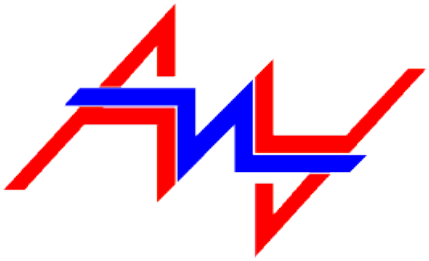
Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования
КАЗАНСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ им. А.Н.ТУПОЛЕВА – КАИ (КНИТУ–КАИ)



А.И.Маликов, Б.А.Старостин

ПРАКТИКУМ ПО ИНФОРМАТИКЕ И ИНФОРМАЦИОННЫМ ТЕХНОЛОГИЯМ. I. ОБРАБОТКА ДАННЫХ НА ПК.

Казань 2014

УДК 681.31

Практикум по информатике и информационным технологиям. I. Обработка данных на ПК. Для технических специальностей и направлений университета / Казань: Изд-во Казан. гос. техн. ун-та. 2014. 119 с.

Лабораторный практикум по информатике и информационным технологиям состоит из двух частей. Часть 1 практикума включает 9 лабораторных работ, охватывающих вопросы организации, представления, хранения и обработки информации на ПК в среде MS Windows, редактирования текстов и начала программирования в среде C++Builder. В описании каждой лабораторной работы рассмотрены примеры, приводятся варианты заданий и контрольные вопросы. В приложении даны требования к оформлению отчетов по лабораторным работам, варианты контрольных заданий и тексты программ на языке С для рассмотренных примеров. Способствует углублению знаний и выработке навыков по составлению алгоритмов, программ и использования ПК и программного обеспечения для решения инженерных задач. Практикум ориентирован на использование персональных компьютеров, программно совместимых с компьютерами фирмы IBM. Предназначен для студентов технических специальностей и направлений университета очной и заочной форм обучения.

Табл.: 5. Ил.: 23. Библиогр.: 9 назв.

ОГЛАВЛЕНИЕ

ТЕХНИЧЕСКИЕ СРЕДСТВА ОБРАБОТКИ ДАННЫХ 6
Практическая работа №1 6
Определение характеристик ПК и его основных устройств 6
1.1. Общие сведения 6
1.2. Задание 8
1.3. Метод решения 9
1.4. Порядок выполнения работы 9
1.5. Контрольные вопросы 9
ПРОГРАММНЫЕ СРЕДСТВА 10
Практическая работа №2 10
Настройка операционной системы WINDOWS XP 10
2.1. Настройка средств ввода-вывода данных 10
2.2. Настройка элементов оформления Windows XP 11
2.3. Настройка элементов управления Windows XP 15
2.4. Настройка средств автоматизации Windows XP 18
2.5. Настройка шрифтов 23
2.6. Прочие настройки Windows XP 26
2.7. Справочная система Windows XP 28
ОРГАНИЗАЦИЯ ХРАНЕНИЯ ДАННЫХ НА ПК 31
Практическая работа №3 31
Файловая структура 31
3.1. Структура хранения данных. Файловая система 31
3.2. Операции с файловой структурой 32
3.3. Проводник 33
3.4. Задание 35
3.5. Порядок выполнения работы 36
2.6. Контрольные вопросы 36
Практическая работа №4 37
Архивирование данных 37
4.1. Общие сведения об архивировании данных 37
4.2. Архивирование данных с помощью программы WinRAR 39
4.3. Задание 40
4.4. Порядок выполнения работы 41
4.5. Контрольные вопросы 41
ОБРАБОТКА ТЕКСТОВОЙ ИНФОРМАЦИИ НА ПК 42
Практическая работа №5 42
Разработка текстовых документов в процессоре Microsoft Word 42
5.1. Общие сведения о процессоре Microsoft Word 42
5.2. Приемы работы с текстами в процессоре Microsoft Word 43
5.2.1. Создание документа 44
5.2.2. Ввод текста 44
5.2.3. Форматирование текста 46
5.3. Сохранение документа 48
5.4. Приемы и средства автоматизации разработки документов 48
5.4.1. Работа со стилями 48
5.4.2. Шаблоны 49
5.5. Внедрение объектов, созданных другими приложениями 50
5.6. Задание 50

5.7. Порядок выполнения работы 50
5.8. Контрольные вопросы 51
ОБРАБОТКА ГРАФИЧЕСКОЙ ИНФОРМАЦИИ НА ПК 52
Практическая работа № 6 52
Создание иллюстраций в редакторах Paint и Microsoft Word 52
6.1. Общие сведения 52
6.2. Графический редактор Paint 52
6.2.1. Окно программы Paint 52
6.2.2. Задание размера рабочей области 53
6.2.3. Основные чертежно-графические инструменты 54
6.2.4. Трансформация изображений 56
6.2.5. Ввод текста 57
6.3. Встроенный графический редактор Microsoft Word 57
6.4. Порядок выполнения работы 61
6.5. Контрольные вопросы 61
6.6. Варианты заданий 62
РЕШЕНИЕ ВЫЧИСЛИТЕЛЬНЫХ ЗАДАЧ В C++Builder 64
Лабораторная работа №7 64
Нахождение корней квадратного уравнения 64
7.1. Общие сведения по C++ 64
7.2. Знакомство со средой разработки C++ Builder 66
7.2.1. Создание первой программы 67
7.2.2. Отладка программы: 68
7.3. Условный оператор if 69
7.4. Описание операторов и функций языка C ++ 70
7.5. Нахождение корней квадратного уравнения 71
7.5.1. Постановка задачи 71
7.5.2. Метод и алгоритм решения 71
7.5.3. Блок схема алгоритма 71
7.6. Реализация алгоритма на языке C++ 72
7.7. Порядок выполнения работы 73
7.8. Варианты заданий 73
Лабораторная работа №8 75
Построение таблицы значений функции. Организация циклов в C++ 75
8.1. Организация циклов в C++ 75
8.1.1. Оператор цикла while 75
8.1.2. Оператор цикла do-while 76
8.1.3. Оператор цикла for 76
8.1.4. Оператор break 77
8.1.5. Оператор continue 77
8.2. Построение таблицы значений функции 78
8.2.1. Постановка задачи 78
8.2.2. Алгоритм 78
8.2.3. Блок схема алгоритма 79
8.2.4. Реализация алгоритма на языке C++ 79
8.3. Варианты заданий 81
8.4. Порядок выполнения работы 81
8.5. Контрольные вопросы 82
Лабораторная работа №9 83

Накапливание результата. Итерационные алгоритмы вычисления приближенного значения функций 83
9.1. Накапливание результата 83
9.2. Итерационные алгоритмы 83
9.2.1. Постановка задачи 85
9.2.2. Метод решения 85
9.2.3. Алгоритм 85
9.2.4. Блок-схема алгоритма 86
9.2.5. Пример программы 86
9.3. Варианты заданий 87
9.4. Порядок выполнения работы 88
9.5. Контрольные вопросы 88
Лабораторная работа №10 89
Указатели, функции и одномерные массивы в C++. Задачи поиска и сортировки 89
10.1. Указатели 89
10.2. Функции 90
10.2.1. Параметры функции 91
10.2.2. Передача параметров по значению 92
10.2.3. Передача параметров по ссылке 93
10.4. Методика составления программ поиска и сортировки 94
10.4.1. Задача поиска 94
10.4.2. Задача сортировки и упорядочения массива 95
10.5. Варианты заданий 96
10.6. Порядок выполнения работы 97
Лабораторная работа №11 98
Обработка двумерных массивов 98
11.1. Обработка двумерных массивов 98
11.1.1. Двухмерные массивы в C++ 98
11.1.2. Использование двумерных массивов в качестве параметров функций 100
11.1.3. Примеры работы с двумерными массивами 100
11.1.4. Многомерные массивы 102
11.2. Варианты заданий 102
11.3. Порядок выполнения работы 103
СПИСОК ЛИТЕРАТУРЫ 104
Приложение 1. Варианты контрольных заданий 105
Приложение 2. Требования к оформлению отчета по лабораторным работам 109
1. Объем и содержание отчета 109
2. Оформление отчета 111
Приложение 3. Титульный лист к отчету 113
Приложение 4. Блок-схемы алгоритмов к лабораторной работе № 6 114
Приложение 5. Текст программы на языке С для вычисления корней квадратного уравнения 118
Приложение 6. Текст программы на языке С для вычисления приближенного значения экспоненциальной функции 118

ТЕХНИЧЕСКИЕ СРЕДСТВА ОБРАБОТКИ ДАННЫХ

*Практическая работа №*1

Определение характеристик ПК и его основных устройств

Цель занятия: знакомство с основными устройствами ПК и приобретение навыков в определении конфигурации компьютера средствами ОС Windows.

1.1. Общие сведения

В основу построения подавляющего большинства ЭВМ положены принципы, сформулированные в 1945 году американским ученым венгерского происхождения Джоном фон Нейманом. Прежде всего, компьютер должен иметь следующие устройства:

- *Арифметическо-логическое устройство*, выполняющие арифметические и логические операции;
- *Устройство управления*, которое организует процесс выполнения программ
- *Запоминающее устройство*, или память для хранения программ и данных;
- *Внешние устройства* для ввода-вывода информации.

В основе работы компьютера лежат следующие принципы:

- *Принцип двоичного кодирования*. Согласно этому принципу вся информация, поступающая в ЭВМ, кодируется с помощью двоичных сигналов.
- *Принцип программного управления*. Из него следует, что программа состоит из набора команд, которые выполняются процессором автоматически друг за другом в определенной последовательности.
- *Принцип однородности памяти*. Программы и данные хранятся в одной и той же памяти. Поэтому ЭВМ не различает, что хранится в данной ячейке памяти - число, текст или команда. Над командами можно выполнять такие же действия, как и над данными.
- *Принцип адресности*. Основная память структурно состоит из пронумерованных ячеек; процессору в произвольный момент времени доступна любая ячейка.

Структура современного компьютера представлена на рис.1.1.

Системный блок является наиболее “весомой” частью любого компьютера. Внутри него расположены блок питания, плата с центральным процессором (ЦП), видеоадаптер, жесткий диск, дисковод гибких дисков и другие устройства ввода вывода информации. Зачастую видеоадаптер и контроллеры ввода вывода размещены прямо на плате ЦП. В системном блоке могут размещаться средства мультимедиа: звуковая плата и устройство чтения оптических дисков – CD, DVD-ROM. Кроме того, в понятие “компьютер” входит клавиатура и монитор. Манипулятор мышь является необяза-

тельной, но весьма важной деталью. Теперь коротко о выборе основных компонентов ПК

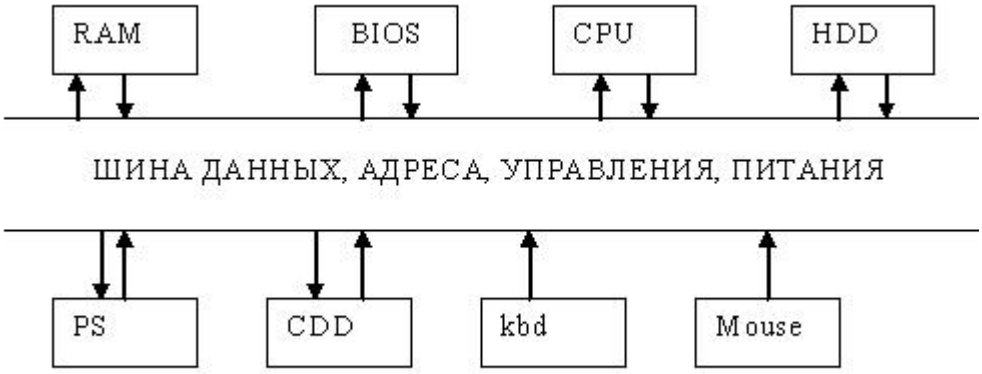


Рис.1.1. Структура современного персонального компьютера. RAM – оперативная память, BIOS – базовая система ввода вывода, CPU – процессор, HDD – накопитель на жестком магнитном диске, PS – блок питания, CDD – cd-rom, Kbd – клавиатура, Mouse – манипулятор мышь

Процессор является основным компонентом любого ПК. Наиболее распространены процессоры фирмы Intel, хотя ЦП других фирм (AMD, Cyrix) составляют им достойную конкуренцию. В настоящее время выпускаются процессоры серии Pentium, Celeron (фирмы Intel), Athlon и Duron (фирмы AMD). В то же время в России имеется достаточно большой парк машин на основе предшествующих серий, которые уже сняты с производства. Какую же информацию несёт в себе маркировка процессора? Рассмотрим, например, вариант Pentium4 2400. Число 2400 характеризует рабочую частоту процессора.

Системная плата. Основной характеристикой системных плат является их архитектура. Основными шинами до недавнего времени считались ISA (Industrial Standard Architecture) и EISA (Extended ISA), работающие на частоте 8 МГц и имеющие разрядность 16 и 32 соответственно. Для обеспечения нормальной работы видеоадаптеров был разработан стандарт AGP. С появлением процессора Pentium была разработана самостоятельная шина PCI, которая на сегодняшний день является наиболее быстрой. USB – Universal Serial Bus - универсальная последовательная магистраль. Этот стандарт позволяет подсоединить до 256 устройств (по принципу общей шины) имеющих последовательный интерфейс. К USB подключаются различные устройства, такие как мышь, клавиатура, цифровые фото- и Web-камеры, модемы, джойстики и т.д. Эта шине практически не конфликтует с другим оборудованием, установленным на компьютере и позволяет подключать устройства не выключая компьютер

Жёсткий диск (винчестер). Начав своё шествие с объема в 5 МБ, достиг небывалых высот. На сегодняшний день не удивят диски объёмом 1 ТБ. Следует придать значение не только емкости диска, но и его временным ха-

рактеристикам. В качестве оптимальных можно порекомендовать винчестеры фирмы Western Digital, Seagate.

Оперативная память (RAM, ОЗУ). Здесь закон простой: чем больше, тем лучше. Для нормальной работы большинства программных продуктов желательно иметь хотя бы 1 ГБ памяти. При увеличении ОЗУ быстродействие ПК увеличивается. В настоящее время лучше иметь 4 ГБ или 8 ГБ памяти.

Клавиатура. Стандартной в России является 101 - клавишная клавиатуры с английскими и русскими символами.

Мышь. Необходима для работы с графическими пакетами, чертежами, при разработке схем и при работе в среде Windows. Следует отметить, что некоторое игровое и программное обеспечение требует наличие мыши. Основной характеристикой мыши является разрешающая способность, измеряемая в точках на дюйм (dpi). Нормальной считается мышь, обеспечивающая разрешение 300-400 dpi. Неплохо иметь также специальный коврик под мышь, что обеспечивает её сохранность и долговечность.

Монитор. Выбор этой части ПК следует уделить особое внимание, поскольку от качества монитора зависит сохранность вашего зрения и общая утомляемость при работе. Мониторы имеют стандартный размер диагонали в 15,17,19,20 и 21 дюйм. Необходимый размер диагонали монитора выбирается исходя из разрешения, при котором вы собираетесь работать. Так, для большинства приложений вполне достаточно иметь 15 дюймовый монитор, который обеспечивает работу при разрешениях до 800 на 600 точек и более.

Звуковая карта. С одной стороны, звуковая карта не является необходимым элементом компьютера, но, с другой стороны, позволяет превратить его в мощное подспорье при обучении и написании музыки, изучении языков, в компьютерных играх.

CD, DVD-ROM. Бывают как для чтения (R), так и для записи (RW) информации на компакт диски. Они находят все большее распространение в связи с тенденцией поставлять профессиональное, обучающее и игровое программное обеспечение на оптических дисках объемом 700 МБ и более 4 ГБ.

1.2. Задание

Требуется определить следующие параметры для конкретного персонального компьютера:

1) Название видео карты, объем видео памяти, текущую, максимальную, минимальную частоту разверстки, поддерживаемое количество цветов, текущее разрешение экрана.
2) Определить марку и частоту процессора, объем ОЗУ.
3) Определить количество, марку и объем жестких дисков.
4) Определить количество, размер, файловую систему логических дисков.
5) Определить количество и марку накопителей CD-ROM.
6) Определить наличие FDD.

7) Определить наличие и марку сетевых карт.

1.3. Метод решения

Для определение конфигурации компьютера воспользуемся средствами ОС Windows. Параметры экрана находятся по следующему пути Пуск\настройки\панель управления\экран. Параметры оборудования можно посмотреть по Пуск\настройки\панель управления\система и далее вкладка устройства. Параметры HDD, флэш-карт памяти можно посмотреть, зайдя в их свойства (нажав правую кнопку мыши на иконке диска) в проводнике или другом файловом менеджере.

1.4. Порядок выполнения работы

1) Ознакомиться с п.1. описания лабораторной работы.
2) Согласно пункту 2 определить параметры компьютера, используя рекомендации из пункта 3.
3) Составить отчёт о проделанной работе в письменном виде в соответствии с требованиями, представленными в приложении 2.
4) При сдаче лабораторной работы студент должен показать и объяснить результаты выполнения задания, ответить на контрольные вопросы.

1.5. Контрольные вопросы

1) Как определить размер HDD?
2) Как определить количество HDD и CDD?
3) Как определить файловую систему логических дисков?
4) Как определить размер свободного места на HDD?
5) Как определить наличие сетевых карт?
6) Как определить марку сетевых карт?
7) Как определить марку CDD?
8) Как определить марку HDD?
9) Как определить объем ОЗУ?
10) Как определить марку процессора?
11) Как определить частоту процессора?
12) Как определить название видеокарты?
13) Как определить объем видеопамяти?
14) Как определить текущую, максимальную, минимальную частоту развертки монитора?
15) Как определить поддерживаемое количество цветов видео картой?
16) Как определить текущее разрешение экрана?

ПРОГРАММНЫЕ СРЕДСТВА

Практическая работа №2

Настройка операционной системы WINDOWS XP

2.1. Настройка средств ввода-вывода данных

Настройка клавиатуры. Настройку клавиатуры выполняют в диалоговом окне Свойства: Клавиатура, которое открывают двойным щелчком на значке Клавиатура в окне Панель управления. На вкладке Скорость представлены средства настройки параметров функции *автоповтора символов* (величина задержки перед началом повтора символов и темп повтора), а также средства управления частотой мерцания курсора.

Настройка мыши. Настройку мыши выполняют в диалоговом окне Свойства: Мышь, которое открывают с помощью значка Мышь в окне Панель управления. На вкладке Кнопки мыши представлены средства назначения левой или правой кнопке функций *основной кнопки,* а также средства настройки интервала времени между щелчками, при котором два отдельных щелчка интерпретируются как один двойной.

На вкладке Указатели представлены средства для выбора схемы указателей мыши. Схема указателей представляет собой именованную совокупность настроек формы указателей мыши, сохраняемую в отдельном файле.

На вкладке Параметры указателя представлены средства для управления *чувствительностью* мыши. Чувствительность мыши определяется величиной экранного перемещения указателя при заданном перемещении прибора. Выбор чувствительности зависит от типа мыши или другого манипулятора, а также от привычного режима работы конкретного пользователя (от характерного размаха движений мыши в процессе управления).

На этой же вкладке имеются средства управления видимостью указателя. Есть возможность скрывать указатель во время работы с клавиатурой, а также задействовать средства подсветки указателя при работе с малоконтрастными дисплеями, например некоторыми жидкокристаллическими дисплеями портативных компьютеров.

Настройка стиля управления операционной системой. Начиная с *Windows* 98 операционные системы семейства *Windows* поддерживают несколько *стилей управления.* До сих пор мы рассматривали только так называемый *классический стиль управления,* восходящий к принципам *Windows* 95. Он характерен тем, что объекты выделяют одним щелчком, а открывают двумя щелчками.

Другой стиль управления характерен для работы в Интернете. Он подразумевает, что объекты выделяют простым наведением указателя, а открывают одним щелчком. Данный стиль позволяет несколько повысить произво-

дительность в ряде операций с объектами, но не очень удобен при проведении групповых операций.

Выбор того или иного стиля управления выполняют включением переключателя на вкладке Общие диалогового окна Свойства папки (Пуск > Настройка > Панель управления > Свойства папки).

Для использования классического стиля надо установить переключатель Открывать двойным, а выделять одним щелчком. Для использования стиля, характерного для Интернета, установите переключатель Открывать одним щелчком, выделять указателем. В этом случае подписи значков выделяются подчеркиванием. Это может происходить всегда или, для более привычного вида Рабочего стола и окон, только при наведении указателя. Соответствующая настройка также выбирается установкой переключателя.

2.2. Настройка элементов оформления Windows XP

Настройка фона Рабочего стола. Операционная система *Windows XP* позволяет использовать в качестве фона Рабочего стола заливку сплошным цветом, фоновый рисунок или же документ или иллюстрацию в формате, принятом в Интернете. Выбор метода оформления осуществляют на вкладке Рабочий стол диалогового окна Свойства: Экран, которое открывают с помощью значка Экран в окне Панель управления или посредством пункта Свойства контекстного меню Рабочего стола

Выбор одноцветного фона Рабочего стола осуществляется в раскрывающейся палитре Цвет. При выборе рисунка, используемого в качестве фона, предполагается, что он находится в системной папке \Windows. Если это не так, отыскать подходящий рисунок можно с помощью командной кнопки Обзор. При выборе фонового рисунка предоставляются средства для выбора способа его расположения (По центру экрана или на полном экране). В последнем случае возможен выбор варианта Растянуть (с перемасштабированием изображения в соответствии с размером Рабочего стола) или варианта Замостить (без перемасштабирования, но с размножением копий рисунка по всему полю Рабочего стола).

Выбор в качестве фона документа *HTML* (формат страниц Интернета) применяют в тех случаях, когда на Рабочем столе надо разместить текстовую информацию, а также в тех случаях, когда необходимо обеспечить динамическое изменение фонового изображения под управлением внешней программы или удаленного Web-сервера.

Настройка экранной заставки. *Экранные заставки* – это динамические изображения, воспроизведение которых включается автоматически при отсутствии в течение заданного времени событий, вызванных пользователем. Первоначальное назначение заставок состояло в том, чтобы снизить угрозу «выгорания люминофора» на тех участках экрана, которые подвержены особо длительному стационарному воздействию электронного луча. Результатом этого эффекта было образование в местах длительного воздействия луча бу-

рых пятен. Современным мониторам эффект «выгорания люминофора» не грозит, но экранные заставки продолжают использовать как средство сокрытия экранной информации от посторонних наблюдателей в период отсутствия владельца компьютера на рабочем месте.

Выбор и настройку режима действия экранной заставки осуществляют на вкладке Заставка диалогового окна Свойства: Экран. Представленные здесь средства настройки позволяют задать интервал времени, по прошествии которого при отсутствии событий, вызванных пользователем, происходит автоматический запуск заставки, а также настроить параметры заставки. Если при начале сеанса текущего пользователя требовался пароль, то его необходимо ввести и для того, чтобы отключить заставку и вернуться к текущей работе. В предыдущих версиях *Windows* специальный пароль для отключения заставки можно было ввести прямо здесь. На этой же вкладке имеются средства для управления энергосберегающими функциями монитора (кнопка Питание).

Настройка оформления элементов управления Windows. Концепция оформления внешнего вида элементов управления *Windows XP* претерпела существенные изменения по сравнению с предыдущими версиями *Windows.* Совокупность всех визуальных и звуковых настроек интерфейса *Windows* рассматривается как *тема Рабочего стола.* Тема включает набор реквизитных значков Рабочего стола, шрифты и цвета, используемые для элементов оформления, указатели мыши, звуки, экранная заставка.

Выбрать одну из заранее определенных тем можно на вкладке Темы диалогового окна Свойства: Экран. При изменении в ходе настройки любого элемента предварительно определенной темы операционная система рассматривает возникшую совокупность настроек как особую тему оформления.

Стиль оформления на основе заданной темы – это особый стиль оформления *Windows XP.* На вкладке Оформление диалогового окна Свойства: Экран такой стиль задается выбором пункта Стиль Windows XP в раскрывающемся списке Окна и кнопки.

Пункт Классический стиль использует приемы оформления, типичные для предыдущих версий *Windows.*

Совокупность настроек, описывающих только графические характеристики окон и значков *Windows,* называется *цветовой схемой.* Такая схема может быть сохранена и загружена впоследствии. Средства настройки оформления позволяют загружать готовые цветовые схемы, создавать на их основе новые схемы путем редактирования и сохранять их под заданными именами.

Для редактирования текущих цветовых и шрифтовых настроек надо щелкнуть на кнопке Дополнительно. В диалоговом окне Дополнительное оформление возможно изменение каждого из двух десятков элементов оформления по используемому шрифту и цвету. Для некоторых элементов оформления *Windows XP* позволяет использовать многоцветное оформление

путем создания градиентных растяжек (плавных переходов) между двумя заданными краевыми цветами. Выбор цвета осуществляют в раскрывающейся палитре с фиксированным количеством цветов. Любой цвет палитры можно определить самостоятельно – доступ к цветовой матрице открывает командная кнопка Другой в палитре цветов.

Дополнительные средства оформления Рабочего стола. Ряд дополнительных средств оформления Рабочего стола доступен через дополнительные диалоговые окна. Если щелкнуть на кнопке Настройка рабочего стола на вкладке Рабочий стол диалогового окна Свойства: Экран, откроется диалоговое окно Элементы рабочего стола. Здесь можно управлять отображением и внешним видом реквизитных значков Рабочего стола.

Если щелкнуть на кнопке Эффекты на вкладке Оформление диалогового окна Свойства: Экран, откроется диалоговое окно Эффекты. Действие визуальных эффектов, представленных здесь, хорошо прокомментировано названиями соответствующих элементов управления и легко проверяется практическими экспериментами.

Средства оформления активного Рабочего стола. В режиме активного Рабочего стола оформление рассматривается как аналог Полстраницы. На Рабочем столе возможно размещение нескольких *активных объектов.* Активными считаются объекты, которые могут динамически изменяться под управлением внешней программы или удаленного сервера. Таким образом, активный Рабочий стол может выполнять функции динамического отображения поставляемого содержимого.

Для добавления на Рабочий стол активных элементов надо щелкнуть на кнопке Настройка рабочего стола на вкладке Рабочий стол диалогового окна Свойства: Экран и в открывшемся диалоговом окне Элементы рабочего стола выбрать вкладку Веб. Чтобы добавить новый активный элемент, надо щелкнуть на кнопке Создать и далее следовать указаниям мастера.

В качестве активных элементов Можно использовать как локальные файлы, так и документы, принимаемые из Интернета. В последнем случае для создания и обновления таких активных элементов необходимо установить подключение к Интернету.

Размещением активных компонентов Рабочего стола можно управлять непосредственно на Рабочем столе путем перетаскивания их с помощью мыши. Более сложные операции (подключение и отключение активных компонентов, назначение связи между активным компонентом и поставщиком его содержимого) выполняются с вкладки Веб диалогового окна Элементы рабочего стола.

Настройка параметров экрана. К настраиваемым параметрам экрана относятся:

- величина экранного разрешения (измеряется в точках по горизонтали и вертикали);

- величина цветового разрешения (выражается количеством одновременно отображаемых цветов или разрядностью кодирования цвета точки).

Предельные значения обоих параметров зависят от свойств видеоадаптера и монитора. Их можно задать на вкладке Параметры диалогового окна Свойства: Экран. Цветовое разрешение (*глубину цвета)* выбирают в раскрывающемся списке Качество цветопередачи, а разрешение экрана устанавливают с помощью движка Разрешение экрана. При недостаточном объеме видеопамяти, присутствующей на плате устаревшего видеоадаптера, установка повышенного разрешения экрана приводит к сокращению списка возможных значений параметра глубины цвета.

Настройка свойств видеоадаптера и монитора. Настройку свойств видеоадаптера и монитора выполняют в диалоговом окне свойств видеоподсистемы, которое открывают щелчком на кнопке Дополнительно на вкладке Параметры диалогового окна Свойства: Экран. В указанном диалоговом окне настройку свойств монитора выполняют на вкладке Монитор, а настройку свойств видеоадаптера – на вкладке Адаптер. Если и монитор, и видеоадаптер установлены с использованием оригинальных драйверов, возможна настройка частоты обновления экрана. Предельные значения этого параметра зависят от текущего экранного разрешения, и потому данную регулировку следует провести отдельно для каждого из возможных рабочих разрешений экрана. На вкладке Монитор можно выбрать оптимальную частоту для текущего режима экрана, а на вкладке Адаптер можно сразу выбрать оптимальный режим работы (комбинацию разрешения экрана, цветового разрешения и частоты обновления).

Если монитор и видеоадаптер установлены с использованием заменяющих драйверов, управление частотой регенерации экрана может быть ограничено, а в некоторых случаях даже опасно для монитора. В этом случае рекомендуется подходить к изменению частоты обновления с особой осторожностью.

Если видеоадаптер поддерживает на аппаратном уровне функции математической обработки видеоизображений (*видеоускорение),* на вкладке Диагностика можно задать степень использования аппаратного ускорения. Первоначальную настройку проводят установкой соответствующего движка в крайнее правое положение (максимальное использование аппаратных функций видеоадаптера). Если при этом наблюдаются искажения экранных объектов (прежде всего это касается пунктов меню и элементов управления полос прокрутки), то степень использования аппаратных функций последовательно понижают вплоть до полного исключения нежелательных эффектов.

Настройка звуковых схем. Операционная система *WindowsXP* является *объектно-ориентированной.* Управление подобными программными системами обычно организуется с использованием так называемого *событийного механизма.*

Все операции пользователя, которые он выполняет с экранными элементами управления, являются, с точки зрения операционной системы, *событиями пользователя.* Кроме событий пользователя существуют так называемые *системные события,* к которым относятся особые ситуации *(исключения),* возникающие в операционной системе в тех случаях, когда происходит штатное или нештатное программное событие, требующее реакции пользователя.

Оформление *Windows XP* является не только визуальным, но и звуковым, то есть системным событиям и Событиям пользователя могут быть поставлены в соответствие звуковые клипы, которые воспроизводятся при наступлении событий. Такими событиями, например, могут быть открытие или закрытие окна, удаление объекта в Корзину, поступление электронной почты на сервер, запуск *Windows XP* или завершение работы с операционной системой. Именованная совокупность настроек, связанных с назначением определенным событиям определенных звуков, называется *звуковой схемой.*

Для настройки звуковых схем используют диалоговое окно Свойства: Звуки и аудио-устройства, которое открывают с помощью значка Звуки и аудиоустройства в окне Панели управления. Элементы управления вкладки Звуки данного диалогового окна позволяют загружать имеющиеся звуковые схемы, редактировать их и сохранять. Несколько стандартных звуковых схем поставляются совместно с операционной системой. Их редактирование осуществляется путем изменения назначения звуков системным событиям. Результаты редактирования могут быть отдельно сохранены в виде новой звуковой схемы.

Назначение звуков системным событиям выполняют в списке Программные события. Те события, которым в данном списке уже поставлен в соответствие звуковой клип, отмечены значком громкоговорителя. При щелчке на значке события в поле Звуки отображается имя файла, в котором хранится соответствующий звуковой объект. При необходимости удалить звуковое оформление события, выделенного в списке, следует выбрать в раскрывающемся списке Звуки пункт (Нет). При необходимости прослушать звук, назначенный выделенному событию, следует щелкнуть на кнопке Воспроизведение звука.

2.3. Настройка элементов управления Windows XP

Настройка Панели задач. Панель задач в *Windows XP* настраиваемая – ее свойствами можно управлять. В исходном состоянии она расположена вдоль нижней кромки экрана, но методом перетаскивания ее можно расположить вдоль любой другой кромки. Соответственно, вместе с нею изменят свое положение кнопка Пуск и панель индикации.

Размер Панели задач можно настроить протягиванием мыши, если навести указатель на внешнюю рамку и дождаться, когда он сменит форму. Предельный размер Панели задач – половина экрана.

Для изменения свойств Панели задач надо щелкнуть правой кнопкой мыши где-либо на ее свободном месте и в открывшемся контекстном меню выбрать пункт Свойства. Настройка Панели задач производится на вкладке Панель задач. Наиболее важны установки двух флажков: Расположить поверх всех окон и Автоматически убирать с экрана. Установка первого флажка позволяет сделать так, чтобы окна, открытые на Рабочем столе, не могли перекрывать Панель задач. Установка второго флажка делает Панель задач скрытой и освобождает дополнительное место на Рабочем столе. Чтобы вызвать скрытую Панель задач, достаточно подвести указатель мыши к тому краю экрана, за которым она находится.

В операционной системе *Windows XP* Панель задач обладает рядом интересных особенностей. Так, например, в рамках Панели задач можно создать несколько дополнительных инструментальных панелей:

- Панель адресов Интернета;
- Панель ссылок на *Web*-страницы Интернета;
- Панель объектов Рабочего стола;
- Панель быстрого запуска.

