

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ УКРАИНЫ
НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
"ХАРЬКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ"

А.И.Стрельченко

ИНФОРМАТИКА
для инженеров и экономистов

Учебное пособие

Утверждено
редакционно-издательским
советом университета

Харьков 2009

ББК 32.973
С84
УДК 004.42

Рецензенти: **Є.В. Путятин**, професор, докт. техн. наук, зав. кафедрою Інформатики Харківського національного університету радіоелектроніки
А.О. Медолазов, доцент, канд. техн. наук, доцент кафедри Природничих наук Харківського національного автомобільно-дорожнього університету
В.П. Степанов, професор, канд. техн. наук, зав. кафедрою ІКТ Харківського національного економічного університету

Навчальний посібник є вступом до інформатики для майбутніх інженерів, економістів і студентів інших напрямків, у навчанні та роботі яких можуть зустрітися елементи програмування. Посібник відповідає Базовій програмі з курсу "Інформатика та обчислювальна техніка", затвердженої Міністерством освіти і науки України для іноземних студентів підготовчих факультетів.

Призначено іноземним студентам підготовчих факультетів університетів і коледжів України. Може бути використаний у середній школі та при самостійному вивченні інформатики.

С84 Стрельченко А.Й . Інформатика для інженерів та економістів: Навчальний посібник/А.Й. Стрельченко. –Харків: НТУ "ХПІ", 2009. –150 с. –Рос. мовою.

ISBN 978-966-593-693-0

The manual is an introduction to computer science for future engineers, economists and students of other areas, where we can meet the elements of programming. The manual corresponds with the core curriculum of "Computer Science and Engineering", approved by the Ministry of Education and Science of Ukraine to foreign students of preparatory faculties.

Appointed for training foreign students at departments of universities and colleges in Ukraine. Can be used in high school and at self-study computer science.

Іл. 35 Табл. 6

ББК 32.973

ISBN 978-966-593-693-0

© А.Й. Стрельченко, 2009

© Оформлення та макет Д.С. Родних, 2009

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	7
Часть I. ОСНОВНЫЕ ПОНЯТИЯ	9
1. Введение в информатику	9
Что изучает информатика	10
Компьютерные сети	10
История	11
Разделы информатики	12
Применение ИТ	12
Контрольные вопросы	14
Лабораторная работа	14
Практические задания	15
2. Кодирование информации	16
Запись информации в компьютере	17
Кодирование информации	17
Количество информации	18
Проблемы кодирования	18
Кодирование чисел	19
Кодирование текста	20
Кодовая страница	20
Кодирование рисунков	21
Контрольные вопросы	21
Лабораторная работа	22
Практические задания	23
3. Структура компьютера	24
Конфигурация компьютера	25
Основные устройства	26
Процессор	26
Регистры и разрядность	27
Память	27
Принципы работы памяти	28
Работа компьютера	28
Программы	29
Работа программы	29
Контрольные вопросы	30
Лабораторная работа	31
Практические задания	31
4. Программы	32
Системные и прикладные программы	33
Интерфейс	33
Пользовательский интерфейс	34
Графический интерфейс	34
Курсорный интерфейс	35

Меню	35
Командный интерфейс	36
Комбинации клавиш.....	37
Контрольные вопросы.....	38
Лабораторная работа	39
Практические задания	40
5. Операционные системы	41
Что такое операционная система	42
Структура операционной системы	42
Типы операционных систем.....	43
Домашние и офисные ОС	43
Файлы.....	44
Каталоги (папки).....	45
Логические диски	46
BIOS	46
Маршрут	47
Контрольные вопросы.....	48
Лабораторная работа	49
Практические задания	49
6. Работа с дисками.....	50
Запись данных на диске	51
Таблицы расположения файлов.....	51
Таблицы FAT и NTFS	52
Корзина	52
Очистка диска	53
Дефрагментация диска	53
Форматирование диска	54
Архивирование данных.....	55
Архиваторы	55
Резервное копирование	56
Контрольные вопросы.....	57
Лабораторная работа	58
Практические задания	59
7. Компьютерные вирусы.....	60
Что такое вирус.....	61
Воздействие на компьютер.....	61
Вирусы в природе.....	62
Проявление вирусов	62
Типы вирусов	63
Антивирусные программы	63
Компьютерные вирусы и дисциплина.....	64
Контрольные вопросы.....	65
Лабораторная работа	65
Практические задания	66

Часть II. ОСНОВЫ ПРОГРАММИРОВАНИЯ.....	67
8. Основные понятия.....	67
Алгоритмические языки.....	68
Машинные коды.....	68
Язык ассемблера	69
Языки высокого уровня.....	69
Объектно-ориентированные языки.....	70
Языки символьной обработки	71
Среда программирования VBA	71
Контрольные вопросы	76
Лабораторная работа	77
Практические задания	78
9. Простейшие программы	79
Структура программы	80
Переменные	80
Тип переменной	81
Числа	82
Текст.....	83
Числовые и текстовые константы	83
Оператор присваивания	84
Программа 1	86
Программа 2	89
Контрольные вопросы	91
Лабораторная работа	92
10. Операторы управления и алгоритмы	94
Порядок выполнения	95
Логические выражения.....	95
Оператор If.....	97
Программа 1	98
Программа 2	99
Алгоритмы.....	101
Программа 3	103
Программа 4	104
Контрольные вопросы	106
Лабораторная работа	107
11. Циклы	109
Циклические алгоритмы	110
Программа 1	111
Оператор цикла Do...Loop.....	113
Программа 2	114
Оператор While ...Wend.....	115
Программа 3	116
Оператор For.....	117
Программа 4	118
Контрольные вопросы	119

Лабораторная работа	120
12. Массивы	121
Типы массивов	122
Программа 1	123
Массивы в MS Excel.....	124
Программа 2	125
Программа 3	126
Динамические массивы.....	128
Программа 4	129
Контрольные вопросы.....	131
Лабораторная работа	132
13. Процедуры и функции.....	134
Типы процедур.....	135
Описание и вызов процедуры	136
Видимость переменных	137
Программа 1	138
Программа 2	138
Массивы в процедурах.....	139
Программа 3	140
Функции.....	143
Программа 4	144
Контрольные вопросы.....	146
Лабораторная работа	147
Приложение 1	148
Приложение 2	149

ВВЕДЕНИЕ



Это учебное пособие является введением в информатику для студентов инженерных и экономических специальностей, которые в своей учебе и работе будут не только использовать компьютер, но и создавать для него небольшие программы – программировать.

Целью пособия является систематизация представлений учащихся об основах компьютерных технологий, архитектуре компьютера и программного обеспечения, основах алгоритмизации и программирования, а также выработка навыков создания простейших программ для компьютера.



Для изучения программирования в пособии выбран язык Visual Basic, который реализован в среде программирования VBA (*Visual Basic for Application*). VBA есть на каждом компьютере, где установлен Microsoft Office. Если Вы используете Word для написания текста или электронные таблицы Excel, Вы можете выполнить все программы, которые есть в этой книге, и создавать свои.

Язык программирования – алгоритмический язык, – синтаксически строгий язык. Это значит, что если Вы сделали ошибку, компьютер не будет выполнять задание, а сообщит Вам об ошибке и ее характере. Поэтому, если при выполнении некоторой программы появляется сообщение об ошибке, изучите его.

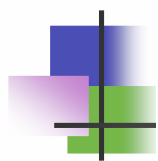


В учебном пособии изучаемый материал разбит на отдельные темы. Каждая из них содержит перечень ключевых слов по теме на русском, английском, китайском и арабском языках. Каждая тема содержит:

- теоретический материал по теме;
- контрольные вопросы;
- лабораторную работу, которую надо выполнить на компьютере;
- практические задания, которые проверяют уровень освоения учащимися навыков работы на компьютере.

Я попытался написать эту книгу как можно проще. Но если в некотором разделе Вы что-то не поняли, попробуйте сначала выполнить описываемые там программы и разобраться, как они работают. Такой способ изучения языка программирования я считаю наилучшим.

Темы, обозначенные звездочкой «*», при первом чтении пособия можно опускать.



Работа на компьютере дома или в лаборатории является наиболее важным этапом изучения информатики. Я думаю, что нельзя изучить информатику, не работая на компьютере.

Вместе с настоящим учебным пособием при изучении информатики полезно использовать учебно-экспертную систему «Эксперт-Центр», разработанную в НТУ "ХПИ". По каждому разделу курса эта система проверяет знание специальных терминов, предлагает контрольные вопросы по теории и практические задания.

Часть I. ОСНОВНЫЕ ПОНЯТИЯ

1. Введение в информатику



Ключевые слова

ВКЛЮЧИТЬ	turn on	打开	بدوره علي
ВЫКЛЮЧИТЬ	turn off	关掉	إيقاف
клавиша	key	键键	مفتاح
ярлык	icon	图标	ايقونه
выполнить	run	运行	البعيد
остановить	stop	停止	توقف
окно	window	窗	النافذة
открыть	open	打开	مفتوح
заккрыть	close	关闭	وثيق
найти	find	查找	بحث
сохранить	save	保存	انقاذ
обработать	process	处理	العملية
вычислить	calculate	计算	بسحا
преобразовать	transform	换算	تحول
послать	send	发送	أرسل
устройство	device	装置	أداة
аппаратура	hardware	硬件	الاجهزه
программа	software	软件	البرامج
сеть	network	网络	شبكة
локальная	local	本地	المحلية
глобальная	global	全球	عالمى
рабочий стол	desktop	桌面	المكتب
панель задач	taskbar	任务栏	شريط المهام



Что изучает информатика

- Информатика – это наука об информации.
- Информация – основное понятие. Оно не определяется. Информация обозначает знания, сведения, новости и др.
- Информацию можно *обрабатывать*: сохранять, находить, преобразовывать, передавать и др.
- ***Информатика изучает методы и устройства для обработки информации.***
- ***Компьютер – это устройство для обработки информации.***
- Компьютеры используются также для *обмена* информацией.
- Сегодня каждый, кто имеет компьютер и подключен к Интернету, может получить любую информацию из Всемирной сети.



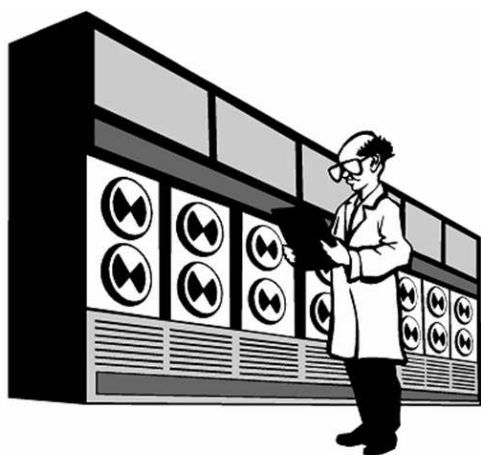
Компьютерные сети

- Компьютеры объединяют в *сети* для обмена и совместного использования информации:
 - *Локальная сеть* LAN объединяет компьютеры в одном здании или в близко расположенных зданиях через кабель (LAN – *Local Area Network*);
 - *Городская сеть* MAN объединяет компьютеры в городе через оптические кабели (MAN – *Metropolitan Area Network*);
 - *Глобальная сеть* WAN объединяет компьютеры через телефон или радио (WAN – *World Area Network*). Глобальная сеть – это Интернет.
- В глобальную сеть можно войти также через мобильный телефон.

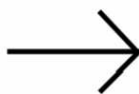


История

- Компьютер был создан как устройство для автоматических вычислений.
- Первый компьютер был построен в США (USA) в 1943 году. Его использовали при создании атомной бомбы.
- Первый компьютер в Европе был создан в Украине в Киеве в 1947 году.
- Первая компьютерная сеть была построена в 1965 году в Калифорнии (США).
- Первый миникомпьютер МИР-1 был создан в Киеве в 1966 году. Он был куплен фирмой IBM. В 1970 году в USA начали серийно выпускать миникомпьютеры PDP-11.
- Основной протокол Интернета TCP/IP был создан в 1985 году.



1965



1995

Рис. 1.1 Эволюция компьютера

- Устройство слева на Рис. 1.1 использовалось для хранения информации. Оно могло хранить меньше данных, чем один диск CD.
- Компьютер справа на Рис. 1.1 быстрее в 100 000 раз.



Разделы информатики

- Информатика включает в себя разделы:
 - **Вычислительная техника** (*Hardware*) изучает схемы компьютеров, как их строить и ремонтировать;
 - **Программирование** (*Software – Programming*) изучает, как управлять компьютером и создавать для этого программы;
 - **Моделирование** (*Brainware – Modeling*) изучает, как использовать компьютеры.
- На основе информатики создаются *Информационные Технологии* (ИТ – IT) для получения новых знаний и товаров во всех областях деятельности Человека.



Применение ИТ

- Применение информационных технологий эффективно потому, что ***компьютеры могут быстро обрабатывать большие объемы информации.***
- В науке компьютеры позволяют:
 - Моделировать процессы в термоядерном реакторе;
 - Моделировать наследственность – геном человека;
 - Моделировать сейсмические волны на Земле;
 - Моделировать погодуи многое другое.
- В сельском хозяйстве компьютеры позволяют быстро создавать новые сорта растений и породы животных:
 - Разработанные с помощью ИТ и генной инженерии новые сорта риса решили проблему питания в Китае, Индии и других странах мира;
 - Разработанная в Новосибирске программа ДИАС позволила создать десятки новых сортов пшеницы.

- В *промышленности* информационные технологии позволяют:
 - Создавать новые технологии и новые машины, которые производят товары высокого качества;
 - Автоматически управлять производственными процессами и целыми предприятиями (Рис. 1.2);
 - Создавать машины, которые работают там, где не может работать человек: в далеком космосе, глубоко под водой и под землей, в ядерных реакторах и многое другое.



Рис. 1.2 Работает автоматизированная система управления заводом, которую создали выпускники ХПИ

- В *медицине* компьютеры позволяют безошибочно ставить диагноз в самом начале заболевания и выбирать наилучший метод лечения.
- В *экономике* и управлении компьютеры позволяют принимать правильные решения.
- В *экономике* обработка информации – основной вид деятельности. Здесь компьютеры используются наиболее широко и эффективно.
- В *образовании* компьютеры и Интернет позволяют всем людям Земли получать любые знания.



Контрольные вопросы

1. Что такое информация?
2. Что изучает информатика?
3. Какие действия можно производить над информацией?
4. Что такое компьютер?
5. Когда был создан первый компьютер?
6. Почему компьютер называли компьютером?
7. Где был создан первый компьютер в Европе?
8. Что такое компьютерная сеть?
9. Что такое локальная сеть?
10. Что такое городская сеть?
11. Что такое Интернет?
12. Что такое электронная почта?
13. Что изучает вычислительная техника?
14. Что изучает программирование?
15. Что такое математическое моделирование?
16. Что такое информационные технологии?
17. Почему эффективно использование информационных технологий?



Лабораторная работа

1. Включите компьютер.
2. Запустите программу для создания текстовых документов Microsoft Word с помощью ярлыка на Рабочем столе.
3. Переключите компьютер в режим ввода текста на русском языке с помощью кнопки **Пуск** Панели задач.
4. Установите удобный масштаб с помощью окна "*Масштаб*" Панели инструментов программы.

5. Напишите в окне программы небольшой текст о себе: "Меня зовут ... Я приехал из... Я учусь на Факультете международного образования НТУ "ХПИ". После окончания университета я буду работать ...".
6. Исправьте в тексте все ошибки, которые обнаружил компьютер.
7. Сверните документ на Панель задач.
8. Покажите Вашу работу преподавателю.
9. Создайте личную папку на одном из компьютеров локальной сети. Компьютер и папку, где Вы ее разместите, согласуйте с преподавателем.
10. Сохраните созданный Вами документ в личной папке.
11. Выключите компьютер.



Практические задания

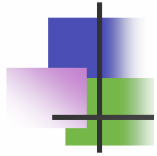
1. Правильно включить компьютер.
2. Запустить программу с помощью ярлыка на Рабочем столе.
3. Запустить программу с помощью Панели задач.
4. При работе программы выйти на Рабочий стол и запустить другую программу в многозадачном режиме.
5. При работе программы выйти на Рабочий стол, вывести контекстное меню этой программы и с его помощью остановить выполнение программы.
6. Переключить компьютер на ввод текста на украинском, русском или английском языке с помощью Панели задач.
7. Ввести в компьютер текст и запомнить его.
8. Правильно выключить компьютер.

2. Кодирование информации



Ключевые слова

непрерывный	continuous	继续	مستمر
импульсный	impact	影响	تأثير
единица	unit	单位	الوحدة
бит	bit	比特	بت
байт	byte	字节	بايت
килобайт	kilobyte	千字节	كيلوبايت
мегабайт	megabyte	兆字节	ميجابايت
гигабайт	gigabyte	千兆字节	جيجابايت
кодировать	code	代码	الرمز
символ	character	符号	الطابع
цифра	figure	图像	الرقم
число	number	数字	عدد
счисление	notation	注解	ترقيم
основание	base	基础	قاعدة
десятичный	decimal	小数点	عشري
шестнадцатеричный	hexadecimal	十六进制	ستعشري
двоичный	binary	二进制	ثنائي
текст	text	文本	نص
рисунок	image	图画	صورة
растр	bitmap	点阵图	صورة
звук	sound	声音	الصوت
музыка	music	音乐	موسيقى
видео	video	视频	فيديو



Запись информации в компьютере

- Современный компьютер – это электронная вычислительная машина. Информация в нем записывается и обрабатывается *с помощью электрических сигналов*.
- Компьютер использует *импульсные* сигналы. Такой сигнал может быть, и его может не быть (Рис. 2.1).

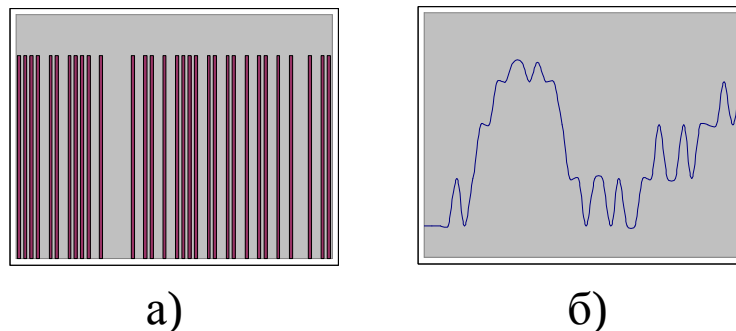


Рис. 2.1 Разные виды сигналов: а) Импульсный сигнал компьютера; б) Непрерывный сигнал телефона



Кодирование информации

- Представление информации с помощью символов из данного набора называется *кодированием*.
- Набор символов, который используется для кодирования, называется *алфавитом*.
- Компьютер использует алфавит

{0, 1},

который состоит из двух цифр 0 и 1 (сигнала НЕТ и сигнал ЕСТЬ).

- В математике с помощью цифр 0 и 1 записываются двоичные числа.
- Поэтому *информация в компьютере кодируется двоичными числами*.



Количество информации

- *Количество информации* измеряется количеством символов, которые используются для ее кодирования.
- Единицами количества информации являются:
 - *Бит* – содержит один символ алфавита {0; 1}. Это минимальная единица количества информации.
 - *Байт* – единица количества информации в системе СИ:

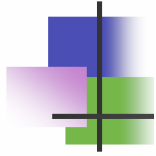
1 байт = 8 бит

- *Килобайт* = 2^{10} байт = 1024 байт (*1Кбайт* = 1K);
- *Мегабайт* = 2^{10} Кбайт = 1024 Кбайт (*1Мбайт* = 1M);
- *Гигабайт* = 2^{10} Мбайт = 1024 Мбайт (*1Гбайт* = 1G);
- *Терабайт* = 2^{10} Гбайт = 1024 Гбайт (*1Тбайт* = 1T);
- *Экобайт* = 2^{10} Тбайт = 1024 Тбайт (*1Ебайт* = 1E).



Проблемы кодирования

- Чтобы закодировать информацию для компьютера, надо записать ее с помощью *двоичных чисел*. Для этого:
 - *Числа* записывается в двоичной системе счисления.
 - Буквы *текста* и другие символы нумеруются целыми числами, а затем используются двоичные коды номеров символов. *Но как можно кодировать китайский текст?*
 - Цифровое *изображение* может содержать несколько миллионов частей – растров. Каждый растр кодируется несколькими целыми числами. *Но что делать, если изображений очень много?*
 - Для показа 1 сек. *видео* надо более 20 изображений. *Как можно закодировать двухчасовой фильм?*



Кодирование чисел

- Десятичные целые числа записываются с помощью цифр: 0, 1, 2, ..., 9. Мы знаем, что:

$$7325 = 7 \cdot 10^3 + 3 \cdot 10^2 + 2 \cdot 10^1 + 5$$

- По такому же правилу с помощью цифр 0 и 1 строятся и двоичные числа:

$$1001101_2 = 1 \cdot 2^6 + 0 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 77^{10}$$

1001101 – это двоичный код числа 77.

- Чтобы записать десятичное число в двоичном коде, надо последовательно выполнить деление на 2 с остатком данного числа и частных до тех пор, пока не получим нулевое частное (Таблица 2.1). Двоичный код числа образуется остатками, записанными в обратном порядке.

Таблица 2.1

Преобразование десятичного числа в двоичное

Число	77	38	19	9	4	2	1
Делитель	2	2	2	2	2	2	2
Частное	38	19	9	4	2	1	0
Остаток	1	0	1	1	0	0	1

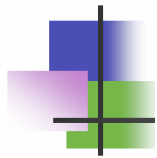
- Первые натуральные числа кодируются так:

Таблица 2.2

Двоичные коды десятичных чисел

Число	Код	Число	Код	Число	Код	Число	Код
0	0	4	100	8	1000	12	1100
1	1	5	101	9	1001	13	1101
2	10	6	110	10	1010	14	1110
3	11	7	111	11	1011	15	1111

- Десятичные дроби записываются в стандартном виде, (например, $0,00125 = 0,125 \cdot 10^{-2}$), а затем отдельно кодируются дробная часть и показатель степени 10.



Кодирование текста

- Символы алфавита нумеруются, а номера символов кодируются двоичными числами.
- Есть разные стандарты кодирования текстов:
 - **ASCII** (*American Standard Code for Information Interchanging*) для MS-DOS и Интернета.
 - **ANSI** (*American National Standards Institute*) для Windows и другие.
- В стандартах ASCII и ANSI размер кода символа один байт, и поэтому можно сразу кодировать 256 разных символов.
- В этих стандартах цифры, знаки и английские буквы имеют одинаковые коды.
- Для китайского, японского и других текстов используют многобайтовые коды. Такие коды имеют EUC (*Extended Unix Code*) и более популярный стандарт (*Unicode Transformation Format*). В стандарте UTF размер кода зависит от количества разных символов в алфавите. Этот стандарт совместим со стандартами ASCII и ANSI.



Кодовая страница

- В стандартах ASCII и ANSI первые 127 кодов с номерами **от 0 до 126** кодируют одинаковые *символы*:
 - Знаки;
 - Цифры;
 - Английские большие и малые буквы;
 - Символы управления выводом.
- Остальные 127 кодов с номерами **от 127 до 255** используются для кодирования *национальных алфавитов*.
- Эти 127 символов называются *Кодовой страницей*.

- Кодовая страница имеет имя и номер.
- Кодовая страница русского алфавита называется *Кириллицей*.
- Кириллица имеет номера: **866** в ASCII и **1251** в ANSI.



Кодирование рисунков

- Есть два основных способа кодирования изображений:
 - о **Растровая графика**. Она использует *растровую сетку*: на рисунок накладывается сетка, которая разбивает его на части – *растры*. Каждый растр имеет свой цвет и уровень яркости, которые кодируются числами.
 - о **Векторная графика** представляет рисунок через отрезки прямых, дуги окружностей и другие элементы. Эти элементы кодируются числами.
- Растровая графика имеет большой объем – чем мельче сетка, тем больше надо запомнить данных для кодирования рисунка.
- Векторная графика применяется для изображения простых фигур.



Контрольные вопросы

1. Какие единицы используются для измерения количества информации?
2. Что такое бит?
3. Что такое байт?
4. Что такое кодирование?
5. Почему двоичные числа используют для кодирования информации на компьютере?
6. Как кодируются числа?

7. Какой двоичный код имеет число 12?
8. Как кодируются цифры?
9. Как кодируется текст?
10. Что такое ASCII?
11. Что такое ANSI?
12. Что такое Кодовая страница?
13. Как называется кодовая страница русского алфавита?
14. Какой номер имеет Кириллица в ASCII?
15. Какой номер имеет Кириллица в ANSI?
16. Сколько бит информации используется для кодирования одной буквы в стандартах ASCII и ANSI?
17. Сколько байт информации используется для кодирования одной цифры в стандартах ASCII и ANSI?
18. Зачем используется стандарт UTF?
19. Какие виды машинной графики Вы знаете?
20. Как кодирует рисунок векторная графика?
21. Что такое растровая сетка?
22. Что такое растровая графика?
23. Как кодируется музыка?
24. Как кодируется видео?
25. Что такое мультимедиа?



Лабораторная работа

1. Включите компьютер.
2. Запустите программу для рисования Microsoft Paint с помощью кнопки **Пуск** на Панели задач. Программа находится в разделе **Стандартные** команды **Программы**.
3. В окне программы Microsoft Paint нарисуйте Государственный флаг своей страны. Для рисования и раскраски флага используйте Панели инструментов программы.
4. Рядом справа нарисуйте Государственный флаг Украины.

5. Под флагами напишите названия стран на русском языке. Воспользуйтесь для этого кнопкой **Текст** Панели инструментов программы.
6. С помощью инструмента "*Карандаш*" Панели инструментов программы напишите такой же текст на родном языке.
7. Покажите Вашу работу преподавателю.
8. Сохраните созданный Вами документ в личной паке.
9. Выключите компьютер.



Практические задания

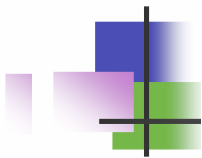
1. Записать в двоичной системе счисления заданное десятичное число.
2. Записать в десятичной системе счисления заданное двоичное число.
3. Запустить программу с помощью Панели задач.
4. Свернуть окно программы на Панель задач и восстановить его.
5. Нарисовать цветок с помощью программы Microsoft Paint.
6. Закрыть окно программы.

