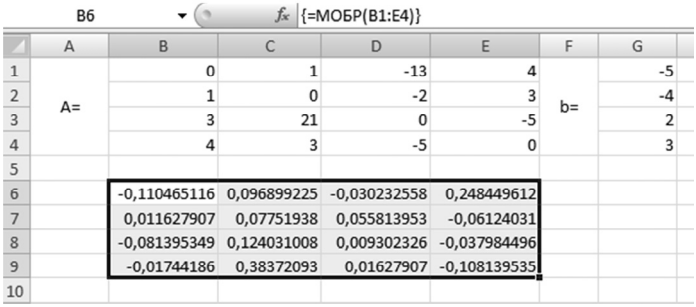


Рис. 19. Аргументы функции МОБР()

в) для получения всей обратной матрицы необходимо нажать клавишу F2 для перехода в режим редактирования, а затем одновременно клавиши Ctrl+Shift+Enter (рис. 20).



	A	B	C	D	E	F	G
1	A=	0	1	-13	4	b=	-5
2		1	0	-2	3		-4
3		3	21	0	-5		2
4		4	3	-5	0		3
5							
6		-0,110465116	0,096899225	-0,030232558	0,248449612		
7		0,011627907	0,07751938	0,055813953	-0,06124031		
8		-0,081395349	0,124031008	0,009302326	-0,037984496		
9		-0,01744186	0,38372093	0,01627907	-0,108139535		
10							

Рис. 20. Результат работы функции МОБР()

3. Для умножения полученной обратной матрицы на вектор *b* необходимо:

а) выделить ячейки для хранения результирующего вектора, например H6:H9;

б) найти функцию МУМНОЖ() в категории Математические;

в) в поле Массив1 (рис. 21) необходимо ввести диапазон ячеек, в котором содержится первая из перемножаемых матриц (B6:E9 обратная матрица), а в поле Массив2 – ячейки, содержащие вторую матрицу (G1:G4 вектор *b*);

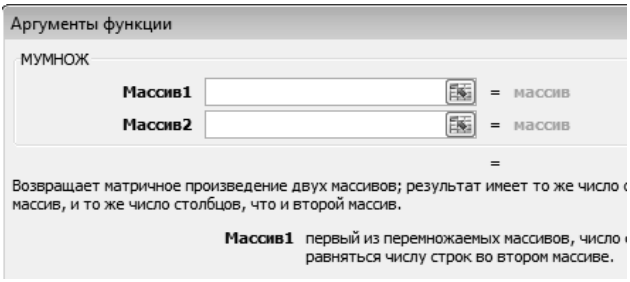


Рис. 21. Аргументы функции МУМНОЖ()

г) после нажатия кнопки OK в первой ячейке выделенного диапазона появится соответствующее число результирующего вектора. Для того чтобы получить весь вектор, необходимо нажать клавишу F2,

а затем одновременно клавиши Ctrl+Shift+Enter. В нашем случае результаты вычислений (вектор *x*) находятся в ячейках H6:H9.

4. Для проверки правильности решения системы уравнений необходимо умножить матрицу *A* на вектор *x* и получить в результате вектор *b*. Умножение матрицы *A* на вектор *x* осуществляется при помощи функции МУМНОЖ(B1:E4;H6:H9) так, как было описанно выше (рис. 22).

	A	B	C	D	E	F	G	H	I
1		0	1	-13	4		-5		
2	A=	1	0	-2	3	b=	-4		
3		3	21	0	-5		2		
4		4	3	-5	0		3		
5									
6		-0,110465116	0,096899225	-0,030232558	0,248449612		0,849612		-5
7		0,011627907	0,07751938	0,055813953	-0,06124031	x=	-0,44031	проверка	-4
8		-0,081395349	0,124031008	0,009302326	-0,037984496		-0,1845		2
9		-0,01744186	0,38372093	0,01627907	-0,108139535		-1,73953		3
10									

Рис. 22. Результаты работы функции МУМНОЖ()

Алгоритм решения системы уравнений методом Крамера на примере системы

1. Ввести матрицу *A* и вектор *b* на рабочий лист.

2. Сформировать четыре вспомогательные матрицы, заменяя последовательно столбцы матрицы *A* на столбец вектора *b* (рис. 23).

3. Вычислить определитель матрицы *A*:

а) установить курсор в ячейку I10 и вызвать функцию МОПРЕД() в категории Математические;

б) в поле ввода Массив указать диапазон матрицы, определитель которой вычисляют (ячейки B1:E4).

4. Вычислить вспомогательные определители, для чего ввести формулы:

I11=МОПРЕД(B6:E9), I12=МОПРЕД(B11:E14),

I13=МОПРЕД(B16:E19), I14=МОПРЕД(B21:E24).

В результате в ячейке I10 хранится главный определитель, а в ячейках I11:I14 – вспомогательные.

5. Воспользуемся формулами Крамера и разделим последовательно вспомогательные определители на главный:

а) в ячейку K11 ввести формулу=I11/I10;

б) скопировать содержимое ячейки I11 в ячейки K12, K13и K14. Система решена.

	A	B	C	D	E	F	G	H	I	J	K
1		0	1	-13	4		-5				
2	A=	1	0	-2	3	b=	-4				
3		3	21	0	-5		2				
4		4	3	-5	0		3				
5											
6		-5	1	-13	4						
7	A1=	-4	0	-2	3						
8		2	21	0	-5						
9		3	3	-5	0						
10								d=	2580		
11		0	-5	-13	4			d1=	2192		
12	A2=	1	-4	-2	3			d2=	-1136		0,849612
13		3	2	0	-5			d3=	-476	x=	-0,44031
14		4	3	-5	0			d4=	-4488		-0,1845
15											-1,73953
16		0	1	-5	4						
17	A3=	1	0	-4	3						
18		3	21	2	-5						
19		4	3	3	0						
20											
21		0	1	-13	-5						
22	A4=	1	0	-2	-4						
23		3	21	0	2						
24		4	3	-5	3						
25											

Рис. 23. Метод Крамера

Алгоритм выполнения действий с матрицами на примере вычисления матрицы *C* по формуле $C = A \cdot 2 + 2 \cdot A \cdot B$, где

$$A = \begin{pmatrix} 3 & 9 & -2 \\ 2 & -13 & 3 \\ 11 & 2 & 4 \end{pmatrix}; \quad B = \begin{pmatrix} 1 & 4 & 11 \\ 4 & 5 & 5 \\ 11 & 3 & 7 \end{pmatrix}.$$

1. Введите исходные данные на рабочий лист (рис. 24).

	A	B	C	D	E	F	G	H
1		3	9	-2		1	4	11
2	A=	2	-13	3	B=	4	5	5
3		11	2	4		11	3	7
4								
5		17	51	64		9	81	4
6	A*B	-17	-48	-22	A^2	4	169	9
7		63	66	159		121	4	16
8								
9		34	102	128		43	183	132
10	2A*B	-34	-96	-44	C=A^2+AB	-30	73	-35
11		126	132	318		247	136	334
12								

Рис. 24. Операции с матрицами

2. Для умножения матрицы A на матрицу B выделить диапазон B5:D7 и воспользоваться функцией МУМНОЖ(B1:D3;G1:I3).

3. Результат вычисления $A^2 = A \cdot A$ поместить в ячейки G5:I7, воспользовавшись формулой МУМНОЖ(B1:D3;B1:D3).

4. Умножение (деление) матрицы на число можно выполнить при помощи элементарных операций. В нашем случае необходимо умножить матрицу из диапазона B5:D7 на число 2. Выделим ячейки B9:D11 и введем формулу =2 · B5:D7.

5. Сложение (вычитание) матриц выполняется аналогично. Например, выделим диапазон G9:I11 и введем формулу =B9:D11+ G5:I7.

6. Для получения результата в обоих случаях необходимо нажать комбинацию клавиш Ctrl+Shift+Enter.

Решение нелинейных уравнений

Алгоритм решения нелинейного уравнения на примере уравнения $e^x - (2x-1)^2 = 0$

1. Решить уравнение графически:

а) представить уравнение в виде $f(x) = g(x)$, т. е. $e^x = (2x-1)^2$ или $f(x) = e^x$, $g(x) = (2x-1)^2$. Графическим решением уравнения $f(x) = g(x)$ будет точка пересечения линий $f(x)$ и $g(x)$;

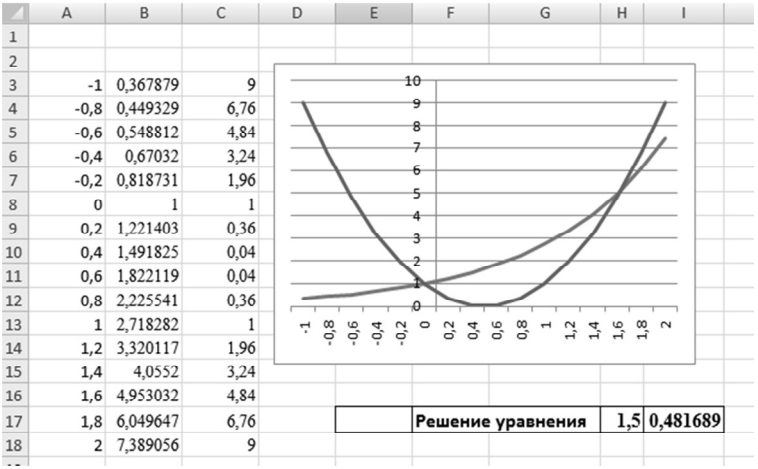
б) построить графики $f(x)$ и $g(x)$. В диапазон A3:A18 ввести значения аргумента. В ячейку B3 введем формулу для вычисления значений функции $f(x) := \text{EXP(A3)}$, а в C3 для вычисления $g(x) := (2 \cdot \text{A3} - 1) \wedge 2$ (рис. 25).

2. Проанализировать полученное графическое решение:

а) На графике видно, что линии $f(x)$ и $g(x)$ пересекаются дважды, т. е. данное уравнение имеет два решения. Одно из них тривиальное и может быть вычислено точно:

$$(x = 0) \Rightarrow \begin{cases} e^x = 1 \\ (2x-1)^2 = 1 \end{cases} \Rightarrow y(x) = 1;$$

б) для второго можно определить интервал изоляции корня: $1{,}5 < x < 2$.



	A	B	C
3	-1	0,367879	9
4	-0,8	0,449329	6,76
5	-0,6	0,548812	4,84
6	-0,4	0,67032	3,24
7	-0,2	0,818731	1,96
8	0	1	1
9	0,2	1,221403	0,36
10	0,4	1,491825	0,04
11	0,6	1,822119	0,04
12	0,8	2,225541	0,36
13	1	2,718282	1
14	1,2	3,320117	1,96
15	1,4	4,0552	3,24
16	1,6	4,953032	4,84
17	1,8	6,049647	6,76
18	2	7,389056	9

	Решение уравнения	1,5	0,481689

Рис. 25. Решение нелинейного уравнения

3. Найти корень уравнения на отрезке [1,5; 2] методом последовательных приближений:

а) ввести начальное приближение в ячейку H17 = 1,5 и само уравнение со ссылкой на начальное приближение в ячейку

I17 = EXP(H17) – (2 · H17-1)^2 (см. рис. 22);

б) воспользоваться пунктом меню Сервис → Подбор параметра (Настройка панели быстрого доступа → Другие команды → Подбор параметра) и заполнить диалоговое окно Подбор параметра (рис. 26);

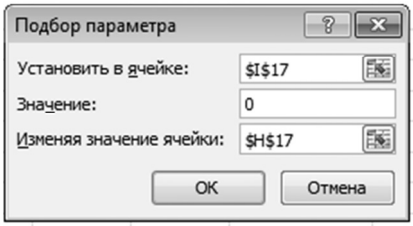


Рис. 26. Подбор параметра

в) результат поиска решения будет выведен в ячейку **H17**.

Алгоритм решения системы уравнений на примере системы $\begin{cases} 2x_1 - 3x_2 + 4 = 0, \\ x_1 + x_2 - 4 = 0. \end{cases}$

Рассмотрим, как можно решить систему уравнений

$$F_1(x) = 0,$$

$$F_2(x) = 0,$$

$$\dots$$

$$F_n(x) = 0$$

с помощью решающего блока (пункт меню Сервис → Поиск Решения или Параметры Excel → Надстройки или Данные → Анализ), который позволяет решать не только оптимизационные задачи, но и обычные уравнения и системы уравнений.

Для решения этой задачи ее можно сформулировать одним из следующих способов.

1. Найти минимум (максимум) функции

$$\Phi(x) = \sum_{i=1}^{n} F_i(x)$$

при системе ограничений, заданной в виде равенств $F_i(x) = 0$.

2. Найти минимум функции

$$\Phi(x) = \sum_{i=1}^{n} F_i^2(x) = F_1^2(x) + F_2^2(x) + \dots + F_n^2(x).$$

В этом случае задача решается без ограничений.

1-й способ

1. В ячейки A1 и A2 ввести числа 0 (здесь мы будем хранить x_1 и x_2).

2. В ячейки B1 и B2 ввести ограничения: B1 = 2 · A1–3 · A2, B2 = A1 + A2.

3. В ячейку C1 ввести функцию цели (эту ячейку мы будем минимизировать): C1 = СУММ(B1:B2).

4. Воспользоваться командой Сервис → Поиск Решения и заполнить появившееся диалоговое окно так, как показано на рис. 27.

5. В результате решения поставленной задачи получим решение системы исходных уравнений: $x_1 = 1{,}6,\ x_2 = 2{,}4$.

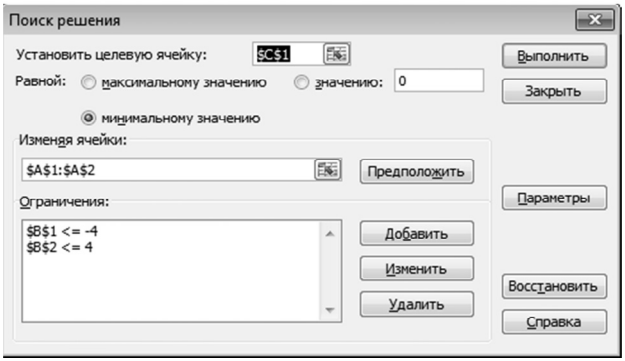


Рис. 27. Поиск решения

2-й способ

1. В ячейках D1 и D2 будем хранить переменные x_1 и x_2.

2. В ячейки E1 и E2 ввести уравнения системы: E1 = 2 · D1 –3 · D2 + 4, E2 = D1 + D2 – 4.

3. В качестве функции цели в ячейку F1 ввести формулу = E1^2+E2^2.

4. Обратиться к решающему блоку (рис. 28) и ввести условие задачи оптимизации.

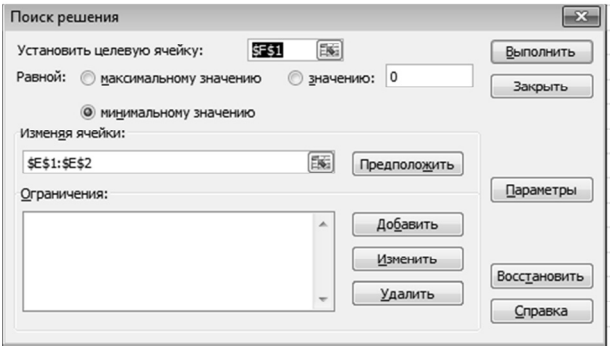


Рис. 28. Поиск решения

5. В результате получаем следующее решение системы: $x_1 = 1{,}600000128,\ x_2 = 2{,}39999949$.