Для создания (или удаления) этих панелей служит команда Панели инструментов, присутствующая в контекстном меню Панели задач. Особенно широко используется Панель быстрого запуска. Методом перетаскивания на ней можно разместить ярлыки наиболее часто используемых программ. Запуск программ с этой панели производится одним щелчком на значке, в то время как для запуска с Рабочего стола или из окна папки нужен двойной щелчок. Поскольку окна открытых папок и программ могут скрыть значки Рабочего стола, но не могут скрыть Панель задач, использование Панели быстрого запуска очень удобно.

Все дополнительные панели необязательно держать на Панели задач. Их можно переместить к любой из кромок экрана или разложить на Рабочем столе. Перемещение инструментальных панелей выполняют методом перетаскивания за специальный рубчик, который присутствует на панели слева. Возможность проведения подобных настроек позволяет персонализировать рабочую среду.

После того как Панель задач настроена наиболее удачно для конкретного пользователя, ее состояние можно закрепить. В этом случае изменение настроек Панели задач блокируется. Чтобы установить такую блокировку, установить флажок Закрепить панель задач в контекстном меню Панели задач или в диалоговом окне ее свойств. После сброса этого флажка свойства Панели задач можно снова изменять.

Настройка Главного меню. Главное меню – основной элемент управления в *Windows.* С его помощью можно запустить любую программу, установленную на компьютере с ведома операционной системы, открыть документы, с которыми выполнялась работа в последние дни, и выполнить боль-

шинство настроек компьютера и операционной системы. Главное меню открывается щелчком на кнопке Пуск.

Главное меню – многоуровневое: Так, например, при наведении указателя мыши на пункт Программы открывается система вложенных меню, отображающая распределение программ по разным категориям. По своим свойствам каждая категория Главного меню имеет статус папки, а каждый пункт – статус ярлыка. Таким образом, структурой Главного меню можно управлять путем управления структурой папок, представляющих его. Простейший способ открыть структуру Главного меню для редактирования – воспользоваться пунктом Проводник в контекстном меню кнопки Пуск.

Настройка свойств Корзины. Корзина представляет собой специальную папку Windows XP, в которой временно хранятся удаленные объекты. Физически Корзина на жестком диске представлена скрытой папкой \Recycled, причем для каждого жесткого диска, имеющегося в вычислительной системе, папка \Recycled – своя. Однако логически Корзина представляет собой одну-единственную папку, соответствующую всем папкам \Recycled, имеющимся в компьютерной системе.

Настройку свойств Корзины выполняют в диалоговом окне Свойства: Корзина, открываемом выбором пункта Свойства в контекстном меню. Данное диалоговое окно содержит одну вкладку для настройки глобальных свойств интегрированной Корзины и по одной вкладке на каждый жесткий диск из числа имеющихся в составе вычислительной системы. Если на вкладке Глобальные установлен переключатель Единые параметры для всех дисков, то элементы управления вкладок, соответствующих конкретным дискам, не активируются.

Основным параметром Корзины является ее предельная емкость. Когда объем файлов в Корзине начинает превосходить установленное значение, операционная система автоматически чистит Корзину, окончательно уничтожая файлы, которые были помещены туда раньше всего. Этот параметр выставляется движком и измеряется в процентах от емкости соответствующих дисков (по умолчанию – 10%). Прочие элементы управления диалогового окна свойств Корзины предусматривают возможность удаления объектов без помещения их в Корзину (используется при глобальной расчистке жесткого диска) и возможность отключения сообщения, предупреждающего об удалении объектов.

Настройка свойств окон папок. К основным настройкам свойств окон папок относится настройка режима отображения скрытых и системных объектов, а также настройка способа обзора вложенных папок.

Настройку свойств окон папок осуществляют в диалоговом окне Свойства папки. Его можно открыть из окна любой папки командой Сервис > Свойства папки или из Главного меню командой Пуск > Настройка > Панель управления > Свойства папки.

Отображение системных и скрытых объектов целесообразно включать перед удалением папок, а также при обслуживании операционной системы. При обычной работе системные и скрытые объекты лучше не отображать, чтобы не перегружать экран излишней информацией. (Если скрытые объекты не отображаются в окне папки, об их наличии можно судить по записи в строке состояния.) Элементы управления для включения и отключения отображения скрытых и системных объектов находятся на вкладке Вид диалогового окна Свойства папки в категории Файлы и папки > Скрытые файлы и папки.

Существует два способа обзора вложенных папок. В одном случае все вложенные папки открываются в одном и том же окне, а в другом для каждой очередной вложенной папки открывается новое окно. Первый способ не перегружает Рабочий стол открытыми окнами, но при этом теряется наглядность навигации в структуре окон папок. Соответственно, достоинства и недостатки второго метода противоположны. Выбор способа обзора выполняют на вкладке Общие диалогового окна Свойства папки путем установки переключателя Открывать папки в одном и том же окне или переключателя Открывать каждую папку в отдельном окне.

2.4. Настройка средств автоматизации Windows XP

Автоматический запуск приложений. Для автоматического запуска приложений после загрузки операционной системы в *Windows XP* предусмотрено очень простое средство – специальная папка \Авто-загрузка (\Главное меню\Программы\Автозагрузка). Настройка автоматического запуска приложений выполняется копированием ярлыков запускаемых приложений в эту папку. Соответственно, отключение автоматического запуска приложения выполняют удалением его ярлыка из папки \Автозагрузка.

С помощью папки \Автозагрузка можно не только запускать приложения, но и открывать документы. Соответственно, в этом случае в папку необходимо предварительно поместить ярлык документа. Открытие документа происходит с одновременным запуском *родительского* приложения, которое предназначено для работы с документами данного типа.

Настройка свойств типов файлов. Многие автоматические операции *Windows XP* основаны на том, что операционная система должна предварительно знать, какое приложение следует использовать для работы с документами того или иного типа. В частности, выше мы видели, что автоматическое открытие документа, ярлык которого находится в папке \Автозагрузка, сопровождается запуском приложения, связанного с данным типом документов. О том, какое именно приложение следует считать связанным с каждым конкретным типом файлов, операционная система судит по расширению имени файла, а сама связь выполняется путем регистрации типов файлов в операционной системе.

Для регистрации (перерегистрации) свойств типов файлов служит вкладка Типы файлов диалогового окна Свойства папки (Пуск > Настройка > Панель управления > Свойства папки). Необходимость в регистрации обычно возникает в тех случаях, когда пользователю надо ввести собственное расширение имени файла (так называемое *пользовательское расширение имени)* и назначить приложение, принятое по умолчанию для обслуживания файлов данного типа. Необходимость в перерегистрации, как правило, связана с некорректной работой некоторых приложений (после установки они могут автоматически «захватывать» себе некоторые типы файлов, не всегда спрашивая согласия пользователя на эту операцию). Разумеется,, после удаления такого приложения файлы данных типов остаются *без родительского приложения,* и для них надо провести регистрацию вручную – автоматически открыть их операционная система уже не может.

На вкладке Типы файлов диалогового окна Свойства папки приведен список Зарегистрированные типы файлов. Если в этом списке выделить один из типов файлов, в нижней части диалогового окна можно увидеть расширение имени, зарегистрированное для данного типа, и приложение, с ним связанное. Более подробную информацию о пути доступа к приложению можно получить, если открыть диалоговое окно Изменение свойств типа файлов щелчком на командной кнопке Дополнительно.

В данном диалоговом окне приведен список действий, которые возможны с файлами данного типа. Одно из действий списка выделено полужирным цветом – оно является основным.

Элементы управления диалогового окна Изменение свойств типа файлов позволяют:

- изменить значок, связанный с данным типом файлов (Сменить значок);
- создать новое действие и назначить ему приложение (Создать);
- изменить приложение, выполняющее действие (Изменить);
- удалить действие (Удалить);
- назначить избранное действие основным (По умолчанию).

Действие, назначенное основным, выполняется при двойном щелчке на значке или ярлыке. Прочие действия, представленные в списке Действия, доступны через контекстное меню. Если в данном списке создать новое действие и назначить ему приложение или удалить одно из действий, то изменится контекстное меню, открывающееся при щелчке правой кнопкой мыши на значках или ярлыках файлов данного типа. Таким образом, диалоговое окно Изменение свойств типа файлов служит не только для настройки свойств типов файлов, но и для редактирования контекстного меню документов.

Настройка команды Отправить. Команда Отправить – мощное средство повышения производительности труда при работе в *Windows XP*. Эта команда имеется в контекстном меню большинства объектов, и с ней связан список объектов, которые могут служить адресатами при пересылке текуще-

го объекта. Использование команды Отправить – простейший способ копирования документа на гибкий диск, отправки его по заданному адресу электронной почты, создания его ярлыка на Рабочем столе и т. п. Пункты меню команды Отправить – настраиваемые и редактируемые. Неиспользуемые пункты можно удалить, а вместо них создать другие, более удобные. Настройка команды Отправить выполняется путем наполнения специальной папки \SendTo ярлыками папок, устройств и каналов связи. Содержимое этой папки специфично для каждого пользователя. Каждый ярлык, присутствующий в данной папке, соответствует одному из пунктов меню команды Отправить.

Автоматизация очистки жесткого диска. Необходимость в автоматической очистке жесткого диска связана с особенностью *Windows XP,* которая заключается в том, что эта операционная система предназначена для круглосуточной работы персонального компьютера. В ночное время система может обеспечивать работу в Интернете и доставку информации от *Web-узлов,* на услуги которых оформлена подписка. Если при обычной работе с компьютером возникает исключительная ситуация, связанная с переполнением жесткого диска, пользователь имеет возможность приостановить текущий процесс, выполнить необходимые операции очистки и продолжить работу. Если такая исключительная ситуация происходит ночью, выполнять операции очистки система должна автоматически – для этого в нее входит *агентское приложение* Очистка диска (*программы-агенты* запускаются автоматически при возникновении связанных с ними исключительных событий).

Агент очистки запускается командой Пуск > Программы > Стандартные > Служебные > Очистка диска. После запуска программы следует указать имя диска, для которого выполняется настройка. Агент включается в работу автоматически, если операционная система обнаруживает на диске недостаток свободного места.

Состав папок, подлежащих очистке, задается на вкладке Очистка диска. Разумеется, далеко не все папки жесткого диска подлежат очистке в автоматическом режиме. Теоретически, папок, в которых не должны храниться невосполнимые данные, не так уж много. Состав списка Удалить следующие файлы различен для разных дисков и зависит от размещения служебных каталогов. Выбор нужных осуществляют установкой соответствующих флажков:

• Temporary Internet Files – папка, в которой кэшируются данные, принятые из Интернета при работе со службой *World Wide Web* (кэширование служит только для ускорения загрузки Weft-страниц при их повторном посещении, поэтому особой ценности данные, хранящиеся в этой папке, не представляют);

• Downloaded Program Files – папка, в которой хранятся активные объекты, содержащие программный код, принятые из Интернета (это объекты динамического оформления *Web*-страниц; их хранение служит для ускорения

загрузки *Web*-страниц при повторном посещении, хотя один и тот же стандартный программный элемент может иногда использоваться для воспроизведения объектов, встроенных в разные *Web*-страницы);

• Корзина – достойный кандидат для автоматической очистки, если пользователь не использует ее для хранения ценных данных;

• Временные файлы – имеется в виду папка \Temp, в которой не принято хранить ценные данные. Приложения нередко автоматически создают в ней свои служебные временные файлы, но не всегда могут их удалить (например, в случае аварийного завершения работы), в результате чего эта папка часто перегружается ненужными отходами.

Запуск приложений по расписанию. Одним из методов автоматизации работ, выполняемых на компьютере под управлением операционной системы *Windows XP*, является запуск приложений по назначенному расписанию. Основным средством такого подхода является программа Назначенные задания (Пуск > Программы > Стандартные > Служебные > Назначенные задания или Пуск > Настройка > Панель управления > Назначенные задания).

Окно программы. Назначенные задания можно рассматривать как окно специальной папки. Ярлыки приложений, размещенные в этой «папке», обладают *особыми атрибутами,* не характерными для обычных объектов: Расписание, Время следующего запуска, Время прошлого запуска, Состояние. В связи с этим невозможно формирование расписания автоматического запуска приложений приемом простого размещения ярлыков в папке, как мы это делали при настройке средств автоматического запуска приложений и команды Отправить. Наполнение папки Назначенные задания выполняется под управлением специальной программы – Мастера планирования заданий. Мастер запускается двойным щелчком на значке Добавить задание. В процессе его работы пользователь имеет возможность выбрать приложение и назначить расписание его запуска с указанием даты и времени первого запуска, а также периодичности последующих запусков.

Программа Назначенные задания позволяет редактировать расписания заданий. Редактирование выполняют в диалоговом окне, которое открывают командой Свойства в контекстном меню задания

На вкладке Задание в командной строке можно указать путь доступа к запускаемому приложению. Важно заметить, что командная строка позволяет указать параметры запуска приложения, если оно такой запуск допускает, В частности, параметры командной строки используют для того, чтобы приложение сразу после запуска открывало (воспроизводило) заданный документ. Этот прием позволяет, например, использовать запуск какого-либо музыкального проигрывателя для воспроизведения файла звукозаписи в заданное время (функция будильника). На вкладке Расписание можно уточнить параметры расписания задания, а на вкладке Параметры более детально определить условия исполнения и завершения задания.

Автоматизация поисковых операций. В связи с тем, что файловая структура компьютера может иметь значительный размер, выполнять поиск необходимых документов путем простой навигации по файловой структуре не всегда удобно. Обычно считается, что каждый пользователь компьютера должен хорошо знать (и помнить) структуру тех папок, в которых он хранит документы. Тем не менее, бывают случаи, когда происходит сохранение документов вне этой структуры. Так, например, многие приложения выполняют сохранение документов в папки, принятые по умолчанию, если пользователь забыл явно указать, куда следует сохранить документ. Такой папкой, принятой по умолчанию, может быть папка, в которую последний раз выполнялось сохранение, папка, в которой размещено само приложение, какая-то служебная папка, например \Мои документы и т. п. В подобных случаях файлы документов могут «теряться» в массе прочих данных.

Необходимость в поиске файлов особенно часто возникает при проведении наладочных работ. Типичен случай, когда в поисках источника неконтролируемых изменений в операционной системе требуется разыскать все файлы, подвергшиеся изменению в последнее время. Средствами автоматического поиска файлов также широко пользуются специалисты, выполняющие наладку вычислительных систем, – им трудно ориентироваться в файловой структуре «чужого» персонального компьютера, и поиск нужных файлов путем навигации для них не всегда продуктивен.

Основное поисковое средство *Windows XP* запускают из Главного меню командой Пуск > Найти > Файлы и папки. Не менее удобен и другой вариант запуска – из любого окна папки (Вид > Панели обозревателя > Поиск > Файлы и папки или клавиша F3).

Локализовать сферу поиска с учетом имеющейся информации об имени и адресе файла позволяют элементы управления, представленные на панели поиска. При вводе имени файла разрешается использовать подстановочные символы «*» и «?». Символ «*» заменяет любое число произвольных символов, а символ «?» заменяет один любой символ. Так, например, поиск файла с именем *.txt завершится с отображением всех файлов, имеющих расширение имени .txt, а результатом поиска файлов с именем *.??t станет список всех файлов, имеющих расширения имени .txt, .bat, .dat и так далее.

При поиске файлов, имеющих «длинные» имена, следует иметь в виду, что если «длинное» имя содержит пробелы (а это допустимо), то при создании задания на поиск такое имя следует заключать в кавычки, например: «Текущие работы.c!oc».

На панели поиска имеются дополнительные скрытые элементы управления. Они отображаются, если щелкнуть на раскрывающей стрелке, направленной вниз.

- Вопрос Когда были произведены последние изменения? позволяет ограничить сферу поиска по дате создания, последнего изменения или открытия файла.

- Вопрос Какой размер файла? позволяет при поиске ограничиться файлами определенного размера.
- Пункт Дополнительные параметры позволяет указать тип файла, разрешить просмотр скрытых файлов и папок, а также задать некоторые другие параметры поиска.

В тех случаях, когда разыскивается текстовый неформатированный документ, возможен поиск не только по атрибутам файла, но и по его содержанию. Нужный текст можно ввести в поле Слово или фраза в файле.

Поиск документа по текстовому фрагменту не дает результата, если речь идет о документе, имеющем форматирование, поскольку коды форматирования нарушают естественную последовательность кодов текстовых символов. В этих случаях иногда можно воспользоваться поисковым средством, прилагающимся к тому приложению, которое выполняет форматирование документов.

2.5. Настройка шрифтов

Важным преимуществом графических операционных систем является возможность гибкого управления как экранными, так и печатными шрифтами в документах. Операционная система обеспечивает единство принципов применения шрифтов в самых разнообразных приложениях.

Растровые и векторные шрифты. Операционная система *Windows XP* позволяет работать с двумя классами шрифтов – *растровыми* и *векторными.* Символы растровых шрифтов образуются как комбинации точек в матрице заданного размера. Достоинством растровых шрифтов является высокая скорость отображения символьных данных на экране. В связи с этим операционная система использует растровые шрифты в качестве экранных при отображении системной информации. Основным недостатком растровых шрифтов является негибкость управления размером и начертанием символов.

Размеры символов растровых шрифтов определяются размерами матрицы, на базе которой эти символы построены из комбинации точек. Характерные размеры: 8x12; 10x16; 13x22 и т. п. Изменение размера или начертания шрифта выполняется подменой одного символьного набора другим. При использовании для печати документов устаревшего оборудования (матричных принтеров) возможно использование растровых шрифтов не только для экранного, но и для печатного вывода. Однако при этом качество оттиска получается неудовлетворительным, и документы, полученные таким способом, принято рассматривать как *черновые.* Для печати документов представительного и полиграфического качества растровые шрифты использовать не принято.

Символы векторных шрифтов представляют собой криволинейные контуры, составные элементы которых описываются математическими формулами. Это позволяет не хранить отдельно символьные наборы разных размеров. Управление размером (и некоторыми видами начертания) шрифта

происходит программно. При отображении на экране или при выводе на печать символы любых размеров строятся из одного и того же символьного набора, поэтому векторные шрифты называют также *масштабируемыми.*

Векторные шрифты могут использоваться как в качестве экранных, так и в качестве печатных. Применение векторных шрифтов при подготовке документов позволяет реализовать *принцип соответствия экранного изображения печатному* – так называемый принцип *WYSIWYG* (*What You See Is What You Get).* В соответствии с этим принципом мы наблюдаем оформление документа на экране таким, каким оно будет при выводе с помощью печатающего устройства.

Типы векторных шрифтов. При реализации концепции векторных шрифтов возможны различные подходы к методу построения контуров символов из простейших кривых линий, а также различия в формате записи файла данных, описывающих шрифт. Выработка единого универсального стандарта долгое время сталкивалась с серьезными трудностями, связанными с корпоративной политикой производителей программных средств, а также с особенностями конкурентной борьбы между ними.

Но даже согласование нового общего стандарта (он получил название *Open Type)* пока что не решило всех проблем. Часто приходится иметь дело с шрифтовыми наборами, подготовленными в рамках одного из старых стандартов *True Type* или *Type 1* (*PostScript).*

Операционная система *Windows XP* изначально поддерживает все три эти стандарта и использование любого типа шрифтов не вызывает проблем у современных приложений. Предыдущие операционные системы семейства *Windows* были ориентированы на поддержку шрифтов *True Type,* которые продвигались корпорацией *Microsoft.*

Стандарт *Type 1* (*PostScript)* возник раньше, чем *True Type,* и продвигался компанией *Adobe,* лидером в области программного обеспечения для устройств печати и полиграфических систем. При работе с приложениями, выпущенными этой компанией, иногда целесообразно использовать векторные шрифты *Type 1* (*PostScript).* Предыдущие версии операционной системы *Windows* не могли работать с ними напрямую и нуждались в специальной программе, работающей в фоновом режиме. В частности, в качестве такой программы обычно применялись различные версии программы *Adobe Type Manager.*

Различные проблемы и нестыковки, связанные с использованием векторных шрифтов разных форматов, могут возникать и сегодня, особенно при применении программ прошлых лет выпуска. В операционной системе *Windows XP* вероятность возникновения проблем при применении векторных шрифтов сведена к минимуму, хотя и не исключена полностью. Проблемы могут возникать при использовании старых файлов шрифтов, не содержащих полного описания свойств используемого шрифтового набора.

Системное средство установки и удаления шрифтов. Файлы, содержащие данные о конструкции шрифтовых наборов, находятся в папке \Windows\Fonts, но эту папку не следует обслуживать традиционным средством для работы с файлами и папками, – программой Проводник. Шрифты не стоит устанавливать и удалять путем простого копирования, перемещения и удаления файлов. Гарантию надежной регистрации шрифтов в Реестре операционной системы дают специальные средства обслуживания. Именно процедура регистрации и дает нам возможность напрямую использовать одни и те же шрифты и символьные наборы в различных приложениях.

Система *Windows XP* использует общее средство установки для всех категорий векторных и растровых шрифтов. Оно находится в папке Панель управления (Пуск > Настройка > Панель управления > Шрифты). Просмотр шрифтов, зарегистрированных операционной системой, можно выполнять в следующих режимах:

- Крупные значки;
- Список;
- Подобие (Группировать схожие шрифты);
- Таблица.

Соответствующие элементы управления представлены кнопками панели инструментов окна и пунктами меню Вид. Режимы просмотра Подобие (Группировать схожие шрифты) и Таблица – особые, характерные только для папки \Fonts. В режиме Подобие (Группировать схожие шрифты) отображаются сведения о «похожести» шрифтов на заданный. Шрифт, с которым производится сравнение, выбирают в раскрывающемся списке Группировка шрифтов по сходству. В режиме Таблица для файлов шрифтов приводятся некоторые специальные сведения.

Перед установкой нового шрифта следует закрыть все работающие приложения. Это не значит, что их работа непременно нарушится, – она просто не гарантируется. Установка шрифтов выполняется в диалоговом окне Добавление шрифтов, открываемом по команде Файл > Установить шрифт. Порядок установки следующий:

1. В раскрывающемся списке Диски выбрать диск, на котором расположены файлы устанавливаемого шрифта.

2. В списке Папки выбрать папку, й которой расположены файлы устанавливаемого шрифта.

3. Подождать некоторое время, пока в поле Список шрифтов не сформируется список шрифтов, найденных в указанном источнике.

4. В поле Список шрифтов выбрать устанавливаемые шрифты (при групповом выделении можно применять клавиши SHIFT и CTRL, для установки всех найденных шрифтов – командную кнопку Выделить все).

5. Запустить процесс установки щелчком на командной кнопке ОК.

§. Если шрифт устанавливается для продолжительной работы, целесообразно установить флажок Копировать шрифты в папку «Fonts».

Удаление шрифтов производится командой Файл > Удалить. Соответствующие шрифты при этом должны быть выделены.

2.6. Прочие настройки Windows XP

Настройка системных часов и системного календаря. При сохранении любого файла вместе с ним сохраняются данные о дате и времени создания или последнего изменения. Это сохранение происходит в полном соответствии с текущими настройками системных часов и системного календаря компьютера.

Средства настройки часов и календаря находятся ниже уровня операционной системы. Они относятся к базовому программному обеспечению компьютера и располагаются в его базовой системе ввода и вывода *(BIOS).* Опираясь на показания системных часов, операционная система *Windows XP* обеспечивает следующие функциональные возможности:

- сохранение показаний системных часов вместе с атрибутами файлов при каждой операции сохранения данных;
- предоставление для настройки системных часов и календаря более удобного интерфейса, чем тот, который предоставляет система *BIOS;*
- автоматический учет таких факторов, как изменение поясного времени (это важно для портативных компьютеров), переход на «летнее» и «зимнее» время, учет последних цифр года при смене века;
- возможность корректировки показаний системных часов через Интернет.

В *Windows XP* настройку системных часов и системного календаря выполняют на вкладке Дата и время диалогового окна Свойства: Дата и время, которое открывают с помощью соответствующего значка Панели управления или из контекстного меню индикатора времени, расположенного на панели индикации. Текущий год выставляют с помощью кнопок счетчика. Текущий месяц выбирают в раскрывающемся списке. День месяца выбирают на панели календаря. Точное время устанавливают поразрядно (часы, минуты, секунды) – разряд выбирают с помощью указателя мыши, а значение изменяют с помощью кнопок счетчика. Настройку даты и времени завершают щелчком на командной кнопке Применить (без закрытия окна) или на кнопке ОК (с закрытием).

На вкладке Часовой пояс диалогового окна Свойства: Дата и время присутствуют только два элемента управления: раскрывающийся список для выбора соответствующего часового пояса и флажок для учета перехода на «летнее» и «зимнее» время. Карта мира, представленная здесь, лишь помогает ориентироваться и не является элементом управления.

Вкладка Время Интернета позволяет автоматически установить на компьютере точное время при наличии подключения к Интернету. Синхронизацию часов обеспечивает специальная служба Интернета. После первичной настройки параметров этой вкладки операционная система автоматиче-

ски раз в неделю обращается на соответствующий сервер и поддерживает правильность показаний системных часов.

Учет рубежа веков. Первоначально в формате записи атрибутов файлов для регистрации года было выделено только два разряда, в которые записывались две последние цифры номера года. Это вызвало появление так называемой «проблемы 2000 года». В связи с тем, что последние две цифры дат начала XXI века представляют меньшее число, чем две последние цифры дат конца XX века, появилась угроза, что автоматические системы обработки данных будут некорректно интерпретировать даты создания файлов. При этом может нарушиться работа алгоритмов, выполняющих сравнения дат и расчеты интервалов времени между датами, относящимися к разным столетиям.

Во всех версиях операционной системы *Windows,* начиная с *Windows 98,* эта проблема решена благодаря подходу, осно-$_{v}$ванному на введении понятия *логического столетия.* По умолчанию логическим столетием считается период с 1930 по 2029 год (пользователь может самостоятельно изменить этот интервал). В операциях сравнения дат и вычисления интервалов времени последние две цифры номера года рассматриваются как относящиеся не к календарному, а к логическому столетию. Необходимый пересчет операционная система выполняет автоматически.

Настройка даты логического столетия выполняется в диалоговом окне Язык и региональные стандарты, которое открывают щелчком на значке Язык и региональные стандарты в окне Панели управления. Далее надо щелкнуть на кнопке Настройка на вкладке Региональные параметры. В новом диалоговом окне необходимые элементы управления представлены в группе Календарь на вкладке Дата.

Настройка национальных стандартов и форматов. Операционная система *Windows XP* в значительной степени учитывает национальные различия, связанные с форматами записи чисел, дат, времени, денежных сумм и т. п. Так, например, в России принято представлять даты в формате дц.мм.гг (день-месяц-год), а в США – ММ.ДЦ.ГГ (месяц-день-год). Например, запись 08.03.03 в России означает 8 марта 2003 года. Та же запись в США означает 3 августа того же года. Другой пример: в России десятичная часть дробного числа отделяется от целой части с помощью запятой, а в США – точкой. Есть различия в форматах записи времени, отрицательных чисел, денежных сумм, единиц измерения физических и денежных величин. Такие различия относятся не только к России и США, но и к другим странам.

Обычно при установке локализованной версии операционной системы настройка национальных стандартов и форматов производится автоматически, в соответствии с указанием страны пребывания в устанавливающей программе. Однако при работе с некоторыми приложениями, не адаптированными к использованию в конкретной стране, необходимо редактировать настройки, принятые по умолчанию. В таких случаях редактирование настроек

выполняют на вкладках диалогового окна Язык и региональные стандарты (Пуск > Настройка > Панель управления > Язык и региональные стандарты).

На вкладке Языки этого окна имеется кнопка Подробнее, которая дает доступ к средствам для установки дополнительных раскладок клавиатуры и элементам управления для выбора клавиатурной комбинации, переключающей раскладку. Здесь же можно выбрать раскладку, принятую по умолчанию, а также задать отображение языковой панели в пределах Панели задач. При наличии этой панели диалоговое окно настройки раскладок клавиатуры удобнее открывать через контекстное меню панели (Параметры).

2.7. Справочная система Windows XP

Современное программное обеспечение отличается высокой сложностью, поэтому и в операционной системе, и в большинстве ее приложений предусмотрено наличие справочных систем. Справочную систему *Windows XP* можно рассматривать как автоматизированное информационно-справочное средство.

Справочная система в диалоговых окнах. В *Windows XP* реализовано несколько уровней доступа к справочной информации. Особенно часто потребность в быстрой и конкретной справке возникает при работе с элементами управления диалоговых окон. Эту возможность предоставляет специальная кнопка подсказки, расположенная в правом верхнем углу диалоговых окон рядом с закрывающей кнопкой. После щелчка на кнопке подсказки указатель мыши принимает форму вопросительного знака. Если навести его в таком состоянии на один из элементов управления диалогового окна и щелкнуть левой кнопкой, появляется всплывающая подсказка, в которой описано назначение данного элемента управления. Этим приемом пользуются при изучении новых диалоговых окон.

Контекстная подсказка. Прием получения контекстной подсказки действует в большинстве диалоговых окон и в некоторых окнах приложений. Его удобно рассматривать на примере стандартной программы Калькулятор, входящей в комплект поставки *Windows XP* (Пуск > Программы > Стандартные > Калькулятор).

Окно Калькулятора не является диалоговым – это рабочее окно приложения, но оно тоже содержит немало всевозможных элементов управления. Поскольку это не диалоговое окно, в его правом верхнем углу нет кнопки подсказки, однако подсказку по назначению элементов управления получить все-таки можно.

Щелкните правой кнопкой мыши на любом элементе управления, и рядом с ним появится кнопка контекстной подсказки с надписью Что это такое? Если щелкнуть на этой кнопке, откроется всплывающая подсказка с описанием назначения элемента управления.

Справочная система Windows. Классический прием вызова справочной системы *Windows* состоит в использовании Главного меню (Пуск >

Справка и поддержка), но то же можно сделать из строки меню любого окна папки или Проводника (Справка > Центр справки и поддержки).

Есть три основных способа использования справочной системы *Windows XP*. В первом случае надо выбрать нужный тематический раздел на панели Раздел справки. Содержание выбранного раздела представляется в виде иерархической структуры данных, похожей на содержание обычных книг. Разделы самого высокого уровня легко охватить беглым взглядом. Раздел раскрывается одним щелчком левой кнопки мыши. Внутри раздела могут содержаться вложенные разделы или отдельные статьи. При щелчке на статье ее содержимое отображается на правой панели.

Статьи справочной системы, представленные на правой панели, могут активно использовать так называемые *перекрестные ссылки*. Перекрестные ссылки оформлены в виде выделенных фрагментов текста. При щелчке на таких фрагментах происходит переход к другой статье, содержимое которой дополняет или уточняет первую. Текст, содержащий ссылки между отдельными статьями, называется *гипертекстом.* Для того чтобы не запутаться при движении по гипертекстовому документу и иметь возможность вернуться к исходному пункту, используют кнопки панели инструментов Назад и Вперед.

Если щелкнуть на кнопке Указатель на панели инструментов, данные на левой панели будут представлены в виде линейной структуры (списка). Фактически это алфавитный указатель, аналогичный тем, которые можно встретить в конце научно-технических изданий. Здесь приведены термины, встречающееся в справочной системе программы. Если нужно найти конкретные данные и не хочется просматривать все содержание справочной системы, используют указатель.

Отыскав нужный термин или понятие, надо щелкнуть на нем дважды. Если этот термин встречается только в одной статье справочной системы, то на правой панели сразу будет отображен текст статьи. Если данный термин встречается несколько раз, появляется диалоговое окно, в котором можно выбрать нужную статью из предлагаемого списка.

Для таких крупных систем, как *Windows XP* последовательный просмотр и содержания, и указателя может быть неудобным. В этом случае искомое слово или словосочетание вводят в поле Найти и щелкают на кнопке Начать поиск. Если это слово встречается в статьях справочной системы, на экране отображается список соответствующих статей. Эти статьи разбиты на две категории. Категория Рекомендуемые разделы отображает статьи, тема которых заведомо связана с запросом. В категории Полнотекстовый поиск перечислены все статьи, содержащие текст запроса. Просмотр включают щелчком на названии статьи.

За время существования операционных систем семейства *Windows* механизм организации поиска менялся несколько раз. В приложениях или в других версиях системы окна справки могут выглядеть иначе. Однако во всех

случаях сохраняется возможность поиска нужной информации по иерархическому дереву разделов, по алфавитному указателю и по содержанию статей.

Задание: Изучить представленный материал и настроить параметры Windows собственного ПК для повышения эффективности своей работы. В отчете о проделанной работе представить информацию о выбранных настройках Windows XP, которые пришлось изменить и обсоновать эти изменения.

ОРГАНИЗАЦИЯ ХРАНЕНИЯ ДАННЫХ НА ПК

*Практическая работа №*3

Файловая структура

Цель занятия: Ознакомление с общими принципами организации хранения данных на компьютере и понятием файловой системы; овладение навыками работы с объектами файловой структуры – файлами и папками средствами операционной системы и файловых менеджеров.

