**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ,
МОЛОДЕЖИ И СПОРТА УКРАИНЫ**

**ХАРЬКОВСКАЯ НАЦИОНАЛЬНАЯ АКАДЕМИЯ
ГОРОДСКОГО ХОЗЯЙСТВА**

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

**для выполнения лабораторных,
самостоятельных и контрольных работ**

по курсу

ИНФОРМАТИКА И КОМПЬЮТЕРНАЯ ТЕХНИКА

(*для студентов 1-го и 2-го курсов заочной формы обучения образовательно-квалификационного уровня бакалавр, направления подготовки 6.030601 «Менеджмент»*)

ХАРЬКОВ – ХНАГХ – 2011

Методические указания для выполнения лабораторных, самостоятельных и контрольных работ по курсу «Информатика и компьютерная техника» (для студентов 1-го и 2-го курсов заочной формы обучения образовательно-квалификационного уровня бакалавр, направления подготовки 6.030601 «Менеджмент») / Харьк. нац. акад. гор. хоз-ва; сост.: Б. И. Погребняк. – Х.: ХНАГХ, 2011. – 82 с.

Составитель: Б. И. Погребняк

Рецензент: к. ф.-м. н., доц. А. Б. Костенко

Рекомендовано кафедрой Прикладной математики и информационных технологий, протокол № 1 от 30 августа 2010 г.

Содержание

Стр.

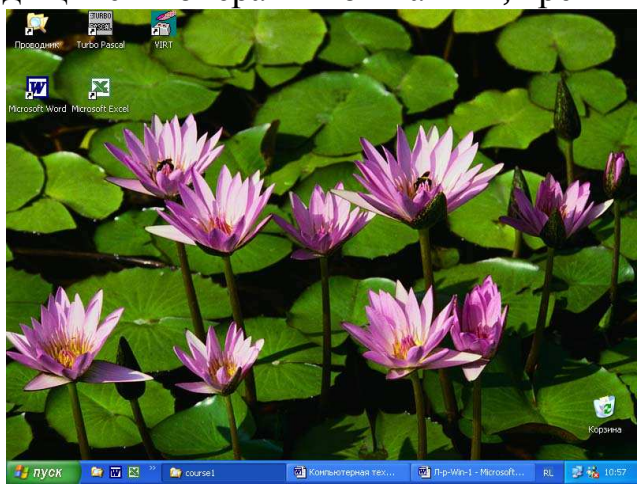
Лабораторная работа № 1

Операционная система Microsoft Windows

После включения компьютера и его самотестирования производится загрузка операционной системы. По окончании этого процесса на экране монитора отображается либо список пользователей данного компьютера, либо Рабочий стол. Список пользователей отображается, если на данном компьютере работает несколько пользователей и установлено разграничение прав доступа каждого из них. Рабочий же стол появляется после регистрации сеанса работы очередного пользователя, либо сразу после загрузки операционной системы – если разграничения прав доступа не установлено.

Сначала познакомимся с элементами рабочего стола Рабочий стол – это вид экрана после загрузки операционной системы (или регистрации очередного пользователя). Он состоит из:

- поля, на котором находятся различные значки, например «Проводник», «Корзина», «VIRT», «Microsoft Word» и др.,
- Кнопки «Пуск»,
- Панели инструментов, состоящей из панели Быстрого запуска и панели Задач. На приведенном ниже рисунке на панели Задач лежит три свернутые задачи: «Проводник» и два документа Microsoft Word.
- Индикаторы, например индикатор раскладки клавиатуры, значки программ, находящихся в оперативной памяти, время и др.



Основным элементом рабочего стола является кнопка «Пуск». Меню кнопки «Пуск» автоматически отображается при первом запуске Windows. Вернуться в это меню можно в любой момент, нажав кнопку «Пуск» на панели задач. В меню кнопки «Пуск» есть все необходимое, чтобы приступить к работе в Windows. В этом меню можно выполнять следующие действия:

- запуск программ;
- открытие файлов;
- настройка системы с помощью **Панели управления**;
- получение справки с помощью команды **Справка и поддержка**;
- поиск элементов на компьютере и в Интернете с помощью команды **Поиск**;
- и многое другое!

Рядом с некоторыми пунктами меню кнопки «Пуск» отображается направленная вправо стрелка. Это означает наличие еще одного уровня меню. Если поместить указатель на пункт меню со стрелкой, появится другое меню.

В левой части меню кнопки «Пуск» отображаются ссылки на наиболее часто используемые программы. В левом верхнем углу отображаются закрепленные или «приколотые» элементы – ярлыки таких средств, как обозреватель Интернета и программа электронной почты.

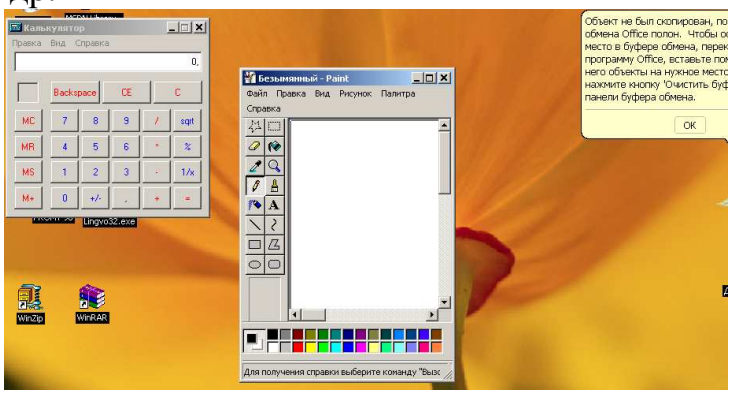
Подробнее обсудим следующие основные пункты:

- ***Все программы*** – перечень программ, установленных на данном компьютере и внесенных в перечень программ главного меню.
- ***Недавние документы*** – список недавно редактированных или просмотренных документов.
- ***Панель управления*** – перечень основных настраиваемых параметров на вашем компьютере.
- ***Поиск*** – перечень элементов поиска, например, файлы и папки, в Интернете и др.
- ***Справка и поддержка*** – запускает электронную справку по Windows.
- ***Выполнить…*** – вызывает окно «Запуск программы», позволяющее запускать программы путем введения их имени, например calc – запуск калькулятора.
- ***Выключение –*** позволяет правильно выполнить парковку (завершение работы), перезагрузку, переход к работе другого пользователя (завершение сеанса), переход в ждущий режим для уменьшения энергопотребления.

Для приобретения практических навыков выполните следующие действия:

1. *Внимательно рассмотрите рабочий стол и запишите в тетрадь: кнопки быстрого запуска, задачи, находящиеся на панели задач, основные команды меню «Пуск».*

Каждое приложение, работающее под Windows, открывается в своем собственном окне, например, программа калькулятор или графический редактор Paint и др.



Программы калькулятор и Paint относятся к стандартным программам. Их можно вызвать следующим образом: Пуск→ Все Программы→ Стандартные→ Paint или Калькулятор.

2. *Вызовите программы Paint и Калькулятор и установите такие же размер и расположение как в приведенном выше примере.*
3. *Запишите в тетрадь 4 основных элемента управления окнами с объяснением действия каждого из них.*

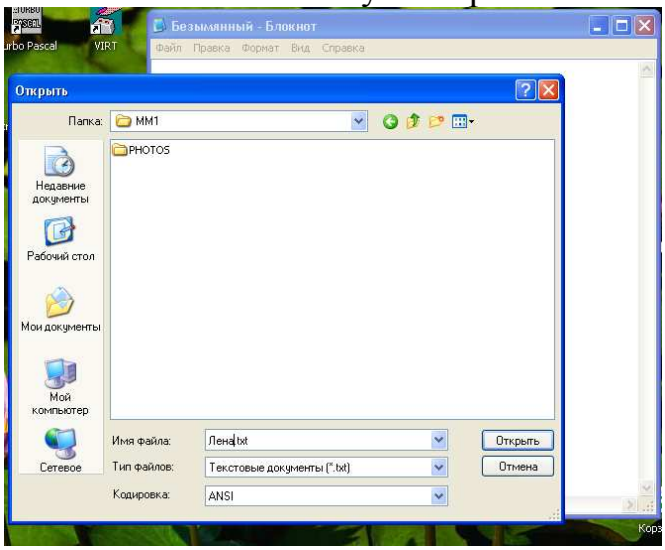
Окна можно не только перетаскивать с помощью мыши, но и упорядочивать *каскадом, сверху вниз и слева направо*. Для этого надо щелкнуть правой мышью на панели задач (между последней задачей и индикаторами), т. е. вызвать контекстное меню и затем выбрать в нем соответствующий пункт.

4. *Вызовите стандартную программу Блокнот (как это делалось ранее).*
5. *Уменьшите размер программы Блокнот так, чтобы были видны другие окна и расположите их каскадом.*
6. *Покажите результат преподавателю.*

Программа Блокнот – это простой текстовый редактор, который создает текстовые документы с расширением .txt. Основные операции, производимые в блокноте, выполняются с помощью меню состоящего из 4 основных пунктов: *Файл, Правка, Формат, Справка*.

Подменю команды «Файл»: *Создать, Открыть, Сохранить, Сохранить как…, Параметры страницы, Печать, Выход.* ***Создать*** означает создать новый файл, которого еще не было. Новый файл называется «Безымянный». При выборе команды создать появляется чистый экран и мигающий курсор, который указывает место ввода символов.

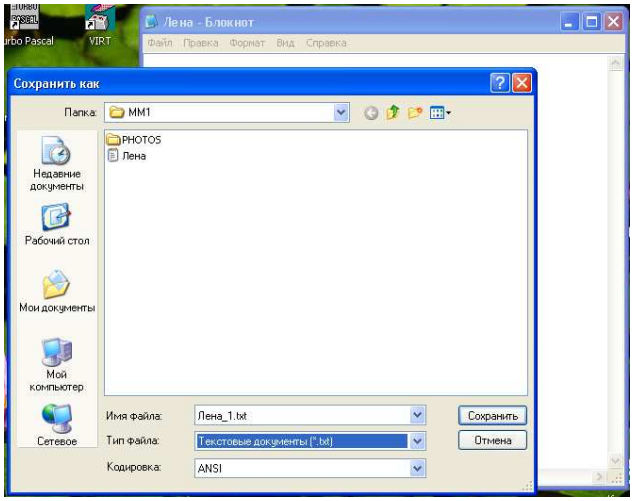
Команду «***Открыть»*** используют тогда, когда хотят изменить (отредактировать) уже имеющийся файл. Чтобы открыть файл нужно указать его имя и местонахождение. Это выполняется в специальном диалоговом окне «Открытие документа», где сначала отыскивают *нужную папку*, а затем *имя файла*, после этого нажимают кнопку «Открыть»



Команду «***Сохранить»*** применяют, когда хотят сохранить последние изменения файла, при этом, не меняя его имени. Сохранять информацию надо довольно часто, поскольку если долго не сохраняться, то можно потерять много информации, т. е. своего труда. Команда «***Сохранить как…***» позволяет

сохранить текущую информацию под новым именем, например под именем Лена_1.txt. Порядок сохранения файла совпадает с порядком открытия.

Команда ***Параметры страницы*** доступна только при наличии принтера и определяет размер бумаги и поля документа. ***Выход*** выполняет завершение работы программы Блокнот.



При наборе текста используются следующие правила и приемы: между словами ставятся пробелы, клавиша «Enter» нажимается только при переходе к новому абзацу, для создания абзацного отступа используется клавиша «Tab», знаки препинания прижимаются к слову, после которого стоят, после чего ставится пробел.

7. *Наберите в Блокноте краткую автобиографию, расскажите что любите, чем увлекаетесь. Напишите несколько абзацев.*
8. *Сохраните файл под именем <Ваша фамилия>.txt (например, Иванов.txt) в каталог <Ваша группа> (например, БУА-1), который нужно создать в папке «course 1».*

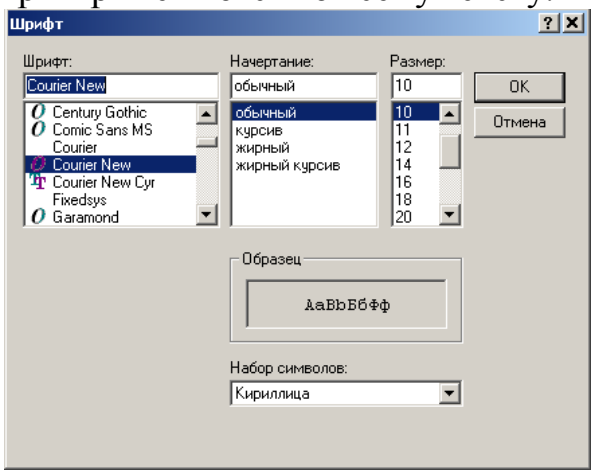
Подменю команды «Правка»: «*Отменить, Вырезать, Копировать, Вставить, Удалить, Найти, Найти далее, Заменить, Перейти, Выделить все, Дата и время*».

Отменить означает отменить последнее выполненное действие, например, Вы изменили шрифт, нажали отменить, шрифт снова стал таким как раньше. ***Выделить все*** означает выделить весь текст целиком, хотя можно выделять отдельные буквы или предложения. Например, выделять можно левой кнопкой мыши, протаскивая ее по тексту словно маркер. При этом текст меняет цвет. Выделенный фрагмент можно *Вырезать, Скопировать* или *Удалить*. ***Вырезать*** означает изъять фрагмент из текста и поместить его в буфер обмена. ***Копировать*** означает скопировать фрагмент в буфер обмена. ***Удалить*** означает просто уничтожить выделенный фрагмент. ***Вставить*** означает вставить фрагмент из буфера обмена в место нахождение курсора. Сначала установите курсор в нужное место, а только затем активизируйте команду *Вставить*. ***Найти, Найти далее*** и ***Заменить*** используется для нахождения слов и словосочетаний и замены их, например синонимами. Команда ***Дата и время*** вставляет в позицию курсора текущую дату и время.

9. *Выделите первое предложение вашего текста и скопируйте его в буфер обмена.*

10. *После последнего предложения, начиная с новой строки, вставьте скопированный текст.*
11. *Вставьте пустую строку перед первым предложением, и вставьте в нее текущую дату и время.*
12. *Покажите работу преподавателю.*
13. *Сохраните изменения под именем <Ваше имя>1.txt.*

Команда «Формат» имеет следующее подменю: *Перенос по словам* и *Шрифт*. Если пункт «***Перенос по словам***» помечен галочкой, то каким бы маленьким не было окно текст всегда будет виден, если же пункт «***Перенос по словам***» не помечен, то часть текста может быть не видна. При вызове команды ***Шрифт*** появляется диалоговое окно Шрифт, в котором можно указать название шрифта, начертание, и размер. В окошке «образец» можно посмотреть, как будут выглядеть выбранные вами параметры. Для применения выставленных параметров следует щелкнуть по кнопке ОК. Выбираемые параметры применяются ко всему тексту.



14. *Посмотрите, как работает команда «Перенос по словам».*
15. *Задайте для набранного файла следующие параметры: размер шрифта 20, начертание – жирный курсив, шрифт – Courier.*
16. *Сохраните файл под именем <Ваше имя>2.txt и покажите преподавателю.*

Контрольные вопросы

1. Как производится запуск операционной системы Windows?
2. Что такое Рабочий стол?
3. Перечислите элементы рабочего стола.
4. Объясните назначение кнопки «Пуск». Где она находится?
5. Объясните, как работает пункт меню, который состоит из одного названия; из названия, которое заканчивается многоточием (…); из названия, которое заканчивается стрелкой в право(►).
6. Объясните состав и назначение левой части основного меню Microsoft Windows.
7. Объясните состав и назначение правой части основного меню Microsoft Windows.
8. Объясните назначение панели Задач Microsoft Windows. Где она находится?
9. Объясните назначение панели Быстрый запуск. Где она находится?
10. Перечислите способы запуска приложений Microsoft Windows.
11. Перечислите способы завершения приложений Microsoft Windows.

12. Перечислите элементы управления окном.
13. Как можно расположить окна на рабочем столе?
14. Что такое активное (текущее) приложение? Как его можно изменить?
15. Какие стандартные программы Windows Вы знаете? Как их вызвать?
16. Перечислите состав и объясните назначение пункта основного меню Файл приложения Блокнот.
17. Перечислите состав и объясните назначение пункта основного меню Правка приложения Блокнот.
18. Перечислите состав и объясните назначение пункта основного меню Формат приложения Блокнот.
19. Перечислите последовательность действий, выполняемых при сохранении файла.
20. Какое расширение имеют файлы, создаваемые программой Блокнот?
21. Как называется новый файл Блокнота?
22. Как выделить фрагмент текста в программе Блокнот?
23. Какие операции можно производить с выделенным фрагментом текста?
24. Для чего используется буфер обмена?
25. Как изменить параметры шрифта? Какими они бывают?
26. Как вставить текущую дату и время?
27. Какие параметры шрифта Вы знаете?
28. Запустите приложения Блокнот и Калькулятор. Измените размер окна приложения Блокнот и попытайтесь изменить размер окна приложения Калькулятор. Объясните результат.
29. Как производится завершение сеанса работы пользователя в операционной системе Windows?
30. Как производится завершение работы операционной системы Windows?

Лабораторная работа № 2
Стандартные элементы управления

Стандартный интерфейс пользователя (GUI – Graphical User Interface) операционной системы Windows служит для обеспечения единообразного управления всеми его программами (приложениями). Он представляет собой набор стандартных элементов управления, которые функционируют и используются во всех приложениях одинаковым образом. Основным (фундаментальным) элементом управления этого интерфейса является Окно – отсюда и название операционной системы – Windows – т.е. окна. Всякое окно, прежде всего, обладает своей границей, которая в некоторых случаях может отображаться явно, а в других – неявно, но присутствует – всегда. Содержание же окна зависит от его назначения – это может быть и элементарная надпись и окно многодокументного приложения.

Всякое приложение Microsoft Windows может использовать один из следующих стандартных интерфейсов:

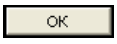
1. Многодокументный интерфейс (MDI – Multiple Document Interface), например, Microsoft Word или Microsoft Excel.
2. Однодокументный интерфейс (SDI – Single Document Interface), например, Блокнот, Paint, WordPad.
3. Интерфейс окна диалога, например, Калькулятор, Таблица символов. Диалоговые окна являются перемещаемыми, но не меняют своих размеров и не могут быть свёрнуты и развёрнуты. Специальным типом окна диалога является Мастер. Он «руководит» последовательностью действий пользователя при ответах на множество задаваемых вопросов, и обязательно содержит кнопки **Далее**, **Назад**, **Отмена** и **Готово**.
4. Интерфейс командной строки служит для запуска старых приложений MS-DOS и активизируется по команде **Пуск** → **Все программы** → **Стандартные** → **Командная строка**.

Окна диалога, которые используются как вспомогательные элементы управления основных приложений Windows, бывают двух видов:

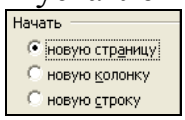
- модальные – которые приостанавливают выполнение приложения, из которого они были вызваны, до своего закрытия (например, окно **Параметры** в большинстве приложений Microsoft Windows) и
- немодальные – которые не приостанавливают выполнение «своего» приложения до их закрытия (например, окно **Найти и заменить** в Microsoft Word).

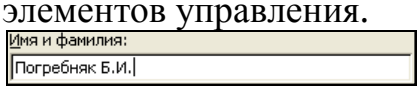
Все окна операционной системы Microsoft Windows организованы в виде некоторой древовидной (иерархической) структуры, которая называется Z-order (Z-порядок). То есть, каждое окно является «подокном» окна более высокого уровня иерархии (родительского), а так же может иметь «подчинённые» (дочерние) окна. На самом верху этой иерархии находится окно, которое называется Рабочий стол – он родительского окна не имеет.

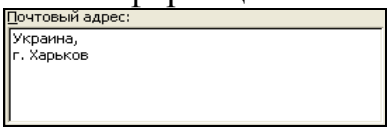
Для единообразного управления всеми приложениями Microsoft Windows имеются такие стандартные элементы управления, которые выполняют следующие функции:

1. ***Кнопка*** 

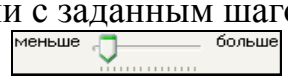
– щелчок по ней вызывает некоторое действие.
2. ***Статический текст*** 

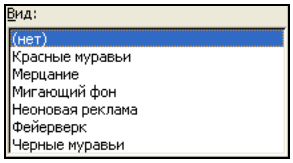
– любая надпись, которую пользователь не может изменить.
3. ***Флажок***  – служит для указания (или отмены) некоторого значения посредством его «установки», или «сброса». Флажки устанавливаются, или сбрасываются, независимо друг относительно друга.
4. ***Переключатель*** (радио-кнопка)  – служит для указания одного из многих альтернативных значений. Среди всех переключателей текущей группы может быть уставлен только один.
5. ***Групповая рамка*** 

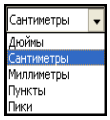
– служит для визуального выделения одного или нескольких, логически связанных, элементов управления.
6. ***Однострочное поле редактирования*** 

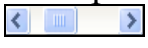
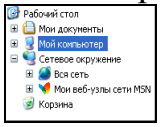
– служит для ввода одной строки алфавитно-цифровой информации.
7. ***Многострочное поле редактирования*** 

– служит для ввода нескольких строк алфавитно-цифровой информации.
8. ***Счётчик*** 

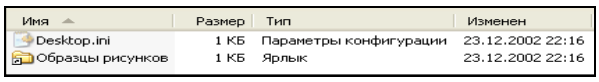
– предназначен для ввода числовой информации вручную или с заданным шагом.
9. ***Ползунок*** 

– предназначен для ввода числовой информации из заданного диапазона.
10. ***Список*** 

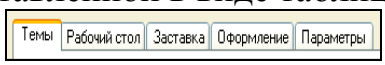
– служит для выбора одного из нескольких альтернативных значений.
11. ***Раскрывающийся список*** 

– также как и Список служит для выбора одного из нескольких альтернативных значений, но в отличие от первого, его необходимо в начале «раскрыть» – щёлкнув по кнопке со стрелкой вниз.
12. ***Полоса прокрутки***  – служит для «пролистывания» информации, которая не поместилась в окне. Бывают горизонтальные и вертикальные полосы прокрутки.
13. ***Полоса прогресса***  – служит для отображения хода выполнения длительных по времени процессов, например, копирования файлов.
14. ***Дерево*** 

– служит для управления отображением информации представленной в виде древовидной (иерархической) структуры. Щелчок по кнопке со знаком «+» вызывает «разворачивание» соответствующей ветви дерева, а по кнопке со знаком «-» – её «сворачивание».

15. ***Таблица*** 

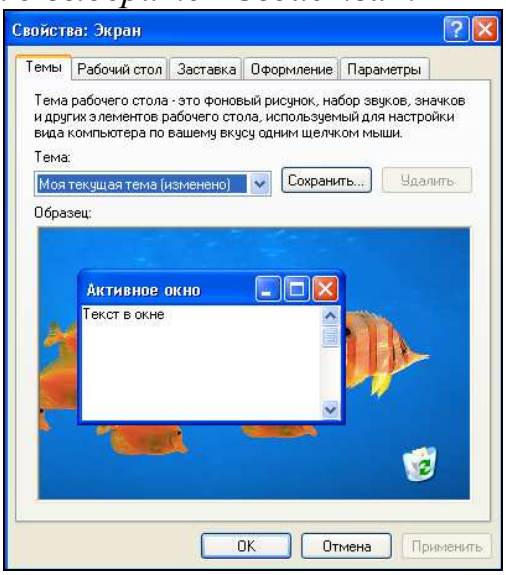
– служит для управления отображением информации представленной в виде таблицы.

16. ***Страницы свойств*** (***вкладки***) 

– напоминают записную книжку, и служат для группировки логически связанных элементов управления в пределах одного окна.