3. Структура компьютера



Ключевые слова

команда	command	指令	القيادة
программа	program	程序	البرنامج
регистр	register	寄存器	سجل
разрядность	capacity	容量	القدرة
цикл	cycle	周期	دورة
частота	frequency	频率	تردد
процессор	processor	处理器	معالج
память	memory	内存	الذاكرة
внутренний	internal	内部	داخلي
внешний	external	外部	خارجي
произвольный	random	随机	عشوائي
последовательный	sequential	连续	متسلسل
доступ	access	进入	الوصول
диск	disk	磁盘	القرص
гибкий	floppy	软盘	مرن
жесткий	hard	硬	بجد
дисковод	drive	驱动	حملة
модем	modem	调制解调器	مودم
принтер	printer	打印机	الطابعه
сканер	scanner	扫描仪	الناسخ الضوئي
плоттер	plotter	绘图机	المتأمر
клавиатура	keyboard	键盘	لوحة المفاتيح
клавиша	key	键	مفتاح
мышь	mouse	鼠标	الفأر
монитор	display	显示器	عرض



Конфигурация компьютера

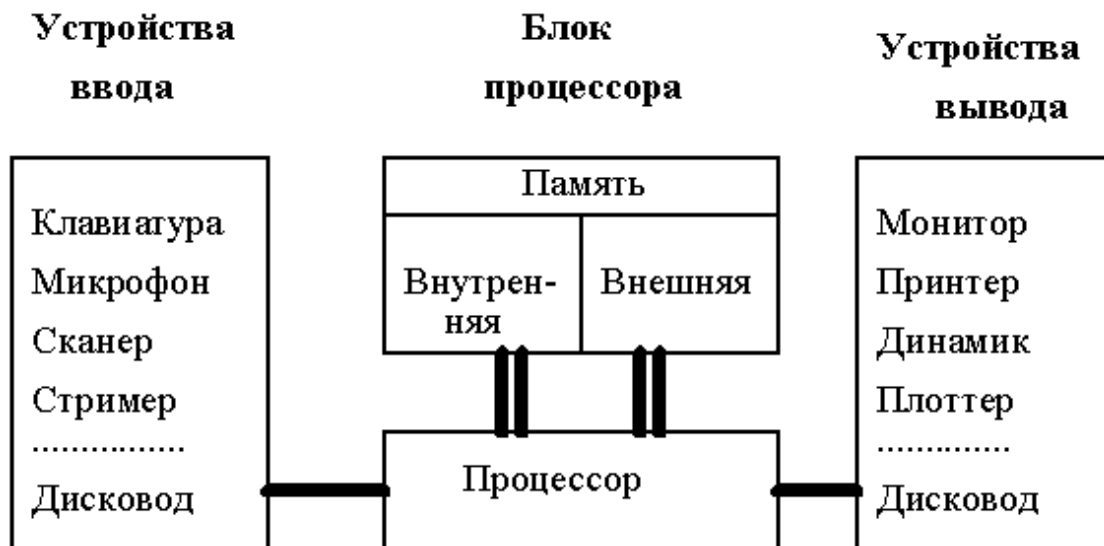


Рис. 3.1 Функциональная схема компьютера

- Толстые линии на схеме компьютера показывают *движение информации*.
- Разные компьютеры отличаются своими устройствами и их характеристиками.
- Отдельные устройства могут быть одновременно и устройствами ввода, и устройствами вывода:
 - Дисковод CD – RW/DVD–RW;
 - Флэш-память;
 - Модеми другие.
- Характеристики устройств определяют возможности компьютера.
- Совокупность устройств, которые имеет компьютер, называется *конфигурацией компьютера*.
- При покупке компьютера пользователь сам выбирает конфигурацию своего компьютера.



Основные устройства



Процессор •
Central Processing Unit - CPU

Внутренняя память •
Random Access Memory - RAM

Внешняя память •
Hard Disk - HD

Устройства ввода/вывода •
Input/Output – I/O

Рис. 3.2 Такие устройства имеет каждый компьютер



Процессор

- **Процессор** (CPU – *центральный процессор*) – это основное устройство компьютера.
- Процессор *обрабатывает* информацию и управляет работой всех устройств компьютера.
- Процессор *выполняет* операции (*команды*) над отдельными байтами (иногда – битами) данных.
- Каждая операция выполняется за несколько шагов, которые называются *тактами*.
- Основной характеристикой процессора является *тактовая частота* – количество тактов, которые компьютер выполняет за 1 сек.
- Тактовая частота измеряется в герцах:

1 герц = 1 такт/с



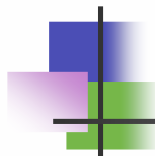
Регистры и разрядность

- Процессор обрабатывает данные в специальных устройствах – *регистрах*.
- *Разрядность компьютера* – это количество двоичных символов, которые одновременно записываются в регистре.
- В разное время использовались компьютеры с разной разрядностью:
 - В 1990 году использовались 16-разрядные компьютеры с операционной системой MS-DOS;
 - В 2005 году использовались 32-разрядные компьютеры с операционной системой Windows;
 - Современные компьютеры 64-разрядные.



Память

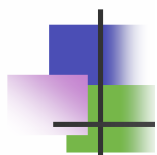
- В памяти компьютера записывается программа и *данные* – информация, которую обрабатывает компьютер.
- Программа и данные закодированы двоичными числами.
- Компьютер имеет два вида памяти:
 - *Внутренняя память (оперативная память)* (RAM – *Random Access Memory*) с быстрым доступом и небольшим объемом – несколько Гбайт;
 - *Внешняя память* – жесткий диск (HD – *Hard Disk*), флэш-память и другие устройства с медленным доступом и большим объемом – сотни Гбайт.



Принципы работы памяти

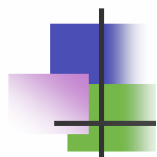


- В RAM двоичный символ представляется *током*.
- В устройствах внешней памяти используют *разные физические явления*:
 - В дискетах и жестких дисках (HD) данные записываются с помощью *магнитных полей*.
 - В оптических дисках (CD и DVD) данные записывают путем *разрушения поверхности* диска лазерным лучем.
 - Во флэш-памяти данные записывают с помощью *электрических зарядов*.
- В ближайшие годы вместо флэш-памяти будут использовать *молекулярную память*, которая учитывает *расположение атомов* в материале.



Работа компьютера

- Работа, которую выполняет компьютер, называется *задачей*.
- Пользователь дает задания компьютеру с помощью команд.
- *Команды* – это обычные слова, которыми пользователь сообщает компьютеру, что надо делать.
- Компьютер может *воспринимать* только несколько десятков разных команд. Они очень простые.
- Разные типы компьютеров имеют разные *системы команд*.
- Система команд компьютера и правила их выполнения образуют *язык машины*.



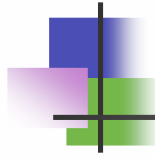
Программы

- Последовательность команд, необходимых для решения задачи, называется *программой*.
- Программы для решения практических задач бывают большими и содержат миллионы команд.
- Программы записываются в памяти компьютера. Компьютер *различает программы по именам*.
- Чтобы *запустить* программу – начать выполнять ее, – надо указать компьютеру имя программы.
- Имя программы может быть командой для ее запуска.



Работа программы

- Пользователь ***не обязан знать*** коды программы.
- Чтобы использовать программу, пользователь ***должен знать***:
 - *Имя программы*;
 - *Задачу*, которую решает программа;
 - *Входные данные* – какую дополнительную информацию надо задать программе;
 - *Формат входных данных* – в какой форме надо задавать данные программе;
 - *Формат выходных данных* – в какой форме программа представляет результат.



Контрольные вопросы

1. Что такое команды?
2. Что такое система команд компьютера?
3. Что такое входные данные?
4. Что такое выходные данные?
5. Как компьютер различает разные задачи?
6. Что такое конфигурация компьютера?
7. Зачем используют устройства ввода?
8. Зачем используют устройства вывода?
9. Что такое разрядность компьютера?
10. Что такое память компьютера?
11. Какие виды памяти Вы знаете?
12. Какие явления используются в оперативной памяти RAM?
13. Какие явления используются для представления данных на магнитном диске HD?
14. Какие явления используются для представления данных на оптическом диске CD?
15. Какие явления используются во флэш-памяти?
16. Что такое процессор?
17. Что такое такт?
18. Что такое тактовая частота компьютера?
19. Что такое дисковод?
20. Зачем используется модем?
21. Какую работу выполняет модем?
22. Зачем используется сканер?
23. Зачем используется мышь?



Лабораторная работа

1. Запустите программу Trk.exe в папке KeyTrain.

2. Выберите в меню программы:

✓ Уровень мастерства – **Новичок**;

✓ Шрифт – **Русский**

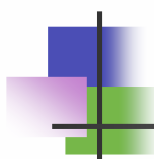
и запустите программу с помощью меню команду **Начать обучение**.

После первого этапа работы программы нажмите клавишу **Esc** и с помощью команды **Статистика** посмотрите свой результат. Если Вы им удовлетворены, запишите его и продолжите работу программы с помощью команды **Продолжить**. Если нет – завершите работу программы и начните сначала. Для выхода в меню используйте клавишу **Esc**.

Выполните задание так, чтобы *средняя скорость* была не меньше 90, а *количество ошибок* – не больше 30%. Запишите и покажите преподавателю свой наилучший результат.

3. Выполните ту ж работу для английского шрифта.

4. Выполните ту ж работу для уровня мастерства **Кандидат**.



Практические задания

1. Определить характеристики компьютера: тип процессора, тактовую частоту, объем оперативной памяти.

2. Определить количество и объем дисков на компьютере.

3. Определить конфигурацию компьютера.

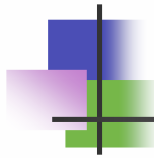
4. Определить, может ли компьютер записывать данные во флэш – памяти.

4. Программы



Ключевые слова

ВВОД	input	输入	مدخلات
ВЫВОД	output	输出	الناتج
прикладной	applied	应用	طبق
системный	system	系统	نظام
резидентный	resident	储存	المقيم
ПОЛЬЗОВАТЕЛЬ	user	用户	مستعمل
интерфейс	interface	接口	الوصله
обмен	exchange	交流	تبادل
меню	menu	菜单	القائمة
всплывающий	popup	弹出	قافزه
выпадающий	dropping	下降	سقوط
пиктограмма	pictogram	图表文字	الرسم التخطيطي
функциональный	functional	功能	وظيفي
отчуждение	alienate	远离	تنفير
непредвиденный	unforeseen	不可预见	غير متوقع
комбинация	combination	组合	مجموعة
горячий	hot	热点	حار
тянуть	drag	拖动	أسحب
отпустить	drop	中止	قطرة
переместить	move	移动	تحرك
переключать	switch	开关	مفتاح
вспомогательный	auxiliary	辅助	مساعد
альтернатива	alternative	替代	البديل
закрепить	fix	确定	فيكس
содержимое	content	内容	المحتوى



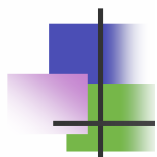
Системные и прикладные программы

- Работа компьютера состоит в том, что он выполняет одну или несколько программ по заданию пользователя.
- Есть два типа программ:
 - *Системные программы*, которые решают общие проблемы управления компьютером;
 - *Прикладные программы*, которые решают конкретные проблемы пользователя – подготовка документов, вычисления, игры и другие.
- Создание программ – важная область деятельности человека.



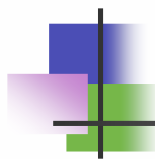
Интерфейс

- *Интерфейс* – это обмен данными между различными объектами.
- Интерфейсом называют также программы и устройства для обмена данными в компьютере:
 - *Кабели*;
 - *Разъемы*. Например, разъем *USB* для подключения внешних устройств;
 - *Схемы*. Например, *Bluetooth* для подключения мобильного телефона.
- *Пользовательский интерфейс* – это обмен данными между компьютером и человеком – пользователем.



Пользовательский интерфейс

- Современный компьютер использует разные виды пользовательского интерфейса:
 - *Графический интерфейс* осуществляется с помощью окон, мыши и ярлыков – рисунков на экране, которые обозначают разные объекты;
 - *Курсорный интерфейс* – задание компьютеру вводят с помощью курсора, в том числе и с помощью системы меню;
 - *Командный интерфейс* – задание компьютеру вводят с помощью специальных слов – команд;
 - *Клавишный интерфейс* – задание компьютеру вводят нажатием специальных клавиш на клавиатуре.



Графический интерфейс

- Графический интерфейс является основным для современных компьютеров.
- Часто используется *оконный интерфейс* – каждая программа выполняется в своем окне (Рис.4.1).
- Окно *открывается* при запуске программы и *закрывается*, когда программа *заканчивает* работу.

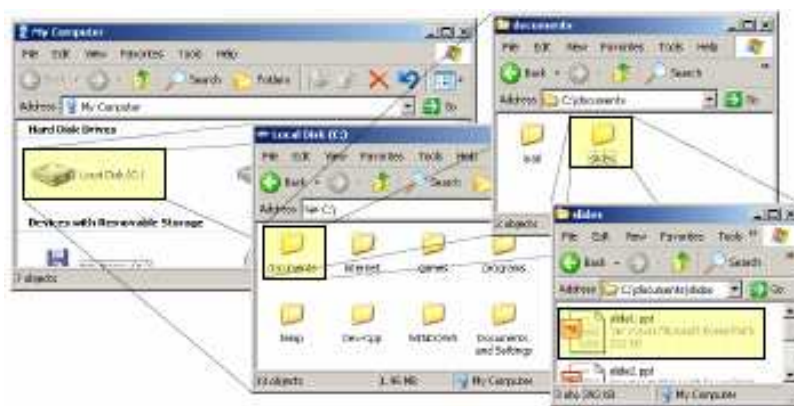


Рис. 4.1 Окна в системе Windows XP



Курсорный интерфейс

- Графический интерфейс выполняется с помощью курсора, который выделяет нужный объект на экране.
- *Курсор* – это изображение, которое можно перемещать на экране с помощью мыши или специальных клавиш – клавиш управления курсором.
- Графический интерфейс является основным в системе Windows:
 - Курсор выделяет нужный объект на Рабочем столе;
 - Двойной щелчок клавишей мыши на объекте запускает программу, которая связана с этим объектом.

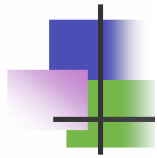


Меню

- Курсорный интерфейс используется в системе меню.
- *Меню* – это список команд, которые компьютер может выполнить в данный момент.
- *Панка меню* – это специальный курсор в меню, который выделяет активную команду.
- При выборе команды с многоточием (...) на экране появляется *выпадающее* (или *всплывающее*) меню со списком предлагаемых действий.



Рис. 4.2 Система меню кнопки **Пуск**



Командный интерфейс

- Команды задаются в специальном окне – окне *Командной строки*.
- В системе Windows XP окно Командной строки вызывается командой `cmd` в окне команды **Выполнить** кнопки **Пуск**.
- В системе Windows Vista в кнопке **Пуск** есть команда **Командное окно**
- Выход из окна Командной строки выполняется командой `exit`

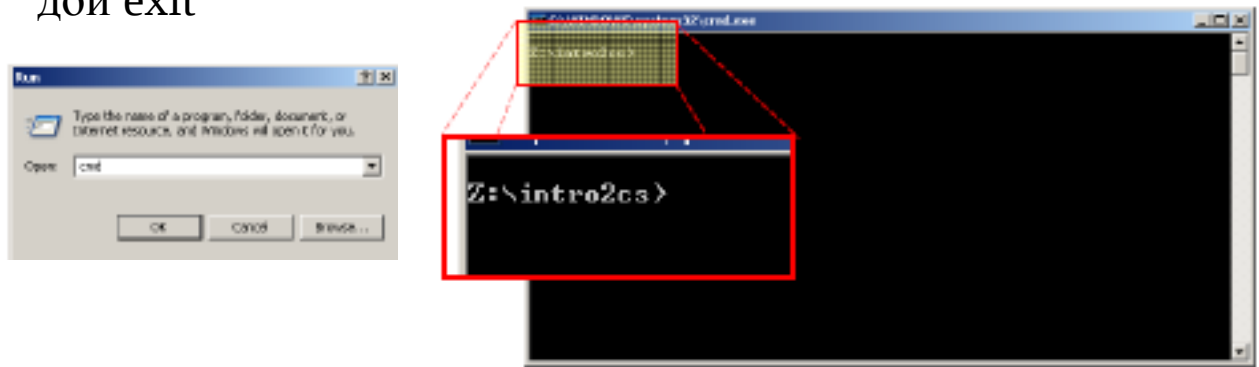
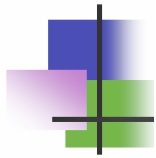


Рис. 4.3 Запуск окна Командной строки в Windows XP



Клавишный интерфейс

- Клавишный интерфейс основан на использовании специальных клавиш клавиатуры:
 - Функциональных клавиш **F1**, **F2**, ..., **F12**, действия которых устанавливаются в программах;
 - Клавиши **Esc**, которая прекращает выполнение программы;
 - Клавиши **Pause**, которая приостанавливает выполнение программы;
 - Клавиши **PrintScreen**, которая запоминает в буфер изображение на экране и других.



Комбинации клавиш

- Клавишный интерфейс использует также комбинации клавиш **Ctrl** или **Alt** с другими клавишами.
- В системе Windows используются такие комбинации:
 - **Alt+F4** – завершает процесс и закрывает его окно;
 - **Ctrl+Z** – отменяет последнее действие;
 - **Ctrl+C** – копирует выделенный объект в буфер;
 - **Ctrl+X** – вырезает выделенный объект в буфер;
 - **Ctrl+V** – вставляет объект из буфера;
 - **Alt+Ctrl+Del** – заканчивает работу компьютера.



Копирование и перемещение

- В графическом интерфейсе есть два способа перенесения информации:
 - *копирование (Copy)* – старый файл сохраняется, а на новом месте появляется его копия с теми же атрибутами;
 - *перемещение (Move)* – на новом месте появляется копия файла, а старый файл уничтожается.

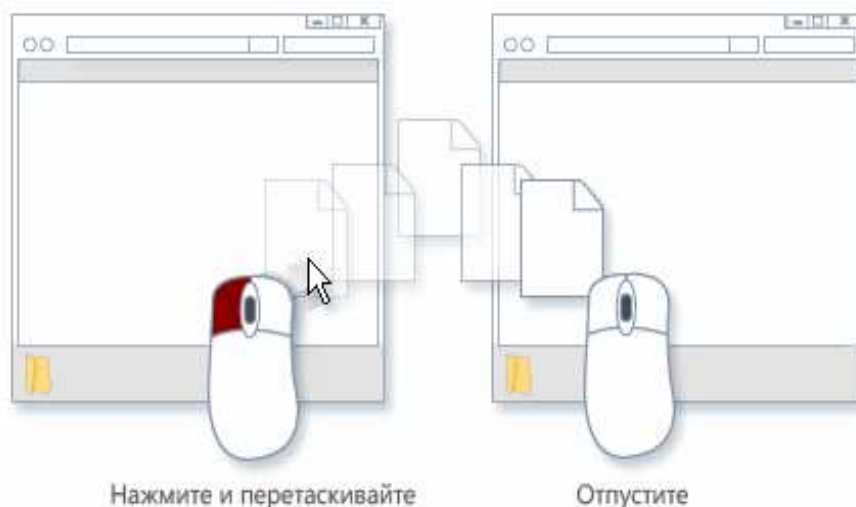


Рис 4.4 Метод drag-and-drop перенесения информации



Контрольные вопросы

1. Что такое прикладная программа?
2. Что такое системная программа?
3. Что такое резидентная программа?
4. Что такое интерфейс?
5. Что такое интерфейс пользователя?
6. Какие виды интерфейса пользователя Вы знаете?
7. Что такое графический интерфейс?
8. Что такое ярлык?
9. Что такое курсор?
10. Как работает курсорный интерфейс?
11. Что такое меню?
12. Как выполнить команду меню?
13. Что такое клавишный интерфейс?
14. Что такое функциональные клавиши?
15. Какие клавиши используются для построения комбинаций клавиш, которые управляют компьютером?
16. Какая комбинация клавиш прекращает выполнение программы?
17. Какая клавиша запоминает в буфере изображение экрана?
18. Что такое командный интерфейс?
19. Как вводятся команды при командной интерфейсе при работе в системе Windows?
20. Чем отличается копирование от перемещения данных?
21. Что такое буфер?
22. Что такое интерфейс drag-and-drop?
23. Как предотвратить перемещение объекта при копировании его из папки в папку методом drag-and-drop?



Лабораторная работа

1. Запустите программу Microsoft Word и введите в новый документ русский текст: "Я уже умею вводить текст в компьютер. Сейчас я буду учиться, как записать этот текст в файл и на дискету".
2. Если в тексте есть подчеркнутые слова, то это значит, что в них есть ошибки. Исправьте ошибки. Для этого к каждому подчеркнутому слову подведите курсор и нажмите правую кнопку мыши. Компьютер сообщит Вам, какую ошибку он обнаружил и подскажет, как исправить ее.
3. С помощью кнопок управления окном выйдите из программы.
4. Откройте программу "Мой компьютер" с помощью ярлыка на Рабочем столе. С помощью этой программы войдите в Вашу личную папку и выделите щелчком мыши созданный файл.
5. Скопируйте созданный Вами файл на дискету. Для этого надо:
 - а) вставить дискету в дисковод;
 - б) выделить файл;
 - в) щелчком правой кнопки мыши вызвать уточняющее меню папки;
 - г) в уточняющем меню выбрать и выполнить команду **Отправить** по адресу **Диск А**;
 - д) вынуть дискету из дисковода.
6. С помощью команд меню правой кнопки мыши скопируйте созданный файл в папку **Work**, которая должна находиться в том же каталоге, что и Ваша личная папка.
7. Удалите созданный файл из папки **Work** в **Корзину** методом drag-and-drop.
8. Очистите Корзину.
9. Покажите Вашу работу преподавателю.



Практические задания

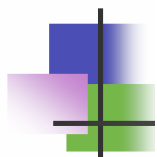
1. Выбрать в меню команду и выполнить ее.
2. Выделить объект на Рабочем столе.
3. Выполнить программу, связанную с ярлыком на Рабочем столе.
4. Скопировать объект в буфер с помощью мыши.
5. Вырезать объект в буфер с помощью мыши.
6. Вставить объект из буфера с помощью мыши.
7. Скопировать объект в буфер с помощью системы меню.
8. Вырезать объект в буфер с помощью системы меню.
9. Вставить объект из буфера с помощью системы меню.
10. Скопировать объект с помощью комбинации клавиш.
11. Вырезать объект с помощью комбинации клавиш.
12. Вставить объект из буфера с помощью комбинации клавиш.
13. Включить и выключить вспомогательную числовую клавиатуру.
14. Закрепить ввод больших букв.
15. Заккрыть окно с помощью комбинации клавиш.
16. Свернуть окно на Панель задач с помощью кнопок управления окном.
17. Восстановить окно, свернутое на Панель задач, с помощью команд меню.
18. Заккрыть окно с помощью кнопок управления окном.
19. Скопировать файл методом drag-and-drop.
20. Переместить файл методом drag-and-drop.
21. Выключить компьютер с помощью комбинации клавиш.

5. Операционные системы

Ключевые слова



операционная	operational	运行	التنفيذية
система	system	系统	نظام
управлять	control	控制	السيطرة
проверять	check	检查	فحص
согласовывать	agree	同意	اتفق
домашний	home	首页	الصفحة الرئيسية
офисный	office	办公室	المكتب
безопасность	security	安全	الامن
файл	file	档案	الملف
папка	folder	文件夹	ملف للاوراق
каталог	directory	目录	دليل
подкаталог	subdirectory	子目录	دليل ثانوي
расширение	extension	扩建	إمتداد
атрибут	attribute	特征	صفة
исполняемый	executable	可执行	قابل للتنفيذ
резервный	backup	备份	الاحتياطي
драйвер	driver	启动	سائق
ядро	kernel	核心	نواة
подчиненный	slave	属于	السلافي
родительский	parent	主体	الوالد
дерево	tree	树图	شجرة
структура	structure	结构	بنية
иерархия	hierarchy	等级	هرمي
текущий	current	当前	الحالية
маршрут	path	路径	الطريق
цепочка	chain	链	السلسلة
подстановка	substitution	替代	البديل
загрузка	boot	开机	الحذاء
установка	install	安装	تركيب



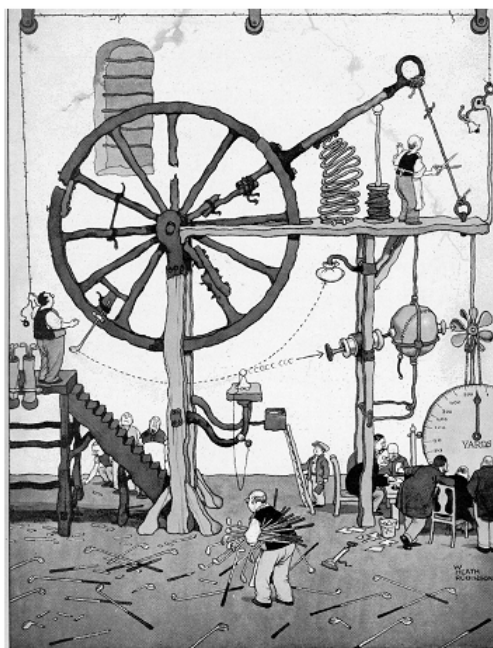
Что такое операционная система

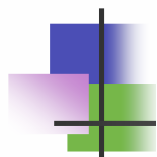
- *Операционная система (ОС)* – это программа, которая управляет работой компьютера и всех его устройств.
- При выполнении любой программы на компьютере операционная система:
 - Обеспечивает *связь между программой* и устройствами компьютера;
 - Обеспечивает *связь с другими компьютерами* в сети;
 - *Вводит и выводит* данные;
 - *Контролирует и согласовывает* работу всех устройств компьютера.



Структура операционной системы

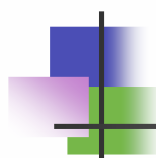
- Операционная система (ОС) – это набор программ. Программы ОС хранятся во внешней памяти. При необходимости они вызываются в RAM и выполняются.
- Некоторые программы ОС постоянно находятся в RAM и работают. Это – *резидентные программы*.
- Резидентные программы образуют *ядро ОС*. Ядро ОС запускается при включении компьютера.
- Для связи с устройствами компьютера ядро использует *драйверы устройств*. Это программы, которые устанавливаются при подключении устройства к компьютеру и включаются в библиотеки программ ОС.





Типы операционных систем

- Первой операционной системой для персональных компьютеров была **MS-DOS** (*Disk Operation System*).
- Разные типы компьютеров используют разные ОС.
- Компьютеры фирмы IBM и совместимые с ними чаще всего используют ОС **Windows**.
- Разные типы компьютеров могут использовать ОС **Unix** (**Linux**). Это универсальная ОС. Она написана на языке программирования C. Все компьютеры имеют компиляторы, которые переводят программы из C в машинные коды.



