СОДЕРЖАНИЕ

Введение

VBA (Visual Basic for Applications) — это язык программирования, встроенный во множество отдельных программ и прикладных пакетов — от приложений Microsoft Office (включая Microsoft Project и Microsoft Visio) и до таких мощных пакетов, как AutoCAD, CorelDraw и Adobe Creative Suite, не говоря уже о многочисленных специализированных приложениях, предназначенных для управления производственными процессами, учета финансовых ресурсов или информационной поддержки клиентов.

Этот практикум поможет изучить основные принципы программирования на языке и получить необходимые навыки для создания собственных программ на этом языке. Ее цель — помочь пользователю самостоятельно научиться программировать на языке VBA. Для восприятия материала книги не требуется знакомства с другими языками программирования и наличия программистского опыта.

Вместе с языком программирования VBA вы изучите мощную и гибкую среду разработки, которая является прекрасным выбором при работе над каким-либо проектом, предназначенным для использования в среде Microsoft Windows. Некоторые разделы практикума посвящены работе с объектами Excel. В практикуме приведена справочная информация по этим объектам. Для закрепления материала в конце каждой лабораторной работы предлагаются задания для самостоятельного выполнения.

Лабораторная работа № 1
Макросы

Макрос — это программа, состоящая из списка команд, которые должны быть выполнены приложением. Макрос служит для объединения нескольких различных действий в одну процедуру, которую вы можете легко вызвать. Этот список команд состоит в основном из *макрокоманд,* которые тесно связаны с командами приложения, в котором вы создаете макрос — т. е. с командами Word, Excel или других приложений Microsoft Office. Запись макросов позволяет не просто запомнить последовательность вызовов команд меню, нажатий на кнопки мыши и ввод данных с клавиатуры, но и перевести эти действия на объектно-ориентированный язык программирования Visual Basic для приложений (Visual Basic for Applications, в дальнейшем просто VBA) и сохранить их в виде готовой к выполнению программы.

VBA является полноценным языком программирования, позволяющим записать не только последовательно выполняемые пользователем действия, но и содержащим все необходимые конструкции языка программирования высокого уровня, включая разнообразные средства организации ветвлений, циклов и ведения диалога с пользователем. Весьма удобный редактор VBA позволяет не только писать и редактировать программы, но и вести их отладку.

Можно выделить следующие разновидности макросов:

- *Командные макросы* — это наиболее распространенные макросы, обычно состоящие из операторов, эквивалентных тем или иным командам меню или параметрам диалоговых окон. Основным предназначением такого макроса является выполнение действий, аналогичных командам меню — т. е. изменение окружения и основных объектов приложения
- *Пользовательские функции* — работают аналогично встроенным функциям Excel. Отличие этих функций от командных макросов состоит в том, что они используют значения передаваемых им аргументов, производят некоторые вычисления и возвращают результат в точку вызова, но не изменяют среды приложения
- *Макрофункции* — представляют собой сочетание командных макросов и пользовательских функций. Наряду с тем, что они могут использовать аргументы и возвращать результат, подобно пользовательским функциям, они могут также изменять среду приложения, как и командные макросы. Чаще всего эти макросы вызываются из других макросов, и активно используются для модульного программирования. Если необходимо выполнить ряд одинаковых действий в различных макросах, то обычно эти действия выделяются в отдельную макрофункцию (подпрограмму), которая вызывается всякий раз, когда необходимо выполнить эти повторяющиеся действия.

Автоматическая запись макросов

Для автоматической записи макроса надо выполнить следующие действия:

1. В обычном режиме выполнить команды, которые надо сохранить в макросе. Это надо сделать, чтобы четко выполнить последовательность команд.
2. Перейти в режим записи макроса.
3. Задать имя макроса.
4. Выполнить необходимую последовательность команд.
5. Выйти из режима записи макроса.

Рассмотрим подробнее пункты со 2 по 5.

Для переход в режим записи макроса надо на вкладке Разработчик выполнить команду **Макрос → Запись макроса**. В появившемся окне задать имя макроса.

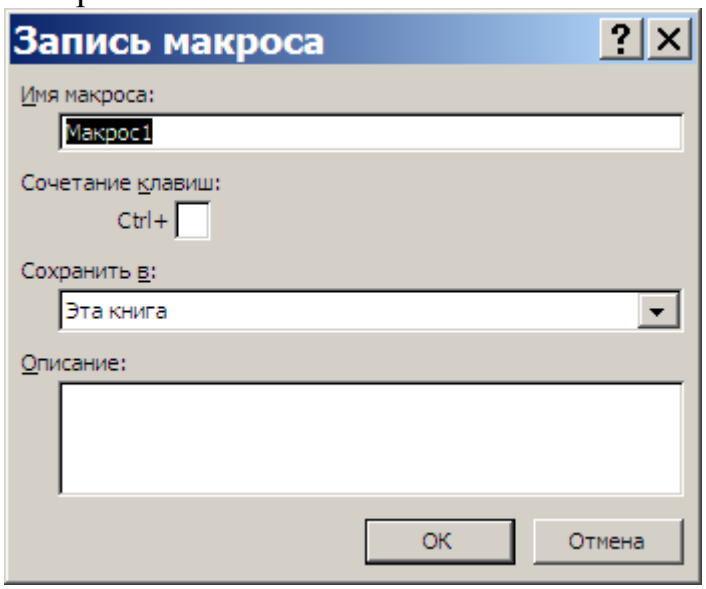


Рис.1

Имя макроса может содержать буквы и цифры и не может содержать пробелы. Можно связать выполнение макроса с комбинацией клавиш. Эта комбинация так же задается в окне записи макроса. В этом же окне можно описать, что делает данный макрос.

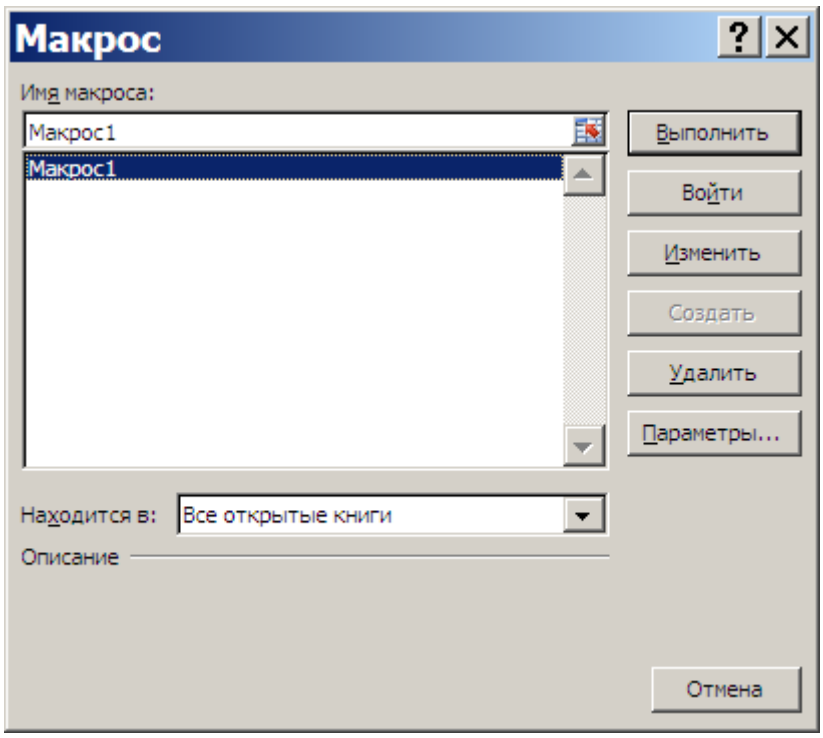
После щелчка по кнопке OK. Выполняется последовательность команд, которую надо запомнить в макросе.

В конце выполняется команда Макрос → Остановить запись.

Вызвать макрос можно двумя способами.

1 способ

Выполнить команду **Макросы → Макросы**. В окне Макрос (Рис.2) выбрать нужный макрос и щелкнуть по кнопке Выполнить.



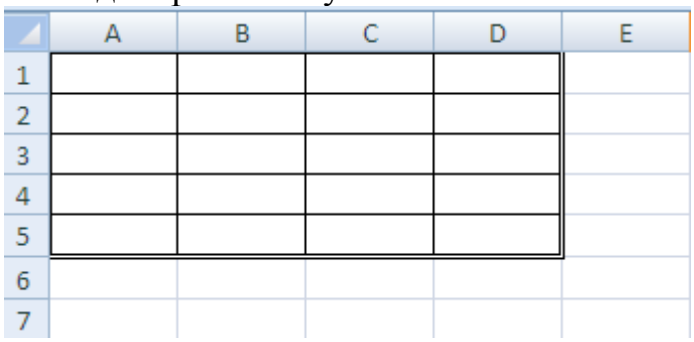
В этом же окне можно удалить макрос, который работает не правильно.

2 способ

Нажать комбинацию клавиш, связанную с данным макросом.

Задание 1

Создать макрос, который выделяет диапазон таблицы A1:D5, и выполняет его обрамление. Внешние границы обрамления - двойная линия. Внутренние линии - одинарные. Результат:



Имя макроса – *обрамление1*. Комбинация клавиш: *Ctrl+й*.

После записи макроса перейдите на новый лист и вызовите макрос любым из описанных способов. Обратите внимание, что во всех случаях выполнение макроса связано с конкретным диапазоном A1:D5, но не привязано к листу данной книги.

Можно записать командный макрос привязанный не к конкретным ячейкам, а к положению курсора. Для этого надо записывать макрос в режиме относительной адресации.

Задание 2

Создать макрос, который выполняет обрамление 4 столбцов и 5 строк. Внешние границы обрамления - двойная линия. Внутренние линии - одинарные. Порядок выполнения:

1. Установите курсор в ячейку A1.
2. Выполните команду Макрос → Запись макроса.
3. Задайте имя макроса обрамление2. комбинация клавиш *Ctrl+ц*.
4. После щелчка по кнопке OK выполните команду **Макрос → Относительные ссылки**.
5. Выделите диапазон A1:D5.
6. Выполните заданное обрамление диапазона.
7. Выполните команду Макрос → Остановить запись.

После записи макроса поставьте курсор в ячейку D8 и выполните макрос *обрамление2*.

В обоих заданиях выделение диапазона выполнялось в макросе. Это приводило к тома, что обрамление выполнялось для диапазона определенных размеров – 4 столбца и 5 строк. Если выделение диапазона выполнять до записи макроса, то размер диапазона обрамления будет произволен.

Задание 3

Выполните обрамление диапазона A1:D5. Но сначала надо выделить данный диапазон, а потом выполнить команду **Макрос → Запись макроса.** Задайте имя макроса *обрамление3*, комбинация клавиш – *Ctrl+y*.

После записи макроса выделите ячейки B10:K12 и вызовите макрос *обрамление3*.

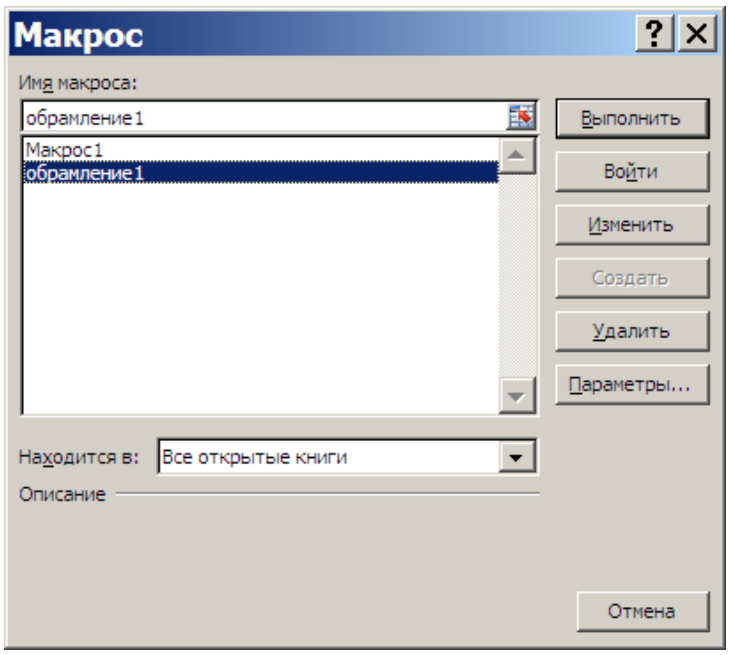
Сохраните книгу под именем Макросы.

Просмотр макросов.

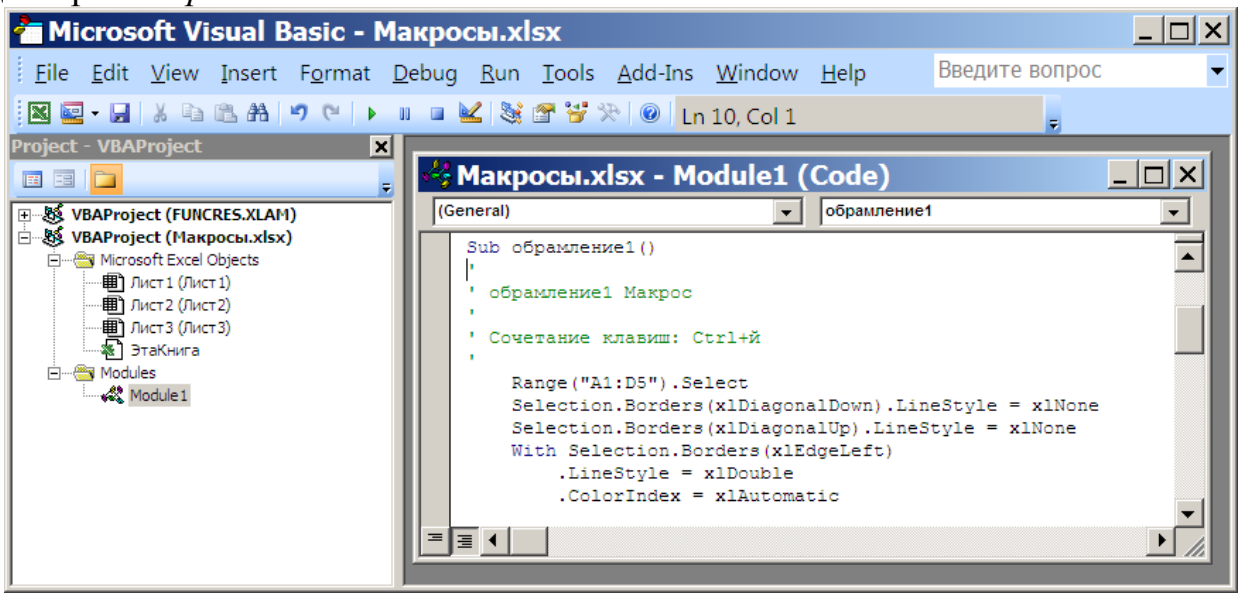
При записи макросов Макрорекодер записывает все выполненные команды в макрос. Эти макросы находятся в модулях и можно увидеть текст этих макросов и, при необходимости, изменить. Это широко используется в программировании VBA. Нет необходимости точно запоминать команды форматирования, копирования и т.д. Достаточно записать их в командный макрос, а потом изменить так как надо. И в нашем случае не надо помнить команды обрамления на языке VBA. Макрорекодер их сам запишет.

Для просмотра текста макроса надо:

1. Выполнить команду **Макрос→Макрос**.
2. Выбрать нужный макрос.
3. Щелкнуть по кнопке **Изменить**.



После этого откроется окно Visual Basic и в окне Модуля1 будет выведен код макроса *обрамление1*.



Любой макрос имеет следующую структуру:

Sub НазваниеМакроса()

Команда VBA

Команда VBA

. . .

End Sub

Текст зеленого цвета – это комментарии. Комментарии начинаются со знака «апостроф» и не выполняются компьютером.