3.1. Структура хранения данных. Файловая система

Файловая система является важной частью любой операционной системы, которая отвечает за организацию хранения и доступа к информации на каких-либо носителях. Рассмотрим в качестве примера файловые системы для наиболее распространенных в наше время носителей информации – жёстких магнитных дисков (HDD). Как известно, информация на жестком диске хранится в секторах (объем сектора составляет обычно 512 байт) и само устройство может выполнять лишь команды чтения/записи информации в определенный сектор на диске. В отличие от этого файловая система позволяет пользователю оперировать с более удобным для него понятием – файл (документ).

Файл – это последовательность произвольного числа байтов данных одного типа, обладающая уникальным собственным именем. Тип данных определяет тип файла, который указывается в расширении имени. Хранение файлов организуется в иерархической структуре, которая в данном случае называется файловой структурой. В качестве вершины структуры служит имя носителя, на котором сохраняются файлы. Далее файлы группируются в каталоги (папки), внутри которых могут быть созданы вложенные каталоги (папки). Под каталогом (папкой) в файловой системе понимается, с одной стороны, группа файлов, объединенных пользователем исходя из некоторых соображений, с другой стороны каталог – это файл, содержащий системную информацию о группе составляющих его файлов. Уровни в файловой структуре создаются за счет каталогов, содержащих информацию о файлах и каталогах более низкого уровня (рис. 2.1). Путь доступа к файлу начинается с имени устройства и включает все имена каталогов (папок), через которые он проходит. В качестве разделителя используется символ \. Полный путь к файлу index.dat выглядит таким образом: F:\Program Files\ABBYY Lingvo 8.0\Dic\Index \index.dat; он показывает вложенность каталогов, начиная с верхнего уровня (раздел жесткого диска F:) до непосредственно самого файла, находящегося в каталоге Index. Файловая система берет на себя организацию взаимодействия программ с файлами, расположенными на дисках.

В широком смысле понятие "файловая система" включает:

- совокупность всех файлов на диске (файловая структура);

- наборы служебных структур данных, используемых для управления файлами, такие как, например, каталоги файлов, дескрипторы файлов, таблицы распределения свободного и занятого пространства на диске,
- комплекс программных средств, реализующих управление файлами, в частности операции по созданию, уничтожению, чтению, записи, именованию файлов, установке атрибутов и уровней доступа, поиску и т.д.

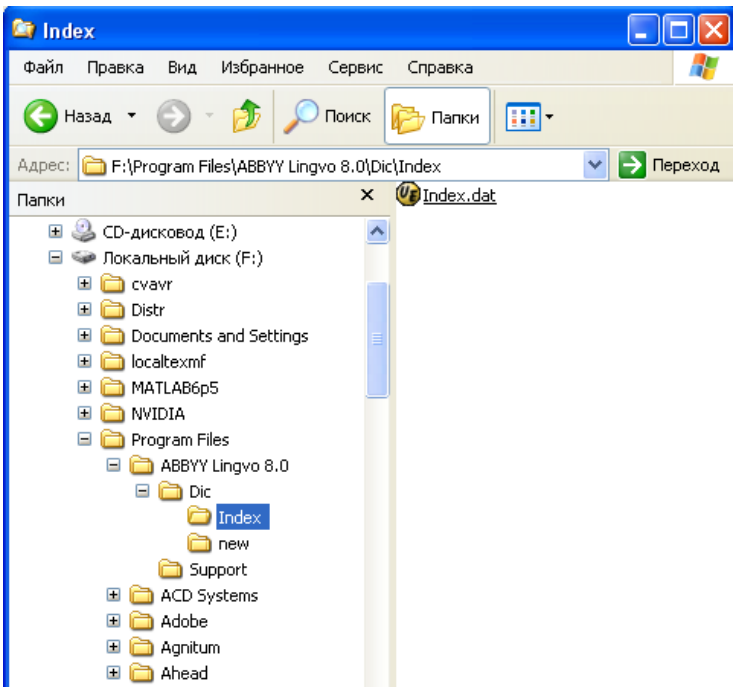


Рис. 2.1. Иерархическая структура каталогов (дерево папок)

Различие между файловыми системами заключается, в основном, в способах распределения дискового пространства между файлами и организации на диске служебных областей. Современные операционные системы стремятся обеспечить пользователя возможностью работать одновременно с несколькими файловыми системами. При этом файловая система рассматривается как часть подсистемы ввода-вывода. В большинстве операционных систем реализуется механизм переключения файловых систем, позволяющий поддерживать различные их типы.

Современные файловые системы должны обеспечивать эффективный доступ к файлам, поддержку носителей данных большого объема, защиту от несанкционированного доступа к данным и сохранение целостности данных. Под целостностью данных подразумевается способность файловой системы обеспечивать отсутствие ошибок и нарушений согласованности в данных, а также восстанавливать поврежденные данные.

3.2. Операции с файловой структурой

Практически все задачи, выполняемые на компьютере, включают работу с файлами и папками (каталогами), которые и составляют операции с файловой структурой. Эти операции можно разделить на три категории:

1) *Организация и управление файлами и папками.* С папками и файлами можно выполнять простейшие операции, такие как создание, удаление, копирование и перемещение, а также более сложные операции: изменение свойств файлов и папок и управление доступом к папкам
2) *Поиск файлов и папок.* Запросы на поиск файлов или папок могут уточняться заданием дополнительных критериев поиска, например даты, типа, размера файла или учёта регистра.
3) *Обеспечение безопасности папок и файлов.* Эти возможности включены в семейство Windows NT, они могут осуществляться с помощью учётных записей групп и пользователей, групповой политики, аудита и прав пользователей. Если используется файловая система NTFS, то можно задавать разрешения на файлы и папки, а также включить шифрование.

Эти операции можно производить как средствами самой операционной системы (Мой компьютер, Проводник в Windows) так и при помощи специализированных приложений – файловых менеджеров. Среди наиболее распространённых можно отметить Norton Commander, Far, Total Commander и другие.

3.3. Проводник

Запуск Проводника осуществляется двойным щелчком мыши на иконку «Мой компьютер» на Рабочем столе Windows или нажать кнопку Пуск и выбрать команды Программы>Стандартные>Проводник. Изменить способ отображения файлов в окне позволяют команды меню Вид. Для изменения параметров файла или папки также можно использовать вкладку Вид диалогового меню Свойства папки.

Окно Проводника показано на рис. 2.2.

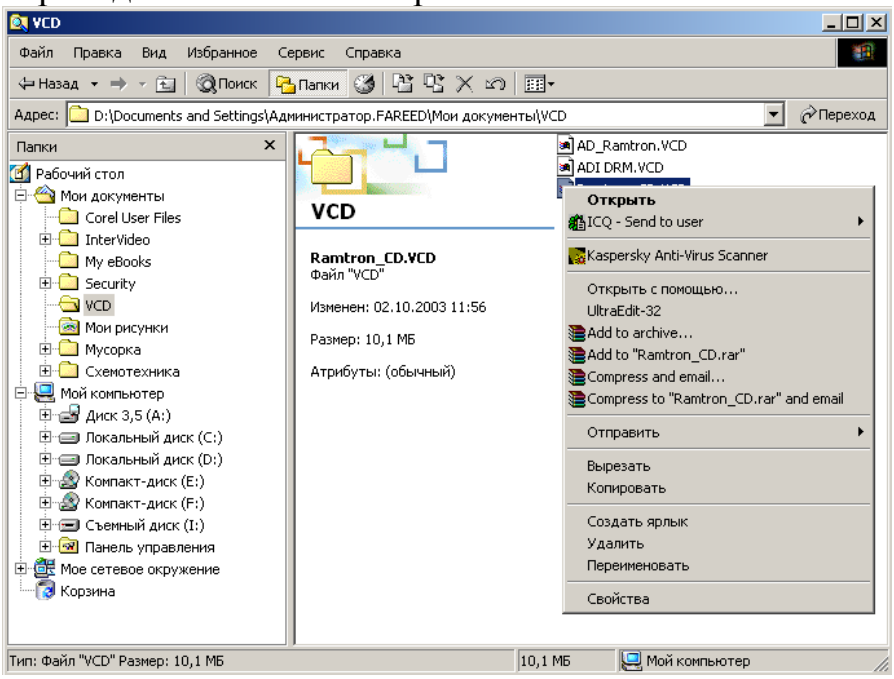


Рис. 2.2. Окно Проводника с контекстным меню для выбранного файла

Чтобы открыть файл или папку в Проводнике необходимо дважды щелкнуть левой клавишей мыши на иконке требуемого объекта. Если открываемый файл не связан ни с каким приложением, можно указать программу для открытия файла, щёлкнув на иконке правой кнопкой мыши, выбрать команду Открыть с помощью и требуемое приложение из открывшегося списка.

Для создания новой папки или файла необходимо выбрать в Проводнике диск или папку, в которой они будут создаваться. В меню Файл перейти на команду Создать и выбрать Папку или требуемый тип файла.

Чтобы скопировать или переместить файл или папку в Проводнике, необходимо выделить требуемые объекты и в меню Правка выбрать команду Копировать (или нажать комбинацию клавиш CTRL+C) или Вырезать (CTRL+X). Затем открыть папку или диск, куда нужно скопировать, и в меню Правка выбрать команду Вставить (CTRL+V). Для того, чтобы выбрать несколько объектов, расположенных непоследовательно, нужно щёлкнуть по каждому левой кнопкой мыши, удерживая при этом нажатой клавишу CTRL. Для копирования или перемещения объектов на дискету, необходимо также скопировать или вырезать файлы и затем в меню Файл выбрать команду Отправить и далее Диск 3,5 (A). Там же находятся и другие места для возможного перемещения.

Чтобы изменить имя файла или папки необходимо в Проводнике выбрать требуемый объект, затем в меню Файл выбрать команду Переименовать, ввести новое имя и нажать клавишу ENTER. Нет необходимости открывать файл или папку, чтоб их переименовать. Имена некоторых системных папок не могут быть изменены пользователем.

Для удаления файла или папки необходимо в Проводнике выбрать требуемые объекты, затем в меню Файл выбрать команду Удалить. При этом объекты перемещаются в Корзину и хранятся там, пока она не будет очищена, если во время удаления объектов не была нажата клавиша SHIFT. Иначе объекты будут удалены без помещения в Корзину. Для восстановления удалённого объекта дважды щёлкните на иконке Корзины на Рабочем столе Windows, щёлкните нужный объект правой кнопкой мыши и выберите команду Восстановить. Для очистки Корзины и полного удаления хранимых в ней объектов щёлкните по её иконке на Рабочем столе и выберите команду Очистить корзину.

Для более удобного и быстрого запуска приложений и открытия файлов можно создать ярлык (ссылка в виде файла на любой объект, доступный на компьютере или в сети). Для этого в Проводнике Windows следует выделить нужный объект и в меню Файл выбрать команду Создать и далее Ярлык. Следуйте инструкциям мастера создания ярлыка. После этого можно запускать приложение двойным щелчком мыши по ярлыку из любого места, где бы он не находился.

Для поиска требуемого файла или папки нажмите кнопку Пуск, выберите Найти, а затем Файлы и папки. В поле Искать имена файлов и папок введите имя или часть имени объекта, который требуется найти. Для поиска файлов, содержащих конкретный текст, введите искомый текст в поле Искать текст. В поле Поиск в выберите диск, папку или сетевой ресурс, где требуется выполнить поиск. Можно задать дополнительные условия поиска по ссылке Параметры поиска: Дата – дата создания файла, Тип – конкретный тип файла (например, текстовый), Размер – поиск по размеру файла. Затем следует нажать кнопку Найти и система выведет на экран все результаты поиска. Для нового поиска следует нажать кнопку Назад.

Стоит отметить, что к большинству вышеуказанных команд имеется доступ и в контекстном меню файла или папки (оно вызывается щелчком правой кнопки мыши над указанным объектом). Также в меню есть пункт Свойства, где можно устанавливать или изменять свойства и параметры файлов и папок (в том числе параметры доступа и безопасности в NT-системах при наличии соответствующих прав).

3.4. Задание

1) Создать в своей папке систему каталогов со структурой, приведенной на рис. 3.4.
2) Создать в папке Beta два файла с расширением *txt* и три файла с расширением *doc*.
3) В папке Odin создать файл с расширением *pas* и два файла с расширением *txt*.
4) В папке Dva создать 3 файла с расширением *pas* и два файла с расширениями *txt* и *doc*. Содержание файлов может быть произвольным.
5) Создать в своей папке каталог Rabota. Скопировать в него все файлы из других папок в своей директории.
6) Отсортировать файлы по имени, размеру, расширению в убывающем и возрастающем порядке.
7) Получить информацию об объёме, занимаемом файлами в папке Rabota.
8) Оценить общий объем логического диска, процент занятого и свободного места на нем.
9) Удалить все файлы из папки Rabota в Корзину. Восстановить все файлы из Корзины. Удалить папку Rabota, не помещая её в Корзину.
10) Скопировать из всех созданных папок в папки Pas и Txt все файлы с расширениями *pas, doc* и *txt* соответственно.
11) Переместить из папки Doc в папку Txt все находящиеся там файлы с одновременным переименованием расширений с *doc* на *txt*.
12) Удалить все созданные папки в Корзину. Очистить Корзину.

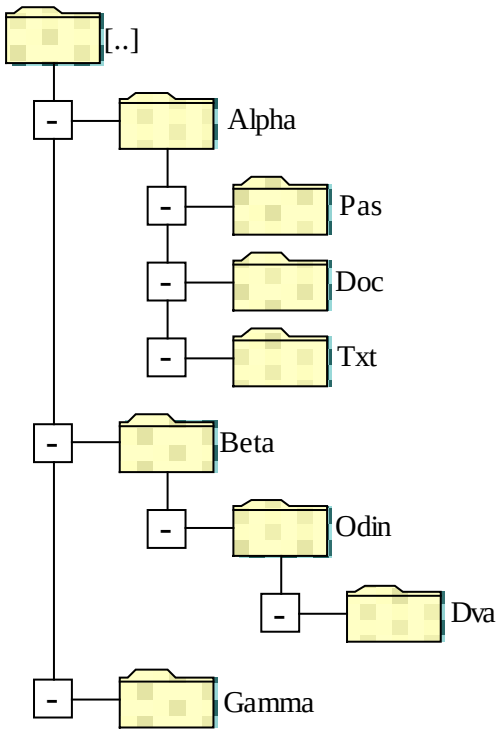


Рис. 3.4. Дерево каталогов

3.5. Порядок выполнения работы

1. Ознакомиться с теоретическими материалами пп. 1-3 описания лабораторной работы.
2. Выполнить задание на ПК средствами Проводника Windows или любого установленного на компьютере файлового менеджера.
3. Составить отчёт о проделанной работе в письменном виде с описанием последовательности действий по каждому пункту задания. В отчете необходимо дать сравнительную оценку возможностям Проводника Windows и файлового менеджера по операциям с объектами файловой структуры.
4. При сдаче работы необходимо продемонстрировать результаты на ПК, представить отчёт и ответить на контрольные вопросы.

2.6. Контрольные вопросы

1) Что такое файловая структура; файловая система?
2) Назовите основные функции файловой системы?
3) Перечислите распространенные файловые системы и их основные характеристики и различия?
4) Каким образом операционная система сможет работать с жесткими дисками с разными файловыми системами?
5) Какие основные характеристики системы NTFS делают её незаменимой для корпоративного применения?
6) Перечислите основные операции с файловой структурой?

*Практическая работа №*4

Архивирование данных

Цель занятия: ознакомление с общими принципами архивирования и резервного копирования данных, с основными программами-архиваторами и приобретение навыков работы с архивами.

4.1. Общие сведения об архивировании данных

Для повышения эффективности использования дискового пространства ПК хранимые файлы можно подвергать архивированию (сжатию, компрессии) – обработке по специальному алгоритму, уменьшающему длину файла без потери информации. Соответственно, должен быть обратный алгоритм декомпрессии, абсолютно точно восстанавливающий исходный файл из сжатого. Существует ряд методов, различающихся достижимой степенью сжатия и необходимыми для этого вычислительными ресурсами (объём используемой оперативной памяти и время работы процессора). По общим соображениям, очевидно, что большее сжатие требует больших вычислительных затрат. Достижимая степень сжатия зависит и от характера данных – некоторые файлы при попытке сжатия даже увеличиваются в объёме. Хорошо, например, сжимаются текстовые файлы (просто символьные или файлы MS Word). Практически не сжимаются файлы типа pdf (формат Adobe Acrobat), даже если они содержат просто текст.

С ростом мощности персональных компьютеров становились приемлемыми алгоритмы, обеспечивающие всё большую степень сжатия. Так, первые архиваторы файлов ARC, PKARC и PACK сменились более мощными PKZIP, ACE, CAB, TAR, ARJ, RAR и другими. Хранение файлов в виде упакованных архивов даёт экономию места на диске сразу по двум статьям: во-первых, уменьшается длина файла (число байт), во-вторых, если архивируется группа файлов, то уменьшаются потери на недоиспользованных кластерах (в среднем по половине кластера на файл).

Современные архиваторы совместно с операционной системой позволяют работать с файлами прямо из архива – выбранный файл распаковывается в каталог для временного хранения и оттуда передаётся требуемому приложению. Для этого достаточно открыть архив программой архиватором и щёлкнуть мышкой по требуемому файлу. Кроме, того, архивирование файлов широко применяется при периодическом резервном копировании важных данных на съёмные носители: гибкие диски, ZIP или ZIV диски, компакт-диски CD-R и CD-RW, съёмные жесткие диски, и т.п.

Кроме сжатия отдельных файлов применяют и дисковые компрессоры – программные средства, сжимающие данные на диске «прозрачно» для приложений (и пользователя). Каждый раз при записи файла (или его фрагмента) выполняется компрессия, а при чтении – декомпрессия. Конечно, для исполнения в реальном времени пригодны не всякие алгоритмы компрессии, и ра-

ди экономии времени жертвуют достижимой степенью сжатия. Возможность сжатия заложена в такие сложные файловые системы, как, например, Novell NetWare (начиная с версии 4.x) и NTFS. Для файловой системы FAT (MS DOS и Windows 9x) встроенных компрессоров не предусмотрено, но с ними широко используются загружаемые компрессоры типа Stacker, DoubleSpace и DriveSpace.

Идея этих компрессоров заключается в следующем. На обычном логическом диске, называемом *несущим*, размещается большой файл-образ сжатого диска CVF (Compressed Volume File). Во время загрузки операционной системы (ОС) в оперативную память помещается резидентный драйвер, файл которого находится в корневом каталоге несущего диска. Этот драйвер эмулирует обращения к реальному диску операциями доступа к файлу-образу, на ходу осуществляя компрессию/декомпрессию. Для ОС эмулируемый диск выглядит как обычный логический диск и для удобства ему может назначаться логическое имя (буква), ранее принадлежащее несущему диску. Несущий диск при этом получает новое (ранее неиспользованное) имя, и к нему, в принципе, тоже можно обращаться обычным образом. Несущий диск можно и скрыть от приложений и пользователя (чтобы не было попыток удалить «никому не нужный» громадный файл-образ). На несущем диске должны оставаться файлы, необходимые для загрузки ОС, драйвер программы-компрессора и сам файл-образ. В состав программы-компрессора также входит утилита, позволяющая перемещать обычные файлы в файл-образ, сжимая их при этом. При желании (и наличии достаточного свободного места) можно выполнить обратное перемещение файлов («разжать» диск).

Применение дисковых компрессоров имеет свои плюсы и минусы. С одной стороны, на диск, в среднем, удаётся поместить данных в 1,5-2 раза больше, чем на несжатый диск. При этом скорость обращения к файлам может даже и не упасть (при мощном процессоре), поскольку реальный объём обменов данными с физическим диском сокращается. Однако компрессор является дополнительным звеном, повышающим уязвимость данных в случае сбоев, аварий и вирусных атак. Пользователям сжатых дисков особенно рекомендуется регулярно архивировать важные данные на внешних носителях. Используя сжатые диски, довольно просто переносить сложные структуры файлов и каталогов – достаточно скопировать один файл-архив и подключить его там, где нужно. При использовании сжатого диска объём свободного пространства точно неизвестен, поскольку компрессор ориентируется на ожидаемую степень сжатия, а реальная может заметно отличаться от неё в любую сторону. Файлы-архивы (ZIP, RAR, ARJ и др.), а также файлы со сложными данными (например, pdf) дисковым компрессором уже не сожмутся.

4.2. Архивирование данных с помощью программы WinRAR

Рассмотрим кратко некоторые функции и возможности, предоставляемые современными архиваторами, на примере одной из широко распространённой программы – WinRAR версии 3.xx. Приложение может быть использовано как в режиме командной строки, так и в оконном режиме обычного Windows-приложения. Кроме того, при установке программы на ПК есть возможность встроить вызов наиболее распространённых функций архивирования и разархивирования в контекстное меню операционной системы, что делает использование программы максимально удобным.

Запустить WinRAR в режиме командной строки можно следующим образом: *WinRAR <command> -<sw1>...-<swN> <archive> <files> <path_to_extract>* где

- *command* – комбинация символов, соответствующая вызываемой функции WinRAR;
- *sw1, ..., swN* – ключи, используемые для задания параметров вызываемой функции;
- *archive* – имя архива;
- *files* – имена файлов для архивирования;
- *path_to_extract* – используется с командами *–e* и *–x* для обозначения папки, в которую распаковываются файлы (в случае указания несуществующей папки она будет создана в указанном месте).

Некоторые команды представлены ниже:

a – добавить файлы в архив;
c – добавить комментарий к архиву;
d – удалить файлы из архива;
e – извлечь файлы из архива без учёта пути;
f – обновить файлы в архиве;
k – защитить архив;
m – переместить файлы и папки в архив;
r – восстановить повреждённый архив;
rn – переименовать архивируемые файлы;
rr[*N*] – добавить информация для восстановления данных;
rv[*N*] – создать тома восстановления;
s[*name*] – преобразовать архив в самораспаковывающийся тип;
t – проверить заархивированные файлы;
u – обновить файлы в архиве с добавлением новых файлов;
x – извлечь файлы из архива с учётом всех путей.

При запуске WinRAR в оконном режиме приложение имеет вид, представленный на рис. 4.1. Все команды доступны в наглядном виде в меню приложения. Программа имеет развитую систему помощи; кроме того, есть ряд мастеров (wizards), помогающих быстро освоить возможные функции архиватора.

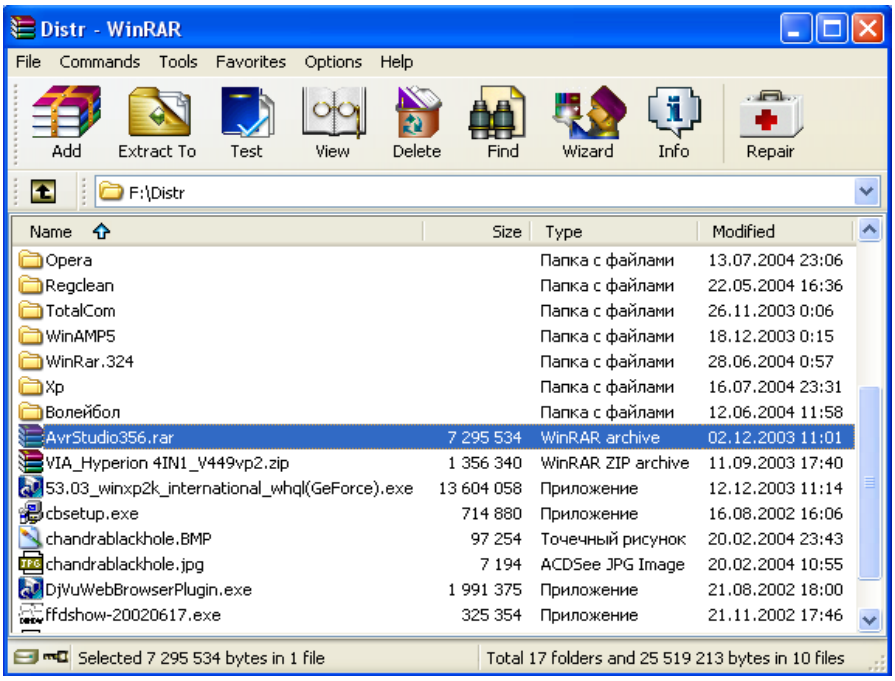


Рис. 4.1. Внешний вид окна приложения-архиватора WinRAR

На рис. 4.2 показано контекстное меню, содержащее встроенные команды WinRAR. Здесь же в пункте меню “Свойства” находится вкладка “Archive”, где можно просмотреть степень сжатия и другие параметры конкретного файла архива.

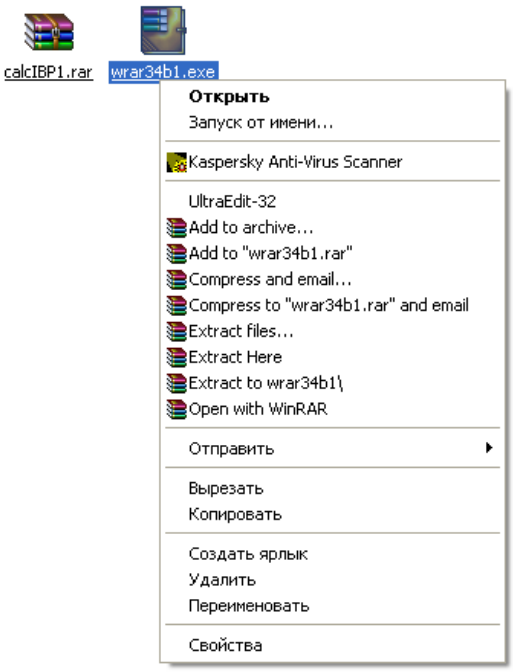


Рис. 4.2. Контекстное меню со встроенными командами WinRAR

4.3. Задание

1) Создать в своей папке каталог с именем “Архив”, скопировать в него несколько файлов различного типа.

2) Создать архив, включив в него скопированные файлы (без удаления исходных файлов из папки), используя WinRAR в режиме командной строки, оконном режиме и из контекстного меню.
3) Оценить степень сжатия файлов разных типов в архиве при помощи контекстного меню и в окне приложения WinRAR.
4) Распаковать архив в любую другую папку в пределах своего каталога, сравнить исходные и распакованные файлы.
5) Создать самораспаковывающийся архив, включив в него исходные файлы из каталога “Архив”, используя команды WinRAR в режиме командной строки и оконном режиме (архив создать с одновременным удалением исходных файлов из папки).

4.4. Порядок выполнения работы

1) Ознакомиться с теоретическим материалом пп. 1-3 описания лабораторной работы.
2) Выполнить задание на ПК.
3) Составить отчёт о проделанной работе в письменном виде. В отчете привести характеристики файлов до и после архивирования.
4) При сдаче лабораторной работы необходимо продемонстрировать результаты на ПК, представить отчёт и ответить на контрольные вопросы.

4.5. Контрольные вопросы

1) Какие виды архивирования существуют?
2) Для чего необходимо проводить архивирование?
3) Чем определяется максимальная степень сжатия файлов при архивировании?
4) Почему разные типы файлов по-разному сжимаются?
5) Как можно предотвратить повреждение и потерю информации в архиве?

ОБРАБОТКА ТЕКСТОВОЙ ИНФОРМАЦИИ НА ПК

Практическая работа №5

Разработка текстовых документов в процессоре Microsoft Word

Цель занятия**:** изучение возможностей текстового процессора Microsoft Word и овладение навыками подготовки комплексных документов на ПК.

5.1. Общие сведения о процессоре Microsoft Word

Приложение MS Word из пакета MS Office представляет собой мощное полнофункциональное средство для редактирования документов, содержащих, помимо текста, рисунки, диаграммы, таблицы, анимацию и звуковые элементы, а также внедренные и связанные объекты из других приложений Windows. Описать все возможности данного приложения в методическом пособии не представляется возможным, но следует отметить, что MS Word имеет встроенную справочную систему, включающую интерактивную систему помощи – так называемого Помощника, кроме того, недостающие разделы Справки можно оперативно найти на сайте Microsoft.

Запуск программы осуществляется следующим образом: нажмите кнопку Пуск и перейдите в меню Программы>Microsoft Word. Окно приложения показано на рис. 5.1 и состоит из рабочей области, в которой происходит набор и редактирование текста, горизонтальной и вертикальной полос прокрутки, внизу располагается строка состояния. Вверху окна расположена строка заголовка с именем редактируемого файла, строка меню, и панель инструментов.

Строка меню содержит все команды приложения MS Word. Меню Файл дает доступ к командам создания нового документа, открытия существующего текста из файла, сохранения документа в файл, настройки свойств отображения страницы (поля, ориентация бумаги и т.п.) и вывода на печать. Меню Правка содержит команды редактирования текста, отмены последнего действия, поиска и замены фрагментов текста.

Меню Вид содержит команды настройки отображения документа, указания масштаба, вставки колонтитулов на страницу, построения схемы документа. Пункт Вставка позволяет вставить в текст различные объекты: символы из предлагаемого набора символов, номера страниц, дату и время, ссылку (сноски, указатели, пункты оглавления), закладку, организационную диаграмму, гиперссылку, объекты из других приложений Windows и т.д.

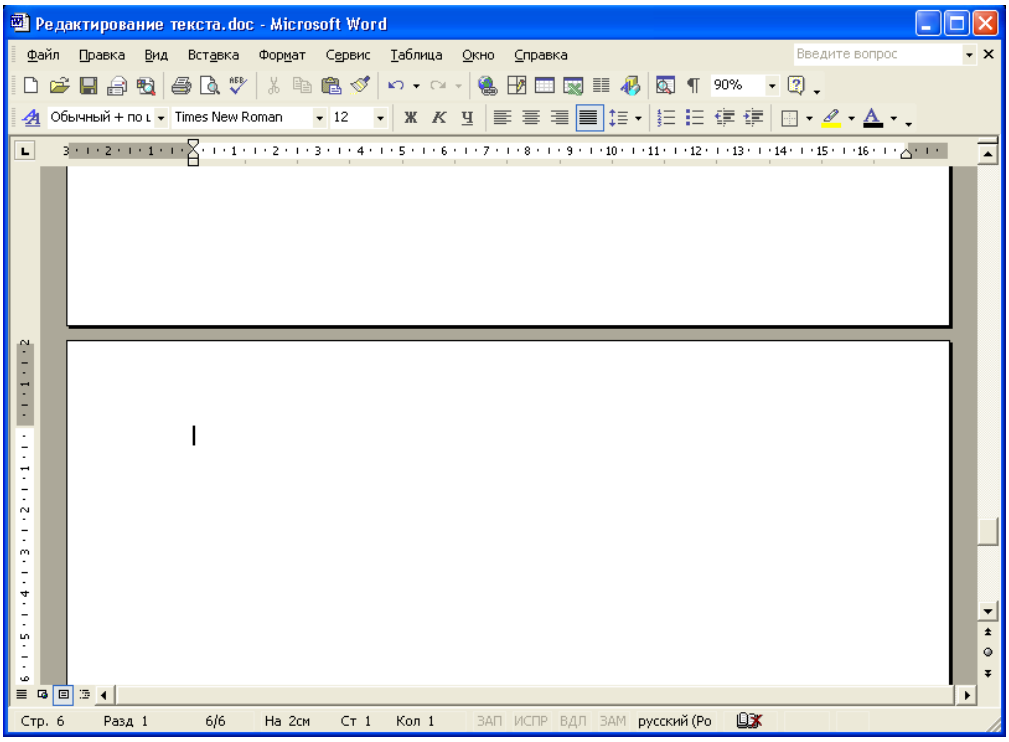


Рис. 5.1. Окно приложения MS Word

Меню Формат содержит команды настройки шрифтов, форматирования абзацев, оформления списков, границ, заливки текста, позволяет разбить текст на колонки, а также содержит набор автоформатов – готовых стилей для автоматического оформления текста.

Меню Сервис содержит инструменты для проверки правописания, выбора языка документа, расстановки переносов, настройки редактора MS Word. Меню Таблица предназначено для вставки, оформления, редактирования таблиц.

Панель инструментов предназначена для быстрого доступа к различным командам редактора, она состоит из нескольких панелей (Стандартная, Форматирование, Рисование и т.д.). В меню Сервис>Настройка во вкладке Панели инструментов (или меню Вид>Панели инструментов) можно выбрать те панели, которые необходимы в данный момент для работы и будут отображены в панели инструментов.

5.2. Приемы работы с текстами в процессоре Microsoft Word

К базовым приемам работы с текстами в текстовом процессоре Microsoft Word относятся следующие:

- создание документа;
- ввод текста;
- редактирование текста;
- форматирование текста;
- сохранение документа;
- печать документа.

5.2.1. Создание документа

В текстовом процессоре MS Word принято использовать два метода создания нового документа: на основе готового шаблона или на основе существующего документа. При создании документа на основе существующего документа открывают готовый документ (Файл > Открыть), сохраняют его под новым именем (Файл > Сохранить как), затем выделяют в нем все содержимое (Правка > Выделить все) и удаляют его нажатием клавиши DELETE, после чего получают пустой документ, имеющий собственное имя и сохраняющий все настройки, ранее принятые для исходного документа.