3.6. MS Access. Работа с таблицами Сортировка и фильтрация информации

База данных – это поименованная совокупность структурированных данных, относящихся к определенной предметной области.

Реляционная база данных – множество взаимосвязанных двумерных таблиц, каждая из которых содержит сведения об одном объекте предметной области.

Таблицы состоят из ***полей*** (столбцов) и ***записей*** (строк), рис. 29.

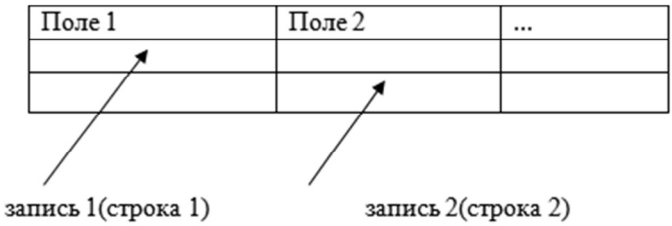


Рис. 29. Структура таблицы

Поле – элементарная единица логической организации данных, которая соответствует неделимой единице информации.

Для описания поля используются следующие характеристики:

- ***Имя***, например, фамилия;
- ***Тип***, например, символьный, числовой, дата, счетчик, логический и др.;
- ***Длина***, максимальное количество символов.

Запись – совокупность логически связанных полей.

Экземпляр записи – отдельная реализация записи, содержащая конкретные значения ее полей.

Каждая таблица базы данных обладает следующими ***свойствами*** (на рис. 30 представлен пример таблицы Заготовки базы данных Завод):

- все записи содержат одинаковые поля;
- данные одного поля для всех записей имеют один и тот же тип;

- каждый столбец имеет уникальное имя;
- одинаковые строки в таблице отсутствуют;
- порядок следования строк и столбцов произвольный.

Заготовки

Код заготов	категория	заготовка	количество	Добавить поле
1	круг	круг 30	117	
2	круг	круг 50	55	
3	круг	круг 70	31	
4	лист	лист 4	85	
5	лист	лист 6	255	
6	лист	лист 8	345	
(№)				

Рис. 30. Пример структуры таблицы Заготовки

Этапы создания базы данных

1. ***Логическая организация данных*** (анализ объектов автоматизации, анализ исходной информации, определение количества и структуры таблиц базы данных в соответствии с объектами).

2. ***Заполнение базы данных*** информацией.

При проектировании БД необходимо знать и использовать ***следующие правила***.

Правило 1. БД должна быть ***нормализована***, т. е. в БД не должно быть повторяющихся данных.

Это достигается разбиением таблиц на более мелкие (выделение данных в справочники).

Правило 2. В БД должна быть обеспечена ***целостность*** данных, т. е. при изменении одних данных автоматически происходит соответствующее изменение связанных с ним данных.

Это достигается наличием логических связей между таблицами. Для связи таблиц между собой используют ***ключевое поле*** или ***простой ключ***.

Простым ключом называют поле, каждое значение которого однозначно определяет соответствующую запись.

Обычно в качестве простого ключа используют поле, называемое кодом – индивидуальный номер записи (рис. 31).

MS Access – система управления базами данных (СУБД).

СУБД **(*система управления базой данных*)** – это комплекс программных и языковых средств, необходимых для создания базы дан-

ных, поддержания их в актуальном состоянии и организации поиска в них необходимой информации.

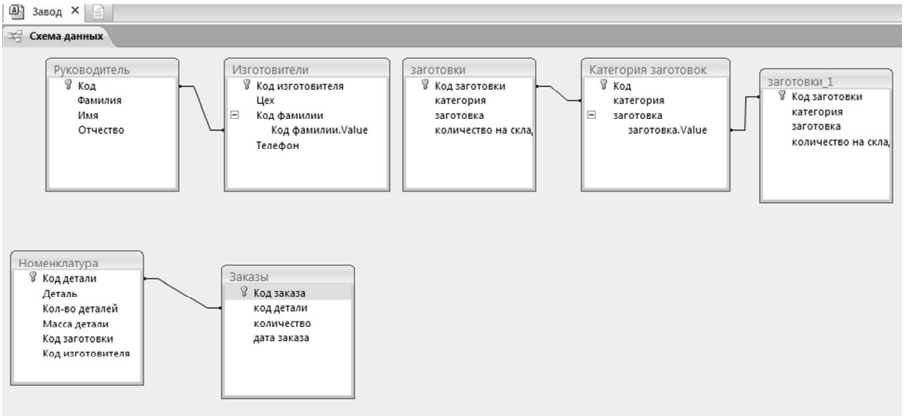


Рис. 31. Структура базы данных (схема данных)

Основные функции СУБД

1. **Определение данных**. Можно определить, какие данные будут храниться в БД, тип данных и связи между ними.

2. **Обработка данных**. Добавление, изменение, просмотр, удаление данных. Можно фильтровать и сортировать данные.

3. **Управление данными**. Определение прав пользователей базы данных и порядка совместного использования и обновления данных несколькими пользователями.

Основные объекты СУБД

1. ***Таблица***. Объект, используемый для непосредственного хранения данных.

2. ***Запрос***. Объект, обеспечивающий настраиваемый вывод данных из одной или нескольких таблиц.

3. ***Форма***. Объект предназначен для ввода и вывода данных, а также для управления работой приложения.

4. ***Отчет***. Объект предназначен для форматирования, вычисления, печати и обобщения выбранных данных.

5. ***Макрос***. Объект представляет собой структурированное описание одного или нескольких действий, которые необходимо выполнить в качестве реакции на определенные события

Приемы работы с базами данных: создание базы данных в системе MS Access

1. Создание новой базы данных: Кнопка *Office*, команда *Создать – Создание базы данных*.

2. Создание таблиц базы данных в режиме конструктора, используется вкладка Создание инструмент Конструктор таблиц.

3. Импорт таблицы MS Excel в базу данных MS Access: инструмент *Excel* на вкладке *Внешние данные.*

4. Создание связей между таблицами: инструмент *Мастер подстановок* в режиме *Конструктора таблиц*.

5. Создание схемы данных созданной базы: инструмент *Схема данных* вкладки *Работа с базами данных*.

Контрольные вопросы

1. Что такое база данных? Что такое СУБД? Что такое MS Access?
2. Назовите основные функции СУБД.
3. Что такое реляционная база данных?
4. Назовите основные элементы реляционной базы данных?
5. Что такое ключ или ключевое поле?
6. Назовите и дайте понятие основным объектам базы данных.
7. Назовите два этапа процесса проектирования базы данных.
8. Что такое логическая организация данных?
9. Укажите назначение связей между таблицами.
10. Назовите основные типы связей между таблицами.

3.7. MS Access. Работа с формами Представление на форме информации различных типов

Формы в MS Access – объект базы данных, который можно применять для создания пользовательского интерфейса для приложения базы данных.

Форму можно использовать для ввода, изменения или отображения данных из таблицы или запроса.

Формы в БД MS Access можно создавать с помощью различных средств:

- инструмента Форма;

- инструмента Разделенная форма;
- инструмента Несколько элементов;
- инструмента Пустая форма;
- инструмента Мастер форм;
- инструмента Конструктор форм.

Создание новой формы с помощью инструмента Форма

1. В области переходов необходимо нажать на элементы Таблица или Запрос с данными, которые должны отображаться в форме.

2. На вкладке **Создать** в группе **Формы** нужно нажать кнопку **Форма**.

В результате создана новая форма и отображена в режиме макета. В режиме макета можно внести изменения в структуру формы при одновременном отображении данных.

Создание разделенной формы с помощью инструмента Разделенная форма

Разделенная форма позволяет одновременно отображать данные в двух представлениях: в режиме формы и в режиме таблицы.

Разделенная форма отличается от сочетания формы и подчиненной формы тем, что этих два представления связаны с одним источником данных и всегда синхронизированы друг с другом. При выделении поля в одной части формы выделяется то же поле в другой части. Данные можно добавлять, изменять или удалять в любой части (при условии, что источник записей допускает обновление, а параметры формы не запрещают такие действия).

Разделенная форма позволяет использовать преимущества обоих типов форм в одной форме. Например, можно воспользоваться табличной частью формы, чтобы быстро найти запись, а затем просмотреть или изменить запись в другой части формы.

Чтобы создать разделенную форму при помощи инструмента Разделенная форма, выполните следующие действия.

1. В области переходов необходимо выбрать таблицу или запрос с данными, которые должны отображаться в форме, или открыть таблицу или запрос в режиме таблицы.

2. На вкладке **Создать** в группе **Формы** нужно нажать кнопку **Другие формы** и выбрать команду **Разделить форму**.

Создание формы, в которой отображается несколько записей, с помощью инструмента Несколько элементов

В форме, созданной с помощью инструмента ***Форма***, одновременно отображается только одна запись. Если нужна форма, в которой отображается сразу несколько записей, и при этом требуются более широкие возможности настройки, чем у таблицы, можно воспользоваться инструментом ***Несколько элементов.***

1. В области переходов необходимо выбрать таблицу или запрос с данными, которые должны отображаться в форме.

2. На вкладке **Создать** в группе **Формы** нужно нажать кнопку **Другие формы** и выбрать пункт **Несколько элементов**.

Создание формы при помощи мастера форм

Средство позволяет включить в форму поля из нескольких связанных таблиц или запросов. На созданной форме можно группировать и сортировать данные и использовать поля из нескольких таблиц или запросов при условии, что заранее указаны связи между таблицами и запросами (рис. 32).

1. На вкладке **Создание** в группе **Формы** необходимо выбрать команду **Мастер форм**.

2. Далее нужно следовать инструкциям на страницах мастера форм.

3. На последней странице необходимо нажать кнопку **Готово**.

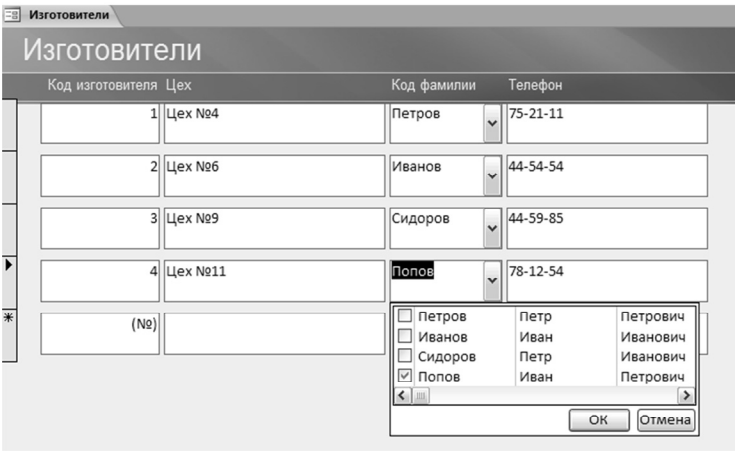


Рис. 32. Пример формы «Изготовители», созданной при помощи мастера форм

Создание формы с помощью средства «Пустая форма»

Если мастер или инструменты создания форм не подходят, для создания формы можно воспользоваться инструментом ***Пустая форма***.

1. На вкладке **Создать** в группе **Формы** необходимо нажать кнопку **Пустая форма**.

В MS Access откроется пустая форма в режиме макета и отобразится область **Список полей**.

2. В области **Список полей** нужно нажать знак плюс (+) рядом с таблицей или таблицами, содержащими поля, которые нужно включить в форму.

3. Чтобы добавить поле в форму, дважды щелкните по нему и перетащите на форму.

После добавления первого поля можно добавить одновременно несколько полей. Для этого необходимо выбрать несколько полей, удерживая при этом нажатой клавишу CTRL, а затем одновременно перетащить их на форму.

4. Используя инструменты группы **Колонтитулы** на вкладке **Конструктор**, можно добавить в форму логотип, заголовок или дату и время.

5. Для добавления в форму элементов управления других типов используйте инструменты группы **Элементы управления** на вкладке **Конструктор**.

Немного больший выбор элементов управления доступен в режиме конструктора, если выбрать форму правой кнопкой мыши и выбрать пункт **Режим конструктора**.

Режим макета и режим конструктора

Режим макета. Режим макета является наиболее интуитивно понятным представлением для изменения формы, которую можно использовать практически для всех изменений, которые нужно внести в формы в MS Access.

В режиме макета форма уже запущена. Данные отображаются в основном так же, как при реальном использовании формы. При этом в данном режиме можно изменять структуру формы. Так как при изменении формы отображаются реальные данные, в режиме макета удобно задавать размер элементов управления и выполнять иные задачи, влияющие на внешний вид и удобство использования формы.

Режим конструктора. Режим конструктора обеспечивает более подробное представление структуры формы. В нем отображаются раз-

делы колонтитулов и данных формы. В этом режиме форма не выполняется, поэтому при внесении изменений невозможно просмотреть соответствующие данные. Однако некоторые задачи удобнее выполнять в режиме конструктора, а не макета, например, следующие:

- добавление в форму дополнительных элементов управления, таких как границы привязанных объектов, разрывы страниц и диаграммы;
- изменение источника элемента управления **Поле** непосредственно в поле без использования окна свойств;
- изменение размеров разделов формы, таких как **Заголовок формы** или **Область данных**;
- изменение определенных свойств формы, которые нельзя изменить в режиме макета.

Доработка формы в режиме макета

Создав форму, можно легко доработать ее в режиме макета. Ориентируясь на фактические данные формы, можно изменить расположение элементов управления и подобрать их размеры. Можно добавить в форму новые элементы управления, а также задать свойства формы и входящих в нее элементов управления.

Чтобы переключиться в режим макета, необходимо щелкнуть правой кнопкой мыши имя формы в области переходов и выбрать команду **Режим макета**. Форма будет открыта в режиме макета.

Изменить свойства формы, ее разделов и элементов управления можно с помощью окна свойств. Чтобы открыть его, нужно нажать клавишу F4.

Из области **Список полей** можно добавить в макет формы поля из базовой таблицы или базового запроса. Для отображения области **Список полей** нужно выполнить одно из указанных ниже действий.

- На вкладке **Конструктор** в группе **Сервис** выбрать пункт **Добавить существующие поля** или нажать сочетание клавиш, нажав клавиши ALT + F8.

Можно перетащить поля непосредственно из области **Список полей** в форму.

- Чтобы добавить одно поле, необходимо дважды щелкнуть по нему или перетащить его из области **Список полей** в тот раздел формы, где оно должно отображаться.
- Чтобы добавить сразу несколько полей, нужно щелкнуть по ним последовательно, удерживая нажатой клавишу CTRL. Затем можно перетащить выбранные поля в форму.

Доработка формы в режиме конструктора

Для выполнения точной настройки конструктора формы при работе в режиме конструктора можно добавлять новые элементы управления и поля в форму. Окно свойств обеспечивает доступ к множеству свойств для настройки форм.

Чтобы переключиться в режим конструктора, необходимо щелкнуть правой кнопкой мыши имя формы в области переходов и выбрать команду **Конструктор**. Форма откроется в режиме конструктора.

Изменить свойства формы, ее разделов и элементов управления можно с помощью окна свойств. Чтобы открыть его, нужно нажать клавишу F4.

Из области **Список полей** можно добавить в макет формы поля из базовой таблицы или базового запроса. Для отображения области **Список полей** необходимо выполнить одно из указанных ниже действий.

- На вкладке **Конструктор** в группе **Сервис** выберите пункт **Добавить существующие поля** или нажмите сочетание клавиш, нажав клавиши ALT + F8.