Рассмотрим код макроса *обрамление1*.

```
Sub обрамление1()
'
' обрамление1 Макрос
'
' Сочетание клавиш: Ctrl+й
'
```

```
    Range("A1:D5").Select 'Выделяется диапазон A1:D5
'Все команды выполняются для выделенного диапазона
' диагонального обрамления нет
    Selection.Borders(xlDiagonalDown).LineStyle = xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
' задается стиль левой границы
    With Selection.Borders(xlEdgeLeft)
        .LineStyle = xlDouble 'двойная линия
        .ColorIndex = xlAutomatic
        .TintAndShade = 0
        .Weight = xlThick
    End With
' задается стиль верхней границы
    With Selection.Borders(xlEdgeTop)
        .LineStyle = xlDouble 'двойная линия
        .ColorIndex = xlAutomatic
        .TintAndShade = 0
        .Weight = xlThick
    End With
' задается стиль нижней границы
    With Selection.Borders(xlEdgeBottom)
        .LineStyle = xlDouble 'двойная линия
        .ColorIndex = xlAutomatic
        .TintAndShade = 0
        .Weight = xlThick
    End With
' задается стиль правой границы
    With Selection.Borders(xlEdgeRight)
        .LineStyle = xlDouble 'двойная линия
        .ColorIndex = xlAutomatic
        .TintAndShade = 0
        .Weight = xlThick
    End With
' задается стиль внутренней вертикальной
    With Selection.Borders(xlInsideVertical)
        .LineStyle = xlContinuous 'одинарная сплошная линия
        .ColorIndex = xlAutomatic
        .TintAndShade = 0
        .Weight = xlThin
    End With
' задается стиль внутренней горизонтальной
    With Selection.Borders(xlInsideHorizontal)
        .LineStyle = xlContinuous 'одинарная сплошная линия
        .ColorIndex = xlAutomatic
        .TintAndShade = 0
        .Weight = xlThin
    End With
End Sub
```

Рассмотрим код макроса *обрамление2*.

```
Sub обрамление2()
'
' обрамление2 Макрос
'
' Сочетание клавиш: Ctrl+ц
'
```

ActiveCell.Range("A1:D5").Select

‘от активной ячейки выделяется диапазон A1:D5

Остальной текст программы совпадает с *обрамление1*.

```
    Selection.Borders(xlDiagonalDown).LineStyle = xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    With Selection.Borders(xlEdgeLeft)
        .LineStyle = xlDouble
        .ColorIndex = xlAutomatic
        .TintAndShade = 0
        .Weight = xlThick
    End With
    With Selection.Borders(xlEdgeTop)
        .LineStyle = xlDouble
        .ColorIndex = xlAutomatic
        .TintAndShade = 0
        .Weight = xlThick
    End With
    With Selection.Borders(xlEdgeBottom)
        .LineStyle = xlDouble
        .ColorIndex = xlAutomatic
        .TintAndShade = 0
        .Weight = xlThick
    End With
    With Selection.Borders(xlEdgeRight)
        .LineStyle = xlDouble
        .ColorIndex = xlAutomatic
        .TintAndShade = 0
        .Weight = xlThick
    End With
    With Selection.Borders(xlInsideVertical)
        .LineStyle = xlContinuous
        .ColorIndex = xlAutomatic
        .TintAndShade = 0
        .Weight = xlThin
    End With
    With Selection.Borders(xlInsideHorizontal)
        .LineStyle = xlContinuous
        .ColorIndex = xlAutomatic
        .TintAndShade = 0
        .Weight = xlThin
    End With
End Sub
```

Код макроса обрамление 3

```
Sub обрамление3()
'
' обрамление3 Макрос
'
' Сочетание клавиш: Ctrl+y
'Все команды выполняются для выделенного диапазона
    Selection.Borders(xlDiagonalDown).LineStyle = xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    With Selection.Borders(xlEdgeLeft)
        .LineStyle = xlDouble
        .ColorIndex = xlAutomatic
        .TintAndShade = 0
        .Weight = xlThick
    End With
```

```
    With Selection.Borders(xlEdgeTop)
        .LineStyle = xlDouble
        .ColorIndex = xlAutomatic
        .TintAndShade = 0
        .Weight = xlThick
    End With
    With Selection.Borders(xlEdgeBottom)
        .LineStyle = xlDouble
        .ColorIndex = xlAutomatic
        .TintAndShade = 0
        .Weight = xlThick
    End With
    With Selection.Borders(xlEdgeRight)
        .LineStyle = xlDouble
        .ColorIndex = xlAutomatic
        .TintAndShade = 0
        .Weight = xlThick
    End With
    With Selection.Borders(xlInsideVertical)
        .LineStyle = xlContinuous
        .ColorIndex = xlAutomatic
        .TintAndShade = 0
        .Weight = xlThin
    End With
    With Selection.Borders(xlInsideHorizontal)
        .LineStyle = xlContinuous
        .ColorIndex = xlAutomatic
        .TintAndShade = 0
        .Weight = xlThin
    End With
End Sub
```

1. Задания для самостоятельной работы

2. Создать макрос, который объединяет выделенные ячейки и форматирует текст в этой объединенной ячейке по центру по горизонтали и по вертикали.
3. Создать макрос, который защищает лист с паролем.
4. Создать макрос, который снимает защиту листа с паролем.

Лабораторная работа № 2
Объекты листа

Свойство объекта Range

Свойство Range возвращает объект Range, определяемый аргументами. Используются два разных способа записи свойства Range.

Первый способ object.Range(Cell1)

Второй способ object.Range(Cell1 [,Cell2])

- object - ссылка на объект, например, на рабочий лист или на интервал ячеек. Ссылка необязательна. По умолчанию используется активный лист;

- Cell1, Cell2 - аргументы для задания интервала ячеек. Cell1 - указание обязательно при обоих способах записи свойства Range.

Первый способ

Аргумент Cell1 задает интервал ячеек произвольного размера.

ВАЖНО

- Могут использоваться имена, определенные в таблице, или координаты ячеек, столбцов, строк или интервалов.
- Координаты задаются в стиле A1.
- Координаты и имена заключаются в кавычки.
- При задании интервалов координаты левого верхнего угла и правого нижнего угла интервала разделяются двоеточием.
- Для задания несмежных интервалов используется запятая.
- Для задания пересечения интервалов используется пробел.

Примеры записи оператора Range (1 способ)

Запись	Возвращаемый объект
ActiveSheet.Range("A1:A10")	интервал ячеек A1:A10 на активном листе
Range("A:B")	столбцы A:B
Range("налог")	интервал с именем налог
Range("1:3")	строки с первой по третью
Range("A1:C2, B10:D24")	объединение двух несмежных интервалов A1:C2 и B10:D24
Range("A1:C10 B10:D24")	пересечение двух интервалов A1:C10 и B10:D24, т.е. интервал B10:C10

Второй способ

Аргументы задают координаты интервала:

- Cell1 - единственная ячейка (строка или столбец), задающая левый верхний угол интервала;
- Cell2 - единственная ячейка (строка или столбец), задающая правый нижний угол интервала. Необязательный аргумент.

Допустимо задание аргументов переменными, выражениями, свойствами или методами, представляющими объект Range - одну ячейку, одну строку или один столбец рабочего листа.

Примеры записи оператора Range (2 способ)

Запись	Возвращаемый объект
Range("A5","D18")	интервал A5:D18
Range(Columns(1), Columns(5))	интервал, содержащий первые пять столбцов рабочего листа

ЗАПОМНИТЕ

- Если свойство Range применяется к объекту Range, то ссылка на интервал ячеек считается относительной и возвращается смещенный объект Range.

Например, если выделен интервал **C1:D5**, то запись Selection.Range("B2") возвратит ячейку **D2**.

Свойство Cells

Свойство Cells возвращает единственную ячейку рабочего листа, которая находится на пересечении строки и столбца, задаваемых целыми числами.

Синтаксис object.Cells (RowIndex,ColumnIndex)

- object - ссылка на объект. Ссылка необязательна. По умолчанию используется активный лист;
- RowIndex - индекс строки;
- ColumnIndex - индекс столбца.

ЗАМЕЧАНИЯ

- В свойстве Cells индекс строки является первым аргументом, а индекс столбца - вторым аргументом, тогда как при задании адреса ячейки в стиле A1 сначала указывается столбец, а затем строка.
- Понятие "индекс" (Index, ColumnIndex, RowIndex) всегда подразумевает целое число, целочисленную переменную или выражение, результат вычисления которого есть целое число или может быть преобразован в целое число.

Примеры записи свойства Cells

Запись	Комментарий	Возвращаемый объект
ActiveSheet.Cells	Свойство Cells без аргументов	все ячейки активного рабочего листа
Range("C5:C10").Cells(1,1)	Свойство Cells применяется к объекту Range (относительная ссылка)	ячейка C5
Range(Cells(7,3),Cells(10,4))	Свойство Cells используется в качестве аргументов свойства Range	интервал ячеек C7:D10

Свойство Offset

Свойство Offset позволяет задавать ячейки или интервалы при помощи числа строк и колонок, которые отделяют нужную ячейку от исходной ячейки, т.е. указывая смещение относительно выбранной ячейки. Например, Range("A5").Offset(-2,1) возвращает ячейку **B3**.

Синтаксис object.Offset([RowOffset][,ColumnOffset])

- object - ссылка на объект Range. Ссылка обязательна и определяет объект, относительно которого задается смещение;
- RowOffset - смещение строки искомой ячейки относительно исходной ячейки;
- ColumnOffset - смещение столбца искомой ячейки относительно исходной ячейки.

Необязательные аргументы RowOffset и ColumnOffset - числовые выражения. Если какой-то аргумент не задан, то соответствующее смещение равно нулю.

Например, если выделен интервал **C1:D5**, то запись Selection.Offset(2,1).Select выделяет интервал **D3:E7**.

Метод Union и свойство Areas

Метод Union используется для объединения двух и более объектов Range, заданных ссылками на непересекающиеся интервалы, в один объект Range.

Синтаксис Object.Union (arg1,arg2,...)

object - всегда объект Application. Ссылка необязательна;

arg1,arg2 - интервалы ячеек. Количество аргументов произвольно. Обязательно наличие хотя бы двух аргументов.

Например, оператор Union(Range("A1:C5"),Range("B10:D12")).Select выделяет несмежные интервалы **A1:C5** и **B10:D12**.

Свойство Areas выполняет обратное действие, разделяя объединенные интервалы на несколько объектов Range.

Синтаксис Object.Areas(index)

- object - ссылка на объект Range, состоящий из нескольких интервалов;
- index - номер интервала в объекте. Аргумент необязателен.

Примеры

Оператор	Комментарий	Результат
p=Union (Range("A1:C5"), Range("B10:D12")).Areas(2).Count	Если аргумент задан, то свойство Areas возвращает интервал - объект Range, определенный индексом интервала	равен девяти, так как во втором интервале ровно 9 ячеек
p=Union(Range("A1:C5"), Range("B10:D12")).Areas.Count	Свойство Areas без аргументов рассматривает каждый из несмежных интервалов как элемент коллекции объектов Range	равен двум, так как объект, определенный методом Union, состоит из двух областей - коллекции из двух элементов

p=Range("B10:D12").Areas.Count		равен единице, так как объект Range представляет один элемент коллекции

Свойства Column и Row (R/O Integer)

Свойства возвращают целое число, показывающее индекс первого столбца или первой строки соответственно для заданного объекта. Синтаксис свойств

object.Column

object Row

object - обязательная ссылка на объект Range.

Например, запись Range("C5").Column возвращает число 3, а запись Range("C5").Row возвращает число 5.

Свойства Columns и Rows

Свойство Columns (не путайте со свойством Column!) возвращает объект Range, представляющий колонку или коллекцию колонок в объекте, к которому это свойство было применено.

Синтаксис Object.Columns(index)

object - ссылка на объект. Указание необязательно, по умолчанию используется активный рабочий лист;

index - индекс колонки в объекте.

Например, запись Columns(1) возвращает колонку **A** активного рабочего листа, а запись Range("C1:D5").Columns(1) возвращает колонку **C** заданного интервала, а именно, ячейки **C1:C5**.

ВАЖНО

- Если не указан индекс колонки, то возвращаются все колонки объекта в виде объекта Range.
- Индекс колонки можно указывать числом или буквой, при этом буква заключается в кавычки. Ссылки Columns(2) и Columns("B") указывают на одну и ту же колонку **B**.

Свойство Rows (не путайте со свойством Row!) возвращает объект Range, представляющий строку или коллекцию строк в объекте, к которому это свойство было применено.

Синтаксис Object.Rows(index)

- object - ссылка на объект. Указание необязательно, по умолчанию используется активный рабочий лист;
- index - индекс строки в объекте.

ВАЖНО

- Если не указан номер строки, то возвращаются все строки объекта в виде объекта Range.

Например, оператор nr=Selection.Rows(Selection.Rows.Count).Row позволяет получить номер последней строки в выделенном интервале ячеек.

Свойство CurrentRegion

Текущий регион (CurrentRegion) - это диапазон ячеек, ограниченный пустыми строками и колонками или сочетанием пустых строк, колонок и границ рабочего листа.Пример

В процедуре сравниваются значения первой ячейки первой строки и первой ячейки каждой следующей строки заполненного данными интервала, включающего первую ячейку. Если значения совпадают, то очередная строка удаляется.

Предполагается, что данные начинаются с ячейки **A1** и занимают несколько строк и столбцов, при этом расположены не плотно, т.е. внутри интервала с данными могут находиться пустые строки или пустые столбцы. Анализируются только строки заполненного данными интервала ячеек вокруг ячейки **A1**, не содержащего пустых строк и столбцов.

Свойства, связанные с шириной и высотой ячейки

Свойства	Примеры и комментарии
ColumnWidth (R/W Variant)	Возвращает или изменяет ширину колонки в единицах, эквивалентных одному символу в стиле Обычный (Normal). Шрифт стиля по умолчанию Arial Cyr и размер шрифта 10. Range("A1").ColumnWidth=15 устанавливает ширину колонки A в 15 символов
Width (R/O Variant)	Возвращает ширину интервала ячеек в пунктах. Range("A1").Width возвращает значение 93.75, если ширина колонки 15 символов, шрифт Times New Roman, размер шрифта 12 пунктов (72 пункта равны 1 дюйму или приблизительно 2,54 см). Debug.Print Range("A1:C3").ColumnWidth распечатает значение 8.43, а оператор Debug.Print Range("A1:C3").Width распечатает значение 144, если для колонок установлена стандартная ширина, шрифт Arial Cyr и размер шрифта 10
RowHeight (R/W Variant)	Возвращает или изменяет высоту строк интервала в пунктах. ActiveCell.RowHeight = 14 устанавливает высоту строки, в которой находится активная ячейка, в 14 пунктов
Height (R/O Variant)	Возвращает суммарную высоту интервала строк, зависящую от названия и размера шрифта. Если шрифт Arial Cyr и размер шрифта 10, то Debug.Print Range("A1").Height распечатает 12,75 и Debug.Print Range("A1:C3").Height распечатает 38,25
WrapText (R/W Boolean)	Range("A1").WrapText=True Значение True разбивает текст ячейки на несколько строк, если

	ширина столбца недостаточна для размещения текста целиком

ЗАМЕЧАНИЕ

- Свойства Width и Height имеют статус Read-Only для объектов Range, но для других объектов, например, для объекта Window, они имеют статус Read-Write.

Свойство Address

Address — позволяет вернуть адрес текущего диапазона. Этому свойству можно передать много параметров — для определения стиля ссылки, абсолютного или относительного адреса для столбцов и строк, по отношению к чему этот адрес будет относительным и т.п. Свойство доступно только для чтения. AddressLocal — то же самое, но с поправкой на особенности локализованных версий Excel.

На практике встречается множество ситуаций, когда адрес ячейки нужно разобрать на части и вернуть из него имя столбца или номер строки. Это очень просто сделать при помощи строковых функций

Методы Select и Activate

Метод Select выделяет интервал ячеек.

Синтаксис object.Select(Replace)

- object - выделяемый объект типа Range. Ссылка на объект обязательна;
- Replace - для расширения выделения аргумент устанавливается в False. Если аргумент не задан или принимает значение True, то вместо старой области выделения создается новая область выделения. Необязательный параметр.