Создание документа на основе готового шаблона выполняется командой Файл>Создать. В открывшемся диалоговом окне Создание файла, включают переключатель Создать документ и выбирают шаблон. Если никаких предпочтений нет, следует выбрать шаблон Обычный на вкладке Общие. Созданный документ приобретает имя Документ1, принятое по умолчанию. Его целесообразно сразу же сохранить под «правильным» именем, выбрав для него соответствующую папку и дав команду Файл> Сохранить как.

При необходимости сохранить документ в произвольную папку, не представленную в предлагаемом списке, следует выполнить навигацию по файловой структуре с использованием раскрывающей кнопки на правом краю поля Папка.

5.2.2. Ввод текста

Технология ввода текста и переключения языковых раскладок клавиатуры, применение регистровых клавиш и буфера обмена аналогична технологии при использовании текстовых редакторов Блокнот и WordPad. Microsoft Word имеет ряд дополнительных возможностей, позволяющих автоматизировать ввод текста.

Средства отмены и возврата действий. Все операции ввода, редактирования и форматирования текста протоколируются текстовым процессором, и потому необходимое количество последних действий можно отменить. Последнее действие отменяют комбинацией клавиш CTRL+Z. Другие аналогичные средства команда Правка>Отменить действие и кнопка Отменить действие на панели инструментов Стандартная.

После отмены ряда действий существует возможность вернуться к состоянию, предшествовавшему отмене. Для этого служит команда Правка > Вернуть действие или кнопка Вернуть действие на панели инструментов Стандартная.

Расширенный буфер обмена. Необходимые элементы управления находятся на панели инструментов Буфер обмена (Вид>Панели инструментов>Буфер обмена). О содержании конкретной ячейки можно судить по всплывающей подсказке, отображаемой при наведении указателя мыши на ячейку. При переполнении расширенного буфера обмена ячейки сдвигаются

вниз, очередной элемент поступает в последнюю ячейку, содержимое первой ячейки теряется.

Автотекст. Автотекст — это режим автоматического ввода фрагментов текста. Он представлен двумя функциями: *автозавершением* и собственно *автотекстом.*

Текстовый процессор хранит *словарь автотекста,* состоящий из слов и фраз, встречающихся в документах достаточно часто. При вводе первых четырех символов словарного элемента на экране появляется всплывающая подсказка с полным текстом слова или фразы. Если это то, что имел в виду пользователь, он завершает ввод всего фрагмента нажатием клавиши ENTER – так работает функция *автозавершения.* Однако пользователь может самостоятельно выбрать необходимый элемент текста из списка с иерархической структурой — это функция *автотекста.* Список элементов автотекста открывается с помощью панели инструментов Автотекст (Вид>Панели инструментов>Автотекст).

Настройку словаря автотекста выполняют в диалоговом окне Автозамена (Сервис>Автозамена>Автотекст).

Использование средства автозамены при вводе. Настройку средства Автозамена выполняют в диалоговом окне Сервис>Автозамена. Для этого надо установить флажок Заменять при вводе, ввести заменяемую комбинацию в поле Заменить, а замещающую комбинацию в поле На, после чего пополнить список автозамены щелчком на кнопке Добавить.

Ввод специальных символов. Основным средством для ввода специальных и произвольных символов, а также для закрепления их за избранными клавишами является диалоговое окно Символ (Вставка>Символ). Если символ надо вставить только один раз, достаточно щелкнуть на командной кнопке Вставить. Если предполагается многократное использование данного символа, за ним можно закрепить постоянную комбинацию клавиш (кнопка Клавиша) или создать элемент для списка Автозамена с помощью одноименной кнопки.

Режимы вставки и замены символов. Текстовый процессор предоставляет возможность выбора между двумя режимами редактирования текста: *режимом вставки* и *режимом замены.* В режиме вставки вводимый текст «раздвигает» существующий текст, а в режиме замены новые символы замещают символы предшествующего текста, находившиеся в точке ввода. Текущий режим правки текста идентифицируется на экране индикатором Замена. В режиме замены включен индикатор ЗАМ в строке состояния окна программы, в противном случае он выключен. Двойной щелчок на этом индикаторе позволяет переключать режимы. Настройка режима правки выполняется на вкладке Правка диалогового окна Параметры (Сервис > Параметры > Правка).

Если установлены флажки Режим замены и Использовать клавишу INS для вставки, правка осуществляется в режиме замены символов. Если оба эти флажка сброшены, то режим можно выбирать с помощью клавиши INSERT.

Использование Тезауруса. Тезаурус представляет собой словарь смысловых синонимов. При подготовке технической документации особую роль играют смысловые синонимы к используемым глаголам. Для выделенного слова тезаурус удобно вызывать через пункт Синонимы контекстного меню. Однако этот прием срабатывает далеко не для всех слов (преимущественно для глаголов в неопределенной форме). Общий прием вызова тезауруса состоит в использовании команды строки меню Сервис > Язык>Тезаурус.

Окно Тезаурус имеет две панели. Его интересная особенность состоит в том, что в то время как на левой панели отображаются синонимы выделенного слова, на правой панели могут отображаться синонимы к выбранному синониму, то есть поиск синонима является как бы двухуровневым. Заменяющий синоним можно выбирать как на левой панели, так и на правой. Замена производится щелчком на командной кнопке Заменить. Кроме синонимов в некоторых случаях тезаурус позволяет находить *антонимы* слов и *связанные* (как правило, однокоренные) слова.

Средства автоматизации проверки правописания. Средства автоматизации проверки правописания включают средства проверки орфографии и грамматики. Текстовый процессор позволяет реализовать два режима проверки правописания – *автоматический* и *командный.*

Для работы в автоматическом режиме надо установить флажки Автоматически проверять орфографию и Автоматически проверять грамматику на вкладке Правописание диалогового окна Параметры (Сервис>Параметры>Правописание).

В командном режиме проверка правописания выполняется независимо от установки элементов управления на вкладке Сервис > Параметры > Правописание. Запуск средства проверки выполняют командой Сервис > Правописание. Проверка начинается от начала документа и продолжается до появления первой ошибки. В тех случаях, когда пользователь отказывается от предлагаемых исправлений и дает команду Пропустить, в документе накапливается список пропускаемых слов, то есть слов и выражений, не подлежащих проверке. Для того чтобы очистить этот список и начать проверку заново, используют командную кнопку Сервис>Параметры >Правописание>Повторная проверка.

5.2.3. Форматирование текста

Форматирование текста осуществляется средствами меню Формат или панели Форматирование. Основные приемы форматирования включают:

- выбор и изменение гарнитуры шрифта;
- управление размером шрифта;
- управление начертанием и цветом шрифта;

- управление методом выравнивания;
- создание маркированных и нумерованных списков (в том числе многоуровневых);
- управление параметрами абзаца.

Настройку шрифта выполняют в диалоговом окне Шрифт (Формат > Шрифт).

На вкладке Шрифт выбирают:

- гарнитуру шрифта;
- его размер (измеряется в полиграфических пунктах);
- вариант начертания;
- цвет символов;
- наличие подчеркивания;
- характер видоизменения.

Большинство гарнитур шрифтов являются пропорциональными. Это означает, что и ширина отдельных символов, и расстояние между соседними символами не являются постоянными величинами и динамически меняются так, чтобы сопряжение символов было наиболее благоприятным для чтения.

Особую группу представляют так называемые моноширинные шрифты. В них каждый символ вместе с окаймляющими его интервалами имеет строго определенную ширину. Такие шрифты применяют в тех случаях, когда надо имитировать шрифт пишущей машинки, а также при вводе текстов, представляющих листинги программ. Характерными представителями таких шрифтов являются шрифты семейства Courier.

Использование средств управления шрифтом (выбор размера, начертания, подчеркивания и других видоизменений) определяется стилевым решением документа, которое задает заказчик. Приступая к выполнения заданию, следует выяснить, какие стилевые решения уже существуют и Вы должны их использовать, каковы ограничения на использование средств оформления и форматирования.

Настройка метода выравнивания. Все последние версии текстового процессора Microsoft Word поддерживают четыре типа выравнивания:

- по левому краю;
- по центру;
- по правому краю;
- по ширине.

Выбор метода выполняют соответствующими кнопками панели инструментов Форматирование или из раскрывающегося списка Формат > Абзац > Отступы и интервалы > Выравнивание.

Настройка параметров абзаца. Кроме режима выравнивания настраиваются следующие параметры абзаца:

- величина отступа слева (от левого поля);
- величина отступа справа (от правого поля);

- величина отступа первой строки абзаца («красная строка»);
- величина интервала (отбивки между абзацами) перед абзацем и после него.

Средства создания маркированных и нумерованных списков. Для создания нумерованных и маркированных списков нужно сначала выполнить настройку, затем вход в список и, наконец, выход из него. Настройку выполняют в диалоговом окне Список, открываемом командой Формат > Список. Данное окно имеет три вкладки: Маркированный список, Нумерованный список и Многоуровневый список. В качестве элементов управления здесь представлены образцы оформления списков. Для выбора нужного достаточно щелкнуть на избранном образце.

Для завершения маркированного или нумерованного списка и выхода из режима его создания достаточно по завершении ввода последней строки дважды нажать клавишу Enter.

5.3. Сохранение документа

Для сохранения документа на жестком диске следует выполнить команду Файл>Сохранить или щелкнуть на кнопке Сохранить Стандартной панели инструментов, или нажать CTRL+S. Появится диалоговое окно сохранения документа. В текстовом поле Имя файла введите имя, которое вы хотите присвоить своему документу. Имя может иметь до 256 символов. Если нужно сохранить документ в другой папке или на другом диске, выберите их из раскрывающегося списка Папка. Щелкните на кнопке Сохранить. Документ будет сохранен на диске, и имя, которое ему присвоено, появится в строке заголовка окна MS Word.

Редактируемый документ нужно периодически сохранять, чтобы минимизировать потерю данных в случае отключения электроэнергии или возникновения других проблем с системой.

5.4. Приемы и средства автоматизации разработки документов

Наиболее общими средствами автоматизации разработки и оформления документов являются стили оформления абзацев и шаблоны документов.

5.4.1. Работа со стилями

Абзац — элементарный элемент оформления любого документа. Каждый заголовок документа тоже рассматривается как отдельный абзац. Выше мы видели, что в меню Формат > Абзац имеется немало различных элементов управления, и выполнять их настройку для каждого абзаца отдельно — неэффективная и утомительная задача. Она автоматизируется путем использования понятия стиль.

Стиль оформления — это именованная совокупность настроек параметров шрифта, абзаца, языка и некоторых элементов оформления абзацев (линий и рамок). Благодаря использованию стилей обеспечивается простота

форматирования абзацев и заголовков текста, а также единство их оформления в рамках всего документа.

Работа со стилями состоит в создании, настройке и использовании стилей. Некоторое количество стандартных стилей присутствует в настройке программы по умолчанию, сразу после ее установки. Их используют путем выбора нужного стиля из раскрывающегося списка на панели управления Форматирование.

Настройка стиля. Настройку стиля выполняют в диалоговом окне Стиль (Формат > Стиль). Настраиваемый стиль выбирают в списке Стили (при этом на панелях Абзац и Знаки отображаются образцы применения данного стиля). Для изменения стиля служит командная кнопка Изменить, открывающая диалоговое окно Изменение стиля. Каждый из компонентов стиля настраивается в отдельном диалоговом окне. Выбор компонента выполняют в меню, открываемом с помощью командной кнопки Формат.

Создание стиля. Для создания стиля служит командная кнопка Создать в диалоговом окне Стиль (Формат>Стиль) – она открывает диалоговое окно Создание стиля.

В данном окне следует:

- ввести название нового стиля в поле Имя;
- выбрать тип стиля (стиль абзаца или знаковый стиль);
- выбрать стиль, на котором основан новый стиль;
- указать стиль следующего абзаца;
- приступить к настройке элементов стиля щелчком на кнопке Формат.

Важной чертой программы является принцип наследования стилей. Он состоит в том, что любой стиль может быть основан на каком-то из существующих стилей. Это позволяет, во-первых, сократить до минимума настройку стиля, сосредоточившись только на его отличиях от базового, а во-вторых, обеспечить принцип единства оформления всего документа в целом.

Стиль следующего абзаца указывают для обеспечения автоматического применения стиля к следующему абзацу, после того как предыдущий абзац закрывается клавишей Enter.

5.4.2. Шаблоны

Совокупность удачных стилевых настроек сохраняется вместе с готовым документом, но желательно иметь средство, позволяющее сохранить их и вне документа. Тогда их можно использовать для подготовки новых документов. Такое средство есть — это шаблоны, причем некоторое количество универсальных шаблонов поставляется вместе с текстовым процессором и устанавливается на компьютере вместе с ним.

По своей сути, шаблоны – это заготовки будущих документов.

Использование шаблона для создания документа. По команде Файл > Создать открывается диалоговое окно Создание документа, в котором можно выбрать шаблон, на базе которого документ будет разрабатываться. В этом

случае документ сразу получает несколько готовых стилей оформления, содержавшихся в шаблоне.

Изменение шаблона готового документа выполняется с помощью диалогового окна Шаблоны и настройки (Сервис > Шаблоны и настройки). Для смены текущего шаблона следует использовать кнопку Присоединить и в открывшемся диалоговом окне Присоединение шаблона выбрать нужный шаблон в папке C:\Program Files\Microsoft Office\Шаблоны.

Создание нового шаблона на базе имеющегося шаблона осуществляется по команде Файл>Создать. Открывается диалоговое окно Создание документа, в котором следует включить переключатель Шаблон и выбрать стандартный шаблон, на базе которого он создается. После настройки стилей и редактирования содержания выполняется сохранение шаблона командой Сохранить как с включением пункта Шаблон документа в поле Тип файла.

Если готовый документ может быть использован в качестве заготовки для создания других документов, его целесообразно сохранить как шаблон. Командой Файл>Открыть открывают готовый документ, в нем правят содержание и настраивают стили, а потом сохраняют документ как шаблон командой Сохранить как с включением пункта Шаблон документа в поле Тип файла.

5.5. Внедрение объектов, созданных другими приложениями

В Ms Word имеется возможность вставлять объекты, созданные как с помощью самого редактора (формулы, рисунки, таблицы), так и с помощью других приложений Windows. Для этого используются команды меню Вставка > Рисунок или Объект или Файл.

5.6. Задание

С помощью текстового редактора MS Word создать документ, в котором определить стили «основной текст», «основной текст с отступом первой строки», заголовок 1, 2, 3 с настройками, удовлетворяющими требованиям, предъявляемых к оформлению отчетов по лабораторным работам (приложение 2). С использованием этого шаблона и определенных в нем стилей произвести форматирование отчётов по лабораторным работам 1 – 3.

5.7. Порядок выполнения работы

1) Ознакомиться с теоретическим материалом пп. 1-5 описания лабораторной работы.
2) Запустить приложение MS Word.
3) Создать новый документ, в котором определить указанные в задании стили в соответствии с требованиям, предъявляемых к оформлению отчетов по лабораторным работам (приложение 2).
4) Сохранить его в своей папке и как документ MS Word, и как шаблон документа.

5) На основе созданного шаблона создать новый документ, скопировать в него отчеты по лабораторным работам 1 – 3.
6) С использованием определенных в документе (шаблоне) стилей произвести форматирование отчётов по лабораторным работам 1 – 3 в соответствии с требованиями представленными в приложении 2.
7) Отчеты сохранить в тех же файлах с именами, образованным из инициалов студента (три латинские буквы) и номера лабораторной работы.
8) При сдаче лабораторной работы необходимо продемонстрировать оформленные отчеты на ПК и ответить на контрольные вопросы.

5.8. Контрольные вопросы

1) Как изменить в тексте гарнитуру шрифта, его размер и начертание.
2) Как скопировать фрагмент текста из одного места в другое.
3) Как вставить в документ в MS Word произвольную картинку из файла, центрировать положение картинки на странице.
4) Каким образом создать элемент автозамены, заменяющий в тексте инициалы пользователя на полное имя (фамилия, имя и отчество).
5) Как сохранить текст документа в файл с другим именем.
6) Как закрыть приложение MS Word.
7) Из каких основных элементов состоит текстовый документ.
8) Какими параметрами характеризуется абзац текста.
9) Что такое стиль, шаблон и как их можно создать, изменить?
10) Как создавать, открывать и сохранять файлы в Microsoft Word.
11) Какие операции составляют процесс редактирования текста и как они выполняются в Microsoft Word.
12) Способы выделения фрагментов текста в Microsoft Word.
13) Способы копирования, перемещения и удаления фрагментов текста в Microsoft Word.
14) Что такое буфер обмена и для чего он используется.

ОБРАБОТКА ГРАФИЧЕСКОЙ ИНФОРМАЦИИ НА ПК

Практическая работа № 6

Создание иллюстраций в редакторах Paint и Microsoft Word

Цель работы: ознакомление с графическим редактором Paint и панелью рисования в текстовом редакторе Word, овладение навыками работы в этих редакторах.

6.1. Общие сведения

Графическими называют редакторы, предназначенные для создания и редактирования изображений (рисунков).

Программа Paint – простейший однооконный графический редактор, который, тем не менее, позволяет создать достаточно сложный рисунок, эскиз эмблемы, фирменного знака. Эта программа, в силу своей простоты, остается необходимым компонентом операционной системы Windows и доступна миллионам людей. Не разобравшись с принципами управления этой программой, трудно осваивать другие, более мощные средства работы с графикой. Многие приемы работы с Paint могут пригодиться и в более сложных программах.

Программа Paint является редактором растровой графики. Кроме редакторов растровой графики существуют еще редакторы векторной графики. Приемы и методы работы с этими двумя различными классами программ различны. В растровой графике мельчайшим элементом изображения является точка, которой на экране соответствует экранная точка (пиксел). Мельчайшим элементом векторной графики является линия.

Более сложные графические объекты позволяет создавать панель инструментов Рисование в текстовом редакторе Word. Это редактор векторной графики. С помощью него можно рисовать, а также вставлять в документ готовые типовые изображения.

6.2. Графический редактор Paint

6.2.1. Окно программы Paint

Программа запускается командой Пуск>Программы>Стандартные> Paint.

Рабочее окно программы Paint представлено на рис 6.1.

В состав его элементов управления кроме строки меню, входит панель инструментов, палитра настройки инструмента и цветовая палитра.

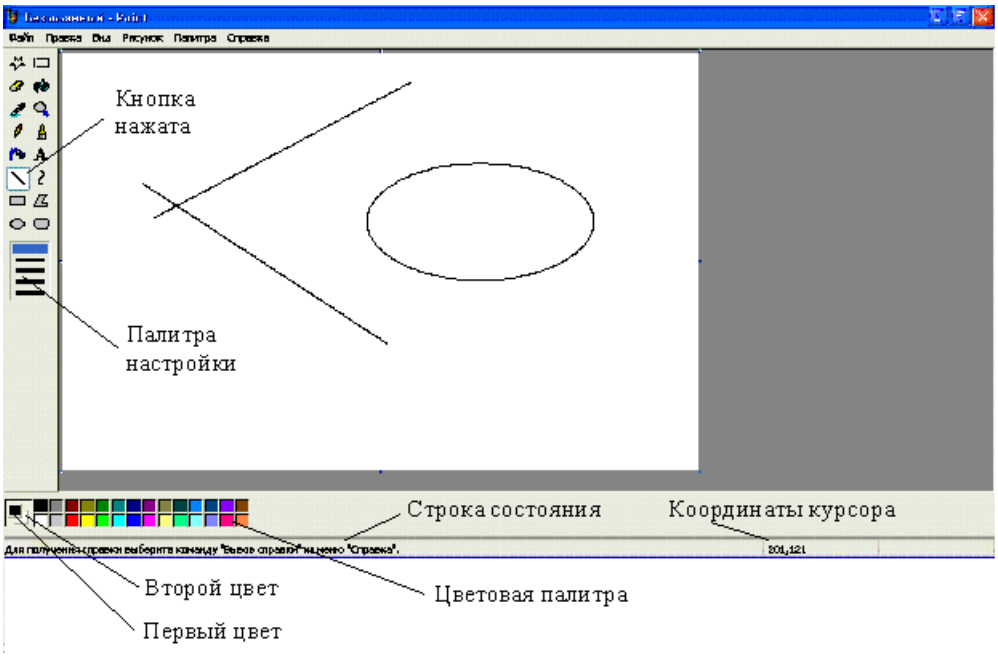


Рис. 6.1. Окно графического редактора Paint

Кнопки панели инструментов служат для вызова чертежно-графических инструментов. На палитре настройки можно выбрать параметры инструмента (толщину линии, форму оттиска, метод заполнения фигуры и т. п.). Вид палитры будет меняться в зависимости от выбранного инструмента.

Элементы цветовой палитры служат для выбора основного цвета (щелчком левой кнопки мыши) и фонового цвета (щелчком правой кнопки). Все, что будет рисоваться левой кнопкой мыши, рисуется основным цветом, а все, что правой – фоновым цветом.

Помимо обычной всплывающей подсказки по кнопкам, в строке состояния (в нижней части окна) появляется развернутая характеристика инструмента, к которому подвели курсор.

По краям самого рисунка стоят черные квадратики-узелки. Если взяться за них (курсор должен превратиться в двустороннюю стрелку) и потащить, то можно изменить размеры поля, на котором находится изображение. При этом в строке состояния можно прочесть размеры этого поля: первая цифра – ширина в пикселях, вторая – высота.

6.2.2. Задание размера рабочей области

Перед началом работы следует хотя бы приблизительно задать размер будущего рисунка. Размеры задают в диалоговом окне Атрибуты, которое открывается командой Рисунок > Атрибуты. Размеры вводят в поля Ширина и Высота. До ввода размеров следует выбрать принятую единицу измерения с помощью одного из переключателей:

Дюймы;

См (сантиметры);

Точки (пикселы).

В России не принято задавать размеры документов в дюймах. Размер в сантиметрах задают в тех случаях, когда предполагается вывод работы на печатающее устройство (принтер) или встраивание его на страницу с текстовым документом. В тех случаях, когда рисунок предназначен для воспроизведения на экране, в качестве единицы измерения выбирают точки (пикселы). Так, например, если рисунок готовится для использования в качестве фона Рабочего стола, его размеры следует принять равными величине экранного разрешения, установленного на компьютере (640x480; 800x600; 1024x768 точек и т. д.).

6.2.3. Основные чертежно-графические инструменты

Все инструменты, кроме Ластика, выполняют рисование основным цветом (при использовании левой кнопки мыши) или фоновым цветом (при использовании правой кнопки мыши). Основной цвет задается щелчком левой кнопки мыши в палитре красок, фоновый цвет – щелчком правой кнопки.

Ластик стирает изображение, заменяя его фоновым цветом.

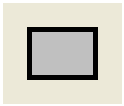
Инструмент Линия предназначен для вычерчивания прямых. Толщину линии выбирают в палитре настройки. Линии вычерчивают методом протягивания мыши с нажатой левой клавишей. Чтобы линия получилась строго вертикальной, горизонтальной или наклонной под углом 45°, при вычерчивании линии следует держать нажатой клавишу SHIFT.

Инструмент Карандаш предназначен для рисования произвольных линий. Толщина линии - всегда 1 пиксель.

Инструмент Кривая служит для построения гладких кривых линий. Толщину предварительно выбирают в палитре настройки. Концы линии ставятся так же, как у прямой, но после этого линию можно дважды деформировать. Сначала взять за какой-то участок ближе к левому концу линии и потащить вверх или вниз, тем самым, изогнув ее, а потом – ближе к правому концу и снова изогнуть. Лишь после этого кривая зафиксируется.

Инструмент Кисть можно использовать для свободного рисования произвольных кривых, как Карандаш, но чаще его используют для рисования методом набивки. Сначала выбирают форму кисти в палитре настройки, а потом щелчками левой или правой кнопки мыши наносят оттиски на рисунок без протягивания мыши.

Инструмент Распылитель используют как для свободного рисования, так и для рисования методом набивки. Форму пятна выбирают в палитре настройки.



Инструмент Прямоугольник применяют для рисования прямоугольных фигур. Рисование выполняется протягиванием мыши. В палитре настройки можно выбрать метод заполнения прямоугольника. Возможны три варианта: Без заполнения (рисуется только рамка), Заполнение фоновым цветом и Заполнение основным цветом. Если при создании прямоугольника держать нажатой клавишу SHIFT, образуется правильная фигура. Для прямоугольника правильной фигурой является квадрат.

Аналогичный инструмент Скругленный прямоугольник действует точно так же, но при этом получается прямоугольник со скругленными углами.

Инструмент Многоугольник предназначен для рисования произвольных многоугольников. Рисование выполняют серией последовательных щелчков с протягиванием, Если конечная точка многоугольника совпадает с начальной, многоугольник может быть автоматически залит краской в соответствии с тем вариантом заполнения, который выбран в палитре настройки.

Инструмент Эллипс служит для изображения эллипсов и окружностей. Окружность – это частный случай «правильного» эллипса. Она получается при рисовании с нажатой клавишей SHIFT.

Инструмент Заливка служит для заполнения замкнутых контуров основным или фоновым цветом. Заполнение основным цветом производится щелчком левой кнопки мыши, а заполнение фоновым цветом – щелчком правой кнопки. Если контур не замкнут, инструмент работает неправильно. В этом случае ошибочное действие надо немедленно отменить командой Правка > Отменить или комбинацией клавиш CTRL+Z.

Комбинацию CTRL+Z следует запомнить. Она отменяет последнее действие в большинстве приложений Windows и является удобным общесистемным приемом.

Инструмент Выбор цветов (во многих графических программах он называется «пипеткой») позволяет точно выбрать основной или дополнительный цвет не из палитры красок, а непосредственно из рисунка. Это важно, когда надо обеспечить одинаковость цвета в разных областях. После

выбора инструмента наводят указатель на участок рисунка нужным цветом и щелкают кнопкой мыши. Если произошел щелчок левой кнопкой, текущий цвет становится основным, а если правой – фоновым.


Инструменты Выделение и Выделение произвольной области предназначены для работы с выделенными областями. Действуют они одинаково, разница лишь в том, что инструмент Выделение формирует не произвольную, а прямоугольную выделенную область. С выделенной областью можно поступать так, как это принято во всех приложениях Windows: ее можно переместить, удалить клавишей DELETE, скопировать в буфер обмена (CTRL+C), вырезать в буфер обмена (CTRL+X) и вставить из буфера обмена (CTRL+V). Для операций копирования, вставки, удаления можно также пользоваться правой кнопкой мыши или меню Правка.

Прием копирования и вставки выделенной области применяют для размножения повторяющихся фрагментов рисунка. Фрагмент, вставленный из буфера обмена, можно немедленно перетащить в нужное место рисунка.

При размножении выделенных областей возможны два режима вставки: с сохранением фоновой графики или без нее (точки фонового цвета игнорируются). Переключение режима выполняют в палитре параметров.

Масштаб. Для точной доводки рисунка иногда необходимо увеличить масштаб просмотра. Максимальное увеличение – восьмикратное. Для изменения масштаба служит команда Вид>Масштаб. То же можно сделать с помощью инструмента Масштаб, в этом случае величину масштаба выбирают в палитре параметров.

В режиме восьмикратного увеличения на рисунок можно наложить вспомогательную сетку. Каждая ячейка этой сетки представляет собой одну увеличенную точку изображения. В этом режиме удобно редактировать изображение по отдельным точкам. Наложение сетки выполняют командой Вид >Масштаб > Показать сетку.

6.2.4. Трансформация изображений

Трансформациями называют автоматические изменения формы, расположения или размеров графических объектов. В программе Paint не слишком много инструментов трансформации, но все-таки они есть. Их можно найти в меню Рисунок.

Команда Рисунок>Отразить/повернуть вызывает диалоговое окно Отражение и поворот, содержащее элементы управления для симметричного отображения рисунка относительно вертикальной или горизонтальной оси симметрии, а также для поворота на фиксированный угол, кратный 90°.

Команда Рисунок > Растянуть/наклонить вызывает диалоговое окно Растяжение и наклон. Его элементы управления позволяют растянуть рисунок по горизонтали и вертикали или наклонить относительно горизонтальной

или вертикальной оси. Параметры растяжения задают в процентах, а параметры наклона – в угловых градусах.

Команда Рисунок>Обратить цвета действует как переключатель. При использовании этой команды цвет каждой точки изображения меняется на «противоположный». В данном случае мы назвали «противоположным» тот цвет, который дополняет данный цвет до белого.

6.2.5. Ввод текста

Программа Paint – графический редактор и не предназначена для работы с текстом. Поэтому ввод текста в этой программе является исключением, а не правилом. Поскольку редактор относится к растровым, он строит изображение по точкам, следовательно, текст после ввода станет «рисунком» и будет состоять из достаточно крупных точек растра. Поэтому избегайте использования мелких символов, которые смотрятся весьма неопрятно. Рассматривайте режим работы с текстом в программе Paint только для создания кратких и крупных заголовков, не забывая, конечно, о том, что размер заголовка должен быть согласован с размером самого изображения.

Для ввода текста используют инструмент Надпись. Выбрав инструмент, щелкните на рисунке примерно там, где надпись должна начинаться, – на рисунке откроется поле ввода. В это поле вводится текст с клавиатуры. О типе шрифта, его размере и начертании заботиться пока не надо – главное набрать текст без ошибок, а остальное все можно изменить позже. Размер поля ввода можно изменить, потянув за маркеры области ввода – небольшие прямоугольные метки, расположенные по сторонам и углам области ввода. Разрешается переместить поле ввода текста, осторожно взявшись за пунктирную границу.

Закончив ввод текста, вызовите панель атрибутов текста – это делается командой Вид>Панель атрибутов текста. Элементами управления этой панели можно выбрать форму шрифта, его начертание и размер.

Как только текст зафиксируется, а это происходит при смене инструмента или щелчке по рабочему листу, работа с ним становится невозможной. Войти в поле ввода этого текста уже нельзя.

6.3. Встроенный графический редактор Microsoft Word

Чтобы отобразить панель инструментов Рисование (рис. 6.2), нужно в стандартной панели инструментов щелкнуть на кнопке Рисование, либо открыть меню Вид>Панели инструментов, выбрать из списка Рисование.

Рис. 6.2. Панель инструментов Рисование в MS Word

Инструмент Линия позволяет рисовать прямые линии протягиванием мыши с нажатой левой кнопкой. Если держать нажатой клавишу SHIFT, то линия рисуется строго вертикальная, горизонтальная или под правильным углом ($30^\circ, 45^\circ, 60^\circ$). По умолчанию линия идет с шагом 2 мм, но если вы будете рисовать ее с нажатой клавишей ALT, то шаг будет отключен. Если требуется нарисовать несколько линий, нужно дважды щелкнуть на кнопке Линия, чтобы режим рисования линий оставался включенным после рисования очередного отрезка. Чтобы выйти из этого режима, нужно снова щелкнуть на кнопке Линия или на любой другой кнопке. Эти замечания относятся также и к рисованию других фигур.

Чтобы изменить цвет, толщину линии, и тип штриха можно воспользоваться следующими кнопками:

Цвет,

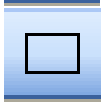
Толщина линии,

Тип штриха.

Для этого нужно выделить объект щелчком левой кнопки мыши, вызвать меню и выбрать нужный тип.

Кнопка Стрелка служит для рисования стрелок.

Изменить направление и тип стрелки можно, воспользовавшись меню Стрелки.



Кнопка Прямоугольник позволяет нарисовать прямоугольник. Для этого нужно щелкнуть левой кнопкой мыши в месте одного из углов будущей фигуры и вести мышь по диагонали по направлению к противоположному углу прямоугольника. Если держать нажатой клавишу Shift, получится квадрат.

Чтобы нарисовать окружность или эллипс следует воспользоваться кнопкой Овал. Щелкнув левой кнопкой мыши, нужно вести мышь в нужном направлении в зависимости от формы эллипса. Окружность рисуется с нажатой клавишей Shift.

Можно рисовать и другие фигуры: линии, фигурные стрелки, элементы блок-схемы, звезды, ленты, выносные линии и т.д. Список имеющихся фигур выводится щелчком на кнопке Автофигуры (рис. 6.3).

Для всех фигур, помимо вида и цвета линии, можно задать также цвет заливки. Если выбрать в списке, который открывает нам эта кнопка, строку

Другие способы заливки, то, кроме плоского цвета, вы получите возможность закрашивать фигуры с переходами цветов (градиентная заливка), а также текстурами, узорами и рисунками.

Одно из существенных преимуществ векторных графических редакторов перед Paint состоит в том, что каждый элемент в любое время доступен

изменению, удалению, перетаскиванию, повороту, уменьшению, увеличению и т.д.

Для изменения размера фигуры в двух направлениях достаточно выделить фигуру, щелкнув левой кнопкой мыши, и потянуть за угловой размерный маркер. Размер будет изменяться строго пропорционально, если при этом держать нажатой клавишу SHIFT. Чтобы изменить размер в одном направлении, нужно потянуть за размерный маркер, расположенный на одной из четырех сторон контура выделения.