Домашние и офисные ОС

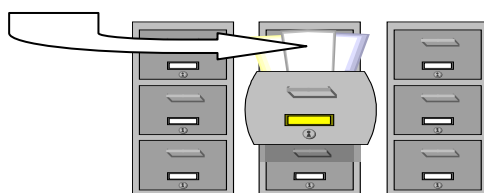
- Существуют домашние и офисные ОС для предприятий.
- *Домашние* ОС удобно использовать. Это – системы:
 - Windows 98;
 - Windows XP Home Edition;
 - Windows Vista Home Edition.
- *Офисные* ОС надежно защищают данные, но требуют больше ресурсов памяти и времени. Это – системы:
 - Windows NT 4.0;
 - Windows 2000;
 - Windows XP Professional Edition;
 - Vista Professional Edition;
 - Unix.
- Популярная Windows XP Home Edition требует:

	Минимально	Рекомендуется
Тактовая частота	233 MHz	≥ 600 MHz
Память RAM	64 MB	≥ 128 MB
Объем диска HD	1,5 GB	≥ 4 MB



Файлы

- Любая информация записывается на компьютере в виде файлов.
- **Файл – это именованный набор данных одного типа.**
- *Типы данных* могут быть разными. Различают:
 - *Текстовые файлы*, которые содержат только символы клавиатуры – буквы, цифры, знаки препинания;
 - *Бинарные файлы* кроме текстовых символов могут содержать символы, которых нет на клавиатуре, но которые используются для управления компьютером.
- Операционная система *различает файлы по именам*.
- Для записи имени файла *нельзя* использовать символы (*? " / | : < >) и некоторые другие.



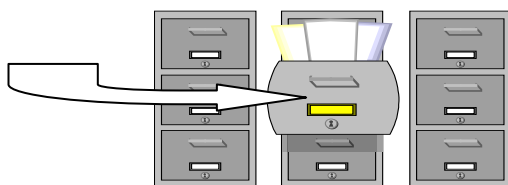
- *Полное имя* файла состоит из двух частей, которые разделены точкой. Например, Lesson1.doc.
- Первая часть имени (*имя*) идентифицирует файл и дается пользователем.
- Вторая часть имени – *расширение*, – указывает тип и назначение файла. Его нельзя выбирать произвольно.
- ОС Windows имеет программы для работы с файлами, которые имеют расширения:

exe, com	Исполняемые программы
txt, rtf	Текстовые файлы
doc, xls, mdb и др	Документы Microsoft Office
bmp, gif, jpg, jpeg и др.	Рисунки
mov, mp4, avi, wmv	Мультимедиа (музыка, видео)
zip, rar, exe	Архивные файлы



Каталоги (папки)

- Операционная система может объединять любое количество файлов и работать с ними как с одним объектом. Такое объединение файлов называется *каталогом*.
- В ОС Windows каталог называется *папкой*.
- ***Каталог – это именованный набор файлов и других каталогов.***
- Имя каталога не имеет расширения.



- Каталог может содержать внутри себя другой каталог.
- Если А и В – каталоги, и А содержит В, то А называется *родительским каталогом*, а В – *подчиненным каталогом*.
- Каталог, который не имеет родительского каталога, называется *корневым каталогом*.
- Каталоги образуют иерархическую систему – *дерево файлов*.
- В каждый момент времени пользователь работает с файлами какого-нибудь одного каталога. Такой каталог называется *текущим каталогом*.

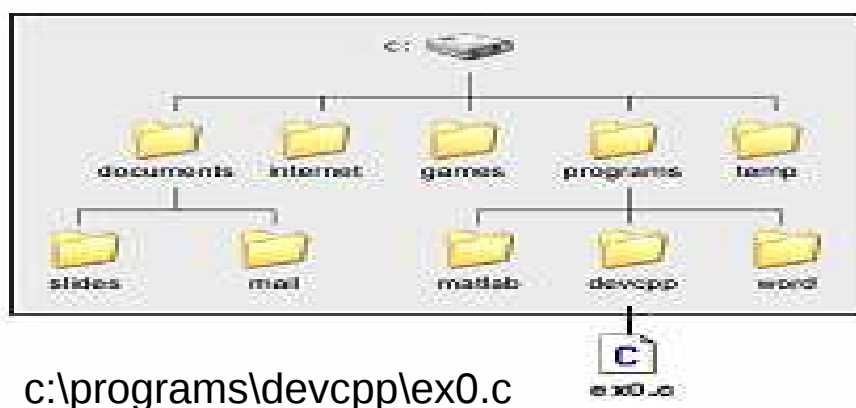
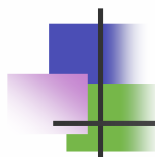
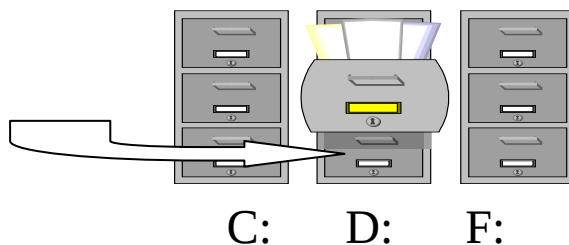


Рис. 5.1 Дерево каталогов на диске С и маршрут файла.



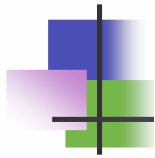
Логические диски

- Компьютер может иметь несколько дисков (жестких дисков, дисков CD, CD, DVD, флэш-дисков и др.) на разных дисковых устройствах. Это – *физические диски*.
- Операционная система работает с логическими дисками.
- **Логический диск – это некоторая часть внешней памяти, где размещаются каталоги и файлы.**
- Внешнюю память компьютера разбивают на логические диски для того, чтобы информация, которая не используется программой, не была случайно испорчена.
- ОС различает логические диски по именам.
- В ОС Windows имя диска обозначается одной английской буквой: A, B, C, D ...
- Диск может иметь также другое (альтернативное) имя, которое записано на нем.
- Сама операционная система записана на одном из дисков. Такой диск называется *системным*.



BIOS

- В компьютерах, совместимых с IBM PC, связь между операционной системой и устройствами выполняет *базовая система ввода-вывода BIOS*.
- Программы BIOS записаны на микросхемах системной платы и устройств. Они дублируются также на системном диске. Программы BIOS запускаются при включении компьютера, проверяют устройства и передают управление программам ядра ОС.



Маршрут

- При описании файла для ОС надо указать *маршрут файла* – цепочку подчиненных каталогов, последним из которых является каталог, который содержит файл.
- При описании файла маршрут записывается как префикс имени файла. При этом двоеточие (..) обозначает родительский каталог для текущего каталога, где находится программа, которая использует такое описание.

Пример 1. Запись:

`..\..\documents\slides\slide1.ppt`

указывает маршрут из каталога `devcpp` к файлу `slide1.ppt` – подняться на два уровня вверх в дереве каталогов, а затем перейти в каталог `documents`, потом в каталог `slides`.

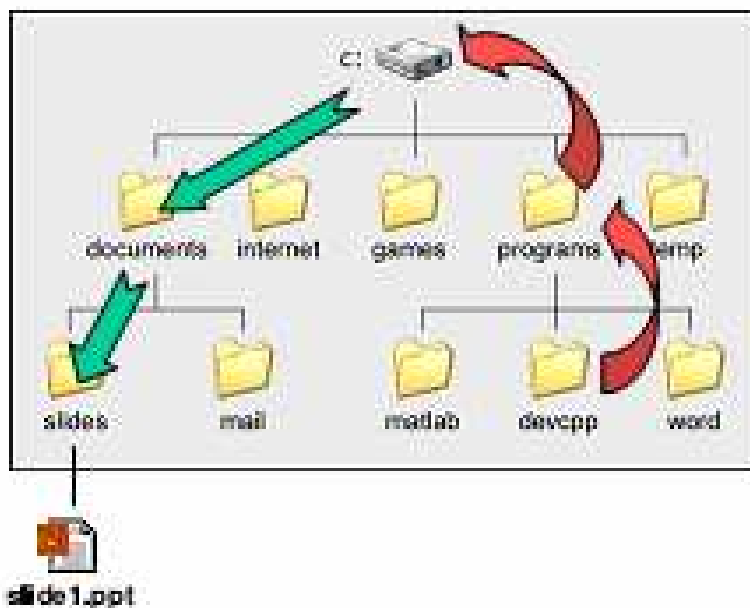
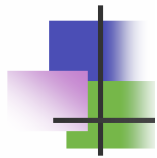


Рис. 5.2 Маршрут файла `slide1.ppt` из каталога `devcpp`

Этот маршрут можно описать иначе:

`C:\documents\slides\slide1.ppt`

Первый способ описания удобнее тем, что он не изменится, если все содержимое диска `C` поместить в новую папку.



Контрольные вопросы

1. Что такое операционная система?
2. Что такое MS-DOS?
3. Что такое Windows?
4. Что такое Linux?
5. Какие операционные системы семейства Windows созданы для использования на предприятиях?
6. Когда запускается операционная система?
7. Что такое файл?
8. Что такое расширение файла?
9. Что обозначает расширение файла?
10. Что такое атрибуты файла?
11. Какие данные содержит бинарный файл?
12. Какие расширения имеют файлы с исполняемыми программами?
13. Что такое каталог?
14. Что такое папка?
15. Что такое подкаталог?
16. Что такое родительский каталог?
17. Что такое подчиненный каталог?
18. Какой каталог называется корневым?
19. Что такое текущий каталог?
20. Что такое маршрут?
21. Что такое логический диск?
22. Что такое системный диск?
23. Что такое базовая система ввода-вывода BIOS?
24. Что такое ядро операционной системы?



Лабораторная работа

1. Прочитайте в Приложении 1 десять заповедей современного специалиста, которые разработаны в Институте компьютерной этики в США. Подумайте, согласны ли Вы с ними.
2. С помощью Поисковой системы ОС Windows найдите на компьютер файл **ПисьмоМохамеда.doc** и откройте его. Текст письма находится также в Приложении 2.
3. Сверните файл на Панель задач.
4. Прочитайте письмо и напишите в тетради ответ.
5. С помощью программы "Мой компьютер" войдите в личную папку и создайте в ней новый файл типа "Документ Microsoft Word".
6. Напишите в этом файле текст Вашего письма Мохамеду и сохраните его в личной папке.
7. Сверните файл с Вашим письмом на Панель задач.
8. Покажите Вашу работу преподавателю.
9. Закройте все открытые файлы.



Практические задания

1. Определить маршрут Вашей личной папки.
2. Определить имя и размер родительского каталога для Вашей личной папки.
3. Определить количество файлов родительского каталога для Вашей личной папки.
4. Определить размер папки и количество файлов в ней.
5. Упорядочить файлы в папке.
6. Определить атрибуты файла.
7. Найти в компьютере нужный файл.



6. Работа с дисками

Ключевые слова

поверхность	surface	表面	السطح
дорожка	track	路径	يتعقب
сектор	sector	区域	القطاع
сегмент	segment	部分	الجزء
кластер	cluster	集群	المجموعة
расположение	layout	布局	التخطيط
загружать	load	加载	تحميل
копировать	copy	复制	نسخة
переместить	move	移动	حركة
переименовать	rename	重新命名	بدل اسم
удалить	delete	删除	حذف
восстановить	restore	恢复	استعادة
целостность	integrity	完整性	النزاهة
конфигурация	configuration	配置	التشكيل
устанавливать	set	设置	يعين
видимый	seen	显著的	شهد
продолженный	expended	继续	منفق
расширенная	extended	延长	مدد
базовая	conventional	常规	تقليدية
форматировать	format	格式	تهيئة
предостеречь	preserve	保存	المحافظة
архив	archive	存档	ارشيف
упаковать	pack	装入	علبة
распаковать	unpack	解压	افتح
отправлять	send	发送	ايرسل
расписание	sheduller	附表	جدول



Запись данных на диске

- Компьютеры используют диски в качестве устройств внешней памяти.
- Информация на всех дисках записывается одинаково.
- Поверхность диска разделена магнитными записями на концентрические кольца - *дорожки* и радиально – на *сектора*.
- Так весь диск оказывается разбитым на отдельные участки – *сегменты*.
- Обычно размер сегмента равен 512 байт.
- Дорожки нумеруются, начиная с 0.



Таблицы расположения файлов

- Операционная система записывает файл на диск и читает его оттуда порциями – кластерами.
- Все кластеры диска имеют одинаковый размер. Обычно этот размер кратный 512К.
- Кластеры нумеруются подряд, начиная с 0.
- Кластеры одного файла могут быть записаны в разных местах диска.
- Расположение кластеров каждого файла на диске записывается в *таблице расположения файлов*.
- Вместе с таблицей на диске записывается ее копия для восстановления таблицы при сбое машины.
- Таблица расположения файлов и ее копия записываются на нулевой дорожке диска.

Размер кластера=Размер диска/Длина адреса



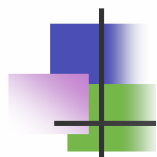
Таблицы FAT и NTFS

- Домашние и офисные ОС используют разные таблицы расположения файлов.
- Домашние ОС используют таблицу FAT (*File Allocation Table*):
 - Windows 95 – таблицу FAT16 из 2^{16} кластеров;
 - Windows 98, XP – таблицу FAT32 из 2^{32} кластеров.
- В таблице FAT для каждого кластера файла записывается адрес и номер следующего кластера.
- Офисные ОС (Windows NT, 2000, XP) используют таблицу NTFS (*New Technology File System*) из 2^{64} кластеров. В таблице NTFS для каждого файла записано право доступа к этому файлу для каждого пользователя.



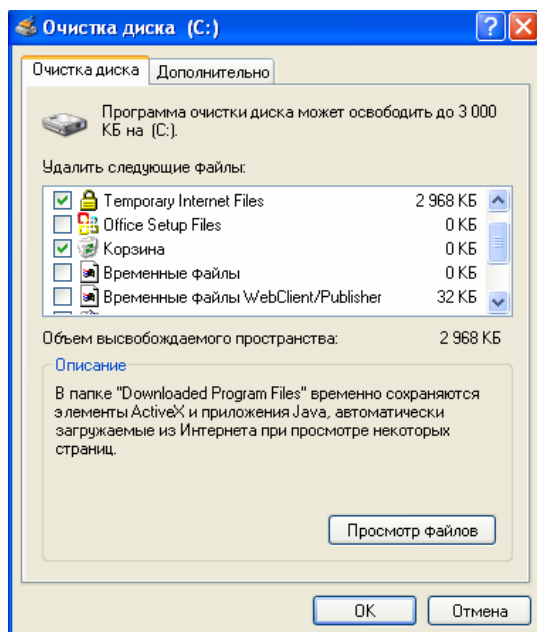
Корзина

- Файлы, которые пользователь удаляет с компьютера, временно сохраняется в специальной папке – **Корзине**. Это позволяет восстановить файл в исходной папке, если окажется, что его удалили ошибочно.
- Файл в **Корзине** всегда можно восстановить – вернуть в исходную папку.
- Чтобы окончательно удалить файлы с компьютера и освободить место в памяти, которое они занимают, необходимо *очистить* корзину – удалить файлы из корзины.
- Из корзины можно удалить отдельные файлы, а можно сразу очистить всю корзину.
- Операционную систему можно настроить так, что она не будет сохранять файлы в **Корзине**.



Очистка диска

- При работе компьютер создает много вспомогательных файлов, которые не нужны после завершения программы или сеанса работы.
- Такие временные файлы часто (но не всегда!) имеют расширение .tmp.
- Компьютер сам удаляет временные файлы, но не все и не всегда.
- Для удаления ненужных файлов используют программу «Очистка диска» из раздела **Служебные папки Стандартные кнопки Пуск**.



Дефрагментация диска

- Файлы на жестком диске разбиваются на части (фрагментируются) при сохранении или внесении в них изменений.
- Внесенные в файл изменения часто сохраняются в другом месте жесткого диска, удаленном от места исходного файла. Изменения, которые вносятся позже, сохраняются еще в других местах диска. Так со временем части файла разносятся по всему диску.
- Это замедляет работу компьютера, так как для открытия файла приходится считывать информацию из разных мест жесткого диска.

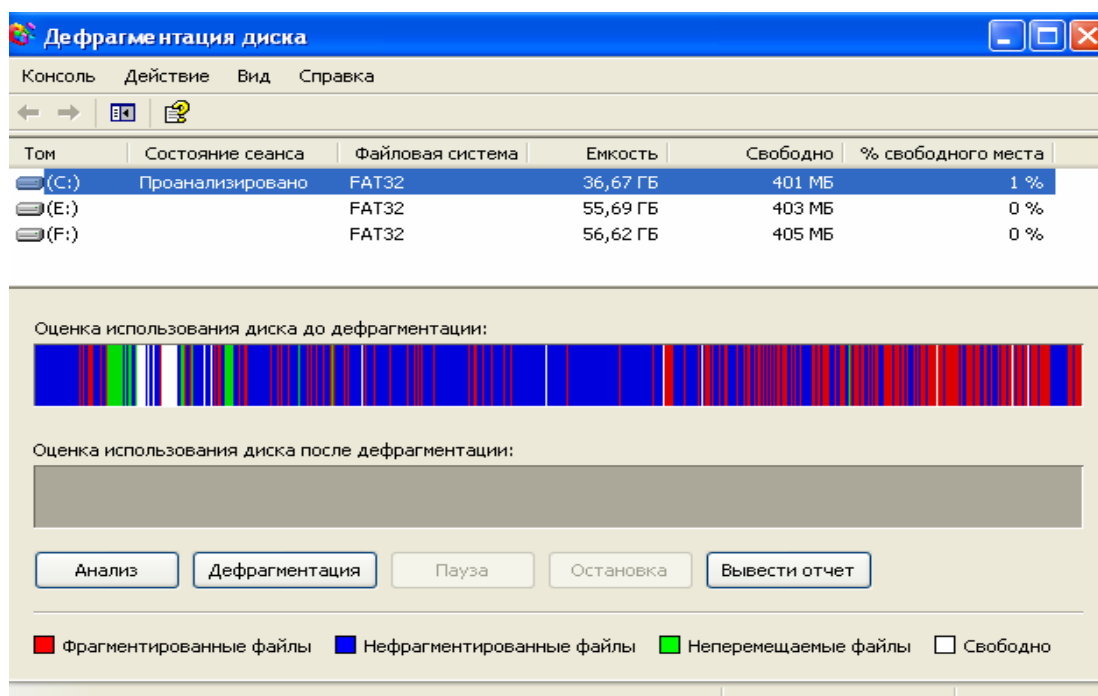


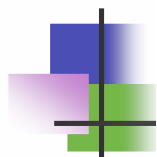
Рис. 6.1 Программы дефрагментации диска в Windows XP

- Программа дефрагментации диска (Рис. 6.1) собирает кластеры каждого файла в одно место. Это обеспечивает более эффективную работу компьютера.
- Программа дефрагментации работает по заданному расписанию, но ее можно запускать и вручную. Она может работать долго — несколько часов.



Форматирование диска

- Перед использованием диск надо подготовить к работе — форматировать.
- При форматировании на диск записываются файл загрузки и таблицы размещения файлов FAT или NTFS.
- Оптические диски и дискеты форматируют перед продажей.
- Для форматирования дисков используют специальные программы, например "Partition Manager".



Архивирование данных

- Для рационального использования внешней памяти данные *сжимают* (*упаковывают*, *архивируют*) – сохраняют только значимую информацию.
- Архивирование используют большинство графических форматов – gif, jpeg и другие.
- Архивирование используется для передачи данных через Интернет. При этом увеличивается скорость передачи данных.



Архиваторы

- Существуют разные способы сжатия информации и различные программы, которые это делают. Такие программы называются программами *архивирования файлов* или *архиваторами* (Рис. 6.2).
- Они используются для создания архивных файлов.
- *Архивный файл* (или *архив*) – это совокупность из одного или нескольких файлов, записанных в сжатом виде в один файл.
- Существуют архивные файлы разных типов: .zip, .rar, .tar и другие. Для их создания используются разные методы архивирования.
- Архиваторы используют также для извлечения файлов из архива – *распаковки* (*розархивирования*).
- Архивный файл может содержать и программу распаковки. Это – *самораспаковывающийся архив*. Он имеет расширение .exe.
- Многие операционные системы имеют встроенный архиватор.

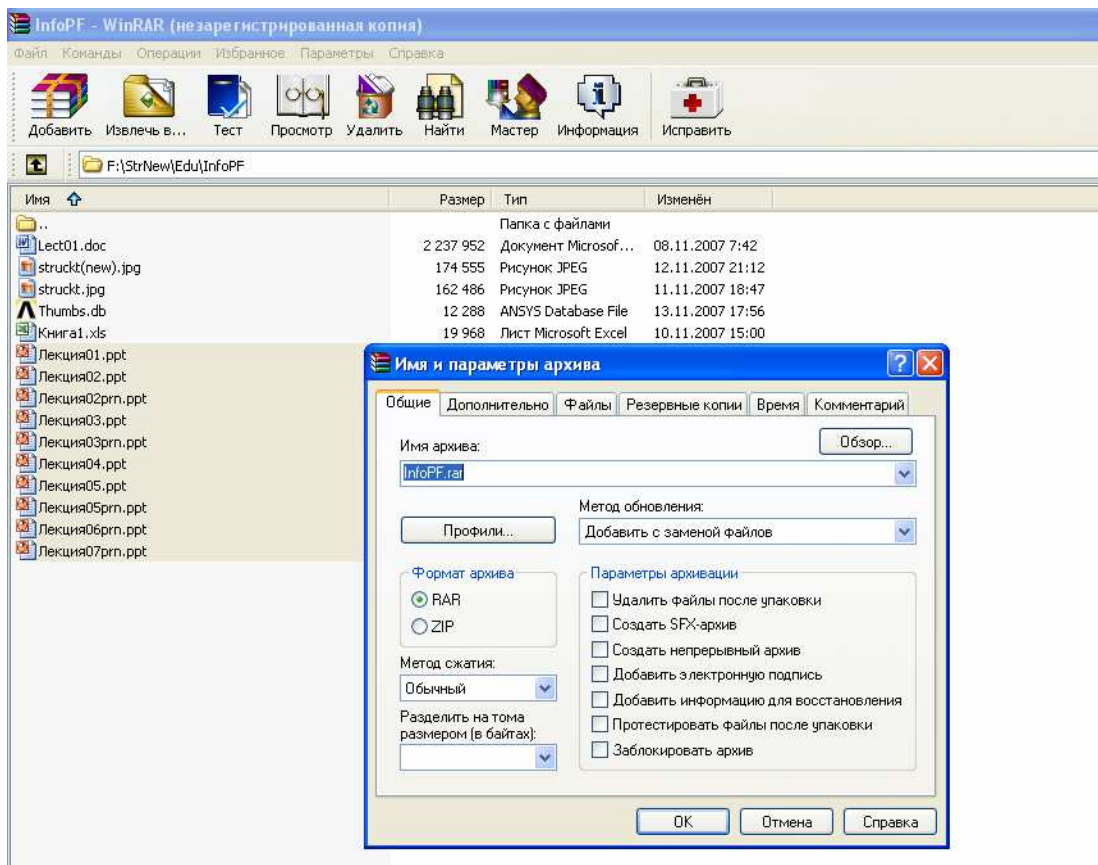


Рис. 6.2 Популярный архиватор WinRAR упаковывает файлы презентаций лекций



Резервное копирование

- Для того чтобы не потерять файлы, которые создаются, изменяются и хранятся на компьютере, надо регулярно выполнять их резервное копирование.
- Резервные копии файлов архивируют. Для этого можно использовать архиваторы.
- Современные ОС (Windows 2000, Vista и др.) имеют *Службу резервного копирования*, которая имеет встроенный архиватор. Она может работать автоматически по заданному расписанию.
- ***Резервное копирование на компьютерах в офисе и на производстве ОБЯЗАТЕЛЬНО.***



Контрольные вопросы

1. Как разбивается поверхность магнитного диска?
2. Что такое сегмент?
3. Какой размер имеет один сегмент?
4. Что такое кластер?
5. Как нумеруются дорожки диска?
6. Что такое таблицы FAT?
7. Где расположены таблицы FAT?
8. Что такое NTFS?
9. Чем отличаются таблицы FAT и NTFS?
10. Что такое файл загрузки?
11. Зачем используется Корзина?
12. Что такое временные файлы?
13. Зачем выполняется программа «Очистка диска»?
14. Что такое форматирование диска?
15. Можно ли восстановить информацию на диске после форматирования?
16. Что такое фрагментация файла?
17. Можно ли собрать вместе кластеры фрагментированного файла?
18. Что такое архивный файл?
19. Как узнать архивный файл?
20. Что такое самораспаковывающийся архив?
21. Зачем используются архивные файлы в сети Internet?
22. Зачем выполняется резервное копирование?
23. Что такое Служба резервного копирования?
24. В каком виде сохраняют резервные копии файлов?



Лабораторная работа

Цель работы. Научиться упаковывать и распаковывать файлы.

Задание. Упаковать игру ALADDIN в архив, который можно записать на дискеты, перенести его в другую папку и распаковать там.

1. С помощью поисковой системы ОС Windows найдите на компьютере папку, в которой есть файл Aladdin.exe.
2. В личной папке создайте временную папку Temp.
3. Запустите программу WinRAR.
4. С помощью кнопки выбора над окном программы сделайте активной найденную папку.
5. Убедитесь, что в активной папке нет архивных файлов. Если архивы в папке есть, удалите их в Корзину. Общий объем файлов в папке не должен быть больше, чем 4 Мбайта.
6. С помощью команды раздела меню **Файл** выделите все файлы в папке для занесения их в архив.
7. С помощью кнопки **Добавить** Панели инструментов вызовите окно архивирования (Рис. 6.2) и укажите в нем:
 - а) Имя архива (Aladd.rar);
 - б) Формат архива (RAR);
 - в) Размер том (1 457 000 байт).
8. Выполните программу.
9. Закройте программу WinRAR.
10. Методом drag-and-drop скопируйте созданные архивные файлы из папки, которая была активной при архивировании, в подкаталог Temp личной папки.
11. Раскройте в папке Temp созданный архив. Для этого на-до:

- а) Выполнить двойной щелчок на имени первого файла архива и тем самым войти в программу WinRAR;
 - б) В окне программы WinRAR выделить архивные файлы;
 - в) Распаковать архив с помощью кнопки **Извлечь** Панели инструментов архиватора.
12. Выйдите из программы WinRAR.
 13. Запустите программу Aladdin.exe в личной папке Temp.
 14. Покажите работу преподавателю.
 15. Завершите работу компьютерной игры Aladdin.
 16. Выйдите из личной папки Temp.
 17. Удалите личную папку Temp в Корзину.
 18. Очистите Корзину.



Практические задания

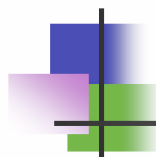
1. Форматировать дискету.
2. Определить размер кластера диска.
3. Очистить Корзину.
4. Запаковать заданные файлы в архив.
5. Извлечь файлы из архива.
6. Очистить диск.
7. Выполнить дефрагментацию диска.