Метод Activate активизирует единственную ячейку.

Синтаксис object.Activate

- object - активизируемая ячейка. Ссылка на объект обязательна.

Примеры

Оператор	Активная ячейка
Range("C7:E9").Select	C7
Range("C7:E9").Offset(1,1).Activate	D8
Range("C7:E9").Activate	C7
Range("C7:E9").Cells(2,1).Activate	C8

ЗАМЕЧАНИЯ

- Активная ячейка выделяется фоном среди всех выделенных ячеек.
- Метод Select выделяет интервал ячеек, тогда как метод Activate активизирует только одну ячейку.
- При использовании метода Select первая ячейка интервала становится активной.

- Если выделена только одна ячейка, то она является активной и свойства ActiveCell и Selection возвращают одну и ту же ячейку (объект Range).

Метод Clear

Очищает интервал ячеек, изменяя, таким образом, свойство Value каждой ячейки интервала.

Задание 1

Создать 2 макроса.

1. Макрос ***оформление***. Выполнить обрамление для созданной таблицы. Залить первую строку и задать для нее жирный шрифт.

2. Макрос ***ряд.*** Для выделенного диапазона (столбца) получить натуральный ряд.

Первоначальный вариант макроса оформление.

```
Sub оформление()
Selection.Borders(xlDiagonalDown).LineStyle = xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    With Selection.Borders(xlEdgeLeft)
        .LineStyle = xlDouble
        .ColorIndex = xlAutomatic
        .TintAndShade = 0
        .Weight = xlThick
    End With
    With Selection.Borders(xlEdgeTop)
        .LineStyle = xlDouble
        .ColorIndex = xlAutomatic
        .TintAndShade = 0
        .Weight = xlThick
    End With
    With Selection.Borders(xlEdgeBottom)
        .LineStyle = xlDouble
        .ColorIndex = xlAutomatic
        .TintAndShade = 0
        .Weight = xlThick
    End With
    With Selection.Borders(xlEdgeRight)
        .LineStyle = xlDouble
        .ColorIndex = xlAutomatic
        .TintAndShade = 0
        .Weight = xlThick
    End With
    With Selection.Borders(xlInsideVertical)
        .LineStyle = xlContinuous
        .ColorIndex = xlAutomatic
        .TintAndShade = 0
        .Weight = xlThin
    End With
    With Selection.Borders(xlInsideHorizontal)
        .LineStyle = xlContinuous
        .ColorIndex = xlAutomatic
        .TintAndShade = 0
        .Weight = xlThin
```

```
    End With
    Range("E6:H6").Select
    With Selection.Interior
        .Pattern = xlSolid
        .PatternColorIndex = xlAutomatic
        .Color = 65535
        .TintAndShade = 0
        .PatternTintAndShade = 0
    End With
    Selection.Font.Bold = True
End Sub
```

Заливка строки в данной макросе будет выполняться не правильно для произвольной таблицы, т.к. выделение привязывается к конкретному диапазону. (выделенная строка)

Для решения этой задачи надо запомнить первоначально выделенный диапазон в объектной переменной в начале макроса.

```
Dim r As Range
Set r = Selection
```

Перед заливкой надо выделить первую строку этого диапазона, а затем выполнить заливку. Для этого надо вместо выделенной строки в тексте макроса выполнить команду:

```
r.rows(1).Select
```

Окончательный вариант макроса ***оформление***.

```
Sub оформление()
Dim r As Range
Set r = Selection
Selection.Borders(xlDiagonalDown).LineStyle = xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    With Selection.Borders(xlEdgeLeft)
        .LineStyle = xlDouble
        .ColorIndex = xlAutomatic
        .TintAndShade = 0
        .Weight = xlThick
    End With
    With Selection.Borders(xlEdgeTop)
        .LineStyle = xlDouble
        .ColorIndex = xlAutomatic
        .TintAndShade = 0
        .Weight = xlThick
    End With
    With Selection.Borders(xlEdgeBottom)
        .LineStyle = xlDouble
        .ColorIndex = xlAutomatic
        .TintAndShade = 0
        .Weight = xlThick
    End With
    With Selection.Borders(xlEdgeRight)
        .LineStyle = xlDouble
        .ColorIndex = xlAutomatic
        .TintAndShade = 0
        .Weight = xlThick
    End With
    With Selection.Borders(xlInsideVertical)
        .LineStyle = xlContinuous
```

```
            .ColorIndex = xlAutomatic
            .TintAndShade = 0
            .Weight = xlThin
        End With
        With Selection.Borders(xlInsideHorizontal)
            .LineStyle = xlContinuous
            .ColorIndex = xlAutomatic
            .TintAndShade = 0
            .Weight = xlThin
        End With
       r.Rows(1).Select
        With Selection.Interior
            .Pattern = xlSolid
            .PatternColorIndex = xlAutomatic
            .Color = 65535
            .TintAndShade = 0
            .PatternTintAndShade = 0
        End With
        Selection.Font.Bold = True
    End Sub
```

Для получения макроса ***ряд*** натуральные числа следует получить через команду прогрессия.

Первоначальный вариант макроса таблица

```
    Sub ряд()
        ActiveCell.FormulaR1C1 = "1"
        Range("E6:E24").Select
        Selection.DataSeries     Rowcol:=xlColumns,     Type:=xlLinear,
Date:=xlDay, _
            Step:=1, Trend:=False
    End Sub
```

Для создания макроса для произвольного диапазона следует исключить конкретные адреса.

Окончательный вариант макроса ряд

```
    Sub ряд()
      Dim r As Range
      Set r = Selection ‘Сохранение выделенного диапазона
      n = r.Row 'Номер верхнего ряда
      m = r.Column 'Номер левого столбца
      k = r.Rows.Count 'Количество выделенных рядов
      Cells(n, m).Select
      ActiveCell.FormulaR1C1 = "1"
      Range(Cells(n, m), Cells(n + k, m)).Select
      Selection.DataSeries      Rowcol:=xlColumns,      Type:=xlLinear,
Date:=xlDay, _
            Step:=1, Trend:=False
    End Sub
```

Задание 2

В некоторых командах надо указывать конкретный адрес диапазона. Для этого используется свойства Address.

Создать макрос ***сумма***, который суммирует числа в выделенном диапазоне и помещает результат под этим диапазоном в последний столбец.

Заполним диапазон какими-нибудь числами. В макросе вызовем функцию суммирования и укажем диапазон суммирования.

В результате макрос будет иметь вид:

```
Sub Сумма ()
    Range("K20").Select
    ActiveCell.FormulaR1C1 = "=SUM(R[-14]C[-5]:R[-1]C[-1])"
    Range("K21").Select
End Sub
```

Из текста макроса видно, что преобразовать данный макрос к нужной форме достаточно сложно.

Для создания данного макроса надо использовать функцию СУММ(диапазон суммирования). Для определения диапазона суммирования необходимо выделить этот диапазон перед началом записи макроса, а затем в макросе определить адрес этого диапазона, его местоположение и размеры, для определения местоположения ячейки, где будет находится сумма.

Макрос будет иметь вид:

```
Sub Сумма ()
    s = Selection.Address 'определение адреса диапазона
    n = Selection.Row 'номер ряда начальной ячейки диапазона
    m = Selection.Column 'номер колонки начальной ячейки диапазона
    a = Selection.Rows.Count 'количество рядов диапазоне
    b = Selection.Columns.Count 'количество столбцов диапазоне
    Cells(n + a, m + b).FormulaLocal = "=СУММ(" + s + ")"
End Sub
```

В макросе используется свойство FormulaLocal, потому, что используется функция на русском языке.

Еще один пример использования свойства Address.

Создать макрос ***диаграмма***, в котором построить диаграмму по выделенному диапазону и разместить эту диаграмму на отдельном листе.

Создаем диапазон с числовыми данными.

До записи макроса выделяем диапазон. В макросе указываем тип диаграммы (объемная гистограмма), убираем легенду и размещаем диаграмму на отдельном листе.

Записанный макрос имеет вид:

```
Sub Диаграмма()
    ActiveSheet.Shapes.AddChart.Select
    ActiveChart.SetSourceData Source:=Range("'Лист1'!$A$1:$A$7")
    ActiveChart.ChartType = xl3DColumnClustered
    ActiveChart.SetElement (msoElementLegendNone)
    ActiveChart.Location Where:=xlLocationAsNewSheet
End Sub
```

В данном макросе конкретный адрес диапазона данных находится в выделенной строке. Поэтому надо определить адрес выделенного диапазона и подставить его в команду.

```
Sub Диаграмма()
     s = Selection.Address
    ActiveSheet.Shapes.AddChart.Select
    ActiveChart.SetSourceData Source:=Range(s)
    ActiveChart.ChartType = xl3DColumnClustered
    ActiveChart.SetElement (msoElementLegendNone)
    ActiveChart.Location Where:=xlLocationAsNewSheet
End Sub
```

Задания для самостоятельной работы

1. Создать макрос итог, к тором для созданной таблицы выполнить форматирование:

1. Для столбцов выполнить автоподбор ширины.
2. Выполнить обрамление таблицы.
3. Для шапки выполнить заливку и задать полужирный шрифт.
4. для строки над таблицей задать объединение ячеек и центрирование по вертикали и по горизонтали.
5. Для последнего столбца задать денежный формат.
6. Под последней колонкой поместить функцию суммирования и выделить ее жирным шрифтом..
7. Объединить ячейки перед суммой на написать туда слово ИТОГО жирным шрифтом.

2. Создать макрос, который строит график функции по заданным данным. На графике изобразить линии вертикальной и горизонтальной сетки. Убрать легенду.

Лабораторная работа № 3
Функции

При записи строковой константы ее надо заключать в кавычки. В выражениях, наряду с константами могут встречаться и функции.

Функция (function) — это встроенная формула, выполняющая действия над выражениями и генерирующая значение. Функция всегда возвращает значение, которое VBA вставляет в программу в том месте, где появляется имя функции. Функции VBA делятся на несколько групп в зависимости от типа операции или вычисления, которое они выполняют.

Чтобы использовать функцию, надо просто ввести имя функции в оператор VBA вместе с любыми аргументами, которые требуются для этой функции, в том месте в операторе, где необходимо использовать результат функции. [Помещение имени функции в оператор VBA для активизации функции называют *вызовом (calling)* функции.] При использовании функций в выражениях существуют следующие правила:

- Можно использовать результат функции как часть выражения.
- Можно присваивать результат функции какой-либо переменной.
- Можно использовать результат функции для предоставления значения в список аргументов другой процедуры или функции.
- Функции имеют списки аргументов, заключенные в круглые скобки.

В основном функцию можно использовать для предоставления значения в любом месте в любом операторе VBA, где может быть оправданно использование значения константы или переменной. Тип данных значения, возвращаемого функцией, зависит от этой конкретной функции.

Большинство функций возвращают значения типа Variant, хотя некоторые функции возвращают данные определенных типов, таких как String, Double и Integer. VBA во многих случаях может автоматически преобразовывать результат какой-либо функции в данные типа, совместимого с другими типами значений в выражении, содержащем эту функцию, точно, как VBA преобразует типы данных в присваиваниях переменных и вычислении выражений.

Встроенные функции VBA делятся на несколько категорий на основе общего назначения функций (математические, преобразования данных, даты и времени, строковые и работы с диском). Далее обсуждаются категории функций и описываются их действия. Большинство функций VBA, такие как математические функции, являются довольно ясными из их названия и не требуют подробного объяснения. Другие функции, такие как функции преобразования типа данных и обработки строк, описаны более подробно. VBA-функции обработки строк имеют важное значение, поэтому рассмотрены способы их использования.

Математические функции

VBA предоставляет стандартный набор математических функций. В табл. 2 приведены математические функции, имеющиеся в VBA. В этой таблице N означает любое численное выражение; все аргументы функций являются обязательными, если только не указано иначе.

Таблица 2

Функции (аргументы)	Возвращает /действие
Abs(N)	Возвращает абсолютное значение N.
Atn(N)	Возвращает арктангенс N как угол в радианах.
Cos(N)	Косинус угла N, где N — это угол, измеренный в радианах.
Exp(N)	Возвращает константу *e,* возведенную в степень N. (e — это основание натуральных логарифмов и она (приблизительно) равна 2,718282).
Fix(N)	Возвращает целую часть N. Fix не округляет число, а отбрасывает любую дробную часть. Если N является отрицательным, Fix возвращает ближайшее отрицательное целое большее, чем или равное N.
Int(N)	Возвращает целую часть N. Int не округляет число, а отбрасывает любую дробную часть. Если N является отрицательным, Int возвращает ближайшее отрицательное целое меньшее, чем или равное N.
Log(N)	Возвращает натуральный логарифм N.
Rnd(N)	Возвращает случайное число; аргумент является необязательным. Используйте функцию Rnd только после инициализации VBA-генератора случайных чисел оператором Randomize.
Sgn(N)	Возвращает знак числа: -1, если N — отрицательное; 1, если N — положительное; 0, если N равно 0.
Sin(N)	Возвращает синус угла; N — это угол, измеренный в радианах.
Sqr(N)	Возвращает корень квадратный из N. VBA отображает ошибку времени исполнения, если N — отрицательное.
Tan(N)	Возвращает тангенс угла; N — угол в радианах.

Функции **Fix** и **Int** укорачивают целые, то есть они отбрасывают дробную часть числа без округления. Единственное различие между функциями **Fix** и **Int** — это то, как они обрабатывают отрицательные числа.

Дополнительные тригонометрические функции можно выводить из базовых математических функций VBA. Например, если необходимо вычислить котангенс угла, для его нахождения можно использовать формулу **1/Tan(x).**

Примеры использования математических функций:

```
gipot=Sqr(kat2^2+kat2^2)
chislo=Int(Rnd(1)*100)
```

Функции преобразования данных

Visual Basic предоставляет несколько функций для преобразования одного типа данных в другой. Надо использовать эти функции для устранения ошибок несовпадения типов и обеспечения явного контроля за типами данных в выражениях.

Например, при получении сообщения об ошибке несовпадения типов в определенном выражении можно преобразовать значения в выражении в типы, совместимые друг с другом, используя функции преобразования. Или же можно сохранять результат выражения в диапазоне численного типа **Single** (большинство численных выражений имеют результатом значение типа **Double);** в таком случае следует использовать функцию **CSng** для преобразования результата выражения в число типа **Single.**

В табл. 3 приведены функции преобразования данных в VBA. В этой таблице N — это любое численное, S — любое строковое, а E — выражение любого типа. Аргументы каждой функции являются обязательными, если не указано иначе.

Таблица 3

Функция (аргументы)	Возвращает/действие
Asc(S)	Возвращает число кода символа, соответствующее первой букве строки S. Буква "A", например, имеет код символа 65.
Chr(N)	Возвращает строку из одного символа, соответствующего коду символа N, который должен быть числом между 0 и 255, включительно. Код символа 65, например, возвращает букву "A".
Format(E, S)	Возвращает строку, содержащую значение, представленное выражением E, в формате в соответствии с инструкциями, содержащимися в S.
Hex(N)	Возвращает строку, содержащую шестнадцатиричное представление N.
Oct(N)	Возвращает строку, содержащую восьмиричное представление N.
RGB(N, N, N)	Возвращает целое типа **Long,** представляющее значение основных цветов изображения. N в каждом аргументе должно быть целым в диапазоне 0 — 255, включительно. Аргументы (слева направо) — это значения для красного, зеленого и синего цвета.
Str(N)	Возвращает строку, эквивалентную численному выражению N.
Val(S)	Возвращает численное значение, соответствующее числу, представленному строкой S, которая должна содержать только цифры и одну десятичную точку, иначе VBA не может преобразовать ее в число. Если VBA не может

	преобразовать строку в S, то функция **Val** возвращает 0.
CBool(N)	Возвращает Boolean-эквивалент.численного выражения N.
CByte(E)	Возвращает численное значение типа **Byte** (от 0 до 255); E — любое допустимое численное или строковое выражение, которое может быть преобразовано в число.
CCur(E)	Возвращает численное значение типа **Currency;** E — любое допустимое численное или строковое выражение, которое может быть преобразовано в число.
CDate(E)	Возвращает значение типа **Date.** E может быть любым допустимым выражением (строкой или числом), представляющим дату в диапазоне 1/1/100 — 12/31/9999, включительно.
CDbl(E)	Возвращает численное значение типа **Double;** E — любое допустимое численное или строковое выражение, которое может быть преобразовано в число.
CInt(E)	Возвращает численное значение типа **Integer;** E — любое допустимое численное или строковое выражение, которое может быть преобразовано в число.
CLng(E)	Возвращает численное значение типа **Long;** E — любое допустимое численное или строковое выражение, которое может быть преобразовано в число.
CSng(E)	Возвращает численное значение типа **Single;** E — любое допустимое численное или строковое выражение, которое может быть преобразовано в число.
CStr(E)	Возвращает значение типа **String;** E — любое допустимое численное или строковое выражение.
CVar(E)	Возвращает значение типа **Variant;** E — любое допустимое численное или строковое выражение.