Чтобы перемещать графический объект по листу, следует выделенный объект тащить мышью (взявшись за контур, а не за размерные маркеры) в нужном направлении. Шаг передвижения по умолчанию – 2 мм, но его можно отключить, если держать нажатой клавишу SHIFT.

Графический объект может состоять из одной простой фигуры или нескольких. Для того, чтобы объект, состоящий из нескольких фигур, стал единым целым, нужно сгруппировать фигуры. Для этого следует последовательно выделить все фигуры щелчками мышью, держа нажатой клавишу Shift. Есть и другой способ, когда элементов много.

Следует нажать на кнопку Выбор объектов, растянуть прямоугольную рамочку левой кнопкой мыши из одного угла в другой (противоположный). Все рисованные объекты, которые попадут в рамочку целиком, выделятся. Затем щелкнуть на кнопке Действия (в Word 2003 – Рисование) и выбрать из списка Группировать (рис. 6.4). Благодаря этому, можно будет передвигать по листу сложный рисунок, поворачивать, деформировать, копировать, вставлять и т.д. Обратное действие можно произвести, выделив сгруппированную фигуру, нажав Действия и выбрав Разгруппировать.

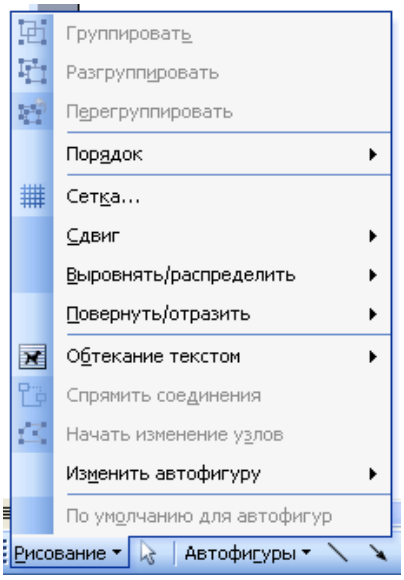


Рис. 6.4. Меню Рисование (Действия) в MS

Разрешена также многоуровневая группировка. Например, сначала можно сгруппировать один вид элементов, затем – другой, а после этого объединить эти две группы в единый рисунок. Тогда команда Разгруппировать разобьет рисунок сначала на две группы, а затем – на отдельные элементы.

Чтобы снять выделение с элемента, нужно щелкнуть мышью где-нибудь в стороне от него. Чтобы снять выделение с одного из выделенных объектов, следует щелкнуть по нему с клавишей CTRL.

Следующее подменю в меню Действия называется Порядок. Имеется в виду порядок расположения фигур относительно друг друга и относительно текста. Команды На передний план, На задний план, Переместить вперед, Переместить назад и т.д. позволят все это определить, создавая объекты довольно сложной формы.

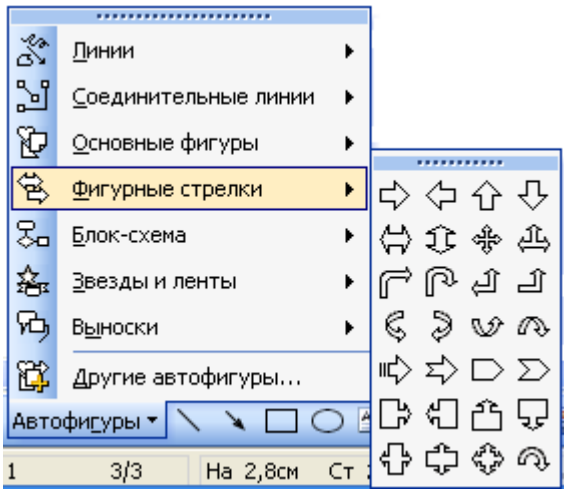


Рис. 6.3. Меню Автофигуры

Для вращения, поворота графического объекта и других операций можно также воспользоваться кнопкой Действия, выбрав соответствующую операцию.

Например, кнопка Свободное вращение позволит повернуть выделенную фигуру на произвольный угол. Есть возможность задать угол поворота точно. Нужно открыть в строке меню Формат>Автофигура. На странице Размер есть для этого строка Поворот.

Команда Сетка в меню Действия предназначена для тонких работ. Открыв эту команду, можно сделать сетку видимой, задать свои параметры – шаг, привязку и т.д.

Эта кнопка на панели инструментов называется Добавить объект WordArt. С ее помощью на страницу вставляется фигурный текст.

Следующие две кнопки предназначены для вставки картинок из коллекции клипов и рисунков (сканированных или выполненных в других графических редакторах).

Нарисованный, а также вставленный готовый графический объект можно оформить текстовыми пояснениями. Кнопка Надпись ставит на странице рамочку, в которую

можно вводить текст. Для надписи можно задать любое форматирование, а для самой рамочки – вид, толщину и цвет линии и заливку.

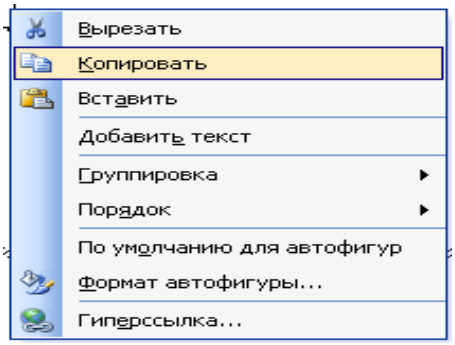


Рис. 6.5. Контекстное меню

У всех линий и фигур есть контекстное меню (рис. 6.5). Многие операции по их изменению, в том числе и перечисленные выше, можно производить, пользуясь правой кнопкой мыши. Так открывается контекстное меню. В нем можно найти такие команды, как Вырезать, Копировать, Вставить, Группировать, Порядок, Формат автофигуры и т.д.

В частности, команда Формат автофигуры или Формат рисунка (если рисунок вставлен из другого редактора) позволяет помимо выбора цвета, толщины и вида линии еще и правильно расположить рисунок на странице с текстом. В открывающемся окне этой команды на странице Положение можно выбрать вид расположения рисунка: В тексте, Вокруг рамки, По контуру, За текстом, Перед текстом; а также выравнивание относительно текста: По центру, По правому или левому краю. В нижней части окна имеется кнопка Дополнительно. В открывающемся окне Дополнительная разметка есть две страницы – Положение рисунка и Обтекание текстом, которые позволяют более точно расположить рисунок (рис. 6.6).

На странице Обтекание текстом можно цифрами задать размер полей вокруг рисунка, а также выбрать, каким способом текст будет обтекать рисунок.

6.4. Порядок выполнения работы

1) Ознакомиться с теоретическим материалом пп. 1-3 описания лабораторной работы.
2) Для указанного варианта задания из п.6 составить и нарисовать в редакторе Microsoft Word блок схему алгоритма для решения задачи.
3) Скопировать рисунок в окно графического редактора Paint. Сохранить рисунок в отдельном файле в формате jpeg. Подготовить отчёт по лабораторной работе в соответствии с требованиями, представленными в приложении 2.
4) При сдаче лабораторной работы необходимо продемонстрировать оформленный отчет на ПК и ответить на контрольные вопросы по теме работы.
5) Сопоставить блок-схему алгоритма с имеющимися в приложении 4.

6.5. Контрольные вопросы

1) Что такое графический редактор? Какие графические редакторы Вы знаете?
2) Чем отличаются редакторы растровой графики от редакторов векторной графики?
3) Как запускается Paint? Какие элементы управления отображены в окне программы Paint?
4) Как выбрать основной и фоновый цвет? Как рисовать основным и фоновым цветом?
5) Как отобразить панель инструментов Рисование в Word?
6) Какие, на ваш взгляд, существуют недостатки и неудобства при работе в графическом редакторе Paint?
7) Как добиться рисования линии строго вертикальной или горизонтальной?
8) Как нарисовать окружность, квадрат?
9) Какие приемы можно использовать для более точного рисования в Paint и графическом редакторе Word?

10) В каком из этих двух редакторов удобнее вводить текст? Как это делать?
11) Как изменить цвет, толщину, вид линии, направление стрелки?
12) Как сгруппировать, разгруппировать несколько объектов?
13) Как правильно расположить рисунок на странице с текстом?
14) Как вы думаете, какой графический редактор более пригоден для рисования схем, блок-схем, небольших чертежей?

6.6. Варианты заданий

1) Даны два действительных различных числа x и y. Найти наибольшее из них.
2) Даны числа a и b. Найти z, зная, что z=x2+y; x=a+b; y=7x+ab.
3) Вычислить y=x2+1 для x=1, x=1.2, x=1.4, …, x=10.
4), 5) Вычислить S=1+2+3 … 100.

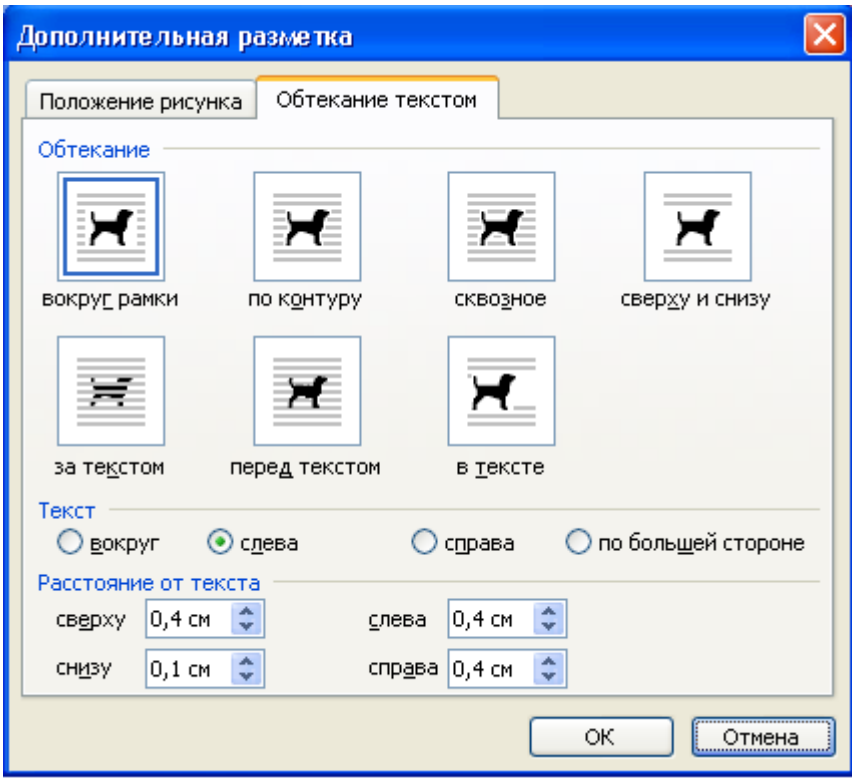


Рис.6.6. Виды обтекания текстом в редакторе Word 2003

6) Найти сумму четных натуральных чисел, не превосходящих 150.
7) Даны действительные числа a и b. Найти x=y+z, где y=a+b, z=5y+3b.
8) Даны числа a и b, найти разность между большим и меньшим.
9) Сколько корней имеет уравнение ax=b.
10) Вычислить y=x+z, если x изменяется от 1 до 15 с шагом hx=1.1, z изменяется от 15 до 1000 с шагом hz=1.5.

11) Даны четыре натуральных числа. Найти квадратные корни суммы и произведения этих чисел.

12) Для данного вещественного числа x вычислить значение функции y:

$$y = \begin{cases} 5x - 3, & \text{если } x \geq 5, \\ 3x, & \text{если } -1 \leq x < 5, \\ \dfrac{1}{x}, & \text{если } x < -1. \end{cases}$$

13) Имеет ли уравнение $ax^2+bx+c=0$ вещественные корни?

14) Вычислить y=x+z, если x изменяется от 1 до 15 с шагом hx=1.2, z изменяется от 15 до 200 с шагом hz=2.5.

15) Даны два числа: a и b. Найти наибольший общий делитель.

РЕШЕНИЕ ВЫЧИСЛИТЕЛЬНЫХ ЗАДАЧ В C++Builder

Лабораторная работа №7

Нахождение корней квадратного уравнения

Цель занятия – знакомство с языком C++, средой разработки C++ Builder, разработка программы решения квадратного уравнения на языке C++.

7.1. Общие сведения по C++

Любая программа, написанная на языке C++, состоит из одного или нескольких блоков, именуемых функциями. Каждая функция имеет уникальное имя, которое используется для ее вызова, содержит операторы C++ и служит для выполнения одной или несколько задач. При выборе имен надо иметь ввиду, что в C++ большие и маленькие буквы различаются, т.е. *MyFun* и *MYFUN* считаются совершенно разными именами Имя функции обязательно сопровождается парой круглых скобок "(" и ")", внутри которых могут размещаться аргументы (параметры) функции. Операторы, находящиеся внутри функции, выделяются с помощью фигурных скобок "{" и "}".

```
int MyFun(int a, int b)
{
 * * *
 Операторы C++
 * * *
}
```

Функция может возвращать или не возвращать значение в вызвавшую ее программу. Если функция возвращает значение, то его тип указывается в описании функции непосредственно перед ее именем. Такие функции могут вызываться внутри операторов (например, в операторе присвоения "=") или как аргументы других функций. Если функция не возвращает значения, то в качестве возвращаемого типа указывается *void*. Вызывается такая функция как обычный оператор. Для выполнения стандартных задач в C++ входит большое количество уже готовых функций. Они образуют библиотеку стандартных функций. К функциям этой библиотеки всегда можно обратиться из программы. При обращении из программы к функциям, которые содержатся в других файлах или библиотеках, в программе необходимо поместить описание этих функций (прототипы функций). В такое описание входит тип возвращаемого функцией значения, имя функции и список аргументов (параметров) функции с указанием их типов. Для облегчения работы такие описания формируются в виде отдельных файлов, которые называют *заголовочными файлами*. Заголовочные файлы включаются в программу с помощью специальных директив.

Любая программа обязательно должна содержать одну функцию с именем *main*. Функция *main* называется *главной*, с нее всегда начинается выполнение программы.

Каждый оператор С++ заканчивается символом ";". В строке программы разрешается размещать несколько операторов. Исполнение операторов осуществляется в соответствии с порядком их размещения в программе слева направо сверху вниз. Несколько операторов можно объединить в группу, заключив их в фигурные скобки, например {x=1; y=2; z=3;}. Такая группа операторов называется *составным оператором*. Составной оператор ведет себя в программе как один отдельный оператор.

Для хранения данных в программе используются переменные. Все переменные обязательно должны быть объявлены (описаны). Объявление переменной может быть проведено в любом месте программы, но до ее первого использования. Объявление состоит из указания *типа переменной*, после которого следует *имя переменной* или *список переменных*, состоящий из нескольких имен переменных этого типа, разделенных запятыми. При описании переменной ей сразу может быть присвоено начальное значение.

Существуют следующие основные типы данных:

char – символьный тип;
int – целый тип;
float – вещественный тип;
double – вещественный тип с повышенной (двойной) точностью;
bool – логический тип (возможны два значения *true* - истина и *false* - ложь);
void – отсутствие значения (используется при описании функций).

Пример:

```
char a, b, c;
int i, j, k;
double x, y;
```

Для введения в программу различных пояснений используются комментарии. Существуют два вида комментариев: однострочные и многострочные.

Однострочные комментарии начинаются в любом месте программы с последовательности символов "//" и действуют до конца текущей строки. Многострочные комментарии также начинаются в любом месте программы с последовательности символов "/*" и завершаются "*/". Содержимое таких комментариев может занимать часть строки или несколько строк.

Пример:

```
// это однострочный комментарий
/* это многострочный
комментарий */
```

При работе транслятора C++ исходный текст программы подвергается предварительной обработке. Для этой цели используются специальные команды, которые называются *директивами* (*командными строками*) *препроцессора*. Такие команды в качестве первого символа в строке содержат "#". Примером является директива включения файлов:

#include <*имя файла*>

Данная директива заменяет текущую строку содержимым файла с указанным именем. С помощью этой директивы можно включить в текст программы содержимое любого файла или нескольких файлов.

7.2. Знакомство со средой разработки C++ Builder

C++Builder – это продукт фирмы Borland, предназначенный для быстрой разработки приложений (*RAD – rapid application development*) на языке C++. С помощью C++ Builder можно быстро и легко создавать Windows-программы на C++. Можно создавать как консольные приложения Win32, так и использовать графический интерфейс пользователя (*GUI – graphical user interface*). При создании GUI-приложений Win32 с помощью C++Builder доступна вся мощь языка C++, заключенная в среду *RAD*.

Внешний вид интерфейса программы C++Builder представлен на Рис.7.1. То, что вы видите – это *интегрированная среда разработки* (*IDE – integrated development environment*), включающая в себя четыре основных элемента. Наверху находится *главное окно*. На строке заголовка проекта находятся кнопки свертывания, восстановления и закрытия окна. Под заголовком размещается строка главного меню, которая предоставляет доступ ко всем функциям и командам среды разработки. Под главным меню располагаются кнопки быстрого запуска команд, объединенные в группы по назначению. Они позволяют получить быстрый доступ к наиболее часто используемым командам.

Работа над программой в среде C++ Builder начинается с создания нового проекта. *Проектом* называется вся группа программных файлов, которые необходимы для создания конечной исполняемой программы. Так, например, в состав проекта могут включаться файлы с текстами программ, файл ресурсов с рисунками курсоров и иконок (значков), звуковые файлы и т. п. Первоначально проект хранится в памяти компьютера, и для того чтобы сохранить его на диске, необходимо будет выполнить стандартные операции сохранения, создав при этом отдельную папку. Кроме того, интерфейс сам предложит сохранить проект, если вы решите выйти из программы или попытаетесь создать новый проект.

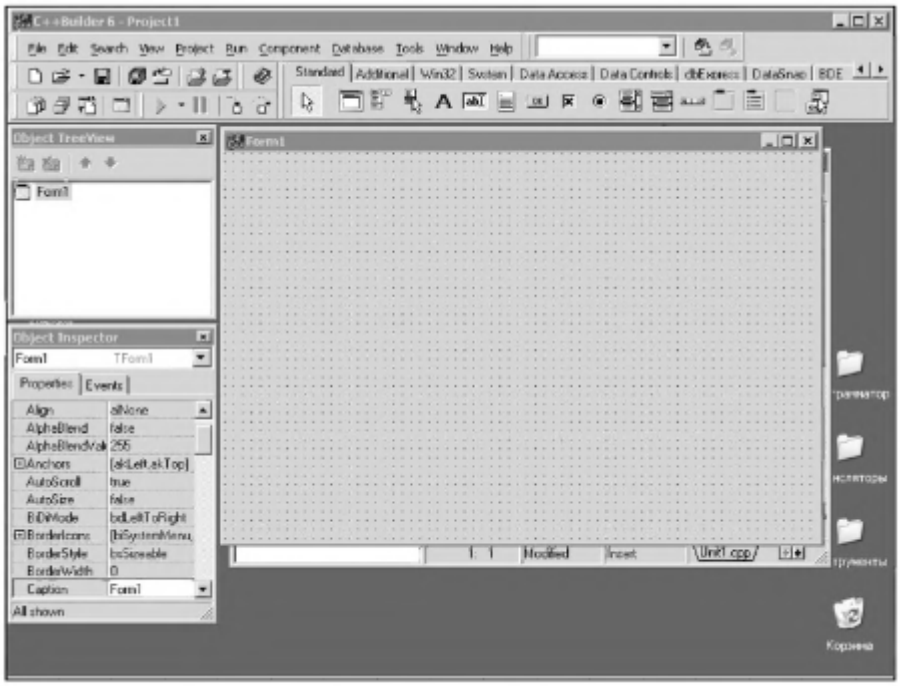


Рис.7.1. Внешний вид интерфейса Borland C++ Builder

7.2.1. Создание первой программы

Начнем работу с создания проекта для консольного приложения.

Консольное приложение – это программа, ориентированная на символьный ввод-вывод. Что делает его полезным при изучении стандартных функций ввода-вывода и классов *стандартных потоков C++*.

Чтобы создать в C++Builder консольное приложение, выполните следующие действия:

- Выберите в главном меню *File* | *New* | *Other* ...; появится многостраничная диалоговая панель *New Items* (Рис.7.2). Этот диалог является интерфейсом так называемого *хранилища объектов* C++Builder (*Object Repository*). Помимо уже имеющихся объектов вы можете сохранять в нем свои собственные формы, диалоговые панели и прочие элементы, пригодные для повторного использования в новых программах.

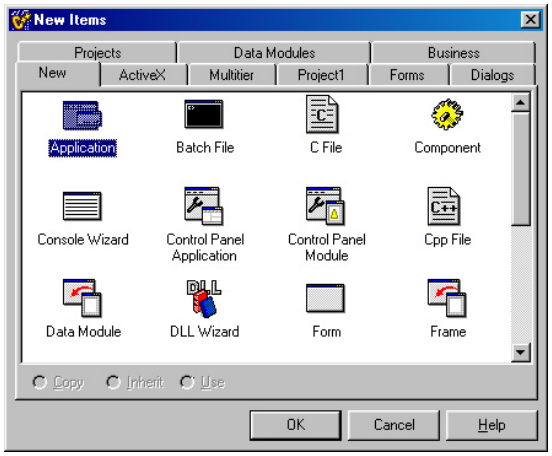


Рис. 7.2. Диалог *New Items*

- На странице *New* выберите *Console Wizard* и нажмите кнопку *OK*. В появившемся окне снимите все флажки кроме *Console Application*. C++ Builder создаст проект консольного приложения и откроет окно редактора кода с именем *Unit1.cpp*.
- В окне редактирования удалите имеющийся там код и введите текст исходной программы:

```
#include <stdio.h>
#include <conio.h>              /* директивы подключения заголовочных
                                   файлов stdio.h и conio.h, содержащих
                                   описание функций printf и getch
                                   из библиотеки стандартных функций */

void main()    /* главная функция программы */
{
 printf("Hello, World!");                  /* стандартная функция вывода
                                              на печать */
 getch();              /* стандартная функция ввода символов с помощью
                          клавиатуры (для формирования паузы в работе
                          программы)*/
}
```

- Выберите в главном меню *File* | *Save Project as*... для сохранения файлов проекта в отдельном каталоге (папке), а затем раздел и *File* | *Save as*... – для сохранения модуля с программой. Назовите проект *Prog1.bpr*, а модуль – *Main.cpp*. Команда меню *File* | *Save All* позволяет сразу сохранить файл проекта и файл модуля.
- Для трансляции и создания исполняемого кода программы необходимо выбрать в меню *Project* | *Make Prog1*.
- Запуск программы осуществляется с помощью раздела меню *Run* | *Run*, либо с помощью кнопки быстрого запуска  . В этом случае при необходимости будет автоматически проведена трансляция программы и запуск ее на исполнение.

7.2.2. Отладка программы:

После трансляции программы и исправления синтаксических ошибок проводится отладка программы. Для того используются возможности встроенного в среду C++ Builder отладчика:

- выполнение программы до места расположения курсора – команда меню *Run* | *Run to Cursor* или "горячая" клавиша [F4];

- пошаговое выполнение программы (по одному оператору за каждый шаг) без захода внутрь содержимого функций – команда меню *Run* | *Step Over* или "горячая" клавиша [F8];

- пошаговое выполнение программы (по одному оператору за каждый шаг) с заходом внутрь содержимого функций – команда меню *Run* | *Trace Into* или "горячая" клавиша [F7];

- остановка программы в местах, отмеченных с помощью контрольных точек. Для установки контрольной точки необходимо установить курсор в интересующее место программы и выбрать раздел локального меню *Debug* | *Toggle Breakpoint* либо нажать "горячую" клавишу [F5]. Точка останова отмечается с помощью красной полосы в окне редактора. Повторное выполнение указанных действий приведет к удалению контрольной точки.

Просмотр или изменение содержимого переменных производится с помощью окна *Inspect*, вызов которого осуществляется с помощью раздела главного меню *Run* | *Inspect*. Можно также выделить курсором интересующую переменную, затем вызвать с помощью правой клавиши мыши локальное меню и выбрать из него раздел *Debug* | *Inspect* или нажать комбинацию "горячих" клавиш [*Alt*+F5]. Для этих же целей служит окно *Evaluate*/*Modify*, вызываемое командой *Run* | *Evaluate*/*Modify…* главного меню, разделом *Debug* | *Evaluate*/*Modify…* локального меню, либо с помощью комбинации "горячих" клавиш [*Ctrl*+F7].

При необходимости постоянного наблюдения за содержимым отдельных переменных может использоваться окно *Watch*, вызываемое с помощью раздела главного меню *Run* | *Add Watch*. Можно также выделить курсором интересующую переменную, затем вызвать локальное меню и выбрать из него раздел *Debug* | *Add Watch At Cursor* или нажать комбинацию "горячих" клавиш [*Ctrl*+F5].

7.3. Условный оператор if

Оператор *if* относится к условным операторам языка C++. Этот оператор имеет следующий вид:

if(*условие*) *оператор1*; **else** *оператор2*;

Здесь *оператор1* и *оператор2* могут состоять из одного или нескольких операторов (представлять собой составной оператор) или отсутствовать вовсе. Раздел *else* является необязательным.

Условие представляет собой условное выражение. В языке C++ результатом условного выражения может являться логическое значение (истина или ложь), целое число, символ, указатель или число с плавающей точкой. Если *условие* истинно (или не равно 0), то выполняется *оператор1*, в противном случае выполняется *оператор2*. Операторы, указанные в разделах *if* и *else*, являются взаимоисключающими. Схема действия оператора *if* представлена на Рис.7.3.

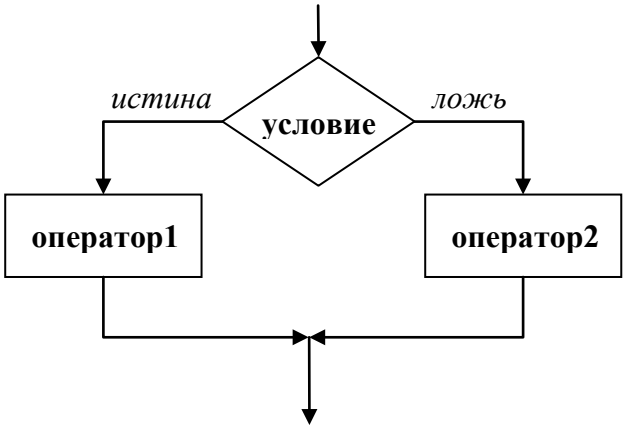


Рис.7.3. Блок схема условного оператора *if*.

В условном выражении могут использоваться операции сравнения: равно (==), не равно (!=), больше (>); больше или равно (>=), меньше (<), меньше или равно (<=).

Условное выражение также может состоять из нескольких сравнений, связанных логическими операторами отрицания (!), логического И (&&), логического ИЛИ (||).

Пример:

```
if(a>0 || a==b) { x=1; y=2; } else { x=-1; y=-2; }
```

7.4. Описание операторов и функций языка С ++

void main() – первый оператор программы, в нем указывается имя главной программы *main*.

{} – операторы начала и конца программы или составного оператора.

double a, b, c, d, x1, x2; – оператор описания переменных. При трансляции данного оператора для каждой переменной выделяется память в таком размере, сколько необходимо для размещения данных указанного типа.

#include <*filename*>– директива включения в программу содержимого файла *filename*. Используется в программе для подключения заголовочных файлов стандартной библиотеки языка С++.

getchar(); – функция ввода символа со стандартного устройства ввода (клавиатуры). Описание этой функции находится в заголовочном файле *conio.h* стандартной библиотеки. Здесь используется для формирования паузы в работе программы;

scanf(*формат*, *список аргументов***);** – функция ввода текста и численных значений переменных со стандартного устройства ввода (клавиатуры). Описание этой функции находится в заголовочном файле *stdio.h* стандартной библиотеки.

Здесь *формат* – текстовая строка, определяющая формат ввода. Для ввода значений целого типа используется обозначение *%d*, вещественного типа – *%g*, вещественного типа с двойной точностью – *%lg*.

список аргументов – список адресов переменных, которым присваиваются вводимые значения.

Пример:

scanf("%d %g", &x, &y); – ввести два значения целого и вещественного типа и присвоить их соответственно переменным *x* и *y*;

printf(*формат***,** *список аргументов***);** – функция вывода текста и численных значений переменных на стандартное устройство вывода (дисплей). Описание этой функции находится в заголовочном файле *stdio.h* стандартной библиотеки.

Здесь *формат* – текстовая строка, определяющая формат вывода. Для вывода значений целого типа используется обозначение *%d*, вещественного типа – *%g*.

список аргументов – список переменных, значения которых выводятся.

Пример:

print("x= %d, y= %g", x, y); – вывести значения переменных *x* и *y*;

x = a*b+c; – оператор присваивания, слева – имя переменной, которой присваивается значение, справа – арифметическое выражение, значение которого вычисляется и присваивается;

sqrt(*аргумент***);** – вызов стандартной функции, вычисляющей квадратный корень от *аргумент*. Описание этой функции находится в заголовочном файле *math.h* стандартной библиотеки функций C++.

7.5. Нахождение корней квадратного уравнения

7.5.1. Постановка задачи

Пусть дано квадратное уравнение $a \cdot x^2 + b \cdot x + c = 0$.

Требуется получить решения для любых заданных коэффициентов уравнения.

7.5.2. Метод и алгоритм решения

Известно, что решение квадратного уравнения находится по формуле

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a},$$

которая имеет смысл для случая $a \neq 0$. Если же $a=0$, то квадратное уравнение превращается в линейное $b \cdot x + c = 0$, решение которого находится в виде $x = -c/b$ и имеет смысл когда $b \neq 0$. Если $a=0$ и $b=0$, а $c \neq 0$, то решений уравнение не имеет. При $a=0$, $b=0$ и $c=0$ уравнение имеет решением любое x. Все эти условия должны быть учтены в программе.

7.5.3. Блок схема алгоритма

На Рис.7.4 представлена блок схема алгоритма решения задачи.

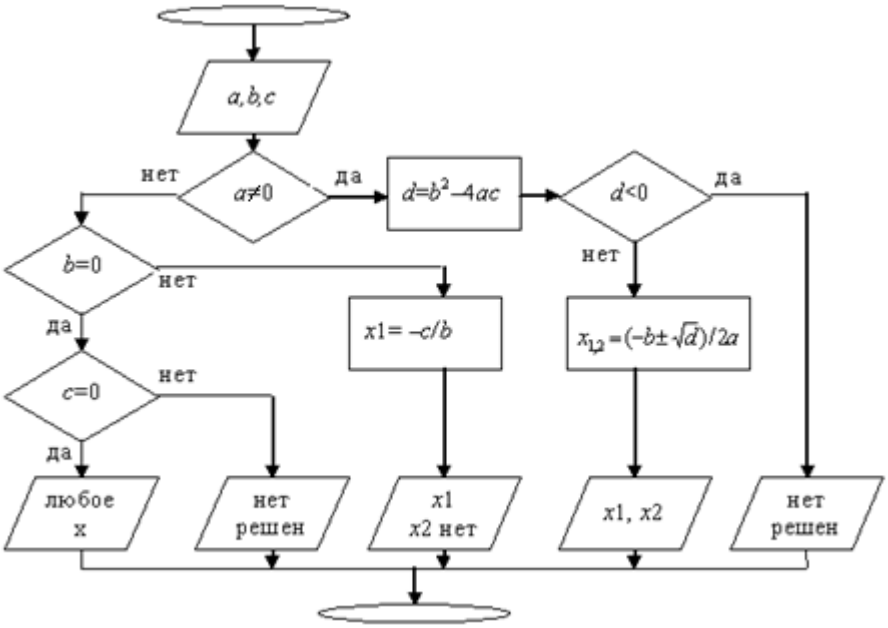


Рис.7.4. Блок схема алгоритма решения квадратного уравнения

7.6. Реализация алгоритма на языке C++

Ниже приводится текст программы на языке C++, которая вычисляет корни квадратного уравнения x_1 и x_2 по известным коэффициентам *a*,*b* и *c*.

```
#include <stdio.h>
#include <conio.h>
#include <math.h>

void main()
{
 float a, b, c, d, x1, x2;
 printf(" Введите a ");   scanf("%g", &a);
 printf(" Введите b ");   scanf("%g", &b);
 printf(" Введите c ");   scanf("%g", &c);

 if(a!=0) {
   d=b*b-4*a*c;
   if(d>=0) {
     x1=(-b-sqrt(d))/(2*a);
     x2=(-b)+sqrt(d)/(2*a);
     printf("X1 = %g,   X2 = %g", x1, x2);
   }
   else printf(" дискриминант отрицательный, решения комплексные");
 }
 else if(b!=0) {
       x1=-c/b;
       printf("X1 = %g,   X2 нет");
     }
     else
        if(c!=0) printf("решений нет");
```

```
        else printf("решение - любое X");

  getch();
}
```

7.7. Порядок выполнения работы

1. Создать в среде C++ Builder программу *Prog1*, осуществляющую вывод на экран дисплея текст "Hello, World!".
2. Для указанного варианта задания из табл.1 сформулировать постановку задачи, выбрать и обосновать метод ее решения.
3. Разработать блок-схему алгоритма.
4. Создать новый проект и разработать программу на языке C++ для решения поставленной задачи. Сохранить проект на жестком диске в своей папке.
5. Провести отладку программы, используя отладочные средства C++ Builder, требуемые расчёты и получить численные результаты для 3-х вариантов значений коэффициентов.
6. Проверить полученные результаты подстановкой их в уравнение.
7. С использованием текстового редактора MS Word оформить отчёт о проделанной работе.
8. При сдаче лабораторной работы студент должен показать и объяснить результаты выполнения задания на компьютере, ответить на контрольные вопросы.