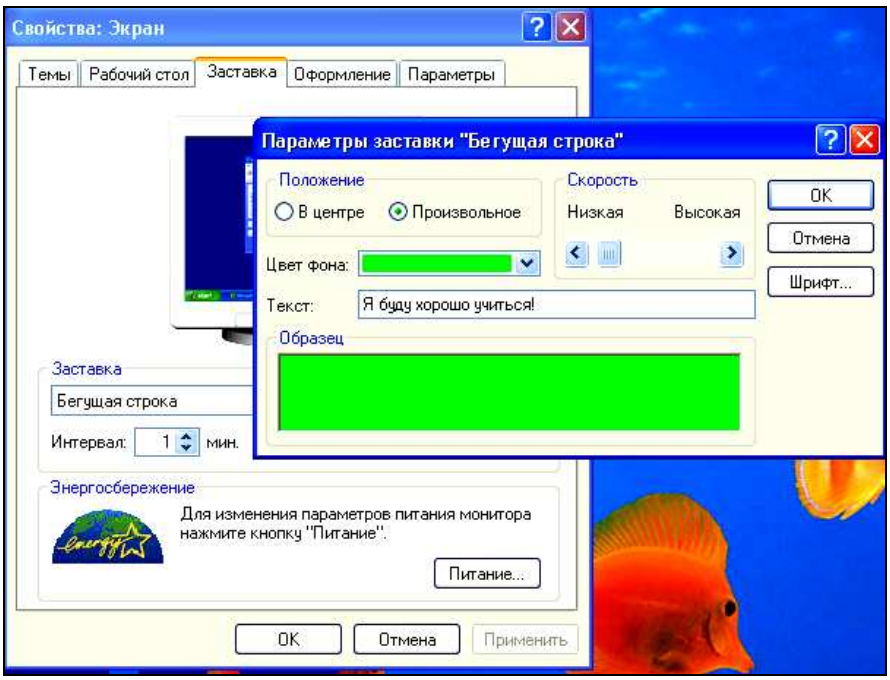
Для приобретения практических навыков выполните следующие действия:

1. *Вызовите контекстное меню к экрану рабочего стола, щелкнув правой клавишей мыши на свободном пространстве экрана.*
2. *В появившемся меню выберите «Свойства».*

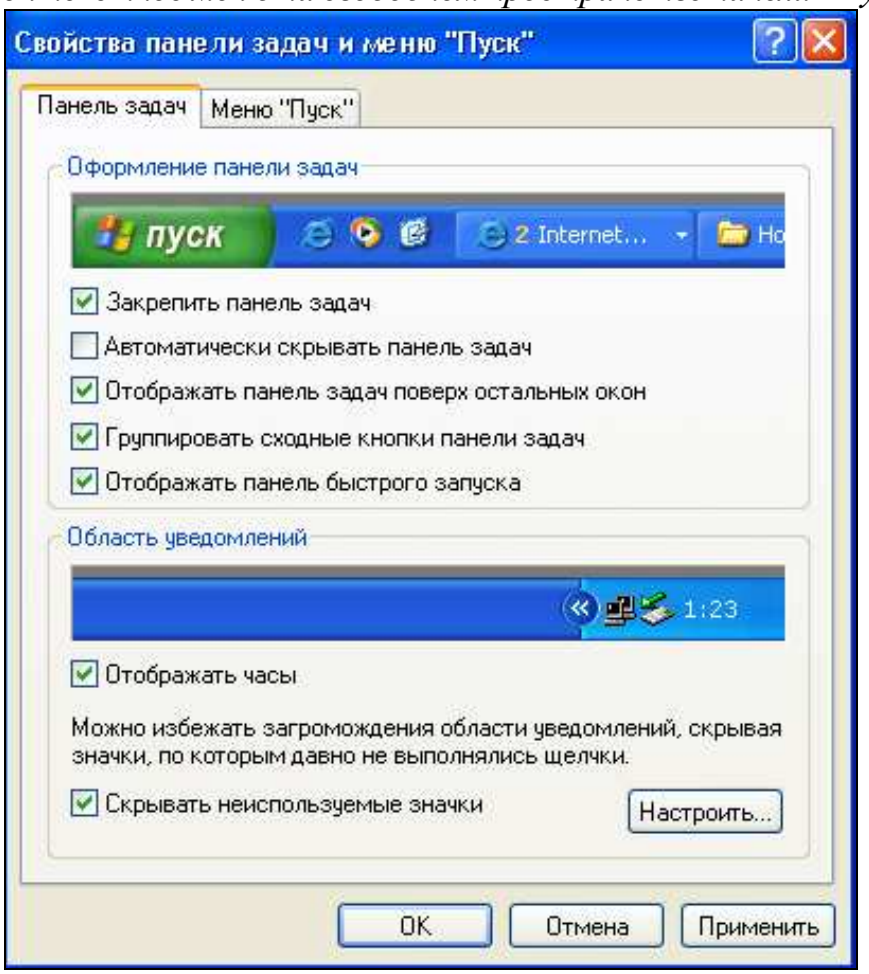


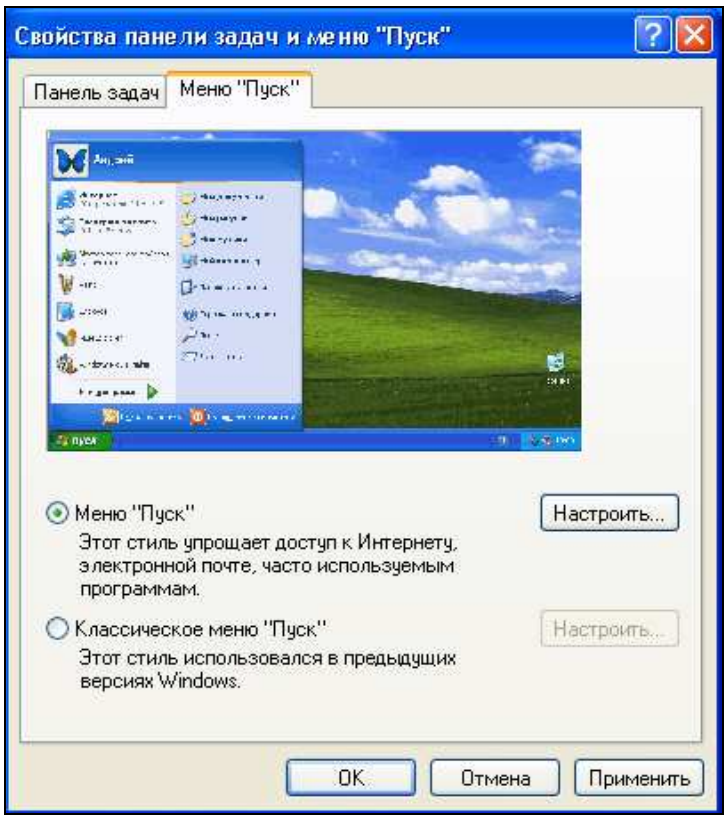
3. *Просмотрите все вкладки данного окна и найдите все известные вам элементы управления. Найденные элементы управления запишите в тетрадь.*
4. *Измените фоновый рисунок на Вашем рабочем столе и покажите преподавателю.*

5. *Сделайте заставку – пожелание вашим коллегам.*

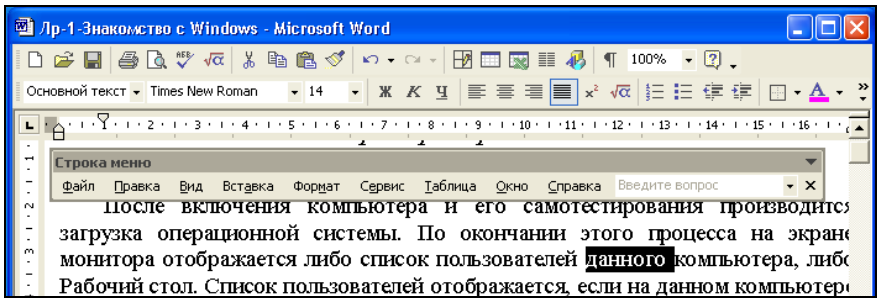


6. *Настройте параметры заставки таким образом, чтобы наличие заставки было легко обнаружить.*
7. *Покажите результат преподавателю.*
8. *Вызовите и просмотрите **не изменяя** настройки панели задач и меню «Пуск», вызвав контекстное меню на свободном пространстве панели «Пуск».*





9. *Запишите в тетрадь элементы управления, найденные в этом окне.*
10. *Запустите программу проводник по соответствующему ярлычку и найдите «дерево папок», «раскрывающийся список» и «таблицу», используя для этого кнопку «Папки», «адресную строку» и меню «вид» пункт «таблица».*
11. *Адресную строку, дерево и содержимое текущей папки зарисуйте в тетрадь. Покажите преподавателю.*



12. *Запустите программу MsWord или MsExcel, вызовите окно диалога «Параметры» (Сервис→Параметры) и записать в тетрадь все найденные там элементы управления.*
13. *Запишите в тетрадь, какие элементы управления вы не нашли.*

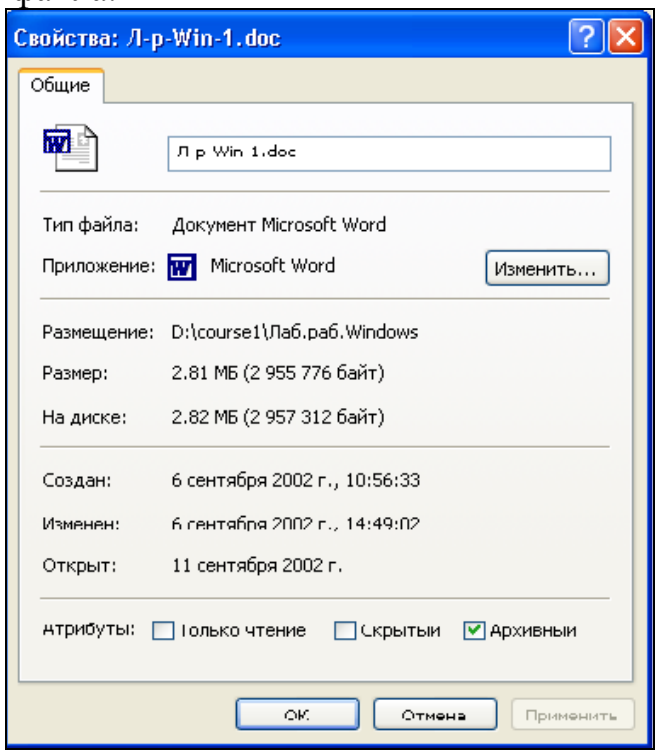
Контрольные вопросы

1. Что такое графический интерфейс пользователя (GUI – Graphical User Interface)?
2. Что такое многодокументный интерфейс (MDI – Multiple Document Interface)?
3. Что такое однодокументный интерфейс (SDI – Single Document Interface)?
4. Что такое интерфейс окна диалога? Что такое модальное окно диалога, что такое немодальное окно диалога, что такое Мастер?
5. Что такое интерфейс командной строки?
6. Что такое основное меню Приложения, что такое контекстное меню?
7. Что такое панель инструментов?
8. Покажите панель инструментов в «пристыкованном» и «плавающем» состоянии.
9. Покажите и объясните назначение заголовка окна приложения.
10. Покажите и объясните назначение Системного меню приложения.
11. Покажите и объясните назначение стандартного элемента управления (ЭУ) Кнопка.
12. Покажите и объясните назначение кнопок Свернуть, Развернуть/Свернуть в окно и Закрыть.
13. Покажите и объясните назначение ЭУ Флажок.
14. Покажите и объясните назначение ЭУ Переключатель.
15. Что общего, и чем отличаются стандартные элементы управления Флажок и Переключатель?
16. Покажите и объясните назначение ЭУ Статический текст.
17. Покажите и объясните назначение ЭУ Групповая рамка.
18. Покажите и объясните назначение ЭУ Список.
19. Покажите и объясните назначение ЭУ Раскрывающийся список.
20. Покажите и объясните назначение ЭУ Полоса прокрутки.
21. Покажите и объясните назначение ЭУ Полоса прогресса.
22. Покажите и объясните назначение ЭУ Поле редактирования.
23. Покажите и объясните назначение ЭУ Счётчик.
24. Покажите и объясните назначение ЭУ Ползунок.
25. Покажите и объясните назначение ЭУ Таблица.
26. Покажите и объясните назначение ЭУ Дерево.
27. Покажите и объясните назначение стандартного элемента управления Страницы свойств (Вкладки).
28. Какой элементы управления используются для ввода и текстовой и числовой информации?
29. Какие элементы управления используются для ввода только числовой информации?
30. Какие элементы управления используются для выбора (указания) одного значения из множества допустимых.

Лабораторная работа № 3

Файловая система

Файл – это поименованная совокупность данных, записанных на диске. Наиболее важными характеристиками файлов являются: имя файла, размер (занимаемое пространство на диске) файла и размещение (логическое местонахождение) файла.

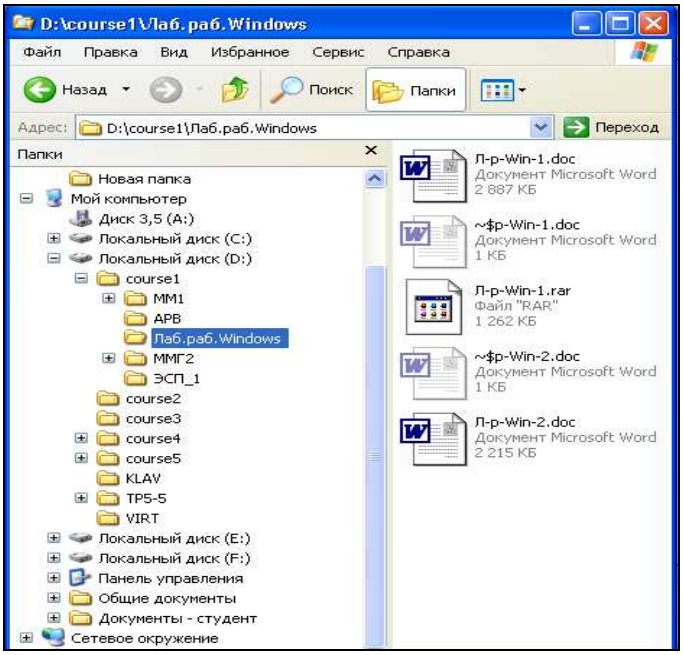


Файлы также различаются по типу. О типе файлов свидетельствуют расширения. Чаще других встречаются следующие типы файлов:

<Имя>.<расширение>	Тип файла	Программа, обрабатывающая эти файлы
*.txt	Текстовый файл	Блокнот (Notepad)
*.bmp	Точечный рисунок	Графический редактор Paint
*.doc	Документ	Microsoft Word
*.xls	Электронная таблица	Microsoft Excel
*.ppt	Презентация	Microsoft PowerPoint
*.exe	Приложение (выполнимая программа)	Специальные программы
*.hlp	Файл справки	Специальные программы
*.fon	Файл шрифта	Специальные программы
*.wma	Звуковой файл	Windows Media Player и др.
*.wmv	Аудио/видео файл	Windows Media Player и др.

Папки кардинальным образом отличаются от файлов. Папки в отличие от файлов не содержат внутренней информации (следовательно, не имеют размера). Папки предназначены для структуризации информации с целью облегчения доступа к файлам. Папка и каталог это одно и то же. В любой

папке могут находиться как файлы, так и папки. Иерархическая структура папок образует дерево папок. Дерево папок можно просмотреть в окне Папки программы Проводник.



Как показано выше, непосредственно в папке «course1» находятся следующие папки: course1, course2, course3, course4, course5, KLAV, TP5-5, VIRT. Если слева от имени папки находится ⊞, это означает что, в этой папке есть еще папки. Если такой кнопки нет, то эта папка не содержит в себе папок.

Для приобретения практических навыков выполните следующие действия:

1. *Запустите программу Проводник.*
2. *Сделайте текущей папку «course1».*
3. *Запишите в тетрадь имена папок находящихся в папке «course1».*

В папках можно создавать файлы и вложенные (или дочерние) папки. Для этого:

- Открываем папку, в которой будем создавать папки или файлы;
- В правом окне (показывает содержимое текущей папки) на свободном месте производим однократный щелчок правой кнопкой мыши (высвечивается контекстное меню).
- В контекстном меню выбираем пункт Создать (появляется дополнительное меню с типами файлов и папок).
- Из дополнительного меню выбираем (один раз левой кнопкой) соответствующий объект, например, папку.
- В текущей папке появляется 

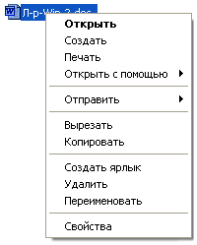
. Пока курсор мигает в рамке можно изменить текущее имя папки «Новая папка» на любое другое.

4. *Войдите в папку <Ваша группа>, например «ЭПГХ-2».*
5. *Создайте в папке <Ваша группа> папку <Ваша фамилия>, например «Иванов» (каждый сидящий за компьютером создает свою папку со своей фамилией).*

6. *В папке <Ваша фамилия> создать дерево папок, соответствующее Вашему варианту (номеру в журнале) из приведенного далее списка вариантов деревьев.*
7. *Показать созданное дерево папок преподавателю.*
8. *В папке, выделенной **полужирным** начертанием, создать два файла: текстовый документ (<Ваше имя>.txt) и точечный рисунок (<Ваше имя>.bmp).*
9. *В текстовом файле записать краткие сведения о себе (не забывайте сохранять файлы).*
10. *В точечном рисунке изобразите что-то приятное.*
11. *Покажите преподавателю созданные вами файлы.*

Файлы и папки можно *копировать, перемещать и переименовывать*. **Копировать** означает создавать точную копию файла или папки (при копировании количество объектов увеличивается). **Перемещать** означает менять месторасположение файла или папки (при перемещении количество объектов остается прежним). **Переименовывать** означает изменять имя файла или папки. Переименование производится через контекстное меню, вызываемое однократным правым щелчком по значку файла или папки. В контекстном меню вызывается пункт Переименовать. Для закрепления нового имени нажимается клавиша Enter или производится однократный левый щелчок на пустом пространстве окна.

Операции копирования и перемещения выполняются с помощью буфера обмена – пространства памяти, в котором может находиться файл, папка, фрагмент текста, рисунок и др. С буфером обмена производится в основном две операции: поместить объект в буфер обмена и вставить объект из буфера обмена. Операции копирования и перемещения выполняются в основном или через контекстное меню или с помощью меню Правка программы Проводник.



Алгоритм копирования файлов и папок

1. Сделать текущей папку, в которой находится копируемый объект;
2. Щелкнуть на значке объекта правой кнопкой мыши (вызов контекстного меню);
3. Выбрать пункт «Копировать» (выделенный объект помещается в буфер обмена);
4. Сделать текущей папку, в которую будет скопирован наш объект;
5. В окне, показывающем содержимое папки-назначения, на свободном месте щелкнуть правой клавишей мыши и в контекстном меню выбрать пункт «Вставить»(При многократном копировании следует повторять только последние два пункта)

Алгоритм перемещения файлов и папок

1. Сделать текущей папку, в которой находится перемещаемый объект;
2. Щелкнуть на значке объекта правой кнопкой мыши (вызов контекстного меню);

3. Выбрать пункт «Вырезать» (выделенный объект помещается в буфер обмена и исчезает из прежней папки (его значок бледнеет));
4. Сделать текущей папку, в которую будет перемещен наш объект;
5. В окне, показывающем содержимое папки-назначения, на свободном месте щелкнуть правой клавишей мыши и в контекстном меню выбрать пункт Вставить(При перемещении объектов новые объекты не появляются, они лишь меняют свое местонахождение)

*12. Скопируйте текстовый файл в папку, выделенную **полужирным курсивом**.*

13. Переименуйте копию файла из <Ваше имя>.txt в <Ваше имя1>.txt (добавьте в конце имени единицу).

*14. Переместите точечный рисунок из папки, выделенной **полужирным** начертанием, в папку, выделенную **полужирным курсивом**.*

15. Покажите преподавателю результат.

Индивидуальные задания

1.
```
──Папка1──┬Папка11──Папка111
          ├Папка12
          └Папка13┬Папка131
                  └Папка132
```

2.
```
──┬Folder1
  ├Folder2
  └Folder3──┬Folder31
            ├Folder32
            └Folder33
```

3.
```
──┬Папка1
  ├Папка2
  └Папка3──┬Папка31
           └Папка32
```

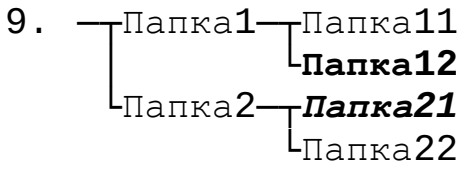
4.
```
──Folder1──┬Folder11──┬Folder111
           └Folder12  ├Folder112
                      ├Folder113
                      └Folder114
```

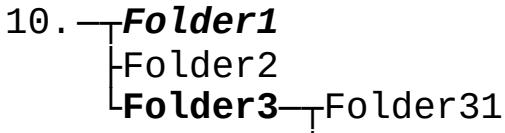
5.
```
──┬Папка1──┬Папка11
  └Папка2  ├Папка12──┬Папка121
           └Папка21  └Папка123
```

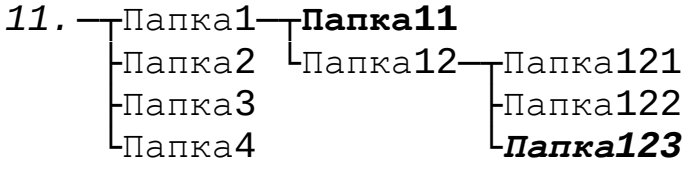
6.
```
──┬Folder1──Folder11──┬Folder111
  ├Folder2  ├Folder12  ├Folde112
  └Folder3  └Folder13  └Folder113
```

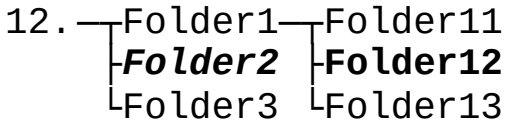
7.
```
──┬Папка1──┬Папка11
  ├Папка3  ├Папка12
  └Папка3  └Папка13
```

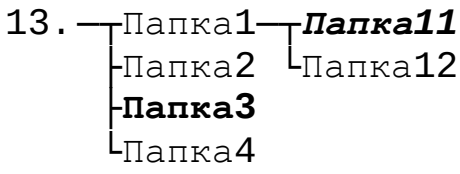
8.
```
──┬Folder1  ┌Folder31
  ├Folder2  ├Folder32
  └Folder3──└Folder33
```

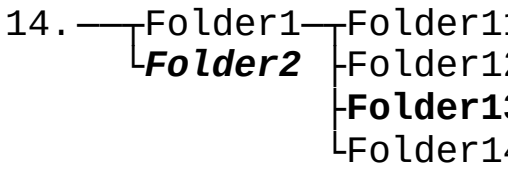
9.


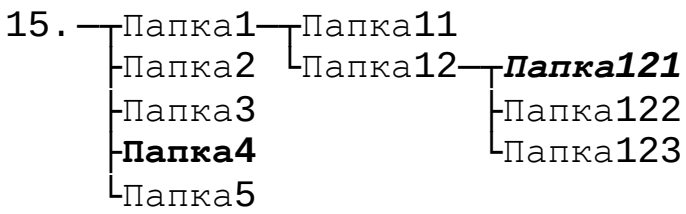
10.


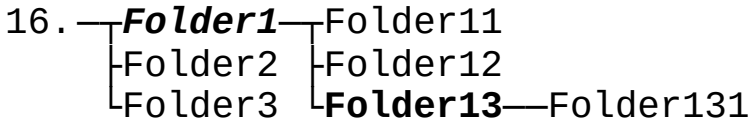
11.


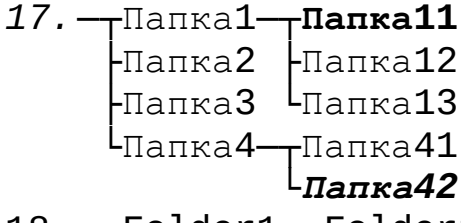
12.


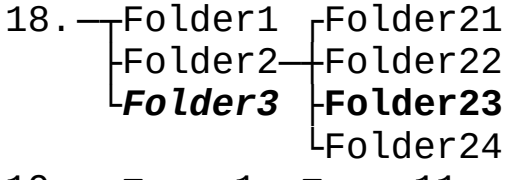
13.


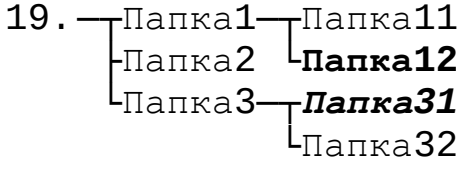
14.


15.


16.


17.


18.


19.