Наиболее часто используемые функции — это функции (объединенные в конце табл. 3 в группу), начинающиеся с буквы C (от слова *conversion),* за которыми следует сокращение имени типа: **CStr, CSng, CDbl** и так далее.

Примеры использования функций преобразования данных:

```
d=CDat("1.9.2006")
text=CStr(36*3)
```

Функции даты и времени

Язык VBA содержит специальный тип данных Date, предназначенный для представления значений даты и времени. Внутри программ значения типа Date представлены как числа с плавающей запятой:

- целая часть значения отображает дату как количество дней, прошедшее с 30 декабря 1899 года;

- дробная часть значения представляет время в виде доли 24-часового дня (например, .25 — это 6 часов утра, 5 — это полдень и т.д.);
- дробное число без целой части отображает время без даты.

Для пользователя или программиста представленные таким образом значения даты и времени являются бесполезными. К счастью, в языке VBA предусмотрены инструменты, позволяющие преобразовывать эти значения к более знакомому пользователям виду. Исходное "сырое" представление значений даты и времени в виде чисел с плавающей запятой скрыто от посторонних глаз и используется программой только в процессе проведения внутренних вычислений.

Язык VBA способен распознавать значения даты и времени, представленные практически в любом стандартном формате. Чтобы указать на текстовые константы как на значения даты/времени, необходимо заключить их в символы #. Используйте обычный оператор присваивания, чтобы присвоить переменной типа Date значение даты/времени:

dt = #01.05.2003#

VBA-функции даты и времени обычно используются для получения текущей даты и времени, разбиения значения даты на ее составляющие части или для преобразования строк и чисел в значения типа **Date.** В таблице 4 приведены VBA-функции даты и времени и их действие. В этой таблице N — любое допустимое численное выражение, а D — любое допустимое выражения типа **Date** (включая значения типа **Date,** числа или строки, которые VBA может преобразовать в дату). Все аргументы функций в этой таблице являются обязательными, если не указано иначе.

Таблица 4

Функции(аргументы)	Возвращает/действие
Date	Возвращает системную дату. Можно также использовать эту функцию как процедуру для установки системных часов компьютера. Более подробно можно узнать из справочной системы VBA.
Time	Возвращает системное время компьютера как значение типа **Date.** Можно также использовать эту функцию как процедуру для установки системных часов. Более подробно можно узнать из справочной системы VBA.
Now	Возвращает системную дату и время.
Year(D)	Возвращает целое, являющееся частью выражения типа **Date** и содержащее год. Год возвращается как число между 100 и 9999.
Month(D)	Возвращает целое, являющееся частью выражения типа **Date,** содержащее месяц. Месяц возвращается как число между 1 и 12, включительно.

Day(D)	Возвращает целое, являющееся частью выражения типа **Date** и содержащее день. День возвращается как число между 1 и 31, включительно.
Weekday(D)	Возвращает целое, содержащее день недели для выражения типа **Date.** День недели возвращается как число между 1 и 7, включительно; 1 — это воскресенье, 2 — понедельник и так далее.
WeekdayName(N1,B, N)	Возвращает строку с наименованием дня недели, номер которого задается параметром N1.
Hour(D)	Возвращает целое, содержащее часы как часть времени, содержащегося в выражении типа **Date.** Часы возвращаются как число между 0 и 23, включительно. Если выражение D не содержит значения времени, то **Hour** возвращает 0.
Minute(D)	Возвращает целое, содержащее минуты как часть времени в выражении типа **Date.** Минуты возвращаются как число между 0 и 59, включительно. Если выражение D не содержит значения времени, **Minute** возвращает 0.
Second(D)	Возвращает целое, содержащее секунды как часть времени в выражении типа **Date.** Секунды возвращаются как число между 0 и 59, включительно. Если выражение D не содержит значения времени, **Second** возвращает 0.
DateAdd(S, N, D)	Возвращает значение [тип Variant (Date)], содержащее дату, к которой добавлен заданный интервал времени.
DateDiff(S, D1, D2[, N1 [, N2]])	Возвращает значение [тип **Variant (Long)]** числа временных интервалов между двумя определенными датами.
DatePart(S, D,[,N1 [, N2]])	Возвращает определенную часть [тип **Variant (Integer)]** заданной даты.
DateSerial(N, N,N)	Возвращает значение последовательной даты для заданной даты. Слева направо аргументы представляют год, месяц и день. Аргумент года должен быть целым числом между 100 и 9999, месяца — между 1 и 12, дня — между 1 и 31 (все диапазоны являются включающими).
TimeSerial(N, N,N)	Возвращает значение последовательного времени. Слева направо аргументы представляют часы, минуты и секунды. Аргумент часов должен быть целым числом между 0 и 23, аргументы минут и секунд должны оба быть числами 0 и 59 (все диапазоны являются включающими).

DateValue(E)	Возвращает значение типа **Date,** эквивалентное дате, заданной аргументом E, который должен быть строкой, числом или константой, представляющей дату.
TimeValue(E)	Возвращает значение типа **Date,** содержащее время, заданное аргументом E, который может быть строкой, числом или константой, представляющей время.
Timer	Возвращает число, представляющее количество секунд от полуночи в соответствии с системным временем компьютера.

Примеры использования функций даты и времени:

```
god=year(Date())
nomer_den=WeekDay(#25/12/2006#)-1
```

Строковые функции

Строковые функции VBA часто применяются для нахождения заданных строк внутри других строк, для сравнения одной строки с другой и копирования выбранных частей строк. Строковые функции VBA используются довольно часто, потому что строковые данные очень важны и встречаются в каждом приложении VBA — Word, Excel, Access или другом host-приложении VBA. Часто необходимо манипулировать строковыми данными, полученными как пользовательский ввод функцией **InputBox.** В других случаях строковые данные появляются в коде VBA как имена файлов для документов Word, рабочих книг Excel, баз данных Access и других типов данных, сохраняемых в файлах на дисках.

В Word важность манипулирования строковыми данными очевидна: данные, сохраняемые в документах Word, в основном являются ничем иным, как строковыми данными. Строковые данные также важны в Excel в качестве имен рабочих листов, именованных диапазонов данных и так далее. В этом разделе приводятся имеющиеся в VBA строковые функции; в последующем разделе более подробно описывается, как использовать наиболее важные и полезные строковые функции.

В табл. 5, где приведены основные строковые функции VBA, N — это любое допустимое численное выражение, а S — это любое допустимое строковое выражение. Все аргументы функций являются обязательными, если не указано иначе.

Таблица 6

Функция(аргумент)	Возвращает/действие
InStr([Nl,] S1, S2[, N2])	Возвращает положение S2 в S1. N1 — начальное положение для поиска; N2 определяет тип сравнения. N1 и N2 необязательны. Если N2 опускается, то для поиска используется текущая установка **Option Compare.**

InStrRev(Sl, S2 [, Nl[, N2]])	Возвращает позицию появления строки S2 внутри S1, в направлении от конца (или N1) к началу строки. N2 определяет тип сравнения. Если N2 опускается, то для поиска используется текущая установка **Option Compare.**
LCase(S)	Возвращает строку (тип **String),** содержащую копию S со всеми символами верхнего регистра, преобразованными в символы нижнего регистра.
Left(S, N)	Возвращает строку; копирует N символов из S, начиная с левого крайнего символа S.
Len(S)	Возвращает число символов в S, включая начальные и конечные пробелы.
LTrim(S)	Возвращает копию строки S после удаления символов пробела из левой части строки (начальные пробелы).
Mid(S, N1, N2)	Возвращает строку; копирует N2 символов из S, начиная с позиции символа в S, заданной аргументом N1. N2 является необязательным; если N2 опущен, то **Mid** возвращает все символы в строке S от позиции N1 до конца строки.
Right(S, N)	Возвращает значение типа **String;** копирует N символов из S, начиная с правого крайнего символа S. Например, **Right(** outright , 5) возвращает строку right .
RTrim(S)	Возвращает копию строки S после удаления символов пробела из правой части строки (конечные символы).
Space(N)	Возвращает строку пробелов длиной N символов.
StrComp(Sl, S2, N)	Сравнивает S1 с S2 и возвращает число, обозначающее результат сравнения: -1, если S1 < S2; 0, если S1 = S2; и 1, если S1 > S2. N является необязательным и указывает, следует ли выполнять сравнение с учетом регистра. Если N опускается, строки сравниваются с использованием текущей установки **Option Compare.**
StrConv(S, N)	Возвращает строку, преобразованную в новую форму в зависимости от числового кода, заданного аргументом N. VBA предоставляет внутренние константы для использования с функцией **StrConv;** наиболее полезными являются: **vbProperCase** (преобразует строку так, что каждая буква, начинающая слово, становится заглавной), **vbLowerCase** (преобразует строку в буквы нижнего регистра) и **vbUpperCase** (преобразует строку в

	буквы верхнего регистра).
String(N, S)	Возвращает строку длиной N символов, состоящую из символа, заданного первым символом в S. Например, **String(5,** "x") возвращает строку "xxxxx".
Trim(S)	Возвращает копию строки S после удаления начальных и конечных символов пробела из этой строки.
UCase(S)	Возвращает S со всеми символами нижнего регистра, преобразованными в символы верхнего регистра.

Несколько перечисленных в табл. 5 функций преобразования типа данных относятся также к манипулированию строками: **Chr, Format, CStr,** в частности.

Примеры использования строковых функций:

n_probel=Instr(“Найти пробел”,” “)

perv_simv=Left(“Выделение текста”,1)

Пользовательские функции

Создание пользовательских функций или, как их иногда еще называют, UDF-функций (User Defined Functions) принципиально не отличается от создания макроса в обычном программном модуле. Разница только в том, что макрос выполняет последовательность действий с объектами книги (ячейками, формулами и значениями, листами, диаграммами и т.д.), а пользовательская функция - только с теми значениями, которые мы передадим ей как аргументы (исходные данные для расчета).

При написании функций-процедур для использования в качестве UDF в рабочих листах Excel необходимо знать несколько фактов, помимо общих требований для определенных пользователем функций:

- Определенные пользователем функции, которые будут использоваться в Excel, не должны иметь имена, похожие на записи ссылок на ячейку (например, A2; C1B5);
- Любые строковые данные, возвращаемые из VBA в Excel, не должны иметь более 255 символов в длину. Если UDF возвращает строку, имеющую больше 255 символов в длину, в ячейку рабочего листа, Excel укорачивает строку до максимальной длины 255 символов перед вставкой в ячейку;
- При написании UDF, возвращающей значение даты для Excel убедитесь, что задаете тип результата функции как Date. Excel

применяет формат Date для результата функции в ячейке рабочего листа только, если результат имеет VBA-тип Date.

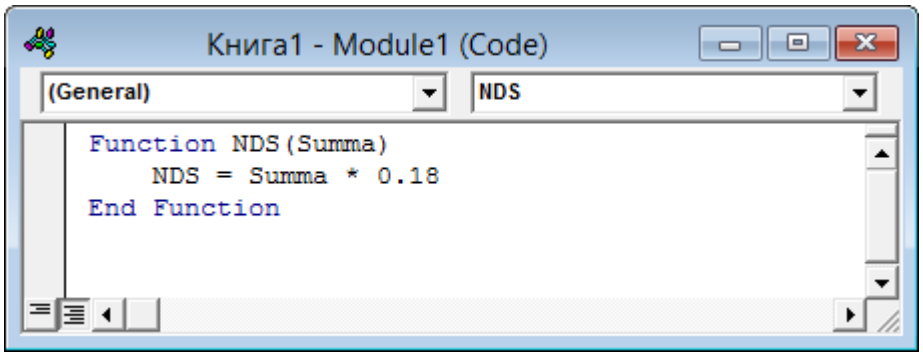
Структура пользовательской функции

```
Function имя_функции(арг1 As тип, арг2 As тип …) As тип
    Тело функции
    имя_функции = выражение
End Function
```

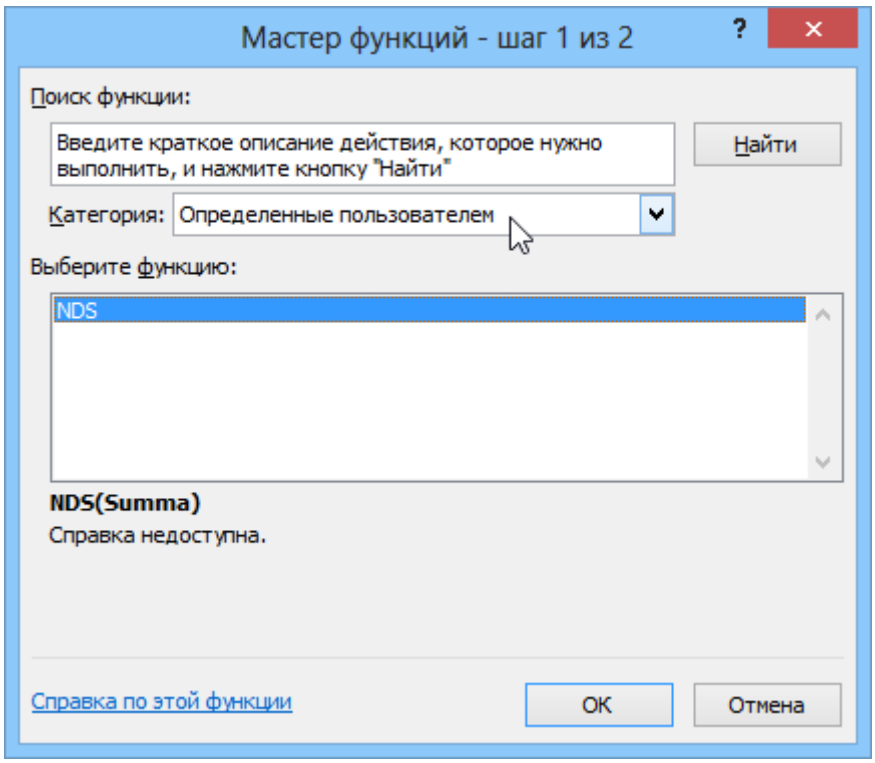
Досрочный выход из функции выполняется с помощью оператора Exit Function