7.8. Варианты заданий

Таблица 7.1. Варианты заданий к лабораторной работе

Номер варианта	Уравнение	Номер варианта	Уравнение
1.	$6(ax–b)–a=2\|a+x\|–c$	21.	$\|x+a\|–\|x–b\|=c$
2.	$\|ax–b\|=c–2a(x–2)$	22.	$\|x+a\|=c–\|x+b\|$
3.	$\|x–c\|/(x+b)=(a–x)/(x+b)$	23.	$\|a–2x\|+\|x+b\|=c–3x$
4.	$2\|c–2x\|=ax+b$	24.	$\|ax+b\|=cx$
5.	$a^2x=a\|x+b\|–c$	25.	$\|x+a\|=b/(c–x)$
6.	$\|a+5x\|/(b–x)=2c$	26.	$\|x–a\|=\|x^2–5x+9\|$
7.	$(x^4–a^2)/(a–x^2)=–(bx+c)$	27.	$(x^2+bx+a)(x^2+bx)=c$
8.	$(x+a)(x^2–bx)+c(x+a)=0$	28.	$x(x+a)(x+b)(x+a+b)=c$
9.	$2/(x^2+a)+4/(x^2+b)=c$	29.	$x/(ax+b)=c/x$
10.	$(a–x)/(1–x^2)–(x+b)/(1–x^2)=(x+c)/(x+x^2)$	30.	$a/x+b/(x+b)=c$
11.	$(x^2+bx+c)–3\sqrt{x^2+bx+a}=6$	31.	$a(x^2+\frac{1}{x^2})+b(x+\frac{1}{x})=c$

12.	$\sqrt[2]{x+a}-\sqrt[2]{x-b}=c$	32.	$a^{2x}–ba^{x}–c=0$
13.	$\sqrt{a^2-x\sqrt{x^2-b^2}}=x-a$	33.	$a^{2x}–ba^{x}–c=0$
14.	$\sqrt{x+a}=c+\sqrt{x-b}$	34.	$a5^{2x}–b5^{x+1}+c=0$
15.	$\sqrt{x-a}-\sqrt{x-2a}=b$	35.	$a\ \lg^2 x+b\ \lg x–c=0$
16.	$\log_2(ax+b)–\log_2(x)=c$	36.	$\lg(x^2–bx–c)=a–\lg b$
17.	$\log_5(x+a)+\log_5(x-b)=\log_5 c$	37.	$\log_3(a/(x+b))=c$
18.	$\log_3(ax–b)=c$	38.	$a\lg^2 x^4–b\ \lg x^{14}–c=0$
19.	$\lg(a–x)–\lg(x+b)=\lg c$	39.	$\log^2_{a–x} b+2\log_{a–x} b–c=0$
20.	$\ln(x–a)^2=\ln c+\ln(x+a)$	40.	$\log_{ax+b} c–\log_c(ax+b)=2$

Лабораторная работа №8

Построение таблицы значений функции. Организация циклов в С++

Цель работы – ознакомление и приобретение навыков организации циклов средствами языка С++, разработка программ для вычисления табличных значений функций.

8.1. Организация циклов в С++

В языке С++, как и во всех других современных языках программирования, операторы цикла предназначены для выполнения повторяющихся инструкций, пока выполняется определенное условие. Это условие может быть задано заранее, так и меняться во время выполнения цикла. В языке С++ имеются три оператора цикла: *while*, *do-while* и *for*.

8.1.1. Оператор цикла while

Имеет следующий вид:

while (*условие***)** *оператор1*;

Здесь *условие* может задаваться любым выражением. Если *условие* истинно, то *оператор1* выполняется до тех пор, пока *условие* не станет ложным. Как только условие цикла становится ложным, программа передает управление оператору, стоящему сразу за оператором ***while***. Условие цикла считается истинным, если значение этого выражения имеет значение *true* или не равно нулю. *Оператор1* может быть пустым, отдельным оператором или составным оператором (группой операторов).

Проверка *условия* осуществляется до выполнения оператора *оператор1*. Поэтому оператор ***while*** называется оператором цикла с *предусловием*. Если *условие* ложно с самого начала, то *оператор1* вообще не выполняется.

Пример:

```
Void main()
{
 int k, x, y, n;
 printf("x= %d", x);  scanf("%d", &x);
 printf("n= %d", n);  scanf("%d", &n);

 k=0;   y=1;
 while(k<n) {
   y=y*x;
   k=k+1;
 }
 printf("x^n = %d", y);
}
```

Данный фрагмент программы производит вычисление $y=x^n$.

8.1.2. Оператор цикла do-while

В отличие от оператора цикла ***while***, который проверяет условие в начале цикла, оператор ***do-while*** делает это в конце, поэтому он называется оператором цикла с *постусловием*. Это значит, что цикл ***do-while*** выполняется по крайней мере один раз. Оператор имеет следующий вид:

do *оператор1*; **while (***условие***);**

Сначала выполняется *оператор1*, затем проверяется *условие*. Это повторяется до тех пор, пока *условие* не станет ложным. Если *условие* ложно, то выполняется следующий по тексту программы оператор.

Также как и в случае с оператором ***while,*** условие цикла считается истинным, если значение этого выражения имеет значение *true* или не равно нулю. *Оператор1* может быть пустым, отдельным оператором или составным оператором (группой операторов).

Пример:

```
do scanf("%d", &num); while(num<1 || num>5);
```

В примере ввод значений в переменную *num* будет продолжаться до тех пор, пока не будет введено число из диапазона от 1 до 5.

8.1.3. Оператор цикла for

Оператор имеет следующий общий вид:

for(*инициализация; условие; приращение***)** *оператор1*;

Имеется много вариантов реализации цикла ***for***. Однако наиболее общая форма этого оператора работает следующим образом. Сначала производится оператор присвоения, находящийся в разделе *инициализация,* который задает начальное значение счетчика циклов, Затем проверяется *условие,* представляющее собой условное выражение. *Оператор1* выполняется в цикле до тех пор, пока значение выражения *условие* остается истинным. В разделе *приращение* изменяется значение счетчика цикла при очередном его выполнении. Как только *условие* цикла станет ложным, программа прекратит его исполнение и перейдет к следующему оператору. *Оператор1* может быть пустым, отдельным оператором или составным оператором (группой операторов). В следующем примере на экран выводятся числа от 1 до 100.

```
void main(void)
{
 int x;
 for(x=1; x<=100; x++) printf(" %d", x);
}
```

Здесь оператор x++ увеличивает содержимое переменной *x* на 1 (эквивалентен оператору x=x+1). Сначала переменной присваивается число 1, а затем

она сравнивается с числом 100. Поскольку ее значение меньше 100, вызывается функция ***printf***. Затем переменная *x* увеличивается на 1, и условие цикла (x<=100) проверяется вновь. Как значение *x* превысит число 100, выполнение цикла прекратится. В данном случае переменная *x* является счетчиком цикла, который изменяется и проверяется в каждом цикле.

В разделе *инициализация* можно размещать не только оператор присвоения, но и оператор описания переменных.

Пример:

```
void main(void)
{
 for(int z, x=100; x!=50; x-=5;) {
   z = x*x;
   printf(" %d", x);
}
```

Возведение числа *x* в квадрат и вызов функции ***printf*** выполняются до тех пор, пока значение переменной *x* не станет равным 50. Действие оператора x-=5 эквивалентно оператору x=x-5, т.е. значение переменной *x* уменьшается с шагом 5. В данном примере переменные *x* и *z* будут существовать только внутри оператора ***for***. За его пределами они будут недоступны.

Все разделы оператора ***for*** (*инициализация*, *условие* и *приращение*) не являются обязательными, т.е. каждый из них может отсутствовать. Если выражение в разделе *условие* не указано, то оно считается истинным. Оператор

```
for(; ;) printf("Этот цикл выполняется бесконечно");
```

будет вызывать функцию ***printf*** бесконечное число раз.

8.1.4. Оператор break

Прекращает выполнение текущего цикла ***while***, ***do-while*** или ***for*** и передает управление оператору, следующему сразу вслед за этим циклом.

Пример:

```
int i=0;
while(1) {
   printf("%d", i);
   if(i==10) break;
   i++;
}
```

В рассмотренном примере цикл прекратится, как только переменная *i* получит значение 10.

Если циклы вложены друг в друга, то оператор break выполняет выход из внутреннего цикла во внешний.

8.1.5. Оператор continue

Передает управление в начало ближайшего оператора цикла ***while***, ***do-while*** или ***for***, пропуская все оставшиеся в теле цикла операторы, и вызывает

начало следующей итерации. В цикле ***for*** оператор ***continue*** вызывает проверку условия и приращение счетчика цикла. В циклах ***while*** и ***do-while*** оператор ***continue*** передает управление оператору, входящему в условие цикла.

Пример:

```
for(int x=0; x<100; x++) {
  if(x>10 && x<20) continue;
  printf("%d ", x);
}
```

Данный пример выводит на экран числа от 0 до 99, за исключением диапазона между 11 и 19.

8.2. Построение таблицы значений функции

8.2.1. Постановка задачи

Необходимо вычислить заданное количество значений функции

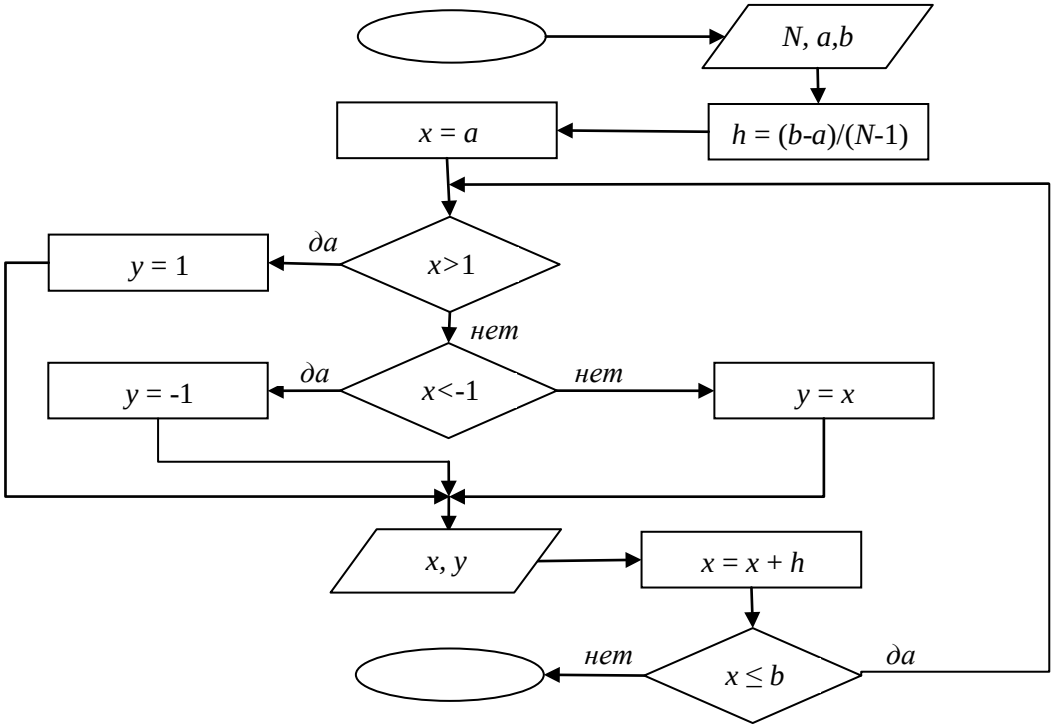
$$y(x)=\begin{cases} 1 & при \quad x>1 \\ x & при \quad -1<=x<=1 \\ -1 & при \quad x<-1 \end{cases}$$

на указанном интервале $[a, b]$.

8.2.2. Алгоритм

1. Ввод исходных данных: количество вычисляемых значений функции N, границы интервала a и b.
2. Искомые N значений функции разбивают заданный интервал $[a, b]$ на N-1 отрезков. Определяем шаг разбиения интервала: $h=(b-a)/(N-1)$.
3. Задание начального значения аргумента $x=a$;
4. Вычисление значения функции $y(x)$ в точке x.
   Если $x>1$, то $y=1$; в противном случае если $x<-1$, то $y=-1$; в противном случае $y=x$;
5. Вывод x и y на экран дисплея.
6. Увеличение значения аргумента x на величину шага h: $x=x+h$;
7. Проверка условия $x \le b$. Если оно выполняется, то переход к п.4 алгоритма, в противном случае – останов.

8.2.3. Блок схема алгоритма



8.2.4. Реализация алгоритма на языке C++

Реализация с использованием оператора ***while***:

```
#include <stdio.h>
#include <conio.h>

void main()
{
 int N;
 float a, b, h, x, y;
 printf("Введите N ");   scanf("%d", &N);
 printf("Введите a ");   scanf("%g", &a);
 printf("Введите b ");   scanf("%g", &b);

 h=(b-a)/(N-1);
 x = a;
 while(x<=b) {
   if(x>1) y=1;
    else if(x<-1) y=-1;  else y=x;

   printf("x=%g   y=%g\n", x,y);
   x=x+h;

 }
 getch();
}
```

С использованием оператора *do - while*:

```
#include <stdio.h>
#include <conio.h>

void main()
{
 int N;
 float a, b, h, x, y;
 printf("Введите N ");   scanf("%d", &N);
 printf("Введите a ");   scanf("%g", &a);
 printf("Введите b ");   scanf("%g", &b);

 h=(b-a)/(N-1);
 x = a;
 do {
    if(x>1) y=1;
     else if(x<-1) y=-1;  else y=x;

    printf("x=%g   y=%g\n", x,y);
    x=x+h;

 } while(x<=b);
 getch();
}
```

С использованием оператора *for*:

```
#include <stdio.h>
#include <conio.h>

void main()
{
 int N;
 float a, b, h, y;
 printf("Введите N ");   scanf("%d", &N);
 printf("Введите a ");   scanf("%g", &a);
 printf("Введите b ");   scanf("%g", &b);

 h=(b-a)/(N-1);
 for(float x=a; x<=b; x+=h) {
    if(x>1) y=1;
     else if(x<-1) y=-1;  else y=x;

    printf("x=%g   y=%g\n", x,y);
 }
 getch();
}
```

8.3. Варианты заданий

1. Напечатать таблицу значений функции:

$$y=\begin{cases}\sin(x) & при \quad \sin(x) < \cos(x)\\ \cos(x) & при \quad \cos(x) <= \sin(x)\end{cases}$$

2. Напечатать таблицу значений функции:

$$y=\begin{cases}x-1 & при \quad x>1\\ 0 & при \quad -1<=x<=1\\ x+1 & при \quad x<-1\end{cases}$$

3. Напечатать таблицу значений функции:

$$y=\begin{cases}0.5 & при \quad \sin(x)>0.5\\ \sin(x) & при \quad -0.5<=\sin(x)<=0.5\\ -0.5 & при \quad \sin(x)<-0.5\end{cases}$$

4. Напечатать таблицу значений функции:

$$y=\begin{cases}1 & при \quad x>1\\ x*x & при \quad -1<=x<=1\\ 0 & при \quad x<-1\end{cases}$$

5. Напечатать таблицу значений функции:

$$y=\begin{cases}x*x & при \quad x>x*x\\ \\ x*x*x & при \quad x<=x*x\end{cases}$$

8.4. Порядок выполнения работы

1. Для полученного варианта задания составить программу на приближенное вычисление функции на C++. В программе предусмотреть вычисление и вывод значения функции.

2. Отладить программу и выполнить задание на компьютере.

3. С использованием текстового редактора оформить отчет о проделанной работе. В отчете необходимо отразить следующие пункты:

- постановка задачи;
- алгоритм решения (блок-схема);
- текст программы;
- описание программы;
- контрольный пример;
- результаты счета;
- анализ результатов и выводы.

4. При сдаче лабораторной работы студент должен показать и объяснить результаты выполнения задания на компьютере, ответить на вопросы преподавателя по теме лабораторной работы.

8.5. Контрольные вопросы

1. Каковы особенности организации цикла с помощью оператора ***while***?
2. Каковы особенности организации цикла с помощью оператора ***while-do***?
3. Каковы особенности организации цикла с помощью оператора ***for***?
5. Какие операторы C++ используются для реализации разветвляющей структуры?
6. Как работает оператор *continue*?
7. Для чего нужен оператор *break*?
8. Что произойдет при выполнении операторов *i++* и *x–=2*?

Лабораторная работа №9

Накапливание результата. Итерационные алгоритмы вычисления приближенного значения функций

Цель работы – ознакомление и приобретение навыков составления программ для накапливания результата и приближенного вычисления значения функций по итерационным формулам.

9.1. Накапливание результата

Во многих задачах результат образуется за счет многократного сложения или умножения чисел, составляющих некоторую последовательность. В качестве примеров таких задач можно привести отыскание значения функции, представленной в виде ряда, определение статистических характеристик множества чисел или вычисление факториала.

Все подобные задачи решаются с помощью циклического алгоритма, в котором на каждом новом этапе алгоритма к вычислению привлекается новый элемент множества. При использовании такого циклического алгоритма возникает вопрос о начальном значении вычисляемого результата. Здесь возможны два подхода. В одном случае в качестве начального значения используют первый элемент последовательности. В другом случае выбирают такие начальные значения, которые не могут исказить результат. При накапливании суммы начальное значение принимают равным 0, а при накапливании произведения равным 1.

Пример 1. Определить сумму всех целых чисел в диапазоне от 1 до 20 $s = \sum_{i=1}^{20} i.$

Приведем фрагмент программы на C++ для решения данной задачи

```
int s, i;
s=0;
i=1;
while(i<=n){ s=s+i;  i++; }
```

Пример 2. Вычислить $f = k! = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot k$

```
int f, i;
f=1;
for(i=1; i<=k; i++) f=f*i;
```

9.2. Итерационные алгоритмы

Практически все численные методы основаны на последовательном приближении вычисленного значения к искомому результату.

Итерационными (пошаговыми) алгоритмами называются алгоритмы, в которых на каждом шаге используется одна и та же формула, выраженная через значения, полученные на предыдущих шагах алгоритма. Выполнение

такого алгоритма сводится к генерации некоторой числовой последовательности результатов:

$$\{x_0, x_1,\ldots,x_k, x_{k+1}, \ldots\},$$

где k – это номер итерации, x_k – это значение, полученное на k-ом шаге процесса.

Итерационной формулой в общем виде называется выражение

$$x_{k+1}=f(x_0, x_1,\ldots, x_k),$$

позволяющее генерировать последующие члены последовательности через вычисленные ранее (на предыдущих шагах алгоритма). Чаще же всего итерационные формулы имеют более простой вид:

$$x_{k+1} = f(x_k).$$

Итерационная последовательность своим пределом должна иметь искомое значение – x^*:

$$\lim_{k\to\infty} x_k = x^*$$

Если такой предел существует, то итерационный процесс называется *сходящимся.* Если нет, то *расходящимся.*

Сам факт сходимости, впрочем, как и скорость сходимости, итерационных процессов может зависеть от выбора начального приближения x_0. Если алгоритм генерирует сходящуюся последовательности при любом выборе x_0, то такой процесс называется *глобально сходящимся*. Если же сходимость итерационной последовательности к искомому пределу x^* имеет место только при задании x_0 из некоторой окрестности x^*, то соответствующий итерационный процесс называют *локально сходящимся*.

Реальный вычислительный процесс всегда должен заканчиваться при конечном значении k, поэтому всегда возникает проблема выбора условия для окончания итераций. Так как заранее неизвестно, сколько потребуется вычислить и просуммировать членов последовательности, то требуется задать условие выхода из цикла. В качестве такого условия чаще всего принимается требуемая точность вычисления функции. Однако не всегда бывает известно, будет ли достигнута заданная точность при суммировании конечного числа членов. Поэтому, чтобы не было зацикливания, необходимо предусмотреть дополнительное условие выхода из цикла, например, по количеству вычисленных и просуммированных членов последовательности.

При реализации итерационных алгоритмов для уменьшения количества вычислений целесообразно вывести зависимость k-го члена последовательности через предыдущие (если это возможно) и затем использовать эту зависимость для вычислений новых членов на каждой итерации. Получить эту зависимость можно путем деления k+1-го члена последовательности на k-й.

9.2.1. Постановка задачи

Пусть функция $F(x)$ задана в виде бесконечного, абсолютно сходящегося в некоторой области значений x, ряда

$$F(x) = \sum_{k=1}^{\infty} a_k(x),$$

где $a_k(x)$ –k-й член ряда. Требуется вычислить приближённое значение функции в некоторой заданной точке x на основе суммирования членов ряда.

В качестве примера рассмотрим функцию e^x:

$$F(x) = e^x = 1 + x + x^2/2! + \ldots + x^k/k! \qquad (1)$$

9.2.2. Метод решения

Выведем зависимость k-го члена ряда через предыдущие. Получить эту зависимость можно путем деления k+1-го члена ряда на k-й. В рассматриваемом примере

$$a_k = \frac{x^k}{k!},\ a_{k+1} = \frac{x^{k+1}}{(k+1)!},\ \frac{a_{k+1}}{a_k} = \frac{x^{k+1}k!}{x^k(k+1)!} = \frac{x}{k+1}.$$

В результате получаем такую зависимость:

$$a_0 = 1,\ a_{k+1} = a_k x/(k+1),\ k=0,1,2,\ldots$$

В качестве условия выхода следует принять неравенство

$$|a_{k+1}| < e$$

где e - заданное достаточно малое положительное число (точность).

9.2.3. Алгоритм

1. Ввод исходных данных: значения x, e, допустимое количество итераций k_{max}.
2. Положитьa_0=l, $s = a_0$, $k = 0$.
3. Проверка условия $|a_0|<e$. Повторять пункты 4-8 алгоритма до тех пор, пока данное условие выполняется. Если условие не выполняется, то переход к пункту 7.
4. Проверка условия $k<k_{max}$. Если оно выполняется, то переход к пункту 3 алгоритма, если не выполняется, то вывести на печать сообщение "за заданное количество итераций точность не достигнута" и останов.
5. Вычислить $k=k$+l.
6. Вычислить $a_k(x)$: $a_l = a_{0*}x/k$.
7. Вычислить сумму: $s = s+a_l$.
8. Вычислить $a_0 = a_l$.
9. Вывод значения функции.
10. Останов.

9.2.4. Блок-схема алгоритма

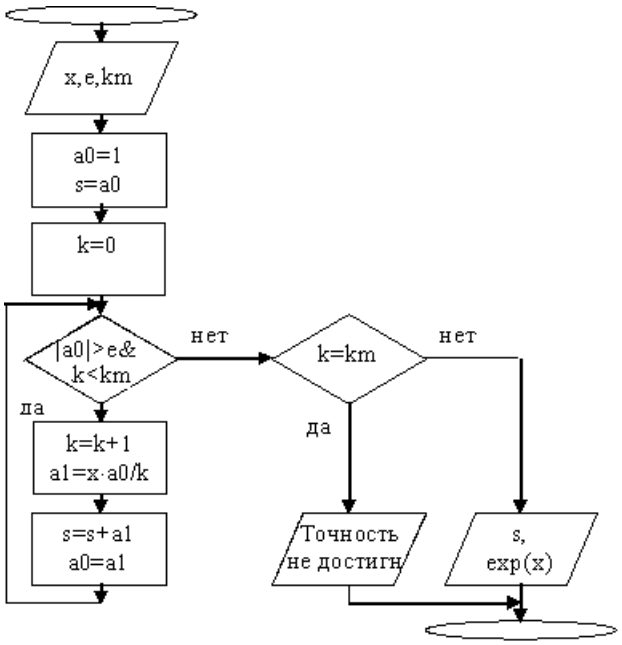


Рис.9.1. Блок схема алгоритма приближенного вычисления экспоненты

9.2.5. Пример программы

```
#include<stdio.h>
#include <math.h>
#include <conio.h>

void main()
{
 int k,kmax;
 double x, a0,a1,s,e;

/* Ввод исходных данных */
 printf("x=");      scanf("%lg", &x);
 printf("eps=");    scanf("%lg", &e);
 printf("kmax=");   scanf("%d", &kmax);

/* Вычислениеряда */
 a0=1;   s=a0;   k=0;
 while(fabs(a0)>e && (k<kmax)) {
    k=k+1;
    a1=a0*x/k;
    s=s+a1;
    a0=a1;
}
 if(k==kmax) printf("За %d итераций точность не достигнута", k);
 else printf("Число итераций k= %d\n", k);
 printf("Приближенное значениеexp(x)= %g, %g", s, exp(x));
 getch();
}
```

9.3. Варианты заданий

Написать программу вычисления приближенного значения функции (табл.1) с абсолютной погрешностью, не превышающей δ. Для определения погрешности воспользоваться правилом теории рядов, согласно которому для знакочередующихся сходящихся рядов $\sum_{i=1}^{\infty} a_i$ справедлива следующая оценка остатка ряда $|R_n|<|a_{n+1}|$, где $R_n = \sum_{k=n+1}^{\infty} a_k$. Сравнить результат с точным значением функции.

Таблица 1

Номер варианта	Функция и ее разложение в ряд	δ
1.	$\cos z = \sum_{k=0}^{\infty} \frac{(-1)^k z^{2k}}{(2k)!} \; (\lvert z\rvert < \infty)$	0.001
2.	$\sin z = \sum_{k=0}^{\infty} \frac{(-1)^{k-1} z^{2k-1}}{(2k-1)!} \; (\lvert z\rvert < \infty)$	0.01
3.	$\ln(1+z) = \sum_{k=0}^{\infty} \frac{(-1)^{k-1} z^{k}}{k} \; (\lvert z\rvert < 1)$	0.001
4.	$\operatorname{arctg}(z) = \sum_{k=0}^{\infty} \frac{(-1)^k z^{2k+1}}{2k+1} \; (\lvert z\rvert \le 1)$	0.01
5.	$\frac{1}{(1+x)^2} = 1 - 2\cdot x + 3\cdot x^2 - 4\cdot x^3 + 5\cdot x^4 + \ldots \; (\lvert x\rvert < 1)$	0.01
6.	$e^{-x^2} = 1 - \frac{x^2}{1!} + \frac{x^4}{2!} - \ldots + (-1)^N \cdot \frac{x^{2*N}}{N!} \; (\lvert x\rvert < \infty)$	0.01
7.	$\frac{\sin(x)}{x} = 1 - \frac{x^2}{3!} + \frac{x^4}{5!} - \frac{x^6}{7!} + \ldots \; (\lvert x\rvert < \infty)$	0.001
8.	$e^{-x} = 1 - \frac{x}{1!} + \frac{x^2}{2!} - \ldots \; (\lvert x\rvert < \infty)$	0.01
9.	$\sqrt{1+x} = 1 + \frac{1}{2}\cdot x - \frac{1\cdot 3}{2\cdot 4}\cdot x^2 + \frac{1\cdot 3\cdot 5}{2\cdot 4\cdot 6}\cdot x^3 - \ldots \; (\lvert x\rvert < 1)$	0.01
10.	$\frac{1}{(1+x)^3} = 1 - \frac{2\cdot 3}{2}\cdot x + \frac{3\cdot 4}{2}\cdot x^2 - \frac{4\cdot 5}{2}\cdot x^3 + \ldots \; (\lvert x\rvert < 1)$	0.01

9.4. Порядок выполнения работы

1. Для полученного варианта задания составить программу на приближенное вычисление функции на C++. В программе предусмотреть вычисление и вывод значения функции.
2. Отладить программу и выполнить задание на компьютере.
3. С использованием текстового редактора оформить отчет о проделанной работе. В отчете необходимо отразить следующие пункты:
   - постановка задачи;
   - алгоритм решения (блок-схема);
   - текст программы;
   - описание программы;
   - контрольный пример;
   - результаты счета;
   - анализ результатов и выводы.
4. При сдаче лабораторной работы студент должен показать и объяснить результаты выполнения задания на компьютере, ответить на вопросы преподавателя по теме лабораторной работы.

9.5. Контрольные вопросы

1. Каковы особенности программирования итерационных алгоритмов приближенного вычисления функций, заданных бесконечным рядом?
2. Какие операторы используются в программе для реализации циклических структур?
3. Какие операторы C++ используются для реализации разветвляющей структуры?

*Лабораторная работа №*10

Указатели, функции и одномерные массивы в С++. Задачи поиска и сортировки

Цель занятия – ознакомиться с понятиями указателей, функций и одномерных массивов в языке С++. Получить практические навыки в решении задач поиска и сортировки.

10.1. Указатели

Указатель – это переменная, значением которой является адрес. То есть это такая переменная, в которой можно сохранить адрес какой-то другой переменной. Если более точно, то указатель хранит адрес ячейки памяти, в которой содержится значение другой переменной.

Так же как переменные различаются типами, указатели тоже бывают разными. Так при объявлении указателя обязательно задается, на переменную какого типа это указатель. То есть, по сути – адрес переменной какого типа будет содержаться в этом указателе.

Переменная, хранящая адрес, должна быть объявлена как указатель. Объявление указателя состоит из имени типа той переменной, адрес которой хранится в указателе, символа "*" и имени указателя:

```
тип *имя_указателя;
```

Рассмотрим пример. Чтобы объявить переменную *p* указателем на переменную целого типа необходимо использовать следующую инструкцию

```
int *p;
```

Для объявления указателя *q* на тип *float* необходимо написать

```
float *q;
```

В общем случае использование символа "звездочка" (*) перед именем переменной при ее объявлении превращает эту переменную в указатель.

С указателями используются два оператора: "*" и "&".

Оператор "&", стоящий перед именем переменной, возвращает адрес этой переменной. Например, при выполнении следующего фрагмента кода

```
int balance = 100;
int *ptr;
ptr = &balance
```

в переменную *ptr* помещается адрес переменной *balance*. Этот адрес соответствует области памяти компьютера, которая принадлежит переменной *balance*. Выполнение этой инструкции никак не влияет на значение переменной *balance*. Назначение оператора "&" можно выразить фразой "получить адрес переменной".

Оператор "*" ставится непосредственно перед указателем. Он обозначает обращение к значению переменной, адрес которой содержит данный указатель. Если в продолжение предыдущего фрагмента написать

```
int value;
value = *ptr;
```

то переменной *value* будет присвоено значение переменной *balance*. Т.е. после выполнения последнего оператора переменная *value* будет содержать значение 100. Действие оператора "*" можно выразить фразой "значение, расположенное по адресу". В данном случае последний оператор можно прочитать так: "переменная *value* получает значение, расположенное по адресу *ptr*".

Переменные-указатели должны всегда указывать на данные соответствующего типа. Несоблюдение этого условия может привести к ошибке. Например, следующий фрагмент кода некорректен

```
int *p;
double f;
p = &f;          // ОШИБКА!
```

Здесь указателю типа *int* присваивается адрес переменной типа *double*. Это неверно.

При присваивании значения области памяти, адресуемой указателем, указатель можно использовать с левой стороны от оператора присвоения. Например, после исполнения следующего фрагмента кода

```
int *p, num;
p = &num;
*p = 10;
```

переменной *num* будет присвоено значение 10.

Помимо операций "&" и "*" с указателями также можно производить арифметические операции и операции сравнения.

10.2. Функции

Функция – это самостоятельная единица программы, созданная для решения конкретной задачи. Функция в языке C++ играет ту же роль, что и подпрограммы или процедуры в других языках. Функциями удобно пользоваться, например, если необходимо обработать один и тот же код программы.

Описание функции содержит заголовок, за которым следует программный блок (*тело функции*).Если функция имеет аргументы, то в ее заголовке должны быть объявлены переменные, принимающие их значения. Эти переменные называются *формальными параметрами*. В заголовке функции определяется тип результата, возвращаемого функцией, имя функции иформальные параметры (если они имеются). Список формальных параметров заключается в круглые скобки. В нем перечисляются типы параметров и их имена, разделенные запятыми.