20.
- **Folder1**
- Folder2
- Folder3
  - Folder31
  - Folder32
- ***Folder4***
- Folder5

21.
- Папка1
  - Папка11
  - ***Папка12***
  - Папка13
- **Папка2**
- Папка3

22.
- ***Folder1***
  - Folder11
  - Folder12
  - Folder13
- Folder2
- Folder3
- Folder4
  - **Folder14**

23.
- Папка1
  - Папка11
  - Папка12
  - **Папка13**
- Папка2
- Папка3
- ***Папка4***

24.
- Folder1
  - Folder21
  - ***Folder22***
  - Folder23
- **Folder2**
- Folder3

25.
- **Папка1**
- Папка2
- Папка3
- Папка4
  - Папка41
  - Папка42
  - ***Папка43***

26.
- Folder1
  - **Folder21**
  - Folder22
  - Folder23
  - Folder24
- Folder2
- Folder3
- ***Folder4***

27.
- Папка1
  - **Папка11**
  - Папка12
- Папка2
  - Папка21
  - ***Папка22***

28.
- Folder1
  - Folder11
  - Folder12
  - Folder13
- **Folder2**
  - Folder21
  - Folder22
  - ***Folder23***

29.
- **Папка1**
  - Папка31
  - Папка32
  - Папка33
- Папка2
- Папка3
- Папка4
- Папка5
  - Папка51
  - ***Папка52***

30. ─┬Folder1─┬Folder11─┬Folder111
       ├Folder2 ├**Folder12** └Folde112
       └Folder3 └Folder13─┬Folder131
                          └***Folder132***

Контрольные вопросы

1. Что такое файловая система? Перечислите основные объекты файловой системы.
2. Перечислите названия файловых систем, которые поддерживает операционная система Windows XP.
3. Что такое файл?
4. Перечислите основные характеристики файла.
5. Что такое имя файла? По каким правилам оно строится?
6. Что такое расширение имени файла? Для чего оно предназначено?
7. Что такое полное имя файла? По каким правилам оно строится?
8. Что такое папка? Для чего она предназначена? Чем она отличается от файла?
9. Что такое текущая папка? Как можно ее изменить?
10. Что такое путь? Что он показывает?
11. Что такое имя папки? По каким правилам оно строится?
12. Что такое дерево папок?
13. Как вызвать дерево папок в Проводнике?
14. Как визуально отличить папку от файла?
15. Что такое физический диск, что такое логический диск? Покажите физические и логические диски на Вашем компьютере. Как именуются диски?
16. Как выделить одиночный объект (диск, папку, файл)?
17. Как выделить (указать, пометить) группу подряд идущих объектов?
18. Как выделить группу не подряд идущих объектов?
19. Объясните разницу между содержанием контекстного меню при щелчке правой кнопкой мыши на свободном пространстве текущей папки, и на каком-либо объекте (папке, или файле)?
20. Можно ли создать диск в папке?
21. Можно ли создать папку в папке?
22. Можно ли создать файл в папке?
23. Можно ли создать папку в файле?
24. Можно ли создать файл на диске?
25. Как называются программы, предназначенные для работы с папками и файлами? Как называется такая программа в Microsoft Windows?
26. Какая программа создает и редактирует текстовые файлы?
27. Какая программа работает с точечными рисунками?
28. Что такое копирование? Что можно скопировать? Как это сделать?
29. Что такое перемещение? Что можно переместить? Как это сделать? Чем отличается копирование от перемещения?
30. Что такое переименование? Что можно переименовать? Как это сделать?

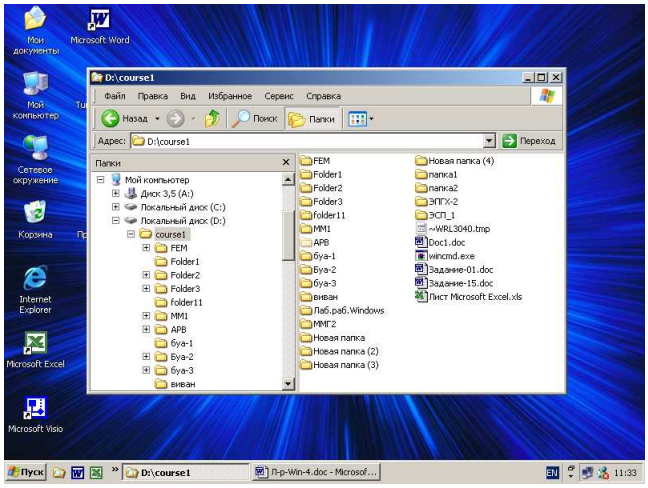
Лабораторная работа № 4

Проводник

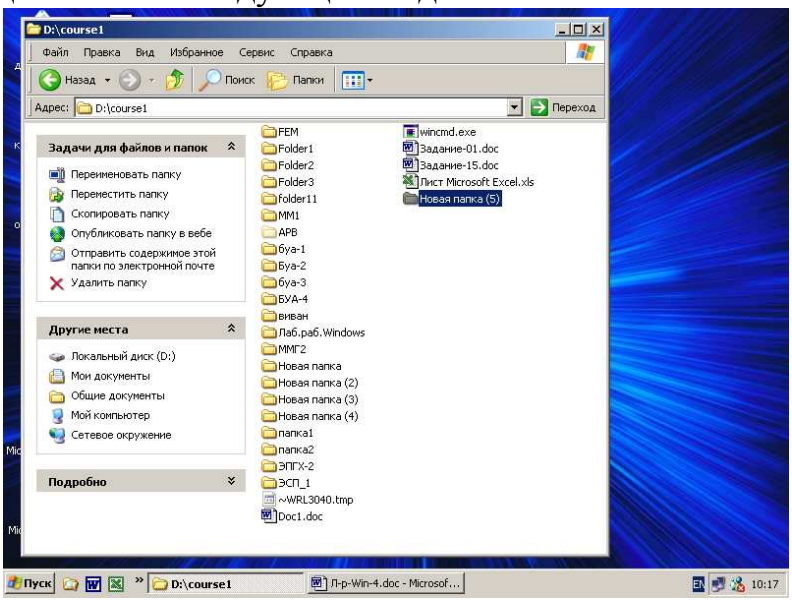
Программа Проводник относится к классу программ, который называется Файловые менеджеры. Они служат для манипуляции такими объектами файловой системы, как диски, папки и файлы.

Настройка программы Проводник

Программа Проводник, как и многие другие программы, имеет ***основное меню*** и несколько ***панелей инструментов***. **Основное меню** содержит следующие пункты: *Файл*, *Правка*, *Вид*, *Избранное*, *Сервис*, *Справка*. Обычно в Проводнике присутствуют две ***панели инструментов***: *Обычные кнопки* и *Адресная строка*.



Каждый пункт ***Основного меню*** имеет подпункты, которые предназначены для выполнения функций программы Проводник. Для выбора некоторого пункта основного меню необходимо один раз щелкнуть левой клавишей мыши по соответствующему пункту меню. При отключении всех кнопок Проводник имеет следующий вид:



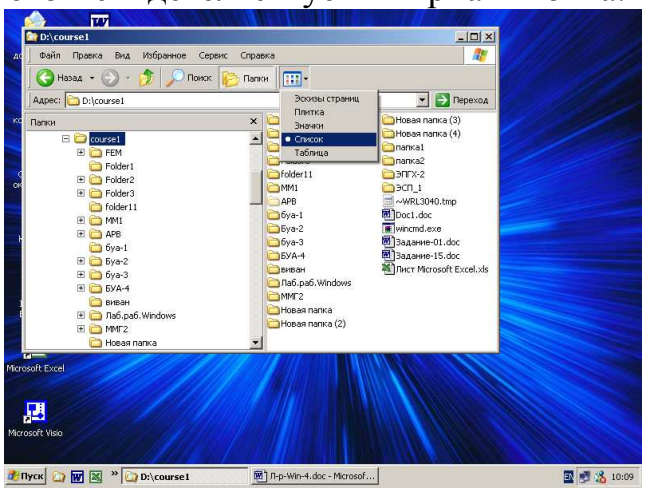
Для освоения этой темы выполните следующие действия:

1. *Запишите в тетрадь подпункты Основного меню.*

Против некоторых пунктов меню стоят комбинации клавиш, например «Копировать Ctrl+C», эти комбинации называются Горячими клавишами и

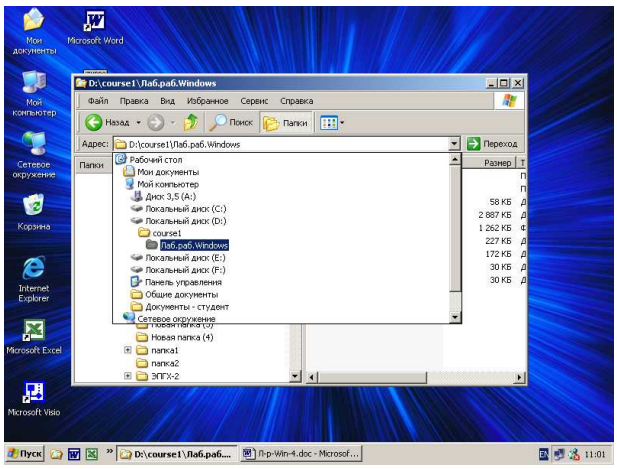
выполняют соответствующее действие. Например, поместить объект в буфер обмена можно двумя способами: «***Правка → Копировать»*** или ***Ctrl + C***. Некоторые функции, выполняемые с помощью пунктов основного меню, дублируются также с помощью кнопок на панели инструментов.

Рассмотрим функции панели инструментов ***Обычные кнопки***. Программа Проводник «запоминает» все просмотренные Вами папки и позволяет возвращаться назад или вперед к просмотренным папкам с помощью кнопок «Назад» и «Вперед». Кнопка «Назад» позволяет перейти к той папке, которую Вы просматривали ранее (на один, на два или более шагов назад). Кнопка «Вперед» становится доступной только тогда, когда Вы возвращались назад и теперь хотите перейти вперед. Кнопка «Вверх» позволяет подниматься по дереву папок вверх к корневой папке, например, если текущая папка <Ваша группа>, то при однократном щелчке по кнопке «Вверх», Вы перейдете в папку course1, которая является родительской для папки <Ваша группа>. Кнопка «Поиск» позволяет находить файлы, папки, компьютеры и др. Подробнее поиск будет рассмотрен здесь же, ниже. Кнопка «Папки» позволяет включать или отключать дерево папок в левой части окна. Кнопка «Вид» позволяет изменять способ отображения папок и файлов в правой части окна. Существует 5 способов отображения папок и файлов: Эскизы страниц, Плитка, Значки, Список, Таблица. На следующем рисунке выбран Список, об этом свидетельствует «жирная» точка.



2. Проверить работу описанных кнопок на практике.

Обычно ниже панели Обычные кнопки находится панель Адресная строка, которая показывает текущую папку и путь к ней, а также позволяет изменить текущую папку.



3. *Сделайте с помощью адресной строки текущей папку <Ваша фамилия> и с помощью кнопки «Вид» просмотрите все возможные способы отображения содержимого этой папки. Кратко запишите в тетрадь эти отличия.*

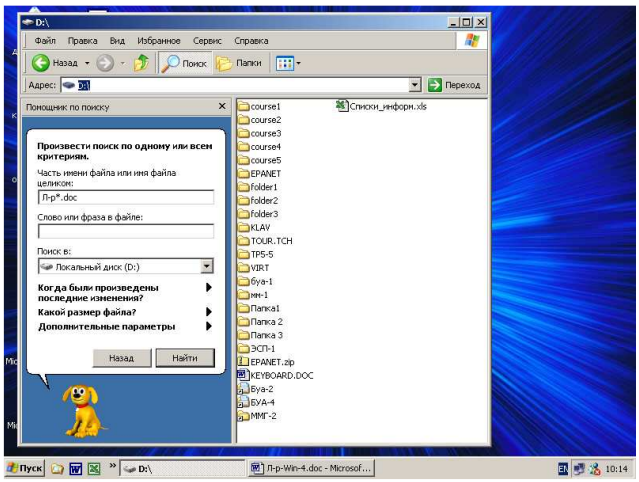
Поиск файлов и папок

Иногда бывает так, что забыто местоположение файла или папки, или Вы точно не помните имя файла или папки, или Вас интересуют файлы определенного типа, созданные в определенный период и др. Во всех этих случаях используют функцию поиск. Поиск файлов и папок можно осуществить из меню кнопки «Пуск» или вызвав функцию «Поиск» из программы Проводник, путем щелчка на кнопке «Поиск».

Для поиска файлов и папок надо выполнить следующие действия:

1. Щелкнуть по кнопке «Поиск», слева откроется окно «Помощник по поиску» см. ниже.

2. Выбрать раздел «Файлы и папки».



3. Указать имя файла или папки.

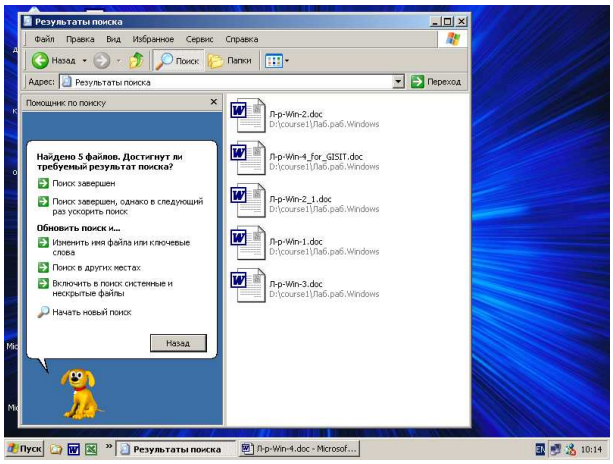
Файл может быть найден, если или указать **точное его имя** (все буквы с учетом регистра) или неизвестные буквы заменить (одна буква – знак «?», несколько букв «*»). Например, *.doc – означает любой файл, созданный программой Microsoft Word, или Ф??.txt – означает файл, начинающийся с буквы Ф, состоящий из трех букв, а кроме этого, это текстовый файл.

4. Указать область поиска.

Область поиска – это область жесткого диска, на котором будет производиться поиск. Областью поиска может быть папка (группа папок), диск, или группа дисков. Например, Мой компьютер (все диски A:, C:, D:, E: и т.д.), папка Мои документы, папка D:\course1, и др.

5. Вы можете указать некоторые дополнительные характеристики файлов и папок, например размер файла, время создания или изменения и др.
6. Нажать кнопку «Найти».
7. Результаты поиска отображаются в специальной папке «Результаты поиска».

В примере, приведенном на рисунках, демонстрируется поиск всех файлов типа документы Microsoft Word, которые начинаются с букв «Л-р» и находятся на локальном диске D:. Результатом поиска является 5 файлов приведенных ниже.



4. *С помощью кнопки Поиск найти все текстовые файлы, созданные в папке <Ваша фамилия>. Результаты записать в тетрадь.*

5. *Показать результаты поиска преподавателю.*

Создание ярлыков

Ярлык – это указатель на объект (папку, файл, диск). Ярлык – это файл, который содержит информацию, позволяющую получить быстрый доступ к объекту. Поэтому принято создавать ярлыки или на рабочем столе или в вашей папке. Ярлыки создаются на те объекты, которыми Вы часто пользуетесь. Ярлык можно создать двумя способами:

1) с помощью мастера;
2) через контекстное меню объекта.

6. *На диске D: в корневом каталоге создать ярлык на папку <Ваша группа>.*

Создание ярлыка с помощью мастера:

1) Зайти в папку, где будет находиться ярлык;
2) Указать путь (адрес) к объекту, на который делается ярлык.

Зайти на диск D: в корневой каталог → вызвать контекстное меню → выбрать Создать → Ярлык → В появившемся окне мастера создания ярлыков нажать кнопку «Обзор…» *и с помощью дерева указать на каталог <Ваша группа>→Нажать «ОК» →Нажать «Далее» →В появившемся окне ввести имя ярлыка «Группа <Ваша группа>»→Нажать «Готово»*

В результате на диске D: должен появится ярлык с названием, например, «Группа ЭПГХ-2», при щелчке по которому Вы увидите содержимое этой папки.

7. *Щелкните по созданному ярлыку мышкой.*
8. *Запишите в тетрадь, что происходит при щелчке по созданному вами ярлыку.*
9. *Пользуясь результатами прошлой лабораторной работы, для двух файлов, находящихся в папке, выделенной* ***полужирным*** *начертанием, создать два ярлыка, которые будут находиться в папке, выделенной* ***полужирным курсивом****.*
10. *Покажите результат преподавателю.*
11. *С помощью мастера создания ярлыков в папке <Ваша фамилия> создайте ярлык на диск D:.*

Создание ярлыка с помощью контекстного меню:

1) Вызвать контекстное меню к объекту, для которого создается ярлык;
2) В контекстном меню выбрать пункт «Создать ярлык». После этого ярлык появится в текущей папке;
3) Переместить ярлык из текущей папки в папку, в которой данный ярлык должен находиться.

12. *С помощью контекстного меню в папке <Ваша фамилия> создать ярлык на папку «Course1».*

Открыть корневую папку логического диска D: → Выделить папку «Course1» → Щелкнуть правой кнопкой мыши и в появившемся контекстном меню выбрать пункт «Создать ярлык» → Выделить созданный ярлык и, вызвав контекстное меню, выбрать пункт «Вырезать» → Сделать текущим папку <Ваша фамилия> → Вызвать контекстное меню и выбрать пункт «Вставить».

В папке <Ваша фамилия> появится ярлык, который будет называться «Ярлык для course1»

13. *Переименуйте ярлык таким образом, чтобы из названия исчезли слова «Ярлык для»*
14. *Покажите результат преподавателю.*

Контрольные вопросы

1. Как называются программы, предназначенные для управления дисками, папками и файлами? Как называется такая программа в операционной системе Microsoft Windows?
2. Какой тип интерфейса имеет программа Проводник?
3. Перечислите пункты основного меню программы Проводник. Какие функции они выполняют?
4. Что может отображаться в левой половине окна программы Проводник, а что – в правой?
5. Объясните назначение области задач программы Проводник? Где она находится?
6. Перечислить панели инструментов программы Проводник.
7. Перечислите элементы управления на панели инструментов Обыкновенные кнопки, а также объясните и их назначение.
8. Как отобразить или, наоборот, «спрятать» панель инструментов?
9. Какие способы отображения содержимого текущей папки Вы знаете? Чем они отличаются друг от друга?
10. Как отобразить дерево папок в программе Проводник?
11. Что такое текущая папка? Как выполняется «навигация» (перемещение) по папкам?
12. Объясните назначение панели инструментов Адресная строка? Как ею пользоваться?
13. Как выполняется поиск папок и файлов с помощью программы Проводник?
14. Как выполняется поиск папок и файлов, если их точные имена неизвестны?
15. Как указывается область поиска папок и файлов?
16. Как указываются дополнительные параметры поиска папок и файлов (например, созданных за последнюю неделю)?
17. Где отображаются результаты поиска?
18. Какие операции можно выполнять над найденными объектами? Как наиболее просто они реализуются?
19. Как перейти в папку, содержащую найденный объект?
20. Объясните назначение ярлыков.
21. Перечислите способы создания ярлыков.
22. Объясните способ создания ярлыка с помощью контекстного меню.
23. Объясните способ создания ярлыка с помощью мастера.
24. На какие объекты можно создать ярлык?
25. Что происходит при щелчке по ярлыку?
26. Что происходит при удалении ярлыка?
27. Что происходит при перемещении объекта, на который указывает ярлык?
28. Что происходит при удалении объекта, на который указывает ярлык?
29. Что происходит при переименовании объекта, на который указывает ярлык?
30. Чем внешне ярлык отличается от других объектов файловой системы?

Лабораторная работа № 5

Знакомство с Microsoft Word

Текстовый редактор Microsoft Word позволяет создавать текстовые и электронные документы, снабженные рисунками, таблицами, формулами, эффектами анимации и др. Основной особенностью современных текстовых редакторов является принципиально иной способ набора текста – клавиша «Enter» нажимается не в конце каждой строки, а в конце абзаца, а перенос слов со строки на строку осуществляется самим редактором с учетом заданных параметров форматирования.

Основное окно редактора Microsoft Word

После запуска текста лабораторной работы Вы увидите следующее:

В первой строке открывшегося окна, строке заголовка, содержится название открытого документа «Л-р-Word-1.doc» и через черточку – имя программы, с которой вы будите работать, Microsoft Word.

В следующей строке находится меню данного приложения «Панель строка меню», которое содержит такие пункты как «Файл», «Правка», «Вид» и т.д. Каждый из этих пунктов содержит свои подменю, которые можно активизировать путем щелчка по ним мышкой или клавишами перемещения курсора.

В следующей строке находятся кнопки «Стандартной панели инструментов». Каждая кнопка имеет подсказку, говорящую о назначении данной кнопки. Подсказка появляется при зависании указателя мыши над соответствующей кнопкой.

В следующей строке находятся кнопки «Панели форматирования».

Ниже находится окно, содержащее текст открытого файла. Справа от текста находится вертикальная полоса прокрутки. Снизу горизонтальная полоса прокрутки. Слева от горизонтальной полосы прокрутки находятся четыре маленьких кнопочки: «Обычный режим», «Режим Web-документа», «Режим разметки», «Режим структуры». Выбирая некоторый режим, Вы меняете способ представления текста. Каждый из режимов имеет свое назначение.

В самой последней строке Microsoft Word, строке состояния, находится справочная информация, например, номер страницы, номер раздела, общее количество страниц, номер строки, номер столбца, где находится курсор, далее перечисляются режимы, которые могут быть включены или выключены, язык, на котором набирается текст, и др.

Для освоения редактора Microsoft Word выполните следующие действия:

1. *Откройте поочередно все команды меню и запишите их в тетрадь.*
2. *Запишите в тетрадь в столбик название кнопок «Стандартной панели инструментов» и против каждой из них название команды меню, в котором есть соответствующая операция.*
3. *Покажите тетрадь преподавателю.*
4. *Просмотрите этот документ в четырех различных режимах просмотра документов.*
5. *Включите режим показа непечатаемых символов (кнопка на стандартной панели инструментов).*

Приложение Microsoft Word можно настраивать, например можно добавлять или убирать панели инструментов. Это делается следующим образом. *Сервис → Настройка → Панели инструментов*. В появившемся окне перечислены все возможные панели инструментов. Если против названия панели стоит галочка, то данная панель активна и находится в рабочем окне Microsoft Word.

6. Устанавливая галочки, просмотрите все возможные панели инструментов.

7. Оставьте в рабочем окне только две панели - «Стандартную» и «Форматирование».

8. Покажите преподавателю.

Работа с текстом

Перемещаться по тексту можно несколькими способами: ***клавишами*** перемещения курсора: вверх, вниз, вправо, влево; клавишами PageUp – экранная страница вверх; PageDown – экранная страница вниз; Ctrl+PageUp – начало документа; Ctrl+PageDown – конец документа; Home – начало строки; End – конец строки; ***мышью***, щелкая в нужном месте левой кнопкой. Если Вам нужен фрагмент, который находится за пределами видимости, то надо воспользоваться полосой прокрутки.

9. Проверьте описанные выше способы.

Выделить фрагмент текста можно несколькими способами. Самый простейший – в начале (или в конце) фрагмента нажать левую клавишу мыши и, не отпуская ее, тащить до тех пор, пока весь фрагмент не будет окрашен в черный цвет, затем отпустить кнопку. Этот способ универсальный, но не всегда удобный. Он не удобен тогда, когда надо выделить или очень большой или очень маленький фрагмент.

Текст можно выделить, пользуясь ***клавиатурой:*** установите курсор в начале (или в конце) фрагмента, нажмите Shift + → (или Shift + ←) и не отпускайте, пока нужный вам фрагмент не будет выделен.

Третий способ состоит в ***комбинировании мыши и клавиатуры***. *Одно слово* – двойной щелчок; *одно предложение* – Ctrl + щелчок; *один абзац* – тройной щелчок; *весь текст* – такими же способами как предложение или абзац только указатель мыши должен находиться слева от текста и иметь вид стрелочки слева направо. Если нужно выделить несколько слов подряд – нужно щелкнуть в начале, а затем в конце с нажатием клавиши Shift.

10. Освойте изложенные выше способы.

Если вы “испортили” исходный текст, нажмите кнопку «Отменить ввод» на панели инструментов «Стандартная». Чтобы снять выделение достаточно щелкнуть мышкой на тексте.

Любой фрагмент документа можно перенести или скопировать в тот же или другой документ с помощью «Буфера обмена». Чтобы поместить фрагмент в буфер надо выделить его и нажать кнопку «Копировать» (или воспользоваться контекстным меню). Чтобы вставить фрагмент надо установить курсор в то место, куда вы хотите вставить этот фрагмент и щелкнуть мышкой по той ячейке, которая содержит нужный фрагмент (или нажать кнопку «Вставить», при этом будет вставлен последний вставленный в буфер обмена фрагмент).