Можно перетащить поля непосредственно из области **Список полей** в форму.

- Чтобы добавить одно поле, дважды щелкните по нему или перетащите его из области **Список полей** в тот раздел формы, где оно должно отображаться.
- Чтобы добавить сразу несколько полей, щелкните по нему, удерживая нажатой клавишу CTRL. Затем перетащите выбранные поля в форму.

Контрольные вопросы

1. Что такое формы? Для чего они используются?
2. Какие варианты создания форм существуют?
3. В чем состоит процесс создания формы режиме конструктор форм?
4. Какие возможности предоставляет Мастер форм?
5. Что такое Подчиненная форма? Для чего используется Подчиненная форма?
6. Какого типа форма используется для работы с несколькими записями базы данных одновременно?

7. Для чего необходим инструмент создания форм Пустая форма?

8. В каких целях используют Разделенную форму?

9. Какие возможности создания и представления графической информации существуют в MS Access?

3.8. MS Access. Запросы

Запросы упрощают просмотр, добавление, удаление или изменение данных в базе данных MS Access. Среди других целей использования запросов можно отметить:

- быстрый поиск определенных данных путем фильтрации с применением определенных критериев (условий);
- вычисление или сведение данных;
- автоматизированное управление данными, например, регулярный просмотр актуальных данных.

Создание запроса на выборку

Запрос на выборку позволяет просматривать данные только из определенных полей таблицы или из нескольких таблиц одновременно, или же находить данные, которые соответствуют определенным условиям.

Просмотр данных из выбранных полей

Например, если база данных содержит таблицу «Номенклатура» с информацией о номенклатуре деталей, выпускаемой на заводе, и необходимо просмотреть список деталей с массой >30, запрос на выборку создается таким образом, чтобы возвратить только названия детали и соответствующую массу (рис. 33).

1. Откройте базу данных «Завод» и на вкладке **Создание** нажмите кнопку **Конструктор запросов**.

2. В диалоговом окне **Добавление таблицы** на вкладке **Таблицы** дважды щелкните по таблице **Номенклатура**, затем закройте диалоговое окно.

3. В первом столбце бланка запроса выберите поле Деталь, во втором столбце – поле Масса детали. В области Условие отбора запишите условие >30.

4. На вкладке Конструктор нажмите кнопку Выполнить. Запрос будет выполнен, и отобразится список деталей и масса.

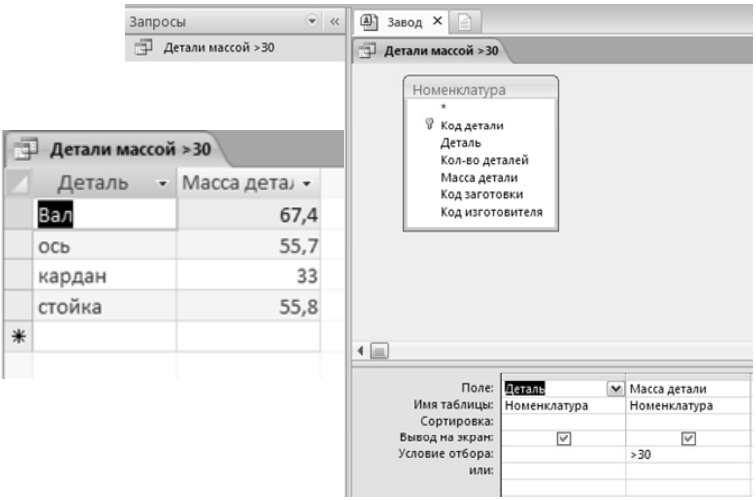


Рис. 33. Пример запроса Детали массой >30

Одновременный просмотр данных из нескольких связанных таблиц

Рассмотрим пример базы данных «Завод», в которой требуется просмотреть заказы, полученные от клиентов из определенного города. Допустим, данные о заказах и о номенклатуре хранятся в двух таблицах под названием «Номенклатура» и «Заказы» соответственно. В каждой таблице имеется поле Код детали, которое является основой отношения «один-ко-многим». Между этими двумя таблицами можно создать запрос, который возвратит сведения о заказах (название детали и ее количество) за 2016 год (рис. 34).

1. Открыть базу данных. На вкладке **Создание** в группе **Запросы** необходимо нажать кнопку **Конструктор запросов**.

2. В диалоговом окне **Добавление таблицы** на вкладке **Таблицы** нужно выбрать таблицы **Номенклатура** и **Заказы**.

3. В бланке запроса выбрать поля **Деталь**, **Кол-во деталей** таблицы **Номенклатура** и **дата заказа** таблицы **Заказы**.

4. В строке Условие отбора столбца **Дата заказа** ввести условие <#31.12.2016# And >#01.01.2016#, соответствующее периоду 2016 года.

5. На вкладке Конструктор в группе Результаты нажать кнопку Выполнить. Происходит выполнение запроса, и отображаются список заказов (название детали и ее количество) и дата заказа (за период 2016).

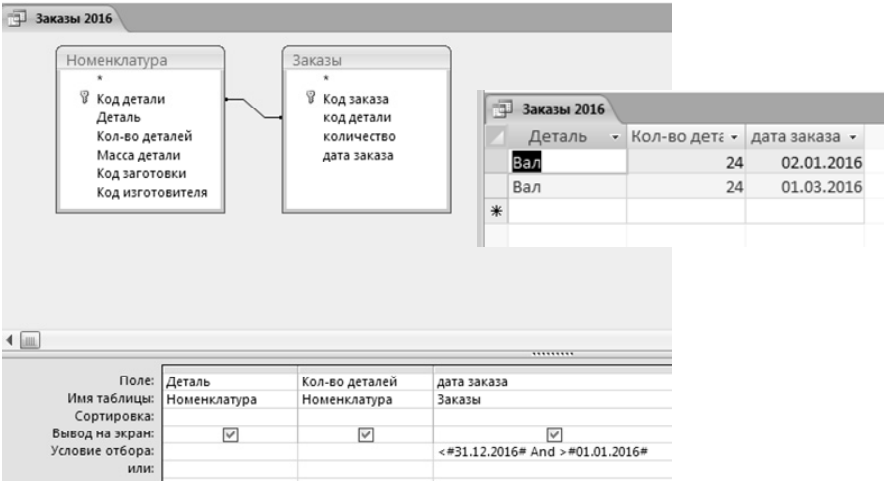


Рис. 34. Пример запроса Заказы 2016

Создание запроса с параметрами

Если требуется выполнять варианты определенного запроса, можно использовать запрос с параметрами. При выполнении запроса с параметрами у пользователя запрашиваются значения полей, которые затем используются для создания условий для запроса. Например, необходимо вывести информацию о детали, указанной пользователем (рис. 34).

1. Создать запрос **Информация о детали** в режиме **Конструктор запросов**.

2. В бланке запроса выбрать поля **Детали**, **Кол-во деталей**, **Масса детали**.

3. В бланке запроса в строке **Условие отбора** столбца **Детали** введите **[Для какой детали?]**. Строка **[Для какой детали?]** является предложением ввести параметр. Квадратные скобки показывают, что при выполнении запроса должно появиться предложение ввода данных, а текст представляет собой вопрос, отображаемый в предложении.

4. На вкладке **Конструктор** в группе **Результаты** необходимо нажать кнопку **Выполнить**. Запрос предложит ввести значение в строке **Детали**.

5. Ввести слово «вал» и нажать клавишу ВВОД, чтобы увидеть информацию о количестве и массе детали (вал).

Указание типов данных для параметра

Можно также указать, какого типа данные разрешается вводить в качестве значения параметра. Если параметр настроен таким образом, чтобы принимать текстовые данные, любое введенное значение интерпретируется как текст, и сообщение об ошибке не отображается.

Чтобы указать тип данных для параметра в запросе, выполните процедуру, описанную ниже.

1. Когда запрос открыт в конструкторе, на вкладке **Конструктор** в группе **Показать или скрыть нужно** нажать кнопку **Параметры**.

2. В диалоговом окне **Параметры запроса** в столбце **Параметр** необходимо ввести текст запроса на ввод значения для каждого параметра, для которого требуется указать тип данных. Каждый из параметров должен соответствовать запросу, который используется в строке **Условие отбора** в бланке запроса.

3. В столбце **Тип данных** выбрать тип данных для каждого параметра.

Создание итогового запроса

Строка Итог в таблице очень удобна, но для ответа на более сложные вопросы используется запрос итоговых значений (рис. 35).

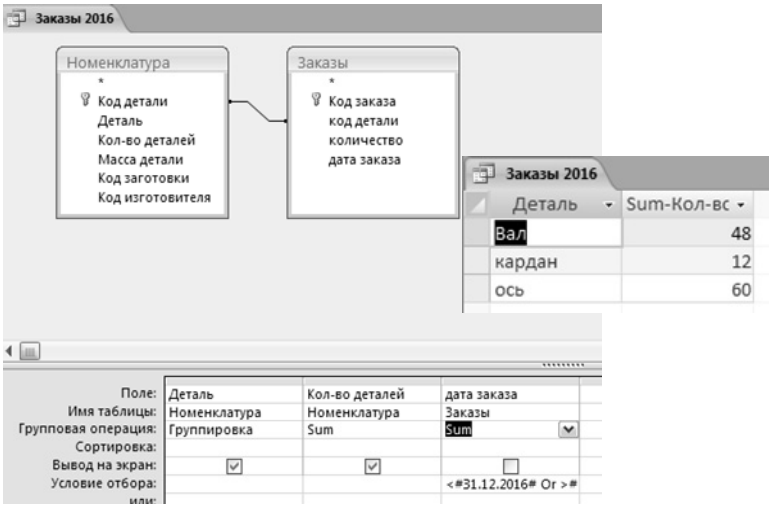


Рис. 35. Пример запроса Информация о детали

Чтобы получить итоговые значения промежуточных сумм по количеству деталей для каждого вида детали за 2016 год, можно изменить запрос **Заказы 2016**.

1. Открыть запрос **Заказы 2016** в режиме **Конструктора запросов**.
2. На вкладке **Конструктор** в группе **Показать или скрыть** нужно нажать кнопку **Итоги**.
3. В бланке запроса отобразится строка **Групповая операция**. Для поля **Детали** выбрать **Группировка**, для поля **Кол-во деталей** – параметр **Sum**, для поля **дата заказа** – параметр **Sum**.
4. Значение поля Дата заказа не выводить на экран (убрать галочку в поле Вывод на экран).
5. На вкладке **Конструктор** в группе **Результаты** нажмите кнопку **Выполнить**. Происходит выполнение запроса, а затем отображается список деталей с промежуточными суммами.

Контрольные вопросы

1. Что такое запрос и для чего он используется?
2. В чем суть запроса на выборку?
3. В чем состоит процесс создания запроса с помощью конструктора?
4. Основные шаги построения запроса с помощью мастера запроса?
5. Построение параметрического запроса.
6. Создание вычисляемых полей в запросах.
7. Формирование итогового запроса.
8. Создание запроса на удаление.
9. Запрос на обновление и добавление.
10. Что такое перекрестный запрос?

3.9. MS Access. Отчеты. Кнопочная форма Настройка параметров приложения Область переходов

С помощью отчетов можно просматривать, форматировать и группировать информацию в базе данных MS Access.

Чтобы создать отчет для базы данных MS Access, необходимо выполнить следующие действия.

Действие 1. Выбор источника записей

Источником записей для отчета может быть таблица, именованный или внедренный запрос. Источник записей должен содержать все строки и столбцы данных, которые требуется отобразить в отчете.

- Если нужные данные содержатся в существующей таблице или запросе, нужно выделить эту таблицу или запрос в области навигации и перейти к действию 2.
- Если источник записей еще не создан, выполните одно из перечисленных ниже действий.
- Перейдите к действию 2 и воспользуйтесь инструментом Пустой отчет ИЛИ
- Создайте таблицы или запрос, которые будут содержать нужные данные, выберите их в области навигации и перейдите к действию 2.

Действие 2. Выбор инструмента отчета

Инструменты отчета расположены на вкладке Создать в группе Отчеты. В табл. 4 описаны параметры средств отчета.

Таблица 4

Средства отчета

Средство	Описание
Отчет	Позволяет создать простой табличный отчет, содержащий все поля из источника записей, который выбран в области навигации
Конструктор отчетов	Открывает в режиме конструктора пустой отчет, в который можно добавить необходимые поля и элементы управления
Пустой отчет	Позволяет открыть пустой отчет в режиме макета и отобразить область задач «Список полей», из которой можно добавить поля в отчет
Мастер отчетов	Служит для вызова пошагового мастера, с помощью которого можно задать поля, уровни группировки и сортировки и параметры макета
Наклейки	Вызывает форму, в которой можно выбрать стандартный или настраиваемый размер подписей, набор отображаемых полей и порядок их сортировки

Действие 3. Создание отчета

1. Для выбора требуемого инструмента нужно нажать соответствующую кнопку панели инструментов. После появления мастера следовать всем его командам и на последней странице нажать кнопку **Готово**. MS Access отображает отчет в режиме макета.

2. Отформатируйте отчет, чтобы добиться желаемого внешнего вида. Возможности форматирования:

- изменять размер полей и подписей;
- располагать поля в нужном порядке, выделяя их и перетаскивая в нужное место;
- с помощью команд контекстного меню можно объединять или разбивать ячейки, удалять и выделять поля и выполнять другие задачи форматирования.

Для создания и настройки отчетов в MS Access предусмотрены следующие инструменты.

1. **Отчет** на вкладке **Создание**.
2. **Мастер отчетов** на вкладке **Создание.**
3. **Конструктор отчетов** на вкладке **Создание.**
4. **Наклейки** вкладки **Создание.**
5. **Диспетчер кнопочных форм**. Если инструмента нет на вкладке, то нужно добавить эту кнопку на **Панель быстрого доступа**, выбрав пункт меню **Другие команды**.
6. **Область переходов.**

Контрольные вопросы

1. Что такое отчет? Для чего необходимо использовать отчеты?
2. Какие данные могут быть источником отчета?
3. Изучите основные способы создания отчета.
4. В чем состоит процесс создания отчета с помощью конструктора?
5. Особенности построения отчета с помощью инструмента Мастер отчетов?
6. Как создаются вычисляемые поля в отчетах?
7. Что такое кнопочная форма и для чего она нужна?
8. В чем состоит принцип создания кнопочных форм?
9. Функции инструмента Наклейки.
10. Возможности настройки параметров приложения.

4. МАТЕМАТИЧЕСКИЙ ПАКЕТ MATHCAD

4.1. Основные приемы работы с математическим пакетом Mathcad Простейшие и символьные вычисления

Интерфейс системы Mathcad

Интерфейс Mathcad идентичен с другим приложением Windows: заголовок, строка меню, панели инструментов, рабочая область, строки состояния (рис. 36). Поэтому при ознакомлении с интерфейсом используйте знания, полученные при работе с другими Windows-приложениями.