Порядок создания пользовательской функции

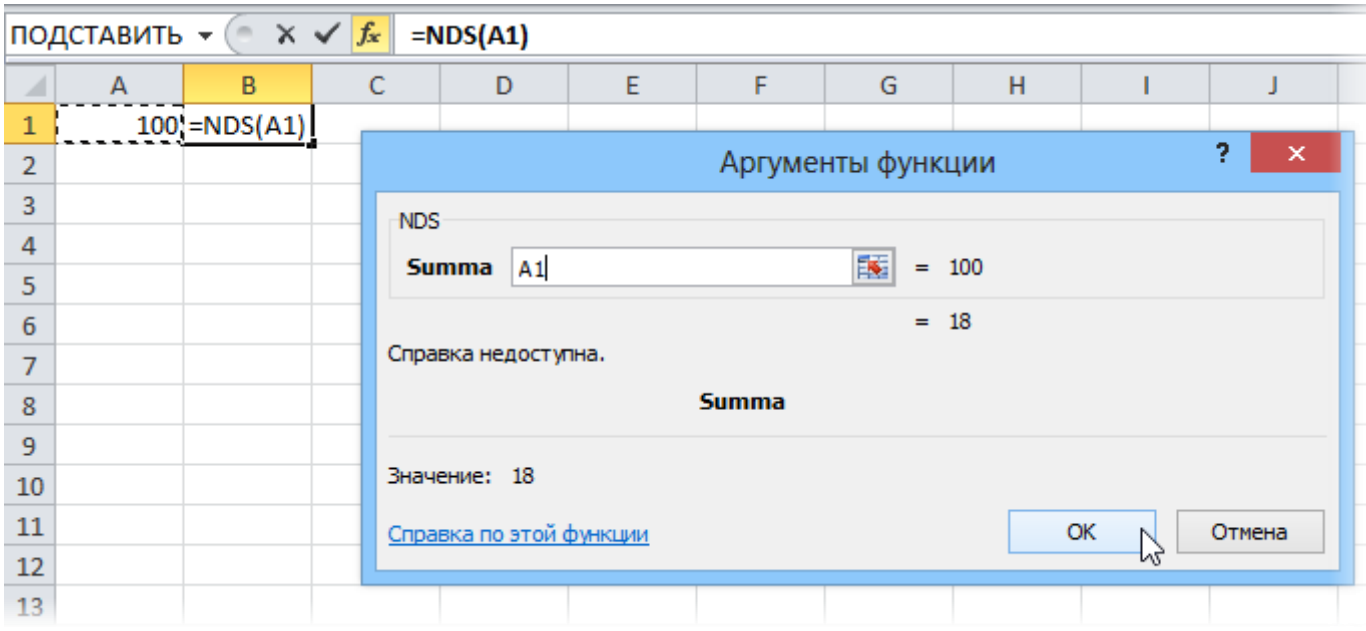
Чтобы создать пользовательскую функцию для расчета, например, налога на добавленную стоимость (НДС) откроем редактор VBA, добавим новый модуль через меню ***Insert - Module*** и введем туда текст нашей функции:



Обратите внимание, что в отличие от макросов функции имеют заголовок **Function** вместо **Sub** и непустой список аргументов (в нашем случае это *Summa*). Если тив переменных и функции не задан, то тип интерпретируется как **Variant.** После ввода кода наша функция становится доступна в обычном окне Мастера функций (***Вставка - Функция***) в категории **Определенные пользователем (User Defined)**:

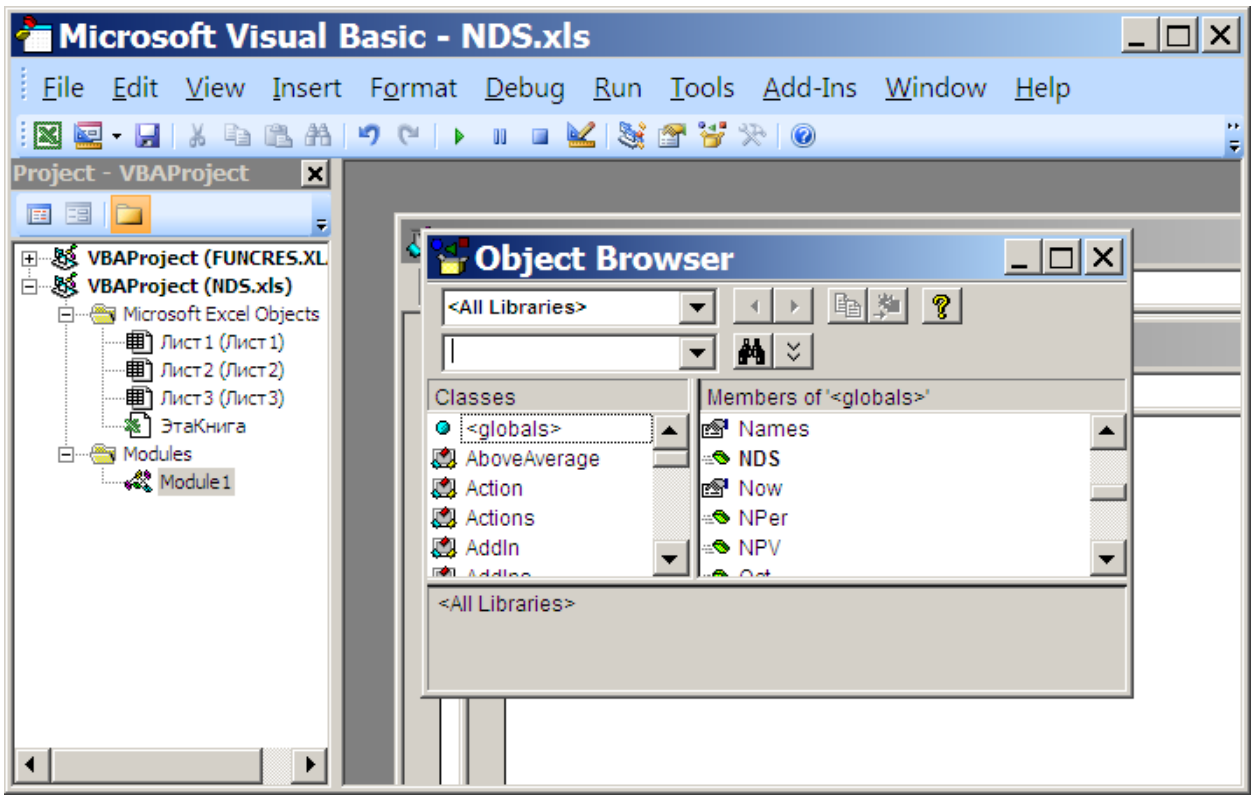


После выбора функции выделяем ячейки с аргументами (с суммой, для которой надо посчитать НДС) как в случае с обычной функцией:

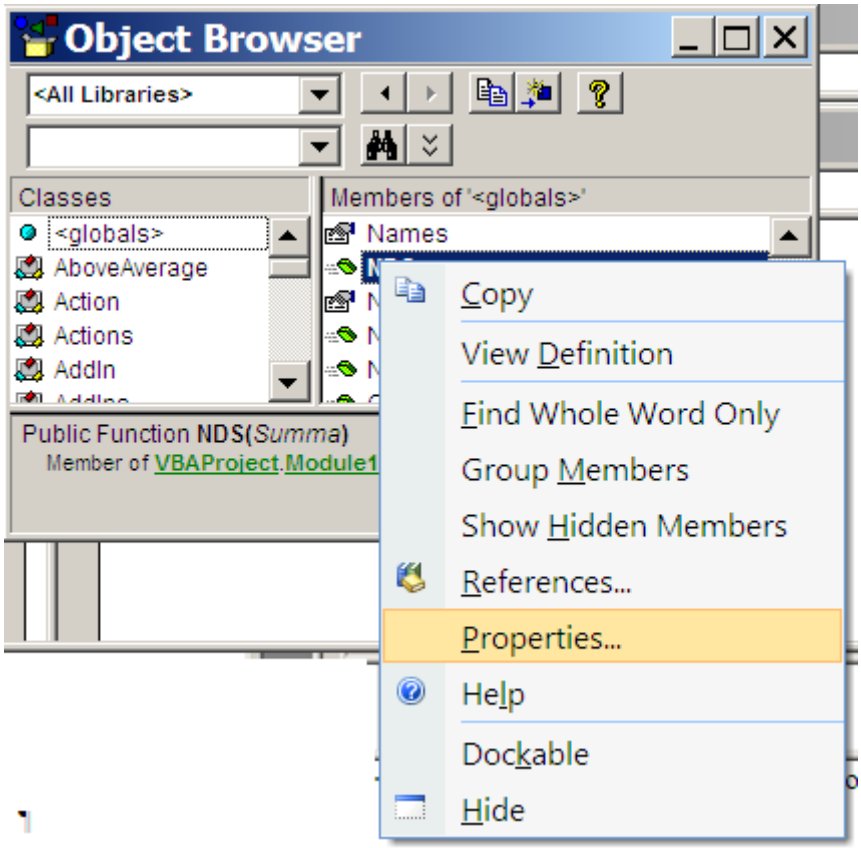


Обратите внимание, что в диалоговом окне выводится : «Справка недоступна». Для того, чтобы отображать справку для пользовательской функции надо:

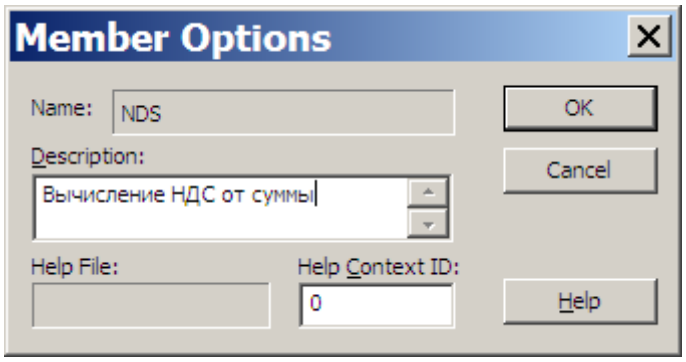
1. В редакторе VBA открыть окно Object Browser ***View → Object Browser***.
2. В списке выбрать пользовательскую функцию. Эта функция располагается по алфавиту.



3. Щелкнув правой кнопкой по названию функции, выбрать Properties …



4. В диалоговом окне Member Options ввести пояснения в текстовое окно Description.

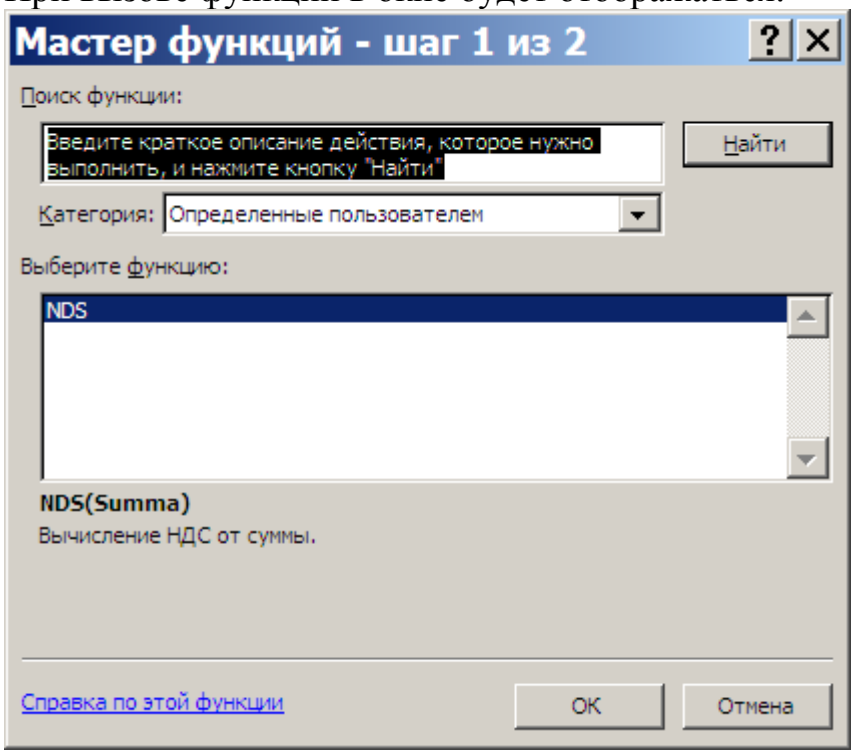


ВАЖНО

Чтобы пояснения отображались в окне Мастер функций, надо Перед пользовательской функции указать доступ Public:

```
Public Function NDS(Summa)
   NDS = Summa * 0.18
End Function
```

При вызове функции в окне будет отображаться:



Рассмотрим примеры пользовательских функций.

Задание 1

Создать новую книгу Функции. Набрать и отладить функции Vozr и ИНИЦИАЛЫ/

Функция, вычисляющая возраст по дате рождения.

```
Function Vozr(d As Date) As Integer
Dim k As Integer
k = Date - d
Vozr = Int(k / 365.25)
End Function
```

Функция получения фамилии и инициалов

```
Function ИНИЦИАЛЫ(ФИО As String) As String
Dim fam As String, im As String, ot As String
Dim n1_pr As Integer, n2_pr As Integer
ФИО = StrConv(ФИО, vbProperCase) 'сделать первые буквы в словах
                                 'прописными
n1_pr = InStr(ФИО, " ") 'найти первый пробел
fam = Left(ФИО, n1_pr - 1) 'выделить фамилию
n2_pr = InStr(n1_pr + 1, ФИО, " ") 'найти второй пробел
im = Mid(ФИО, n1_pr + 1, 1) 'выделить первую букву имени
ot = Mid(ФИО, n2_pr + 1, 1) 'выделить первую букву отчества
ИНИЦИАЛЫ = fam + " " + im + "." + ot + "."
End Function
```

Обеспечить вывод справки по данным функциям.
Продемонстрировать работу функций преподавателям.

Задания для самостоятельной работы

Вариант 1

1. Даны длина катета, найти длину гипотенузы
2. Найти количество дней между датами.
3. Написать функцию вставляющую подстрока в строку с заданной позицией.

Вариант 2

1. Дана гипотенуза и катет, найти второй катет
2. Найти количество месяцев между датами.
3. Дана строка, содержащая название книги, а затем фамилия и инициалы автора. Получить фамилию автора.

Вариант 3

1. Дан радиус окружности найти ее площадь
2. Найти количество недель между датами.
3. Дана строка, содержащая фамилию и группу. Получить строку фамилия и курс.

Вариант 4

1. Дан радиус найти длину окружности
2. Найти количество кварталов между датами.
3. Написать функцию, удаляющую из строки заданную подстроку.

Вариант 5

1. Даны стороны прямоугольника найти площадь прямоугольника

2. Сколько дней прошло с начала года
3. Дана строка с информацией о фирме содержащей расчетный счет. Расчетный счет начинается со знака №. Выделить расчетный счет фирмы.

Вариант 6

1. Даны стороны прямоугольника. Найти периметр прямоугольника
2. Сколько дней осталось до конца года
3. Дано вещественное число получить из него цену в виде целая часть руб дробная часть из двух цифр коп.

Вариант 7

1. Даны длины сторон равнобедренной трапеции. Найти высоту трапеции.
2. Определить день недели первого дня заданного года.
3. Дана строка с ценой число руб. число коп. Преобразовать ее в вещественное число.

Вариант 8

1. Даны координаты начала и конца прямой линии.
2. Определить день недели сотого дня заданного года.
3. Дана дата в виде строки в американском формате месяц/день/год. Переделать ее в русский формат день.месяц.год.

Вариант 9

1. Найти координаты середины линии.
2. Определить день недели 8 марта в заданном году.
3. Из строки Фамилия Имя Отчество дата рождения в виде день.месяц.год получить:
   Фамилия Имя Отчество возраст.

Вариант 10

1. Даны стороны равнобедренного треугольника. Найти его высоту.
2. По дню рождения определить 10000 день жизни.
3. Из города улицы дома и квартиры получить адрес в виде:
   г. Город, ул. Улица, д.дом, кв. квартира

Вариант 11

1. Даны стороны параллелограмма. Найти площадь параллелограмма.
2. Определить день недели для 25-летнего юбилея.
3. Функцию выделяющую из строки подстроку по начальному и конечному символу.

Вариант 12

1. Даны длины сторон равнобедренной трапеции. Найти ее площадь.
2. По введенной дате определить век.
3. Даны отдельно фамилия, имя отчество получить фамилию и инициалы.

Контрольные вопросы

1. Где пишется код пользовательской функции.
2. Что такое аргументы функции.
3. Для чего надо указывать Public в заголовке функции.
4. Как задать справку для пользовательской функции.
5. Для чего указывается тип для функции.
6. Чем отличаются стандартные функции от пользовательских функций.