11. *Создайте новый документ путем щелчка на кнопке «Создать». Обратите внимание на название этого документа. Текст лабораторной работы будет лежать в виде свернутого окна на панели задач.*
12. *Сформируйте текст путем копирования фрагментов текста лабораторной работы через буфер обмена. Формируемый Вами текст должен содержать фрагменты текста в следующем порядке: Тема, Контрольные вопросы, Способы перемещения по тексту, Панели инструментов, Способы выделения текста.*
13. *Перед фрагментами текста «Способы перемещения по тексту», «Панели инструментов», «Способы выделения текста» вставьте по пустой строке, в которые поместите одноименные заглавия.*
14. *В конце текста наберите три абзаца текста, в которых Вы вкратце расскажите о себе, своей семье, своих увлечениях. Озаглавьте набранный вами текст словом «Автобиография».*
15. *Покажите преподавателю.*
16. *Сохраните новый документ в папке D:\course1\<Ваша группа> под именем «Фрагменты-<Ваша фамилия>.doc» Файл → Сохранить как … → в графе «Папка» высветить папку <Ваша группа>,в графу «Имя файла» вписать Фрагменты-<Ваша фамилия>, проверить, чтобы в графе «Тип файла» было выбрано «Документ Word (*.doc)» → Сохранить*
17. *Покажите преподавателю с помощью проводника каталог, в котором сохранен Ваш файл.*

Контрольные вопросы

1. Назовите способы запуска Microsoft Word, их достоинства и недостатки.
2. Назовите стандартные элементы управления в окне Microsoft Word. К какому типу интерфейса оно относится?
3. Что общего, и чем отличаются, меню и панели инструментов? Назовите особенности контекстного меню.
4. Какие режимы просмотра документа существуют в Microsoft Word? Как их установить?
5. Как убрать/добавить панель инструментов?
6. Назовите способы ввода текста в документ.
7. Перечислите способы перемещения по документу, покажите, где начинается и заканчивается документ.
8. Перечислите способы выделения фрагментов текста.
9. Что такое курсор, чем он отличается от указателя мыши?
10. Что такое непечатаемые знаки? Перечислите их. Как они отображаются?
11. Что такое абзац? Покажите его в документе.
12. Назначение клавиши **Enter** при работе с текстом.
13. Как продолжить ввод текста с новой строки, не создавая нового абзаца?
14. Как вставить одну или несколько пустых строк перед или после текущего абзаца?
15. Как «разломить» строку не две, как их «склеить»?
16. Как удалить один или несколько символов справа от курсора, слева от курсора? Как удалить фрагмент текста произвольного размера?
17. Что такое режим «Вставки»? Что такое режим «Замены»?
18. Как вводятся строчные и прописные буквы с клавиатуры?
19. Как переключить язык ввода информации с клавиатуры?
20. Почему при вводе текста некоторые его фрагменты бывают подчёркнутыми красной или зелёной волнистыми линиями?
21. Что такое буфер обмена? Как им пользоваться?
22. Что такое операция «Вырезать»? Чем она отличается от операции «Удалить»?
23. В чём состоит операция «Вставить»?
24. Что такое копирование фрагмента документа, что такое перемещение? Покажите способы выполнения этих операций.
25. Как заменить один фрагмент документа другим?
26. Как отменить неправильно выполненную операцию? Как вернуть отменённую операцию?
27. Сколько окон занимают два открытых документа?
28. Как создать новый документ?
29. Как сохранить текущий документ?
30. Назовите способы завершения работы Microsoft Word, их особенности.

Лабораторная работа № 6
Знакомство с Microsoft Excel

В процессе изучения этой темы Вы познакомитесь с рабочим окном табличного процессора Microsoft Excel, панелями инструментов, научитесь перемещаться по рабочему листу, а также по листам в пределах рабочей книги, пользоваться помощником.

Основное назначение электронных таблиц (табличных процессоров) – обработка информации, представленной в табличном виде. Базовым компонентом электронной таблицы является *ячейка*, которая обозначается неким *адресом (именем, ссылкой, координатами)*, например ячейка A1. Группа ячеек образует диапазон, например A1:C5.

Часть ячеек таблицы содержит некоторые исходные данные, а другая – формулы. Операндами формул являются адреса ячеек. При этом изменение содержимого какой-либо ячейки исходных данных приводит к немедленному изменению содержимого зависимых ячеек с формулами – результирующих, или выходных. То есть, содержимое таблицы обновляется (пересчитывается) автоматически.

Документ (т.е. объект обработки) в Microsoft Excel называется **рабочая книга**. Каждая рабочая книга имеет собственное имя и хранится в отдельном файле на диске. По умолчанию, новым рабочим книгам Excel дает имена **Книга 1**, **Книга 2** и т.д. При записи рабочей книги на диск к имени добавляется расширение **.xls**, так что на диск запишутся файлы, соответственно, **Книга 1.xls**, **Книга 2.xls**.

Каждая рабочая книга состоит не более чем из 255 **листов** различных типов. Самый распространенный тип листа – **Рабочий лист**. Он представляет собой набор ячеек, организованный в виде трок и столбцов и может состоять из одной или нескольких таблиц. По умолчанию рабочим листам присваиваются имена **Лист 1, Лист 2** и т.д. В окне документа программы Excel отображается только *текущий (активный)* лист, с которым ведется работа. Каждый лист имеет уникальное, в пределах книги имя, которое отображается на ярлычке листа, расположенном в его нижней части.

По умолчанию после запуска Microsoft Excel открывается окно рабочей книги **Книга 1 с тремя рабочими листами – Лист 1, Лист 2 и Лист 3.** В верхней части окна документа (рабочей книги) отображаются названия столбцов, по левому краю – номера строк, а внизу – ярлыки листов рабочей книги. В дополнение к традиционным панелям инструментов Стандартная и Форматирование, между ними и окном документа расположена *строка формул*, которая состоит из: раскрывающегося списка **Имя,** кнопки со стрелкой справа от списка **Имя,** кнопки **Отмена,** кнопки **Ввод,** кнопки **Изменить формулу, Поля ввода.**

Для освоения Microsoft Excel выполните следующие действия:

1. *Запустите Excel, щелкнув на соответствующем значке, на рабочем столе. Изучите возможность запуска Excel через* ***Пуск → Программы → Excel.***
2. *Изучите основные элементы открывшегося окна Excel . Укажите на строку заголовка, строку меню, стандартную панель инструментов, панель инструментов форматирования, окно адреса ячейки, строку*

формул, полосы прокрутки листа, строку состояния, панель задач.

3. *Изучите строку меню Excel. Законспектируйте вложенные меню каждой команды строки меню. Обратите внимание на активные и пассивные команды.*
4. *Изучите открывшееся пустое окно книги. Обратите внимание на основные объекты, с которыми работает Microsoft Excel*:
   - рабочая книга,
   - лист,
   - строка,
   - столбец,
   - ячейка,
   - диапазон,
   - список.

Навигация по листам

Для перемещения по листам в пределах рабочей книги необходимо, чтобы в левой нижней части окна документа отображались ярлычки листов и кнопки их прокрутки. 

Отображение (или скрытие) ярлычков листов рабочей книги выполняется с помощью установки (или сброса) флажка **ярлычки листов** в группе **Параметры окна** на вкладке **Вид** диалогового окна **Параметры**, которое активизируется по одноименной команде из меню **Сервис**.

Перейти на нужный лист рабочей книги можно одним из следующих способов.

- Щелкнуть ярлычок с именем необходимого листа (если он, конечно, виден на экране). Если нужного ярлычка не видно, то можно:
- с помощью вешки ярлычков увеличить пространство под них, или
- щелкать по кнопкам прокрутки ярлычков до тех пор, пока не появится ярлычок с нужным названием листа.
- Нажимать сочетание клавиш **Ctrl** + **Page Up** или **Ctrl** + **Page Down** до тех пор, пока не будет выбран ярлычок необходимой рабочей книги.
- Щелкнуть правой кнопкой мыши в области кнопок прокрутки ярлычков и выбрать, затем, из раскрывшегося контекстного меню имя требуемого листа.

5. *Переместитесь на **Лист 2**, используя ярлычки листов. Вернитесь на **Лист 1**.*
6. *Измените размер листа, воспользовавшись кнопкой «**Восстановить**».*
7. *Сверните окно книги, воспользовавшись кнопкой «**Свернуть**».*
8. *Разверните окно книги до максимального размера.*
9. *Последовательно сделайте активными ячейки A1, A19, K19, K1,A1.*

Навигация по рабочему листу

Перемещение в пределах рабочего листа – наиболее распространенный вид перемещения. Оно может выполняться:

- с помощью клавиатуры,
- с помощью мыши,
- с помощью диалогового окна **Переход**.

Перемещение по рабочему листу **с помощью клавиатуры** выполняется по-разному, в зависимости от состояния переключателя клавиатуры **Scroll Lock**. Если он включен, то с помощью клавиш-стрелок перемещения курсора

выполняется прокрутка («скроллинг») видимой части рабочего листа относительно окна просмотра документа на один столбик или одну строку в соответствующем направлении. Если же он сброшен, нажатие одной из клавиш-стрелок управления курсором вызывает перемещение активной рабочей ячейки в указанном направлении. (Нажимайте клавиши ↓, ↑, →, PgDn, PgUp, Ctrl+→, Ctrl+↓, Scroll Lock+→.) Законспектируйте комбинации клавиш, позволяющие перемещаться по листу с помощью клавиатуры.

10. *Установите указатель на любой **ячейке** и щелкните правой клавишей мыши. Изучите и законспектируйте открывшееся **контекстное меню**.*
11. *Установите указатель на любой **панели инструментов** и щелкните правой клавишей мыши. Изучите и законспектируйте открывшееся **контекстное меню**.*

Сверху над листами, как правило, находятся три основные панели инструментов Excel: **строка меню**, которое содержит такие пункты как «Файл», «Правка», «Вид» и т.д.; **стандартной панели инструментов**, представленная такими кнопками как создать, открыть, сохранить и др.; **панели форматирования** с кнопками «Шрифт», «Размер», «Начертание» и т.д.

12. *Изучите панель инструментов. Законспектируйте наиболее часто используемые команды стандартной панели инструментов.*
13. *Переместите стандартную панель инструментов к правому краю экрана, потянув мышью за вертикальную черту в начале панели. Сделайте панель «висячей».*
14. *Выберите меню **Вид** → **Панели инструментов** → **Настройка** и установите на вкладке **Панели инструментов** флажки возле Стандартная, Форматирование и Граница. Расположите появившуюся панель Граница в удобном для Вас месте экрана.*
15. *Выберите вкладку **Параметры** открытого окна настройки и законспектируйте все вкладки данного окна.*
16. *Удалите кнопку **Вырезать** с панели инструментов Стандартная. Выберите меню **Вид** → **Панели инструментов** → **Настройка**. Выделите на панели инструментов кнопку **Вырезать** и перетащите ее за пределы панели.*
17. *Добавьте кнопку **Вырезать** на панель инструментов (**Вид** → **Панели инструментов** → **Настройка** → **Команды** → **Правка**, выбрать кнопку и перетащить ее на панель инструментов; либо выбор контекстного меню на панели инструментов).*
18. *Добавьте на панель инструментов кнопку **Очистить.***
19. *Щелкните на кнопке **Справка** строки меню. Изучите открывшееся окно справки. Получите справку по вводу формул в таблицу (Вызовите помощника и введите в окно для поиска Формула, затем из перечня предлагаемых ссылок выберите Создание формулы). Покажите результат преподавателю.*
20. *Скопируйте полученную справку через буфер обмена в свой каталог в файл с именем Справка.doc. Закройте окно справки.*
21. *Щелкните правой клавишей мыши на **Помощнике** и выберите Параметры. Установите флажки возле всех опций, кроме «Отображать только важные советы». Щелкните на **ОК**. Попробуйте поменять помощника, выбрав другого на вкладке **Коллекция**.*

22. *Щелкните правой клавишей мыши на* ***Помощнике*** *и выберите* ***Мотор!*** *Отобразите результат в тетради.*
23. *В соответствии со своим вариантом выберите предметную область и составьте в тетради «отчет продаж» и «список товаров», аналогичных приведенным ниже.*

Пример «отчета продаж» для предметной области «компьютерная фирма»

	Фирма K&S				
	Январь	Февраль	Март	Апрель	Май
Статьи дохода					
Компьютеры	34890	45750	43780	51250	65740
Комплектующие	10980	12550	15670	16780	11570
Периферия	13450	11650	10550	4870	4550
Доход всего					
Статьи расхода					
Реклама	750	1200	1450	250	900
Аренда	1000	1000	1000	1000	1000
Налоги	2000	1500	2500	3000	5000
Расход всего					
Прибыль					

Пример «Списка товаров» для предметной области «молочный ларек»

категория	товар	модель	цена(нал.)	количество	дата поступления
Йогурт	ананасовый	виолла	10.7	50	30.04.2003
Йогурт	ананасовый	ромол	9.24	100	12.05.2003
Йогурт	клубничный	старый мастер	52.8	20	24.04.2003
Йогурт	клубничный	эдем	38.3	30	24.04.2003
Молоко	козье	чумак	2.45	240	28.04.2003
майонез	оливковый	фани	3.45	150	08.05.2003
Йогурт	персиковый	королевский	24	32	01.05.2003
Сыр	плавленый	ромол	1.25	30	12.05.2003
майонез	провансаль	заречье	8.65	90	10.05.2003
Масло	сливочное	фани	2.55	150	08.05.2003
Сыр	твердый	заречье	2.3	100	10.05.2003
Сыр	твердый	ромол	2.1	50	12.05.2003
Сыр	твердый	фани	1.8	60	08.05.2003
Молоко	топленое	хжк	2.45	200	28.04.2003
Масло	шоколадное	ромол	2.55	100	12.05.2003

Индивидуальные задания

Вариант	Предметная область	Вариант	Предметная область
1	Молочный ларек.	16	Кондитерский ларек.
2	Хлебный магазин.	17	Винно-водочный магазин.
3	Магазин «Соки – воды».	18	Магазин «Овощи – фрукты».
4	Склад-магазин пиломатериалов.	19	Колбасный магазин.
5	Табачный ларек.	20	Книжный магазин.
6	Аптечный ларек.	21	Игрушечный магазин.
7	Магазин канцпринадлежностей.	22	Обувной магазин.
8	Ювелирный магазин.	23	UMC-ларек.
9	Магазин оргтехники.	24	Хозяйственный магазин.
10	Ларек по продаже бытовой химии.	25	Магазин бытовой техники.
11	Магазин верхней одежды.	26	Мебельный магазин.
12	Автомагазин.	27	Цветочный магазин.
13	Магазин «Часы».	28	Магазин по продаже обоев.
14	Магазин «Ковры».	29	Магазин осветительных приборов.
15	Магазин по продаже музыкальных инструментов.	30	Косметический ларек.

Контрольные вопросы

1. В чем состоит основное назначение электронных таблиц?
2. Каким другим способом можно запустить Microsoft Excel, если на рабочем столе нет пиктограммы Excel?
3. На панели инструментов отсутствуют некоторые знакомые кнопки и присутствуют ненужные. Как вернуть привычное состояние панели?
4. Что произойдет при нажатии кнопки Сброс меню настройки панели инструментов?
5. Как можно увеличить видимую часть документа?
6. Перечислите основные объекты Microsoft Excel.
7. Какое расширение имеют файлы, созданные в Microsoft Excel?
8. Что такое рабочая книга?
9. Какие типы листов Вам известны?
10. Что такое рабочий лист, из чего он состоит?
11. Какое обозначение имеют строки, столбцы таблицы?
12. Что представляет собой ячейка таблицы, какая информация в ней может содержаться?
13. Какие константы может содержать ячейка?
14. Что означает адрес ячейки?
15. Какие стили адресации используются в Microsoft Excel?
16. Как изменить стиль адресации ячеек?
17. Из каких элементов состоит строка формул?
18. Поясните понятия: текущая (активная) рабочая книга, текущий лист, текущая ячейка?
19. Перечислите способы навигации по листам книги.
20. Как отобразить (скрыть) ярлычки листов?
21. Перечислите способы перемещения по рабочему листу.
22. Какие комбинации клавиш позволяют перемещаться по листу с помощью клавиатуры.
23. Как вызвать Помощника, как можно изменить его вид?
24. Как активизировать неактивные команды вложенных меню строки меню?
25. Перечислите режимы, в которых может находиться Microsoft Excel?
26. Как активизировать закрытую рабочую книгу?
27. Какими способами может осуществляться переход между открытыми рабочими книгами?
28. Как создать новый документ Microsoft Excel?
29. Как сохранить текущий документ?
30. Назовите способы завершения работы Microsoft Excel.

Лабораторная работа № 7

Создание и выполнение макросов

Цель работы: Изучение, освоение и приобретение практических навыков записи и выполнения макросов, работы в интегрированной среде разработки – Редакторе Visual Basic.

Краткие теоретические сведения

Понятие **макрос** относится к способности приложения записывать, а затем, многократно воспроизводить последовательности выполняемых пользователем действий. То есть макрос – это «протокол» выполняемых пользователем действий. Запись макроса выполняется при помощи **макрорекордера** за 4-е шага:

1. Создание начальных условий.
2. Запуск процедуры записи.
3. Выполнение действий, которые должны быть записаны в макрос.
4. Останов записи макроса.

Записываемый макрос можно сохранить в одной из книг:

- личной книге макросов (PERSONAL.XLS),
- новой рабочей книге, или
- текущей рабочей книге.

В любой рабочей книге макросы записываются в специальные места, называемые **модулями**, которые по умолчанию именуются, соответственно, **Module1**, **Module2** и т.д. Каждый такой модуль может содержать один или несколько макросов. Наиболее быстро и удобно записывать и выполнять макросы – при помощи кнопок на панели инструментов **Visual Basic** Microsoft Excel.

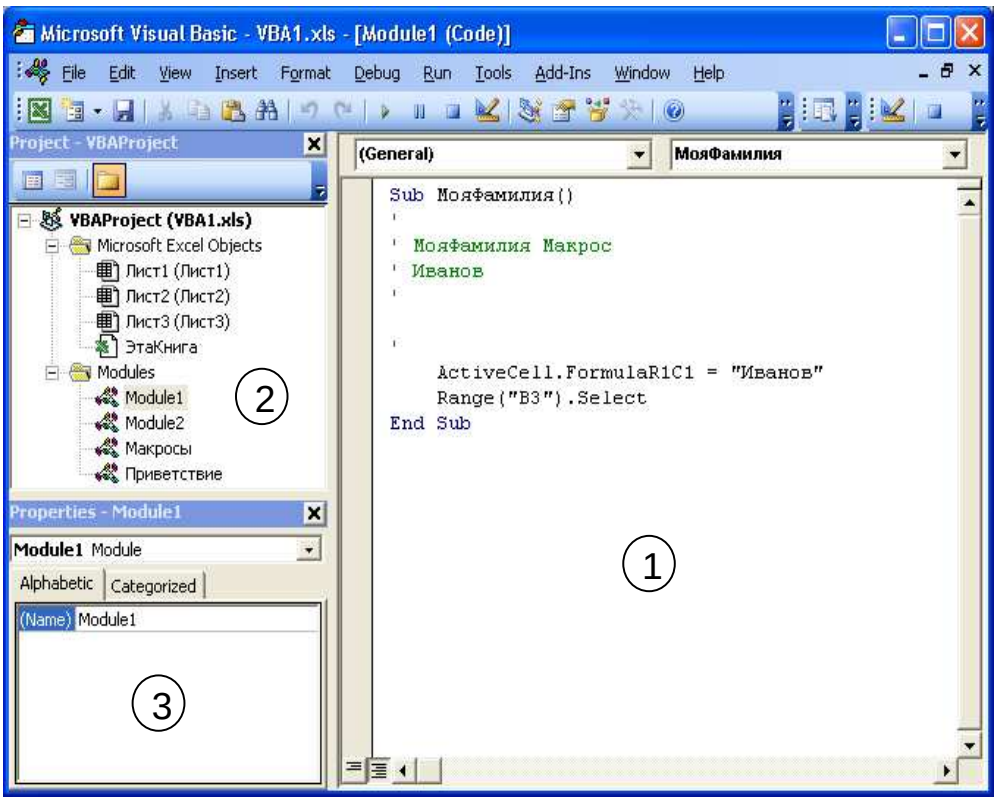
В процессе записи макроса команды, выполняемые пользователем, трансформируются в исходный текст языка **VBA – Visual Basic for Applications**. Язык VBA является языком программирования, который:

- с одной стороны является универсальным языком программирования, а
- с другой стороны – поддерживает средства доступа ко всем, специфическим, возможностям того приложения, из которого он был запущен.

Для создания, просмотра, редактирования и отладки макросов существует специальный инструмент – интегрированная среда разработки (IDE – Integrated Development Environment) – **Редактор Visual Basic** (Visual Basic Editor). На выполнение его можно запустить несколькими различными способами. Наиболее простые из них такие:

1. по команде **Сервис** → **Макрос** → **Редактор Visual Basic**,
2. по нажатию кнопки  **Редактор Visual Basic** на панели инструментов **Visual Basic** (если она, конечно, активизирована), или
3. нажав сочетание клавиш **Alt** + **F11**.

В результате откроется основное окно приложения Microsoft Visual Basic, состоящее из заголовка, меню, панели инструментов и рабочей области – т.е., стандартное окно многодокументного (MDI – Multiple Document Interface) приложения Microsoft Windows (Рис. 1).



1. *Окно Code*
2. *Окно Project*
3. *Окно Properties*

Рис. 1. Многодокументный интерфейс Редактора Visual Basic

Рабочая область основного окна приложения Редактора Visual Basic может содержать одно, или несколько, дочерних окон. Основными среди них являются следующие:

1. **Code** (**Код**) – предназначено для просмотра, редактирования и создания новых кодов (или, исходных текстов) макросов (Рис. 1 поз. 1).
2. **Project** (**Проект**) – предназначено для управления проектами. *Под проектом понимается совокупность взаимосвязанных модулей, и некоторых других компонентов, хранящихся в рабочей книге, или в ее шаблоне* (Рис. 1 поз. 2). По умолчанию имя проекта совпадает с именем рабочей книги.
3. **Properties** (**Свойства**) – отображаются свойства и значения, присвоенные каждому выбранному в окне **Project** компоненту (Рис. 1 поз 3).

Каждое из этих окон может быть закрыто щелчком по кнопке **Закрыть** справа в строке заголовка, а открыто – при помощи соответствующих команд пункта меню **View**, или кнопок на панели инструментов **Standard**. Одновременно открытых окон **Code** может быть несколько – в каждом из них отображается отдельный модуль, а вот окон **Project** и **Properties** может быть только по одному. Кроме того, окна **Project** и **Properties** могут находиться как в пристыкованном к одной из сторон родительского окна, та и в плавающем состоянии. Это положение регулируется при помощи флажка **Dockable** в контекстном меню. Пристыковываются они к одной из сторон родительского окна Редактора Visual Basic методом перетаскивания за заголовок. Границы дочерних окон в пределах рабочей области основного

окна приложения можно перемещать с помощью левой кнопки мыши, указатель которой при этом принимает вид двухсторонней стрелки.

Любая программа, на любом языке программирования состоит из:

1. *инструкций* (*операторов*, *команд*, *предложений* и т.д.), и
2. *комментариев*.

Инструкции служат для описания самого процесса обработки информации, и предназначены для выполнения компьютером. Комментарии же после трансляции никаких машинных команд не порождают. Они предназначены для пояснения наиболее сложных мест программы, и адресованы человеку – т.е. *программы пишутся для компьютеров, а читаются человеком*.

Если под термином «макрос» понимается код, записанный автоматически при помощи макрорекордера, то под «процедурой» – код созданный вручную. Исходный текст макроса (или процедуры), каким бы способом не была создан, состоит из *инструкций*, которые образуют:

1. *заголовок*,
2. *тело*, и
3. *окончание*.

Заголовок определяет имя процедуры, и ее начало в тексте модуля, а тело – выполняемые действия. Заголовок, в свою очередь, состоит из:

1) *ключевого слова* ***Sub*** (от Subroutine – подпрограмма),
2) имени процедуры, и
3) пары круглых скобок – ().

После тела макроса всегда следует пара ключевых слов ***End Sub***, которые говорят о том, что макрос закончился.

Тогда, структура процедуры, в общем виде, будет выглядеть следующим образом:

Sub <имя>()
[<инструкции>]
End Sub.

При создании и макросов, и процедур, используются отступы. Они улучшают читабельность программ и подчеркивают их структуру. Лучше всегда придерживаться один раз выбранной системы отступов.

Для вывода из программы сообщений пользователю в языке программирования VBA имеется встроенная функция ***MsgBox ( )***. Она имеет переменное число параметров. Только первый из них обязательный, остальные – нет. Ее первый параметр – это сообщение, которое необходимо вывести; второй параметр – количество и тип кнопок, которые будут в окне; третий – текст, отображаемый в заголовке окна. Если второй и третий параметры опущены, то по умолчанию в окне будет одна кнопка **ОК**, а в заголовке – имя того приложения, из которого запущен Редактор Visual Basic. При вызове функции на месте пропущенного, *непоследнего*, параметра ставится запятая.

В процессе создания практически любой компьютерной программы всегда возникают ошибки. При этом Редактора Visual Basic обнаруживает

большинство из них и выдает необходимые сообщения. Все ошибки, которые могут встретиться в программе, делятся на две категории:

1. *синтаксические*, и
2. *семантические*, или *времени выполнения* (*run-time error*).