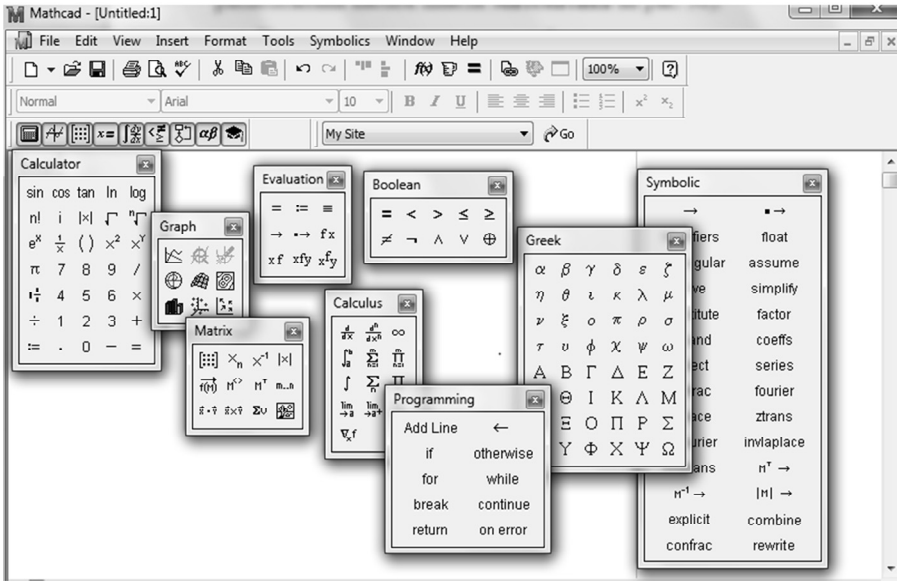


Рис. 36. Окно Mathcad с активизированными панелями инструментов

В большинстве своем команды меню и панели инструментов **Стандартная** и **Форматирование** аналогичны другим приложениям (например, Microsoft Word). Отличные от других приложений пункты меню **Symbolic** содержат команды управления для вычислений и сим-

вольных вычислений. Для удобства работы с различными математическими выражениями имеется панель **Math** (View/Toolbars), которая служит для вывода на экран еще девяти панелей (по порядку расположения кнопок панели Math, показаных на рис. 36):

- **Calculator (Калькулятор)** – для вставки цифр и основных математических операторов;
- **Graph (График)** – для вставки графиков;
- **Matrix (Матрица)** – для вставки матриц и матричных операторов;
- **Evalutions (Вычисления)** – для вставки операторов управления вычислениями;
- **Calculus (Матанализ)** – для вставки операторов дифференцирования, интегрирования, пределов, сумм и произведений;
- **Boolean (Логический)** – для вставки логических операторов;
- **Progamming (Программирование)** – для программирования средствами Mathcad;
- **Greek (Грек)** – для вставки греческих символов;
- **Symbolic (Символы)** – для вставки ключевых слов и операторов символьных вычислений.

Ввод текстового комментария

- Ввести знак двойной кавычки (“) на английском регистре или выбрать пункт меню **Insert/Test** – появится прямоугольник с курсором ввода в виде красной вертикальной черты (рис. 37).

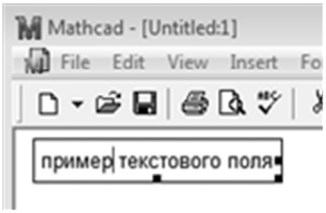


Рис. 37. Пример текстового поля

- Выбрать стиль **Normal** и шрифт, заканчивающийся словом **Cyr** при вводе русскоязычного текста.
- Набрать текст с использованием типовых средств набора и форматирования, характерных для текстовых редакторов.
- Для завершения ввода отведите указатель мыши в строну от текстового блока и щелкните левой кнопкой мыши.

Ввод и вычисление математических выражений

- Вывести на экран необходимые панели, щелкнув левой кнопкой мыши по обозначающим их кнопкам панели Math.

• Установить указатель мыши в требуемом месте окна редактирования (место ввода выражения) и щелкнуть левой кнопкой мыши.

• Указать с помощью мыши нужный объект или символ на панели инструментов и щелкнуть левой кнопкой мыши для помещения объекта (символа) в требуемое место.

• Охватить синим уголком выражение и ввести оператор вывода (знак =). После этого на экране появится результат вычислений (рис. 38).

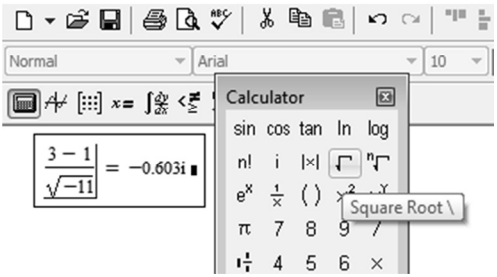


Рис. 38. Пример вычисления математических выражений

Вычисление выражений, содержащих переменные

Имена переменных могут состоять из одного или нескольких символов. В качестве символов можно использовать большие и маленькие буквы, греческие буквы, цифры, символ подчеркивания.

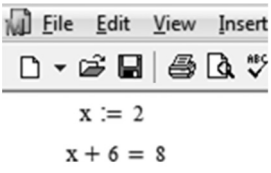


Рис. 39. Пример вычисления выражений

Для вычисления выражений, содержащих переменные, необходимо:

• присвоить переменным нужные числовые значения;

• для присвоения переменной некоторого значения используется оператор присваивания (:=), который с клавиатуры вводится двоеточием (рис. 39).

Ввод встроенных функций

Mathcad поддерживает множество встроенных функций. Основные простейшие функции:

• $\exp(x)$ – экспонента (соответствует e^x);

- $\ln(x)$, $\log(x)$ – натуральный и десятичный логарифм;
- $\log(x, n)$ – логарифм x по основанию n;
- $\sin(x)$, $\cos(x)$, $\tan(x)$, $\cot(x)$, $\sec(x)$, $\csc(x)$ – тригонометрические функции соответственно синус, косинус, тангенс, котангенс, секанс и косеканс;
- $\text{asin}(x)$, $\text{acos}(x)$, $\text{atan}(x)$, $\text{acot}(x)$, $\text{asec}(x)$, $\text{acsc}(x)$ – обратные тригонометрические функции;
- $\text{if}(\text{cond}, x, y)$ – принимает значение x, если условие cond выполняется, и значение y в противном случае (рис. 40).

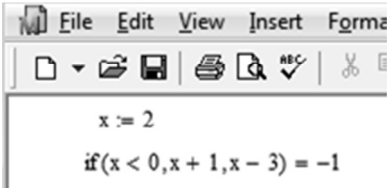


Рис. 40. Пример использования функции if()

Для проверки нескольких условий функция if() может быть вложенной в другую функцию if(), например, $\text{if}\left(\text{cond1}, x, \text{if}(\text{cond2}, z, y)\right)$.

Создание функций пользователя

Для того чтобы определить функцию пользователя, необходимо:

- ввести в желаемом месте документа имя функции;
- в скобках после имени ввести через запятую имена переменных, от которых зависит значение функции;
- ввести оператор присваивания с панели инструментов или нажатием клавиши двоеточие;
- ввести в появившийся местозаполнитель выражение, определяющее функцию, пользуясь клавиатурой или панелями инструментов (рис. 41).

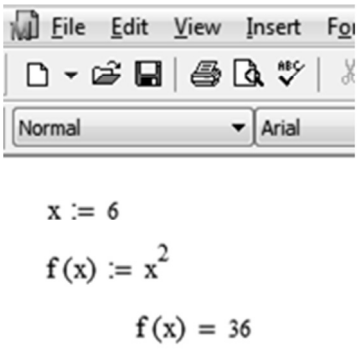


Рис. 41. Пример создания функции пользователя

Символьные вычисления в Mathcad

Символьные вычисления в Mathcad можно осуществлять в двух вариантах:

1) с помощью команд меню;

2) с помощью оператора символьного вывода → и ключевых слов.

Первый способ используется, когда требуется быстро получить результат для однократного использования.

Второй способ сохраняет символьные преобразования в документе и выдает результат с учетом содержания документа, обеспечивающего пересчет при его изменении. Оператор символьного вывода и ключевые слова вводятся с помощью панели **Symbolic**.

Упрощение выражений (simplify*)*

1. С помощью команд меню:

- во введенном выражении выделить синим уголком выражение или ту часть, которую нужно упростить;
- выбрать пункт меню **Symbolic**/**simplify**. При этом строкой ниже появится результат упрощения. Если результат не изменился, то упрощение произвести не удалось.

2. С помощью ключевого слова simplify:

- ввести выражение и ключевое слово simplify с помощью панели Символы;
- для получения результата щелкнуть мышкой за пределами выражения или нажать клавишу Enter (рис.42).

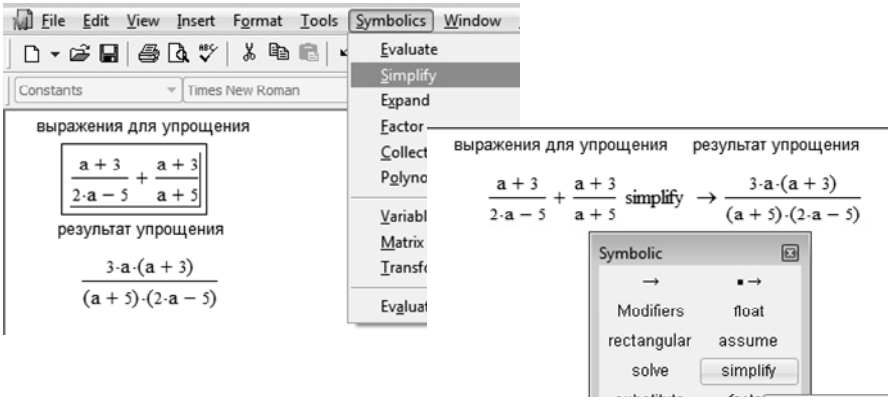


Рис. 42. Результат работы функции Упрощение выражений

Аналогичным образом выполняется полный набор символьных операций над выражениями:

- разложение на множители (factor);
- приведение подобных слагаемых полинома (collect);
- разложение на простые дроби (parfrac).

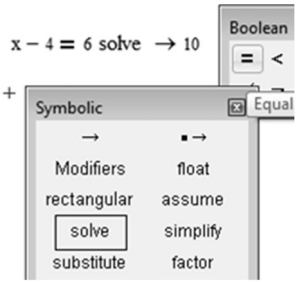


Рис. 43. Пример решения уравнения (*solve*)

Решение уравнений

1. С помощью меню:

- ввести уравнение, в котором вводится **логический** знак равенства (на экране он будет жирным) при помощи панели **Логический**;
- выделить в уравнении переменную, относительно которой оно решается;
- выбрать пункт меню **Symbolic**/**Variable**/**solve**. Если решение оказалось сложным, можно попытаться его упростить.

2. С помощью ключевого слова *solve*:

- ввести уравнение;
- ввести ключевое слово *solve* с помощью панели **Symbolic** и нажать Enter (рис. 43).

Контрольные вопросы

1. Основные элементы интерфейса системы Mathcad.
2. Математические панели инструментов, их назначение.
3. Основные правила ввода и редактирования текста.
4. Основные правила ввода и вычисления математических выражений.
5. На примерах показать, от чего зависит внешний вид курсора ввода.
6. Пояснить на примерах назначение синего уголка при вводе выражений.
7. Основные встроенные функции и их использование в выражениях.
8. Создание и применение функций пользователя.
9. Способы символьных вычислений в Mathcad.
10. Порядок упрощения и расширения выражений.
11. Возможности и порядок разложения выражений на множители.

12. Приведение подобных слагаемых и определение коэффициентов полиномов.

13. Каким образом производится разложение дробей на простые дроби?

14. Символьное решение алгебраических уравнений.

15. Какие символьные операции можно выполнять с матрицами и каким образом?

4.2. Матричные вычисления в Mathcad Построение графиков

Матричные вычисления в Mathcad

Массивами называют упорядоченные последовательности чисел или элементов массива. Доступ к любому элементу массива возможен по его индексу, т. е. номеру в последовательности чисел.

В Mathcad можно выделить два типа массивов:

- **векторы** (одноиндексные массивы), **матрицы** (двухиндексные) и **тензоры** (многоиндексные);
- **ранжированные переменные** – это разновидность вектора, в котором соседние элементы отличаются друг от друга на одну и ту же величину.

Номер первого элемента массива определяется значением системной переменной ORIGIN, которое по умолчанию равно нулю. Для того чтобы нумерация массивов начиналась с единицы, нужно в окне Mathcad присвоить этой переменной значение 1.

Mathcad допускает обращение к отдельным элементам матриц и векторов с помощью нижних индексов и к столбцам матриц с помощью верхних индексов. Нижние индексы вводятся с помощью кнопки ***Xn*** панели **Matrix** или клавишей, открывающей квадратные скобки ([). Нижние индексы, если их несколько, отделяются друг от друга запятой. Верхние индексы вводятся кнопкой $M^{\langle\rangle}$ панели **Matrix** или комбинацией клавиш Ctrl+*6*.

Создание ранжированных переменных

- Ввести имя переменной и оператор присваивания.
- Нажать кнопку **m..n** панели **Matrix** и в появившиеся местозаполнители ввести начальное и конечное значения переменной (шаг равен 1).

• Если необходимо другое значение шага, поместить синий уголок после первого числа, ввести запятую и значение второго числа.

Создание матриц и векторов с помощью панели Matrix

• Ввести имя переменной и оператор присваивания.

• Нажать кнопку панели с изображением матрицы (или комбинацию клавиш Ctrl+*M*).

• В появившемся диалоговом окне указать количество строк и столбцов и нажать кнопку OK.

• Ввести необходимые данные.

Создание матриц и векторов с помощью функции *matrix(M,N,f)*, где *M* и *N* – количество строк и столбцов; *f* – функция, определяющая значение элемента в зависимости от номера строки и столбца.

Основные операции с матрицами

• Сложение и вычитание.

• Умножение.

• Сложение, вычитание со скаляром.

• Умножение и деление на скаляр.

• Транспонирование матрицы.

• Вычисление определителя.

• Нахождение обратной матрицы.

• Возведение в целую степень (применяется к квадратным матрицам).

• Скалярное произведение векторов (используется кнопка (*x*) панели **Калькулятор**).

• Сумма элементов вектора.

Основные матричные функции

Выделение подматриц:

– submatrix(*A*,*ir*,*jr*,*ic*,*jc*) – выделение подматрицы матрицы *A* между строками *ir*, *jr* и столбцами *ic*, *jc* включительно;

– $A^{<i>}$ – выделение *i*-го столбца матрицы *A*.

Слияние матриц:

– augment(*A*,*B*,*C*,…) – слева направо;

– stack(*A*,*B*,*C*,…) – сверху вниз.

Размер матриц:

– rows(*A*) – количество строк;

– cols(*A*) – количество столбцов;

– length(*v*) – число элементов вектора;

– last(*v*) – индекс последнего элемента вектора.

Сортировка матриц (по возрастанию):
– sort(v) – сортировка элементов вектора;
– csort(A,i) – сортировка строк по столбцу i;
– rsort(A,i) – сортировка столбцов по строке i;
– reverse(v) – перестановка элементов вектора в обратном порядке.
Примеры работы с матрицами представлены на рис. 44.