Лабораторная работа 3
Ветвления

Как и во всех других языках программирования, в VBA имеются различные *управляющие конструкции,* позволяющие изменять порядок выполнения программы При отсутствии управляющих конструкций операторы программы выполняются последовательно, начиная с первого и кончая последним Хотя в некоторых самых простых случаях этого и бывает достаточно, обычно все-таки требуется изменять порядок выполнения операторов при выполнении определенных условий, либо пропуская выполнение некоторых операторов, либо, наоборот, многократно повторяя их. Оказывается, для реализации любых алгоритмов достаточно иметь только два вида управляющих конструкций: ветвления и циклы

Управляющие конструкции ветвления позволяют проверить некоторое условие, а затем, в зависимости от результатов этой проверки, выполнить ту или иную группу операторов Для организации ветвлений в VBA используются различные формы оператора ветвления if и оператор выбора Select `Case`

Краткая форма оператора ветвления if может иметь как однострочную, так и блочную форму. Простейшая, краткая форма оператора if используется для проверки одного условия, а затем, в зависимости от результата проверки, либо выполнения, либо пропуска одного оператора или блока из нескольких операторов

```
If <условие> Then
<блокОператоров1>
[Else
<блокОператоров2>]
End If
```

В качестве условия можно использовать логическое выражение, возвращающее значение True (ИСТИНА) или False (ЛОЖЬ). Если условие истинно, выполняется первый блок операторов, заключенный между ключевыми словами Then и Else, а в противном случае — второй блок, заключенный между ключевыми словами Else и End If.

Условия, которые записываются в условном операторе после слова If, бывают простые и сложные. Простые условия имеют следующую структуру:

Выражение Операция отношения Выражение

Операции отношения

=	равно
<>	неравно
>	Больше
<	Меньше
>=	Больше или равно
<=	Меньше или равно

Например: a+b>6, Sin(x)<=0, f*h<>k+d

Оператор if может иметь краткую и полную форму. В краткой форме, если условие возвращает значение False, оператор или блок операторов, заключенных между ключевыми словами Then и End if, составляющих тело краткого оператора ветвления, не будет выполняться.

Пример краткой формы оператора if:

```
If a<0 Then
      a=-a
End If
```

Полная форма оператора if используется в тех случаях, когда имеются два различных блока операторов, и по результатам проверки условия нужно выполнить один из них.

Пример полной формы оператора if:

```
If x=0 Then
    x=1
Else
    z=y/x
End If
```

Для того чтобы текст вашей процедуры был понятным и удобным для восприятия, рекомендуется делать отступы для групп операторов так, как это указано при описании их синтаксиса. В VBA предусмотрено удобное средство изменения отступов — нажатие на клавишу <Tab> увеличивает отступ вправо, нажатие комбинации клавиш <Shift>+<Tab> уменьшает этот отступ.

Использование оператора if делает функции, позволяющие избежать некоторых ошибок. Например, использование функции ИНИЦИАЛЫ предполагает, что исходная строка обязательно содержит фамилию, имя и отчество. А если отчество отсутствует, то возникнет ошибка. Поэтому в коде программы надо проверить значение переменной n2_pr – положение второго пробела. Если эта переменная равна нулю, то отчество в строке отсутствует. Функция будет иметь вид:

```
Dim fam As String, im As String, ot As String
Dim n1_pr As Integer, n2_pr As Integer
ФИО = StrConv(ФИО, vbProperCase)
n1_pr = InStr(ФИО, " ")
fam = Left(ФИО, n1_pr - 1)
im = Mid(ФИО, n1_pr + 1, 1) + "."
n2_pr = InStr(n1_pr + 1, ФИО, " ")
If n2_pr > 0 Then
   ot = Mid(ФИО, n2_pr + 1, 1) + "."
Else
   ot = ""
End If
ИНИЦИАЛЫ = fam + " " + im + ot
End Function
```

Задание 1

Набрать и отладить код функции ИНИЦИАЛЫ.

Еще один пример необходимости использования оператора If.

Задание 2

По году определить день недели для 1 января данного года.

Существует стандартная функция для определения номера для недели. Но эта функция дана для американского стандарта, в котором первым днем недели является воскресенье. Для русого стандарта, если день недели не воскресенье, то из него достаточно вычесть единицу. Но тогда воскресенье будет нулевым днем. Здесь требуется использовать оператор If. Функция будет иметь вид:

```
Function Начало_Года(год As Integer) As Integer
Dim d As Date
d = DateSerial(год, 1, 1)
If Weekday(d) = 1 Then
Начало_Года = 7
Else
Начало_Года = Weekday(d) - 1
End If
End Function
```

Иногда приходится делать выбор одного из целой группы альтернативных действий на основе проверки нескольких различных условий. Для этого можно использовать цепочку операторов ветвления If...Then...ElseIf.

```
If <условие1> Then
<блокОператоров1>
Elself <условие2> Then
<блокОператоров2>
Elself <условие3> Then
<блокОператоров3>
. . .
Elself <условиеN> Then
<блокОператоровN>
Else
<блокОператоров_Else>
End If
```

Пример использования блочного оператора if:

```
If n=1 Then
   текст=”год”
ElseIf n<5 Then
 текст =”года”
Else
 текст =”лет”
End If
```

В данном примере для числа n, обозначающем срок обучения от 1 до 6, определяется текст комментария.

Задание 3

Написать функцию определяющую по заданному весу (в килограммах) и росту (в метрах) ИНДЕКС МАССЫ ТЕЛА (Индекс Кетле) по формуле: вест/(рост*рост) и вывести комментарии согласно таблице

ИМТ	Комментарии
<18,5	Излишняя худоба
>=18,5 и <=22,9	Нормальный вес
>=23 и <=27,4	Избыточный вес
>27,5	Ожирение

Данная функция будет иметь вид:

```
Function Индекс_Кетле(Рост_в_см As Integer, Вес_в_кг As Integer) As String
Dim imt As Single
imt = Вес_в_кг / (Рост_в_см / 100) ^ 2
If imt < 18.5 Then
  Индекс_Кетле = "Излишняя худоба"
ElseIf imt <= 22.9 Then
   Индекс_Кетле = "Нормальный вес"
ElseIf imt <= 27.4 Then
   Индекс_Кетле = "Избыточный вес"
Else
   Индекс_Кетле = "Ожирение"
End If
End Function
```

Набрать и отладить данную функцию.

Задания для самостоятельной работы

Вариант 1

1. Написать функцию вставляющую подстроку в строку с заданной позицией. Проверить, не выходит ли заданная позиция за пределы строки.
2. В зависимости от возраста и пола выдать приветствие: Здравствуй девочка, мальчик , девушка, юноша, женщина, мужчина.

Вариант 2

1. Дана гипотенуза и катет, найти второй катет. Проверить, может ли существовать этот треугольник.
2. В зависимости от возраста ребенка выдать сообщение: младенец, ясельник, детсадовец, школьник.

Вариант 3

1. Найти количество недель между датами. Проверить, чтобы первая дата была меньше второй, иначе переставить их местами.
2. В зависимости от введенной суммы и валюты: доллар, евро, фунт, иена, перевести сумму в рубли.

Вариант 4

1. Написать функцию, удаляющую из строки заданную подстроку. Проверить существует в стоке данная подстрока.
2. В зависимости от номера месяца вывести: зима, лето, осень, весна.

Вариант 5

1. Дана строка с информацией о фирме содержащей расчетный счет. Расчетный счет начинается со знака №. Выделить расчетный счет фирмы. Проверить присутствует ли в адресе расчетный счет.
2. В зависимости от часа вывести время суток: утро, день, вечер, ночь.

Вариант 6

1. Дано вещественное число получить из него цену в виде целая часть руб. дробная часть из двух цифр коп. Проверить положительное ли данное число.
2. В зависимости от возраста ребёнка вывести лет, года, год.

Вариант 7

1. Дана строка с ценой число руб. число коп. Преобразовать ее в вещественное число. Проверить, есть ли в данной строке руб. или коп.
2. Задумать число от 1 до 10 и запросить ответ. В зависимости от введенного числа вывести больше, меньше или равно.

Вариант 8

1. Дана дата в виде строки в американском формате месяц/день/год. Переделать ее в русский формат день.месяц.год. Проверить правильно ли заданы месяц и день.
2. Ввести год, месяц, день. В зависимости от даты вывести: прошлое, настоящее будущее.

Вариант 9

1. Из строки Фамилия Имя Отчество дата рождения в виде день.месяц.год получить:
   Фамилия Имя Отчество возраст. Проверить правильно ли заданы месяц и день.
2. Ввести рост человека и пол. В зависимости от роста и пола вывести низкий, средний и высокий.

Вариант 10

1. Даны стороны равнобедренного треугольника. Найти его высоту. Проверить, существует ли такой треугольник.
2. Ввести летнюю температуру. И вывести комментарии: тепло прохладно, холодно и мороз.

Вариант 11

1. Функцию выделяющую из строки подстроку по начальному и конечному символу. Проверить, не выходит ли конечный символ за пределы строки.
2. Ввести температуру в помещении. Вывести комментарий: жарко тепло прохладно и холодно.

Вариант 12

1. По введенной дате определить век. Проверить, чтобы год относился или к 20 или к 21 веку.
2. Ввести пол и возраст взрослого человека и вывести работающий человек или пенсионер.

Ветвления. Сложные условия

Сложные условия состоят из нескольких простых условий, соединенных логическими операциями. Существуют следующие логические операции:

not	Операция отрицания
And	Логическое «и»
Or	Логическое «или»

Если два условия соединены логическим «и» (and), то условие выполняется если оба условия выполняются одновременно.

Пример: a>6 and a<20 – условие выполняется если a находится в интервале от 6 до 20 (например a = 10).

Если условия соединены логическим «или» (or), то условие выполняется, если выполняется хотя бы одно условие.

Пример: x<0 or b>10 – если x отрицательный, то независимо от того, чему равен b, условие выполняется.

Логические операции, как и арифметические имеют приоритет. Высший приоритет у операции not, следующий приоритет имеет операция and и самый низкий приоритет у операции or.

Использование сложных условий повышает наглядность операции и сокращает количество операторов If.

Задание 4

Создать функцию СЕЗОН, которая по дате выводит название сезона.

```
Function СЕЗОН(Дата As Date) As String
Dim mes As Integer
mes = Month(Дата)
If mes > 2 And mes < 6 Then СЕЗОН = "весна"
If mes > 5 And mes < 9 Then СЕЗОН = "лето"
If mes > 8 And mes < 12 Then СЕЗОН = "осень"
If mes = 1 Or mes = 2 Or m = 12 Then СЕЗОН = "зима"
End Function
```

Набрать и отладить код функции.

Задание 5

Создать функцию A:
Если X или У отрицательные вычислить A по формуле
A=X * Y,
Иначе

$$A=\begin{cases}1, \text{ если } X>Y\\ 0, \text{ если } X=Y\\ -1, \text{ если } X<Y\end{cases}$$

`

Представим код данной функции:

```
Function A(X As Single, Y As Single) As Single
If X < 0 Or Y < 0 Then
   A = X * Y
Else
   If X > Y Then
      A = 1
   ElseIf X = Y Then
      A = 0
   Else
      A = -1
   End If
End If
End Function
```

Если программа вывела результат, это еще не значит, что она работает правильно. Для проверки необходимо протестировать программа, задавая значения X и Y, чтобы они попали на каждую ветку нашей развилки и вычислить значения A для этих Xи Y.

X	Y	A
-2	4	-6
3	4	-1
5	5	0
6	3	1

Задания для самостоятельной работы

Вариант 1

Даны a и b. Если a и b попадают в область $3 < a < 10$ и $b < 0$, то вычислить значение y по формуле
$y = a + b^2 - 4$,
иначе

$$y=\begin{cases}5, \text{ если } x \geq 2\\ y^2 - 1, \text{ если } x = 3\\ y + 2 , \text{ если } x < 2\end{cases}$$

для произвольного x.

Вариант 2

Даны a и b. Если a и b попадают в область $a < 0$ и $b < 5$, то вычислить значение y по формуле

$y = a^2 + b + 10$,

иначе

$$y = \begin{cases} x + 2, \text{ если } 2x < -1 \\ x, \text{ если } -1 \le 2x \le 1 \\ x^2 + 1, \text{ если } 2x > 1 \end{cases}$$

для произвольного x.

Вариант 3

Даны a и b. Если a и b попадают в область $a < 5$ и $b > 0$, то вычислить значение y по формуле

$y = a + 5b - 10$,

иначе

$$y = \begin{cases} z + x - 1, \text{ если } z \le 5 \\ z^2 + 1, \text{ если } z > 5 \text{ и } x \ge 1 \\ x - z, \text{ если } z > 5 \text{ и } x < 1 \end{cases}$$

где $z = 4x + 3$ для произвольного x.

Вариант 4

Даны a и b. Если a и b попадают в область $a > 5$ и $b < -3$, то вычислить значение y по формуле

$y = a + 2b^2 + 3$,

иначе

$$y = \begin{cases} x, \text{ если } x < 0 \\ 5, \text{ если } x = 0 \\ x + 1, \text{ если } x > 0 \end{cases}$$

для произвольного x.

Вариант 5

Даны a и b. Если a и b попадают в область $a < 0$ и $b > 0$, то вычислить значение y по формуле

$y = a^2 + 2b$,

иначе

$$y = \begin{cases} 2x^2 - 5x - 6, \text{ если } x > 5 \\ x/10 - 3, \text{ если } x = 5 \\ 2x - x^2 + 10, \text{ если } x < 5 \end{cases}$$

для произвольного x.

Вариант 6

Даны a и b. Если a и b попадают в область $a > 0$ и $b < 0$, то вычислить значение y по формуле

$y = a - b$,
иначе

$$y = \begin{cases} x, \text{ если } 2x < -1 \\ x^2, \text{ если } -1 \le 2x \le 0 \\ x + 1, \text{ если } 2x > 0 \end{cases}$$

для произвольного x.

Вариант 7

Даны a и b. Если a и b попадают в область $a > 3$ и $b < 10$, то вычислить значение y по формуле

$y = 2a^2 + 3b - 1$,
иначе

$$y = \begin{cases} z + 4, \text{ если } z^2 - 2z - 3 < 0 \\ 0, \text{ если } z^2 - 2z - 3 = 0 \\ z^2 - 1, \text{ если } z^2 - 2z - 3 > 0 \end{cases}$$

где $z = 2x + 1$ для произвольного x.

Вариант 8

Даны a и b. Если a и b попадают в область $a > -5$ и $b < 0$, то вычислить значение y по формуле

$y = a + b$,
иначе

$$y = \begin{cases} xt + 1, \text{ если } x > 0 \text{ и } t \le 5 \\ x + t, \text{ если } x > 0 \text{ и } t > 5 \\ 3, \text{ если } x \le 0 \end{cases}$$

где $x = 2t + 5$ для произвольного t.

Вариант 9

Даны a и b. Если a и b попадают в область $a < 0$ и $b > 1$, то вычислить значение y по формуле

$y = a^2 - 2b$,
иначе

$$y = \begin{cases} 2z + x - 4, x \le 5 \\ z^2 + 1, x > 5 \text{ и } x \ne 6 \\ x - 6, x > 5 \text{ и } x = 6 \end{cases}$$

если $z = 4x - 5$ для произвольного x.

Вариант 10

Даны a и b. Если a и b попадают в область $a > 5$ и $b < 0$, то вычислить значение y по формуле

$y = a - 4b^2$,
иначе

$$y = \begin{cases} x + 3, \text{ если } x < 1 \\ 4, \text{ если } x = 1 \end{cases}$$

$x + 1$, если $x > 1$

для произвольного x.

Вариант 11

Даны a и b. Если a и b попадают в область $a < 0$ и $b < 0$, то вычислить значение y по формуле

$y = (a + b)^2 - 2$,

иначе

$$y = \begin{cases} c^2, \text{ если } c + 1 > 1 \\ c + 1, \text{ если } -1 \le c + 1 \le 1 \\ x + 1 , \text{ если } c + 1 > -1 \end{cases}$$

где $c = 2x - 6$ для произвольного x.