Наиболее распространенной категорией ошибок являются синтаксические ошибки, связанные с нарушением грамматики языка программирования. Они информируют о таких неприятностях, как пропущенные запятые, недостающие кавычки, отсутствующие обязательные аргументы функции, и т. д.

Если после редактирования очередной инструкции кода, и после того как из нее будет переведен курсор, в ней не будет обнаружено синтаксических ошибок, то Редактора Visual Basic раскрашивает различные части этой строки в разные цвета в соответствии со следующими правилами:

- зеленым цветом выделяются комментарии,
- синим цветом выделяются ключевые слова, такие как ***Sub***, ***End Sub*** и другие,
- черным цветом – все остальное.

Если же в строке будет обнаружена ошибка, то Редактора Visual Basic локализует ее (на сколько это ему удается) – выделяет красным цветом, – и выводит соответствующее сообщение.

Вполне возможно написать такую программу, которая не будет содержать синтаксических ошибок, но при этом правильно ее выполнить на компьютере будет не возможно. Ошибки такого рода называются *семантическими ошибками*, или *ошибками времени выполнения* (*run-time error*). Подобные ошибки могут быть вызваны различными причинами:

- пропущенными обязательными аргументами процедуры или функции,
- несоответствием типов аргументов,
- попыткой доступа к несуществующему ресурсу, либо
- просто ошибкой в логике работы программы.

Ход работы

Задание 1. Создать макрос, который в активную ячейку будет записывать Вашу фамилию.

Шаг1. Запись макроса

1. Привести приложение в исходное состояние. Для этого необходимо выполните следующее:
   1) В своей рабочей папке (например, D:\course1\БУА-1\Иванов\), или в папке указанной преподавателем, создайте папку VBA, в которой будут храниться рабочие книги Microsoft Excel, создаваемые в ходе выполнения лабораторных работ по VBA.
   2) Запустите на выполнение Microsoft Excel.
   3) Вновь созданную рабочую книгу сохраните под именем VBA1 в своей рабочей папке, например, D:\course1\БУА-1\Иванов\VBA.
   4) На рабочем листе **Лист1** выберите ячейку **B2**, щелкнув по ней мышкой.
2. Запустить процедуру записи макроса. Для этого необходимо:
   1) Выполните команду **Сервис → Макрос → Начать запись**.

2) В раскрывшемся окне диалога **Запись макроса**:
   1) в поле редактирования **Имя макроса** введите текст «МояФамилия» (*обратите внимание на то, что в именах макросов пробелы не допускаются*),
   2) в раскрывающемся списке **Сохранить в** выберите пункт **Эта книга**,
   3) в поле редактирования **Описание** укажите Вашу фамилию,
   4) Щелкните по кнопке **ОК**.

   В результате будет включен макрорекордер и в строке состояния Microsoft Excel появится надпись **Запись**, говорящая о том, что все дальнейшие действия пользователя будут «протоколироваться».
3. Выполнить действия, которые должны быть записаны в макрос. Для этого необходимо:
   1) на клавиатуре набрать свою фамилию и
   2) нажать клавишу **Enter**.
4. Остановить запись макроса. Для этого необходимо выполнить команду **Сервис** → **Макрос** → **Остановить запись**. Макрос, который будет записывать в активную ячейку Вашу фамилию готов. Далее необходимо проверить его работу.

Шаг2. Выполнение макроса

1. Установите необходимый уровень безопасности выполнения макросов. Для этого:
   1) Выполните команду **Сервис** → **Макрос** → **Безопасность**.
   2) В раскрывшемся окне диалога **Безопасность** на вкладке **Уровень безопасности** установите переключатель в положение **Средняя** и щелкните по кнопке **ОК**.
2. На рабочем листе **Лист 1** выберите ячейку **D4**, щелкнув по ней мышкой.
3. Выполните команду **Сервис** → **Макрос** → **Макросы**.
4. В раскрывшемся окне диалога **Макрос**:
   1) в списке **Имя макроса** выберите пункт **Моя Фамилия**,
   2) щелкните по кнопке **Выполнить**.

   В результате в ячейке **D4** на листе **Лист 1** появится Ваша фамилия.
5. Проверьте правильность работы макроса на других рабочих листах, и на других ячейках.

Шаг3. Завершение и повторный запуск приложения

1. Сохраните рабочую книгу, выполнив команду Файл → Сохранить, или щелкнув по кнопке **Сохранить** на панели инструментов **Стандартная.**
2. Завершите работу приложения Microsoft Excel.
3. Запустите вновь на выполнение Microsoft Excel и откройте в нем ранее созданную рабочую книгу VBA1. *Если при этом появится окно сообщения с предупреждением системы безопасности, то щелкните по кнопке* **Не отключать макросы**.
4. Активизируйте панель инструментов **Visual Basic**, щелкнув правой кнопкой мыши на свободном месте в строке основного меню, или любой панели инструментов, и установите флажок против пункта **Visual Basic**.

Задание 2. Создать макрос, который будет форматировать содержимое активной ячейки шрифтом Times News Roman полужирного начертания размером 14 пунктов.

Шаг4. Запись макроса

1. Привести приложение в исходное состояние. Для этого необходимо щелкнуть мышкой по ячейке, содержащей текст, например, по ячейке **B2** на рабочем листе **Лист 1**.
2. Запустить процедуру записи макроса, щелкнув по кнопке **Запись макроса** на панели инструментов **Visual Basic** и в раскрывшемся окне диалога **Запись макроса** в качестве имени создаваемого макроса указать **Шрифт**.
3. Выполнить действия, которые должны быть записаны в макрос. Для этого необходимо установить параметры шрифта Times News Roman полужирного начертания размером 14 пунктов либо с помощью вкладки **Шрифт** окна диалога **Формат ячеек**, либо при помощи элементов управления на панели инструментов **Форматирование**.
4. Остановить запись макроса, щелкнув по кнопке **Остановить запись** на панели инструментов **Visual Basic**.

Шаг5. Выполнение макроса

1. Выделите ячейку уже содержащую текст, например, **D4** на рабочем листе **Лист1**, щелкнув по ней мышкой.
2. Запустите на выполнение созданный макрос, щелкнув по кнопке **Выполнить макрос** на панели инструментов **Visual Basic**, а в раскрывшемся окне диалога **Макрос** в списке **Имя макроса** выберите пункт **Шрифт** и щелкните по кнопке **Выполнить**, или просто выполните по нему двойной щелчок. В результате в выделенной ячейке шрифт изменится на Times News Roman полужирного начертания размером 14 пунктов.
3. Проверьте правильность работы макроса на других рабочих листах и на других ячейках.

Шаг6. Просмотр макроса

1. Выполните команду Сервис → Макрос → Макросы или щелкните по кнопке **Выполнить макрос** на панели инструментов **Visual Basic**.
2. В раскрывшемся окне диалога **Макрос** в списке **Имя макроса** выберите пункт **МояФамилия** и щелкните по кнопке **Изменить**. В результате запустится Редактор Visual Basic и в окне **Code** будет отображаться модуль **Module1** с код выбранного макроса.
3. В тетрадь для выполнения лабораторных работ перепишите текст макроса **МояФамилия** и имя модуля, в котором он находится.

Шаг7. Управление окнами Редактора Visual Basic

1. Если отображено окно **Project**, щелкните правой кнопкой мыши в его пределах и в раскрывшемся контекстном меню укажите пункт **Hide**, или щелкните по кнопке **Закрыть** в его правом верхнем углу. В результате окно **Project** будет закрыто.
2. Выполните команду **View → Project Explorer**, или щелкните по кнопке **Project Explorer** на панели инструментов **Standard**, или нажмите сочетание клавиш **Ctrl + R**. В результате откроется дочернее окно **Project**.

3. Щелкните правой кнопкой мыши в пределах окна **Project** и в раскрывшемся контекстном меню выберите пункт **Dockable**. При этом, если окно **Project** находилось в пристыкованном состоянии, оно перейдет в плавающее, и наоборот – из плавающего в пристыкованное.
4. Установите окно **Project** в пристыкованное положение.
5. Нажав, и не отпуская, левую кнопку мыши на строке заголовка окна **Project** перетащите, и пристыкуйте его, по очереди ко всем 4-м сторонам родительского окна Редактора Visual Basic.
6. Пристыкуйте окно **Project** к левому краю родительского окна Редактора Visual Basic.
7. Установите указатель мыши на границу между окнами **Project** и **Code** – при этом он примет вид двунаправленной стрелки. Нажмите левую кнопку мыши и, не отпуская ее, отрегулируйте размеры этих окон.
8. Второй макрос – **Шрифт** – был записан в **Module2**. Для его просмотра необходимо в окне **Code** отобразить **Module2**, – выполнив в окне **Project** двойной щелчок по нему, или в контекстном меню указав пункт **View Code**.
9. Завершите работу Редактора Visual Basic, щелкнув по кнопке **Закрыть** в его правом верхнем углу, или выполнив команду **File → Close and Return to Microsoft Excel**, или нажав сочетание клавиш **Alt + Q**.
10. Вновь запустите на выполнение Редактор Visual Basic, щелкнув по кнопке **Редактор Visual Basic** на панели инструментов **Visual Basic**, или выполнив команду **Сервис → Макрос → Редактор Visual Basic**, или нажав сочетание клавиш **Alt + F11**.

Задание 3. Создать процедуру, которая в окне сообщения с именем «Окно приветствия» будет выдавать приветствие – «Здравствуй, Мир!».

Шаг8. Добавление модуля и создание процедуры

1. Щелкните правой кнопкой мыши в пределах окна **Project** и в раскрывшемся контекстном меню выберите команду **Insert → Module**, или на панели инструментов **Standard** в палитре **Insert** укажите значение **Module**, или выполните команду основного меню **Insert → Module**. В результате будет добавлен модуль **Module3**.
2. Отобразите дочерние окно **Properties** Редактора Visual Basic, выполнив команду **View → Properties Window**, или щелкнув по кнопке **Properties Window** на панели инструментов **Standard**, или нажав клавишу **F4**.
3. Пристыкуйте дочерние окно **Properties** к левому краю родительского окна Редактора Visual Basic, ниже окна **Project**.
4. В дочернем окне **Project**, в ветви **Modules** выберите элемент **Module3**, щелкнув по нему левой кнопкой мыши.
5. В дочернем окне **Properties** Редактора Visual Basic выделите значение свойства **Name** – текстовое поле редактирования **Module3**, а поверх него наберите слово «Приветствие» и нажмите клавишу **Enter**, или щелкните левой кнопкой мыши где-либо на свободном месте. В результате модуль **Module3** будет переименован в **Приветствие**.
6. В дочернем окне **Code** Редактора Visual Basic отобразите содержимое модуля **Module1**, выполнив двойной щелчок по его названию в окна

Project, или в контекстном меню указав пункт **View Code**, или щелкнув по кнопке **View Code** в верхней части окна **Project**.

7. В дочернем окне **Code** Редактора Visual Basic отобразите содержимое модуля **Приветствие**.
8. В окне **Code** модуля **Приветствие** введите следующий код процедуры:

Sub Приветствие()
MsgBox "Здравствуй, Мир!", , "Окно приветствия"
End Sub

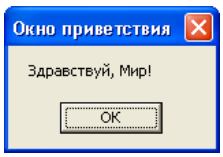
При вводе процедуры обратите внимание как Редактора Visual Basic «помогает» в создании кода:

- после ввода названия процедуры – **Приветствие** – и нажатия клавиши **Enter** будут автоматически добавлены круглые скобки и конец процедуры – **End Sub**;
- при вводе функции **MsgBox** будет отображаться ее прототип, показывающий ее параметры;
- ключевые слова, такие например, как **Sub** лучше набирать строчными буквами – если оно набрано правильно, то после того, как курсор покинет строку, оно будет выведено с прописной буквы – еще один способ контроля правильности ввода кода.

Шаг9. Запуск процедуры на выполнение

1. Установите курсор в пределах созданной процедуры **Приветствие** – т.е. щелкните левой кнопкой мыши где-либо в пределах 3-х ее строк.
2. Запустите ее на выполнение одним из способов:
   - выполните команду **Run** → **Run Sub/UserForm**, или
   - щелкните по кнопке **Run Sub/UserForm** на панели инструментов **Standard**, или
   - нажмите клавишу **F5**.

   В результате выполнения процедуры **Приветствие** будет выведено следующее окно сообщения:



3. Сохраните созданную процедуру любым из способов:
   - выполните команду **File** → **Save**,
   - щелкните по кнопке **Save** на панели инструментов **Standard**,
   - нажмите сочетание клавиш **Ctrl + S**.

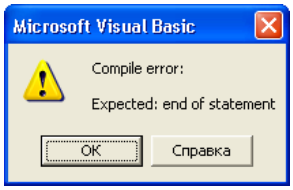
   В результате она будет сохранена в текущей рабочей книге – **VBA1.XLS**.

Шаг10. Обработка ошибок

1. Отредактируйте процедуру **Приветствие**, удалив двойные кавычки вокруг 3-го параметра – «Окно приветствия». В результате ее код будет следующим:

Sub Приветствие()
MsgBox "Здравствуй, Мир!", , Окно приветствия
End Sub

2. Запустите на выполнение процедуру **Приветствие**. В результате будет выдано окно сообщения с описанием ошибки:



Согласно этому сообщению Редактора Visual Basic обнаружил ошибку трансляции (**Compile error**) – **Ожидается: конец инструкции** (**Expected: end of statement**). Если при этом щелкнуть кнопку **Справка** – будет выведена справочная информация по данной ошибке.

3. Щелкните по кнопке **OK**. В результате произойдет возврат в окно документа **Code**.
4. Не исправляя ошибку, снова запустите на выполнение процедуру **Приветствие**. В результате снова будет выдано окно сообщения. Но, в данном случае информация об ошибке менее конкретна – локализована только строка, содержащая ошибку, и сообщается, что **ошибка синтаксическая**.
5. Щелкните по кнопке **OK**. Процедура **Приветствие** будет запущена на выполнение в пошаговом режиме – подсветится желтым цветом ее первая строка **Sub Приветствие()** – и против нее будет установлена стрелка.
6. Щелкните по кнопке **Reset** на панели инструментов **Standard**, или выполните команду **Run** → **Reset**. Выполнение процедуры **Приветствие** будет остановлено, и ее снова можно будет редактировать.
7. Отредактируйте процедуру **Приветствие**, восстановив двойные кавычки вокруг 3 - го аргумента, но удалив одну из двух подряд идущих запятых. В результате ее код будет выглядеть следующим образом:

Sub Приветствие()
MsgBox "Здравствуй, Мир!", "Окно приветствия"
End Sub

8. Запустите на выполнение процедуру **Приветствие**. Будет выдано необходимое сообщение об ошибке, которое гласит о том, что произошла **ошибка времени выполнения** 13 (**Run-time error '13'**) – недопустимое смешивание (или несоответствие) типов (**Type mismatch**). Если при этом щелкнуть по кнопке **End**, выполнение процедуры **Приветствие** будет завершено.
9. Щелкните по кнопке **Debug**. Выполнение процедуру **Приветствие** будет переведено в пошаговый режим.
10. Остановите выполнение процедуры **Приветствие**, щелкнув по кнопке **Reset** на панели инструментов **Standard**, или выполнив команду **Run** → **Reset**.
11. Отредактируйте процедуру **Приветствие**, исправив в ней все ошибки, и снова запустите на выполнение.
12. Сохраните отлаженную процедуру **Приветствие** в текущей рабочей книге – **VBA1.XLS**.
13. Перепишите в тетрадь для выполнения лабораторных работ исходный код процедуры **Приветствие** и имя модуля, в котором она записана.

Индивидуальные задания

1) В текущей рабочей книге – **VBA1.XLS**, добавить модуль **Макросы**, а в нем написать процедуру **Моя Фамилия**, которая в окне сообщения с именем «Моя фамилия» будет отображать Вашу фамилию.
2) Определить свой вариант индивидуального задания:
   - как номеру по списку группы, или
   - как остаток целочисленного деления двух последних цифр номера зачетной книжки (студенческого билета) на 30; например, две последние цифры зачетной книжки образуют число 75 (18), тогда номер варианта будет равен: 75 Mod 30 = 15 (18 Mod 30 = 18), или
   - в соответствии с указанием преподавателя.
3) Согласно Вашему варианту индивидуального задания в текущей рабочей книге создать макрос с именем Ваша фамилия, а в поле **Описание** – записать (скопировать) условия индивидуального задания, который:
   1. Записывает в активную ячейку формулу вычисления площади прямоугольника, если его высота хранится в ячейке A1, а ширина – в B1.
   2. Записывает в активную ячейку формулу вычисления площади прямоугольного треугольника, высота и ширина которого хранятся в ячейках A1 и B1, соответственно.
   3. Записывает в активную ячейку формулу =A1 + A2.
   4. Записывает в активную ячейку формулу вычисления площади круга, если радиус хранится в ячейке A1.
   5. Создает новую рабочую книгу.
   6. Добавляет в текущую рабочую книгу лист.
   7. Устанавливает в активной ячейке шрифт размером в 14 пунктов.
   8. Устанавливает в активной ячейке полужирное начертание шрифта.
   9. Устанавливает в активной ячейке курсивное начертание шрифта.
   10. Устанавливает в активной ячейке шрифт Arial.
   11. Записывает в активную ячейку Ваше имя.
   12. Устанавливает вокруг активной ячейки широкую рамку.
   13. Открывает рабочую книгу.
   14. Делает активным рабочий лист.
   15. Делает активной указанную ячейку.
   16. Выравнивает содержимое активной ячейки по центру.
   17. Устанавливает в активной ячейке красный цвет шрифта.
   18. Устанавливает в активной ячейке в зелёный цвет заливки.
   19. Устанавливает имя для рабочего листа.
   20. Объединяет две смежные ячейки.
   21. Копирует содержимое ячейки в активную.
   22. Устанавливает в активной ячейке Финансовый формат.
   23. Устанавливает в активной ячейке произвольный денежный формат.
   24. Записывает в активную ячейку символ авторского права ©.
   25. Записывает в активную ячейку символ Товарный знак ™.
   26. Вставляет пустую строку перед активной ячейкой.
   27. Вставляет пустой столбец перед активной ячейкой.

28. Удаляет строку в позиции активной ячейки.
29. Удаляет столбец в позиции активной ячейки.
30. Присваивает активной ячейке имя.

4) В тетрадь для выполнения лабораторных работ занести созданные ранее процедуру и макрос.

Контрольные вопросы

1. Что такое макрос? Для чего он используется?
2. Перечислите шаги, необходимые для создания макроса.
3. Как запустить программу записи макросов – макрорекордер?
4. Где Microsoft Excel хранит записанные макросы?
5. Как в программе Microsoft Excel выполнить записанный ранее макрос?
6. Где необходимо сохранить макрос, чтобы он был доступен для всех открытых рабочих книг, в любом сеансе работы?
7. Как наиболее быстро и наиболее удобно создавать и выполнять макросы?
8. Для чего служит Редактор Visual Basic?
9. Как запустить на выполнение Редактор Visual Basic?
10. Какой тип интерфейса имеет Редактор Visual Basic?
11. Перечислите основные дочерние окна Редактора Visual Basic, для чего служит каждое из них?
12. Как закрыть или, наоборот, отобразить любое из дочерних окон Редактора Visual Basic?
13. Сколько экземпляров дочерних окон **Code**, **Project** и **Properties** может отображаться в Редакторе Visual Basic?
14. Как перевести окно **Project** или **Properties** из пристыкованного состояния в плавающее, и наоборот?
15. Как пристыковать окно **Project** или **Properties** к одной из сторон родительского окна Редактора Visual Basic?
16. Как отрегулировать размеры дочерних окон Редактора Visual Basic?
17. Как добавить модуль в проект?
18. Как переименовать модуль?
19. Из чего состоит любая компьютерная программа?
20. Что понимается под терминами «макрос» и «процедура»?
21. Перечислите основные составляющие процедуры.
22. Может ли имя процедуры содержать пробелы?
23. Как Редактора Visual Basic «помогает» в создании кода?
24. Каким цветом выделяются различные составляющие процедуры?
25. Для чего служат комментарии? Как их вставить в текст макроса, процедуры?
26. Для чего используются отступы при создании процедур?
27. Как запустить на выполнение процедуру?
28. Как остановить выполнение процедуры, запущенной в пошаговом режиме?
29. Какие бывают категории ошибок в процедурах VBA?
30. О чем говорят синтаксические ошибки?
31. О чем говорят семантические ошибки?
32. Что такое ключевое слово языка VBA? Приведите примеры.
33. Для чего предназначена функция MsgBox? Перечислите и объясните назначение ее параметров.

Лабораторная работа № 8

Линейный вычислительный процесс

Изучим: типы данных, переменные, константы, операторы и выражения, стандартные функции VBA, обработка ошибок, отладка программы, окно отладки.

Познакомимся с типами данных

Тип данных – это термин, описывающий природу данных, которые VBA различает и умеет с ними обращаться. В таблице перечислены типы данных VBA и дано краткое описание типов и диапазон значений для каждого типа.

Таблица. Типы данных языка Visual Basic

Имя типа	***Размер в байтах***	***Описание и диапазон значений данных***
Byte	1 (8 бит)	Положительные числа от 0 до 255
Boolean	2 (16 бит)	Логические значения. Только значения True и False
Currency	8 (64 бит)	От -9223372036854775808 до 9223372036854775807
Date	8 (64 бит)	Хранит комбинацию даты и времени. Дата от 1 января 100 г. до 31 декабря 9999 г. Время от 00:00:00 до 23:59:59
Double	8 (64 бит)	Отрицательные числа от - *1.79* * 10^{308} *по* - 4.9 *10^{-324}. Положительные числа от 4.9 * 10^{-324} до 1.79 * 10^{308}
Integer	2 (16 бит)	Числа от -32768 до 32767
Long	4 (32 бит)	Числа от-2147483648 до 2147483647
Object	4 (32 бит)	Используется для доступа к любому объекту VBA. Хранит адрес объекта
Single	4 (32 бит)	Отрицательные числа от -3.4*10^{38} до -1.4*10^{45} Положительные числа от 1.4*10^{45} до 3.4*10^{38}
String	1 байт на символ	Служит для хранения текста. Может содержать до 2 миллионов символов. Строки в программе заключаются в двойные кавычки.
Variant	16 байт + 1 байт	Служит для хранения данных любого типа

Вы можете преобразовывать большинство типов данных из одного вида в другой, кроме того, Visual Basic автоматически преобразует данные.

Данные типа Variant

Variant – это специальный тип данных, в котором можно хранить все типы данных, перечисленные в таблице, включая типы Object и Array. VBA использует тип данных Variant всегда, когда вы не указываете тип явно.

Данные типа Variant имеют свойства тех данных, которые фактически в них записаны. Например, если в данных типа Variant записана текстовая строка, они имеют свойства данных типа String, а если там записано число, то они имеют свойства числового типа, чаще всего типа Double, хотя это может быть и Integer, и Long, и Single, и Currency.

Данные типа Variant используют самое компактное из возможных представлений, например, если в данных типа Variant записано целое число, то тот эти данные будут трактоваться как Integer или как Long, в зависимости от размера этого числа. Число 15 в переменной типа Variant будет трактоваться как Integer, а число 1000000 – как Long.

VBA хранит числа с плавающей запятой в переменных типа Variant как Double и обращается с ними так, что вы не обязаны всегда заботиться о

точном типе переменных в своих программах. Любая переменная, для которой вы не указали тип явно, становится переменной типа Variant.

Хотя использование переменных типа Variant удобно и освобождает вас от многих забот при написании процедур, такие переменные требуют больше памяти и большинство операций выполняется на них медленнее, чем с другими типами. Вообще говоря, вам следует избегать неоправданного использования переменных типа Variant, так как это может сделать ваши программы медленными и затруднить их чтение, кроме того, это может привести к появлению трудно обнаруживаемых ошибок.

Понятие о переменных

Переменная – это имя, которое вы, программист, дали участку компьютерной памяти для того, чтобы хранить в нем данные некоторого типа. Вы можете представить себе переменную как гнездо, в котором записано какое-то значение для дальнейшего использования. Имя переменной – это ярлык, по которому осуществляется доступ к участку памяти. Содержимое участка (значение переменной) может меняться, но имя остается неизменным. В переменной VBA могут храниться данные любого из типов, перечисленных в таблице.

В некотором смысле переменная подобна имени ячейки в листе Excel. Вы можете дать ячейке листа Excel имя и потом обращаться к этой ячейке по данному имени, не заботясь о том, какие номера столбца и строки имеет ячейка.

Имена переменных должны подчиняться следующим правилам.