7. Компьютерные вирусы



Ключевые слова

вирус	virus	病毒	فيروس
природа	nature	性質	طبيعته
заражать	infect	感染	اصب
размножать	reproduce	增加	اعد انتاج
присоединять	add	连接	اضافة
носитель	supporter	儲存器	مؤيد
воздействие	effect	效果	تأثير
уничтожать	destroy	摧毁	تدمر
похищать	steal	窃取	سرقة
нарушать	violate	违反	انتهاك
исчезать	disappear	消失	اختف
заполнять	fill	装满	يملأ
размножаться	reproduce	再生	اعد انتاج
лицензия	license	許可	رخصة
этика	ethics	规矩	الاخلاق
червь	worm	蠕蟲	الدودة
троянский	trojan	木馬	طروادة
бактерия	bacteria	細菌	البكتيريا
кролик	rabbit	兔	أرنب
бомба	bomb	炸彈	قنبلة
предохранять	prevent	防止	منع
обнаружить	discover	發現	اكتشف
лечить	treat	對待	يعامل
проверять	check	檢查	فحص
протокол	protocol	记录	بروتوكول



Что такое вирус

- Компьютерные вирусы – это программы, которые запускаются на компьютере без разрешения пользователя.
- Компьютерные вирусы самостоятельно размножают свой код. Они могут присоединять его к другим программам – *заражать* эти программы.
- Программы, которые заражены вирусом, становятся носителями вируса. При своей работе они заражают этим вирусом другие программы.



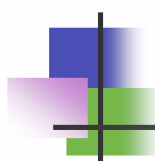
Воздействие на компьютер

- Вирусы по-разному воздействуют на компьютер. Одни из них работают при каждом запуске зараженной программы. Другие ожидают некоторого события, после которого зараженная программа может, например, уничтожить саму себя и одновременно все файлы, которые доступны ей в данный момент времени.
- Не все вирусы опасны. Одни ограничиваются тем, что выводят на экран монитора какое-нибудь сообщение. Другие очень опасны. Они могут:
 - Уничтожить всю или часть информации в компьютере, включая операционную систему;
 - Похищать из компьютера важную информацию коммерческого или личного характера: коды доступа к банковским счетам, контракты, деловые и личные письма;
 - Нарушать работу компьютера.



Вирусы в природе

- Воздействия компьютерного вируса на компьютер похоже на воздействие биологических вирусов на живые организмы.
- Биологические вирусы нарушают информацию, которая находится в генетическом коде живой клетки. Вирус изменяет небольшой участок цепочки ДНК и РНК и захватывает управление жизненными процессами клетки.
- Так биологический вирус обеспечивает себе возможность свободно и неограниченно размножаться.



Проявление вирусов

- Внешним проявлением заражения организма вирусом является болезнь всего организма.
- Внешним проявлением заражения вирусом компьютера являются различные нарушения его работы:
 - Замедление работы программ;
 - Исчезновение файлов из компьютера;
 - Исчезновение свободной памяти;
 - Несанкционированный обмен данными с Интернетоми другие.
- **Замечание.** Не всегда обмен данными с Интернетом связан с наличием вирусов. Компьютер может получать обновления ОС или лицензированных программ, которые установлены на нем.





Типы вирусов



- "Вирусы" –это общее название для программ, которые проникают на чужие компьютеры через другие программы и работают там без разрешения пользователя. Различают такие вирусы, как:
 - "Черви", которые сами могут переходить от машины к машине по сети, выполняя заложенное в них задание;
 - "Троянские кони", которые могут работать без разрешения на чужих компьютерах;
 - "Логические бомбы", которые начинают работать на чужих компьютерах при выполнении некоторых условий;
 - "Бактерии" и "кролики", которые быстро размножаются и заполняют память компьютера.



Антивирусные программы

- Для поиска и уничтожения компьютерных вирусов используют антивирусные программы (Рис. 7.1).
- Работа таких программ основана на изучении структуры текста проверяемой программы в машинных кодах.
- Они позволяют не только обнаруживать вирусы, но и "лечить" зараженные программы – уничтожать изменения, внесенные в программу вирусом.
- Обычно антивирусная программа работает постоянно и в автоматическом режиме защищает компьютер от заражения вирусами.
- Нет антивирусной программы, которая может защитить компьютер от любого компьютерного вируса.
- ***Никто не гарантирован от появления вирусов.***

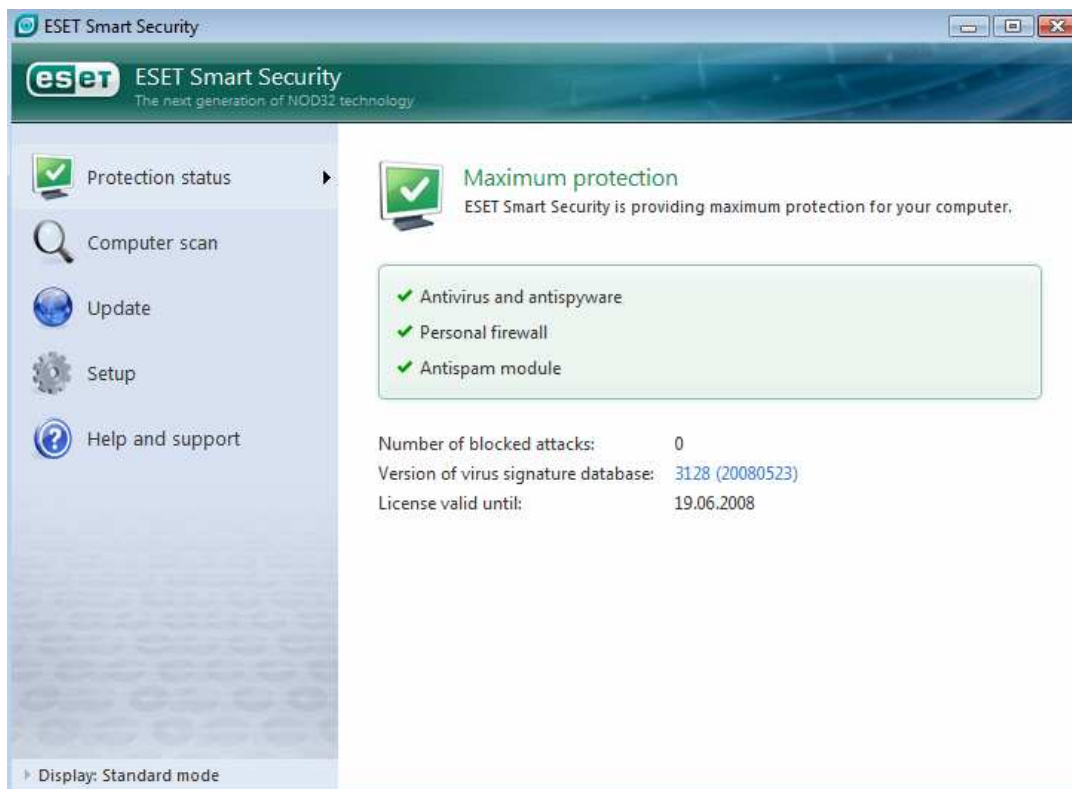


Рис. 7.1 Популярная антивирусная программа NOD32



Компьютерные вирусы и дисциплина

- Вероятность появления вирусов на компьютере можно значительно уменьшить, если придерживаться правил *компьютерной этики* работы на компьютере:
 - Не использовать нелегальных программ;
 - Не переписывать программное обеспечение с других компьютеров, так как оно может быть заражено вирусами;
 - При считывании любой информации с диска проверять ее на наличие вирусов;
 - Диски с файлами, которые не надо изменять, защищать от записи.
 - Использовать новейшие версии антивирусных программ.



Контрольные вопросы

1. Что такое компьютерный вирус?
2. Почему вредоносные программы на компьютере называются вирусами?
3. Как компьютерные вирусы заражают компьютер?
4. Что такое "компьютерный червь"?
5. Что такое "логическая бомба"?
6. Как работает вирус "троянский конь"?
7. Что такое "компьютерный кролик"?
8. Как проявляет себя вирус "бактерия"?
9. Как "вылечить" компьютер, зараженный вирусами?
10. Как предохранять компьютер от появления на нем вирусов?
11. Как работает антивирусная программа?
12. Какая программа может обнаружить любой компьютерный вирус?
13. Как можно получить компьютерный вирус через электронную почту?
14. Когда надо запускать антивирусную программу?
15. Какие программы не содержат вирусов?
16. Какой вирус самый опасный?



Лабораторная работа

Цель работы. Научиться работать с антивирусной программой.

Задание. Выполнить антивирусную проверку компьютера.

1. Узнайте, какая антивирусная программа установлена на Вашем компьютере.
2. Определите дату последнего обновления базы вирусов Вашей антивирусной программы и сделайте вывод, как надежно защищает антивирусная программа Ваш компьютер.
3. Узнайте, когда последний раз выполнялась антивирусная проверка Вашего компьютера. Такую информацию можно найти в протоколе проверки – файле с расширением .log.
4. Узнайте, были ли обнаружены вирусы на компьютере при последней антивирусной проверке.
5. Узнайте, когда последний раз были обнаружены вирусы на Вашем компьютере.
6. Запустите антивирусную программу и проверьте Ваш компьютер.
7. После завершения работы программы просмотрите список зараженных файлов.
8. Покажите результаты Вашей работы преподавателю.



Практические задания

1. Запустить антивирусную программу.
2. Определить дату последнего обновления информационной базы данных антивирусной программы.
3. Определить, когда последний раз на компьютере выполнялась антивирусная проверка.
4. Определить, когда последний раз были обнаружены вирусы на компьютере.
5. Объяснить значения установок антивирусной программы.

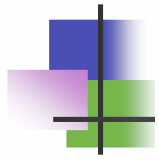
Часть II. ОСНОВЫ ПРОГРАММИРОВАНИЯ

8. Основные понятия



Ключевые слова

программирование	programming	编程	البرمجة
алгоритм	algorithm	算法	خوارزمية
строгий	strict	精确	صارم
ошибка	mistake	错误	خطأ
трудоемкий	laborious	劳动密集的	مضنية
наглядный	obvious	直观	واضح
ассемблер	assembler	汇编程序	التجميع
детальный	detailed	详细	مفصل
компиляция	compilation	汇编	تصنيف
компилятор	compiler	编译	مؤلف
интерпретатор	interpreter	口译	مترجم
исходный	source	来源	مصدر
объект	object	对象	موضوع
исполняемый	executable	可执行	للتنفيذ
процедура	procedure	程序	إجراء
обязательный	obligatory	约束力	ملزم
операнд	operand	操作	العملية الرياضية
суть	heart	心脏	قلب
содержимое	content	内容	محتوى
объект	facility	设施	مرفق
модуль	module	模块	وحدة
запустить	start	开始	يبدأ
сообщение	message	讯息	رسالة



Алгоритмические языки

- Программы для первых компьютеров составлялись в машинных кодах. Эта трудоемкая работа требовала большой внимательности программиста и много времени.
- Сейчас для программирования компьютеров используют алгоритмические языки.
- Алгоритмические языки являются строгими: если в задании есть ошибка, компьютер остановит работу, сообщит об ошибке и ее характере.
- При составлении программ для компьютера используют:
 - Машинные коды;
 - Язык ассемблера;
 - Языки высокого уровня;
 - Объектно-ориентированные языки;
 - Языки символьной обработки.



Машинные коды

- Программа в машинных кодах состоит из команд, записанных в двоичном коде – с помощью 0 и 1.
- Каждая команда имеет две части: код операции и адресную часть.
 - *Код операции* указывает, какое действие должен выполнить процессор.
 - *Адресная часть* указывает, где в памяти компьютера хранятся операнды и куда поместить результат выполнения операции. *Операнды* – это данные, над которыми выполняется действие (*операция*).
- Сейчас программисты редко используют машинные коды, так как это трудоемко и не наглядно.



Язык ассемблера

- Ассемблер – это язык низкого уровня, близкий к машинным кодам.
- Ассемблер могут использовать только опытные программисты, знакомые с архитектурой процессора.
- Отличия ассемблера от машинных кодов состоят в том, что двоичный код операции заменяется буквенным обозначением, подсказывающим суть операции, а двоичные адреса операндов заменяются именами переменных. Например, команда `MOV X1, C` означает: переместить содержимое ячейки памяти, обозначенной как `X1`, в регистр процессора `C` (по-английски `MOVE` означает движение, перемещение).
- Процессор может выполнять только задания, представленные в машинных кодах. Поэтому программы на языке ассемблера перед выполнением переводятся в машинные коды – *ассемблируются*.



Языки высокого уровня

- Язык высокого уровня позволяет записывать программу без детализации каждой операции.
- К языкам высокого уровня относятся Pascal, C, C++, FORTRAN, Java, BASIC и другие.
- Программы, написанные на языке высокого уровня, должны переводиться в машинные коды перед тем, как они выполняются. Такой перевод называется *трансляцией*.
- Одна команда на языке высокого уровня, такая как запись строки текста в файл, может преобразовываться в десятки или даже сотни двоичных машинных команд.

- Перевод программы с языка высокого уровня в машинные коды выполняет специальная программа – *транслятор*. Транслятор входит в состав программного обеспечения компьютера.
- Существует два подхода к построению трансляторов.
- *При первом* из них программа или ее часть переводится с языка высокого уровня в машинные коды и сразу же выполняется. Транслятор, который реализует такой подход, называется *интерпретатором*.
- *При другом* подходе на основе текста программы на языке высокого уровня строится и запоминается в памяти машины исполняемая программа в машинных кодах.
- Исполняемая программа постоянно хранится в компьютере, а выполняется по необходимости. Транслятор, который реализует такой подход, называется *компилятором*.
- С компиляторами связаны понятия:
 - *Исходный код* – текст программы на алгоритмическом языке;
 - *Объектный код* – результат компиляции. Объектный код обычно содержит обращения к другим модулям;
 - *Исполняемый код* – окончательный результат компиляции, машинный код.



Объектно-ориентированные языки

- Объектно-ориентированные языки позволяют создавать программы - приложения для среды Windows.
- Программист создаёт не только текст программы, но и её графический интерфейс. При этом используются разные объекты, их свойства и методы.
- Набор объектов, которые можно использовать, предлагается средой программирования, где реализованы эти языки, и может пополняться пользователем.

- Объектно-ориентированный язык обычно создает не одну программу, а проект – набор процедур, которые связаны с объектами и между собой. Эти процедуры активизируются, если происходят определённые события.
- Популярные среды программирования: Visual Net, VBA.



Языки символьной обработки

- Эти языки базируются на математической логике.
- Они позволяют разрабатывать аналитические и интеллектуальные системы.
- Иногда такие языки называют языками сверхвысокого уровня. Например, язык Prolog.



Среда программирования VBA

- Пакет офисных программ MS Office имеет встроенную среду программирования VBA (*Visual Basic for Application*).

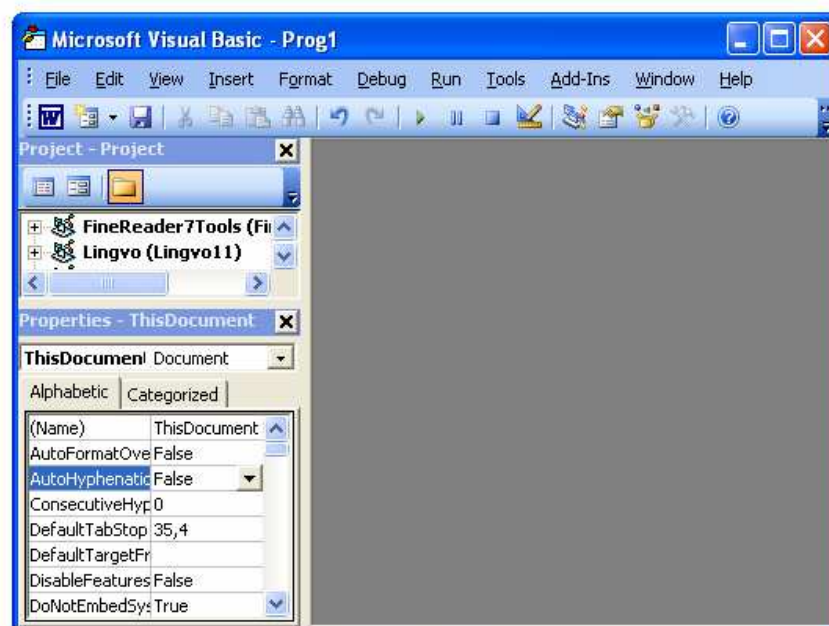


Рис. 8.1 Окно среды программирования VBA

- Среда программирования VBA реализует язык высокого уровня Visual Basic.
- В среду VBA можно войти с любого приложения MS Office командами главного меню

Сервис → Макрос → Редактор Visual Basic

или с помощью комбинации клавиши **Alt+F11**. Эти команды вызывают на экран окно среды программирования VBA (Рис. 8.1).

- Чтобы подготовить VBA к вводу программы, надо с помощью функциональной клавиши **F7** или команд главного меню среды

View → Code

вызвать окно кода программ (Рис. 8.2).

- Программа в VBA всегда начинается заголовком программы, который имеет вид

Sub ИМЯ ()

где имя – любое английское или русское слово.

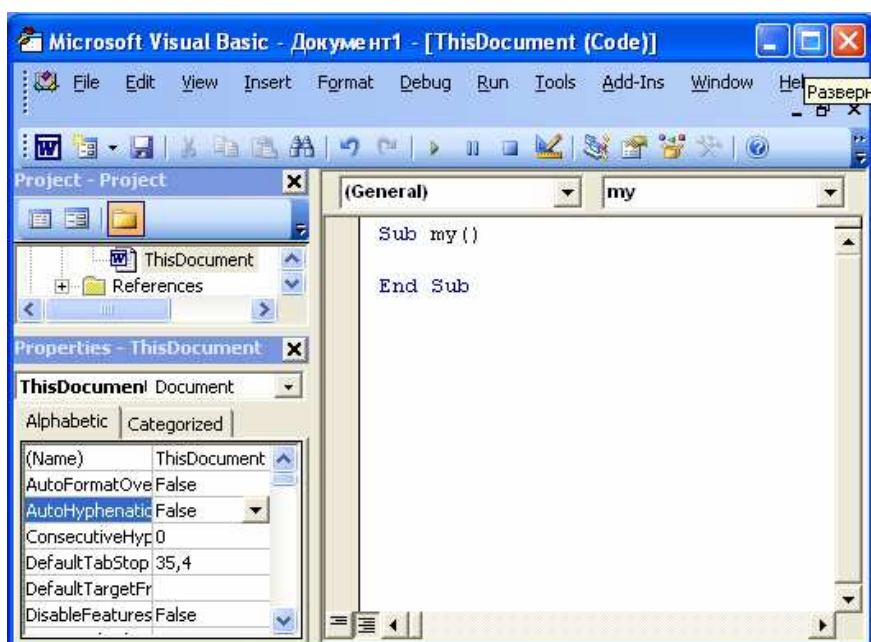


Рис. 8.2 Окно кода программы с введенным заголовком программы

- Если Вы дадите имя своей программе (я дал ей имя my) и введете заголовок программы в окне кода, то VBA доба-

вит через строку второй обязательный элемент кода программы – последнюю строку (Рис. 8.2)

End Sub

- Код текста программы вводится между этими двумя обязательными строками.
- Составим нашу первую простую программу, которая поприветствует программиста. Текст той программы состоит из одной строки (Эту программу написал Мохамед):

```
MsgBox "Привет, Мохамед!"
```

- Здесь **MsgBox** – оператор языка Visual Basic, который выводит на экран текст, записанный после этого оператора в кавычках. Сами кавычки к тексту не относятся.
- Введем эту строку (тело программы) между заголовком и строкой конца программы (Рис. 8.3). Программа готова к выполнению!
- Запустим программу с помощью функциональной клавиши **F5** или команд главного меню среды

Run → RunSub/UserForm

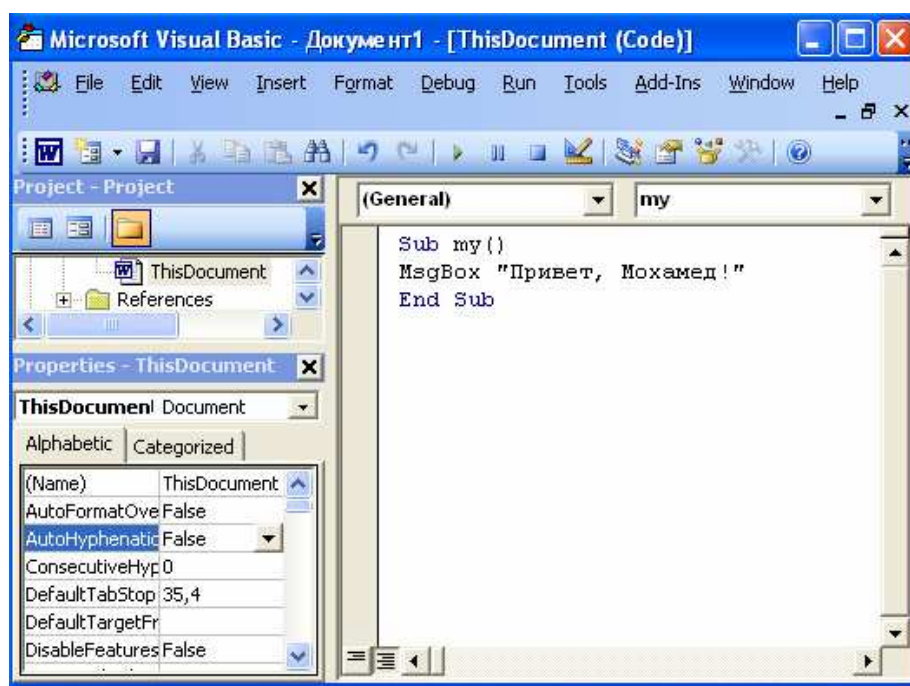


Рис. 8.3 Текст программы в окне кода программы

- Если программа запускается первый раз, то VBA попросит подтвердить ее имя в окне макросов (Рис. 8.4а). Кнопкой **Run** или с помощью клавиши **Enter** подтвердим, что мы действительно хотим выполнить эту программу.
- В результате работы программы на экране появится окно сообщений программы (Рис. 8.4б), в котором будет выведен нужный текст. Программа работает!

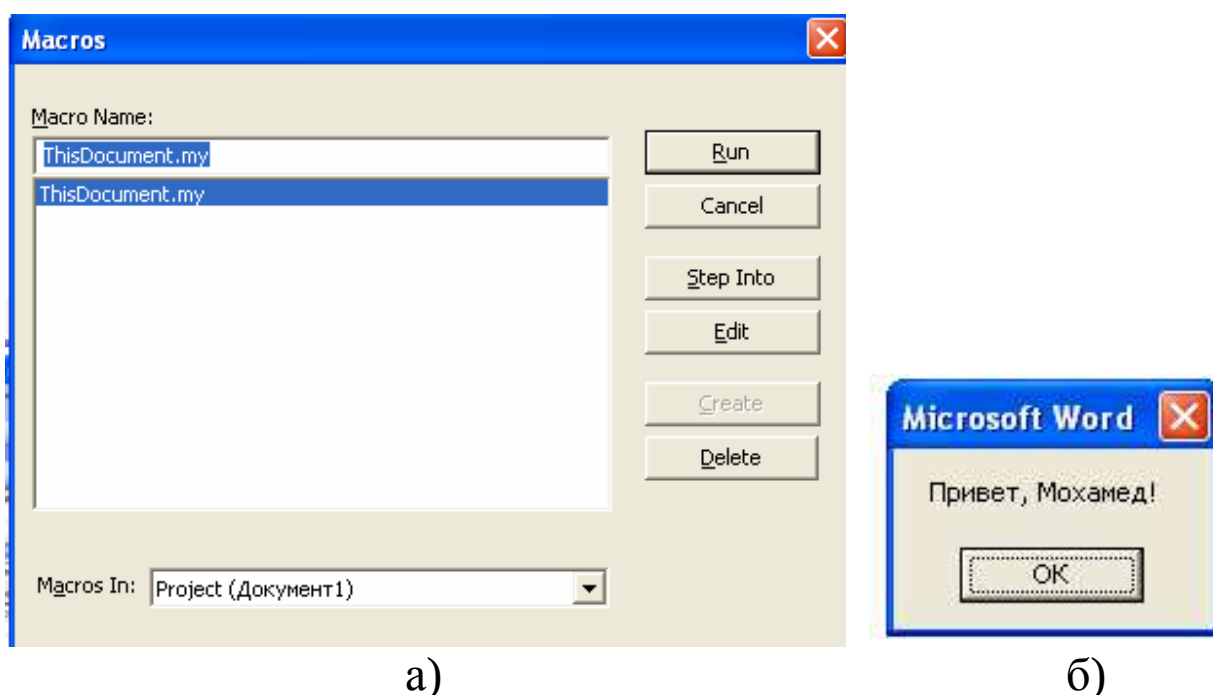


Рис. 8.4 Окно макросов (а) и окно сообщений программы (б)

- Чтобы продолжить работу программы, а для нашей программы – закончить ее работу, – нажмем кнопку **OK** в окне сообщений (Рис. 8.4б).
- Выход из среды VBA выполняется командой **Close and Return..** меню VBA или комбинацией клавиш **Alt+Q**.
- VBA является интерпретатором для языка Visual Basic.
- Исходный код программы хранится вместе с документом приложения MS Office, при работе с которым программа была создана.
- При запуске программы она транслируется и сразу же выполняется – интерпретируются.

- Программы (макросы) в документах пакета MS Office, о которых не знает пользователь, могут быть источником вирусов на компьютере. Поэтому при открытии документов с программами приложения MS Office спрашивает разрешение на использование макросов (Рис.8.5).

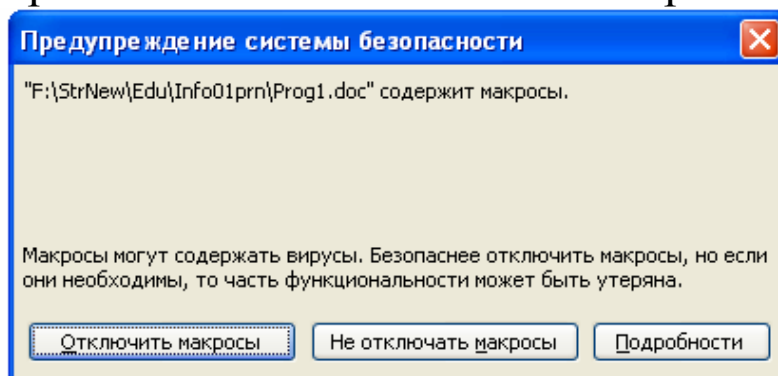


Рис. 8.5 Запрос приложения MS Office на использование программ (макросов)

Приложение MS Office (Word, Excel или Access) может быть настроено так, что программы на языке VBA не выполняются. Эту настройку можно изменить командой

Сервис → Параметры → Безопасность → Защита от макросов.. если выбрать уровень безопасности "Средняя" (Рис. 8.6).