Вариант 12

Даны a и b. Если a и b попадают в область $a > -5$ и $b < 5$, то вычислить значение y по формуле

$y = a^2 - b^2$,

иначе

$$y = \begin{cases} 3, \text{ если } f \le 0 \\ y^2 - 2, \text{ если } f \le 0 \text{ и } x = 6 \\ y + 5, \text{ если } f > 0 \text{ и } x \ne 2 \end{cases}$$

если $f = x^2 - 40x + 3$ для произвольного x.

Контрольные вопросы

1. Когда используется краткий вид функции if/
2. Что происходит, когда при использовании краткого вида функции if условие не выполняется.
3. В каких случаях используется блочный вид функции if.
4. Для чего нужно тестировать функцию.
5. В каких ситуациях используется логическая операция And.
6. Если оператор if имеет вид:

```
If d< -1 then y=7
  ElseIf d<5 then y=10
  ElseIf d<10 then y=20
  Else y=30
```

Чему равен y при d=25,

при d=9.

Лабораторная работа № 5
Оператор выбора

Если выбор одной из нескольких возможностей все время основан на различных значениях одного и того же выражения, гораздо удобнее использовать специально предназначенный для этого оператор выбора select case, имеющий следующий синтаксис:

```
Select Case <проверяемоеВыражение>
Case <списокЗначений1>
<блокОператоров1>
Case <списокЗначений2>
<блокОператоров2>
Case <списокЗначений3>
<блокОператоров3>
Case Else
<блокОператоров_Else>
End Select
```

Проверяемое выражение вычисляется в начале работы оператора select Case. Это выражение может возвращать значение любого типа, например, логическое, числовое или строковое.

Список выражений представляет собой одно или несколько выражений, разделенных запятой. При выполнении оператора проверяется, соответствует ли хотя бы один из элементов этого списка проверяемому выражению. Эти элементы списка выражений могут иметь одну из следующих форм:

- <выражение>

  в этом случае проверяется, совпадает ли значение проверяемого выражения с этим выражением;
- <выражение1> To <выражение2>

  в этом случае проверяется, находится ли значение проверяемого выражения в указанном диапазоне значений;
- Is <логическаяОперация> <выражение>

  в этом случае проверяемое выражение сравнивается с указанным значением с помощью заданной логической операции, например, условие Is >= 10 считается выполненным, если проверяемое значение не меньше 10.

Если хотя бы один из элементов списка соответствует проверяемому выражению, то выполняется соответствующая группа операторов, и на этом выполнение оператора select case заканчивается, а остальные списки выражений не проверяются, т. е. отыскивается только первый подходящий элемент списков выражений. Если же ни один из элементов всех этих списков не соответствует значению проверяемого выражения, выполняются операторы группы Else, если таковая присутствует.

Задание 1

В некоторых ситуациях оператор выбора может заменять оператор if. Как будет выглядеть уже рассмотренный пример с определением сезонов с использованием оператора выбора.

```
Function СЕЗОН(Месяц As Integer) As String
Select Case Месяц
Case 1, 2, 12
СЕЗОН = "Зима"
Case 3 To 5
СЕЗОН = "Весна"
Case 6 To 8
СЕЗОН = "Лето"
Case 9 To 11
СЕЗОН = "Осень"
Case Else
СЕЗОН = "Не правильный месяц"
End Select
End Function
```

Задание 2

Ввести два числа и операцию, выполняемую с этими числами. Вывести результат операции.

```
Function РЕЗУЛЬТАТ(Число1, Число2, Операция As String)
Select Case Операция
Case "+"
РЕЗУЛЬТАТ = Число1 + Число2
Case "-"
РЕЗУЛЬТАТ = Число1 - Число2
Case "*"
РЕЗУЛЬТАТ = Число1 * Число2
Case "/"
If Число2 <> 0 Then
РЕЗУЛЬТАТ = Число1 / Число2
Else
РЕЗУЛЬТАТ = "Деление на ноль"
End If
Case Else
РЕЗУЛЬТАТ = "Нет такой операции"
End Select
```

Задание 3

Ввести день и месяц и вывести комментарий: праздничный день, выходной день, рабочий день.

```
Function ПРАЗДНИКИ(Дата As Date) As String
m = Month(Дата)
d = Day(Дата)
w = Weekday(Дата)
If w = 1 Or w = 7 Then
ПРАЗДНИКИ = "Выходной"
Else
ПРАЗДНИКИ = "Рабочий день"
End If
```

```
Select Case m
 Case 1
If d >= 1 And d <= 8 Then
ПРАЗДНИКИ = "Новогодние праздники"
End If
Case 2
If d = 23 Then
ПРАЗДНИКИ = "День защитника отечества"
End If
Case 3
If d = 8 Then
ПРАЗДНИКИ = "Международный  женский день"
End If
Case 5
If d = 1 Or d = 2 Then
ПРАЗДНИКИ = "Первомайские праздники"
End If
If d = 9 Then
ПРАЗДНИКИ = "День Победы"
End If
Case 6
If d = 12 Then
ПРАЗДНИКИ = "День Независимости"
End If
Case 11
If d = 4 Then
ПРАЗДНИКИ = "День Примирения и согласия"
End If
End Select
End Function
```

Задания для самостоятельной работы

Используя оператор выбора создать функции

Вариант 1

1. Ввести зимнюю температуру. И вывести комментарии: тепло, прохладно, холодно и мороз.
2. Составьте программу определения по дате рождения знака по календарю друидов

Яблоня	22.10 – 1.1	25.6 - 4.7
Пихта	2.1 – 11.1	5.7 – 14.7
Вяз	12.1 – 24.1	15.7 - 25.7
Кипарис	25.1 – 3.2	26.7 – 4.8
Тополь	4.2 – 8.2	5.8 – 13.8
Картас	9.2 – 18.2	14.8 – 23.8
Сосна	19.2 – 28/29.2	24.8 – 2.9
Ива	1.3 – 10.3	3.9 – 12.9
Липа	11.3 – 20.3	13.9 – 23.9
Орешник	22.3 – 31.3	24.9 – 3.10
Рябина	1.4 – 10.4	4.10 – 13.10
Клен	11.4 – 20.4	14.10 – 23.10

Орех	21.4 – 30.4	24.10 – 2.11
Жасмин	1.5 – 14.5	3.11 – 11.11
Каштан	15.5 – 24.5	12.11 – 21.11
Ясень	25.5 – 3.6	22.11 – 1.12
Граб	4.6 – 13.6	2.12 – 11.12
Инжир	14.6 - 23.6	12.12 – 21.12
Дуб	21.3	
Береза	24.6	
Маслина	23.9	
Бук	21.12 – 22.12	

Вариант 2

1. В зависимости от возраста и пола выдать приветствие: Здравствуй девочка, мальчик , девушка, юноша, женщина, мужчина.
2. Составить программу, в которой определить по введенной дате, к какому знаку цветка он относится.

Знак	Дата	Знак	Дата
ГОРЕЧАВКА ЖЕЛТАЯ	(1 января-10 января	КУВШИНКА	2 июля-12 июля
ЧЕРТОПОЛОХ	11 января-20 января	ФИАЛКА	13 июля-23 июля
БЕССМЕРТНИК	21 января-31 января	ШИПОВНИК	24 июля-2 августа
ОМЕЛА	1 февраля-10 февраля	ПОДСОЛНУХ	3 августа-12 августа
КРАСАВКА	11 февраля-19 февраля	РОЗА	13 августа-23 августа
МИМОЗА	20 февраля-29 февраля	ДЕЛЬФИНИУМ	24 августа-2 сентября
МАК	1 марта-10 марта	ГВОЗДИКА	3 сентября-11 сентября
ЛИЛИЯ	11 марта-20 марта	АСТРА	12 сентября-22 сентября
НАПЕРСТЯНКА	21 марта-31 марта	ВЕРЕСК	23 сентября-3 октября
МАГНОЛИЯ	1 апреля-10 апреля	КАМЕЛИЯ	4 октября-13 октября
ГОРТЕНЗИЯ	11 апреля-20 апреля	СИРЕНЬ	14 октября – 23 октября
ГЕОРГИН	21 апреля-30 апреля	ФРЕЗИЯ	24 октября-2 ноября
ЛАНДЫШ	1 мая-10 мая	ОРХИДЕЯ	3 ноября – 12 ноября
ПОРТУЛАК	11 мая-21 мая	ПИОН	13 ноября-22 ноября
РОМАШКА	22 мая-31 мая	ГЛАДИОЛУС	23 ноября-2 декабря
КОЛОКОЛЬЧИК	1 июня-11 июня	ОДУВАНЧИК	3 декабря-12 декабря
МАРГАРИТКА	12 июня-21 июня	ЛОТОС	13 декабря – 22 декабря
ТЮЛЬПАН	22 июня-1 июля	ЭДЕЛЬВЕЙС	23 декабря-31 декабря

Вариант 3

1. В зависимости от возраста ребенка выдать сообщение: младенец, ясельник, детсадовец, школьник.
2. Составьте программу определения по дате рождения знака Зодиака

Знак	Дата
Овен	21 марта — 19 апреля
Телец	20 апреля — 20 мая
Близнецы	21 мая — 20 июня
Рак	21 июня — 22 июля
Лев	23 июля — 22 августа
Дева	23 августа — 22 сентября
Весы	23 сентября — 22 октября
Скорпион	23 октября — 21 ноября
Стрелец	22 ноября — 21 декабря
Козерог	22 декабря — 19 января
Водолей	20 января — 18 февраля
Рыбы	19 февраля — 20 марта

Вариант 4

1. В зависимости от введенной суммы и валюты: доллар, евро, фунт, иена, перевести сумму в рубли.
2. Составить программу, в которой определить по введенной дате, к какому знаку гороскопа Мифов он относится.

Знак	Даты
Кентавр	8 января - 12 февраля
Гарпия	13 февраля - 18 марта
Пегас	19 марта - 24 апреля
Цербер	25 апреля - 30 мая
Сатир	31 мая - 3 июля
Сирена	4 июля - 10 августа
Грифон	11 августа - 15 сентября
Химера	16 сентября - 22 октября
Сфинкс	23 октября - 30 ноября
Минотавр	1 декабря - 7 января

Вариант 5

1. В зависимости от суммы вывести единицы измерения копейка, копеек, копейки.
2. Составить программу, в которой определить по введенной дате, к какому знаку гороскопа Викингов он относится.

Знак	Даты
Месяц бога Одина	21.03 - 20.04
Месяц бога Тора	21.04 - 20.05
Месяц бога Тира	21.05 - 21.06

Месяц бога Бальдра	22.6 - 22.7
Месяц бога Браги	23.7 - 23.8
Месяц бога Видара	24.8 - 22.9
Месяц бога Ходера	23.09 - 23.10
Месяц бога Хермеда	24.10 - 22.11
Месяц бога Хенера	23.11 - 21.12
Месяц бога Ньерда	22.12 - 20.01
Месяц бога Локи	21.01 - 19.02
Месяц бога Вали	20.02 - 20.03

Вариант 6

3. В зависимости от часа вывести время суток: утро, день, вечер, ночь.
4. Составить программу, в которой определить по введенной дате, к какому знаку гороскопа Славян он относится.

Знак	Дата	Знак	Дата
Мороз	24 декабря – 30 января	Червень	7 – 31 июля
Сечень	31 января – 28 февраля	Зарев	1 – 28 августа
Сухий	1 – 31 марта	Сева	29 августа – 13 сентября
Березень	1 – 30 апреля	Мокошь	14 – 27 сентября
Тавень	1 – 14 мая	Сварожич	28 сентября – 15 октября
Леля	15 мая – 2 июня	Морена	16 октября – 8 ноября
Изок	3 июня – 6 июля, кроме 24 июня	Зима	9 – 28 ноября
День Ивана Купалы	24 июня	Студень	29 ноября – 23 декабря

Вариант 7

1. В зависимости от возраста ребёнка вывести лет, года, год.
2. Составить программу, в которой определить по введенной дате, к какому знаку греческого гороскопа он относится.

Знак	Дата
Месяц богини Афины	21.03 - 20.04
Месяц богини Афродиты	21.04 - 20.05
Месяц бога Аполлона	21.05 - 21.06
Месяц бога Гермеса	22.6 - 22.7
Месяц верховного бога Зевса	23.7 - 23.8
Месяц богини Деметры	24.8 - 23.9
Месяц бога Гефеста	24.10 - 23.09
Месяц бога Ареса	24.10 - 22.11
Месяц богини Артемиды	23.11 - 21.12

Месяц богини Гестии	22.12 - 20.01
Месяц богини Геры	21.01 - 19.02
Месяц бога Пойседона	20.02 - 20.03

Вариант 8

1. В зависимости от количества цифр в числе вывести единицы, десятки, сотни и тысячи.
2. Составить программу, в которой определить по введенной дате, к какому знаку языческого гороскопа он относится.

Знак	Дата
Медведь	11 декабря - 10 января
Росомаха	11 января - 10 февраля
Ворон	11 февраля - 10 марта
Горностай	10 марта - 10 апреля
Жаба	11 апреля - 10 мая
Кузнечик	11 мая - 10 июня
Хомяк	11 июня - 10 июля
Равлик	11 июля - 10 августа
Муравей	11 августа - 10 сентября
Хрущ	11 сентября - 10 октября
Бобер	11 октября - 10 ноября
Пес	11 ноября - 10 декабря

Вариант 9

1. Ввести год, месяц, день. В зависимости от даты вывести: прошлое, настоящее будущее.
2. Составить программу, в которой определить по введенной дате, к какому знаку старославянского гороскопа он относится.

Знак	Дата
Ярило (Яр)	(21 марта - 20 апреля)
Лада (Леля)	(21 апреля - 21 мая)
Летница (Леля)	(22 мая - 2 июня)
Кострома	(2 - 12 июня)
Додола (Доля)	(13 июня - 21 июня)
Велес (Коляда)	(22 июня - 6 июля и 8 июля - 22 июля)
Купало	(7 июля)
Даждьбог (Вышень)	(23 июля - 23 августа)
Майя (Сева)	(24 августа - 8 сентября)
Рожаницы	(9 - 11 сентября)
Мокошь	(12 - 27 сентября)
Сварожич	(28 сентября - 15 октября)
Мара (или Морена) .	(16 октября – 1 ноября)
Семаргл	(2 – 8 ноября)

Скипер-зверь	(9 – 30 ноября)
Скипер-зверь.	(9 – 30 ноября)
Выргонь.	(1 по 10 декабря)
Китоврас (или иначе Карачун).	(11 по 23 декабря)
Перун (или иначе Единорог).	(24 декабря по 20 января).
Стрибог.	(21 января по 20 февраля)
Род.	(21 февраля по 20 марта)

Вариант 10

1. Ввести рост человека и пол. В зависимости от роста и пола вывести низкий, средний и высокий.
2. Составить программу, в которой определить по введенной дате, к какому знаку кельтского гороскопа он относится.

Знак	Дата
Олень	(24 декабря- 20 января)
Журавль	(21 января- 17 февраля)
Тюлень	(18 февраля – 17 марта)
Медведь	(18 марта -14 апреля)
Змея	(15 апреля- 12 мая)
Пчела	(13 мая- 9 июня)
Выдра	(10 июня – 7 июля)
Кот	(8 июля- 4 августа)
Лосось	(5 августа – 1 сентября)
Лебедь	(2 сентября – 29 сентября)
Гусь	(30 сентября – 27 октября)
Сова	(28 октября – 24 ноября)
Ворон	(25 ноября – 23 декабря)

Вариант 11

1. Ввести температуру в помещении. Вывести комментарий: жарко, тепло, прохладно и холодно.
2. Составьте программу, которая бы по году рождения определяла знак по восточному календарю и стихию.