- Имя переменной начинается с буквы.
- За буквой может следовать любая комбинация букв, цифр и символов подчеркивания.
- Имя переменной не должно содержать пробелов, точек и символов математических операций.
- Имя переменной не должно превышать 255 символов в длину.
- Имя переменной не должно совпадать ни с одним из ключевых слов (называемых также зарезервированными словами). Если такое совпадение будет иметь место, VBA выдаст сообщение о синтаксической ошибке.
- Имя переменной должно быть уникальным в процедуре. Если вы дадите двум переменным одинаковые имена или имя переменной совпадет с именем процедуры в том же самом модуле, при выполнении процедуры VBA выдаст сообщение об ошибке.

Вот примеры правильных имен переменных.

MyVar
PayDate
New_Item
Percent
Whole
Part
Linel2

А это примеры неправильных имен переменных.

New Item	Не разрешен символ пробела
1SthDimention	Начинается с цифры
Dim	Совпадает с зарезервированным словом
Week/Day	Содержит знак арифметической операции

Выбирая имена переменных, старайтесь делать их описательными. Гораздо лучше дать переменной имя Percent, чем X или Y. Например, если в переменной записывается температура в градусах по Цельсию, назовите переменную CelsiusTemp или DegreesC.

Создание переменных

Самый простой способ создать переменную – это просто упомянуть ее в выражении. VBA создает переменную и выделяет для нее память, как только обнаружит ее, обычно это происходит при записи в переменную некоторого значения.

Запись значения в переменную называется присваиванием значения переменной. Вы присваиваете значение переменной, используя оператор присваивания, который выглядит как знак равенства (=). Вот пример оператора присваивания.

MyVar=15

Этот оператор записывает число 15 в участок памяти, с которым связано имя переменной MyVar. Если это первое упоминание имени переменной, то VBA создает переменную, выделят для нее память и записывает в нее число 15.

Если такая переменная была создана ранее, то оператор присваивания просто записывает в участок памяти, на который указывает переменная, число 15, при этом все, что было записано в эту память ранее, безвозвратно стирается.

Такой способ создания переменной простым ее упоминанием называется неявным объявлением переменной. Упоминая переменную в выражении, вы косвенно говорите программе VBA о том, что хотите создать такую переменную. Все переменные, созданные таким способом, получают тип Variant. Неявное создание переменных называют также созданием переменных "на лету" (on - the-fly declaration).

По некоторым причинам VBA предоставляет вам способ объявлять переменные явно, что избавит вас от возможных ошибок. Явное объявление переменных имеет много преимуществ:

- При явном объявлении переменных ваша программа работает заметно быстрее. VBA создает все переменные перед тем, как программа начнет работать, и на это не будет уходить время при работе программы.
- Явное объявление переменных помогает избежать многих ошибок, которые подстерегают вас при неявном объявлении.
- Явное объявление переменных облегчает чтение программы человеку. Видя в начале процедуры или модуля список всех переменных, человеку легче понять логику работы программы.
- При явном объявлении переменных вам не нужно следить за правильным использованием верхнего и нижнего регистров, так как за вас это будет делать VBA. Вместо того чтобы подгонять все вхождения имени переменной под

последнее ее упоминание, VBA будет исправлять регистр имени в соответствии с тем написанием, которое было использовано при явном объявлении.

Для того чтобы объявить переменную, используется инструкция Dim со следующим синтаксисом:

Dim имя

Имя – это любой идентификатор. Например:

Dim PcntProfit

Такая конструкция создает переменную с именем PcntProfit. (Ключевое слово Dim является сокращением от dimension – размерность.) Все переменные, которые создаются этим способом, имеют тип Variant. Имена нескольких переменных могут быть перечислены через запятую.

Объявлять переменную можно только один раз. Это может быть сделано в любом месте программы, но обязательно до первого упоминания переменной в процедуре. Однако хорошей практикой следует считать вынесение всех объявлений в один блок в начале программы.

В листинге приведена процедура, преобразованная так, что переменная HelloMsg объявляется явно. Процедура Hello выводит сообщение

Процедура Hello с явным объявлением переменных

```
1:  Sub Hello()
2:  Dim HelloMsg   'текст сообщения для функции MsgBox
3:  HelloMsg = "Hello, World!"
4:  MsgBox HelloMsg, , "Приветствие"
5:  End Sub
```

Указание типа переменной в инструкции Dim

Для того чтобы объявить тип переменной в инструкции Dim, добавьте за именем переменной ключевое слово As и укажите тип переменной. Синтаксис этой конструкции следующий.

Dim имя As тип

где имя в данном случае – идентификатор переменной, а тип – один из типов переменных, приведенных в таблице.

В следующих строках приведены примеры правильного объявления типа переменных.

Dim PcntProfit As Single
Dim Gross_Sales As Currency
Dim PayDay As Date
Dim Massage As String
Dim Counter As Integer

Объявляя переменную неявно, вы можете указать ее тип, добавляя в конце имени переменной специальный символ. Такие символы называются символами определения типа. В таблице перечислены символы определения типа и соответствующие им типы.

Символы определения типа

!	*Single*	*@*	*Currency*
#	*Double*	*$*	*String*
%	*Integer*	*&*	*Long*

Обратите внимание на то, что таких символов всего шесть. Не существует символов определения типа для таких типов, как Byte, Boolean, Date, Object или Array. Символы определения типа могут находиться только в конце имени переменной.

Понятие о константах

Константа – это такая величина в программе, которая никогда не меняется. Константы могут быть поименованными и непоименованными. В общем случае следует использовать поименованную константу всегда, когда в программе встречается повторяющееся значение, или значение, которое трудно запомнить, или если само значение на момент написания программы неизвестно.

Имя константы подчиняется тем же правилам, что и имя переменной. Правила создания имен переменных мы рассмотрели выше в этой главе.

Как и в случае с переменной, константа должна быть объявлена до того, как вы первый раз к ней обратитесь. Но в отличие от переменной, константа всегда объявляется явно, и служит для этого ключевое слово Const. Вот синтаксис объявления поименованной константы.

Const имя = значение

Значение может быть любого типа – числом, строкой или датой. Приведем несколько примеров правильного объявления констант.

Const BillingPoint = 212

Const Цена =8.25

Const Приветствие ="Привет, коллеги!"

При желании можно объявить несколько констант в одной строке, разделяя их запятыми. Следующая строка полностью эквивалентна предыдущим трем.

Const BillingPoint = 212, Цена = 8.25, Приветствие = "Привет, коллеги!"

Вот пример правильного объявления констант.

Const BillingPoint = 212

Const DangerZone = BillingPoint+50

Объявлять переменную можно только один раз. Это может быть сделано в любом месте программы, но обязательно до первого упоминания переменной в процедуре. Однако хорошей практикой следует считать вынесение всех объявлений в один блок в начале программы.

Инструкция Dim появляется во второй строке листинга программы Hello. Когда VBA выполняет эту строку, он создает новую переменную и выделяет для нее память. (Переменная HelloMsg имеет тип Variant, поскольку специально ее тип не указан.) В этой же строке вы видите комментарии, объясняющие назначение этой переменной.

В строке 3 листинга присваивается значение переменной HelloMsg. В этой переменной содержится текст "Hello, World!". В следующей строке листинга переменная HelloMsg передается процедуре MsgBox, которая выводит на экран окно сообщения.

Когда вы создаете поименованную константу, VBA приписывает ей тип в соответствии с типом выражения, которое ей присваивается. Например, считается, что константа, содержащая строку, имеет тип String.

Но иногда вы можете указать тип константы явно. Это может понадобиться для повышения точности вычислений: константа, тип которой указан как Double, использует более широкий диапазон значений. Вы можете указать тип константы Integer, Long, Currency или другой для того, чтобы результат вычислений имел, в свою очередь, нужный тип.

Константа может иметь любой из типов, перечисленных для переменных, кроме Object и Array. Тип константы объявляется точно так же, как и тип переменной, просто объявление начинается со слова Const. Синтаксис объявления типа константы выглядит следующим образом.

Const имя As Тип = значение

Здесь имя – это идентификатор константы, тип – один из возможных VBA-типов, значение – это то значение, которое вы присваиваете константе. Вот пример правильного объявления константы.

Const Pi As Double = 3.14

Здесь объявлена константа Pi, имеющая тип Double, и ей присвоено значение 3.14.

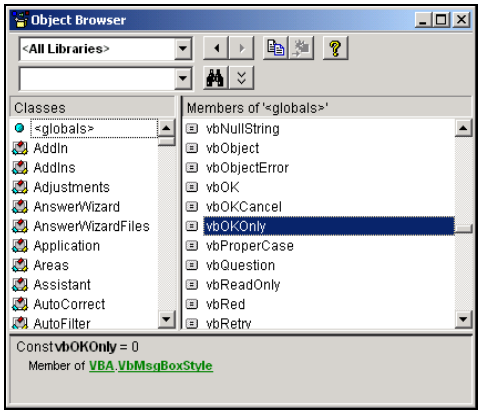
В VBA есть несколько констант, значение которых заранее определено для вас. Такие константы называют еще внутренними. Кроме внутренних констант VBA, существует несколько констант, определяемых приложением, из которого вызывается VBA. В частности, Excel 2000 определяет несколько констант для работы с книгами Excel, MS Word определяет константы для работы с документами и шаблонами, MS Access имеет несколько констант для работы с базами данных.

Все внутренние константы, определенные в VBA, имеют имя, начинающееся с букв vb. Например, такие константы, как vbOKOnly, vbOKCancel, vbAbortRetrylgnore определены в языке Visual Basic. В приложении Excel определены, например, такие константы: xlChart, xlCountrySetting, xlWorksheet. Как видите, их имена начинаются с букв xl.

Внутренние константы служат для упрощения работы с некоторыми встроенными процедурами VBA, такими как уже знакомая вам MsgBox или InputBox, с которой вы познакомитесь позже. Процедура MsgBox имеет необязательный аргумент, указывающий количество кнопок в окне. Аргументы, соответствующие кнопкам, обычно задаются с помощью внутренних констант VBA.

Для того чтобы просмотреть полный список предопределенных констант как VBA, так и главного приложения, необходимо воспользоваться окном **Object Browser** (**Просмотр объектов**), которое активизируется:

- по команде **View** → **Object Browser**,
- по нажатию кнопки  **Object Browser** на панели инструментов **Standard**, или
- по нажатию клавиши **F2**.



Ввод данных пользователя в процедуру

Вы уже научились пользоваться процедурой MsgBox для вывода на экран сообщений пользователю. Кроме того, вы познакомились с переменными и константами и теперь готовы к тому, чтобы научиться получать информацию от пользователя.

Получение данных, сохранение их в переменной, обработка этих данных или другой информации на основе этих данных – вот главные составляющие умения писать интерактивные программы. (Интерактивной называется программа, которая интенсивно общается с пользователем, выводя на экран информацию и получая информацию от пользователя.) Такие программы часто оказываются более эффективными, чем просто записанные макросы, так как пользователь может управлять их работой в зависимости от изменившихся условий.

Данные, полученные от пользователя, называются введенными данными, или просто вводом. Для получения данных от пользователя процедуры применяется функция InputBox. (Функция – это особый вид процедуры, которая возвращает некоторое значение.) Функция InputBox выводит на экран окно с текстом, предлагающим ввести некоторое значение, и полем ввода для этого значения. Кроме того, в окне функции InputBox присутствуют кнопки ОК и Cancel.

Перед тем как вызывать функцию InputBox, вы должны сформировать текст приглашения. Кроме того, вы можете задать заголовок окна, которое выведет на экран InputBox.

Синтаксис вызова InputBox выглядит следующим образом.

Строка = InputBox (Приглашение [, Заголовок])

где строка – это любая переменная, в которой может храниться текстовая строка; может иметь тип String или Variant. Приглашение – поименованная или непоименованная константа или переменная, содержащая текст. Это обязательный аргумент, вы должны указывать его в вызове функции InputBox.

Заголовок – это необязательный второй аргумент функции InputBox. (Все необязательные элементы в описании синтаксиса берутся в квадратные скобки.) Заголовком также может быть поименованная или непоименованная константа или переменная, содержащая текст. Если вы опустите этот аргумент в вызове функции, в заголовке будет выведена строка Microsoft Excel. В листинге, приведенном ниже, вы видите модуль, содержащий единственную процедуру, и константу и переменную, объявленные на уровне

модуля. Эта процедура вычисляет площадь круга. Эта процедура спрашивает у пользователя, для какого радиуса следует вычислять площадь.

Получение ввода пользователя с помощью функции InputBox

```
Dim CircleArea As Single
Const Pi As Single = 3.14
Const BoxTitle = "Площадь круга"
Sub Calc_CircleArea()
Dim Radius As Single, Temp As String
Temp = InputBox("Введите радиус круга", "BoxTitle")
Radius = CSng(Temp)
CircleArea = Pi * (Radius * Radius)
MsgBox CircleArea, , "BoxTitle"
End Sub
```

Процедура называется Culc_CircleArea. В процедуре объявляются *константы*: Pi и BoxTitle, которая используется для вывода заголовков в окна, создаваемые этой процедурой, и *переменные* **CircleArea**, **Radius, Temp**.

Переменная **CircleArea** и константы Pi и BoxTitle объявляются за пределами процедуры, т.е. будут доступными всем процедурам данного модуля.

Обратите внимание на строку 6. В этой строке вызывается функция InputBox, которая выводит свой первый аргумент в качестве приглашения, поясняя пользователю, чего от него ждут. В данном случае выводится строка “Введите радиус круга”. Второй аргумент функции – это заголовок окна; в данном случае для заголовка окна используется переменная BoxTitle.

После вызова функции, возвращаемое ею значение присваивается переменной Temp; результат функции InputBox– это всегда строка, поэтому переменная Temp была объявлена с типом String. Если пользователь введет цифру и щелкнет ОК, в переменной Temp окажется строка, введенная в окне сообщения. Перед тем, как использовать результат в математических вычислениях, нужно преобразовать его в число. Это делается в следующей строке.

В строке 8 вычисляется площадь круга, а в строке 9 – выводится на экран вычисленное значение.

1. Запустите Excel и откройте свой файл.
2. Войдите в редактор VBA и отредактируйте записанную Вами процедуру Hello, введя явное описание переменных.
3. Запустите на выполнение процедуру Hello. Изменился ли результат выполнения процедуры?
4. Напишите программу, которая будет запрашивать значение переменной с экрана, считывать это значение и выводить его опять на экран вместе с именем переменной.

Понятие об операторах и выражениях

Выражение – это величина или группа величин, объединенных в единую сущность. Каждое выражение имеет определенное и единственное значение. В состав выражения могут входить: константы, переменные любого типа, операторы, массивы, элементы массивов, функции. Выражение может иметь одно из особых значений – Empty (пусто) и Null. Для обозначения этих особых

значений в языке VBA существуют соответствующие зарезервированные слова. Значение Empty соответствует неинициализированной переменной типа Variant или выражению, в которое входит такая переменная. Значение Null соответствует выражению, содержащему неверные данные, т.е. неопределенному выражению. Ниже приведено несколько примеров выражений.

Выражение	Пояснение
5	Значением является число 5
"5"	Значение – строка, состоящая из единственного символа 5
"сего"&"дня"	Значение – строка «сегодня»
C*(p/100)	Значение – число, полученное после умножения значения переменной C на значение, полученное после деления переменной P на константу 100
CStr(UOO)	Значение равно результату, возвращаемому функцией CStr. В данном случае это строка 1200 (функция CStr преобразует числа в строки)
MyValue <= 7	Логическое выражение. Его значение зависит от того, действительно ли значение переменной MyValue меньше или равно числу 7

Для объединения, сравнения или иного манипулирования значениями, входящими в выражение, используются операторы. Величины, входящие в выражение (это могут быть переменные или константы), называются операндами; большинство операторов требует наличия двух операндов. Например, в выражении 2+1 числа 2 и 1 – операнды, объединенные оператором сложения. В выражении может быть любое количество операторов.

Выражения могут входить в состав других выражений, сравниваться операторами сравнения или передаваться функциям и процедурам в качестве аргументов. Каждое выражение имеет единственное значение определенного типа. Выражения языка VBA в некотором смысле подобны предложениям человеческого языка. Каждая инструкция может содержать сколько угодно выражений. Рассмотрим типы выражений, которые встречаются в языке Visual Basic.

- Дата. Это любое выражение, которое имеет тип Date. Такие выражения могут содержать константы даты, переменные числового типа или типа Date, числовые константы и даты, возвращаемые функциями.
- Числовое выражение. Это выражение одного из следующих типов: Byte, Integer, Long, Single, Double или Currency. В числовое выражение могут входить числовые переменные и константы, функции, возвращающие числа, и знаки арифметических операций. Кроме того, эти выражения могут включать строки, если их можно преобразовать в число. (Строку можно преобразовать в число, если в нее входят только цифры; например, строку "36" можно преобразовать в число 36.)
- Строковые выражения. Это выражения имеющие тип String. Такое выражение может включать строковые константы и переменные, функции, возвращающие строки, и операторы конкатенации строк. В строковые выражения могут также входить числовые выражения, если их можно преобразовать в строки.

- Логические выражения. Это выражение, которое имеет одно из булевых значений, т.е. True или False. Такие выражения могут состоять из логических переменных и констант, функций, которые возвращают логическое значение, операторов сравнения и логических операторов.
- Объектное выражение. Это выражение, значением которого является ссылка на объект.

Не все типы данных совместимы друг с другом. Нельзя комбинировать несовместимые типы в одном выражении. Например, нельзя складывать текстовую строку "Привет!" с числом – такая операция не имеет смысла, и ее значение не определено.

В то же время некоторые типы являются совместимыми. Вы можете свободно комбинировать различные числовые типы между собой, VBA автоматически выполнит необходимые преобразования. Иногда VBA может автоматически преобразовывать и другие типы данных, добиваясь их совместимости, но не всегда.

Очень важно знать типы входящих в выражение значений, так как если в выражении встретятся несовместимые данные, то при выполнении программы VBA выдаст вам сообщение о несоответствии типов. Когда VBA встречает в выражении различные типы данных, он сначала пытается уладить конфликт, автоматически преобразуя данные к совместимым типам. Но если это не удается, вы получаете сообщение о несоответствии типов. То же самое произойдет, если вы попытаетесь присвоить переменной значение выражения, тип которого несовместим с типом переменной. Сообщение о несоответствии типов не выдается, если один из операндов имеет тип Variant. Как мы уже говорили, проще всего преобразование типов выполняется тогда, когда переменная имеет тип Variant, но не следует этим злоупотреблять.

Во многих процедурах вы можете видеть такую конструкцию.

Count = Count + 1

GrossTotal = GrossTotal + SubTotal

Это часто применяется для накопления значения какого-нибудь счетчика. Смысл написанного не сразу очевиден; на первый взгляд даже может показаться, что это противоречит логике. Но вы поймете написанное, если вспомните, что при выполнении оператора присваивания сначала вычисляется выражение, стоящее справа от оператора, а потом полученное значение присваивается переменной, стоящей слева.

Оператор присваивания (=) служит для того, чтобы записать значение выражения в переменную или значение одной переменной в другую. Оператор присваивания имеет две синтаксические формы, которые обе законны и совершенно эквивалентны. В первом случае используется ключевое слово Let и синтаксис имеет следующий вид.

Let имя = выражение

где имя – идентификатор переменной, а выражение – любое допустимое в VBA выражение. Такая форма оператора присваивания пришла из старого языка Basic. Вот пример использования оператора присваивания.

Let X = Y 'Значение У записывается в переменную X

Вторая форма более общепринята и используется повсеместно. Синтаксис ее таков:

имя = выражение

где имя – идентификатор переменной, а выражение – любое допустимое в VBA выражение. Вот пример использования оператора присваивания в этой простой и общепринятой форме.

X = Y

MyVar = 12 - YourVar

В обеих формах оператора присваивания переменная слева от него получает значение выражения справа. При выполнении оператора присваивания сначала вычисляется выражение, стоящее справа от оператора, а потом полученное значение присваивается переменной, стоящей слева.

Арифметические операторы

VBA может выполнять все стандартные арифметические действия: сложение, вычитание, умножение и деление. Кроме того, есть оператор для возведения числа в степень и дополнительные операторы для целочисленного деления и для деления по модулю. В таблице собраны арифметические операторы VBA.

Оператор	*Синтаксис*	*Пояснение*
+	N1 + N2	Складывает N1 И N2
-	N1 - N2	Вычитает N2 из N1
–	–N1	Унарный минус (отрицательное число)
*	N1 *N2	Умножает N1 на N2
/	N1 / N2	Делит N1 на N2
\	N1 \ N2	Целочисленное деление. Делит N1 на N2, отбрасывая дробную часть
Mod	Nl Mod N2	Деление по модулю. Делит N1 на N2, возвращая остаток от деления
^	N1 ^ N2	Возводит N1 в степень N2

5. Запишите программу, выполняющую вычисление площади круга с вводом радиуса с экрана и выводом результата на экран.
6. Запустите программу и выполните ее в пошаговом режиме.

Операторы сравнения

Операторы сравнения иногда называют операторами отношения. Обычно их применяют для задания критерия на принятие решения или для формирования условия выполнения некоторой последовательности операций. Результат оператора сравнения имеет тип Boolean и принимает только два значения – True и False. Сравнивать можно поименованные и непоименованные константы и переменные сходных типов.

Таблица. Операторы сравнения

Оператор	*Синтаксис*	*Описание*
=	E1 = E2	Равно. True, если E1 равно E2, False в противном случае
<	E1 < E2	Меньше. True, если E1 меньше E2, False в противном случае

Продолжение табл.

<=	E1 <= E2	True, если E1 меньше или равно E2, False в противном случае
>	E1 > E2	True, если E1 больше E2, False в противном случае
>=	E1 >= E2	True, если E1 больше или равно E2, False в противном случае
<>	E1 <> E2	Не равно. True, если E1 не равно E2, False в противном случае
Is	El Is E2	Является. Оба операнда должны иметь тип Object. True, если E1 и E2 указывают на один и тот же объект, False противном случае
Like	El Like E2	Подобно. Оба операнда должны иметь тип String. True, если El соответствует образцу, содержащемуся в E2, False в противном случае

Логические операторы

Обычно логические операторы применяются для того, чтобы, комбинируя условные выражения, создавать сложные условия для принятия решений в процедуре, или для формирования условий выполнения некоторой группы инструкций.

В качестве операнда в логическом операторе может использоваться любое значение типа Boolean или число, которое можно преобразовать к этому типу. VBA считает, что 0 соответствует значению False, а любое другое число – значению True.

Таблица Логические операторы

Оператор	*Синтаксис*	*Объяснение*
And	El And E2	Конъюнкция. Выражение истинно, если истинны оба его операнда
Or	El Or E2	Дизъюнкция. Выражение истинно, если истинен хотя бы один его операнд
Not	Not El	Отрицание. Выражение истинно, если E1 ложно, и наоборот

Таблица истинности оператора And

Операнд1	*Операнд2*	*Значение выражения*
True	False	False
False	True	False
False	False	False

для того, чтобы проверить, являются ли оба условия истиной одновременно. Например, рассмотрим оператор And в следующем выражении.

(Gross_Sales < 50000) And (Net_Profit < 10000)

Это выражение имеет значение True, если значение переменной Gross_Sales меньше 50000, а значение переменной Net_Profit меньше 10000.

Таблица истинности оператора Or.

Операнд1	*Операнд2*	*Значение выражения*
True	True	True
True	False	True
False	True	True
False	False	False

Оператор Or используется для того, чтобы проверить, является ли хотя бы одно из условий истиной. Например, рассмотрим оператор Or в следующем выражении.

(Grossjales < 50000) Or (Net_Profit < 10000)

Это выражение имеет значение True, если значение переменной Gross_Sales меньше 50000 или значение переменной Net_Profit меньше 10000.

Оператор Not называется логическим *отрицанием*. Он инвертирует любое значение, к которому применяется. Его значением будет True, если значение операнда – False, и наоборот.

Синтаксис оператора Not имеет следующий вид.

Not Операнд1

Операнд1 – это любое логическое выражение языка VBA.

Основной оператор конкатенации: "&"

Знак амперсанда имеет единственное применение в языке VBA – для слияния строк. Никаких других функций этот знак не выполняет. Синтаксис оператора конкатенации имеет следующий вид.

Операнд1 & Операнд2

Операнд1 и Операнд2 – это любые выражения строкового или числового типа, VBA String. Если один из операндов имеет значение Null или Empty, VBA трактует его как строку нулевой длины, т.е. строку, не содержащую символов.

Приоритет операций

Приоритет операторов указывает порядок их обработки в сложном выражении. Если в выражении используется более одного оператора, то все они выполняются в строго определенном порядке, что обеспечивает однозначное трактование значения этого выражения.