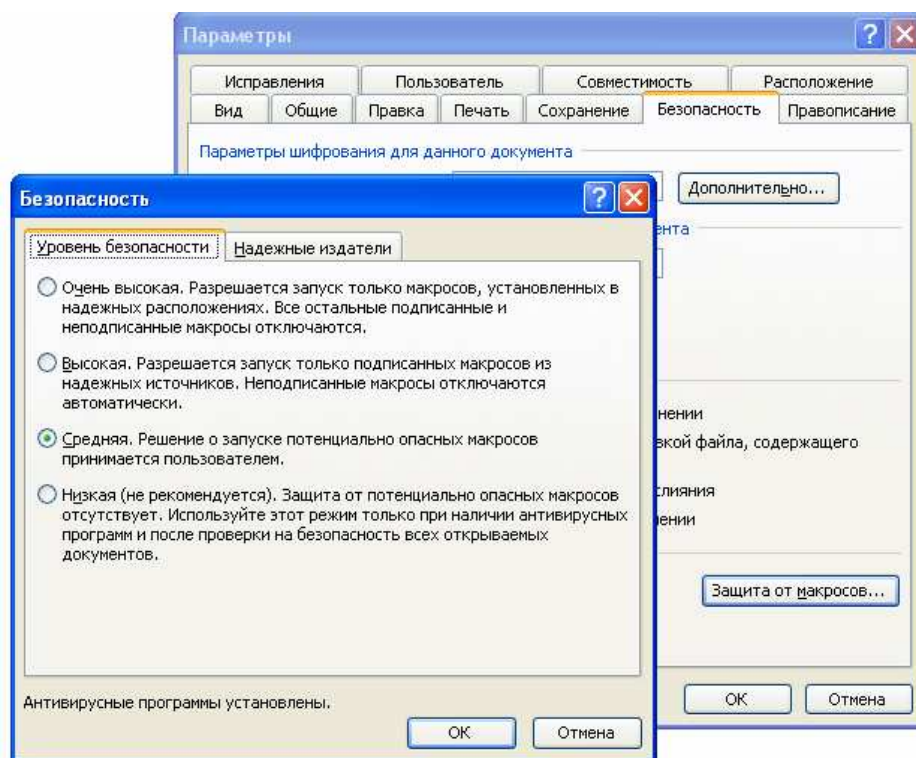


Рис. 8.6 Настройка приложения для работы с VBA



Контрольные вопросы

1. Зачем используются алгоритмические языки?
2. Почему алгоритмические языки называют строгими?
3. Является ли строгим русский язык?
4. Какие типы алгоритмических языков Вы знаете?
5. Что такое язык ассемблера?
6. Что такое ассемблирование программы?
7. Какие алгоритмические языки называют языками высокого уровня?
8. Что такое транслятор?
9. Что такое интерпретатор?
10. Что такое компилятор?
11. Что такое исходный код?
12. Что такое объектный код?
13. Что такое исполняемая программа?
14. Какие алгоритмические языки называют объектно-ориентированными?
15. Зачем используют объектно-ориентированные языки?
16. Зачем используют языки символьной обработки?
17. Что такое VBA?
18. Зачем используется VBA?
19. Какой язык реализует среда программирования VBA?
20. Как вызвать среду программирования VBA?
21. В каком окне среды VBA вводят текст программы?
22. Как запустить программу, введенную в окне кода?
23. Какое действие выполняет оператор **MsgBox** в языке Visual Basic?
24. Как выйти со среды программирования VBA?
25. Как сохранить программу, созданную в среде VBA?
26. Почему MS Office при открытии документов спрашивает разрешение на использование макросов?
27. Как настроить приложение MS Office для работы с VBA?



Лабораторная работа

Цель работы. Научиться вводить и выполнять программы на языке Visual Basic в среде программирования VBA.

Задание. Составить и выполнить в среде VBA программу, которая выведет на экран компьютера сообщение о том, как Вас зовут и в какой стране живут Ваши родители.

Выполнение программы

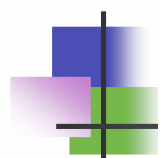
1. Запишите в тетради текст программы, которая выведет Ваше имя. Имя программы запишите английскими буквами.
2. Запустите текстовый редактор Windows Word.
3. С помощью функциональных клавиш приложения:
 - а) войдите в среду программирования VBA: **Alt+F11**;
 - б) откройте окно кода: **F7**;
 - в) введите в окне кода текст программы;
 - г) запустите программу и выполните ее: **F5**;
 - д) выйдите из среды VBA: **Alt+Q**.
4. Запишите в окне приложения текст "Мои первые программы", сохраните файл в личной папке и закройте его.
5. Откройте документ с программой. При открытии разрешите использовать макросы.
6. Запишите в тетради текст программы, которая сообщит, в какой стране живут Ваши родители. Имя программы запишите русскими буквами.
7. С помощью команд главного меню приложения:
 - а) Войдите в среду программирования VBA: **Сервис → Макрос → Редактор Visual Basic**;
 - б) При входе в среду VBA окно кода откроется автоматически. В нем уже будет записана Ваша первая программа. Если окно кода не появится, откройте его командами: **View → Code**;

- в) Перейдите в окне кода в новую строку после конца Вашей первой программы и, начиная с этой строки, введите Вашу вторую программу;
 - г) Запустите эту программу: **Run → RunSub/UserForm**;
 - д) Выйдите из среды VBA: **File → Close and Return..**
8. Сохраните файл.
9. Покажите Вашу работу преподавателю.



Практические задания

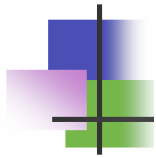
1. Запустить среду программирования VBA из текстового редактора MS Word и выйти из нее с помощью команд главного меню.
2. Запустить среду программирования VBA из текстового редактора MS Word и выйти из нее с помощью комбинации клавиш.
3. Запустить среду программирования VBA из табличного процессора MS Excel и выйти из нее.
4. В среде VBA открыть окно кода с помощью команд меню среды VBA.
5. В среде VBA открыть окно кода с помощью функциональной клавиши.
6. В среде VBA запустить программу с помощью команд меню среды VBA.
7. В среде VBA запустить программу с помощью функциональной клавиши.



9. Простейшие программы

Ключевые слова

переменная	variable	变量	متغير
оператор	operator	运营商	مشغل
действие	action	行动	عمل
комментарий	comment	评论	تعليق
разделять	separate	单独	مستقل
конструкция	construction	建设	البناء
целое	integer	整数	جمله
дробное	fractional	分数	جزئي
рациональный	rational	合理	عقلاني
иррациональный	irrational	不合理	غير منطقي
вещественный	real	真正的	حقيقي
строка	string	字符串	سلسلة
точно	exactly	正是	بالضبط
приблизительно	approximately	大约	حوالي
степень	degree	权力编号	عدد من
объединить	unite	合并	توحد
единичный	single	单个	مفرد
двойной	double	双	مزدوج
вариант	variant	变	البديل
константа	constant	常数	مستمر
описание	description	说明	وصف
округлить	round	四舍五	جولة
отбросить	throw off	抛开	التخلص من
точность	accuracy	的精度	دقة
недостаток	defect	缺陷	خلل



Структура программы

- В среде программирования VBA реализован алгоритмический язык высокого уровня Visual Basic.
- В алгоритмическом языке программа состоит из операторов.
- В языках высокого уровня операторы записываются с помощью ключевых слов, имен переменных и знаков действий.
- В Visual Basic каждый оператор записывается на отдельной строке. Несколько операторов можно писать и в одной строке, тогда их разделяют двоеточием (:).
- Количество пробелов и знаков табуляции между отдельными частями оператора не имеет значения.
- В тексте программы после апострофа (') можно записывать *комментарий* – текст, который объясняет операцию и не относится к программе.



Переменные

- Переменные используются в программе для хранения данных.
- *Переменные* – это величины, которые могут принимать разные значения. Переменные различают по именам.
- *Имя переменной* – это совокупность букв, цифр и других символов, которая начинается с буквы.
- Есть имена, которые нельзя давать переменным. Это – *ключевые слова*, которые используются в языке программирования для написания конструкций языка.
- Среда VBA настроена так, что в тексте программы в окне кода ключевые слова изображаются другим цветом (синим цветом).

- В языке Visual Basic для написания имени переменной можно использовать только латинские буквы, цифры и некоторые специальные знаки.
- В среде VBA для записи имени переменной, наряду с английскими буквами, можно использовать буквы национального алфавита, который установлен на компьютере.
- Первым символом имени должна быть буква.
- При записи имени переменной большие и малые буквы не различаются.
- В имени переменной нельзя использовать пробелы.



Тип переменной

- Переменные бывают разных типов в зависимости от типа значений, которые они могут принимать (числовые, текстовые и др.) и диапазона их значений (Таблица 9.1).

Таблица 9.1

Некоторые типы данных в VBA и их описание

Описание	Тип данных	Память, байт	Диапазон
Integer	Целое	2	$[-2^{15}; 2^{15}-1]$
Long	Целое	4	$[-2^{31}; 2^{31}-1]$
Byte	Целое	1	$[0; 255]$
Single	Вещественное	4	$[-2^{128}; 2^{128}-1]$
Double	Вещественное	8	$[-2^{1024}; 2^{1024}-1]$
String	Текст	1 на символ	До 2^{16} символов
Date	Дата	5	До 31.18.9999

- В языке Visual Basic можно использовать также неопределенный тип Variant. Переменная этого типа может правильно хранить и числа, и текст, и дату. Она занимает много места в памяти – 16 байт и один байт на каждый символ.

- В языке Visual Basic тип переменной обычно описывается оператором

Dim переменная **As** тип

Можно сразу описывать несколько переменных одного типа. Например

Dim Цена **As Double**, Стоимость **As Double**

- В описании переменной тип можно не указывать:

Dim переменная

Тогда переменная будет иметь тип `Variant`.

- В программе переменную можно использовать без описания. Тогда эта переменная также будет иметь тип `Variant` и ее можно использовать только в модуле, где она введена.



- В математике числа могут быть целыми и дробными – рациональными и иррациональными. Совокупность целых и дробных чисел образует множество вещественных (действительных) чисел.
- Числа в языке Visual Basic и в других алгоритмических языках различаются способом представления: точно и приближенно.
- Все числа, которые используются для вычисления, и целые и дробные, представляются в компьютере *приближенно*. Числа, которые представляются приближенно, в языке Visual Basic называются *вещественными*.
- Есть задачи, для которых необходимо *точно* представлять целые числа. Например, если требуется выполнить некоторое действие заданное число раз. Числа, которые представляются точно, называются *целыми*.
- В языке Visual Basic переменная *целого типа* используется для точного представления чисел, а переменная *вещественного типа* – для приближенного представления.

- В алгоритмическом языке Basic можно запретить использовать переменные без описания. Для этого перед описаниями нужно записать оператор **Option Explicit**

Использование этого оператора позволяет избегать ошибок, связанных с неправильной записью переменных и использованием их не по назначению.

- Система программирования VBA может автоматически включать оператор **Option Explicit** в код программы. Для этого должна быть включена опция Require Variable Declaration на странице **Editor** команды **Tools** → **Options** главного меню среды VBA.



Текст

- Значением символьной типа String переменной может быть последовательность любых символов, которые воспринимает компьютер. Количество символов при этом называется *длиной* символьной переменной.
- В языке Visual Basic длина символьной переменной может изменяться в процессе выполнения программы и может принимать значения от 0 до $2^{16} - 1 = 327613$.



Числовые и текстовые константы

- В тексте программ, кроме операторов и переменных, используют числовые и текстовые константы.
- Целые числа записывают с помощью цифр и знаков. Например: 7, +7, -113.
- Знак (+) можно опускать. Поэтому записи 7 и +7 обозначают одно и то же число.
- Для вещественных чисел используют две формы записи:

- С десятичной точкой. Например: 0.5, –18.5;
- В экспоненциальной форме: запись 9.3E–3 обозначает число $9.3 \cdot 10^{-3}$ или 0.0093.
- Если целая часть числа равна нулю, то ее можно опускать. Например, 0.5 одинаково с .5, а 0.3E–2 с .3E–2.
- Текстовая константа – это текст в кавычках. Например: "Его зовут Ахмед". Кавычки в тексте повторяют. Поэтому константа "НТУ "ХПИ"" обозначает: НТУ "ХПИ".
- Константы могут иметь имена. Константа с именем описывается оператором:

Const имя = значение

Например,

Const pi = 3.14159



Оператор присваивания

- Переменной можно дать значение с помощью оператора присваивания, который имеет вид

переменная = константа

или

переменной = выражение

- Выражение – это формула в алгоритмическом языке. Выражение строится из переменных и констант, которые соединяются знаками действий и обозначениями функций (Таблица 8.2).
- Для установления порядка действий в выражениях используют круглые скобки.

Таблица 9.2

Основные математические функции

Запись в языке	Математическая запись	Название
Abs(x)	$ x $	Модуль
Sqr(x)	$\sqrt{x}$	Квадратный корень
Sgn(x)	$\text{sign } x = \begin{cases} 1, & x > 0 \\ 0, & x = 0 \\ -1, & x < 0 \end{cases}$	Знак
Int(x)		Целая часть
Fix(x)		Отбрасывание дробной части
Round(x, n)		Округление с точностью до n знаков
Exp(x)	e^x	Экспонента
Log(x)	$\ln x$	Логарифм
Sin(x)	$\sin x$	Синус
Cos(x)	$\cos x$	Косинус
Tan(x)	$\text{tg } x$	Тангенс
Atn(x)	$\text{arctg } x$	Арктангенс

- Оператор присваивания выполняется так: вычисляется выражение в правой части оператора и полученное значение присваивается переменной. Старое значение переменной при этом теряется.
- Для записи оператора присваивания в языке Visual Basic используется знак равенства. Однако этот знак в Visual Basic имеет другой смысл, чем в математике:
 - В математике уравнение $x = x + 1$ не имеет решения;
 - В программировании, если переменная x имеет значение 1, то после выполнения оператора

$$x = x + 1$$

эта переменная будет иметь значение 2.

- Над числами выполняются арифметические действия с учетом их приоритета в таком порядке:
 - Возведение в степень (^);
 - Умножение (*) и деление – обычное (/) и целое (\);
 - Остаток от деления (Mod);
 - Сложение (+) и вычитание (–).
- Для текстовых строк выполняется операция объединения, которая также обозначается знаком (+). Если типы данных разные, то для этого используют операцию (&).



Программа 1

- Эта программа вычисляет и выводит сумму двух постоянных чисел 2 и 3.

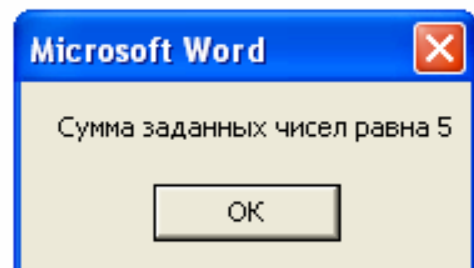
```
Sub Prog1()
  Dim a, b, c As Integer
  a = 2: b = 3: c = a + b
  MsgBox c
End Sub
```



- Видим, что в одной строке программы можно записывать несколько операторов. Тогда их разделяют двоеточием (:).

- В приведенной программе непонятно, какой результат выводится. Это можно исправить, если сформировать строку вывода, как в программе:

```
Sub Prog2()
  Dim a, b, c As Integer
  Dim Text As String
  a = 2: b = 3: c = a + b
  Text = "Сумма заданных чисел равна " & c
```



```
MsgBox (Text)
```

```
End Sub
```

- Общий недостаток обеих программ в том, что они складывают два конкретных числа. Следующая программа находит сумму двух любых целых чисел.

```
Sub Prog3()
```

```
Dim a, b, c As Integer
```

```
Dim Text As String
```

```
Text = "Введите первое слагаемое"
```

```
a = Val(InputBox(Text))
```

```
Text = "Введите второе слагаемое"
```

```
b = Val(InputBox(Text))
```

```
c = a + b
```

```
Text = "Сумма заданных чисел " & a & " и " & b & "  
равна " & c
```

```
MsgBox (Text)
```

```
End Sub
```

- В этой программе используется функция **InputBox** ввода данных с помощью диалогового окна (Рис. 9.1).
- Значением функции **InputBox** является строка текста.

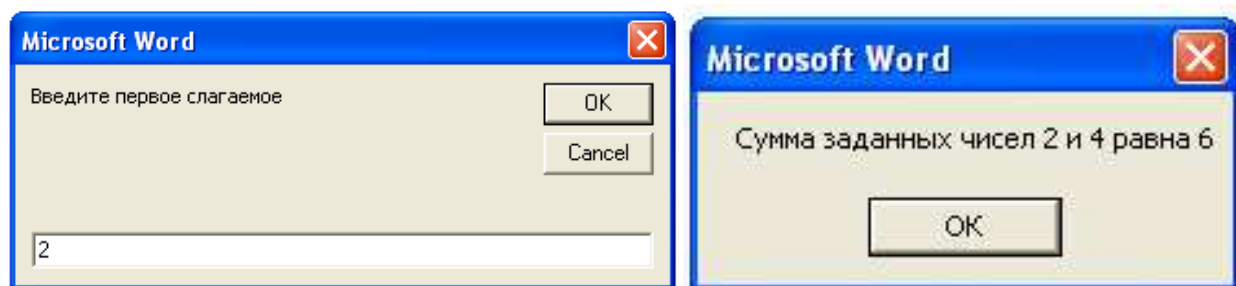


Рис. 9.1 Диалоговые окна ввода и вывода данных

- Для преобразования текста в число используется функция **Val**. Эта функция может брать из текста только его часть до появления нечисловых данных. Например, `Val(" 1615 198th Street N.E.")` даст 1615198.

- Функцию Val нельзя использовать для ввода дробей с десятичной запятой, а не точкой. Для этого надо использовать функции преобразования типов из следующей Табл. 9.3, аргументом которых есть текст.

Таблица 9.3

Функции преобразования типов

Функция	Тип результата
cInt	Integer
cLng	Long
cByte	Byte
cSng	Single
cDbl	Double
cStr	String
cdate	Date

- В диалоговых окнах ввода и вывода данных можно использовать заголовки, которые объясняют действия (Рис. 9.2). Для этого можно использовать программу:

```
Sub Prog4()
Dim a, b, c As Integer
Dim Title, Text As String
Title = "Сложение чисел"
Text = "Введите первое слагаемое"
a = CInt(InputBox(Text, Title))
Text = "Введите второе слагаемое"
b = CInt(InputBox(Text, Title))
c = a + b
Title = "Результат"
Text = "Сумма заданных чисел " &a &" и " &b _
&" равна " &c
Response = MsgBox(Text, , Title)
End Sub
```

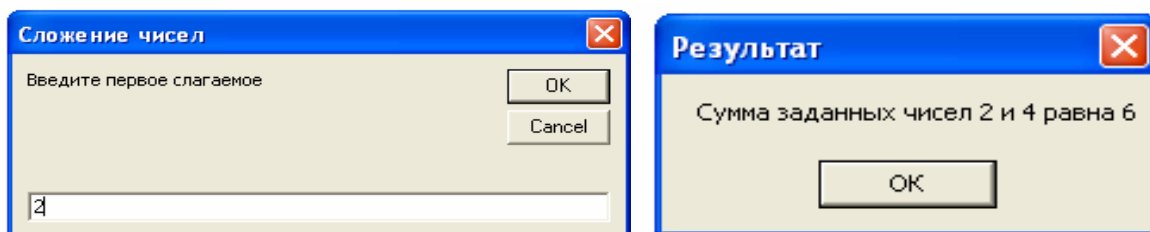
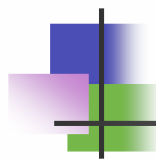


Рис. 9.2 Диалоговые окна программы с заголовками

- Обратите внимание на другой способ формирования окна вывода. Здесь уже используется не оператор, а функция **MsgBox**. Ее значение присваивается переменной **Response**.
- Переменная **Response** не описана в программе, и поэтому она имеет тип **Variant**.
- Значение переменной **Response** в программе не используется. Она введена только для того, чтобы можно было использовать функцию **MsgBox**.
- Обратите внимание на две запятые в операторе вызова функции **MsgBox**. Их запись обозначает, что функция **MsgBox** имеет второй параметр, который в программе не используется.
- Знак подчеркивания (_) используется для переноса оператора на другую строку.



Программа 2

- Следующая простая программа использует ввод текстовых данных в диалоге.
- Программа спрашивает имя пользователя и приветствует его.

```
Sub Hello()  
  Dim name As String  
  name = InputBox("Как Вас зовут?")  
  MsgBox "Здравствуйте, " + name + "!"  
End Sub
```
- Здесь для объединения фрагментов текста используется операция (+), потому что объединяются элементы одного типа.
- Обратите внимание на пробелы, которые используются в константах для формирования правильного текста.

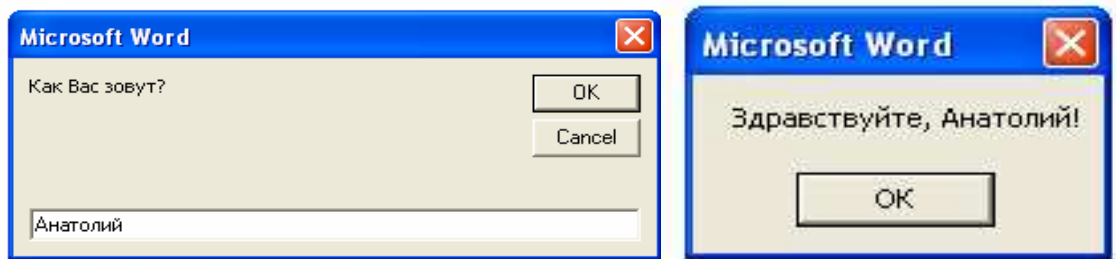


Рис. 9.3 Диалоговые окна программы Hello



Программа 3

- Эта программа вычисляет по заданному значению x значение выражения $y = \frac{x - \sqrt{x^2 + 1}}{x + \frac{1}{x^4 + 1}} e^{\frac{1}{x^2 + 1}} + \sin \frac{\pi}{2} x$.

```
Sub Formula()  
Dim x, y As Double  
Const pi = 9.14159  
x = Val(InputBox("Введите X = "))  
y = (x - Sqr(x ^ 2 + 1)) / (x + x / (x ^ 4 + 1))  
y = y * Exp(1 / (x ^ 2 + 1))  
y = y + sin(pi / 2 * x)  
MsgBox "y = " & y  
End Sub
```

- **Анализ.** В задачах на вычисление используются вещественные переменные двойной точности типа Double.
- При построении таких программ важно правильно построить выражение.
- В программе вычисление формулы разбито на три оператора. Это не обязательно, но так легче проверить правильность вычислений.
- Для того чтобы было удобно писать и читать формулу, в программе используется константа pi.

- Операции одного порядка (+ и –) или (* и /) выполняются подряд слева направо. Обратите внимание, как записано выражение $\frac{\pi}{2}x$ в программе.
- При вычислении выражений сразу выполняются операции возведения в степень (^), затем умножение и деление (* и /), затем сложение и вычитание.
- ***Знак умножения в выражениях опускать нельзя!***

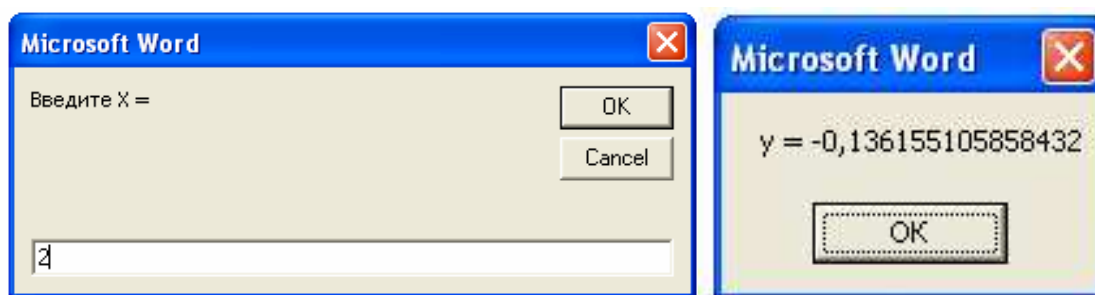


Рис. 9.4 Диалоговые окна программы Formula



Контрольные вопросы

1. Что такое переменная?
2. Какой алфавит можно использовать для написания имени переменной в языке Visual Basic?
3. Какой алфавит можно использовать для написания имени переменной в VBA?
4. Что такое ключевое слово?
5. Что такое комментарий?
6. Как записывают комментарии в языке Visual Basic?
7. Что такое оператор?
8. Можно ли записывать несколько операторов в одной строке в среде программирования VBA?
9. Какие типы переменных Вы знаете?
10. Какие типы переменных можно использовать для хранения целых чисел?

11. Какие типы переменных можно использовать для хранения дробных чисел?
12. Для чего используют переменные целого типа?
13. Что такое строки в языке Visual Basic?
14. Какие операции можно выполнять над целыми числами и нельзя выполнять над вещественными числами?
15. Какая операция объединяет две строки?
16. Какая операция может сформировать строку из строки и числа?
17. Что такое переменная типа Variant?
18. Как описать переменную типа Variant?
19. Можно ли использовать в программе переменные без описания?
20. Для чего используется оператор Option Explicit?
21. Как можно запретить использовать в бреде VBA переменные без описания?
22. Что такое выражение?
23. Зачем используют скобки в выражениях?
24. Каким оператором начинается любая программа?
25. Каким оператором заканчивается любая программа?
26. Какие действия выполняет оператор **MsgBox**?
27. Чем отличаются оператор и функция с именами **MsgBox**?
28. Какие действия выполняет функция **InputBox**?



Лабораторная работа

Цель работы. Научиться строить и выполнять в среде программирования VBA простейшие программы.

Задание. С помощью преподавателя выберите задачу и постройте для нее программу. Документ с программой запомните в личной папке.

Задачи.

1. Составьте программу, которая находит частное и остаток от деления двух заданных целых чисел.
2. Составьте программу, которая находит целую и дробную часть заданного вещественного числа.
3. Составьте программу, которая спросит отдельно Ваше имя и Вашу фамилию, а затем приветствует Вас, назвав Ваше полное имя.
4. Составьте программу, которая находит целое частное и остаток от деления $a+b$ на c , где a , b и c – целые числа.
5. Составьте программу, яка вводит два числа и находит их среднее арифметическое.
6. Составьте программу, которая находит гипотенузу и площадь прямоугольного треугольника по его катетам.
7. Составьте программу, которая находит площадь треугольника по его сторонам.
8. Составьте программу, которая находит длину окружности и площадь круга по его радиусу.
9. Составьте программу, которая переводит в часы, минуты и секунды время, заданное в секундах.
10. Составьте программу для вычисления выражения $\frac{0,5x + 15y - 3xy}{1 + 1,5x - 9y}$ при заданных значениях x и y .