В теле функции задаются операторы, которые должны выполняться при активизации функции. Программный блок выделяется фигурными скобками "{" и "}". Приведем пример задания функции:

```
    doublemax(double x, double y)
/* функция возвращает максимум из значений x и y */
    {
     double m;
     if(x>=y) m=x;  else m=y;
     returnx;
    }
```

При вызове функции указывается имя функции и *фактические параметры*. Функция может возвращать или не возвращать значение в вызвавшую ее программу. Если функция возвращает значение, то его тип указывается в описании функции непосредственно пред ее именем, а возвращаемое значение указывается после оператора *return*. Такие функции могут вызываться внутри операторов (например, в операторе присвоения "=") или как аргументы других функций. Если функция не возвращает значения, то в качестве возвращаемого типа указывается *void*. Вызывается такая функция как обычный оператор.

Вызов функции может включаться в выражения в качестве операнда. Когда выражение вычисляется, функция выполняется и значением операнда становится значение, возвращаемое функцией.

Пример программы, осуществляющей вызов функции *max*():

```
      void main(void)
      {
       double a, b, c;
       printf("Введите значения a и b: "); scanf("%lg %lg",
&a,&b);
       z = 1 + 5*max(a,b);
       printf("c = %g", c);
      }
```

10.2.1. Параметры функции

Параметры используются для передачи входных данных из вызывающей программы в функцию и выходных данных из функции в вызывающую программу.

Чтобы отличать параметры подпрограммы, описанные в её заголовке и теле, от параметров, указываемых при вызове подпрограммы, первые принято называть *формальными* параметрами, вторые – *фактическими* параметрами. При вызове функции фактические параметры, указанные в команде вызова, становятся значениями соответствующих формальных параметров, чем и обеспечивается передача данных в функцию.

В описании функции задается список формальных параметров. Каждый параметр, указанный в списке формальных параметров, является локальным по отношению к данной функции и в программном блоке функции на него можно ссылаться по его идентификатору (имени).

Существует два способа передачи параметров внутрь функции: *по значению* и *по ссылке*. При передаче параметра по значению формальному параметру функции присваивается копия значения фактического параметра. При этом все изменения формального параметра внутри функции никак не отражаются на фактическом параметре.

При передаче параметра по ссылке в качестве формального параметра передается адрес фактического параметра. Внутри функции этот адрес открывает доступ к фактическому параметру. Это значит, что все изменения формального параметра будут отражаться на значении фактического параметра.

10.2.2. Передача параметров по значению

Формальный параметр-значение обрабатывается, как локальная по отношению к процедуре или функции переменная, за исключением того, что он получает свое начальное значение из соответствующего фактического параметра при активизации функции. Изменения, которые претерпевает формальный параметр-значение, не влияют на значение фактического параметра.

Фактический параметр должен иметь тип, совместимый по присваиванию с типом формального параметра-значения.

По умолчанию в C++ используется передача по значению. В этом случае операторы, образующие тело функции, не могут изменять фактические параметры, указанные при ее вызове.

Пример:

```
#include <stdio.h>
int sqr(intx);
void main(void)
{
 int t=10;
 printf("%d %d", sqr(t), t);
}
int sqr(int x)
{
 x=x*x;
 return x;
}
```

В этом примере после выполнения присваивания x=x*x изменится только значение переменной *x* внутри функции *sqr*. Значение переменной *t*, указанной при вызове функции *sqr*(*t*), по-прежнему останется равным 10.

10.2.3. Передача параметров по ссылке

Для передачи параметра по ссылке необходимо передать внутрь функции указатель на этот параметр. В этом случае изменение значения формального параметра внутри функции приведет к изменению значения фактического параметра в вызывающей программе.

В этом случае параметры должны быть объявлены как указатели.

Пример:

```
void swap(int *x, int *y)
{
 int temp;
 temp=*x;  /* сохраняем значение, записанное по адресу x */
 *x=*y;    /* записываем значение, записанное  по адресу y,
              в ячейку по адресу x */
 *y=temp;  /* записываем содержимое temp по адресу y */
}
```

Функция *swap* может менять местами значения двух переменных, на которые ссылаются указатели *x* и *y*. К содержимому этих переменных можно обращаться, используя обычные операции над указателями **x* и **y*.

При вызове функции *swap* в качестве фактических параметров должны быть переданы указатели на переменные.

Пример:

```
void swap(int *x, int *y);
void main(void)
{
 int a=1, b=2;
 swap(&a, &b);     /* передаются адреса переменных a и b */
}
```

10.3. Одномерные массивы

Массив– это тип данных, используемый для представления большого количества однотипных значений, расположенных последовательно в памяти.

Массив описывается следующим образом:

```
тип имя_массива[размер];
```

Здесь тип определяет тип данных каждого элемента, составляющего массив. В C++ возможны массивы любых типов, в том числе и массивы массивов. Значение *размер* определяет количество элементов, которые будут храниться в массиве. Например, описание массива *sample*, состоящего из 10 элементов целого типа выглядит следующим образом:

```
int sample[10];
```

Доступ к отдельному элементу массива осуществляется с помощью индекса. Индекс описывает позицию элемента внутри массива. В C++ *первый элемент массива всегда имеет нулевой индекс*. Поскольку массив *sample* содержит 10 элементов, его индексы изменяются от 0 до 9. Чтобы получить доступ к элементу массива по индексу, достаточно указать нужный номер элемента в квадратных скобках. Так, первым элементом массива *sample* является *sample*[0], а последним – *sample*[9]. Например, следующая программа помещает в массив *sample* числа от 0 до 9.

```
#include <stdio.h>
void main(void)
{
 int sample[10]; // Эта инструкция резервирует область
// памяти для 10 элементов типа int.
 // Помещаемвмассивзначения.
 for(int i=0; i<10; i++) sample[i]=i;
// Отображаеммассив.
 for(int k=0; k<10; k++) printf("   %d", sample[k])
}
```

Все элементы массива занимают смежные ячейки памяти (т.е. располагаются в памяти последовательно друг за другом). Ячейка с наименьшим адресом относится к первому элементу массива, а с наибольшим – к последнему. Например, после выполнения следующего фрагмента кода

```
int i[7];
for(int j=0; j<7; j++)  i[j] = j;
```

массив *i* будет выглядеть так

i[0]	*i*[1]	*i*[2]	*i*[3]	*i*[4]	*i*[5]	*i*[6]
0	1	2	3	4	5	6

10.4. Методика составления программ поиска и сортировки

10.4.1. Задача поиска

Из множества данных, например, из массива чисел, требуется выбрать одно или несколько чисел, удовлетворяющих некоторому критерию для использования их в последующих вычислениях. Алгоритм решения этой задачи называется циклическим, причем в каждом проходе по циклу анализируется один из элементов множества. Поэтому рационально представить рассматриваемое множество в виде массива, чтобы можно было выбирать требуемый элемент массива указанием его индекса.

Пример. Найти наибольшее из данных *n* чисел. Сформируем массив *x*[*n*], в котором размещены эти числа. Введем новую переменную, например, *z*, которая должна получить значение, равное наибольшему из рассматриваемых чисел. Особое внимание следует уделить выбору начального значения *z*.

Необходимо предусмотреть, чтобы в процессе поиска начальное значение *z* по крайней мере один раз было заменено числом из массива *x*. В некоторых случаях заранее указать такое значение *z* затруднительно, поэтому в качестве начального значения принимается первый элемент массива *x*[0], а циклическая работа алгоритма начинается со второго элемента массива. Именно по этому принципу построен следующий фрагмент программы поиска максимального элемента массива:

```
int n=10, z, j, x[10];
. . .
z=x[0];   j=0;
for(int i=1; i<n; i++)
        if(z<x[i]) { z=x[i];   j=i; }
```

В результате вычислений переменная *z* получит значение, равное наибольшему числу в массиве *x*.

10.4.2. Задача сортировки и упорядочения массива

При решении этой задачи требуется сформировать новый массив, состоящий из элементов исходного массива, но расположенных в соответствии с предложенным критерием. Возможно два подхода при решении этой задачи. В первом случае исходный массив требуется сохранить для других задач, во втором случае исходный массив можно изменять, переставляя его элементы. Во втором случае программа получается более компактной, поэтому рассмотрим этот случай, тем более, что первый случай легко сводится ко второму путем предварительного переноса содержимого исходного массива в новый рабочий массив, что позволяет сохранить исходный массив.

Пример. Дан массив *a*[*n*]. Необходимо переставить этот массив таким образом, чтобы элементы массива были размещены в порядке возрастания их величины. Ход решения этой задачи следующий:

1. В исходном массиве *a* отыскать наименьший по величине элемент. Запомнить номер этого элемента.
2. Найденный элемент поменять местами с первым элементом массива.
3. Повторить поиск наименьшего элемента в массиве со второго до *n*-го элемента.
4. Новый наименьший элемент поменять местами со вторым элементом массива.
5. Этапы 3 и 4 повторить с уменьшающимся остатком массива до полного упорядочения массива.

Рассматриваемая задача решается с помощью следующего фрагмента программы:

```
for(int i=0; i<n-1; i++) {
    z=a[i];
    k=i;
```

```
        /* нахождение минимального элемента */
        for(int j=i+1; j<n; j++)
            if(z>=a[j]) { z=a[j];  k=j; }
        /* перестановка */
        c=a[j];
        a[i]=a[k];
        a[k]=c;
}
```

10.5. Варианты заданий

Исходным массивом является один из вариантов массива $a[10]$. Программа должна быть составлена для произвольной размерности n. В программе предусмотреть необходимые операторы для ввода и вывода исходных данных и результатов вычислений.

1. Сформировать массив $b[8]$, в котором элементы массива $a[10]$ размещены в порядке убывания их величины, а наименьшее и наибольшее числа из массива $a[10]$ отброшены.

2. Сформировать массив $b[10]$, в котором разместить элементы

$$b_i = a_i - a_{\text{ср}}, \qquad \text{где } a_{\text{ср}} = \frac{1}{n}\sum_{i=1}^{n} a_i.$$

3. Сформировать массив $c[10]$ по условию

$c_i = +1$, если $a_i > a_{\text{ср}}$;
$c_i = -1$, если $a_i < a_{\text{ср}}$;
$c_i = 0$, если $a_i = a_{\text{ср}}$.

4. Сформировать массив $d[10]$, в котором разместить элементы массива $a[10]$ в порядке возрастания величин $d_i = (a_i - a_{\text{ср}})^2$. Массив $a[10]$ сохранить.

5. Сформировать массив $g[10]$, в котором разместить элементы

$$g_i = \frac{a_i}{\sum_{j=1}^{n} |a_j|}$$

в порядке убывания модулей g_i.

5. Сформировать массивы x_i и y_i, в которых разместить соответственно нечетные и четные элементы массива a. Считая x_i, y_i координатами точек на плоскости, переставить элементы с одинаковыми индексами i в таком порядке, чтобы ломаная линия, соединяющая последовательно точки 1-2-3-…-k, имела наименьшую длину, для чего в качестве последующей точки надо выбирать ближайшую к предыдущей. Если исходный массив содержит нечетное количество элементов, дополнить его одним нулевым элементом.

Таблица 8.1. Исходный массив a[10]

Номер-вари-анта	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}
1	20.5	-45.2	00.0	32.9	22.1	-30.0	99.9	58.8	72.0	-11.3
2	01.2	34.5	67.8	91.0	11.1	2.13	14.1	51.6	17.1	81.9
3	14.1	-45.2	15.3	0.1	-16.6	13.6	15.1	-11.5	12.0	-16.1
4	33.0	63.1	51.3	45.0	-54.5	98.7	75.4	84.6	90.2	-50.9
5	12.2	25.0	-50.4	98. 7	23.6	95.8	35.7	64.2	28.7	10.1
6	99.0	10.0	11.1	-20.2	95.9	34. 1	33.5	98.1	89.7	72.3
7	25.5	99.9	22.2	12.9	00.0	52.1	48.2	10.8	-98.9	77.7

10.6. Порядок выполнения работы

1. Для полученного варианта составить программу на C++ для решения поставленной задачи. В программе предусмотреть вывод исходного массива в форме таблицы и результата в наглядной форме.

2. Отладить программу и выполнить задание на компьютере;

3. Составить отчет о проделанной работе на ПК в виде файла. В отчете необходимо отразить следующие пункты:

- Постановка задачи.
- Алгоритм решения (блок-схема).
- Текст программы.
- Описание программы.
- Контрольный пример.
- Результаты счета.
- Анализ результатов и выводы.

4. При сдаче лабораторной работы студент должен показать и объяснить результаты выполнения задания на компьютере, ответить на вопросы преподавателя по теме лабораторной работы.

Лабораторная работа №11

Обработка двумерных массивов

Цель занятия – ознакомление и получение практических навыков работы с двухмерными массивами.

11.1. Обработка двумерных массивов

11.1.1. Двухмерные массивы в C++

В языке C++, наряду с одномерными, предусмотрены также и ***многомерные массивы***. Простейшим их них является ***двухмерный массив***. По существу, двухмерный массив – это массив, элементами которого являются другие одномерные массивы.

Элементы двухмерного массива определяются двумя индексами – номером строки и номером столбца. Индексы должны всегда задаваться положительными целыми числами. *Отсчёт значений индексов всегда начинается с нуля*. Если в качестве индекса используется переменная, то она должна быть целого типа. При обращении к массиву индексы размещаются в квадратных скобках непосредственно после имени массива:

имя_массива[*номер_строки*][*номер_столбца*]

Например, для обращения к элементу массива A, находящемуся во 2 строке и 3 столбце (отсчёт строк и столбцов начинается с 0), необходимо записать *A*[1][2].

Примеры:

A[0][0] – самый первый элемент массива A (0 строка, 0 столбец);

Num[*i*][1] – *i*-я строка и 1 столбец массива *Num*;

Num[*i*][*j*] – *i*-я строка, *j*-й столбец массива *Num*;

Математическим аналогом двухмерного массива является ***матрица***, где строки и столбцы массива определяют соответствующие строки и столбцы матрицы.

При объявлении двухмерного массива в программе указывается тип его элементов, имя массива, а также размерность массива – количество строк и столбцов, содержащихся в массиве:

тип имя_массива[*количество_строк*][*количество_столбцов*]

При обращении к элементам массива *значения индексов всегда должны быть меньше соответствующих размерностей, указанных при объявлении массива*. Максимально возможное значение индекса должно быть на единицу меньше соответствующей размерности, заявленной при объявлении массива (так как отсчёт значений индексов ведётся с 0).

Примеры:

```
double A[2][4], B[10][10];
char S[5][20];
int Num[3][4];
```

Здесь описаны массив вещественных чисел *A* размерностью в 2 строки на 4 столбца (2x4), массив вещественных чисел *B* размерностью в 10 строки на 10 столбцов (10x10), массив символов *S* размерностью в 5 строк на 20 столбцов (5x20) и массив целых чисел *Num* размерностью в 3 строки и 4 столбца (3x4).

При описании массива для него резервируется место в оперативной памяти. Элементы массива располагаются в памяти по строкам последовательно один за другим. Пример размещения в памяти массива *t* представлен на Рис.11.1.

Рис.11.1. Размещение в памяти элементов двухмерного массива *Num* размерности 3x4.

В C++ предусмотрена возможность инициализации двухмерных массивов, т.е. присвоение начальных значений элементам массива при его объявлении. Для этого сразу после описания массива ставится знак "=" и внутри фигурных скобок последовательно перечисляются значения элементов в том порядке, как они располагаются в массиве. Инициализация массива целых чисел *Num* размерностью 3x4 значениями от 0 до 11 производится следующим образом

```
int Num[3][4] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11};
```

При инициализации двухмерного массива список инициализаторов каждой размерности можно заключить в фигурные скобки. Тогда ещё один вариант записи предыдущего объявления для массива *Num* будет иметь вид:

```
int Num[3][4] = {
      { 0, 1, 2, 3,}
      { 4, 5, 6, 7,}
      { 8, 9, 10, 11}
};
```

11.1.2. Использование двумерных массивов в качестве параметров функций

Если двухмерный массив используется в качестве параметра функции, то в неё передастся только указатель на его первый элемент. При этом обязательно нужно указать количество столбцов в массиве. Можно также задать и количество строк, но это не обязательно. Количество столбцов необходимо компилятору для того, чтобы правильно вычислить адрес элемента массива внутри функции, а для этого должна быть известна длина строки. Например, функция, получающая в качестве параметра двухмерный массив, состоящий из 10 строк и 10 столбцов, может выглядеть так

```
void funar(int x[][10])
{
x[2][4] = 5;
}
void main(void)
{
 int A[10][10];
 . . .
 funar(A);
 . . .
}
```

Компилятор C++ должен знать количество столбцов, иначе он не сможет правильно вычислять выражения, подобные следующему:

```
x[2][4] = 5;
```

Если бы длина строки была неизвестна, то компилятор не нашёл бы начало второй строки.

Так как в качестве параметра при вызове функции *funar* передаётся указатель на начало массива *A*, то все изменения элементов массива *x* внутри этой функции напрямую скажутся на значениях элементов массива *A*. Т.е. после вызова функции *funar* элемент *A*[2][4] получит значение 5.

11.1.3. Примеры работы с двумерными массивами

Для обработки данных, содержащихся в элементах двухмерного массива, обычно используются циклы, последовательно перебирающие значения его индексов.

Организуем ввод значений элементов массива целых чисел *A* размерности *N*x*M* с помощью клавиатуры.

```
#include <stdio.h>
void input(int N, int M, int A[][100])
{
 int i, j;
 for(i=0; i<N; i++)
    for(j=0; j<M; j++) {
      printf("A(%d,%d) = ",i,j);
```

```
      scanf("%d", &A[i][j]);
   }
}

void main(void)
{
 int N, M, A[100][100];

 do {
   printf("Введите количество строк N= "); scanf("%d", &N);
   printf("Введите количество столбцов M= "); scanf("%d", &M);
} while(N<=0 || N>100 || M<=0 || M>100);

 input(N, M, A);
. . .
}
```

Здесь в основной программе *main* описан массив *A* с заведомо большой размерностью 100x100. Затем программа запрашивает текущие значения размерностей *N* (0< *N* ≤100) и *M* (0< *M* ≤100). Функция *input* осуществляет ввод значений в массив *A* размерностью *N*x*M* с помощью клавиатуры. В данной программе с запасом резервируется память под массив размерности 100x100, а затем для работы используется лишь её необходимая часть размерностью *N*x*M*. Функция *input* использует в качестве индексов переменные *i* и *j*, которые пробегают последовательно значения, соответственно, от 0 до *N* и от 0 до *M*, предоставляя доступ к нужным элементам массива *A*.

Рассмотрим пример решения задачи вычисления суммы всех элементов массива целых чисел *Num* размерностью 3x4.

```
#include <stdio.h>
#include <conio.h>
int Sum(int N, int M, int A[][100])
{
 int i, j, S;

 S = 0;
 for(i=0; i<N; i++)
    for(j=0; j<M; j++) S +=A[i][j];
 return S;
}
void main(void)
{
 int Num[100][100];
 int N=3, M=4;

 input(N, M, Num);
 int s = Sum(N, M, Num);
 printf("Сумма элементов равна %d", s);
 getch();
}
```

11.1.4. Многомерные массивы

В C++, помимо двумерных, можно определять массивы трёх и более измерений. Вот как объявляется многомерный массив.

тип имя_массива[*размерность*_1][*размерность*_2] ... [*размерность_N*];

Например, с помощью следующего объявления создаётся трёхмерный целочисленный массив Multi размером 4x10x3.

int Multi [4][10][3];

Массивы с числом измерений, превышающим три, используются редко, так как работать с большим числом размерностей затруднительно и для их хранения требуется большой объем памяти. Например, хранение элементов четырёхмерного символьного массива размером 10x6x9x4 займёт 2160 байт.

11.2. Варианты заданий

Задана матрица *A* вещественных чисел размерностью *N*x*N*. Значение переменной *N* в диапазоне от 1 до 10, а также значения элементов матрицы *A* вводятся с помощью клавиатуры.

1. Написать функцию, осуществляющую вычисление суммы *S*, всех значений элементов матрицы *A*, находящихся на её диагоналях. Сформировать матрицу *B*, где $b_{i,j} = \begin{cases} a_{i,j}, & \text{если } a_{i,j} < S \\ 0, & \text{если } a_{i,j} \geq S \end{cases}$. Вывести на экран содержимое матрицы *B*.

2. Написать функцию, осуществляющую вычисление минимального и максимального значений (*Min* и *Max*) среди элементов матрицы *A*, находящихся на её главной диагонали. Сформировать матрицу *B*, где $b_{i,j} = \begin{cases} a_{i,j}, & \text{если } Min \leq a_{i,j} \leq Max \\ -1 \text{ в противном случае} \end{cases}$.
Вывести на экран содержимое матрицы *B*.

3. Найти максимальные значения в каждом из столбцов матрицы *A* и сформировать из них вектор *B*. Вывести на экран содержимое вектора *B*.

4. Написать функцию, осуществляющую вычисление среднего значения *Sred* для элементов матрицы *A*. Сформировать матрицу *B*, где $b_{i,j} = \frac{a_{i,j}}{Sred}$. Вывести на экран содержимое матрицы *B*.

5. Задать матрицы вещественных чисел *B* и *C* с размерностями *N*x*N*. Ввести значения элементов матрицы *B* с помощью клавиатуры. Сформировать матрицу *C* таким образом, чтобы $c_{i,j} = (a_{i,j} + b_{i,j})/a_{0,j}$. Вывести на экран содержимое матрицы *C*.

6. Задать вектор вещественных чисел *B*, состоящий из *N* элементов и матрицу вещественных чисел *A* размерностью *N*x*N*. Ввести значения элементов вектора *B* и матрицы *A* с помощью клавиатуры. Сформировать вектор *C* таким образом, что $c_i = \frac{b_i}{\sum_{j=0}^{N-1}|a_{i,j}|}$. Вывести на экран содержимое вектора *C*.

7. Задать матрицу вещественных чисел *A* размерностью *N*x*N*. Сформировать матрицу *B* таким образом, что $b_{i,j} = \frac{a_{i,j}}{\sum_{j=0}^{N-1} a_{i,j}}$. Вывести на экран содержимое матрицы *B*.

8. Задать матрицу вещественных чисел *B* размерностью *N*x*N*. Сформировать матрицу *B* таким образом, что $b_{i,j} = \frac{a_{i,j}}{\sum_{i=0}^{N-1} a_{i,j}}$. Вывести на экран содержимое матрицы *B*.

9. Создать функцию, которая формирует одномерный массив *B* из сумм положительных элементов строк матрицы *A*. Вывести на экран содержимое массива *B*.

11.3. Порядок выполнения работы

1. Для полученного варианта составить программу на C++ для решения поставленной задачи. В программе предусмотреть вывод исходного массива в форме таблицы и результата в наглядной форме.

2. Отладить программу и выполнить задание на компьютере;

3. Составить отчет о проделанной работе на ПК в виде файла. В отчете необходимо отразить следующие пункты:

- Постановка задачи.
- Алгоритм решения (блок-схема).
- Текст программы.
- Описание программы.
- Контрольный пример.
- Результаты счета.
- Анализ результатов и выводы.

4. При сдаче лабораторной работы студент должен показать и объяснить результаты выполнения задания на компьютере, ответить на вопросы преподавателя по теме лабораторной работы.

СПИСОК ЛИТЕРАТУРЫ

1. Маликов А.И. Конспект лекций по информатике и информационным технологиям. I. Казань, КНИТУ-КАИ, 2014.
2. Информатика. Базовый курс. Учебное пособие для ВТУЗов / Под ред. С. В. Симоновича. СПб.: Питер, 2009. (100 экз). Интернет ресурс http://www.twirpx.com/file/126065/
3. Информатика. Базовый курс : для бакалавров и специалистов: учеб. пособие для студ. вузов/ под ред. С. В. Симоновича. СПб.: Питер, 2012.
4. Макарова Н.Б., Волков В.Б. Информатика: для бакалавров. Учебник для студ. ВУЗов. С-П.:Питер, 2011.576 с. Интернет ресурс http://www.twirpx.com/file/748567/
5. Павловская Т.А. Программирование на языке высокого уровня. Учебник для вузов. СПб.: Питер, 2009. 432 с.
6. Касюк С.Т. Курс программирования на языке СИ. Конспект лекций. Челябинск.: Издательский центр ЮУрГУ. 2010. 175 с.
7. Маликов А.И., Бушманова И.В., Сюняев А.Я., Яфасов Ф.И. Информатика. Обработка данных на ПК. Лабораторный практикум. Казань: КГТУ им.А.Н.Туполева, 2006 . 130 с.
8. Маликов А.И., Сюняев А.Я., Хайруллин В.Р., Яфасов Ф.И. Информатика. Решение вычислительных задач на ПК. Лабораторный практикум. Казань: КГТУ им.А.Н.Туполева, 2007. 166 с.
9. Единая система программной документации: ГОСТ 19.002-80 Схемы алгоритмов и программ. Правила выполнения (Заменен на ГОСТ 19.701-90 (ИСО 5807-85)); ГОСТ 19.003-80 Схемы алгоритмов и программ. Обозначения условные графические (Заменен на ГОСТ 19.701-90 (ИСО 5807-85)); ГОСТ 19.004-80 Термины и определения (Заменен на ГОСТ 19.781-90); ГОСТ 19.401-78 Текст программы. Требования к содержанию и оформлению; ГОСТ 19.402-78 Описание программы; ГОСТ 19.404-79 Пояснительная записка. Требования к содержанию и оформлению.

Приложение 1. Варианты контрольных заданий

1). Массив $a(n)$ $n \le 25$ пронормировать, чтобы $\sum_{i=1}^{n} a_i^2 = 1$, т.е. разделить каждое a_i на сумму квадратов всех элементов массива.

2). Дан массив d(n) $n \le 20$. Указать (напечатать) номера положительных элементов, после которых следуют отрицательные.

3). В массиве $a(50)$ произвести перестановку, сгруппировав в начале массива все положительные числа. Вывести на печать только положительные числа по 5 штук в строке.

4). Из массива $a(100)$ в массив $b(100)$ переписать элементы, модуль которых меньше среднего значения всех элементов массива a. Вместо остальных элементов в массив b записать нули. Массив b вывести на печать по 10 чисел в строку.

5). В массиве $dd(200)$ указать номер элемента (со 2 по 199), сумма которого с предыдущим и последующим имеет максимальное значение.

6). Из массива $d(100)$ в массив $dd(20)$ записать средние значения 5 последовательных элементов массива d. Вывести на печать dd по 4 элемента в строку.

7). Из массива $JJ(100)$ в массив $NN(100)$ перенести числа (элементы массива) сначала нечетные, а затем четные. Вывести массив NN на печать по 10 элементов в строке.

8). Дан массив $d(20,4)$. Сформировать массив $c(4,4)$ в каждую строку которого записана сумма 5 строк из d. Вывести c на печать в виде матрицы 4×4.

9). Из массива z(20,20) в массив $d(20)$ переписать наибольшие по модулю элементы каждого столбца z. Результат вывести на печать.

10). Вычислить вектор $b=(b_i)$, $i = 1,...,n$, $n \le 10$, по заданной матрице $z=(z_{ij})$, $i,j = 1,...,n$ и заданному x:

$$b_i = \sum_{j=1}^{n} z_{ij} x^j \quad i = 1,\ldots,n$$

11). Массив $d(60)$ разделить на три отдельных массива по принципу

$d_1 = (d_1, d_4, d_7, ...)$

$d_2 = (d_2, d_5, d_8, ...)$

$d_3 = (d_3, d_6, d_9, ...)$

Вывести d_1, d_2, d_3 в три столба

d_1 d_2 d_3

12). Найти максимальные элементы матрицы $d(n,n)$, $n \le 20$ среди элементов, расположенных выше диагонали .

13). Вычислить суммы всех элементов массива $f(n,n)$, $n \le 50$, расположенных выше и ниже его главной диагонали.

14). Вычислить значение функции z по заданному массиву $q(n,n)$, $n \le 15$ и массиву $x(n)$, $z = \max\limits_{i=1,n} \sum\limits_{j=1}^{n} q_{ij} x^j$.

15). Вычислить значение функции y по заданному x и матрице $d=(d_{ij})$ $i,j=1,\ldots,n$, $n \le 10$,

$$y = \sum_{i=1}^{n} \max_{j=1,n} d_{ij} x^{n-i} .$$

16). Вычислить величину $x_1x_n+x_2x_{n-1}+\ldots+x_nx_1$, где x_i – максимальный элемент i-ой строки матрицы $a=(a_{ij})$, $i,j=1,\ldots,n$, $n \le 30$.

17). Вычислить величину $x_1x_n+x_2x_{n-1}+\ldots+x_nx_1$, где x_i – минимальный элемент j-ого столбца матрицы $b=(b_{ij})$, $i,j=1,\ldots,n$, $n \le 30$.

18). Вычислить значение f по заданным массивам $c(n,n)$, $d(n,n)$ и заданным x и y

$$f = \begin{cases} \min\limits_{i} \sum\limits_{j=1}^{n} c_{ij} x^j & \text{при } x < y \\ \sum\limits_{i=1}^{n} \min\limits_{j} (d_{ij}) y^i & \text{при } x > y \end{cases}$$

19). Для матрицы $g(10, 10)$ найти сумму отрицательных элементов главной диагонали и их количество.

20). Из массива $a(50, 50)$ в массив $d(2500)$ переписать все элементы массива a, модуль которых меньше 1. Результат вывести на печать .

21). Из массива q сформировать массив $d(20, 20)$ в котором на главной диагонали расположить элементы массива в порядке возрастания, а остальные элементы обнулить.

22). Для массива $d(20, 20)$ найти и вывести на печать номера всех элементов, равных 0, 1 и –10.

23). Определить и вывести на печать все номера элементов, максимальных в каждой из столбцов матрицы $q(25, 25)$.

24). Из массива $d(100)$ в массив $t(20)$ записать средние значения 5 последовательных элементов массива d. Результат вывести на печать.

25). По заданному массиву $r(10, 10)$ сформировать массив $s(45)$, в котором разместить элементы массива r, расположенные под главной диагональю (по строкам).

26). В массиве $d(200)$ указать номер элемента (со 2 по 199), сумма которого с предыдущим и последующим имеет минимальное значение.

27). Для матрицы $z(10, 10)$ найти максимальный элемент на главной диагонали и определить сумму 9-ти элементов с ним в центре.

28). Из массива $J(100)$ в массив $N(100)$ перенести числа (элементы массива) сначала нечетные, а затем четные. Результат вывести на печать.

29). В массиве $s(100, 100)$ поменять местами минимальный и максимальный элементы и вывести их номера на печать.

30). Из массива $d(100)$ в массив $g(100)$ переписать положительные элементы, в массив q —отрицательные элементы, и определить количество нулевых элементов. Результат вывести на печать.

31). Для массива $g(10, 10)$ найти сумму всех положительных, сумму всех отрицательных элементов и определить количество нулей. Результат вывести на печать.

32). По заданному массиву $q(10, 10)$ сформировать одномерный массив $s(100)$, в котором расположить элементы q_{ij} в порядке возрастания.

33). Вычисление $f(m,n) = n!m!/(n+m)!$, где n и m —неотрицательные целые числа, с определением и без определения функции, вычисляющей факториал. Сравнить получаемые результаты и определить какая программа позволяет получить результат для больших m и n.

34). Сформировать массив $z=(z_i)$

$$z_i = (sign(x_i) + sign(y_i)) \cdot sign(x_i + y_i),$$

где $sign(x) = \begin{cases} -1 & x < 0 \\ 0 & x = 0 \\ +1 & x > 0 \end{cases}$, x_i, y_i, z_i – элементы массивов x,y,z размерности $n \le 20$.

35). Найти тройки чисел из множества, заданного массивами x, y, z размерности $n \le 10$, для которой площадь треугольника со сторонами, определяемыми этой тройкой, будет минимальна.

36). Вычислить элементы массива $A(n)$, $n \le 10$ по членам разложения функции e^x в ряд Маклорена $e^x = 1+x+x^2/2!+x^3/3!+...+x^n/n!$.

37). Вычислить элементы массива $d[i,j]$, $i,j=1,\ldots,n$, по заданному вектору $x=(x_i)$, $i=1,\ldots,n$, $n \le 20$ и членам разложения функции $sin(x_i)$ в ряд Маклорена $sin(x_i)=x_i—x_i^3/3!+x_i^5/5!–x_i^7/7!+x_i^9/9!$ ...:

$d_{i1} = x_i$, $d_{i2} = x_i^3/3!$;...

38). По заданной дате (день и месяц года) вывести на печать название соответствующего знака Зодиака:

20.1 – 18.2 – Водолей	23.7 – 22.8 – Лев
19.2 – 20.3 – Рыбы	23.8 –22.9 – Дева
21.3 – 19.4 – Овен	23.9 –22.10 – Весы
20.4 – 20.5 – Телец	23.10 –22.11 – Скорпион
21.5 – 21.6 – Близнецы	23.11 –21.12 – Стрелец
22.6 – 22.7 – Рак	22.12 – 19.1 – Козерог

39). Для заданного списка студентов группы, включающего фамилию, номер группы и год рождения, определить средний возраст студентов на данный момент.

40) Вычислить матрицу $F = (BCB^T – A^TC)$, где A-$n \times n$, B-$n \times n$, C-$n \times n$ - заданные матрицы.