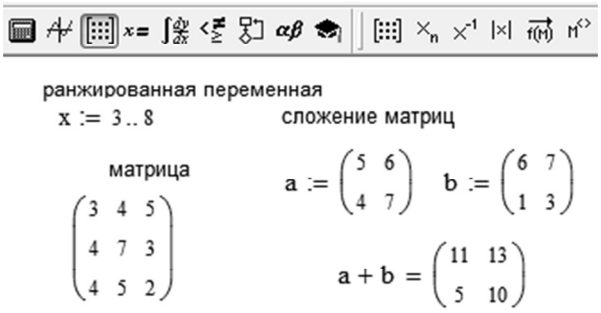


Рис. 44. Работа с матрицами

Решение систем линейных уравнений

Пример системы линейных уравнений.

$$\begin{cases} 2x_1 + 3x_2 + 5x_3 = 7, \\ 7x_1 + 9x_2 + x_3 = 3, \\ 6x_1 + 4x_2 + 3x_3 = 4. \end{cases}$$

1. Решение с помощью встроенной функции lsolve.

$$: X: = lsolve(A,B) \qquad X = \begin{pmatrix} -0.202 \\ 0.349 \\ 1.271 \end{pmatrix}$$

2. Решение матричным методом AX=B.

$$A: = \begin{pmatrix} 2 & 3 & 5 \\ 7 & 9 & 1 \\ 6 & 4 & 3 \end{pmatrix} \qquad B: = \begin{pmatrix} 7 \\ 3 \\ 4 \end{pmatrix} \qquad X: = A^{-1} \cdot B \qquad X = \begin{pmatrix} -0.202 \\ 0.349 \\ 1.271 \end{pmatrix}$$

Построение графиков в Mathcad

Mathcad позволяет строить следующие типы графиков.

- Двумерные графики:
  - *XY* график (в декартовой системе координат);
  - Полярный график (в полярной системе координат).
- Трехмерные графики:
  - график трехмерной поверхности;
  - график линий уровня;
  - трехмерная гистограмма;
  - трехмерное множество точек;
  - векторное поле.

Построение двумерных графиков

На одном графике можно построить до 16 различных зависимостей. Порядок построения следующий.

- Создать векторы для значений аргумента с некоторым шагом и соответствующих значений функции одним из способов.
- Поместить курсор в то место, куда требуется вставить график, и при помощи меню **Insert**/**Graph** или панели **Graph** выбрать *X*–*Y* график или Полярный график.
- В появившейся пустой области графика в местозаполнители возле осей ввести имена образованных векторов.

Mathcad также предоставляет возможность быстрого построения графиков, при этом не требуется предварительного создания векторов. Порядок построения следующий.

- Поместить курсор в то место, куда требуется вставить график, и при помощи меню **Insert**/**Graph** или панели **Graph** выбрать *X*–*Y* график или Полярный график.
- В местозаполнитель по оси *X* ввести имя аргумента, значение которого ранее не определялось. Значения аргумента в данном случае будут приняты по умолчанию в диапазоне от –10 до 10. При необходимости этот диапазон можно изменить.
- В местозаполнитель по оси *Y* ввести выражение, зависящее от этого аргумента, или указать необходимую функцию (в том числе и пользовательскую) этого аргумента (рис. 45).

Форматирование двумерных графиков

Для изменения диапазона осей перейдите к редактированию графика, щелкнув в его границах мышью. График будет выделен, и вбли-

зи каждой из осей появятся по два числа с нижней и верхней границей по оси. Измените эти числа, после выхода из режима редактирования график автоматически изменится для новых границ.

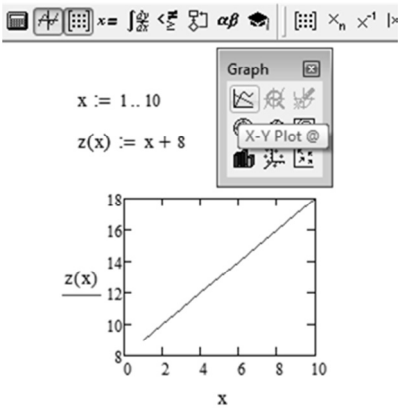


Рис. 45. Пример построения графика функции $z(x)$

Форматирование шкалы производится с помощью окна диалога форматирования графиков, которое вызывается из меню **Format**/**Graph** или контекстного меню.

Построение трехмерных графиков

Порядок построения следующий.

- Создать массив значений функции двух аргументов.
- Выбрать на панели **Graph** или в меню **Insert**/**Graph** одну из кнопок для построения трехмерных графиков.
- В местозаполнитель ввести имя образованного массива.

Функция CreateSpace(F(или f1, f2, f3, [t0, t1, tgrid, fmap]), где

F(t) – векторная функция из трех элементов, заданная параметрически относительно параметра t;

f1(t), f2(t), f3(t) – скалярные функции;

t0 – нижний предел t (по умолчанию –5);

t1 – верхний предел t (по умолчанию 5);

tgrid – число точек сетки по переменной t (по умолчанию 20);

fmap – векторная функция трех аргументов, задающая преобразование координат.

Функция CreateMesh(F(или g, или f1, f2, f3) [s0, s1, t0, t1, sgrid, tgrid, fmap]), где

F(s, t) – векторная функция из трех элементов, заданная параметрически относительно параметров s, t;

g(s, t) – скалярная функция;

f1(t), f2(t), f3(t) – скалярные функции;

s0, t0 – нижние пределы s, t (по умолчанию –5);

s1, t1 – верхние пределы s, t (по умолчанию 5);

sgrid, tgrid – число точек сетки по переменным s, t (по умолчанию 20);

fmap – векторная функция трех аргументов, задающая преобразование координат.

Аргументы, заключенные в квадратные скобки, являются необязательными, и их можно опустить (рис. 46).

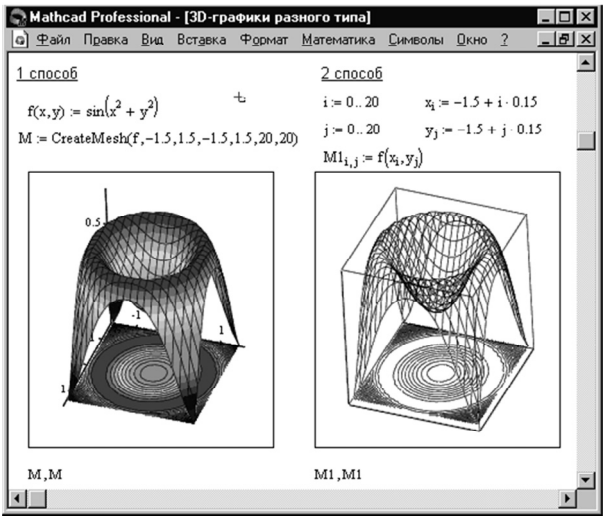


Рис. 46. Пример использования функции *CreateMesh()*

Форматирование трехмерных графиков выполняется аналогично двумерным графикам.

Порядок создания анимации

- Создать выражения и графики, зависящие от переменной номера кадра FRAME (рис. 47). Для предотвращения изменения масштаба графиков при анимации обычно следует отключить автоматическое

масштабирование и самостоятельно установить диапазон значений по осям.

- Выбрать пункт меню **Tools**/**Animation**/**Record**.
- В диалоговом окне задать номер первого и последнего кадров и скорость анимации (например, если установить номер первого кадра 0, последнего 50, то при скорости 10 кадров в секунду время анимации составит 5 с).
- Выделить область анимации. Область анимации выделяется штриховой линией.
- Нажать кнопку **Animate**.
- После окончания процесса появится окно видеопроигрывателя, в котором будет запущен просмотр анимации.
- При желании сохранить видеофайл при помощи **Сохранить как**. Этот файл в дальнейшем можно просматривать отдельно, например, в проигрывателе Windows.

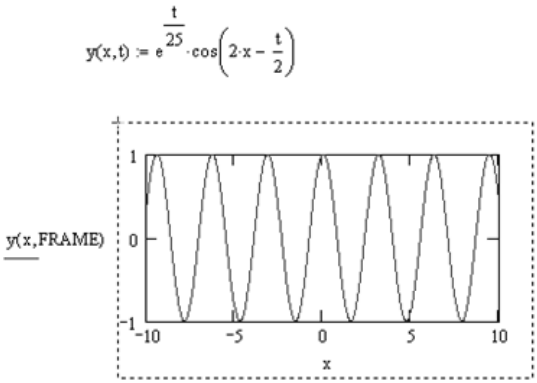


Рис. 47. Пример создания анимации

Контрольные вопросы

1. Типы массивов в Mathcad.
2. Способы создания ранжированных переменных, векторов и матриц.
3. Перечислите основные операции с матрицами.
4. Что такое векторизация? Приведите примеры.
5. Перечислите основные матричные функции.

6. Способы решения систем линейных уравнений.
7. Типы графиков, которые позволяет строить Mathcad.
8. Опишите способы создания двумерных графиков.
9. Перечислите возможности форматирования двумерных графиков.
10. Порядок построения трехмерных графиков.
11. Опишите порядок создания анимации.

4.3. Решение нелинейных алгебраических уравнений и систем в Mathcad. Элементы программирования

Функция root() – для решения нелинейных алгебраических уравнений (рис. 48).

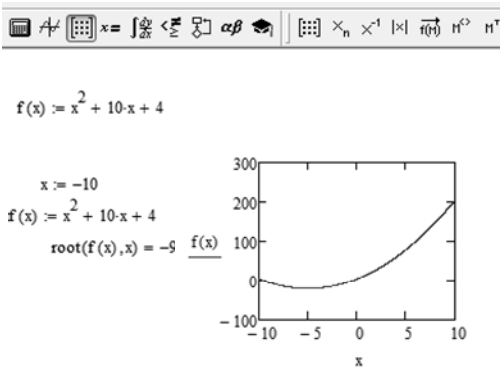


Рис. 48. Пример использования функции root()

Функция root используется для определения значения корня уравнения $f(x) = 0$ в одном из двух вариантов:

- $\text{root}(f(x), x)$;
- $\text{root}(f(x), x, a, b)$, где $f(x)$ – функция, определяющая уравнение; x – переменная, относительно которой решается уравнение; a, b – границы интервала, содержащего корень уравнения.

Функция polyroots() – для решения нелинейных алгебраических уравнений

Все корни полинома (в том числе и комплексные) можно определить при помощи функции polyroots(*v*), где *v* – вектор коэффициентов полинома.

Коэффициенты полинома записываются в вектор *v*, начиная со свободного члена, затем при первой степени аргумента, второй и т. д. Вектор коэффициентов можно получить также и при помощи символьных вычислений.

Пример решения уравнения $x3 - 4.5x2 - 7x + 10 = 0$;

$$f(x) := x3 - 4.5x2 - 7x + 10,$$

$$d := \begin{pmatrix} 10 \\ -7 \\ -4.5 \\ 1 \end{pmatrix} \quad \text{polyroots}(d) = \begin{pmatrix} -1.909 \\ 0.961 \\ 5.448 \end{pmatrix}.$$

Вычислительный блок Given – Find для решения систем уравнения

Для решения систем уравнений используется вычислительный блок, который состоит из трех частей:

- Given – ключевое слово;
- система уравнений, записанная **логическими** операторами в виде равенств и, возможно, неравенств;
- Find(*x*, *y*, *z*...) – встроенная функция для решения системы относительно переменных *x*, *y*, *z*,...

Так как эта функция использует для решения итерационные методы, то всем переменным должны быть присвоены начальные приближения до ключевого слова Given.

Пример решения системы уравнения:

$$\begin{cases} x^4 + y^2 - 3 = 0, \\ x^2 + 2y^3 + 2 = 0. \end{cases}$$

Задаем начальные приближения переменным:

$$x := 1 \quad y := 1.$$

Решаем систему.

Given

$x^4 + y^2 - 3 = 0,$

$x^2 + 2 \cdot y^3 + 2 = 0.$

M: = Find(x, y)

$$M = \begin{pmatrix} 1.127 \\ -1.178 \end{pmatrix}.$$

Погрешность выполнения уравнений определяется системной константой *CTOL*, по умолчанию равной 0.001. При необходимости можно присвоить ей и другое значение.

В некоторых особых случаях (например, если значения производных возле корня близки к нулю) функция Find может и не найти решения. В таких случаях можно попытаться использовать вместо функции **Find()** функцию **Minerr()** с теми же параметрами. Функция Minerr() минимизирует невязку системы уравнений и срабатывает даже при отсутствии решения.

Панель инструментов Mathcad Progamming

Для вставки программного кода в документы в Mathcad имеется специальная панель инструментов **Programming**, которую можно вызвать на экран нажатием кнопки **Programming Toolbar** на панели **Math**, как показано на рис. 49. Большинство кнопок этой панели выполнено в виде текстового представления операторов программирования, поэтому их смысл легко понятен.

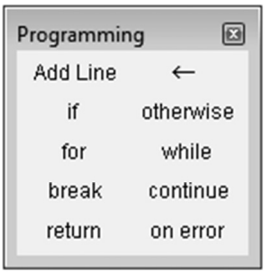


Рис. 49. Панель инструментов Programming

Основными инструментами работы в Mathcad являются математические выражения, переменные и функции. Нередко записать формулу, использующую ту или иную внутреннюю логику (например, возвращение различных значений в зависимости от условий), в одну строку не удается. Назначение программных модулей как раз и заключается в определении выражений, переменных и функций. Несмотря на принципиальную эквивалентность определения функций и переменных через встроенные функции Mathcad или программные модули, программирование имеет ряд существенных преимуществ, которые в ряде случаев делают документ более простым и читаемым:

- возможность применения циклов и условных операторов;
- простота создания функций и переменных, требующих несколько простых шагов;

• возможность создания функций, содержащих закрытый для остального документа код, в том числе преимущества использования локальных переменных и обработка исключительных операций (ошибок).

Создание программного модуля

Программный модуль обозначается в Mathcad вертикальной чертой, справа от которой последовательно записываются операторы языка программирования. Чтобы создать программный модуль, необходимо:

• ввести часть выражения, которая будет находиться слева от знака присваивания, и сам знак присваивания;

• при необходимости вызвать на экран панель инструментов Programming;

• нажать на этой панели кнопку Add Line (Добавить линию);

• если приблизительно известно, сколько строк кода будет содержать программа, можно создать нужное количество линий повторным нажатием кнопки Add Line соответствующее число раз;

• в появившиеся местозаполнители ввести желаемый программный код, используя программные операторы.

Ни оператор присваивания**:=**, ни оператор вывода = в пределах программ не применяются.

Условный оператор IF

Действие условного оператора *if* состоит из двух частей. Сначала проверяется логическое выражение (условие) справа от него. Если оно истинно, выполняется выражение слева от оператора *if*. Если ложно – ничего не происходит, а выполнение программы продолжается переходом к ее следующей строке. Вставить условный оператор в программу можно следующим образом (рис. 50):

• если необходимо ввести левую часть выражения и оператор присваивания;

• создать новую строку программного кода, нажав на панели **Programming** (Программирование) кнопку *Add Line*;

• нажать кнопку условного оператора *if*;

• справа от оператора *if* ввести условие, пользуясь логическими операторами, вводя их с панели Boolean (булевы операторы);

• выражение, которое должно выполняться, если условие истинно, ввести слева от оператора *if*;

- если в программе предусматриваются дополнительные условия, добавить в программу еще одну строку нажатием кнопки *Add Line* и ввести их таким же образом, используя операторы *if* или *otherwise*.

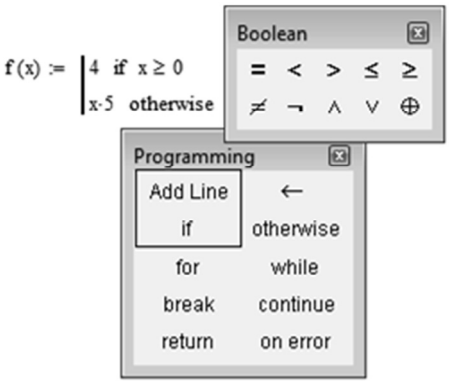


Рис. 50. Вставка условного оператора

Оператор *otherwise* используется совместно с одним или несколькими условными операторами *if* и указывает на выражение, которое будет выполняться, если ни одно из условий не оказалось истинным.