10. Операторы управления и алгоритмы



Ключевые слова

управлять	control	控制	مراقبة
условный	conditional	条件	شرط
логический	logical	逻辑	منطقي
выражение	expression	表达	التعبير
истина	true	正确的	صحيح
ложь	false	虚假	زائف
булевский	boolean	布尔	منطقي
простой	simple	简单	بسيطة
составной	compound	大院	مركب
направленный	directed	针对	توجيه
алгоритм	algorithm	算法	خوارزمية
линейный	line	行	خط
разветвляющийся	bifurcal	分叉	ذو شعبتين
циклический	cyclical	周期性	الدورية
блок-схема	flowchart	流程图	لسير
прямоугольник	rectangle	矩形	المستطيل
параллелограмм	parallelogram	平行四边形	متوازي الاضلاع
ромб	rhombus	菱形	المعين
эллипс	ellipse	椭圆形	القطع الناقص
коннектор	connector	连接器	موصل
ошибка	mistake	错误	خطأ
устранять	remove	删除	يزيل
отладка	debug	调试	صحح
скидка	discount	折扣	الخصم



Порядок выполнения

- Операторы программы выполняются в естественном порядке – последовательно одни за другим так, как они записаны в тексте программы.
- Есть операторы, которые могут изменять порядок выполнения – операторы управления.
- Операторы управления бывают двух типов:
 - *Условные операторы*, результат работы которых зависит от некоторого условия.
 - *Безусловные операторы*, которые работают всегда одинаково.
- Условные операторы записывают с помощью логических выражений.



Логические выражения

- Логические выражения бывают простые и составные.
- *Простые* логические выражения строятся из переменных с помощью операций сравнения (Таблица 10.1).

Таблица 10.1
Операции сравнения в языке Visual Basic

Операция	Название	В математике
<	меньше чем	<
>	больше чем	>
=	равно	=
<=	меньше чем или равно	≤
>=	больше чем или равно	≥
<>	не равно	≠

- Логическое выражение может принимать одно из двух логических значений:
 - Да (правильно, True);
 - Нет (неправильно, False).
- В языке Visual Basic есть переменные логического типа, которые могут принимать значения True и False.
- Логические переменные описываются оператором
Dim переменная **As Boolean**
- Логическое значение принимает и переменная типа Variant, если ей присвоить значение логического выражения, например

$$\text{Cond} = x \lt;> y$$
 или *логической константы*, например

$$\text{Cond} = \text{False}$$
- *Составные* логические выражения строятся из логических выражений и переменных с помощью скобок и знаков логических операций **Not**, **And**, **Or**, **Xor** и др.
- Операция **Not** (отрицание) всегда дает противоположное значение – если $x = \text{True}$, то **Not** x будет False, и наоборот, если $y = \text{False}$, то **Not** y будет True.
- Операции **And**, **Or** и **Xor** двуместные и действуют по правилам Таблицы 10.2. В этой таблице через Т обозначено True, а через F обозначено False.

Таблица 10.2

Логические операции в языке Visual Basic

x	y	x And y	x Or y	x Xor y
T	T	T	T	F
T	F	F	T	T
F	T	F	T	T
F	F	F	F	F

- Из Таблицы 10.2 видно, что x **And** y имеет значение True, если *оба* операнда есть True, а x **Or** y имеет значение True, если *хотя бы один* операнд есть True.



Оператор If

- Оператор **If** позволяет *разветвлять* программу по результату логической операции.
- Есть два тип этого оператора – однонаправленный и двунаправленный.
- Для *однонаправленного* оператора

If условие **Then** действие

оператор действие после **Then** выполняется только тогда, когда логическое выражение условие правильно – принимает значение True .

- Если после **Then** надо выполнить не один, а несколько операторов, однонаправленный оператор используется в виде:

If условие **Then**

оператор

.....

оператор

End If

- *Двунаправленный* оператор

If условие **Then** действие1 **Else** Действие 2

выполняет оператор действие1, если логическое выражение условие правильно (True), и оператор действие2 , если условие неправильно (False).

- После **Then** и **Else** также можно записывать группы операторов:

If условие **Then**

оператор

.....

Оператор

Else

оператор

.....

Оператор

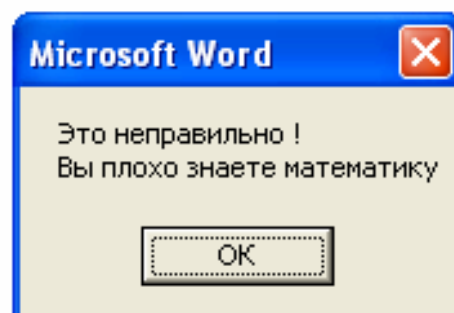
End If



Программа 1

- Следующая простая программа-шутка позволит узнать, знает ли студент математику.

```
Sub MathExam()  
' Программа проверяет знания студента  
' Программист: Кешав Радж Махарджан  
Dim a, b, c As Integer  
Dim ans As Integer  
Dim yn As String  
  
yn = InputBox("Вы знаете математику (д/н)?")  
  
If yn = "д" Or yn$ = "д" Then  
' Если знаете  
    a = Val(InputBox("Введите любое число "))  
    b = Val(InputBox("Введите другое число "))  
    c = a + b  
    ans = Val(InputBox("Чему равно " & a & " + " &  
b))  
    If ans <> c Then  
        MsgBox "Это неправильно !" & Chr(13) & Chr(10) _  
& "Вы плохо знаете математику"  
    Else  
        MsgBox "Правильно !"  
    End If  
Else  
' Если не знаете  
    MsgBox "Это очень плохо !"  
End If  
' Работа закончена  
End  
  
End Sub
```



- В этой программе хороший стиль программирования:
 - Разные логические части программы отделяются пустыми строками;
 - Комментарии объясняют, что делает программа;
 - После заголовка стоит описание программы.

- Чтобы вывести текст в две строки оператором **MsgBox**, в программе текст был разбит добавлением в него двух *машинных кодов управления выводом*:
 - Chr(10)– перейти на новую строку;
 - Chr(13)– установить начало строки.
- В программе использован составной условный оператор: после **Else** снова стоит условие. Такие операторы в Visual Basic имеют вид:


```

      If условие1 Then
          Оператор1
      ElseIf условие2 Then
          Оператор2
          .....
      Else ОператорN
      EndIf
      
```
- Последнее **Else** здесь может отсутствовать, и тогда последний (только последний!) вложенный условный оператор будет однонаправленным.

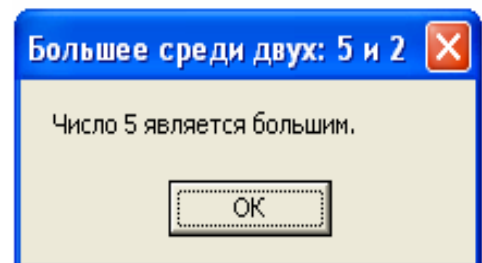


Программа 2

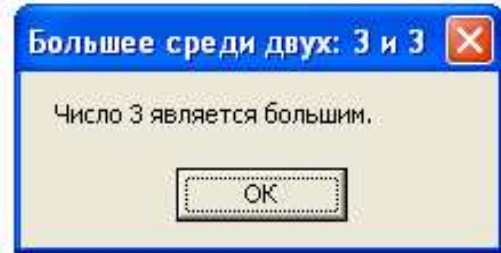
- **Задача.** Составить программу, которая вводит два целых числа и находит среди них большее.
- Казалось бы, задачу решает программа:

```

Sub MaxTwo()
Dim a, b, c As Integer
Dim Title, Text As String
Title = "Нахождение большего"
Text = "Введите первое число"
a = CInt(InputBox(Text, Title))
Text = "Введите второе число"
b = CInt(InputBox(Text, Title))
If a > b Then c = a Else c = b
Title = "Большее среди двух: " & a & " и " & b
Text = "Число " & c & " является большим."
Response = MsgBox(Text, , Title)
End Sub
  
```

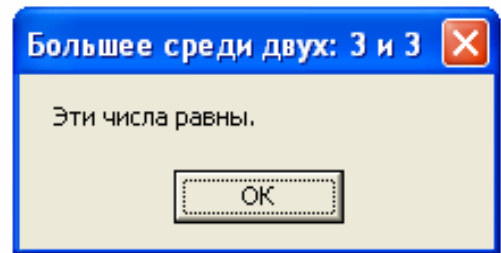


- **Анализ** работы этой программы при разных значениях исходных данных показывает, что если два числа равны, то программа дает неправильный результат – одно из чисел она называет большим.
- Правильный результат получим, если отдельно выделим случай, когда числа равны:



```
Sub Max2a()  
Dim a, b As Integer  
Dim Title, Text As String
```

```
Title = "Нахождение большего"  
Text = "Введите первое число"
```



```
a = CInt(InputBox(Text, Title))  
Text = "Введите второе число"  
b = CInt(InputBox(Text, Title))  
Title = "Больше среди двух: " & a & " и " & b
```

```
If a = b Then  
    Text = "Эти числа равны."  
ElseIf a > b Then  
    Text = "Число " & a & " является большим."  
Else  
    Text = "Число " & b & " является большим."  
End If
```

```
Response = MsgBox(Text, , Title)  
End Sub
```

- Это совсем другая программа! Здесь нет переменной `c`. Попытка "улучшить" предыдущую программу может привести к новым ошибкам.
- Ситуация, с которой мы встретились в этой задаче, для программистов обычная: после создания программы ее надо *отладить* – обеспечить правильную работу при всех допустимых значениях входных данных.
- В процессе отладки ошибки обнаруживаются часто. ***Ошибок не надо бояться, их надо устранять.***



Алгоритмы

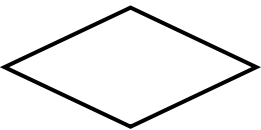
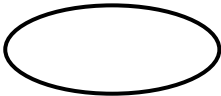
- Алгоритм – это порядок действий, которые надо выполнить, чтобы решить некоторую проблему.
- Проблема может быть любая: математическая (решить задачу), экономическая (получить максимальную прибыль), бытовая (приготовить еду) и др. (Рис. 10.1).
- Алгоритмы бывают:
 - *Линейные*, когда действия выполняются последовательно одно за другим. Линейные алгоритмы использовались при составлении программ предыдущей темы.
 - *Разветвляющиеся*, когда следующее действие зависит от некоторого условия. Такие алгоритмы использовались при составлении программ этой темы.
 - *Циклические*, когда часть действий повторяется.
- Алгоритм можно описать:
 - Словами;
 - Графически – с помощью *блок-схем* (Рис. 10.1);
 - С помощью алгоритмических языков. Приведенные ранее программы являются одновременно и описаниями алгоритма.



Рис. 10.1 Графическое описание алгоритма

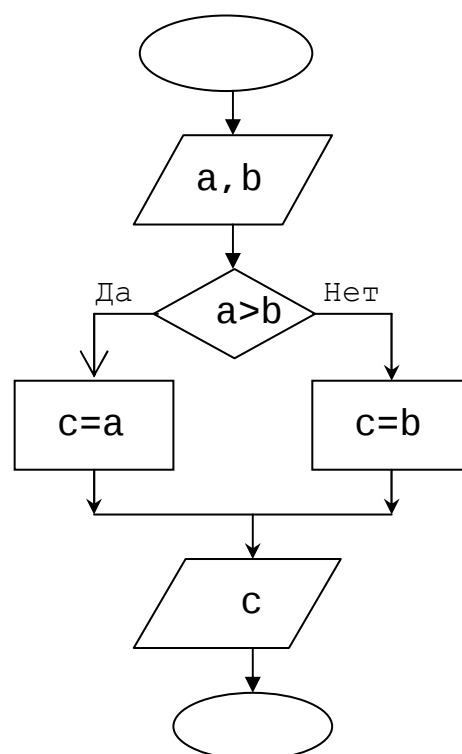
- Для графического описания алгоритмов используют логические схемы или блок-схемы (Рис. 10.1).
- Логические схемы строятся с помощью геометрических фигур (Таблица 10.3).

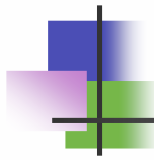
Таблица 10.3
Элементы логических схем

Фигура	Назначение
	<i>Прямоугольник</i> используется для обозначения действия.
	<i>Ромб</i> обозначает проверку условия в виде <i>логического выражения</i> или <i>вопроса</i> , на который могут быть ответы «да» или «нет».
	<i>Параллелограмм</i> используется для обозначения ввода данных или вывода результатов.
	<i>Эллипс</i> указывает начало или конец алгоритма

- Логические схемы используют тогда, когда они делают алгоритм более наглядным и понятным.
- Так алгоритм Программы 1, которая выбирает большее число из двух, описывается логической схемой Рис. 10.2.
- Эта схема удобна тем, что она не содержит ненужных деталей.
- Не надо использовать логические схемы там, где они не нужны.

Рис. 10.2 Логическая схема программы выбора большего числа из двух





Программа 3

- **Задача.** Составить программу, которая находит наибольшее из трех чисел.
- Будем считать, что известен алгоритм нахождения большего числа из двух (Рис. 10.2).
- Иногда удобнее использовать для этих целей алгоритм *max2* Рис. 10.3. Он не содержит ненужных деталей и легче обобщается.
- В логической схеме Рис. 10.3 введен еще один элемент – *коннектор*. Этот элемент удобно использовать для включения данного алгоритма в более общий.
- Для реализации алгоритма Рис. 10.2 в языке Visual Basic введена *функция*

Iif(условие, выражение1, выражение2)

которая используется так:

$c = \text{Iif}(a > b, a, b)$

- Используя алгоритм *max2* как известный, логическую схему поставленной задачи можно описать линейным алгоритмом:

$m = a$

$m = \text{max2}(m, b)$

$m = \text{max2}(m, c)$

- Решение поставленной задачи дает следующая программа.

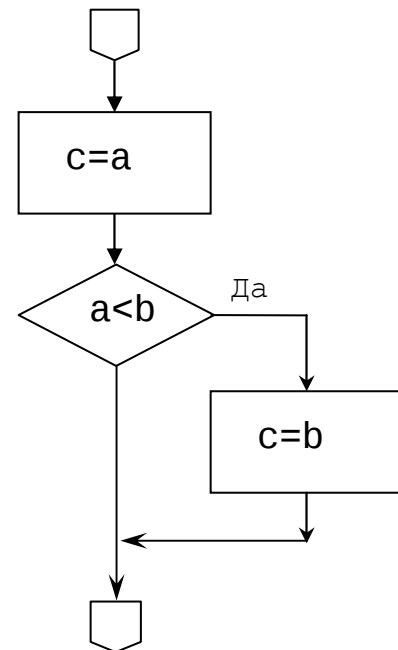


Рис. 10.3 Логическая схема алгоритма *max2*

```

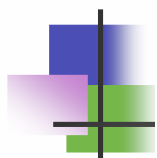
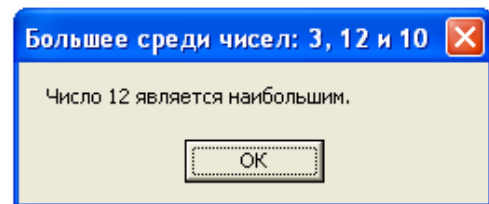
Sub Max3()
Dim a, b, c, m As Integer
Dim Title, Text As String

Title = "Нахождение большего из 3-х целых чисел"
Text = "Введите первое число"
a = CInt(InputBox(Text, Title))
Text = "Введите второе число"
b = CInt(InputBox(Text, Title))
Text = "Введите третье число"
c = CInt(InputBox(Text, Title))

Title = "Большее среди чисел: "&a &", " &b & " и " &c
If a = b And b = c Then
    Text = "Эти числа равны"
Else
    m = a
    If b > m Then m = b
    If c > m Then m = c
    Text = "Число " & m & " является наибольшим."
End If

Response = MsgBox(Text, , Title)
End Sub

```



Программа 4

- **Задача.** Составить программу, которая вычисляет сумму цифр данного трехзначного числа.
- **Алгоритм.** Без компьютера эта задача решается просто. Но для того, чтобы компьютер решил такую задачу, надо уметь вычислить цифры числа с помощью операций, которые можно использовать.
- Для решения задачи воспользуемся тем, что:
 - Последнюю цифру целого числа дает остаток при делении этого числа на 10 (операция **mod**).
 - Целочисленное деление (\) на 10 дает число без последней цифры.
- Например,

$$325 \bmod 10 = \underline{5}, \quad 325 \backslash 10 = 32, \quad 32 \bmod 10 = \underline{2}, \quad 32 \backslash 10 = \underline{3}$$

- С помощью этих операций можно записать следующий линейный алгоритм: если $x = \overline{abc}$, то:
 - 1) $c = x \bmod 10$
 - 2) $x = x \backslash 10$
 - 3) $b = x \bmod 10$
 - 4) $a = x \backslash 10$
- Этот алгоритм работает правильно только для трехзначных чисел, когда $100 \leq x \leq 999$.
- Программа, которая проверяет это условие и реализует такой алгоритм, имеет вид:

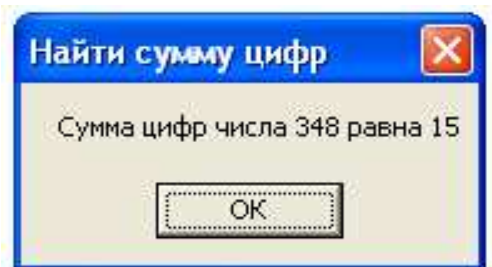
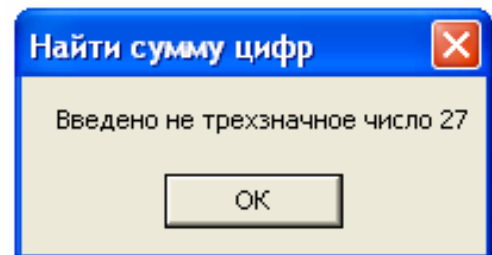
```
Sub СуммаЦифр3()  
Dim a, b, c, x, s As Integer  
Dim Title, Text As String
```

```
Title = "Найти сумму цифр"  
Text = "Введите трехзначное целое число"  
x = CInt(InputBox(Text, Title))
```

```
If x < 100 Or x > 999 Then  
    Text = "Введено не трехзначное число " & x  
Else  
    s = x  
    c = s Mod 10  
    s = s \ 10  
    b = s Mod 10  
    a = s \ 10  
    s = a + b + c  
    Text = "Сумма цифр числа " & x & " равна " & s  
End If
```

```
Response = MsgBox(Text, , Title)
```

```
End Sub
```



- **Анализ.** Отметим, что использование переменной *s* как *рабочей ячейки* для хранения промежуточных результатов вычислений позволило сохранить заданное число для вывода результата.
- В программе выполняется проверка правильности входных данных. *Это хорошо делать всегда.*



Контрольные вопросы

1. В каком порядке выполняются операторы в программе?
2. Зачем используют операторы управления?
3. Какие значения может принимать переменная логического типа?
4. Какие значения может принимать логическое выражение?
5. Какой тип должны иметь переменные a и b в условном выражении $a \text{ and } b$?
6. Какой тип должны иметь переменные a и b в условном выражении $a > b$?
7. Какие действия выполняет условный оператор `If B Then S1`, если выражение B принимает значение `False`?
8. Какие действия выполняет условный оператор `if B Then S1 Else S2`, если выражение B принимает значение `False`?
9. Какие действия выполняет условный оператор `If B1 Then S1 ElseIf B2 Then S2`, если выражение $B1$ принимает значение `False`, а $B2$ – значение `True`?
10. Как использовать несколько условий в условном выражении?
11. Чему равно вещественное x после выполнения фрагмента программы $x = 2$:
`If Sqr(x)>4 then x = Sqr(x)`?
12. Какое значение принимает переменная d после выполнения оператора `If  $x - y < 0$  Then  $d = x$  Else  $d = y$` ?
13. Какое значение принимает переменная d после выполнения фрагмента программы $x = -2$: $y = 2$:
`If  $x < y$  Then  $d = 3$  ElseIf  $x < 0$  Then  $d = 1$  Else  $d = 2$` ?

14. Зачем используется оператор **if**?
15. Что такое алгоритм?
16. Какие типы алгоритмов Вы знаете?
17. Какой алгоритм называется линейным?
18. Какой алгоритм называется разветвляющимся?
19. Какие методы описания алгоритмов Вы знаете?
20. Зачем используются блок-схемы?
21. Зачем используется коннектор в блок-схемах?



Лабораторная работа

Цель работы. Научиться строить и выполнять в среде программирования VBA программы с операторами управления.

Задание. С помощью преподавателя выберите задачу и постройте для нее программу. Документ с программой запомните в личной папке.

Задачи

10. Найти наибольшее целое число из четырех заданных.
11. Найти наибольшую цифру заданного трехзначного числа.
12. Для заданного значения аргумента x вычислить функцию
$$y = \begin{cases} |x+1|, & x < 0, \\ 1-x^2, & x \geq 0. \end{cases}$$
4. Составить программу, которая определяет стоимость товара, если при покупке на сумму свыше 200 грн. дается скидка 2%, а при покупке на сумму свыше 500 грн. дается скидка 5%. вітається (доброго ранку, добрий день чи добрий вечір).
5. Составить программу, которая проверяет, является ли данный год високосным.

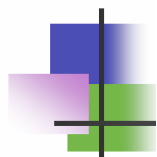
6. Составить программу для решения линейного уравнения $ax + b = 0$ с произвольными значениями a та b .
7. Составить программу для решения неравенства $ax + b > 0$, $a \neq 0$.
8. Составить программу для анализа и решения квадратного уравнения $ax^2 + bx + c = 0$, $a \neq 0$.
9. Составить программу для анализа и решения системы двух линейных уравнений с двома неизвестными.
10. Составить программу, которая определяет, можно ли построить треугольник с данными сторонами.



11. Циклы

Ключевые слова

цикл	cycle	周期	دورة
тело	body	身体	جسم
метка	label	标签	بطاقة
переход	transition	转型期	انتقال
рекуррентный	recurring	经常性	المتكررة
пересчитывать	recalculate	重新计算	حساب
делать	do	做	يفعل
петля	loop	环	أنشطة
предусловие	prerequisite	先决条件	شرط أساسي
постусловие	postcondition	邮政条件	في فترة ما بعد شرطاً
пока	while	一会儿	بينما
до тех пор	until	直到	حتى
для	for	为了	لأجل
шаг	step	一步	خطوة
счетчик	counter	计数器	مضادة
зависать	hover	悬停	تحويم
недостаток	lack of	缺乏	عدم وجود
возможность	possibility	可能性	إمكانية
определенный	certain	确定的	معين
неопределенный	undefined	未定义	غير معروف
повторять	repeat	重复	تكرر
верхний	top	顶端	أعلى
нижний	lower	低	أدنى
многократный	multiple	多	المتعددة
факториал	factorial	阶乘	المضروب



Циклические алгоритмы

- *Циклом* называется часть программы, которая выполняется несколько раз.
- Алгоритм, который имеет цикл, называется *циклическим*.
- Часть программы, которая повторяется, называется *телом цикла*.
- На Рис. 11.1 приведена логическая схема циклического алгоритма.
- Самый простой способ строить циклические алгоритмы состоит в использовании операторов управления.
- При этом кроме оператора **If** , используют оператор безусловного перехода

Go to метка

- *Метка* состоит из имени, которое строится по тем же правилам, что и имя переменной, но может начинаться с цифры.
- В программе метка ставится перед оператором и отделяется от него двоеточием(:).
- В языке Visual Basic метки не описываются.
- Работа оператора **go to** состоит в изменении порядка выполнения операторов в программе: следующим будет выполняться оператор, который стоит после метки – *помеченный оператор*.

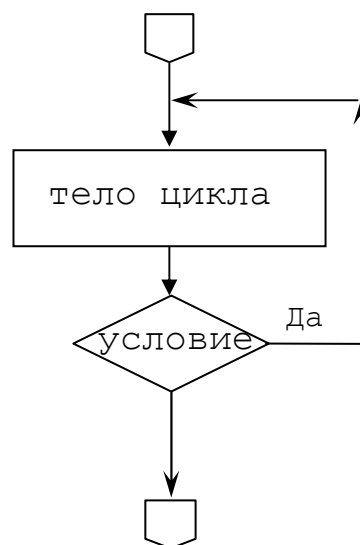


Рис. 11.1 Логическая схема циклического алгоритма



Программа 1

- **Задача.** Составить программу, которая вводит вещественные числа до тех пор, пока не будет введен пустой символ, и находит их среднее значение:

$$S_n = \frac{a_1 + a_2 + \dots + a_n}{n}.$$

- **Алгоритм.** Это сложная задача. Проблема состоит в том, что количество чисел заранее неизвестно.
- Чтобы найти метод решения задачи, преобразуем формулу для среднего:

$$nS_n = a_1 + a_2 + \dots + a_n,$$

откуда

$$nS_n = (n-1)S_{n-1} + a_n.$$

Это дает

$$S_n = \frac{(n-1)S_{n-1} + a_n}{n}, \quad S_1 = a_1 \quad (1)$$

- Формула, которая вычисляет значения некоторого элемента последовательности через предыдущие значения, называется *рекуррентной формулой*.
- Идея построения программы состоит в том, чтобы по полученной формуле пересчитывать значение S_n по S_{n-1} , используя для S_n и S_{n-1} одну и ту же рабочую переменную (Рис. 11.2).
- **Программа.** Описанный алгоритм реализует следующая программа.

```
Sub СреднееПотока()  
Dim a, S As Double  
Dim xTxt, Title, Text As String  
  
Title = "Найти среднее"  
Text = "Введите следующее целое число "  
Text = Next + "или нажмите ENTER"
```

$S = 0$

$n = 0$

once: $x\text{Txt} = \text{InputBox}(\text{Text}, \text{Title})$

If $x\text{Txt} \neq ""$ Then

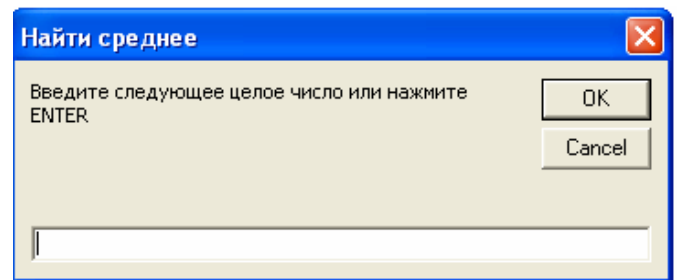
$a = \text{CInt}(x\text{Txt})$

$S = n * S + a$

$n = n + 1$

GoTo once

End If



$S = S / n$

$\text{Text} = \text{"Среднее введенных " \& n \& " чисел равно " \& S}$

$\text{Response} = \text{MsgBox}(\text{Text}, , \text{Title})$

End Sub

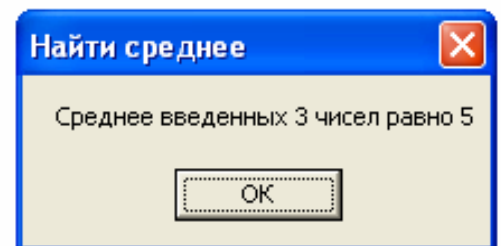
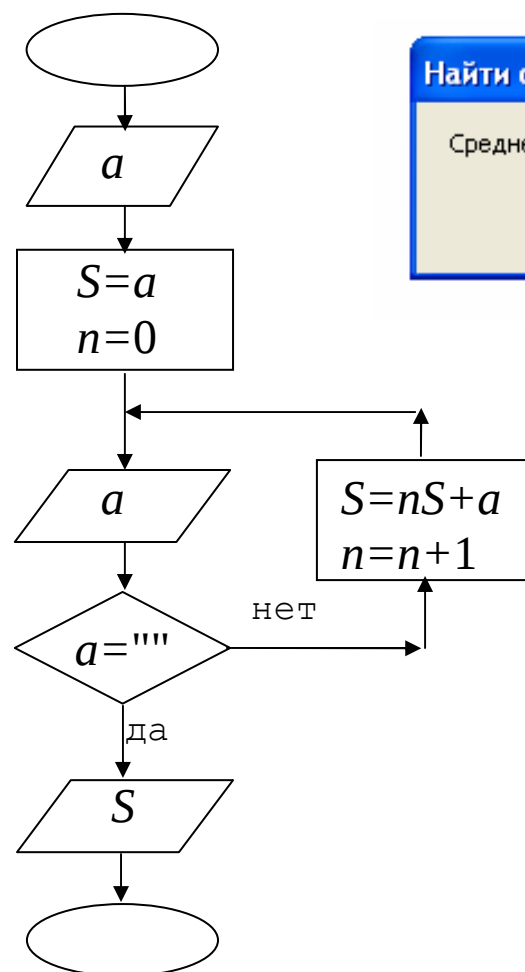


Рис. 11.2 Логическая схема программы
вычисления среднего.