41). Сформировать одномерный массив. Удалить из него элемент с заданным номером, добавить элемент с заданным номером;

42). Сформировать одномерный массив. Удалить из него элемент с заданным ключом, добавить элемент с заданным ключом;

43). Сформировать одномерный массив. Удалить из него К элементов, начиная с заданного номера, добавить элемент с заданным ключом;

44). Сформировать одномерный массив. Удалить из него элемент с заданным номером, добавить К элементов, начиная с заданного номера;

45). Сформировать одномерный массив. Удалить из него К элементов, начиная с заданного номера, добавить К элементов, начиная с заданного номера.

46). Сформировать двумерный массив. Удалить из него строку с заданным номером;

47). Сформировать двумерный массив. Удалить из него столбец с заданным номером;

48). Сформировать двумерный массив. Добавить в него строку с заданным номером;

49). Сформировать двумерный массив. Добавить в него столбец с заданным номером;

50). Сформировать двумерный массив. Удалить из него строку и столбец с заданным номером.

51). Сформировать двумерный массив. Добавить в него строку и столбец с заданным номером.

52). Сформировать двумерный массив. Удалить из него все строки, в которых встречается заданное число.

53). Сформировать двумерный массив. Удалить из него все столбцы, в которых встречается заданное число.

54). Сформировать двумерный массив. Удалить из него строку и столбец, на пересечении которых находится минимальный элемент.

55). Сформировать двумерный массив. Удалить из него строку и столбец, на пересечении которых находится максимальный элемент.

56). Сформировать массив строк. Удалить из него самую короткую строку.

57). Сформировать массив строк. Удалить из него самую длинную строку.

58). Сформировать массив строк. Удалить из него строку, начинающуюся на букву "а".

59). Сформировать массив строк. Удалить из него строку, начинающуюся и заканчивающуюся на букву "а".

60). Сформировать массив строк. Удалить из него строку, начинающуюся и заканчивающуюся на одну и ту же букву.

61). Сформировать массив строк. Удалить из него строку с заданным номером.

62). Сформировать массив строк. Удалить из него k строк, начиная со строки с заданным номером.

63). Сформировать массив строк. Удалить из него одинаковые строки.

64). Сформировать массив строк. Удалить из него k последних строк.

65). Сформировать массив строк. Удалить из него k первых строк.

66). Сформировать массив строк. Добавить в него строку с заданным номером.

Приложение 2. Требования к оформлению отчета по лабораторным работам

1. Объем и содержание отчета

Отчет оформляется с помощью текстового редактора Word и представляется в распечатанном виде на бумажном носителе и в электронном виде на дискете. Отчет должен давать достаточно полное представление о методе решения инженерной задачи с обоснованием правильности решения на ПК. Отчет иллюстрируется блок-схемами алгоритмов и текстами программ, выполненными с соблюдением требований Единой системы программной документации (ЕСПД) [7].

Объем отчета по одной теме (лабораторной работе) не должен превышать 5-7 страниц.

Отчет начинается с титульного листа и должен включать по каждой теме (лабораторной работе) следующие разделы: название темы (лабораторной работы), номер варианта, оглавление отчета, задание, введение; разделы и подразделы основной части; заключение; список литературы; приложения (при необходимости).

Титульный лист должен соответствовать установленному образцу (см. приложение 3).

Содержание основных разделов отчета следующее.

В в е д е н и е содержит анализ актуальности и цели решения задачи (задание выполняется ради наблюдения некоторого эффекта, изучения типичной особенности; задание моделирует распространенную на практике ситуацию и т.п.); какие средства используются; если формулировка задания оставляет свободу выбора каких-либо методов или параметров, произвести и обосновать этот выбор. По заданию отмечаются начальная ситуация (что дано, известно) и конечный результат (подчеркнуть, что будет изменено, дополнено или удалено в ходе выполнения задания). Во введении дается краткий анализ возможных методов решения поставленной задачи, но так, чтобы он не заслонял основного содержания работы. Указываются литературные источники, по которым делается обзор, позволяющий судить, насколько полно изучена литература по методам решения задачи.

В основной части должны быть отражены следующие разделы:

1. Постановка задачи;
2. Метод решения;
3. Алгоритм решения задачи;

4. Текст программы на алгоритмическом языке или языке пакета прикладных программ;
5. Описание программы;
6. Исходные данные;
7. Результаты счета;
8. Анализ полученных результатов и выводы.

В данных разделах рассматривается существо задачи, дается аналитический обзор возможностей решения поставленной вычислительной (инженерной задачи), обоснование выбранного метода решения, описание (концептуальной, физической, математической, информационной) модели объекта исследования, формализацию и алгоритмизацию метода решения, описание выбранного математического и программного обеспечения, описание алгоритмов и программ решения задачи на ПК, результаты решения, анализ полученных результатов, выводы по использованию полученных результатов для исследования и разработки рассматриваемого объекта, оценка погрешностей вычислений, пути их уменьшения.

При составлении отчета следует стремиться к сжатому, но четкому изложению; ясно должна быть видна логика действий и изменения в состоянии системы, произошедшие в результате выполнения задания. Необходимо четко понимать, что негативный эффект (снижение оценки) в равной степени вызывают как отсутствие необходимой информации, так и наличие лишней информации, загромождающей отчет. Избыточные тексты в отчетах наказываются штрафом, пропорциональным избыточности (до 50 баллов). Руководствуйтесь критерием: отчет должен быть понятен не только тому, кто его составил, но и читателю, являющемуся достаточно подготовленным специалистом в этой области (например, преподавателю).

Отчет должен содержать тексты программ и их описание. В описании программы приводятся следующие сведения: назначение программы, структура программы (из каких блоков, процедур и функций она состоит и их назначение), параметры программы (параметры процедур и функций) и их назначение, какие данные и в какой последовательности требуется вводить при запуске программы на выполнение, куда будет выдаваться результат и в какой форме. При большом объеме текст программы выносится в приложение к отчету. Результаты компьютерного эксперимента должны быть представлены в форме графиков, диаграмм, таблиц и т.д. Приводится анализ полученных результатов решения задачи на ПК. При несовпадении расчетных, аналитических, экспериментальных результатов необходимо выяснить и объяснить причины расхождения.

З а к л ю ч е н и е должно содержать обзор выполненных работ, качественные и количественные оценки результатов расчета; затрат машинного времени на расчеты и требуемого объема памяти ПК для программ. Здесь же приводятся пояснения к этапам выполнения задания, отмечаются проблемы, которые возникли при выполнении задания и как они были решены; пере-

числяются достигнутые результаты (выделить наблюдаемые изменения и/или основные параметры, характеризующие решение). Следует представить краткий вывод по результатам решения задачи, отметить достоинства и недостатки выбранного способа решения, алгоритма и программы. Если в процессе решения задачи был выбран не оптимальный способ, то следует указать причины, обусловившие такое решение, а также нерешенные вопросы, рекомендации по возможному в дальнейшем усовершенствованию алгоритма и программы решения задачи.

Список использованной литературы содержит перечень источников, использованных при выполнении лабораторной работы. Указывают только те источники, на которые имеются ссылки в тексте отчета.

Приложение содержит вспомогательный материал (тексты программ, инструкции по их использованию и т.п.) если их объем превышает 2-3 страницы.

2. Оформление отчета

Отчет по лабораторной работе набирается на ПК и распечатывается на одной стороне листа бумаги формата А4 (210х297 мм) при книжной ориентации. При этом необходимо оставлять поля: слева - 30 мм, справа - 10 мм, сверху - 20 мм, снизу 20 мм. Текст должен быть отпечатан на принтере, аккуратно, разборчиво, без помарок, с высотой букв не менее 2,5 мм. Разделы отчета, а также подразделы и пункты должны иметь порядковые номера, обозначенные арабскими цифрами с точкой в конце. Введение и заключение не нумеруются.

Заголовки разделов пишут прописными буквами по середине страницы. Заголовки подразделов пишут с абзаца, отступая слева 15 мм, строчными буквами (кроме первой прописной). В заголовке не допускаются переносы слов. Пробелы над заголовками и под ними - 6 пп. Точку в конце заголовка не ставят, заголовок не подчеркивают. Если заголовок состоит из двух предложений, то их разделяют точкой.

В отчете необходимо выдержать единые обозначения и размерности для используемых параметров, переменных и характеристик. Допускаются сокращения слов, терминов, обозначений, только общепринятых в ГОСТ 15133-77, 22348-77, 17657-79, 7.32-2001.

Иллюстрации (рисунки, таблицы, схемы) располагаются на отдельных страницах отчета. Согласно ЕСКД иллюстрации в отчете, кроме таблиц, имеют подпись "Рис. N". Номер рисунка N состоит из номера раздела и порядкового номера иллюстрации, разделенных точкой. Например, Рис.3.2. (второй рисунок третьего раздела). Иллюстрации снабжаются кратким подрисуночным текстом.

В отчете рисунки должны быть выполнены с использование графических редакторов (Paint и др.) или встроенного графического редактора MS

Word и внедрены в файл отчета. Рисунок располагают на той страницы, где на него дана первая ссылка.

Таблицы служат для оформления цифрового материала. Они приводятся после первого упоминания о них в тексте. На все таблицы должны быть ссылки в тексте, при этом слово "Таблица" в тексте пишут полностью, если таблица не имеет номера, и сокращенно - если имеет номер, например: "в табл.1.2". Каждая таблица должна иметь заголовок. Заголовок и слово "Таблица" начинают с прописной буквы.

Формулы в отчете (если их более одной) нумеруют арабскими цифрами в пределах раздела. Номер формулы состоит из номера раздела и порядкового номера формулы в разделе, отделенных точкой. Номер ставится с правой стороны листа на уровне нижней строки формулы в круглых скобках, например: (3.1) - первая формула третьего раздела. Ссылки на формулу указывают порядковым номером формулы в круглых скобках, например "...в формуле (2.1)".

Пояснение значений символов и числовых коэффициентов следует приводить непосредственно под формулой с новой строки со слова "где" без двоеточия после него в виде списка в той же последовательности, в какой они даны в формуле.

Уравнения и формулы следует выделять из текста свободными строками. Если уравнение не умещается в одну строчку, оно должно быть перенесено после знаков "=","+","–","x",":". При написании формул, выборе справочных данных, цитат необходимо делать ссылки на литературный источник, из которого они были заимствованы. Блок-схемы алгоритмов решения задачи и тексты программ следует оформлять в отчете в соответствии с требованиями ЕСПД [7].

При ссылке в тексте на используемую литературу указывается порядковый номер, выделенный двумя квадратными скобками по списку источников, например [2]. Литературу следует располагать в списке в порядке появления ссылок в тексте. Источник указывается в следующей форме: Фамилия и инициалы автора, полное название книги или статьи, место, издательство и год издания, объем (для журнала - название, год издания, номер, страницы).

Приложения оформляют как продолжение отчета или в виде отдельной части, располагая их в порядке появления ссылок в тексте. Каждое приложение следует начинать с нового листа с указанием в правом верхнем углу слова "Приложение 1","Приложение 2" и т. д., номера пишутся арабскими цифрами, далее следует тематический заголовок.

В отчете все страницы, в том числе титульный лист, содержание, листы с таблицами, рисунками, графиками, нумеруются арабскими цифрами. На титульном листе номер не ставят, на последующих страницах номер проставляют в правом верхнем углу.

Листы отчета с текстом, со схемами, листингами программ и другими иллюстрациями сшиваются вместе с титульным листом.

Приложение 3. Титульный лист к отчету

Отчет по лабораторным работам

дисциплина "ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ"

Выполнил студент группы _____ _____ _____

№ группы подпись Фамилия И.О.

Оценка _____

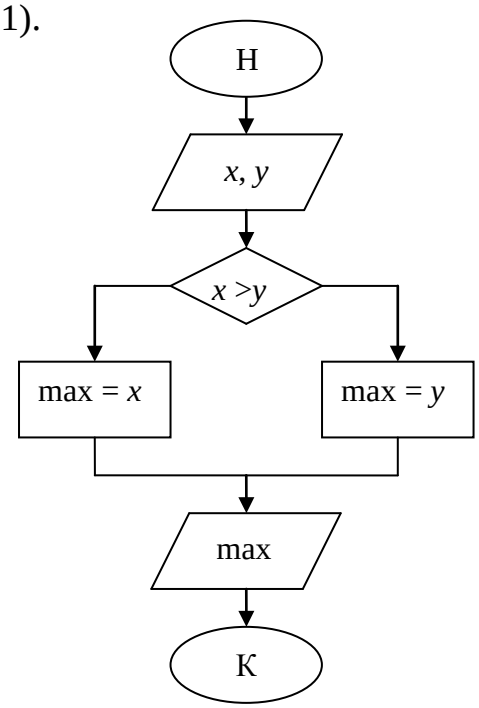
Преподаватель _____ _____

подпись Фамилия И.О.

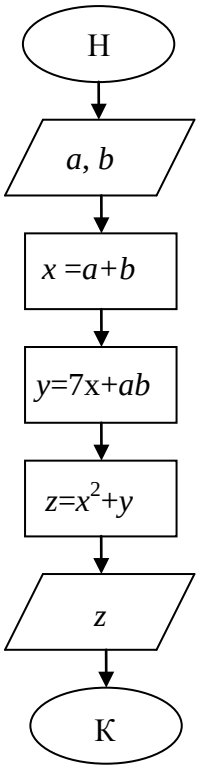
Казань, 2014

Приложение 4. *Блок-схемы алгоритмов к лабораторной работе №* 6

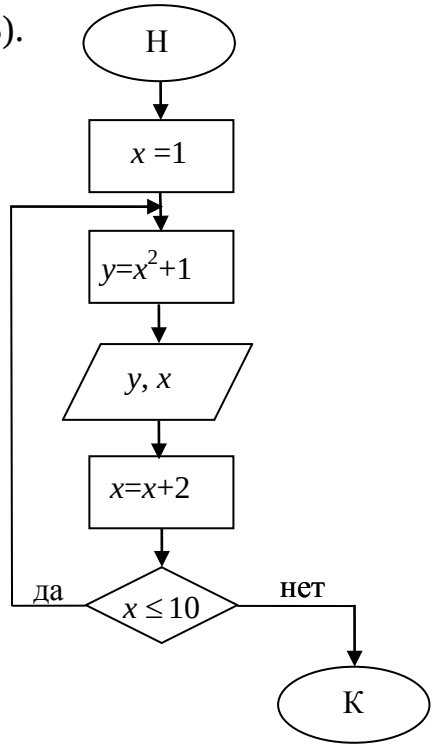
1).



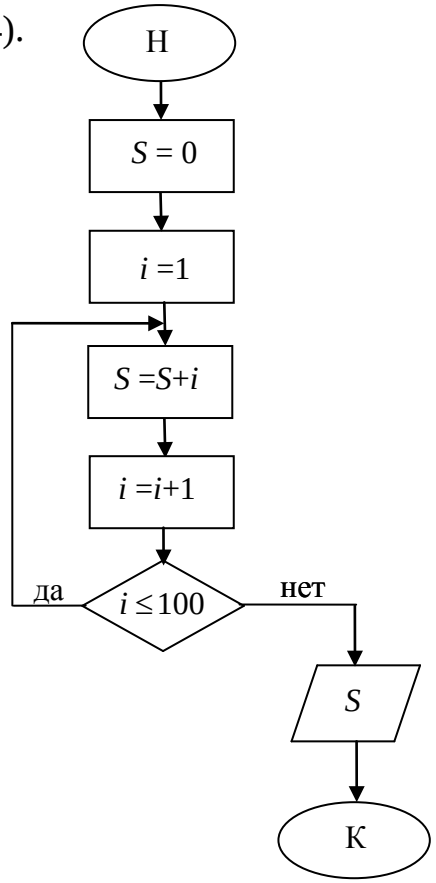
2).



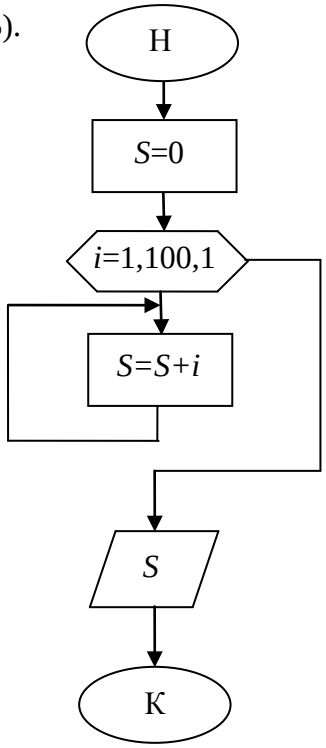
3).



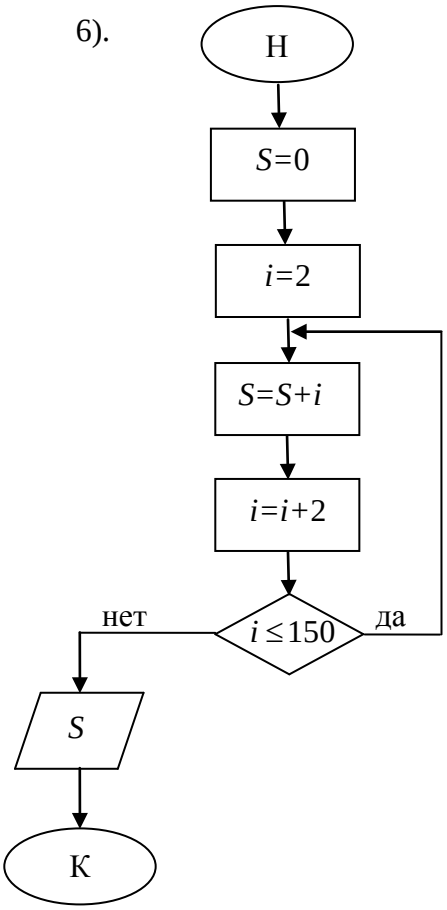
4).



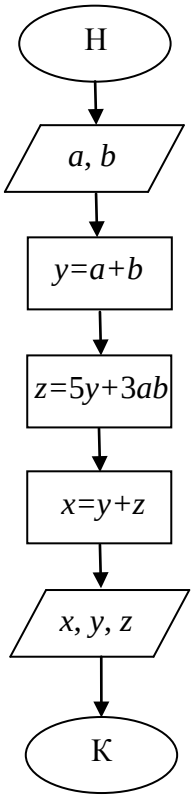
5).


Н
S=0
i=1,100,1
S=S+i
S
К

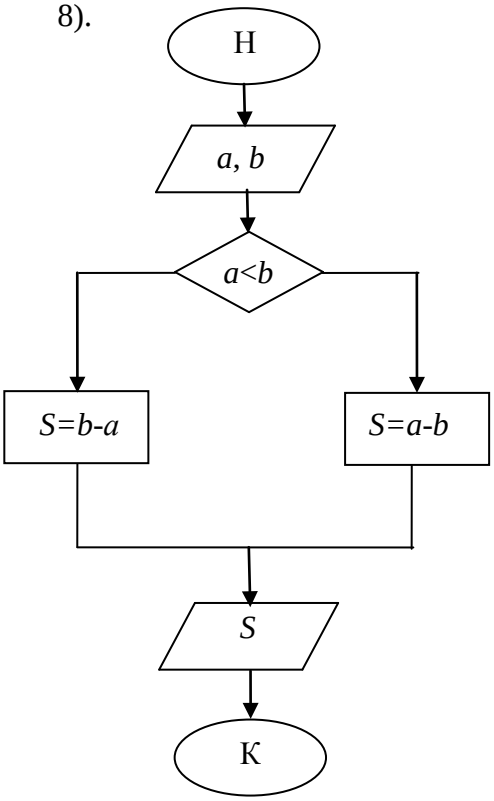
6).

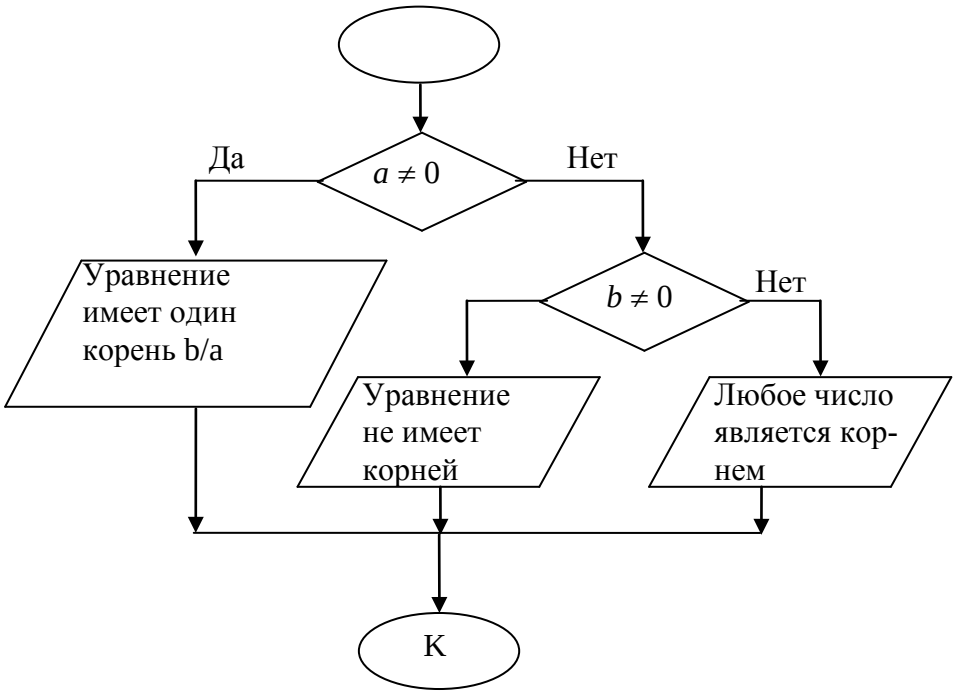

Н
S=0
i=2
S=S+i
i=i+2
нет
i ≤ 150
да
S
К

7).

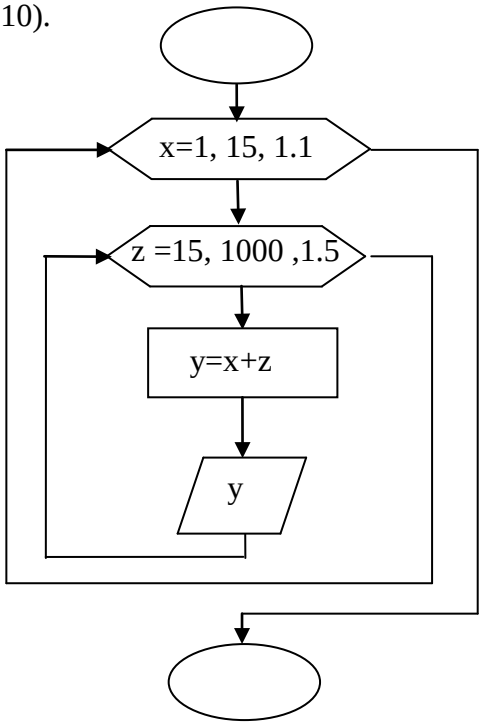

Н
a, b
y=a+b
z=5y+3ab
x=y+z
x, y, z
К

8).

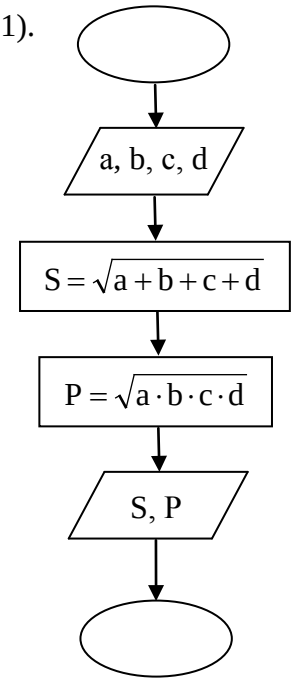

Н
a, b
a<b
S=b-a
S=a-b
S
К


Да
$a \neq 0$
Нет
Уравнение имеет один корень b/a
$b \neq 0$
Нет
Уравнение не имеет корней
Любое число является кор-нем
K

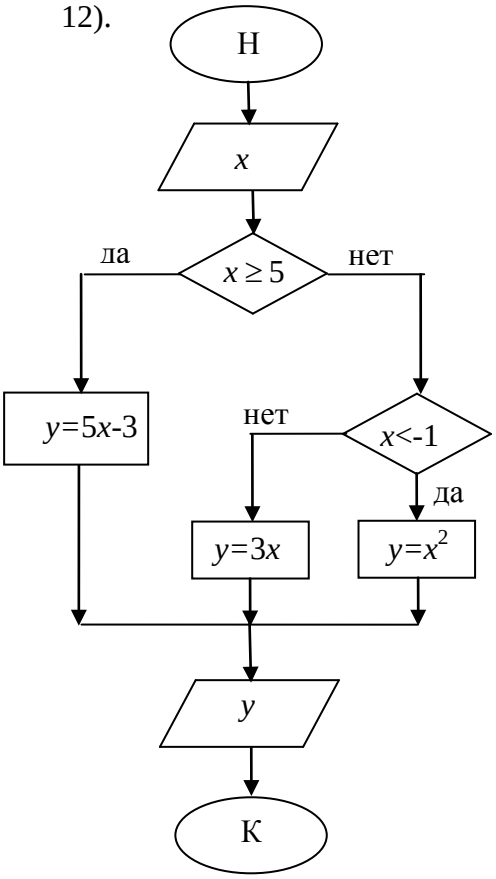
10).


x=1, 15, 1.1
z =15, 1000 ,1.5
y=x+z
y

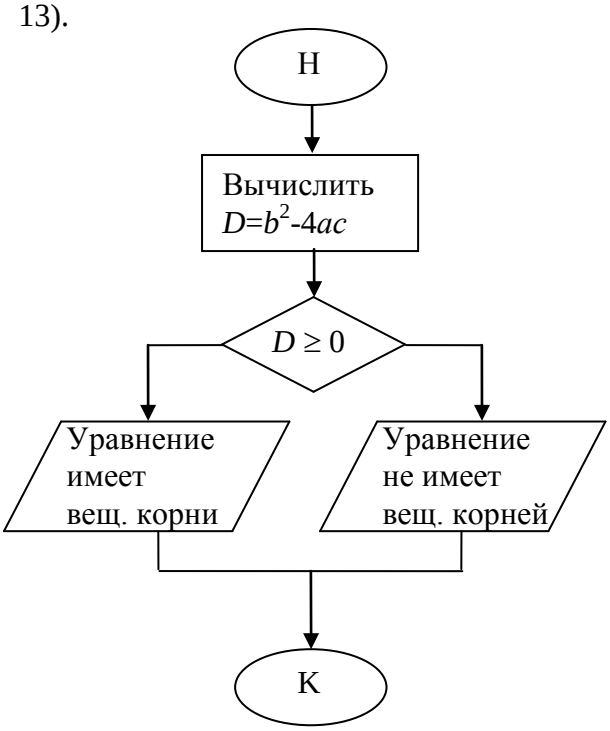
11).


a, b, c, d
$S = \sqrt{a + b + c + d}$
$P = \sqrt{a \cdot b \cdot c \cdot d}$
S, P

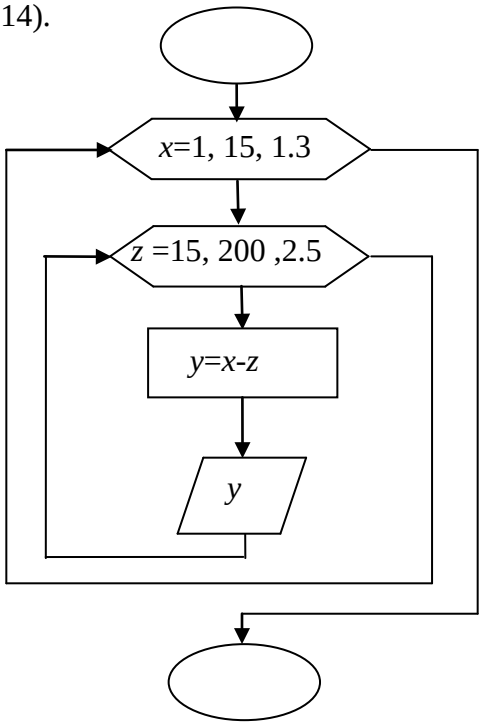
12).


Н
x
да
x ≥ 5
нет
y=5x-3
нет
x<-1
да
y=3x
y=x²
y
К

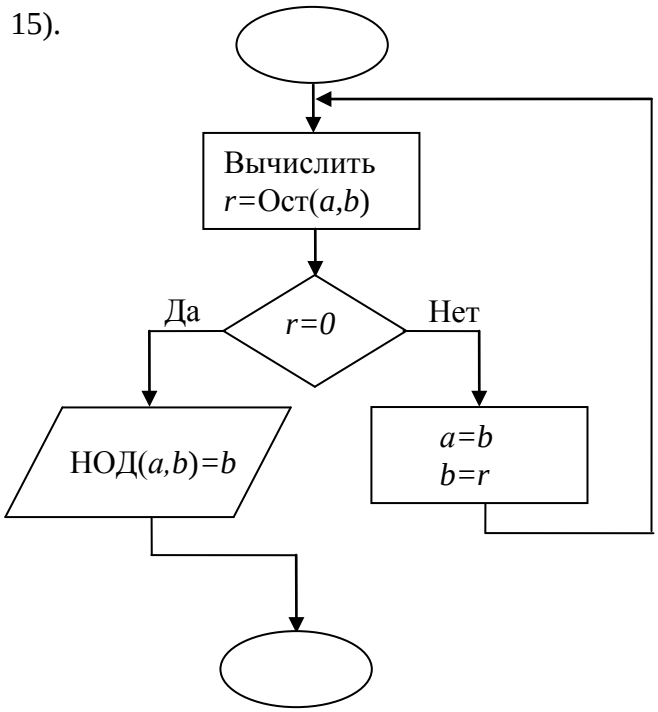
13).


Н
Вычислить
D=b²-4ac
D ≥ 0
Уравнение
имеет
вещ. корни
Уравнение
не имеет
вещ. корней
К

14).


x=1, 15, 1.3
z =15, 200 ,2.5
y=x-z
y

15).


Вычислить
r=Ост(a,b)
Да
r=0
Нет
НОД(a,b)=b
a=b
b=r

Приложение 5. *Текст программы на языке С для вычисления корней квадратного уравнения*

```
#include <conio.h>
#include <stdio.h>
#include <math.h>
int main(int argc, char* argv[])
{
  float a ,b ,c ,d , x1, x2;
              printf (" Enter  a= "); scanf ("%f",&a);
              printf (" Enter  b= "); scanf ("%f",&b);
              printf (" Enter  c= "); scanf ("%f",&c);
              if (a!=0) {     d=b*b-4*a*c;
                              if (d>=0) { x1=(-b-sqrt(d))/(2*a);
                                          x2=(-b+sqrt(d))/(2*a);
                                          printf ("x1= %6.3f  x2= %6.3f", x1, x2);
                                        }
                              else {printf("d<0, Eq has 2 complex roots");}
                        }
             else {
                   if(b!=0) {  x1 =-c/b;
                               printf (" x1 = %6.3f",  x1);
                               printf (" x2 not exists" );
                              }
                   else     {    if (c!=0) { printf (" Eq has no roots ");   }
                                 else      { printf (" x  arbitrary");}
                               }
                 };
      getch();
return 0;
}
```

Приложение 6. Текст программы на языке С для вычисления приближенного значения экспоненциальной функции

```
//Текст программы на языке С
//для вычисления приближенного значения экспоненциальной функции
//вычисление экспоненты суммированием членов ряда
//{exp= 1 + x/1! +  x 2/2! + ...}
#include <conio.h>       // библиотека ввода-вывода
Console I/O Routines (getch(), putch(),..)
#include <stdio.h>       // библиотека стандартного вывода-ввода
Console I/O Routines  (printf, scanf,..)
#include <math.h>        // библиотека мат функций
Math Routines (fabs(), sqrt(), exp()...)
int main(int argc, char* argv[])
{
 int k, kmax;
 float   x, a0, a1, s, eps;
//{ввод исходных данных}
 printf ("Enter x=");   scanf ("%f", &x);
```

```
 printf ("Enter eps="); scanf ("%f", &eps);
 printf ("kmax=");      scanf ("%d",&kmax);
//суммирование ряда
 k=1;
 s=1;
 a0=1;
 a1= a0*x/k;
while ((fabs(a1)>eps)&& (k<kmax)) {
                          printf("kmax= %d  k=%d  \n", kmax, k);
                          k= k+1;
                          s= s+a1;
                          a0= a1;
                          a1= (a0*x)/k;
                          };
// Печать
if(k==kmax) {// Точность не достигнута
            printf("within %d  iteration  exactness   eps= %12.8f  is not attained\n",  k , eps );
            printf("  exp(%f)=%12.8f", x, s );
           }
else       {// Точность достигнута
           printf("  exp(%f)=%12.8f", x, s );
           }
//Ждать нажатия клавиши Enter
    getch();
return 0;
}
```