Пример. Пусть необходимо вычислить значение следующей функции:

$$f(x,y)=\begin{cases} x^2+\cos(y), & \text{если} \quad 15\ge x>-5, \\ 0, & \text{в остальных случаях.} \end{cases}.$$

Вычисления $f(x,y)$ выполнены с помощью оператора *if* и оператора *otherwise* и показаны на рис. 51.

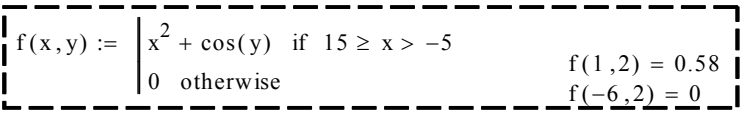


Рис. 51. Пример выполнения оператора *if*

Условный оператор FOR, WHILE

В языке программирования Mathcad имеются два оператора цикла: *for* и *while*. Первый из них дает возможность организовать цикл по

некоторой переменной, заставляя ее пробегать некоторый диапазон значений. Второй создает цикл с выходом из него по некоторому логическому условию. Чтобы вставить в программный модуль оператор цикла, необходимо:

- создать в программном модуле новую линию;
- вставить один из операторов цикла, *for* или *while*, нажатием одноименной кнопки на панели Programming;
- если выбран оператор *for* (рис. 52), то вставить в соответствующие местозаполнители имя переменной и диапазон ее значений, а если *while*, то логическое выражение, при нарушении которого должен осуществляться выход из цикла;

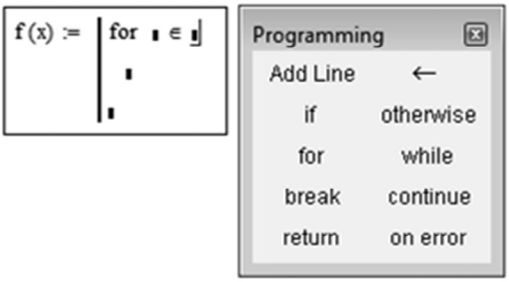


Рис. 52. Вставка оператора цикла

- в нижний местозаполнитель ввести тело цикла, т. е. выражения, которые должны выполняться циклически;
- при необходимости дополнить программу другими строками.

Пример. Найти сумму первых десяти натуральных чисел. До начала следует присвоить сумме *s* нулевое значение. Так как число циклов известно, используем оператор *for*:

$$
\begin{aligned}
s = \;& s \leftarrow 0 \\
& \text{for } x \in 1...10 \\
& s \leftarrow s + x \\
& s = 55
\end{aligned}
$$

Контрольные вопросы

1. Каким образом определяется значение корня линейного уравнения при помощи функции root()?

2. Как выбрать начальное приближение при численном решении алгебраического уравнения?

3. Каким образом задается точность решения алгебраического уравнения?

4. Как влияет выбор точности решения алгебраического уравнения на результат решения?

5. Как определяются корни полинома функцией polyroots()?

6. Приведите структуру блока решения системы нелинейных уравнений.

7. Как выполняется установка параметров решения системы нелинейных уравнений?

5. ОСНОВЫ ПРОГРАММИРОВАНИЯ НА ЯЗЫКЕ СИ

5.1. Язык программирования Си Работа в интегрированной среде разработки программ Borland C

Интегрированная среда разработки программ – программная система, обслуживающая весь цикл создания исполняемых файлов без выхода в операционную систему и обращения к другим программам. В ее состав входят: многооконный текстовый редактор, компилятор, редактор связей (линковщик), отладчик, подсистема работы с файлами, Help-система.

Для вхождения в среду запускается файл bc.exe. После его загрузки на дисплее отображается основной экран среды. Верхняя строка экрана разбита на одиннадцать именованных полей, каждое из которых представляет собой пункт главного меню. В нижней строке (строке состояния) указывается назначение «горячих клавиш», воспринимаемых средой на текущем этапе работы. Остальная часть экрана – рабочая поверхность. На ней размещаются окна – области, очерченные рамкой, в которых выводится информация различного характера. Одновременно может быть открыто несколько окон. По характеру выводимой в окнах информации они делятся: на окна редактирования, окна диалога, окна помощи, окно сообщений, выходное окно (рис. 53).

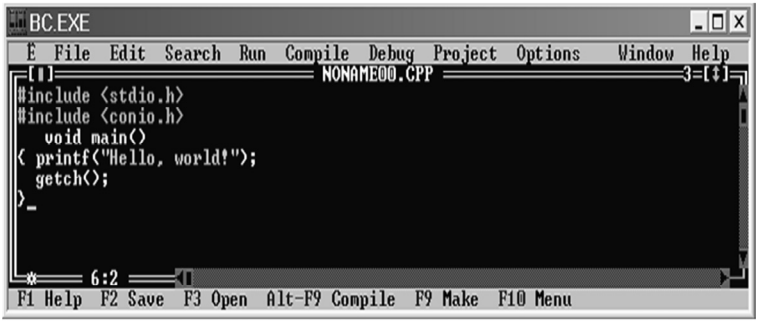


Рис. 53. Окно интегрированной среды Borland C 3.1

При работе в среде главную роль играют окна редактирования. В них создаются тексты программ, и из них программы запускаются на компиляцию и линкование. Если при разработке программы в ней были допущены ошибки, то компилятор или линковщик сообщают об этом в специальном окне сообщений (оно именуется “Message”). В нем для каждого сообщения отводится одна строка. Компилятор выдает три вида сообщений:

- о неисправимых ошибках – начинаются со слова “Fatal”, за которым указывается характер ошибки;
- о синтаксических и лексических ошибках – начинаются со слова “Error”, за которым указывается номер строки в тексте программы, содержащей ошибку, и затем характер ошибки;
- предупреждающие сообщения – начинаются со слова “Warning”, за которым указывается само сообщение. Следует иметь в виду, что предупреждающее сообщение ошибкой не является и в принципе программа компьютером может быть отработана, но в процессе ее исполнения могут возникнуть проблемы.

Все окна (кроме диалоговых) обозначены идентифицирующим номером, расположенным вблизи правого верхнего угла.

В любой момент активно или главное меню, или одно из окон. Активизация окна производится одновременным нажатием клавиши Alt и цифровой клавиши, соответствующей номеру окна; активизация главного меню – нажатием клавиши F10. Каждый пункт главного меню содержит определенные команды среды.

В верхней строке располагаются названия пунктов главного меню, краткое назначение которых таково:

File – общение с файловой подсистемой;

Edit – ввод и редактирование исходной программы;

Search – поиск в тексте исходной программы;

Run – запуск программы в автоматическом или пошаговом режиме;

Compile – компиляция исходной программы;

Debug – отладка программы;

Project – управление проектом сборки программы из нескольких модулей;

Options – настройка параметров интегрированной среды;

Window – управление дополнительными окнами системы программирования;

Help – обращение к файлам помощи.

Команды меню **File** (рис. 54) используются при наборе новой программы (команда **New**), при запоминании в файле на диске набранной или измененной программы (команды **Save** и **Save as**), при вызове ранее сохраненной программы (команда **Open**). Последняя команда этого меню (**Quit**) исполняется при выходе из интегрированной среды.

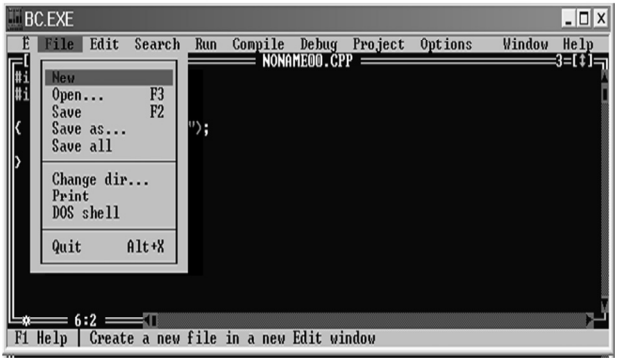


Рис. 54. Команды меню File

Вход в меню **File** происходит либо после щелчка мышью по заголовку **File**, либо после набора клавишной комбинации Alt+F. Выполнение наиболее употребимых команд также продублировано нажатием одной функциональной клавиши или соответствующей клавишной комбинации.

Переход к той или иной команде выбранного меню сопровождается появлением в нижней строке подсказки. На рис. 45 выделена команда **New**, а в строке подсказки находится сообщение «Создание нового файла в новом окне редактирования».

До тех пор, пока при сохранении набранной программы соответствующему дисковому файлу не присвоено индивидуальное имя, вновь набираемая программа выступает под именем NONAMEnn.CPP, т. е. программа «безымянная» (здесь nn – порядковый номер безымянной программы, созданной в течение одного сеанса).

Одной из наиболее часто используемых групп команд являются строки меню **Run** (рис. 55). По команде **Run** производится попытка запуска на выполнение программы, находящейся в текущем окне редактирования. При этом новая программа сначала компилируется, потом редактируются ее связи с другими модулями, затем полученный исполняемый модуль загружается в оперативную память и начинает выполняться.

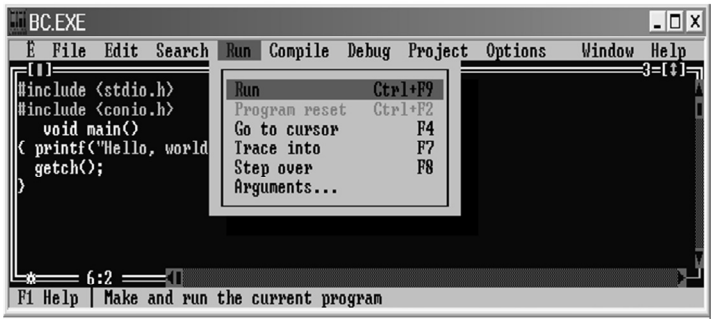


Рис. 55. Команду меню Run

Во время выполнения программы ее работа может быть прервана системой, обнаружившей исключительную ситуацию (деление на нуль, переполнение и т. д.), или пользователем. В ряде случаев для продолжения работы необходимо восстановить первоначальное состояние программы. Для этой цели используется команда **Program reset**.

Командой **Go to cursor** пользуются для автоматического запуска программы с остановом в той строке исходного текста, где находится курсор. Обычно такой режим используется при отладке программы. Также для отладочных целей используется пошаговое выполнение программы.

В этом режиме очередное нажатие функциональной клавиши **F7** или **F8** приводит к останову после выполнения очередной строки исходного текста. Нажатие **F7** приводит к тому, что пошаговое исполнение сохраняется и в вызываемой функции. В отличие от этого нажатие

F8 приводит к автоматическому выполнению вызываемой функции и останову после возврата из нее. Очень важно уметь пользоваться справочной системой интегрированной среды – командами меню **Help** (рис. 56).

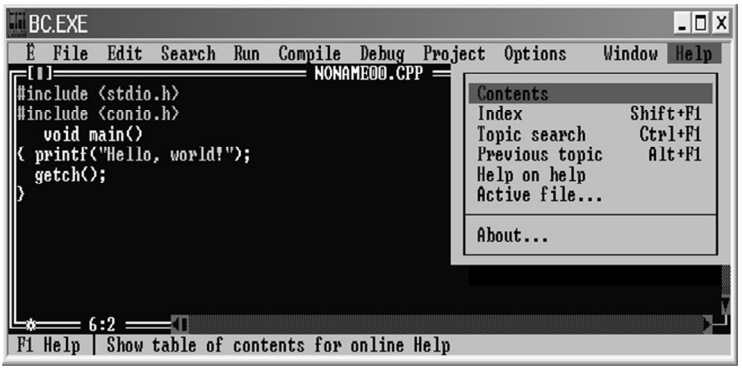


Рис. 56. Команды меню Help

Справочная система предназначена для работы в двух режимах. Первый из них, называемый контекстной помощью, срабатывает следующим образом. Курсор мыши подводится к интересующему вас служебному слову или наименованию системной функции, и нажимается клавиша F1. Второй способ связан с использованием команд меню **Help** – **Contents** (Содержание) и **Index** (Указатель терминов, упорядоченных по алфавиту).

По команде **Contents** на экране (рис. 57) появляются названия справочных разделов:

- How to Use Help – Как использовать помощь;
- Menus and Hot Keys – Меню и «горячие» клавиши (клавишные команды);
- Editor Commands – Команды редактора;
- Assembler (built-in) – Встроенный ассемблер;
- Borland C++ Language – Язык C++ среды Borland;
- Command Line – Использование командной строки;
- Container Classes – Классы-контейнеры;
- Error Messages – Сообщения об ошибках;
- Functions – Функции;
- Header Files – Заголовочные файлы;

- IOStream Classes – Потоковые классы ввода/вывода;
- Utilities – Сервисные программы.

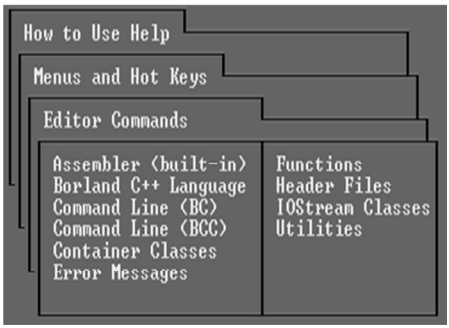


Рис. 57. Разделы файла помощи системы BC 3.1

По команде **Index** на экране появляется упорядоченный по алфавиту список (рис. 58), в котором можно выбрать интересующий вас термин и перейти к соответствующему кадру помощи.

Рис. 58. Фрагмент списка терминов

Структура программы на языке С

Продемонстрируем пример простейшей программы на языке С, которая запрашивает у пользователя два целочисленных значения переменных *a* и *b*, сравнивает их и выводит наибольшее число.

```
01  #include <iostream.h>
02  #include <conio.h>
```

```
03  int main(void)
04  {
05   int a,b,max;
06   printf("Введите a=");      //приглашение ввести значение a
07   scanf("%d", &a);          //ввод значения переменной a
08   printf("Введите b=");      //приглашение ввести значение b
09   scanf("%d",&a);           //ввод значения переменной b
10   if(a>b) max=a;            //если a>b то max=a
11   else max=b;               //иначе max=b
12   printf("max=", max);      //вывод максимального значения
13   getch();                  //останов до нажатия клавиши
15  }
```

Номера, которые проставлены в начале каждой строки программы, не являются принадлежностью программы. Они введены только для упрощения ссылок на описание действия тех или иных строк.

Строки 01 и 02 подключают (include – включить) к тексту программы так называемые *заголовочные* (h от header – заголовок) файлы системы. В этих файлах описаны системные функции и их аргументы.

Строка 03 содержит заголовок функции main. Функция с таким названием обязана присутствовать в каждой программе на языке C, C++. Именно с нее начинается выполнение программы, она – главная (именно так переводится служебное слово main). Предшествующее ей служебное слово int (от integer – целый) сообщает, что результатом работы функции main должно быть целое число. Служебное слово void (дословно – пустота), указанное в круглых скобках, сообщает, что у функции main аргументы отсутствуют.

Текст программы (тело функции) заключается в фигурные скобки (строки 04 и 15). В большинстве последующих строк присутствует пояснительный текст на русском языке, следующий после пары символов // – это комментарий, на содержание которого система не обращает внимания.

В строке 05 объявлены три переменные с именами *a*, *b* и max, которые могут принимать только целочисленные значения (тип – *int*).