- *Анализ.* В этой программе используется оператор **go to** с меткой **onсе**.
- Написано много книг, где не рекомендуются использовать оператор **go to**, потому что он запутывает алгоритм, а это может приводить к ошибкам в программе.
- Эти опасения безосновательны – плохой программист может наделать ошибок и без оператора **go to**. Аккуратное использование оператора **go to** позволяет просто писать хорошие программы.



Оператор цикла Do...Loop

- Этот оператор позволяет строить циклы без использования оператора безусловного перехода **go to**.
- Его применяют, когда число повторений тела цикла заранее неизвестно.
- В языке Visual Basic есть два типа оператора **Do** – с предусловием и с постусловием.
- *Цикл с предусловием* проверяют условие выхода из цикла перед выполнением тела цикла:

```

Do While условие
    тело цикла
Loop

```

Если **условие** правильное и принимает значение **True**, то операторы тела цикла выполняются, а если оно не правильно и принимает значение **False**, то операторы тела цикла не выполняются, а выполняется оператор программы, который следует за **Loop**.

- *Цикл с постусловием* проверяет условие выхода из цикла после выполнения тела цикла:

```

Do
    тела цикла
Loop Until условие

```

Если условие правильное и принимает значение True, то работа цикла заканчивается, а если оно неправильно и принимает значение False, то цикл повторяется.

- Иногда используются другие формы этих операторов: операторы

```
Do Until условие
    тело цикла
Loop
```

а также

```
Do
    тело цикла
Loop Until условие
```

завершают работу цикла, если условие принимает значение True.



Программа 2

- То же задание, что и предыдущая Программа 1, выполняет следующая программа, где используется основная форма оператора **Do** с предусловием.

```
Sub СреднееВЦикле()
Dim a, S As Double
Dim xTxt, Title, Text As String
```

```
Title = "Найти среднее"
```

```
Text = "Введите следующее число или нажмите ENTER"
```

```
S = 0
```

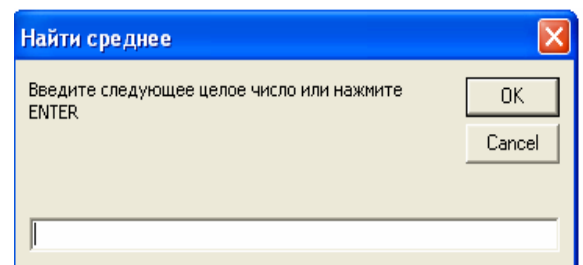
```
n = 0
```

```
xBtxt = InputBox(Text, Title)
```

```
Do While xBtxt <> ""
```

```
    a = CDBl(xBtxt)
```

```
    S = n * S + a
```



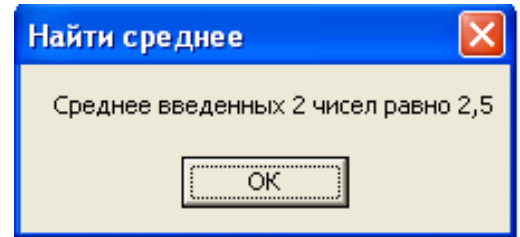
```

    xTxt = InputBox(Text, Title)
    n = n + 1
Loop

If n <> 0 Then
    S = S / n
    Text = "Среднее введенных " &n &" чисел равно " &S
    Response = MsgBox(Text, , Title)
End If

End Sub

```



- **Анализ.** При выводе результатов работы программы условный оператор используется потому, что программа не работает – останавливается с выводом сообщения об ошибке, если не вводится ни одно число. ***Программа должна работать всегда!***
- Недостатком этого алгоритма является необходимость использовать два оператора ввода. Программа 1 использует этот оператор один раз.



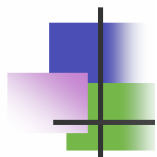
Оператор While ...Wend

- Это другая, более краткая запись основной формы оператора цикла с предусловием **Do While** условие тело цикла **Loop**:

```

        while условие
            тело цикла
        wend

```
- Оператор **Do** имеет больше возможностей. Однако оператор **while** проще, короче, и поэтому он часто используется.



Программа 3

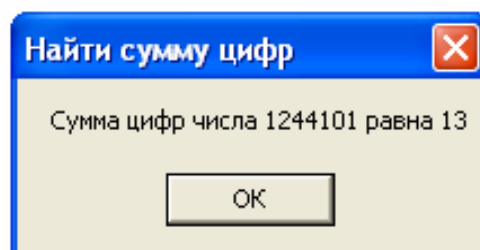
- **Задача.** Составить программу, которая вычисляет сумму цифр данного целого числа.
- **Алгоритм.** Это более общая задача, чем та, которую решает Программа 10.10. Для ее решения используем разработанный там алгоритм, который имеет циклическую структуру. Этот цикл заканчивается, когда в результате очередного целого деления получим нуль.
- **Программа.**

```
Sub СуммаЦифр()  
Dim a, s As Long  
Dim Title, Text As String  
  
Title = "Найти сумму цифр"  
Text = "Введите любое целое число"  
x = CLng(InputBox(Text, Title))
```

```
s = 0  
a = x  
While a <> 0  
s = s + a Mod 10  
a = a \ 10  
Wend
```

```
Text = "Сумма цифр числа " & x & " равна " & s  
Response = MsgBox(Text, , Title)
```

```
End Sub
```



- **Анализ.** Эта общая программа короче и изящнее, чем частная Программа 10.10.
- В программе используются длинные целые переменные для того, чтобы увеличить диапазон чисел, для которых можно использовать программу.



Оператор For

- Оператор цикла со счетчиком **For** используется, когда известно количество повторений тела цикла. Это самый популярный оператор цикла:

```
For счетчик = начало To шаг Step конец  
    тело цикла  
Next
```

где

счетчик — переменная целого типа,

начало, конец, шаг — целые выражения:

- начало — начальное значение переменной счетчик,
- конец — конечное значение переменной счетчик,
- шаг — приращения переменной счетчик при каждом новом повторении тела цикла.

- Оператор **For** работает так:

- 1) Переменной счетчик присваивается значение начало;
- 2) *если* счетчик $\leq$ конец, то выполняется тело цикла, а *иначе* тело цикла не выполняется и оператор заканчивает свою работу;
- 3) Значение счетчика увеличивается на шаг:

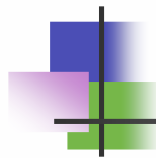
счетчик = счетчик + шаг

- 4) Переходим к пункту 8.

- Если шаг =1, то значение шага не задают и используют оператор

```
For счетчик = начало To конец  
    тело цикла  
Next
```

- После выполнения оператора **For** значение переменной счетчик не определено.
- В теле цикла в операторе со счетчиком **For** нельзя изменять его параметры счетчик, начало, конец и шаг.



Программа 4

- **Задача.** Составить программу для вычисления факториала $n! = 1 \cdot 2 \cdot \dots \cdot n$; $0! = 1$.
- **Алгоритм.** Поскольку величина n известна, для решения этой задачи удобно использовать оператор For:

```
Sub Fact()
    Dim i, n As Integer
    Dim f As Long
    Dim xTxt, Title, Text As String

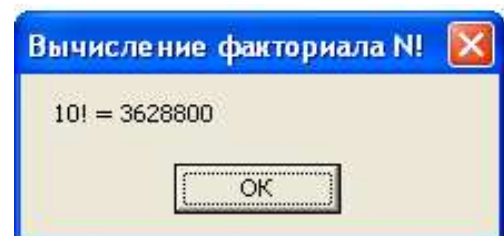
    Title = "Вычисление факториала N!"
    Text = "Введите значение N"
    n = CInt(InputBox(Text, Title))

    f = 1

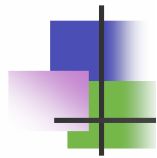
    For i = 1 To n
        f = f * i
    Next

    Text = n & "! = " & f

    Response = MsgBox(Text, , Title)
End Sub
```



- **Анализ.** С ростом n величина $n!$ очень быстро растет. Поэтому для результата в программе используется переменная типа Long.



Контрольные вопросы

1. Что такое цикл?
2. Что такое оператор тела цикла?
3. Какие действия выполняет оператор **GoTo**?
4. Что такое метка?
5. Как ввести метку в программе?
6. Почему не рекомендуют использовать оператор **GoTo**?
7. Что такое оператор цикла с предусловием?
8. Что такое оператор цикла с постусловием?
9. Чем отличаются операторы **Do** с предусловием и постусловием?
10. Чем отличается использование опций **while** и **Until** в операторе **Do**?
11. Могут ли одновременно использоваться опции **while** и **Until** в операторе **Do**?
12. Что такое оператор цикла со счетчиком?
13. Данные каких типов можно использовать в качестве счетчика в операторе цикла со счетчиком?
14. Что такое шаг в операторе цикла со счетчиком?
15. Какой шаг выбирает оператор цикла со счетчиком, если шаг не указан?
16. Можно ли использовать в качестве счетчика в операторе со счетчиком вещественные переменные?
17. Сколько раз выполнится оператор в теле цикла `For i=1 to 0 : x=i` ?
18. Чему равно значение параметра цикла после выполнения оператора `For i=1 to 10 : x=i` ?
19. Как изменить значение счетчика цикла в теле цикла?
22. Как изменить верхнее значение параметра цикла в теле цикла?



Лабораторная работа

Цель работы. Научиться строить и выполнять в среде программирования VBA программы с циклами.

Задание. С помощью преподавателя выберите задачу и постройте для нее программу. Документ с программой запомните в личной папке.

Задачи

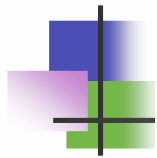
1. Вычислить $10k! = 10 \cdot 1 \cdot 2 \cdot 3 \cdot \dots \cdot k$ с помощью цикла со счетчиком.
2. Вычислить $(2k)!! = 2 \cdot 4 \cdot \dots \cdot 2k$ с помощью цикла с предусловием.
3. Вычислить $(2k+1)!! = 1 \cdot 3 \cdot 5 \cdot \dots \cdot (2k+1)$ с помощью цикла с постусловием.
4. Вывести все простые множители данного целого числа.
5. Вычислить сумму цифр данного целого числа.
6. Найти наибольшую цифру данного целого числа.
7. Вычислить сумму квадратов натуральных чисел от 1 до n .
8. Подсчитать количество цифр данного целого числа.
9. Найти НОД (наибольший общий делитель) двух целых чисел.
10. Найти НОК (наименьшее общее кратное) двух целых чисел.
11. Найти наибольшее целое число из четырех заданных.



12. Массивы

Ключевые слова

массив	array	阵列	مجموعة
описание	description	说明	وصف
размерность	dimension	维	بعد
скобки	brackets	括号	بين قوسين
индекс	index	指数	مؤشر
граница	limit	限制	لحد
элемент	element	内容	عنصر
количество	quantity	数量	الكمية
одномерный	unidimensional	单向	الأحادي
многомерный	multivariate	多元	متعددة
вектор	vector	载体	قوة موجهة
матрица	matrix	矩阵	مصفوفة
квадратный	squared	平方	مربع
прямоугольный	rectangular	矩形	متعامد
случайный	random	随机	عشوائية
генератор	generator	发电机	مولد
выбрать	select	选择	مختارة
переставить	shift	转移	وردية
заменить	replace	取代	يحل محل
наибольший	greatest	最大	أعظم
наименьший	least	最少的	الأقل
сортировка	sorting	排序	فرز
пузырек	bubble	泡沫	فقاعة
всплывать	rise	增加	يرتفع



Типы массивов

- *Массив* – это упорядоченный набор переменных одного типа, которые имеют одинаковое имя и различаются номерами.
- Номер массива называется *индексом*.
- В языке Visual Basic индекс может принимать любые целые значения.
- В языке Visual Basic индексы записываются в круглых скобках после имени массива. Например, запись $a(3)$ обозначает третий элемент массива a .
- Массив можно описать с помощью оператора

Dim переменная(числоЭлементов) **As** тип

где числоЭлементов – целое число, на 1 меньшее числа элементом массива. Например, оператор

Dim a(10) As Integer

описывает массив из 11 элементов целого типа:

$a(0), a(1), \dots, a(10)$.

- В языке Visual Basic можно использовать массив, для которого верхняя и нижняя границы индексов есть любые целые числа типа Long. Такой массив описывается оператором

Dim переменная(начало **To** конец) **As** тип

где начало и конец – любые числовые константы типа Long. Они задают начальное и конечное значение индекса.

- Массив может иметь несколько индексов.
- Массив с одним индексом называется *одномерным*, с двумя индексами – *двумерным* и т.д.
- *Многомерные массивы* описываются и используются так же, как и одномерные. Например, оператор

Dim a(4, 9) As Integer

описывает двумерный массив a с 5×10 элементами.



Программа 1

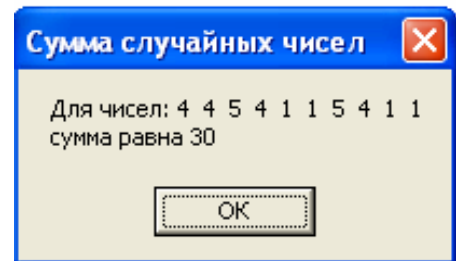
- **Задача.** Составить программу, которая строит массив из 10 случайных целых чисел, не больших заданного целого числа A , и находит их сумму.
- **Алгоритм.** В математике случайным числом называется значение, которое заранее предсказать невозможно. Например, результат очередного бросания игральной кости.
- В языке Visual Basic для построения случайных чисел используется датчик случайных чисел – функция $Rnd(x)$, результатом выполнения которой будет случайное вещественное число, которое больше или равно 0 и меньше 1. Целый аргумент этой функции определяет режим ее работы:
 - Если $x < 0$, то функция выдает "псевдослучайные" значения, которые повторяются при каждом новом запуске программы;
 - Если $x > 0$ или аргумент не задается, то функция при каждом новом обращении выдает новое случайное число;
 - Если $x = 0$, то функция выдает одно и то же случайное значение.
- Сумма построенных чисел накапливается одновременно с вычисление каждого нового значения.
- **Программа.**

```
Sub SummaRnd()  
Dim x(9) As Integer  
Dim s As Integer  
Dim i, A As Integer  
Dim Title, Text As String
```

```
Title = "Сумма случайных чисел"
Text = "Укажите максимальное число"
A = CInt(InputBox(Text, Title))
```

```
Text = "Для чисел: "
s = 0
```

```
For i = 0 To 9
x(i) = Int(A * Rnd(0) + 1)
s = s + x(i)
Text = Text & x(i) & "  "
Next
```



```
Text = Text & Chr(13) & Chr(10) & "сумма равна " & s
Response = MsgBox(Text, , Title)
End Sub
```

- **Анализ.** Обратите внимание, как описывается и используется массив из 10 чисел: индексы меняются от 0 до 9.
- Для массивов важной является проблема вывода. В программе значения элементов массива *добавляются в строку вывода* с помощью операции *конкатенации* (&) – к строке приписывается число.



Массивы в MS Excel

- В приложении MS Excel электронная таблица представляет собой двумерный массив.
- Адресация клеток электронной таблицы методом R1C1 – это другая форма обращения к массиву электронной таблицы.
- Если среда программирования VBA вызывается с некоторого листа приложения MS Excel, то значения в клетках электронной таблицы этого листа доступны в VBA как элементы двумерного массива `Cells(i, j)`, где *i* и *j* – номер строки и столбца электронной таблицы.
- Так что `Cells(i, j)` – это содержимое ячейки `RiCj`.
- Следующая программа показывает, как это делать.



Программа 2

```
Sub WhatIs()  
Dim a As Integer  
Dim Text As String  
  
a = Cells(3, 2)  
Cells(4, 4) = a + Cells(1, 2)  
  
Text = "Это " & a  
MsgBox Text  
  
End Sub
```

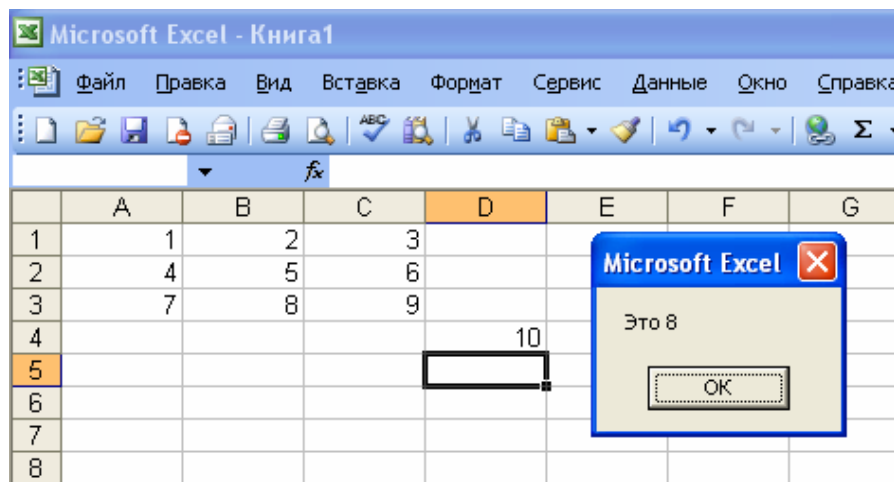
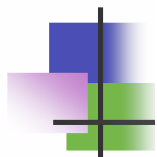


Рис. 12.1 Результат работы ознакомительной Программы 2

- **Анализ.** Эта программа показывает, как получать данные с электронной таблицы и как записывать данные в ячейки электронной таблицы.
- При работе с массивами ввод и вывод данных может представлять проблему, когда их много. Один из выходов из такой ситуации – использовать для этих целей табличный процессор.
- **При использовании VBA совместно с MS Excel надо следить за типами данных в электронной таблице.** Иначе могут быть проблемы!



Программа 3

- **Задача.** Составить программу, которая заполняет ячейки электронной таблицы так, как показано на Рис. 12.2, если адрес левой верхней ячейки места заполнения задается пользователем.

The screenshot shows the Microsoft Excel interface with a table starting from cell B3. The numbers are filled diagonally from the selected cell B3, which contains the value 1. The sequence of numbers is 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30.

	A	B	C	D	E	F	G	H
1								
2								
3		1	2	4	7	11	16	
4		3	5	8	12	17	21	
5		6	9	13	18	22	25	
6		10	14	19	23	26	28	
7		15	20	24	27	29	30	
8								

Рис. 12.2 Результат работы Программы 3

- **Алгоритм.** Конечно, можно написать 30 операторов присваивания. Но это громоздко. А что делать, если таблица будет больше?
- Заметим закономерность – последовательные числа записываются по диагонали – слева направо, сверху вниз. Проблема в том, чтобы так обходить массив.
- Такой обход можно реализовать, если выполнить двойной цикл: первый по диагоналям (i) от 1 до $5+6=11$ а второй – вдоль диагонали (j) от 1 до i по схеме:
$$a(j, i - j) = k$$

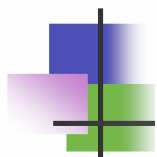
При увеличении j при фиксированном i будем иметь перемещение по массиву в нужном направлении. Этот оператор присваивания надо выполнять, только на допустимом шаге – если значение индексов не выходят за допустимые пределы.

- При каждом допустимом шаге значение k надо увеличивать на 1.

■ **Программа.**

```
Sub Numeration()  
Dim rt, ct As Integer  
Dim a(20, 20) As Integer  
  
Dim Title, Text As String  
Dim i, j, k As Integer  
  
Title = "Укажите место таблицы"  
Text = "Ведите номер строки "  
rt = CInt(InputBox(Text, Title))  
Text = "Ведите номер столбца "  
ct = CInt(InputBox(Text, Title))  
  
k = 0  
For i = 1 To 11      ' номер диагонали  
For j = 1 To i      ' позиция в диагонали  
If (j - 1 < 5) And (i - j < 6) Then  
    k = k + 1  
    Cells(rt + j - 1, ct + i - j + 1) = k  
End If  
Next  
Next  
  
End Sub
```

- **Анализ.** Для правильной работы программы пришлось корректировать идею, описанную в алгоритме – в программе вместо j используется $j-1$.
- Чтобы правильно сделать такую коррекцию, надо было запустить программу, посмотреть ее результат, понять его и сделать нужные изменения.
- Таким же способом корректировались и ограничения на индексы в условном операторе.



Динамические массивы

- Массивы, которые мы использовали, имеют фиксированное количество элементов, которое не может изменяться в программе. Такие массивы называются *статическими*.
- Иногда количество элементов массива заранее неизвестно, и тогда приходится определять длину массива "с запасом" или каждый раз изменять программу.
- Язык Visual Basic позволяет использовать *динамические* массивы, длину которых можно определять при выполнении программы.
- При описании динамического массива его длина не указывается:

Dim переменная() **As** тип

- Когда размерность массива становится известной, она описывается в нужном месте программы оператором:

ReDim переменная(числоЭлементов)

- При повторном выполнении оператора **ReDim** все значения элементов массива будут потеряны.
- Чтобы сохранить значения переменных при выполнении оператора **ReDim**, надо использовать в нем опцию **Preserve**:

ReDim Preserve переменная(числоЭлементов)

- Оператор **ReDim** можно использовать несколько раз. При этом можно менять даже количество измерений – индексов. Например,

Dim x() **As** Integer

. . .
ReDim x(5)

. . .
ReDim x(3,3)

- И в этом можно использовать опцию **Preserve**.



Программа 4

- **Задача.** Составить программу, которая записывает в первый столбец электронной таблицы заданное количество случайных целых чисел из интервала [10; 30], а затем выводит их во второй столбец в порядке возрастания.
- **Алгоритм.** Алгоритм построения случайных чисел возьмем из Программы 1. Для того чтобы получить случайные числа из заданного интервала, используем преобразование
$$x(i) = 10 + \text{Int}(20 * \text{Rnd}(0) + 1)$$
- Для упорядочивания чисел по возрастанию используем метод пузырька: сравнивая по всему массиву соседние пары элементов, меняем их местами, если предыдущий элемент больше последующего (Рис. 12.3).
- После первого прохода Рис. 12.3 последний элемент будет наибольшим.

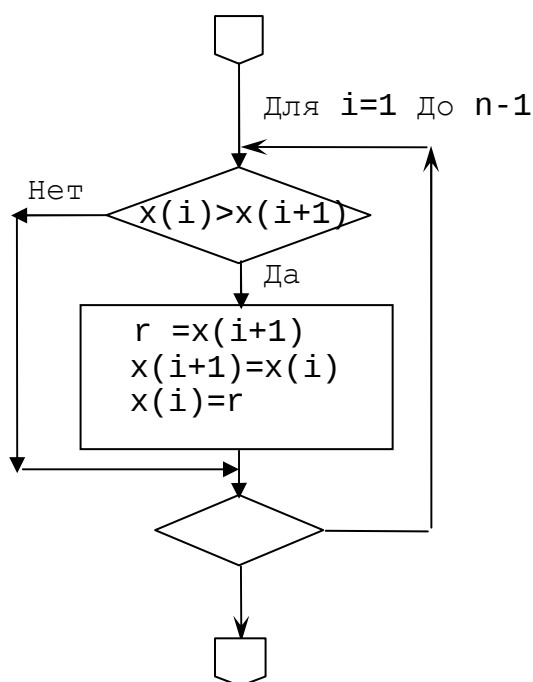


Рис. 12.3 Логическая схема
первого прохода метода пузырька

- Далее повторяем описанные сравнения для оставшихся $n-k$ первых элементов массива, где k – номер прохода. Таких проходов надо выполнить $n-1$.
- Этот простой алгоритм часто используют программисты.

■ **Программа.**

```
Sub Bubble()
```

```
Dim x() As Integer ' динамический массив
```

```
Dim i, k, n, r As Integer
```

```
Dim Title, Text As String
```

```
A = 10: B = 20
```

```
Title = "Метод пузырька"
```

```
Text = "Укажите число элементов"
```

```
n = CInt(InputBox(Text, Title))
```

```
ReDim x(n) 'указание длины динамического массива
```

```
' построение массива случайных чисел
```

```
For i = 1 To n
```

```
x(i) = A + Int(B * Rnd(1) + 1)
```

```
Cells(i, 1) = x(i)
```

```
Next
```

```
' метод пузырька
```

```
For k = 1 To n - 1
```

```
For i = 1 To n - k
```

```
If x(i) > x(i + 1) Then
```

```
r = x(i + 1)
```

```
x(i + 1) = x(i)
```

```
x(i) = r
```

```
End If
```

```
Next : Next
```

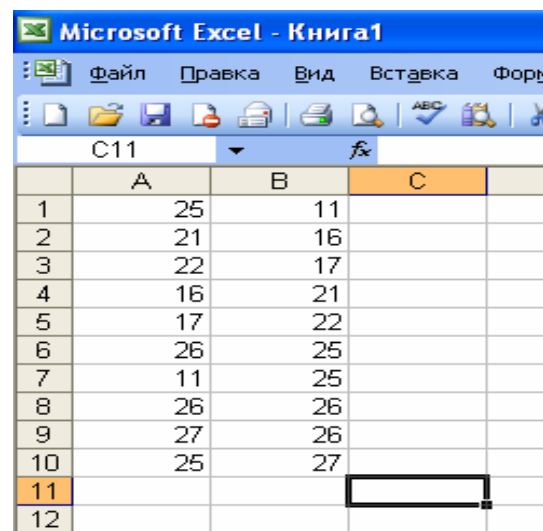
```
' вывод массива
```

```
For i = 1 To n
```

```
Cells(i, 2) = (i)
```

```
Next
```

```
End Sub
```



	A	B	C
1	25	11	
2	21	16	
3	22	17	
4	16	21	
5	17	22	
6	26	25	
7	11	25	
8	26	26	
9	27	26	
10	25	27	
11			
12			

Рис. 12.4 Результат работы программы

- **Анализ.** Здесь используется вспомогательный динамический массив *x*.
- Использование других массивов, кроме ячеек электронной таблицы, в этой программе не обязательно. Но они делают программу проще.
- Обратите внимание, как при обмене значениями двух элементов используется вспомогательная переменная *g*. Без нее трудно сделать обмен.