В таблице, приведенной далее, представлены все операторы VBA в порядке уменьшения их приоритетов.

Приоритет	Операция	Описание
1	()	Круглые скобки
2	^	Возведение в степень
3	-	Унарный минус
4	*, /	Умножение и деление
5	\	Целочисленное деление
6	Mod	Деление по модулю
7	+, -	Сложение и вычитание
8	&	Конкатенация
9	>, <, >=, <=, =, <>, Is, Like	Операторы сравнения
10	Not	НЕ
11	And	И
12	Or	ИЛИ
13	Xor	Исключающее ИЛИ
14	Eqv	Эквивалентность
15	Imp	Импликация
16	=	Присваивание

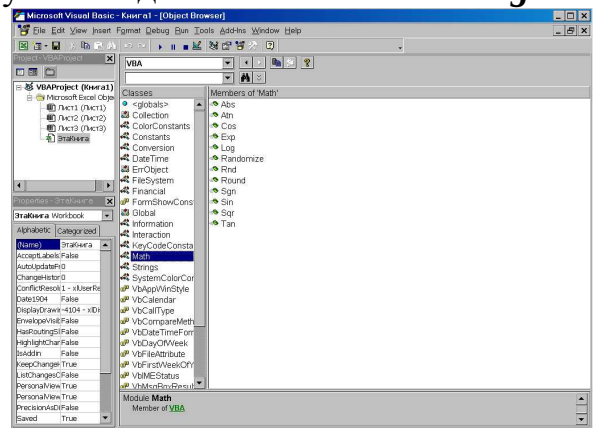
7. Запишите программу согласно вашему варианту. Запустите программу на выполнение.
8. Отладьте программу. Откройте окно наблюдения переменных и выполните программу в пошаговом режиме. Составьте в тетради таблицу изменений значений переменных. Покажите результат преподавателю.

Индивидуальные задания

№	Задание	№	Задание
1	$Y = a \cdot x + b;$	2	$Y = \cos(a) \cdot x + b;$
3	$Y = \sin(a) \cdot x + b / c;$	4	$Y = tg(a) \cdot x + b;$
5	$Y = a / x + b;$	6	$Y = (a + b) / x;$
7	$Y = tg(a) + b / c;$	8	$Y = \lvert a - b \rvert / x;$
9	$Y = \cos(a) + x / b;$	10	$Y = \lvert c - b \rvert / x;$
11	$Y = \cos(a) + b;$	12	$Y = \sin(a) + b;$
13	$Y = \sin(a) + b / c;$	14	$Y = \lvert c - b / a \rvert;$
15	$Y = \sin(a + b);$	16	$Y = a \cdot x + b;$
17	$Y = \cos(a + x) + b;$	18	$Y = \sin(a) / x + b / c;$
19	$Y = tg(a) \cdot x + b;$	20	$Y = a /(x + c);$
21	$Y = (a + b) / x;$	22	$Y = e^{a} + b / c;$
23	$Y = \lvert a - b \rvert / x;$	24	$Y = \sin(a + x / b);$
25	$Y = \lvert c - b / x \rvert;$	26	$Y = e^{a} + b;$
27	$Y = \sin(a + b / x);$	28	$Y = e^{a} + b / c;$
29	$Y = \sin \lvert c - b / a \rvert;$	30	$Y = \cos(a + b);$

Для того чтобы просмотреть полный список функций как VBA, так и главного приложения, необходимо воспользоваться окном **Object Browser** (**Просмотр объектов**), которое активизируется описанным выше способом. В окне **Classes** выберите Math, затем в окне **Members** выберите нужную вам функцию, просмотрите внизу синтаксис данной функции. Функция **Abs** вычисляет модуль аргумента, **Exp** – возводит *e* в степень, тригонометрические функции легко узнаваемы.

Открыть окно наблюдения значений переменных в редакторе VBA можно так: выделить переменную, значение которой вы хотите наблюдать, затем выбрать ***Debug – Quick Watch***. Чтобы добавить для наблюдения еще одну переменную, нужно выделить ее и войти ***Debug – Add.***



Контрольные вопросы

1. Сколько числовых типов данных насчитывается в VBA?
2. Какая разница между числами типа Integer и Single? Между Integer и Long?
3. Что такое символы определения типа?
4. Что означает выражение определить переменную неявно? А что такое явно?
5. Каковы правила в языке VBA для имен переменных? Чем еще следует руководствоваться при выборе имен переменных?
6. Какие преимущества дает явное описание переменных?
7. Почему в процедурах лучше использовать поименованные константы?
8. Для чего служит функция InputBox?
9. Какие аргументы функции InputBox являются обязательными?
10. Какого типа значение возвращает функция InputBox?
11. Как можно воспользоваться результатом выражения? Должны ли вы использовать результат выражения?
12. Для каких целей используется знак равенства как оператор?

Упражнения

1. Исходя из здравого смысла, решите, какая из перечисленных ниже величин должна быть переменной, а какая – константой. Выберите для них имена и напишите определение типа (если можно, сделайте это и с помощью ключевых слов, и с помощью символов определения типа).
   А Вычисленное количество столбцов в листе Excel
   Б Объем продаж для подразделения компании
   В Предполагаемое количество респондентов в опросе
   Г Вычисленная площадь поверхности цилиндра
   Д Коэффициент для перевода дюймов в сантиметры
   Е Рентабельность операции, выраженная в процентах
2. Напишите вручную процедуру с названием EchoThis, в которой используется функция InputBox для ввода пользователем некоторой фразы, а потом эта фраза выводится на экран функцией MsgBox.
3. С помощью программы записи макросов создайте в книге Personal.xls макрос, который будет открывать существующую книгу Excel и выбирать в ней лист с именем Лист 3. Остановите запись макроса. Отредактируйте записанный макрос так, чтобы он спрашивал у пользователя имя файла книги и затем открывал книгу с этим именем. (Подсказка. Сначала внимательно ознакомьтесь с текстом записанного макроса и поймите, в какой строке открывается файл книги. Поместите вызов функции InputBox непосредственно перед этой строкой. Присвойте введенное пользователем значение переменной, а затем подмените этой переменной непоименованную константу, созданную при записи макроса.)
4. Расставьте скобки в приведенных ниже выражениях так, чтобы их результат соответствовал указанному.

*3*5-7*	*-6*
4-7 + 26 /10	*3.07*
312 /47+16-2	*5.114754*

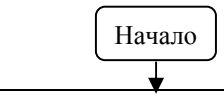
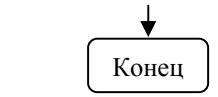
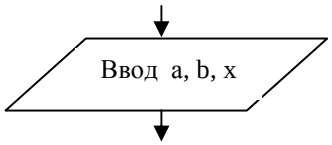
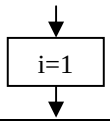
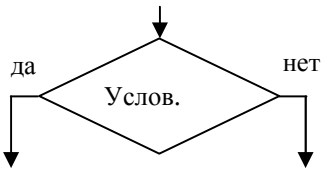
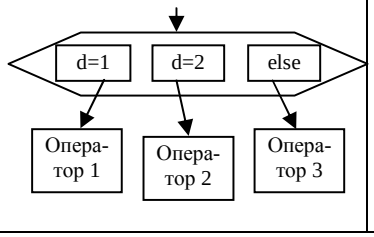
Лабораторная работа № 9

Разветвляющийся вычислительный процесс

Алгоритмизация

Хорошую программу проще написать, если, предварительно проанализировав задачу, составить алгоритм ее решения, т. е. алгоритм программы. Алгоритм можно составить в виде словесного описания последовательности действий или представить в виде блок-схемы.

Блок-схема состоит из графических объектов различных типов, содержащих внутри себя текст, поясняющий, что именно выполняется в этом фрагменте. Графические объекты соединяются стрелками, показывающие последовательность обработки блоков. Перечень типов блоков и их описание приведен в следующей таблице.

№	Тип блока	Описание блока
1		*Признак начала* алгоритма.
2		*Признак конца* алгоритма.
3		*Блок ввода/вывода*, означает реализацию ввода данных с клавиатуры (или вывода данных на экран). В данном случае, это блок ввода, и эквивалентом его в VBA является оператор InputBox.
4		*Блок присвоения* переменным некоторых значений или выражений. В данном примере переменной i присваивается единица.
5		*Блок условия* применяется когда, в зависимости от истинности (или ложности) некоторого условия, выполняться будут те (или иные) операторы. Если *Условие* будет истинным, то выполняться будет левая ветвь, а правая выполняться не будет. Если *Условие* будет ложным, то выполняться будет правая ветвь, а левая выполняться не будет.
6	*Блок множественного ветвления* используется, когда в зависимости от значения некоторого ключа-идентификатора, вычислительный процесс разветвляется на больше чем две ветви. В VBA для этого используется оператор «Select Case». В данном примере образуется три ветви, но их может быть и больше. Вычислительный процесс идет только по одной ветви.	
Пример		«Else» является необязательным параметром и означает случай, когда ключ-идентификатор примет такое значение, которого нет среди перечисленных значений (для данного примера $d \neq 1$ и $d \neq 2$, например 3 или 0 или что-то другое).

Типы вычислительных процессов

1. Линейный вычислительный процесс

Если при запуске программы выполняются все ее операторы, начиная с первого и кончая последним, то это называется *линейным вычислительным процессом*. Например, вычисление площади круга, выполненное в предыдущей работе.

Освоению линейного вычислительного процесса посвящена вторая лабораторная работа.

2. Разветвляющийся вычислительный процесс

Если в результате выполнения программы, процесс вычисления может пройти только по одному из нескольких (в общем случае многих) альтернативных путей, то такой вычислительный процесс называется разветвляющимся. В простейшем случае он реализуется оператором IF, который подразумевает две альтернативы: одну, если условие выполняется, и другую, если условие не выполняется. Например, если студент иногородний, то ему предоставляется общежитие, если нет, то общежитие не предоставляется.

Пример 1

В некотором магазине предоставляется скидка в размере 10%, если сумма приобретенного товара превышает 1000 денежных единиц. Написать процедуру, которая будет подсчитывать сумму этой скидки.

Листинг процедуры **Discount1**, подсчитывающей сумму скидки, приведен далее.

```
1.  Sub Discount1()
2.      Dim dCost As Double     ' Стоимость покупки
3.      Dim dDiscount As Double ' Скидка
4.
5.      dDiscount = 0#          ' Обнуление скидки
6.      dCost = InputBox("Введите стоимость покупки")
7.      If dCost > 1000 Then dDiscount = dCost * 0.1
8.      MsgBox "Скидка равна: " & dDiscount
9.  End Sub
```

В ней:

Строки 2 и 3 – объявление двух переменных типа ***Double*** для хранения стоимости покупки и суммы скидки, соответственно.

Строка 5 – исходно никакая скидка не предоставляется.

Строка 6 – ввод стоимости приобретенного товара.

Строка 7 – подсчет суммы скидки, если стоимость покупки превышает 1000 денежных единиц.

Строка 8 – вывод результата.

Рассмотрим инструкцию ***If*** языка программирования VBA более формально, и более подробно. Синтаксис команды:

If <условие> Then <инструкции$_1$> [Else <инструкции$_2$>]

Работает инструкция ***If*** следующим образом:

10. В начале вычисляется значение `условия`, которое может быть ***True*** или ***False***.

11. Если условие имеет значение ***True***, то выполняется группа инструкции$_1$, причем несколько инструкций должны помещаться в одной строке, а группа инструкции$_2$ – пропускается.
12. Если же условие имеет значение ***False***, то, наоборот, выполняется группа инструкции$_2$, а группа инструкции$_1$ – пропускается.
13. Выполнение программы в любом случае продолжается со следующей за ***If*** инструкции.
14. Конструкции, заключенные в квадратные скобки, являются необязательными. То есть, может отсутствовать конструкция Else и инструкции$_2$, которые выполняются, если условие ложно.

В приведенном выше Примере 1 используется только одна инструкция dDiscount = dCost * 0.1, которая выполняется, если условие истинно, т.е. стоимость приобретенного товара превышает 1000 денежных единиц. Если же условие ложно, то никакие инструкции не выполняются.

Группы инструкции$_1$ и инструкции$_2$ могут состоять не из одной инструкции, как в предыдущем примере, а из нескольких, разделенных двоеточием (:). Например:

If A > 10 Then X = X + 1: Y = Y + X: Z = Z + Y

Приведенную выше процедуру **Discount1** можно написать и по-другому, например, задействовав ветвь ***Else*** инструкции ***If***. Листинг откорректированной таким образом процедуры, теперь уже названной **Discount2**, приведен далее.

```
1:  Sub Discount2()
15.  Dim dCost As Double     ' Стоимость покупки
16.  Dim dDiscount As Double ' Скидка
17.
18.  dCost = InputBox("Введите стоимость покупки")
19.  If dCost > 1000 Then dDiscount = dCost * 0.1 _
20.               Else dDiscount = 0#
21.  MsgBox "Скидка равна: " & dDiscount
22.End Sub
```

В ней значение скидки равное 0 присваивается переменной dDiscount не в начале процедуры априори, а только если сумма покупки не превышает 1000 денежных единиц. При такой записи процедуры длина инструкции ***If*** стала чересчур большой, и ее пришлось перенести на следующую строку (для переноса строки используется знак «_»). Если же в каждой ветви инструкции ***If*** будет задействовано по несколько инструкций, то вся инструкция ***If*** станет совсем нечитабельной. Для этого в VBA имеется блочная форма инструкции If, которая в общем виде выглядит следующим образом:

If <условие> Then
<инструкции$_1$>
[Else
[<инструкции$_2$>]]
End If

Работает она точно так же, как и приведенный ранее ее однострочный вариант.

Тогда, с учетом блочного синтаксиса инструкции ***If***, процедуру определения скидки можно записать следующим образом, назвав ее при этом уже **Discount3**, соответственно:

```
1:  Sub Discount3()
2:     Dim dCost As Double     ' Стоимость покупки
3:     Dim dDiscount As Double ' Скидка
4:
5:     dCost = InputBox("Введите стоимость покупки")
6:     If dCost > 1000 Then
7:        dDiscount = dCost * 0.1 _
8:     Else
9:        dDiscount = 0#
10:    End If
11:    MsgBox "Скидка равна: " & dDiscount
12:  End Sub
```

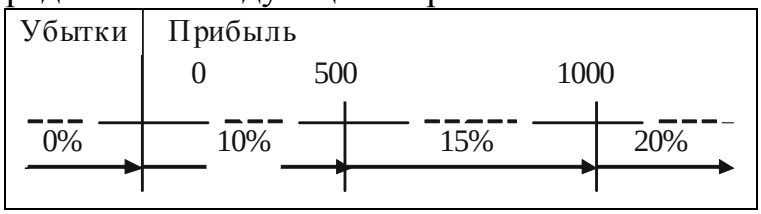
Хотя процедура **Discount3** несколько длиннее своих предыдущих вариантов, зато она более структурирована и зрительно воспринимается гораздо лучше. *Однострочный вариант инструкции* ***If*** *следует использовать, если при выполнении (или невыполнении) некоторого условия необходимо исполнить одно простое действие* (как в Примере 1), *во всех же остальных случаях гораздо эффективнее применять ее блочный вариант*.

Язык программирования VBA допускает любую глубину вложенности инструкций If друг в друга. То есть в качестве инструкции в любой его ветви, в свою очередь, может выступать инструкция ***If***. Единственное и естественное возникающее при этом ограничение состоит в том, что *вложенная инструкция* ***If*** *должна целиком и полностью помещаться в соответствующей ветви внешней инструкции*.

1. Запустите Excel и откройте свой файл.
2. Войдите в редактор VBA и запишите процедуру, вычисляющую скидку на покупку в зависимости от стоимости товара по следующей схеме:
   a. Если стоимость товара<1000, скидка=0%;
   b. Если стоимость товара>=1000 и меньше 5000, скидка=10%;
   c. Если стоимость товара>=5000,скидка=20%.
3. Запустите на выполнение процедуру и выполните ее в пошаговом режиме, вводя значения стоимости товара из разных диапазонов. Обратите внимание, какие операторы выполняются в зависимости от вводимого значения.
4. Написать программу начисления налога на прибыль, используя вложенные операторы *If,* в соответствии со следующей таблицей:

Прибыль [денежная единица]	Ставка налога [%]
до 500	10
от 500 до 1000	15
1000 и более	20

Графически шкалу начисления налога, в зависимости от полученной прибыли, можно представить следующим образом:



Поскольку на убытки, или отрицательную прибыль, налог не начисляется, то в программе необходимо предусмотреть обработку такой ситуации. В ней так же необходимо предусмотреть две переменные типа ***Double*** для хранения: суммы прибыли и суммы начисляемого налога.

Вложенные инструкции ***If*** для проверки трех и более условий визуально становятся мало понятными. Поэтому в VBA была предусмотрена еще одна ее форма:

If <условие> Then
[<инструкции$_1$>]
[ElseIf <условие$_n$> Then
[<инструкцииn>] ...
[Else
[<инструкции$_2$>]]
End If

В ней может быть сколько угодно ветвей ***ElseIf*** для реализации альтернативных вариантов. Единственное ограничение при этом – все они должны идти до последней ветви ***Else***.

5. Составить блок-схему алгоритма в соответствии со своим вариантом. Ввод координат точки осуществить с экрана; в качестве заголовка окна ввода взять свою фамилию; определить, попадает ли точка в заданную область, и вывести на экран координаты точки и ответ о местоположении точки.
6. Написать и отладить программу разветвляющегося процесса с использованием оператора ***If.***

Рассмотрим инструкцию ***Select Case*** языка программирования VBA, которая используется в том случае, если нужно выбрать более чем из двух ветвей выполнения процедуры. Синтаксис команды:

Select Case Выражение
Case Логич.выражение$_1$
Инструкции$_1$
Case Логич.выражение$_2$
Инструкции$_2$
Case Логич.выражение$_N$
Инструкции$_N$
[Case Else
Инструкции$_{Else}$]
End Select

Работает инструкция ***Select Case*** следующим образом:

1. В начале вычисляется значение Выражения, и затем результат этого вычисления с каждым из выражений, хранящихся в каждом Логич. выражении.

2. Если значения Выражения удовлетворяет Логич.выражению для одного из разделов *Case*, Инструкции этого раздела выполняются.
3. Если Выражение соответствует более, чем одному разделу *Case*, выполняются инструкции только первого из них.
4. После выполнения инструкций выбранного раздела *Case* выполняется следующая за ключевыми словами *End Select* инструкция.
5. Если значения Выражения не удовлетворяет Логич.выражению ни для одного из разделов *Case*, выполняются инструкции $Инструкции_{Else}$.

В отдельных разделах *Case* может стоять более, чем одна Инструкция, разделенные запятыми. $Инструкции_N$ имеют следующий синтаксис

$Инструкция_1, Инструкция_2, Инструкция_N$

Логич.выражения могут быть некоторыми числовыми, строковыми или логическими выражениями, они могут быть представлены рядом значений с использованием оператора *To:*

$Выражение_1$ *To* $Выражение_2$.

Пример использования оператора *Select Case:*

```
Sub Temp()
  Dim temp
  Temp=Application.InputBox(_Prompt:=”Сколько градусов?”,_
Title:=”Процедура Temp”,_ Type:=1)
Select Case temp
Case Is>40
MsgBox ”Очень горячо!”
Case 37 To 40
MsgBox ”Нормально!”
Case 34 To 36
MsgBox ”Хорошо!”
Case 30 To 33
MsgBox ”Прохладно!”
Case Else
MsgBox ”Подогрейте!”
End Select
End Sub
```

В языке программирования VBA существуют инструкции безусловного перехода, которые используется в том случае, если нужно перейти в какое-либо место процедуры без проверки условий; в основном эта инструкция применяется при обработке ошибок. Синтаксис команды:

GoTo Line,

где *Line* – метка или номер строки в той же процедуре, что и инструкция *GoTo*. Метка *Line* имеет следующий синтаксис: *name:*

Здесь *name*: - это идентификатор VBA.

6. Написать и отладить программу разветвляющегося процесса с использованием оператора *Select Case* в соответствии со своим вариантом.
7. Показать результаты работы преподавателю.

Индивидуальные задания

1. Попадает ли точка А в круг радиусом R=3 с центром в точке С(3, 4)?
2. Попадает ли точка А в круг радиусом R=5 с центром в точке С(6, 8)?
3. Попадает ли точка А в прямоугольник $2 \le x \le 4$ и $2 \le y \le 5$?
4. Попадает ли точка А в полосу $0 \le x \le 5$?
5. Попадает ли точка А в полосу $4 \le x \le 6$?
6. Превышает ли расстояние от точки А до точки В 5 единиц. Координаты точки В ввести с клавиатуры?
7. Лежит ли точка А на прямой $y = a + 4$?
8. Принадлежит ли точка А графику функции $y = 4x^2 - 5$?
9. Попадает ли точка А в прямоугольник $6 \le x \le 10$ и $7 \le y \le 9$?
10. Попадает ли точка А в один из кругов $R1 = 1$, С1(1, 1); $R2 = 2$, С2(5, 4)?
11. Попадает ли точка А в первый квадрант?
12. Попадает ли точка А во второй квадрант?
13. Попадает ли точка А в третий квадрант?
14. Попадает ли точка А в четвертый квадрант?
15. Попадает ли точка А в область положительных значений X?
16. Попадает ли точка А в область отрицательных значений X?
17. Попадает ли точка А в область неположительных значений оси у?
18. Попадает ли точка А в область неотрицательных значений оси у?
19. Попадает ли точка А в точку пересечения графиков функций $y = x + 1$ и $y = x^2 - 4 \cdot x + 2$?
20. Принадлежит ли точка А графику функции $y = 4x^2 + 8$?
21. Попадает ли точка А в область пересечения кругов С1: R1=3, центр в точке С1(0, 0) и С2: R2=5, С2(5, 0)?
22. Попадает ли точка А в прямоугольник $-1 \le x \le 5$ и $-2 \le y \le 3$?
23. Попадает ли точка А на окружность $R = 4,\ C(1,1)$?
24. Попадает ли точка А на окружность $R = 3,\ C(4,5)$?
25. Превышает ли расстояние от т. А до оси X 6 единиц?
26. Превышает ли расстояние от т. А до оси У 7 единиц?
27. Превышает ли расстояние от т. А до начала координат 5 единиц?
28. Попадает ли точка А на одну из окружностей: $R1 = 1,\ C1(1,2)$ и $R2 = 2,\ C2(2,3)$?
29. Превышает ли расстояние от т. А до прямой $x = 4$ шесть единиц?
30. Превышает ли расстояние от т. А до прямой $y = -5$ десять единиц?

Контрольные вопросы

1. Что такое условный и безусловный переход?
2. Какие вы знаете инструкции условного перехода?
3. Какие вы знаете инструкции безусловного перехода?
4. Когда и почему можно использовать инструкцию Select Case?
5. Сколько предложений Case может включать инструкция Select Case?
6. Как определить ветвь инструкции для выполнения, если в инструкции Select Case нет ни одного предложения Case? Куда в этом случае переходит выполнение внутри инструкции Select Case?

Лабораторная работа № 10
Циклический вычислительный процесс

Одним из недостатков записанных макросов является их неспособность выполниться несколько раз, пока это не будет выполнено вручную. В VBA имеется несколько мощных средств, позволяющих легко повторять операции и управлять их повторением.

Программные структуры, дающие возможность повторяться одной или нескольким инструкциям, называются **циклами.** Каждый раз, когда VBA завершает один полный цикл выполнения всех инструкций, находящихся внутри циклической структуры, называется итерацией цикла.

Некоторые циклические структуры созданы так, чтобы они выполнялись фиксированное число раз. Они называются **определенными** циклами. Другие выполняются неопределенное число раз, зависящее от некоторого условия. Ввиду того, что это число повторений неопределенно, они называются **неопределенными** циклами.

И в тех, и в других структурах имеются выражения, показывающие до каких пор, следует повторять цикл. Они называются **определителями** цикла. В фиксированной структуре такой определитель почти всегда является числовым выражением. В неопределенной структуре определитель обычно представляет собой логической выражение, которое описывает условие, при котором выполнение цикла продолжаются или останавливается. Логические выражения для циклических структур создаются и используются точно так же, как и в случае условных инструкций ветвления в VBA.

Принято два основных способа построения неопределенного цикла. В первом случае VBA будет проверять условия определителя до выполнения цикла. Если условие выполнения имеет значение False, то инструкции внутри цикла просто будет пропущены. Во втором случае можно создать такой цикл, чтобы условие в определителе проверялось после выполнения этих инструкции.

Можно также создавать неопределенный цикл, у которого в определителе нет никакого условия вовсе. Такого рода циклы выполняются вечно и называются **бесконечными**. Они никогда не заканчиваются; большинство таких циклов является результатом ошибки программиста.