Строка 06 является первой строкой программы, которая производит некоторое действие – она выводит на дисплей сообщение, состоящее из двух символов (*a*=). Текст сообщения заключен в двойные кавычки.

Строка 07 организует приостановление работы программы до тех пор, пока пользователь не наберет на клавиатуре какое-либо число и не нажмет клавишу Enter. Поступившее значение будет хорошо воспри-

нято, если оно целое, и направлено в переменную *a*. Точно таким же образом в строках 08 и 09 будет организован ввод значения числовой переменной *b*.

В строке 10 сравниваются (*if* – если) текущие значения переменных *a* и *b*. Если проверяемое условие выполнено, т. е. значение переменной *a* больше, то оно присваивается переменной max – выполняется действие, записанное после проверки условия. В противном случае (else – иначе) в переменную max заносится значение *b*.

Строка 12 выводит на дисплей два сообщения – текстовое (max=) и числовое (значение переменной max).

Обращение к функции getch (строка 13) приводит к задержке на экране сообщения программы до тех пор, пока пользователь не нажмет какую-либо клавишу (getch – от get character, дай символ).

Контрольные вопросы

1. Для чего необходима среда программирования Borland C?
2. Назовите основные возможности среды Borland C.
3. Для чего необходима блок-схема алгоритма?
4. Зачем в среде Borland C необходим текстовый редактор?
5. Какие виды сообщений являются результатом работы компилятора?
6. В чем суть процесса компиляции программы?
7. Какие методы отладки содержит среда Borland C?
8. Что позволяет режим трассировки?
9. В чем отличие работы с локальной и глобальной переменной?
10. Опишите принцип работы с окном вывода значений переменных.

5.2. Язык программирования Си
Основные типы данных в Си
Консольный ввод-вывод данных

Для того чтобы компьютер мог производить обработку данных, они должны быть размещены в его памяти. В Си любая область памяти, которая используется компьютером, называется объектом. Все объекты описываются с помощью специальных записей в программе, называемых объявлениями или определениями. Они задают такие при-

знаки объекта, как имя, тип, область действия и время жизни. Описания объектов имеют следующий формат:

<спецификатор типа> <описатель> [=<инициализатор>];

где **<спецификатор типа>** – одно или несколько ключевых слов, определяющих тип объекта;

<описатель> – элемент описания, по которому устанавливаются имя и структура объекта (при описании простой переменной описатель представляет собой идентификатор, при описании массива за его именем указываются открывающая и закрывающая квадратные скобки и т. д.);

<инициализатор> – начальное значение (список начальных значений), присваиваемое объекту при его объявлении.

Наиболее часто используемые простейшие типы данных приведены в табл. 5.

Таблица 5

Простейшие типы данных

Обозначение (ключевое слово)	Наименование	Значение объекта
int	Целый	Целое число
float	С плавающей точкой одинарной точности	Вещественное число
double	С плавающей точкой двойной точности	
char	Символьный	Символ

При организации ввода информации с клавиатуры и вывода ее на дисплей (консольного ввода-вывода данных) в Си-программах используются стандартные библиотечные функции. Для обеспечения возможности обращения к ним в программе обязательно должна указываться директива препроцессора #include<stdio.h>.

Различают два вида консольного ввода-вывода: форматированный и неформатированный.

Форматированный ввод-вывод позволяет за одно обращение к библиотечной функции обрабатывать несколько объектов данных. Запись оператора обращения к стандартной функции форматного консольного вывода имеет вид

printf (“<строка формата>” [,<аргументы>]);

где **строка формата** – совокупность необязательных символов, задаваемых программистом, и спецификаций формата преобразования данных;

аргументы – имена объектов, перечисленные через запятую.

Каждому объекту из списка, следующего за строкой формата, должна соответствовать своя спецификация формата преобразования данных. При выводе данных на ее место подставляется значение объекта. Общий вид записи спецификации формата преобразования

%[<флаг>][<ширина>][.<точность>]<тип преобразования >

где **<флаг>** – символ, устанавливающий форму вывода ("–" – выравнивание влево, "+" – вывод знака числа и т. д.);

<ширина> – число, устанавливающее максимальную ширину поля вывода;

<точность> – число, устанавливающее максимальное количество позиций после десятичной точки, предназначенных для вывода данных;

<тип преобразования> – символ, указывающий, как должны трактоваться данные, хранящиеся в памяти компьютера при их выводе на экран.

Используются следующие символы для задания типа преобразования данных при их выводе:

c – вывод одного символа;

d – вывод десятичного целого числа;

f – вывод десятичного вещественного числа в естественной форме представления;

e – вывод десятичного вещественного числа в экспоненциальной форме представления;

s – вывод строки символов.

Три первых элемента записи спецификации формата преобразования в функции printf (флаг, ширина, точность) являются необязательными.

Запись оператора обращения к стандартной функции форматного консольного ввода имеет вид

scanf ("<строка формата>" [,<адреса аргументов>]);

где **<строка формата>** – совокупность спецификаций формата преобразования данных, разделяемых между собой пробельными символами;

<адреса аргументов> – адреса объектов, перечисленные через запятую.

В отличие от функции printf в функции scanf в качестве аргументов используются *не имена объектов, а их адреса*. Для того чтобы указать адрес объекта, нужно перед его именем записать символ “&” (символ амперсанд).

Спецификация формата преобразования данных в функции scanf имеет общий вид записи

%[<ширина>]<тип преобразования >

где **<ширина>** – необязательное число, устанавливающее максимальную ширину поля ввода;

<тип преобразования> – обязательный символ, указывающий, как должны трактоваться данные, вводимые с клавиатуры, при помещении их в память компьютера.

Используются следующие символы для задания типа преобразования данных при их вводе:

c – ввод одного символа;

d – ввод десятичного целого числа;

g – ввод десятичного вещественного числа;

s – ввод строки символов.

Неформатированный ввод-вывод обеспечивает за одно обращение к библиотечной функции обработку только одного объекта данных, значениями которых могут быть одиночные символы и их последовательности. Для ввода одиночного символа используется функция getchar(). Пример записи оператора обращения к ней имеет вид

Sy = getchar( );

По данному оператору функция getchar() считывает введенный с клавиатуры символ и помещает его в оперативную память компьютера по адресу переменной Sy.

Кроме функции getchar имеется аналогичная ей функция getch(). Она осуществляет ввод одиночного символа с клавиатуры компьютера без копирования на экран. Это делает ее очень удобной для организации паузы при исполнении программы. Как только встретится оператор

getch( );

компьютер переходит в режим ожидания ввода символа, и никаких действий не производится до тех пор, пока не будет нажата какая-либо клавиша.

Для вывода одиночного символа используется функция putchar(). Пример записи оператора обращения к ней имеет вид

putchar (<аргумент>);

где **<аргумент>** – имя выводимого объекта (аргументом может быть символьная константа).

Неформатированный ввод-вывод символьных строк выполняют соответственно функции

gets(<имя строки>) и **puts(<имя строки>)**

В них имя **<имя строки>** – область памяти, в которой хранится символьная строка.

Для вывода таблицы можно использовать либо вывод стандартных символов клавиатуры «---- или |», либо вывод символов псевдографики.

Пример вывода стандартных символов клавиатуры:

```
printf("--------------------------------");
printf("| |");
```

Пример вывода символов псевдографики:

printf("%c", код), где код – числовое значение кода элемента псевдографики (рис. 59).

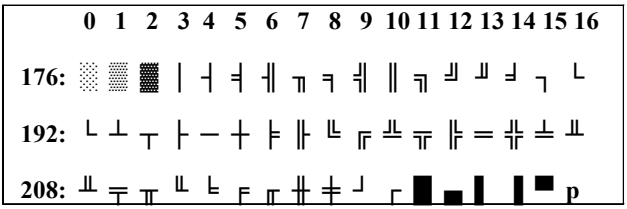


Рис. 59. Фрагмент таблицы элементов псевдографики

Операции и выражения

Действия, которые должны выполняться над данными, задаются в программе с помощью знаков операций. Операция – это один из базовых элементов конструкции программ. Он представляет собой условное обозначение (символ или совокупность символов) команды микропроцессора, которую тот должен выполнить над величинами, называемыми операндами. Комбинация знаков операций и операндов

образует конструкцию языка, называемую выражением. Простейшее выражение состоит из знака операции и одного операнда.

Например, –5 или *–a*.

Простые выражения можно рассматривать как операнды, на базе которых строятся другие более сложные выражения. Например, *–5 + b.*

В языке Си богатый набор операций, классифицируемых по различным признакам и позволяющих выполнять самые разнообразные действия по обработке данных (табл. 6).

Таблица 6

Операции и выражения

№ п/п	Группа	Операция		Выполняемое действие	Пример использования
		Название	Знак		
1	Операции выбора	Обращение к функции	()	Обращение к функции. Выделение компоненты в выражении	main() (2+3)*-5
		Индексирование	[]	Обращение к элементу массива	Stroke[8]
2	Унарные	Логическое отрицание	!	Если величина справа от символа операции равна нулю, то результат равен единице (истина), иначе результат равен нулю (ложь)	!var
		Адресация	&	Дает адрес величины, стоящей справа	&var
		Приведение	(спецификатор типа)	Приведение операторов к одному типу	(float)
		Определение размера	sizeof	Установление размера в байтах типа или объявленной переменной	sizeof(char)
3	Арифметические операции	Изменение знака	–	Изменяет знак величины, стоящий справа от символа операции П, на –П	-5
		Увеличение на единицу	+ +	Прибавляет единицу префиксной или постфиксной форме	*c = a++*, сначала *c = a* потом к *c* + 1
		Умножение	*	Умножает П на Л	*a * d*
		Деление	/	Делит Л на П	*a/b*

Продолжение табл. 6

№ п/п	Группа	Операция		Выполняемое действие	Пример использования
		Название	Знак		
		Уменьшение на единицу	– –	Вычитает единицу из префиксной (П) или постфиксной формы (Л)	*C*=--*a*, сначала a-1, потом резул. в c
		Сложение	+	Прибавляет П к Л	*f*+*b*
		Деление по модулю	%	Дает остаток от деления Л на П	*a*%*b*
		Вычитание	–	Вычитает П из Л	*a*-*b*
4	Операции отношения (сравнения)	Меньше	<	Дает результат 1 (истина), если Л меньше П, иначе результат 0 (ложь)	*a*<*b*
		Меньше или равно	<=	Дает результат 1 (истина), если Л меньше или равно П, иначе результат 0 (ложь)	*a*<=*b*
		Больше	>	Дает результат 1 (истина), если Л больше П, иначе результат 0 (ложь)	*a*>*b*
		Больше или равно	>=	Дает результат 1 (истина), если Л больше или равно П, иначе результат 0 (ложь)	*a*>=*b*
5	Логические операции	Равно	= =	Дает результат 1 (истина), если Л равно П, иначе результат 0 (ложь)	*a*==*b*
		Не равно	!=	Дает результат 1 (истина), если Л не равно П, иначе результат 0 (ложь)	*a*!=*b*
		Логическое «И»	&&	Дает результат 1 (истина), если Л и П имеют значение 1, иначе результат 0 (ложь)	*a*&&*b*
		Логическое «ИЛИ»	||	Дает результат 1 (истина), если Л или П имеют значение 1, иначе результат 0 (ложь)	*a*||*b*

Окончание табл. 6

№ п/п	Группа	Операция		Выполняемое действие	Пример использования
		Название	Знак		
6	Операции присваивания	Присваивание	=	Присваивает Л значение П	*a=b*
		Присваивание произведения	*=	Умножает Л на П и присваивает полученное значение Л	*a*=b*, рез: *a=a · b*
		Присваивание частного	/=	Делит Л на П и присваивает полученное значение Л	*a*/=b*, рез:a=(*a*/*b*)
		Присваивание остатка	%=	Выдает остаток от деления Л на П	*a*%=b*, рез:*a*=остат. от /b
		Присваивание суммы	+=	Прибавляет П к Л	*a*+=b*, рез: *a*=(*a*+b)
		Присваивание разности	-=	Вычитает П из Л	*a*-=b*, рез: *a*=(*a*-*b*)

В сложном выражении применяются различные операции.

Например: $a \cdot (b + c / d)$.

В этом выражении указаны три операции. Какая из них должна выполняться первой? Очевидно, что изменение порядка выполнения действий приведет к различным результатам. Поэтому в языках программирования используется набор непротиворечивых правил, устанавливающих, какие действия и в какой последовательности должны выполняться. Реализуются эти правила путем задания приоритета старшинства операций. Если две операции имеют одинаковый приоритет, то они выполняются в той последовательности, в которой записаны в выражении. Наивысший приоритет имеет операция выделения компонента (круглые скобки). Она выполняется в первую очередь.

Контрольные вопросы

1. Из каких разделов состоит программа на языке Си?
2. Что такое оператор?
3. Какие операторы языка Си вам известны?
4. Зачем нужен оператор присваивания? Какой вид он имеет?
5. Что такое переменная? Что такое константа?
6. Какая команда служит для ввода данных?
7. Какой формат записи имеет команда ввода?

8. Какая команда служит для вывода данных?
9. Что такое форматный вывод?
10. Как разместить комментарии в программе?

5.3. Программирование линейных и разветвляющихся вычислительных процессов

При программировании вычислительных процессов используются математические функции библиотеки компилятора. Для обеспечения возможности обращения к ним в программе должна указываться директива препроцессора #include <math.h>. Все математические функции результатом своей работы имеют вещественные числа типа double. Описание некоторых функций дано в табл. 7.

Таблица 7

Основные математические функции

Функция	Назначение функции	Значения аргументов
sin(x)	Вычисляет синус угла x	Вещественное число В диапазоне от -2π до 2π
cos(x)	Вычисляет косинус угла x	
tan(x)	Вычисляет тангенс угла x	
asin(x)	Вычисляет арксинус величины x	Вещественное число в диапазоне от –1 до 1
acos(x)	Вычисляет арккосинус величины x	
atan(x)	Вычисляет арктангенс величины x	
sqrt(x)	Вычисляет корень квадратный из x	Любое вещественное число
exp(x)	Вычисляет значение e в степени x	
log(x)	Вычисляет натуральный логорифм от x	
log10(x)	Вычисляет десятичный логорифм от x	
ceil(x)	Округляет число x путем отбрасывания дробной части	
floor(x)	Округляет число x до наибольшего целого, не превышающего x	
pow(x, y)	Вычисляет значение x в степени y	Любые вещественные числа
fmod(x, y)	Вычисляет остаток от деления x на y	

Для организации разветвляющихся процессов используются условный оператор **if** и оператор-переключатель **switch**.

Условный оператор if предназначен для организации процедуры выбора одного из двух вариантов продолжения выполнения программы. Запись этого оператора имеет следующий вид.

Формат записи оператора **if:**

if(<условное выражение>) <оператор1>
[else <оператор2>]

Выполнение оператора начинается с вычисления условного выражения. Далее выполнение идет по следующей схеме:

– если условное выражение истинно, то выполняется оператор 1;

– если условное выражение ложно, то выполняется оператор 2;

– если условное выражение ложно и в операторе *if* отсутствует фраза else, то выполняется оператор, следующий за оператором *if* (в данном случае это оператор 2).