Контрольные вопросы

1. Что такое массив?
2. Что такое индекс?
3. Какие значения могут принимать индексы в ?
4. Сколько элементов имеет массив с описанием `Dim x(5)?`
5. Сколько элементов имеет массив с описанием `Dim a(5,5)?`
6. Сколько места в памяти занимает массив целых чисел с описанием `Dim x(5) As Integer?`
7. Сколько места в памяти занимает массив чисел с описанием `Dim x(5) As Double?`
8. Какой массив называется многомерным?
9. Что такое статический массив?
10. Что такое динамический массив?
11. Что такое случайные числа?
12. Какие значения выдает функция `Rnd(1)?`
13. Какие значения выдает функция `Rnd(-1)?`
14. Когда можно использовать в VBA ячейки электронной таблицы MS Excel?
15. Как описывать в VBA для MS Excel ячейки электронной таблицы?
16. Почему метод сортировки назван "методом пузырька"?



Лабораторная работа

Цель работы. Научиться строить и выполнять в среде программирования VBA программы для работы с массивами.

Задание. С помощью преподавателя выберите задачу и постройте для нее программу. Документ с программой запомните в личной папке.

Задачи

1. Составить программу, которая формирует и выводит прямоугольную матрицу размера $N \times M$ из случайных целых чисел в диапазоне $[L, M]$.
2. Составить программу, которая формирует вектор из N случайных вещественных чисел в диапазоне $[A, B]$ и находит среднее значение его элементов.
3. Составить программу, которая формирует вектор из N случайных целых чисел в диапазоне $[L, M]$ и находит его наибольший и наименьший элементы.
4. Сформировать массив из N случайных целых чисел в диапазоне $[L, M]$ и сжать его, выбросив все отрицательные числа.
5. Изменить порядок значений элементов массива на обратный.
6. Определить, является ли заданный массив упорядоченным.
7. Составить программу, которая заполняет массив из N случайных целых чисел в диапазоне $[L, M]$ и упорядочивает его по убыванию.
8. Составить программу, которая формирует квадратную матрицу размера $N \times N$ из случайных целых чисел в диапазоне $[L, M]$ и находит столбец с наибольшим элементом.

9. Составить программу, которая формирует и выводит прямоугольную матрицу

1	3	6	9	12
2	5	8	11	14
4	7	10	13	15

10. Составить программу, которая формирует прямоугольную матрицу размера $N \times M$ из случайных целых чисел в диапазоне $[L, M]$ и упорядочивает ее строки по возрастанию наибольшего элемента.
11. Выяснить, есть ли одинаковые элементы в заданном одномерном массиве.
12. Выяснить, есть ли одинаковые элементы в заданной прямоугольной матрице.
13. Составить программу, которая формирует прямоугольную матрицу размера $N \times M$ из случайных вещественных чисел в диапазоне $[A, B]$ и переставляет местами столбцы с наименьшим и наибольшим элементами.



13. Процедуры и функции

Ключевые слова

подпрограмма	subprogramme	次级	البرنامج الفرعي
процедура	procedure	程序	إجراء
функция	function	功能	وظيفة
параметр	parameter	参数	بارامتر
аргумент	argument	论点	حجة
вызов	call	呼叫	مكالمة
формальный	formal	正式	رسمي
фактический	actual	实际	فعلي
передавать	transfer	转让	نقل
ссылка	link	链接	رابط
исходный	source	来源	مصدر
видимость	visibility	能见度	وضوح
общедоступный	public	市民	العامة
частный	private	私营	الخاصة
четко	clearly	显然	بوضوح
запрещать	forbid	禁止	لا سمح
матрица	matrix	矩阵	مصفوفة
заполнить	fill	充满	يملأ
транспонировать	transpose	移调	ترجم
пустой	empty	空的	فارغ
сверху вниз	down	向下的	أسفل
структурный	structural	结构	الهيكلي
по умолчанию	default	默认	افتراضي



Типы процедур

- Процедуры – это подпрограммы. Подпрограмма – это часть программы, которая сама является программой.
- Использование процедур позволяет не повторять наборы операторов в программе. Для этого операторы, которые выполняются много раз, но не являются циклами, помещают в процедуры.
- Процедуры бывают двух типов:
 - Процедура-подпрограмма, которую обычно называют процедурой;
 - Процедура-функция, которую обычно называют функцией.
- Программа в VBA состоит из набора процедур.
- Если в документе MS Office, из которого вызвана среда VBA, есть несколько процедур, то запуск одной из них можно выполнить, если сделать щелчок левой кнопкой мыши на тексте нужной процедуры и нажать **F5**
- Выбор нужной процедуры можно сделать также в списке процедур (Рис. 13.1).

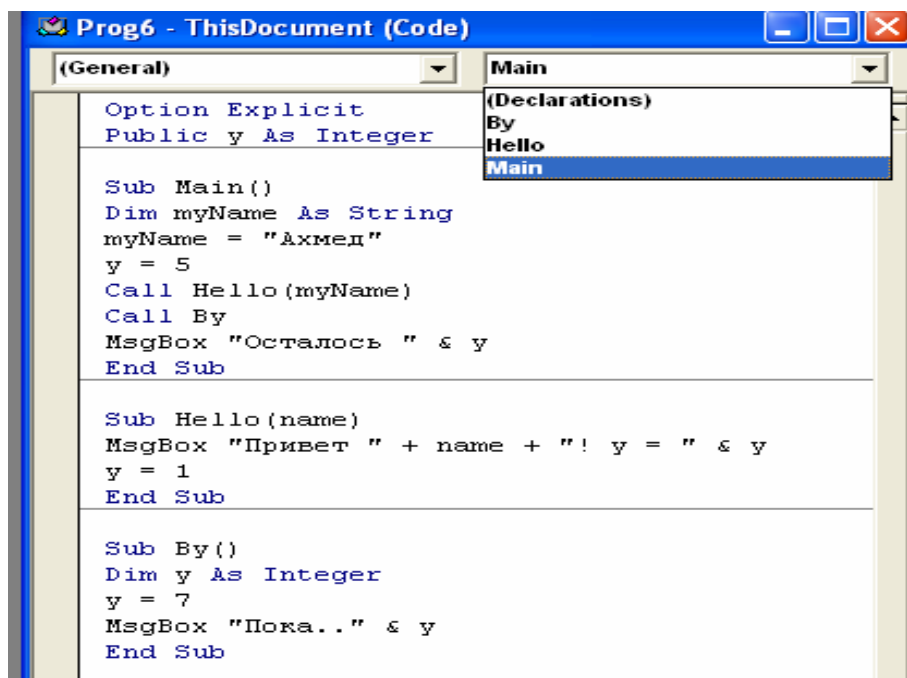
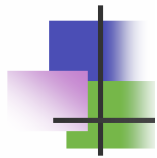


Рис. 13.1 Список процедур в окне кода VBA



Описание и вызов процедуры

- Описание процедуры в VBA имеет вид (Рис. 13.1):

```
Sub имя(списокПараметров)  
    тело процедуры  
Sub End
```

- Здесь списокПараметров – это список имен переменных с описаниями, разделенных запятыми. Этот список может иметь только одну переменную или быть пустым.
- Переменные в списке параметров называются *формальными параметрами*.
- Формальные параметры отдельно не описываются. Их тип указывают в описании процедуры. Если тип формального параметра не указан в описании процедуры, то считается, что этот тип Variant. Например,

```
Sub student(age As Byte, name As String)
```
- В теле процедуры формальные параметры используются как обычные переменные с известными значениями.
- Параметры процедуры задается при вызове процедуры. Вызов процедуры в программе выполняется оператором

```
Call имя(списокФактическихПараметров)
```
- Параметры процедуры могут быть:
 - *Исходными данными*, которые процедура получает и использует для вычислений. Такие параметры являются *параметрами-значениями*. Перед такими параметрами в описании процедуры ставится слово **byVal**;
 - *Результатами*, которые определяются в процессе выполнения процедуры. Это – *параметры-переменные*.
- *Фактическими параметрами для исходных данных* являются выражения или переменные, которым в вызывающей программе присвоены некоторые значения.
- *Фактическими параметрами для результатов* должны быть переменные, которые используются в программе.



Видимость переменных

- При использовании процедур важно знать, какие переменные можно использовать в теле процедуры.
- Правила использования переменных зависят от того, как они введены в программе.
- В языке Visual Basic переменные определяют:
 - С помощью оператора **Dim** внутри процедуры. Такие переменные существуют только внутри процедуры, где они объявлены. Это *локальные переменные*. При выходе из процедуры они исчезают.
 - С помощью оператора **Public** в разделе описаний перед всеми процедурами описываются *глобальные переменные*:

Public переменная **As** тип

- Глобальные переменные можно использовать во всех процедурах без описания. При выходе из процедуры они сохраняют свои значения.
- Глобальные переменные можно описывать в разделе описаний с помощью оператора **Dim**. Эта возможность перешла в Visual Basic со старых версий этого языка.
- Если внутри процедуры с помощью оператора **Dim** описана переменная с тем же именем, что и глобальная переменная, то тогда такая глобальная переменная недоступна в теле этой процедуры.
- Переменные, которые используются в процедурах без описания, являются локальными.
- Использование переменных без описания может привести к ошибкам, которые трудно заметить.
- В языке Visual Basic можно запретить использовать переменные без описания. Для этого в разделе описаний записывают оператор

Option Explicit



Программа 1

- Рассмотрим программу, приведенную на Рис. 13.1. Результаты ее работы приведены на Рис. 13.2.



Рис. 13.2 Результаты работы программы main на Рис. 13.1:

- а) Вывод процедуры Hello; б) Вывод процедуры Ву;
в) Вывод основной процедуры main.

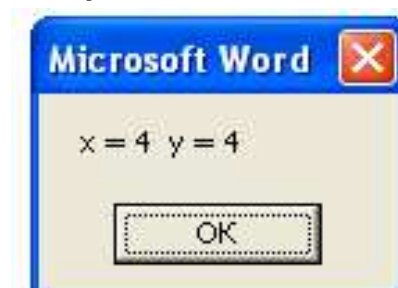
- В этой программе используется глобальная переменная `y`. Однако в процедуре `Ву` есть локальная переменная `y`. Значение этой локальной переменной используется в выводе внутри процедуры. Операции с такой локальной переменной не влияют на значение одноименной глобальной переменной.



Программа 2

- Рассмотрим следующую программу:

```
Sub main()  
Dim x, y As Integer  
x = 3: y = 4  
Call Increase(x, y)  
MsgBox "x = " & x & " y = " & y  
End Sub  
  
Sub Increase(a, ByVal b)  
a = a + 1  
b = b + 1  
End Sub
```



- В процедуре `Increase` этой программы первый параметр *передается по ссылке* и является параметром-переменной. Его значение можно менять в процедуре.
- Вторым параметр этой процедуры *передается по значению* и является параметром-значением. В результате обращения к процедуре он не изменился.



Массивы в процедурах

- Трансляторы используют разные алгоритмы для передачи в процедуру параметров-ссылок и параметров-значений.
- При вызове процедуры программа передает ей тип параметра-переменной и адрес в памяти, где он записан. Все действия над такими параметрами процедура выполняет на месте в памяти, где они записаны.
- Для каждого параметра-переменной транслятор выделяет место в памяти. Выполнение процедуры начинается с того, что фактическое значение всех параметров-значений вычисляются и записывается в выделенную память. После завершения работы процедуры они не сохраняются.
- Если тип формального параметра не указан, то в языке Visual Basic считается, что это – параметр-переменная.
- Массивы в программах могут быть большими. Поэтому обычно они описываются как параметры-переменные. Это предотвращает ненужное использование памяти компьютера для копии массива.
- В описании процедуры массив можно описывать как параметр-переменную типа `Variant`. Например,
`Sub Multiply(A As Variant, B As Variant, C As Variant)`
- При вызове такой процедуры фактический параметр-массив записывается с пустыми скобками:

`Call Multiply(x(), y(), z())`



Программа 3

- **Теория.** Матрицей размера $n \times m$ называется прямоугольная таблица, которая имеет n строк и m столбцов:

$$A = (a_{ik})_{n,m} = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ a_{21} & a_{22} & \dots & a_{2m} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nm} \end{pmatrix}$$

- Числа $a_{i,j}$ называются элементами матрицы, а i и j – индексами.
- Матрицы можно умножать, если их размеры согласованы – количество строк первой матрицы равно количеству столбцов второй.
- Произведением матриц размеров $n \times p$ и $p \times m$ называется матрица размера $n \times m$, каждый элемент $a_{i,j}$ которой равен сумме произведений p элементов i -ой строки первой матрицы на элементы j -го столбца второй.
- **Задача.** Составить процедуру для умножения матриц. Использовать ее для вычисления произведения двух вещественных матриц со случайными значениями элементов из интервала $[-10; 10]$.
- **Алгоритм.** Для решения задачи создадим три процедуры
 - Процедура формирования матрицы: Sub FormMatr(ByVal n As Integer, ByVal m As Integer, X As Variant)
 - Процедура вывода матрицы: Sub PrintMatr(Title As String, ByVal n As Integer, ByVal m As Integer, X As Variant)
 - Процедура умножения матриц: Sub MultiMatr(ByVal n As Integer, ByVal p As Integer, ByVal m As Integer, X As Variant, Y As Variant, Z As Variant)
- Следующая программа решает эту задачу:

Option Base 1

```
Sub Multiply()  
Dim A(), B(), C() As Integer  
Dim s, mx As Integer  
Dim n, m, p, i, j As Integer  
Dim Title, Text As String  
  
Title = "Размеры матриц"  
Text = "Укажите число строк первой матрицы"  
n = CInt(InputBox(Text, Title))  
Text = "Укажите число столбцов первой матрицы"  
p = CInt(InputBox(Text, Title))  
Text = "Укажите число столбцов второй матрицы"  
m = CInt(InputBox(Text, Title))  
  
ReDim A(n, p)  
Call FormMatr(n, p, A())  
Call PrintMatr("Первая матрица", n, p, A())  
  
ReDim B(p, m)  
Call FormMatr(p, m, B())  
Call PrintMatr("Вторая матрица", p, m, B())  
  
ReDim C(n, m)  
Call MultiMatr(n, p, m, A(), B(), C())  
Call PrintMatr("Произведение матриц", n, m, C())  
  
End Sub
```

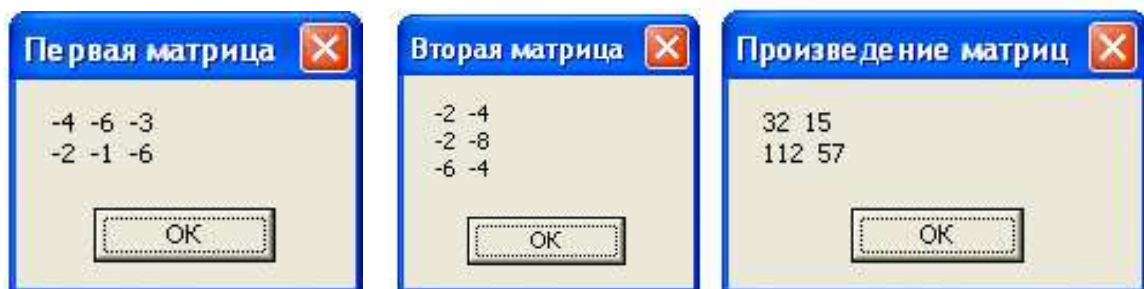


Рис. 13.3 Результаты работы программы Multiply

- Эту программу можно запустить и выполнить, если даже каждая из процедур не будет иметь операторов. При этом все матрицы будут не заполнены, но мы проверим взаимодействие программ и правильность обращения к процедурам.

- Поставленную задачу выполняют процедуры:

```
Sub FormMatr(ByVal n As Integer, ByVal m As Integer,  
X_ As Variant)  
Dim i, j As Integer
```

```
MinVal = -10: Spr = 10  
For i = 1 To n  
For j = 1 To m  
X(i, j) = Int(Spr * Rnd(1) + 1) + MinVal  
Next: Next  
End Sub
```

```
Sub MultiMatr(ByVal n As Integer, ByVal p As Integer,_  
ByVal m As Integer, X As Variant, Y As Variant, Z As_  
Variant)  
Dim i, j As Integer  
Dim s As Integer
```

```
For i = 1 To n  
For j = 1 To m  
s = 0  
For k = 1 To p  
s = s + X(i, k) * Y(k, j)  
Next  
Z(i, j) = s  
Next: Next  
End Sub
```

```
Sub PrintMatr(Title As String, ByVal n As Integer,_  
ByVal m As Integer, X As Variant)  
Dim i, j As Integer  
Dim Text As String
```

```
Text = ""  
For i = 1 To n  
For j = 1 To m  
Text = Text & X(i, j) & "  "  
Next  
Text = Text & Chr(13) & Chr(10)  
Next  
Response = MsgBox(Text, , Title)  
End Sub
```

- **Анализ.** Запуск программы с "пустыми" процедурами для того, чтобы проверить ее интерфейс (диалог с поль-

зователем), широко используется профессиональными программистами.

- Потом будет создаваться тело каждой процедуры. Сразу после создания очередной процедуры ее можно будет проверить таким же способом – с пустыми другими процедурами, которые еще не созданы.
- Такой прием называется программированием *сверху вниз*.
- Создание программ таким способом называется *системным программированием*.



Функции

- Результатом работы функции является некоторое значение – число, текст или логическая величина.
- Это значение имеет тип, и поэтому функция, как и переменная, имеет тип.
- Описание функции в языке Visual Basic имеет вид:
Function имя(списокПараметров)**As** тип
тело функции
Sub End
- Тип функции можно не указывать в описании, и тогда эта функция будет иметь тип `Variant`.
- В теле функции обязательно должен быть оператор, присваивающий значение переменной, имя которой совпадает с именем функции. Такая переменная не описывается в теле функции.
- Функция вызывается указанием ее имени в правой части оператора присваивания
переменная = имяФункции(списокПараметров)
- Функцию можно использовать также как операнд некоторого выражения. Например, для функции `MaxElem`:
`Spread = MaxElem(a()) - MinElem(a())`



Программа 4

- **Задача.** Составить процедуру, которая упорядочивает столбцы матрицы размера $n \times m$ по возрастанию суммы их элементов. Использовать эту процедуру для упорядочивания матрицы со случайными значениями.
- **Алгоритм.** Для решения задачи воспользуемся процедурами, которые были созданы раньше:
 - Процедура формирования матрицы: Sub FormMatr(ByVal n As Integer, ByVal m As Integer, X As Variant)
 - Процедура вывода матрицы: Sub PrintMatr(Title As String, ByVal n As Integer, ByVal m As Integer, X As Variant)
- Создадим новые процедуры и функции:
 - Функция для нахождения суммы элементов заданного столбца матрицы: Function SumElem(ByVal n As Integer, ByVal col As Integer, Y() As Integer) As Integer;
 - Процедура перестановки двух заданных столбцов матрицы: Sub ChangeCol(ByVal n As Integer, ByVal col1 As Integer, ByVal col2 As Integer, Y As Variant);
 - Процедура сортировки матрицы по возрастанию суммы элементов ее столбцов: Function: SortMatr(ByVal n As Integer, ByVal m As Integer, Y() As Integer).
- Эти процедуры имеет код:

```
Function SumElem(ByVal n As Integer, ByVal col As Integer, Y() As Integer) As Integer
Dim i As Integer
Dim r, s As Integer
s = Y(1, col)
For i = 1 To n
    s = s + Y(i, col)
Next
SumElem = s
End Function
```

```

Sub ChangeCol(ByVal n As Integer, ByVal col1 As Integer, ByVal col2 As Integer, Y As Variant)
    Dim i, r As Integer

```

```

For i = 1 To n
    r = Y(i, col1)
    Y(i, col1) = Y(i, col2)
    Y(i, col2) = r
Next

```

```

End Sub

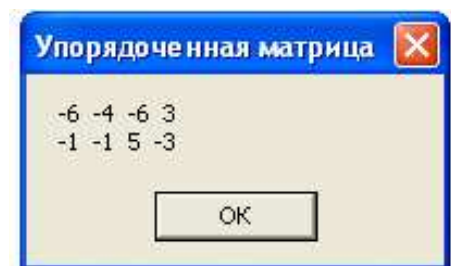
```

- Следующая программа решает задачу путем обращения к указанным функциям и процедурам:

```

Sub Sortify()
    Dim A() As Integer
    Dim s, mx As Integer
    Dim n, m, p, i, j As Integer
    Dim Title, Text As String

```



```

Title = "Размеры матриц"
Text = "Укажите число строк матрицы"
n = CInt(InputBox(Text, Title))
Text = "Укажите число столбцов матрицы"
m = CInt(InputBox(Text, Title))

```

```

ReDim A(n, m)
Call FormMatr(n, m, A())
Call PrintMatr("Исходная матрица", n, m, A())

```

```

Call SortMatr(n, m, A())
Call PrintMatr("Упорядоченная матрица", n, m, A())

```

```

End Sub

```

- **Анализ.** При построении функции был использован другой способ описания массива в списке формальных параметров: `Y() As Integer`. Его действие аналогично описанию `Y As Variant`.
- При решении задачи были использованы процедуры, которые разработаны ранее, при создании Программы 9. **Это обычный стиль работы профессиональных программистов.**



Контрольные вопросы

1. Зачем используются процедуры?
2. Что такое процедура?
3. Как вызывается процедура?
4. Что такое функция?
5. Как используются в программе функции?
6. Чем процедура отличается от функции?
7. Что такое формальные параметры процедуры?
8. Что такое фактические параметры?
9. Что такое параметры-переменные?
10. Что такое параметры-значения?
11. Какие виды параметров используются для исходных данных при работе процедуры?
12. Какие виды параметров используются для результатов?
13. Как передаются в процедуру массивы?
14. Как транслятор реализует передачу параметров-переменных по ссылке?
15. Как транслятор реализует передачу параметров-значений?
16. Что такое локальные переменные?
17. Что такое глобальные переменные?
18. Зачем используется оператор **Option Explicit**?
19. Зачем используется оператор **Option Base 1**?
20. Что такое случайная величина?
21. Что такое псевдослучайная величина?
22. Как с помощью функции Rnd выдавать псевдослучайные величины?
23. Что такое матрица?
24. Какие матрицы можно умножать?



Лабораторная работа

Цель работы. Научиться строить и выполнять в среде программирования VBA программы с массивами.

Задание. С помощью преподавателя выберите задачу и постройте для нее программу. Документ с программой запомните в личной папке.

Задачи

1. Составить функцию для определения суммы цифр числа. Использовать эту функцию в программе, которая вводит два целых числа и определяет число с большей суммой цифр.
2. Составить функцию для определения скалярного произведения двух векторов – суммы произведений элементов двух одномерных массивов одинаковой длины. Использовать эту функцию в программе.
3. Составить функцию для нахождения значения многочлена в данной точке. Использовать эту функцию для вывода таблицы значений многочлена в заданном интервале с заданным шагом.
4. Составить функцию для вычисления факториала $n! = 1 \cdot 2 \cdot 3 \cdot \dots \cdot n$ и использовать ее в программе.
5. Составить функцию для вычисления числа перестановок из n элементов по m элементов $C_n^m = \frac{n(n-1) \cdot \dots \cdot (n-m+1)}{m!}$ и использовать ее в программе.

Составить программу, которая строит прямоугольную матрицу со случайными целыми значениями и упорядочивает ее по возрастанию суммы элементов строк.

10 заповедей современного специалиста

1. Не используй компьютер во вред другим.
2. Не вмешивайся в работу других пользователей.
3. Не смотри без разрешения чужую информацию.
4. Не укради с помощью компьютера.
5. Не сообщай в сеть ложную информацию.
6. Не копируй и не используй программы, за которые не уплатил.
7. Не используй без разрешения ресурсы чужого компьютера.
8. Не присваивай чужую интеллектуальную собственность.
9. Думай о социальных последствиях своей программы.
10. Уважительно относись к другим пользователям.

Приложение 2

26 марта 2007 г.

Г. Харьков. ул. Веснина 5а

Уважаемые преподаватели,

Когда-то я был Вашим студентом. После окончания подфака я поступил на первый курс ХПИ по специальности «Информационные сети и системы». Сейчас я учусь на третьем курсе. Для того, чтобы оплачивать мою учебу, я работаю в свободное время. Я создаю программы по заказам из Интернета, которые я получаю на сайте www.RentAcoder.com. Для этого я купил компьютер, на котором я установил операционную систему **Windows XP Professional**. Мой компьютер я подключил к городской локальной сети (MAN) **www.Sloboda.net**. В этой сети я беру все нужные мне программы из файлов с открытым для копирования доступом. Так я установил систему программирования **Visual Studio.net**, с помощью которой я выполняю заказы из Интернета.

Я думал, что я все делаю правильно и законно. Недавно я прочитал книгу «Деловая этика специалиста», которая мне очень понравилась. В связи с этой книгой у меня появились вопросы, на которые я не знаю правильного ответа:

1. На моем компьютере установлена нелицензионная версия операционной системы. Инсталляционный диск с операционной системой я купил на рынке. Могу ли я у себя дома использовать компьютер с такой операционной системой. Законно ли это?

2. Программы по заказу я создаю с помощью системы программирования, которую я взял (с разрешения владельца компьютера) из городской сети. Я продаю исходный код написанный на языке программирования **C#** и разработанный в системе **Visual.net**. Я создал этот код и поэтому это моя интеллектуальная собственность. В чем я не прав?

С уважением,

Мохаммед, Ваш студент

Навчальне видання

Стрельченко Анатолій Йосипович

**ІНФОРМАТИКА
для інженерів та економістів**

Навчальний посібник

Російською мовою

Роботу до друку рекомендував доц. А.І. Лобода

В авторській редакції

Підп. до друку _____
Друк – лазерний.
Тираж 200 прим.

Формат 60×84 1/16
Умовн. друк. арк. 6,06
Зам. №

Папір – офсетний
Облік.-вид.арк. 6,18
Ціна – договірна.

Видавничий центр НТУ „ХПІ”. 61002, Україна, Харків, вул..Фрунзе,21
Свідоцтво про державну реєстрацію ДК № 116 від 10.07.2000 р.

Віддруковано в типографії СП «Сучасний друк» на цифровому лазерному видавничому комплексі Херох DocuTech135, 61024,м.Харків, вул.Лермонтовська, 27