Повторение фиксированное число раз: циклы For

Простейшим из циклов является. В VBA имеется две различные структуры таких циклов: **For .. Next** и **For Each ... Next**. Обе эти структуры называются циклами **For**, потому что всегда выполняются определенное число раз.

Применение цикла For.. Next

Цикл For.. Next имеет такой синтаксис.

For *counter* **=** *start* **to** *end* **[Step** *StepSize***]**

Statements

Next [*counter***]**

Где **counter** представляет любую числовую переменную, соответствующую правилам VBA, обычно с типом данных Interger или Long. Выражение **start** является любым числовым выражением и определяет

начальное значение переменной счетчика counter. Выражение **end** – также числовое выражение и определяет конечное значение той же переменной.

По умолчанию VBA каждый раз после завершения инструкций из цикла увеличивает значение переменной **counter** на 1. Другое значение увеличения можно задать с помощью необязательного ключевого слова **Step**. После него необходимо указать величину (шаг, на которую надо увеличить значение **counter**). В нашем синтаксисе **StepSize** представляет любое числовое выражение и как раз указывает эту самую величину.

Statements представляет одну или несколько инструкций VBA (или вообще ни одной из них). Они составляют тело цикла, и каждая из них выполняется всякий раз, когда выполняется сам цикл.

Ключевое слово **Next** сигнализирует VBA, что достигнут конец цикла, необязательная переменная **counter** после **Next** должна быть той же переменной, которая находится после ключевого слова **For** в начале циклической структуры. Ее надо писать после **Next**, чтобы легче читать исходный текст программы (особенно с вложенными циклами **For.. Next**).

При выполнении цикла **For.. Next** вначале переменной counter присваивается значение, представленное переменной **start**. Затем VBA выполняет все инструкции, представленные в **Statements**, пока не будет достигнуто ключевое слово Next. Ключевое слово Next сигнализируют VBA, что достигнут конец тела цикла. Затем значение переменной counter увеличивается на величину переменной **StepSize** (шаг), если есть необязательной ключевое слово **Step**. Если этого слова нет, то значение counter увеличивается на 1. Происходит переход на начало цикла, а затем текущее значение counter сравнивается со значением, представленным **end**. Если **counter** будет меньше или равен **end**, то цикл выполнится снова. Если **counter** больше, чем **end** , то выполнение продолжается с первой инструкции, которая находится после ключевого слова **Next**.

Счетчик в рассматриваемом типе циклов может быть как с возрастающим счетчиком, так и с убывающим. Для этого необходимо использовать ключевое слово **Step**, с отрицательным значением **StepSize**, в результате чего значение **counter** будет уменьшаться от большого значения к меньшему.

Пример цикла For.. Next с возрастающим счетчиком

Эта процедура получает от пользователя два числа, в заданном ими диапазоне суммирует все целые числа, а затем выводит на экран полученное значение. Если, например, вы запустили эту программу и ввели значения 4 и 8, то будут просуммированы числа 4, 5, 6, 7 и 8, а на экране будет отображено число 30.

```
Sub Demo_ForNext()
Dim k As Integer
Dim uStart As String
Dim uEnd As String
Dim uSum As Long
uStart = InputBox("Введите натуральное число:")
```

```
uEnd = InputBox("Введите второе натуральное число")
uSum = 0
For k = uStart To uEnd
uSum = uSum + k
Next k
MsgBox "Сумма натурального ряда от" & uStart &
_" до" & uEnd & " равна: " & uSum
End Sub
```

Введенные значения еще не обрабатывались поэтому сумма должны быть нулевой

Цикл **For..Next**

Повторите назначение этого оператора

Пример цикла For.. Next с убывающим счетчиком

Этот пример делает то же самое, но счетчик цикла работает с убыванием.

```
Sum Demo_ForNextDown()
Dim k As Integer
Dim uStart As Integer
Dim uEnd As Integer
Dim uSum As Long
uStart = InputBox("Введите натуральное число:")
uEnd = InputBox("Введите второе натуральное число")
uSum = 0
For k = uStart To uEnd Step -1
   uSum = uSum + k
Next k
MsgBox "Сумма натурального ряда от" & uStart & _
" до" & uEnd & " равна: " & uSum
End Sub
```

Цикл **For..Next,** убывающий

1. Введите оба примера, найдите отличия между ними.
2. Запустите их на выполнение, в качестве первого числа введите 3, а второго 15. Вычислите и запишите результат каждого примера.
3. Сделайте тоже самое, но числа введите наоборот, то есть сначала введите 15 а затем 3.
4. Поясните полученные результаты.

Повторение неопределенное число раз: циклы Do

В VBA имеется очень мощная инструкция для создания в процедурах и функциях циклических структур, повторяющихся неопределенное число раз. Все такие циклы создаются с помощью единственной инструкции – инструкции **Do**. Она обладает такими возможностями и такой гибкостью, что у нее фактически четыре конструкции, относящиеся к двум основным категориям.

Эти категории состоят, из циклов **Do**, которые проверяют условие определителя перед выполнением тела цикла, и, во-вторых, из тех циклов **Do**, которые проверяют это условие после выполнения тела цикла. Независимо от того, проверяется ил определитель до или после выполнения тела цикла, существует два способа контролировать, сколько раз выполняется неопределенный цикл.

- *Циклы, контролируемые счетчиками.* В этом случае тело цикла выполняется, пока значение некоторого счетчика находится ниже или выше указанной границы; в этом есть определенное сходство с циклов For .. Next, за исключением того, что на программисте лежит ответственность за инициализацию этого счетчика, а также за увеличение или уменьшение его значения. Такие циклы Do имеет смысл писать, когда счетчик не является регулярным, а также когда нельзя определить конечную границу, пока не началось выполнение самого цикла. Например, требуется пройти первые 16 строк листа, иногда увеличивая за один раз значение строки на 1, а иногда за один раз – на 2. В связи с тем, увеличение номера строка (то есть шаг счетчика) меняется, цикл For .. Next использовать нельзя; вместо него надо исследовать цикл Do.
- *Циклы, контролируемые событиями.* В этом случае значением определителя имеет значение True или False, в зависимости от некоторого события или действия, имевшего место во время выполнения цикла. Например, вы написали цикл, выполняемый неопределенное число раз, пока пользователь не ведет известное число в диалоговом окне. Ввод такого числа как раз и является тем событием, которое завершит цикл. А вот другой пример: можно выполнять операции над ячейками листа, пока не будет достигнута пустая ячейка. Переход в пустую ячейку и будет событием, завершающим цикл.

Циклы с проверкой условия перед выполнением.

Чтобы условия определителя проверялись перед выполнением тела цикла, надо логическое выражение определителя просто поместить в начале инструкций, составляющих тело цикла. Это можно реализовать с помощью ключевых слов While и Until.

Создание циклов с помощью Do While

Первой циклической структурой, где условие определителя проверяется перед выполнением цикля, является **Do While**, имеющего следующий синтаксис.

Do While *condition*
Statements
Loop

Где **condition** представляет логическое выражение для определителя цикла, **statements** – одну или большое количество инструкций VBA (или вообще ни одной из них), которые составляют тело цикла; все они выполняются каждый раз когда выполняется цикл. Ключевое слово **Loop**, находящееся после **statements**, сигнализирует что достигнут конец цикла, и указывает точку с которой выполняется переход в начало цикла, чтобы проверить условие определителя.

В инструкции **Do While** выражение **condition** находится в начале цикла, поэтому условие определителя проверяется перед выполнением самого цикла. Поскольку в этом случае используется ключевое слово **While**, цикл будет выполняться до тех пор, пока логическое выражение, представленное **condition**, будет иметь значение **True**.

Если логическое выражение, представленное **condition**, будет равно **False** при первом выполнении инструкции **Do While**, то цикл будет просто пропущен без всякого выполнения.

Пример цикла Do While

Эта процедура считает введенные пользователем нечетные числа, прекращая роботу тогда, когда таких чисел станет 10. Затем введенные числа выводятся на экран.

Обратите внимание на использование функции Mod.

```
Sub Count_OddNums()
Const ocTitle = "Подсчет нечетных чисел"
Dim OddCount As Integer ' счетчик нечетных чисел
Dim OddStr As String ' строка для отображения нечетных чисел
Dim Num ' переменная типа Variant для ввода
OddStr = " " ' Строка для чисел пока что пуста
OddCount = 0 ' Счетчик чисел пока что равен нулю
Do While OddCount < 10
Num = InputBox("", ocTitle)
If (Num Mod 2) <> 0 Then
OddCount = OddCount + 1
OddStr = OddStr & Num & "  "
End If
Loop
MsgBox "Вы ввели такие нечетные числа:" & Chr(13) _& OddStr, , ocTitle
End Sub
```

Цикл **Do While…Loop**

5. Введите пример. Выполните его.
6. Как изменится процедура, если подсчитывать все числа кратные 3? а если нечетные? Ответ запишите в тетрадь.
7. Процедуру для ввода пяти чисел кратных 3 введите и проверьте.

Создание циклов с помощью Do Until

В инструкции **Do** вариант **Do While** – это только один из тех способов ее создания, в которых условие определителя проверяется перед выполнением цикла. Вторым из этих способов является вариант **Do Until**, имеющий следующий синтаксис.

Do Until *condition*
statements
Loop

Где **condition** представляет логическое выражение для определителя цикла, а **statements** – инструкции VBA, которые составляют тело цикла. Ключевое слово **Loop**, находящееся после **statements**, сигнализирует, что достигнут конец цикла, и указывает точку, с которой выполняется переход в начало цикла, чтобы проверить условие определителя.

В инструкции **Do until** выражение **condition** находится в начале цикла, поэтому условие определителя проверяется перед выполнением самого цикла. В связи с тем, что в этом случае используется ключевое слово **Until** (сто означает

пока нет чего-либо), цикл будет выполняться до тех пор, пока логическое выражение, представленное condition, будет иметь значение **False**.

Пример цикла Do Until

Эта шуточная процедура предлагает вводить вам дату в формате XX.XX.XX (например 18.03.03) до тех пор, пока вы не введете дату какого-нибудь воскресенья.

Обратите внимание на использование функции WeekDay.

```
Sub Stop_AtEvenNums()
Const evTitle = "В ожидании воскресенья..."
Dim EventFlag As Boolean  ' Признак выхода – Переменная для выхода из
цикла
Dim Num ' Переменная для даты, описана как тип Variant
EventFlag = False ' Признак цикла пока равен ЛОЖЬ
Do Until EventFlag = True
Num = InputBox("Введите дату:", evTitle)
  If Weekday(Num, vbMonday) = 7 Then
MsgBox "Вы ввели дату воскресенья!!!", , "Запомните эту дату! Ура!"
EventFlag = True
Else
MsgBox " Жаль, но отдохнуть не удастся!", , "Не надо печалиться"
End If
Loop
MsgBox "Хорошо повеселиться!!!", , "Мы рады за Вас!!!"
End Sub
```

Цикл **Do Until…Loop**

Если логическое выражение, представленное **condition**, будет равно **True** при первом выполнении инструкции **Do Until**, то цикл будет просто пропущен без всякого выполнения.

Циклы с проверкой условия после выполнения

Чтобы условия определителя проверялись после выполнения тела цикла логическое выражение определителя надо просто поместить в конце инструкции, составляющих тело цикла, а именно после ключевого слова Loop, которое сигнализирует о конце цикла.

8. Введите пример. Выполните его.
9. Добавьте в него еще одну проверку для субботы, с комментарием типа «Ура, это уже завтра!!!» и также выходом из цикла.
10. Найдите в справке описание функции WeekDay и законспектируйте в тетрадь ее аргументы.

Создание циклов с помощью Do…Loop While

Первой циклической структурой, где условие определителя проверяется после выполнения цикла, является **Do…Loop While**, имеющий следующий синтаксис.

Do

statements

Loop While *condition*

Где **condition** представляет логическое выражение для определителя цикла, **statements** – одну или большое количество инструкций VBA (или вообще ни одной из них), которые составляют тело цикла; все они выполняются каждый раз когда выполняется цикл. Ключевое слово **Loop**, находящееся после statements, сигнализирует что достигнут конец цикла, и указывает точку с которой выполняется переход в начало цикла, чтобы проверить условие определителя.

В инструкции **Do…Loop While** выражение **condition** находится в конце цикла, поэтому условие определителя проверяется после выполнения самого цикла.

Поскольку в этом случае используется ключевое слово **While**, цикл будет выполняться до тех пор, пока логическое выражение, представленное **condition**, будет иметь значение **True**.

Каким бы ни было значение логического выражения, представленного **condition**, этот цикл выполниться хотя бы один раз.

Создание циклов с помощью Do…Loop Until

Другой циклической структурой **Do**, где условие определителя проверяется после выполнения цикла, является **Do…Loop Until**, имеющего такой синтаксис.

Do
Statements
Loop Until *condition*

В этой инструкции выражение condition находится в конце цикла, поэтому условие определителя повторяется после выполнения самого цикла. Поскольку в этом случае используется ключевое слово **Until**, то цикл будет выполняться до тех пор, пока логическое выражение, представленное condition, будет иметь значение **False**.

При выполнении цикла **Do…Loop Until** вначале выполняются инструкции, представленные в statements. Когда достигнуто ключевое слово **Loop**, проверяется логическое выражение, представленное **condition**; если его значение **False**, то происходит переход в начало структуры и снова выполняется тело цикла. Когда в конце цикла вновь достигнуто ключевое слово **Loop** и если значение **condition** сменилось на **True**, то выполнение продолжается с инструкций, находящихся после строи со словом **Loop**.

Каким бы ни было значение логического выражения, представленного **condition**, этот цикл выполниться хотя бы один раз.

Пример цикла Do…Loop Until

В процедуре приведенной ниже нужно угадать случайно сгенерированное число от 1 до 10. Цикл будет выводить окно, до тех пор, пока вы не угадаете «загаданное». В конце цикла, вы узнаете, сколько же попыток вам понадобилось.

Обратите внимание на генератор случайных чисел.

```
Sub Угадайка()
Dim EventFlag As Boolean
Dim g As Integer
EventFlag = False
Counter = 0  ’ счетчик ваших попыток
Randomize   ’ запуск генератора случайных чисел
c = Round(10 * Rnd + 1, 0)  ’ генерация случайного числа от 1 до 10
Do
g = InputBox("Угадайте число от 1 до 10")
      If g = c Then
      EventFlag = True
      MsgBox "Ты угадал с " & Counter & "  раза"
      End If
   Counter = Counter + 1
Loop Until EventFlag
End Sub
```

1. Введите пример. Найдите в справке генератор случайных чисел и законспектируйте описание и примеры.
2. Подумайте, с помощью каких циклических конструкций этот пример записать можно, а с помощью каких нельзя. Ответ напишите в тетради и обоснуйте.
3. Запишите эту программу в тетради с помощью другой циклической конструкции по вашему выбору. Введите и выполните ее.
4. Выводите сообщения об удачливости игрока, в зависимости от числа попыток, например, при попытках больше 5 – «Что-то день сегодня не очень!», а меньше «Ну ты и везунчик!».

Индивидуальные задания

1. Напишите программу, которая выводит все числа первой сотни, оканчивающиеся на 5.
2. Напишите программу, которая вводит число, а выводит таблицу умножения этого числа на все числа от 1 до 9.
3. Напишите программу, которая выводит таблицу степеней с 0-й по 9-ю числа 2.
4. Напишите программу, которая выводит таблицу значений квадратного корня на интервале [2;4] с шагом 0.1
5. Напишите программу, вывода таблицы квадратов первых 10 целых положительных чисел.
6. Напишите программу, вывода всех чисел, делящихся на 13 без остатка в интервале [1;100].
7. Напишите программу, вычисления среднего арифметического 10-и введённых чисел.
8. Напишите программу, которая 10 раз выводит Ваше имя и фамилию.
9. Напишите программу, вычисления суммы первых n целых положительных чисел. Количество суммируемых чисел n водится в начале работы программы.

10. Напишите программу, которая вычисляет сумму первых n членов ряда $1+\frac{1}{2}+\frac{1}{3}+...+\frac{1}{n}$. Количество суммируемых членов ряда задаётся во время работы программы.
11. Напишите программу, которая выводит таблицу значений функции $y=-2.4x^2+5x-3$ в диапазоне[-2;2] с шагом 0.5.
12. Напишите программу, которая выводит таблицу значений функции $y=|x|$ в диапазоне[-4;4] с шагом 0.5.
13. Напишите программу, которая выводит таблицу значений функции $y=|x+2|$ в диапазоне[-4;4] с шагом 0.5.
14. Напишите программу, которая выводит таблицу значений функции $y=|x-2|+|x+1|$ в диапазоне[-4;4] с шагом 0.5.
15. Напишите программу, которая выводит таблицу стоимости яблок в диапазоне от 1 кг до 100 г с шагом 100 г. Стоимость 1 кг яблок вводится во время работы программы.
16. Напишите программу, подсчёта суммы и среднего арифметического всех целых положительных чисел из заданного диапазона. Диапазон задаётся во время работы программы.
17. Напишите программу, вычисления факториала $y=n!$ по формуле $n!=1*2*3*\cdots*(n-1)*n;\ 0!=1$. Значение n вводится во время работы программы.
18. Напишите программу вычисления $y=1!+2!+3!+\cdots+n!$. Ввод n осуществляется во время работы программы.
19. Напишите программу определения числа e – основания натурального логарифма, с помощью ряда: $e=1+1/1!+1/2!+1/3!+\cdots+1/n!$ для всех значений n от 1 до m. Ввод m осуществляется во время работы программы.
20. Напишите программу вычисления $e^x=1+\frac{x}{1!}+\frac{x^2}{2!}+\cdots+\frac{x^n}{n!}+\cdots=1+\sum_{i=0}^{n}\frac{x^i}{i!}$ с заданной точностью $\varepsilon>0.0$, а также общее число слагаемых n. Каждое очередное слагаемое вычисляется через предыдущее по следующей рекуррентной формуле: $u_i=\frac{x}{i}u_{i-1}$, где $i=1,2,3$ и т.д. Ввод x и ε осуществляется во время работы программы.
21. Напишите программу, вычисления числа π с заданной точностью $\varepsilon>0.0$ при помощи следующего ряда: $p/4=1/1-1/3+1/5-1/7+1/9\cdots$, общий член которого вычисляется по формуле: $(-1^k)/(2k+1)$, где k – номер члена ряда. Подсчитать и вывести при этом также общее число просуммированных членов ряда. Ввод ε осуществляется во время работы программы.
22. Напишите программу, определения варианта лабораторной работы по двум последним цифрам номера зачетной книжки. Вариант лабораторной работы вычисляется путем вычитания 30 из числа, образованного двумя последними цифрами номера зачетной книжки, до тех пор, пока оно

больше 30. Две последние цифры номера зачётной книжки вводятся во время работы программы.

23. Напишите программу извлечения квадратного корня $y = \sqrt{x}$ методом Ньютона с заданной точностью $\varepsilon > 0.0$ по следующей рекуррентной формуле: $y_{n=1} = y_n + (x / y_n - y_n)/2$, где $n = 1,2,3$ и т.д. Ввод x и ε осуществляется во время работы программы.

24. Напишите программу определения значения величины: $y = \sum_{n=1}^{m} \frac{n}{n+1}$. Ввод m осуществляется во время работы программы.

25. Напишите программу определения значения величины: $y = \prod_{n=1}^{m} ((n+1)/n! + n)$. Ввод m осуществляется во время работы программы.

26. Напишите программу, которая последовательно вводит в переменную целого типа произвольные числа до тех пор, пока очередное введённое число не будет равно нулю, и определяет их сумму.

27. Напишите программу, которая последовательно вводит в переменную целого типа произвольные числа до тех пор, пока очередное введённое число не будет равно нулю, и определяет их среднее арифметическое.

28. Напишите программу, которая последовательно вводит n раз в переменную целого типа произвольные числа и определяет их среднее арифметическое.

29. Напишите программу печати таблицы перевода температуры по шкале Фаренгейта в температуру по шкале Цельсия в диапазоне от x до y с шагом z по формуле $C(F) = (5/9)(F - 32)$, где C – температура по шкале Цельсия, F – температура по шкале Фаренгейта. Ввод x, y и z осуществляется во время работы программы.

30. Напишите программу печати таблицы перевода температуры по шкале Цельсия в температуру по шкале Фаренгейта в диапазоне от x до y с шагом z по формуле $F(C) = (9/5)(C + 32)$, где F – температура по шкале Фаренгейта, C – температура по шкале Цельсия. Ввод x, y и z осуществляется во время работы программы.

Контрольные вопросы

1. Что такое циклы и для чего они предназначены?
2. Какие типы циклов вы знаете?
3. Что такое определитель цикла? В каких структурах он является числом?
4. Чем отличаются циклы предусловия и постусловия?
5. Запишите синтаксис всех циклов и охарактеризуйте его по такому плану: синтаксис; тип цикла; тип условия; характер определителя. Например:

Цикл	Тип цикла	Тип условия	Определитель
Do *Statements* **Loop Until** *condition*	неопределенный	послеусловие	условие

Список источников

1. А. Горячев, Ю. Шафрин. Практикум по информационным технологиям. – М.: Лаборатория базовых знаний, 1999. – 272 с.
2. Александр Левин. Самоучитель работы на компьютере. 6-е издание, исправленное и дополненное. – М.: Изд. «Нолидж», 2000. – 656 с.
3. В.Ф. Ляхович. Основы информатики. – Ростов на Д.: Изд-во «Феникс». – 2000. – 608 с.
4. Зубов Ф.Н. Microsoft Windows 2000 / Планирование, развертывание, установка. – 2-ое изд. испр. – М.: Издательско-торговый дом «Русская Редакция», 2000. – 592 с.: ил.
5. Информатика для юристов и экономистов / Симонович С.В. и др. – СПб.: Питер, 2001. – 688 с.
6. Информатика. Базовый курс / Под ред. С.В. Симоновича - СПб: Издательство «Питер», 2000. – 640 с: ил.
7. Информатика. Задачник-практикум в 2т. / Под ред. И.Г. Семакина, Е.К. Хеннера: - М.: Лаборатория Базовых Знаний, 1999 г.
8. Информатика: Учебник. – 3-е изд. перераб. / Под ред. проф. Н.В. Макаровой. – М.: Финансы и статистика, 2001. – 768 с.
9. Информационные технологии бухгалтерского учета / О.П. Ильина. – СПб.: Питер, 2001. – 688 с.: ил.
10. Кеймен В.А. Інформатика: Учебник. – М.: ИНФРА. – М, 2000 – 232 с. (Серия высшее образование).
11. Клименко С.В., Чичерин А.Л., Колесников А. Windows 98 (русифицированная версия). – К.: Ирина, BHV, 2000. – 368 с.
12. Основы информатики. Лекции по курсу Информатика. Для студентов первого и второго курсов. / А.Б. Костенко, Б.И. Погребняк, Н.В. Гринчак, Т.А. Холодная – Харьков: ХГАГХ, 1997. – 170 с.
13. Рыжиков Ю.И. Информатика. Лекции и практикум. – СПб.: Корона принт, 2000. – 256 с.
14. Сокольский М. Операционная система Windows 2000 Professional для профессионалов. – М.; Познавательная книга плюс, 2000. – 565 с.
15. Стинсон К. Эффективная работа в Windows 95 / Пер с англ. – СПб.: Питер, 1997. – 784 с.
16. Фигурнов В.Э. IBM PC для пользователя. 7-е изд-во, перераб. и доп. – М.: ИНФРА-М, 1999. – 640 с.
17. Экономическая информатика: / Под ред. П.В. Конюховского и Д.Н. Колесова. – СПб.: Питер, 2000. – 560 с.

Навчальне видання

Методичні вказівки

для виконання лабораторних,
самостійних і контрольних робіт

з курсу

«ІНФОРМАТИКА І КОМП'ЮТЕРНА ТЕХНІКА»

(для студентів 1-го і 2-го курсів заочної форми навчання
навчально-кваліфікаційного рівня бакалавр,
напряму підготовки 6.030601 «Менеджмент»)

(*Рос. мовою*)

Укладач **Погребняк** Борис Іванович

Відповідальний за випуск *Б. І. Погребняк*

За авторською редакцією

Комп'ютерне верстання *К. А. Алексанян*

План 2011, поз. 450 М

Підп. до друку 23.06.2011 р.	Формат 60×84/16
Друк на ризографі.	Ум. друк. арк. 4,8
Тираж 50 пр.	Зам. №

Видавець і виготовлювач:
Харківська національна академія міського господарства,
вул. Революції, 12, Харків, 61002
Електронна адреса: rectorat@ksame.kharkov.ua
Свідоцтво суб'єкта видавничої справи:
ДК № 4064 від 12.05.2011 р.