Структурная схема оператора *if* имеет вид, представленный на рис. 60. Оператор *if* может быть вложен во фразу *if* или во фразу else другого оператора *if*. Использование вложенных операторов позволяет строить довольно сложные конструкции по выбору одного варианта из множества. Чтобы сделать программу более понятной для чтения, рекомендуются использовать во вложенных операторах блоки (составные операторы).

$$a = \begin{cases} 5x, & x < 5, \\ 10x, & x = 5, \\ 20x, & x > 5. \end{cases}$$

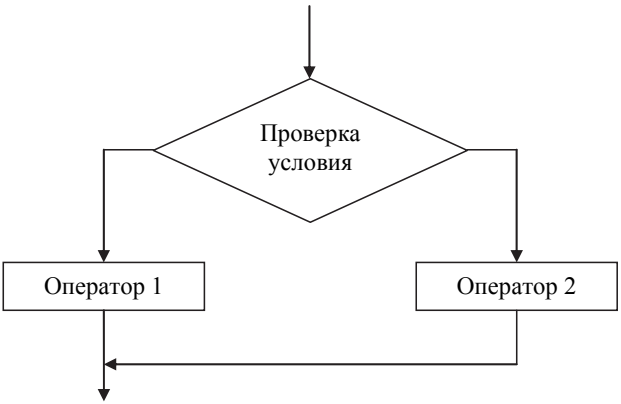


Рис. 60. Условный оператор *if*

Пример использования оператора *if.*

```
#include<stdio.h>
void main(void)
{int a,x;
clrscr();
printf("\nЗадайте значение аргумента");
scanf("%d",&x);
if(x<5)
    a=5*x;
else
  {if(x==5)
     a=10*x;
       else
       a=20*x;
}
printf("\n a=%d",a);
}
```

Кроме оператора *if* в языке Си имеется еще один оператор *switch*, позволяющий организовать выбор одного варианта из множества.

Формат записи оператора **switch:**

```
switch(<выражение>) {
case<константа1>: <оператор1>
                  [break;]
case<константа2>: <оператор2>
                  [break;]
.....................................................
default: <оператор>
}
```

Значением выражения, следующего в круглых скобках за ключевым, должна быть величина целого типа (включая тип char). Она является критерием для выбора из нескольких вариантов одного.

Константы, следующие за ключевым словом case, также должны быть целого типа. В совокупности с ключевым словом case константы являются метками, которыми помечаются возможные варианты выполнения программы.

Порядок выполнения оператора switch следующий:

– сначала вычисляется выражение в круглых скобках;

– значение вычисленного выражения последовательно сравнивается с константами, следующими за ключевым словом case;

– если сравнение найдено, то управление ходом исполнения программы передается оператору, помеченному соответствующей меткой;

– если сравнение не найдено и есть вариант, помеченный ключевым словом default, то управление передается оператору этого варианта;

– если сравнение не найдено и варианта default нет, то управление передается оператору, следующему за оператором switch;

– если в теле выбранного оператора передача управления за пределы оператора switch не предусмотрена (чаще всего для этого используется оператор break), то после его выполнения управление передается следующему оператору.

Пример использования оператора **switch** «Простейший калькулятор»:

```
# include<stdio. h>
void main(void)
{
   float a;
   float b;
   char c;
   clrscr();
   printf("\nВведите первое число a= ";
   printf("\nВведите второе число b= ";
   printf("\nВведите действие над числами +,-,/ или *");

ab: scanf("%c",&c);
switch (c)
      {
         case `+`: printf("\Сумма чисел a и b = %f", a+b); break;
         case `-`: printf("\nРазность чисел a и b = %f", a-b); break;
         case `/`: printf("\nЧастное чисел a и b = %f", a/b); break;
         case `*`: printf("\nПроизведение чисел a и b = %f", a*b "); break;
         default: printf("\nВы непра вильно задали действие ");
                  printf("\nПовторите ввод.");
                  goto ab;
      }
}
```

Контрольные вопросы

1. Что такое алгоритм с ветвлением?

2. Как записывается условный оператор (оператор ветвления) в Си?

3. Что такое полная и сокращенная записи условного оператора?
4. Что используется в качестве условий в операторе ветвления?
5. Какие знаки отношений можно использовать при составлении условий?
6. Что такое составное условие?
7. Каковы правила записи составных условий?
8. Какие вы знаете логические операции?
9. Какие действуют правила старшинства логических операций?
10. Что такое составной оператор? Какую структуру он имеет?

5.4. Программирование циклических вычислительных процессов

При усложнении решаемых задач ход выполнения программ становится более запутанным. Чтобы иметь возможность управлять процессом выполнения программ, его организацией используются такие структуры, как циклы.

Циклом называется блок кода, который для решения задачи требуется повторить несколько раз. В языке С известно три вида оператора цикла: for, while и do-while.

Цикл for – параметрический цикл (цикл с фиксированным числом повторений).

Основная форма цикла for имеет следующий вид:

```
for (инициализация; проверка условия окончания; изменение )
{
блок операторов;
}
```

инициализация – это присвоение начального значения параметру цикла и счетчику;

проверка условия – условное выражение, которое определяет, когда цикл должен быть завершен;

изменение – это приращение параметра цикла каждый раз при повторении цикла.

Особенности работы цикла:

- выполнение цикла происходит до тех пор, пока условное выражение истинно;
- если условие становится ложным, начинает выполняться следующий за циклом for блок операторов;

- инициализация параметра осуществляется только один раз – когда цикл for начинает работать;
- проверка условия окончания осуществляется перед каждым возможным выполнением тела цикла;
- коррекция параметра осуществляется в конце каждого выполнения тела цикла;
- параметр может как увеличиваться, так и уменьшаться;
- в циклах for блок операторов может не выполниться ни разу.

Пример использования цикла **for** «Вычисление суммы чисел от 1 до 10»:

```
# include<stdio. h>
void main(void)
{
  int i; // счетчик цикла;
  int sum = 0; // сумма чисел от 1 до 10;
    for (i = 1; i <= 10; i++) // начальное значение 1, конечное 10 и шаг цикла 1
  { sum = sum + i;
  }
printf("\Сумма чисел от 1 до 10 = %d",sum);
   }
```

Цикл while – цикл с предусловием

Основная форма оператора while имеет следующий вид:

```
while (выражение)
{
блок операций;
}
```

Особенности работы цикла

- Если выражение истинно (не равно нулю), то выполняется блок операций, заключенный в фигурные скобки, затем выражение проверяется снова.
- Последовательность действий, состоящая из проверки и выполнения блока операций, повторяется до тех пор, пока выражение не станет ложным (равным нулю).
- Выражение является условием продолжения работы цикла.
- После выхода из цикла производится выполнение блока операций, стоящего после оператора цикла.
- В цикле while блок операторов может не выполниться ни разу.

Пример использования цикла **while** «Вычисление суммы чисел от 1 до 10».

```
# include<stdio. h>
void main(void)
{
   int i=0; // счетчик цикла;
   int sum = 0; // сумма чисел от 1 до 10;
      while (i < 10) // цикл выполняется, если выполняется условие
      { i++;
      sum = sum+i;
      }
printf("\Сумма чисел от 1 до 10 = %d",sum);
}
```

Цикл do-while – цикл с постусловием.

В отличие от предыдущих циклов, в цикле do-while условие проверяется в конце оператора цикла. Основная форма оператора do-while следующая:

```
do {
блок операций;
} while (выражение);
```

Особенности работы цикла

- Фигурные скобки в данной конструкции необязательны, если внутри них находится один оператор.
- Какое бы условие ни стояло в конце оператора, набор операторов в фигурных скобках один раз выполнится обязательно. В циклах for и while оператор может не выполниться ни разу.
- Использовать цикл do...while лучше в тех случаях, когда должна быть выполнена хотя бы одна итерация либо когда инициализация объектов, участвующих в проверке условия, происходит внутри тела цикла.

Пример использования цикла **do-while** «Вычисление суммы чисел от 1 до 10»:

```
# include<stdio. h>
void main(void)
{
   int i=0; // счетчик цикла;
   int sum = 0; // сумма чисел от 1 до 10;
      do
      {
```

```
        i++;
        sum = sum+i;
        }
        while (i < 10); // цикл продолжается пока выполняется условие
    printf("\Сумма чисел от 1 до 10 = %d",sum);
    }
```

Контрольные вопросы

1. Что называют циклом в языке программирования Си?
2. Из чего состоит каждый цикл в языке программирования Си?
3. Какие основные типы циклических структур существуют в языке программирования Си?
4. Цикл с заданным числом повторений: форма записи и правила использования.
5. Цикл с предусловием: форма записи и правила использования.
6. Цикл с постусловием: форма записи и правила использования.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

Основная литература

1. *Гаврилов М.В.* Информатика и информационные технологии: учебник для бакалавров / М.В. Гаврилов, В.А. Климов. – Москва, 2012. – 349.

2. *Голицына О.Л.* Информационные технологии: учебник / О.Л. Голицына, Н.В. Максимов, Т.Л. Партыка, И.И. Попов. – М.: Форум: ИНФРА-М, 2013. – 608 с.

3. *Елович И.В.* Информатика: учебник для вузов / И.В. Елович, И.В. Кулибаба. – М., 2011.

4. Информатика [Электронный ресурс]: учебник / О.К. Альсова и др. – Новосибирск, 2012. – Режим доступа: http://elibrary.nstu.ru/source?bib_id =vtls000175426.

5. Информатика: учебник / Б.В. Соболь, А.Б. Галин, Ю.В. Панов, Е.В. Рашидова, Н.Н. Садовой. – Ростов-н/Д: Феникс, 2005. – 448 с.

6. Информатика: учеб. пособие / А.Н. Степанов. – СПб.: Питер Пресс, 2007. – 764 с.

7. *Иопа Н.И.* Информатика: (для технических специальностей): учеб. пособие / Н.И. Иопа. – Москва: КноРус, 2011. – 469 с.

8. *Максимов Н.В.* Современные информационные технологии: учеб. пособие / Н.В. Максимов, Т.Л. Партыка, И.И. Попов. – М.: Форум, 2013. – 512 с.

9. Основы информатики: учебник / В.Ф. Ляхович, С.О. Крамаров, И.П. Шамараков. – Ростов-н/Д: Феникс, 2010. – 715 с.

10. *Советов Б.Я.* Информационные технологии: учебник для бакалавров / Б.Я. Советов, В.В. Цехановский. – М.: Юрайт, 2013. – 263 с.

11. *Степанов А.Н.* Информатика: учебник для вузов. – 4-е изд. – СПб.: Питер, 2006. – 684 с.

Дополнительная литература

1. *Акулов О.А.* Информатика. Базовый курс: учебник [для студентов вузов, бакалавров, магистров, обучающихся по направлению «Информатика и вычислительная техника»] / О.А. Акулов, Н.В. Медведев. – М., 2008. – 574 с.

2. *Безручко В.Т.* Информатика. Курс лекций: учеб. пособие. – М.: Форум: ИНФРА-М, 2013. – 432 с.

3. *Велихов А.С.* Основы информатики и компьютерной техники: учеб. пособие / А.С. Велихов. – М.: СОЛОН-Пресс, 2007. – 539 с.

4. *Гришин В.Н.* Информационные технологии в профессиональной деятельности: учебник / В.Н. Гришин, Е.Е. Панфилова. – М.: Форум: ИНФРА-М, 2013. – 416 с.

5. *Дарков А.В.* Информационные технологии: теоретические основы: учеб. пособие / А.В. Дарков, Н.Н. Шапошников. – СПб.: Лань, 2016. – 448 с.

6. *Исаев Г.Н.* Информационные технологии: учебное пособие / Г.Н. Исаев. – М.: Омега-Л, 2013. – 464 с.

7. *Дорогов Б.В., Дорогова Е.Г.* Основы программирования на языке С: учеб. пособие. – М.: Форум: ИНФРА-М, 2012. – 400 с.

8. Информатика. Базовый курс: учебное пособие для высших технических учебных заведений / [С.В. Симонович и др.]. – СПБ.: Питер, 2011. – 639 с.

9. Информатика: учеб. пособие для студ. пед. вузов / А.В. Могилёв, Н.И. Пак, Е.К. Хеннер; под ред. Е.К. Хеннера. – 5-е изд., стер. – М.: Академия, 2007. – 848 с.

10. *Коноплева И.А.* Информационные технологии / И.А. Коноплева, О.А. Хохлова, А.В. Денисов. – М.: Проспект, 2015. – 328 с.

11. *Крайзмер Л.П.* Информатика и вычислительная техника / Л.П. Крайзмер. – М.: Лениздат, 2009. – 270 с.

12. *Онков Л.С., Титов В.М.* Компьютерные технологии в науке и образовании: учеб. пособие. – М.: Форум: ИНФРА-М, 2012. – 224 с.

13. Практикум по информатике: учеб. пособие для студ. высш. учеб. заведений / А.В. Могилёв, Н.И. Пак, Е.К. Хеннер; под ред. Е.К. Хеннера. – 4-е изд., стер. – М.: Академия, 2008. – 608 с.

14. *Рубальская О.Н.* Информатика Windows, Word, Excel. Самоучитель на CD: учеб. пособие. – М.: Форум: ИНФРА-М, 2012. – 224 с.

15. *Румянцева Е.Л.* Информационные технологии: учеб. пособие / Е.Л. Румянцева, В.В. Слюсарь; под ред. Л.Г. Гагарина. – М.: Форум: ИНФРА-М, 2013. – 256 с.

16. *Сырецкий Г.А.* Информатика. Фундаментальный курс. Том I. Основы информационной и вычислительной техники / Г.А. Сырецкий. – М.: БХВ-Петер, 2012. – 832 с.

17. *Сырецкий Г.А.* Информатика. Фундаментальный курс. Том II. Информационные технологии и системы / Г.А. Сырецкий. – СПб.: BHV, 2012. – 848 с.

18. *Федотова Е.Л.* Информационные технологии и системы: учеб. пособие. – М.: Форум: ИНФРА-М, 2013. – 352 с.

19. *Хлебников А.А.* Информационные технологии: учебник / А.А. Хлебников. – М.: КноРус, 2014. – 472 с.

20. *Яшкин В.Н.* Информатика аппаратные средства персонального компьютера: учеб. пособие. – М.: Форум: ИНФРА-М, 2011. – 254 с.

Интернет-ресурсы

1. ЭБС НГТУ: http://elibrary.nstu.ru/
2. ЭБС «Издательство Лань»: https://e.lanbook.com/
3. ЭБС IPRbooks: http://www.iprbookshop.ru/
4. ЭБС "Znanium.com": http://znanium.com/

Методическое обеспечение

Асташова Т.А. Информатика и информационные технологии [Электронный ресурс]: электронный учебно-методический комплекс [для технических направлений] / Т.А. Асташова; Новосиб. гос. техн. ун-т. – Новосибирск, [2016]. – Режим доступа: http://elibrary.nstu.ru/source?bib_id=vtls000230362. – Загл. с экрана.

ОГЛАВЛЕНИЕ

Асташова Татьяна Александровна

ИНФОРМАТИКА

Учебное пособие

Редактор *Л.Н. Ветчакова*
Выпускающий редактор *И.П. Брованова*
Корректор *Л.Н. Киншт*
Дизайн обложки *А.В. Ладыжская*
Компьютерная верстка *Н.В. Гаврилова*

Налоговая льгота – Общероссийский классификатор продукции
Издание соответствует коду 95 3000 ОК 005-93 (ОКП)

Подписано в печать 07.12.2017. Формат 60 × 84 1/16. Бумага офсетная
Тираж 100 экз. Уч.-изд. л. 6,27. Печ. л. 6,75. Изд. 242. Заказ № 39
Цена договорная

Отпечатано в типографии
Новосибирского государственного технического университета
630073, г. Новосибирск, пр. К. Маркса, 20