В восточном календаре за шестидесятилетний календарный цикл каждый год является не только годом какого-нибудь животного, но и относится в какой-нибудь стихии. Каждая стихия охватывает два года подряд. Стихии следуют в следующем порядке: Дерево, Огонь, Земля, Металл, Вода. 2004 год – это первый год стихии дерева

Обезьяна	Петух	Собака	Кабан	Крыса	Вол	Тигр	Кролик	Дракон	Змея	Лошадь	Овца
1920	1921	1922	1923	1024	1925	1926	1927	1928	1929	1930	1931
1932	1933	1934	1935	1936	1937	1938	1939	1940	1941	1942	1943
1944	1945	1946	1947	1948	1949	1950	1951	1952	1953	1954	1955
1956	1957	1958	1959	1960	1961	1962	1963	1964	1965	1966	1967
1968	1969	1970	1971	1972	1973	1974	1975	1976	1977	1978	1979
1980	1081	1982	1983	1984	1985	1986	1987	1988	1989	1990	1091
1992	1993	1994	1995	1996	1997	1998	1999	2000	2001	2002	2003

Вариант 12

1. В зависимости от температуры указать состояние воды: твердое, жидкое, парообразное.
2. Составить программу, в которой определить по введенной дате, к какому знаку индийского гороскопа он относится и какой у него символ.

Знак	Символ
АШВИНИ (с 13 по 27 апреля)	конь
БХАРАНИ (с 28 апреля по 11 мая)	слон
КРИТТИКА (с 12 по 25 мая)	овца
РОХИНИ (с 26 мая по 8 июня)	змея
МРИГАСИРА (с 9 по 21 июня)	змея
АРДРА (с 22 июня по 5 июля)	собака
ПУНАРВАСУ (с 6 по 19 июля)	кошка
ПУШИА (с 20 июля по 2 августа)	баран
АШЛЕША (с 3 по 16 августа)	кот
МАГХА (с 17 по 30 августа)	крыса
ПУРВА ФАЛГУНИ (с 13 августа по 13 сентября)	мышь
УТТАРА ФАЛГУНИ (с 14 по 27 сентября)	буйвол
ХАСТА (с 27 сентября по 20 октября)	буйволица
ЧИТРА (с 11 по 23 октября)	тигрица
СВАТИ (с 24 октября по 6 ноября)	буйвол
ВИШАХА (с 7 по 19 ноября)	тигр
АНУРАДА (с 20 ноября по 2 декабря)	олень
ЙЕШТА (с 3 по 15 декабря)	олень
МУЛА (с 16 по 28 декабря) Животное-символ:	собака
ПУРВА АШАДХА (с 29 декабря по 11 января)	обезьяна
УТТАРА АШАДХА (с 12 по 24 января	мангуста

ШРАВАНА (с 25 января по 6 февраля)	обезьяна
ДХАНИШТА (с7 по 19 февраля)	львица
ШАТАБХИШАК (с 20 февраля по 4 марта)	лошадь
ПУРВА БХАТРА (с 5 по 17 марта	лев
УТТАРА БХАТРА (с 18 по 31 марта)	корова
РЕВАТИ (с 1 по 12 апреля)	слон

Контрольные вопросы

1. Когда используется оператор выбора.
2. Какой тип может иметь ключевое выражение в операторе выбора
3. Обязательна ли ветка Else в операторе выбора.
4. Можно ли с помощью оператора выбора организовать проверку положительное число, отрицательное число или ноль.

Лабораторная работа № 6
Циклы с условием

В VBA имеется богатый выбор средств организации циклов, которые можно разделить на две основные группы — циклы с условием DO...Loop и циклы с перечислением For...Next.

Циклы типа DO...LOOP используются в тех случаях, когда заранее неизвестно, сколько раз должно быть повторено выполнение блока операторов, составляющего тело цикла. Такой цикл продолжает свою работу до тех пор, пока не будет выполнено определенное условие. Существуют четыре вида циклов DO...LOOP, которые различаются типом проверяемого условия и временем выполнения этой проверки. В табл. 6 приведен синтаксис этих четырех конструкций.

Таблица.

Конструкция	Описание
Do While <условие> <блокОператоров> Loop	Условие проверяется *до* того, как выполняется группа операторов, образующих тело цикла. Цикл продолжает свою работу, пока это условие выполняется (т. е. имеет значение True), иными словами, в этой конструкции указывается условие продолжения работы цикла
Do <блокОператоров> Loop While <условие>	Условие проверяется *после* того, как операторы, составляющие тело цикла, будут выполнены хотя бы один раз. Цикл продолжает свою работу, пока это условие остается истинным, иными словами, в этой конструкции указывается условие продолжения работы цикла
Do Until <условие> <блокОператоров> Loop	Условие проверяется до того, как выполняется группа операторов, образующих тело цикла. Цикл продолжает свою работу, если это условие еще не выполнено, и прекращает работу, когда оно станет истинным, иными словами, в этой конструкции указывается условие прекращения работы цикла
Do <блокОператоров> Loop Until <условие>	Условие проверяется после того, как операторы, составляющие тело цикла, будут выполнены хотя бы один раз. Цикл продолжает свою работу, если это условие еще не выполнено, а когда оно станет истинным, цикл прекращает работу, иными словами, в этой конструкции указывается условие прекращения работы цикла

Рассмотрим примеры использования операторов цикла с выходом по условию.

Задание 1

Написать программу, позволяющую пользователю угадать число из интервала от 1 до 100 сгенерированное компьютером. Сосчитать, сколько потребовалось попыток, чтобы пользователь угадал число

```
Option Explicit
Sub угадайка2()
Dim comp As Integer, igrok As Integer, k As Integer
Randomize
comp = Int(100 * Rnd)
k = 0
Do
igrok = InputBox("Угадайте задуманное число")
If igrok < comp Then MsgBox "Ваше число меньше"
If igrok > comp Then MsgBox "Ваше число больше"
k = k + 1
Loop Until igrok = comp
MsgBox "Угадали за " + Str(k) + " попыток"
End Sub
```

Обратите внимание на команду Option Explicit в первой строчке. Для чего она нужна? Вы, наверное, встречались с ситуацией, когда ваша программа считала не правильно. Это часто происходит, если в программе имена одной и тоже переменной записываются по-разному, например, **rez** и **res**. Наличие команды Option Explicit позволяет использовать в программе только имена переменных описанных в команде Dim.

Задание 2

Дано целое число. Определить сколько цифр в этом числе.

Чтобы определить количество цифр надо делить нацело данное число на 10 до тех пор, пока результат деления не станет равным 1.

```
Sub число()
Dim n As Integer, k As Integer
k = 0
n = InputBox("Введите число")
Do
n = n \ 10
k = k + 1
Loop While n > 1
MsgBox "Количество цифр " + CStr(k)
End Sub
```

Циклы широко используются при работе с ячейками в Excel.

Задание 3

Получить в первом столбце ряд чисел начиная с 2. Следующее число получается умножением предыдущего на 2. Ряд заканчивается, когда число становится больше 256.

```
Sub байт()
n = 2
i = 1
Do
Cells(i, 1).Value = n
n = n * 2
i = i + 1
Loop Until n > 256
End Sub
```

Цикл с выходом по условию используется для того, чтобы найти конец таблицы. В этом случае используется условие пока не достигнута пустая ячейка: Cells(i, 2).Text <> "".

Задание 4

Во втором столбце дан ряд целых чисел. Найти количество четных чисел в этом ряду и вывести в конце соответствующее сообщение.

```
Sub количество()
k = 0
i = 1
Do While Cells(i, 2).Text <> ""
If Cells(i, 2).Value Mod 2 = 0 Then k = k + 1
i = i + 1
Loop
Cells(i, 2).Value = k
Cells(i, 3).Value = "Количество четных чисел"
End Sub
```

Задание 5

В седьмом столбце находятся числа и текст.

F	G	H
	3	
	aa	
	5	
	cc	
	hh	
	2	
	ww	
	6	
	0	
	4	
	w	
	nn	
	rr	
	6	
	3	

Удалить из этого столбца все строки, содержащие текст. Просмотр столбца будем продолжать пока не встретиться пустая ячейка. Для того, чтобы определить, является ли содержимое ячейки числом, можно использовать функцию Val, преобразующую текст в число. Если результат этой функции больше 0, значит это число. Также надо исключить и удаления ячейки содержащей 0.

Для удаления ряда надо создать автомакрос, в котором удалить строку.

Этот макрос будет иметь вид:

```
Rows("15:15").Select
Selection.Delete Shift:=xlUp
```

Его надо преобразовать для ряда в котором будет находится текст.

```
Sub удаление()
i = 1
Do While Cells(i, 7).Text <> ""
If Val(Cells(i, 7).Value) = 0 Or Cells(i, 7).Text <> "0" Then
Rows(i).Select
Selection.Delete Shift:=xlUp
Else
i = i + 1
End If
Loop
End Sub
```

Обратите внимание, что переход к следующему ряду выполняется только, когда в ячейке находится число.

Задания для самостоятельной работы

Вариант 1

1. После каждого движения поршня разряжающего насоса их сосуда удаляется 20% находящегося газа. Сколько движений поршня нужно сделать, чтобы достичь давления P при начальном давлении P_n.
2. Дан столбец целых чисел. Закрасить желтым цветом все отрицательные числа.

Вариант 2

1. Спортсмен в первый день пробежал 10 км. Каждый следующий день он увеличивал дневную норму на 10% от результата предыдущего дня. Найти через сколько дней спортсмен будет пробегать более 20 км в день.
2. Генерировать числа в столбец в интервале от 1 до 100 пока не встретиться число 50. Найти сумму сгенерированных чисел.

Вариант 3

1. Спортсмен в первый день пробежал 10 км. Каждый следующий день он увеличивал дневную норму на 10% от результата предыдущего дня. Найти через сколько дней спортсмен пробежит суммарный путь более 100 км.
2. Дан столбец целых чисел. Просуммировать все числа меньше 100 и вывести в конце столбца эту сумму.

Вариант 4

1. Сотрудник взял беспроцентный кредит в размере S рублей. Условия погашения кредита следующие: каждый месяц выплачивается сумма вдвое превышающая сумму, выплаченную в предыдущий месяц. Через сколько месяцев кредит будет погашен, если в первый месяц сотрудник выплатил N рублей.

2. Дан столбец целых чисел. Найти количество чисел, равных первому числу и вывести это количество в конце столбца.

Вариант 5

1. Шары расположены в форме треугольника так, что в первом ряду находится один шар, во втором – два, в третьем – три и так далее. Сколько рядов удастся построить, если имеется N шаров?
2. Дан столбец чисел. Просуммировать все числа в этом столбце попадающие в интервал от -20 до 20 и вывести результат в конце.

Вариант 6

1. Собран урожай из N кг яблок. Определить, через сколько дней будет вынесен весь урожай, если каждый день количество вынесенных корзин удваивается. В одной корзине помещается 5 кг.
2. Дан столбец целых чисел. Удалить стоки, содержащие нули.

Вариант 7

1. Задать целое число и подсчитать сумму цифр в этом числе используя операции целочисленного деления - \ и mod – остатка от деления.
2. Дана строка, содержащая ячейки с текстом. Напечатать в конце этой строки «Итого».

Вариант 8

1. Дан текст. Определить, сколько слов в этом тексте. Слова отделяются пробелами.
2. Дана строка, содержащая ячейки с текстом. Напечатать под этой строкой целые числа, начиная с 1.

Вариант 9

1. На складе можно разместить N ящиков помидор. В первый день привезли K ящиков в последующие дни количество привезенных ящиков увеличивалось на 2. Через сколько дней заполнится склад.
2. Дан столбец, содержащий ячейки с текстом. Удалить в этом столбце тексты из одного символа.

Вариант 10

1. Сколько чисел надо взять в последовательности 1 + 2 + 3 + 4 + …, чтобы получить число, больше чем N?
2. Дан столбец, содержащий ячейки с текстом. Добавить в начале каждого текста номер (1, 2, …).

Вариант 11

1. Дано натуральное n. Выяснить, входит ли цифра 3 в запись числа n^2. Операции со строками не использовать.
2. Первый столбец содержит текст. Вставит перед ним новый столбец, и вывести в нем числа натурального ряда.

Вариант 12

1. Задать целое число и получить новое число из первой и последней цифры этого числа. Строки не использовать.
2. Дано целое 4-значное число. Вывести столбец целых чисел полученных делением на 2, начиная с заданного числа до 0.

Контрольные вопросы

1. Чем отличаются циклы с «пост»-условием от циклов с «пред»-условием.
2. Что будет, если после Do и Loop не поставить условие.
3. Какое условие записывается после слова Until/
4. Как изменится макрос Угадайка если в операторе цикла слово Until заменить на слово While/
5. Что надо изменить в макросе количество, чтобы вместо количества считать сумму.

Лабораторная работа № 7
Цикл со счетчиком

Цикл со счетчиком имеет следующий синтаксис:

```
For счетчик=начальноеЗначение To конечноеЗначение [Step приращение] блокОператоров>
Next   [счетчик]
```

Несколько пояснений к приведенному описанию:

- Необязательные конструкции, как обычно, заключены в квадратные скобки
- *приращение* — может быть как положительным, так и отрицательным числом. Если использовать отрицательное приращение, то конечное значение должно быть меньше либо равно начальному значению для того, чтобы тело цикла выполнилось хотя бы один раз
- После завершения работы цикла For...Next переменная, которая использовалась в качестве счетчика, получает значение, обязательно превосходящее конечное значение в том случае, если приращение положительно, и строго меньшее конечного значения, если приращение отрицательно
- Если начальное и конечное значения совпадают, тело цикла выполняется лишь один раз

Выход из циклов и процедур

Обычно выполнение процедуры заканчивается после выполнения ее последнего оператора, а выполнение цикла — после выполнения тела цикла несколько раз, по достижении условия завершения его работы. Однако в некоторых случаях бывает нужно прекратить выполнение процедуры или цикла досрочно, избежав выполнения лишних операторов процедуры или лишних повторений цикла.

Например, если при выполнении процедуры произошла ошибка, которая делает продолжение ее работы бессмысленным, можно выполнить команду немедленного выхода из процедуры. Другой пример: если цикл For...Next используется для поиска нужного значения в столбце, то когда нужное значение найдено, нет смысла продолжать дальнейший перебор ячеек. Досрочный выход из управляющей конструкции можно осуществить с помощью одного из операторов Exit. Для досрочного выхода из циклов Do …Loop используется оператор Exit Do, а для выхода из циклов For — оператор Exit For. Для досрочного выхода из процедур и функций используются операторы Exit Sub и Exit Function, соответственно.

Следует, однако, отметить, что хотя использование оператора Exit может быть вполне оправданным, следует избегать излишнего употребления этого оператора, прибегая к нему только в крайних случаях. Излишне частое его применение затрудняет понимание написанного текста программы и его отладку.

Задание 1

За 5 попыток угадать число, задуманное компьютером. Задуманное число лежит в интервале от 1 до 100.

```
Sub Угадайка3()
Dim x As Integer, n As Integer, k As Integer
Dim i As Integer
Randomize
x = Int(Rnd(1) * 100) + 1
k = 0
For i = 1 To 5
n = InputBox("Попытка " + CStr(i) + " Введите число от 1 до 100", _
"Угадай число с 5 попыток")
If x > n Then
MsgBox "Ваше число меньше"
k = k + 1
ElseIf x < n Then
MsgBox "Ваше число больше"
k = k + 1
Else
MsgBox "Угадали число с " + CStr(k) + " попытки"
Exit For
End If
Next
If k = 5 Then
MsgBox "Не угадали"
End If
End Sub
```

Задание 2

Записать функцию, которая в выделенном диапазоне-столбце находит номер ряда, с заданным значением.

```
Function Номер_Ряда(Диапазон As Range, значение) As Integer
Dim k As Integer
Dim i As Integer, n As Integer, m As Integer
Номер_Ряда = 0
n = Диапазон.Row
m = Диапазон.Column
k = Диапазон.Rows.Count
For i = n To n + k - 1
If Cells(i, m).Value = значение Then
Номер_Ряда = i
Exit For
End If
Next
End Function
```

Обратите внимание, что для аргумента *значение* не указан тип. Это означает, что данная функция может находить и числа и текст.

Задание 3

В выделенном диапазоне-ряде получить ряд чисел кратный 3.

```
Sub Ряд_3()
Dim r As Range, k As Integer
Dim i As Integer, n As Integer, m As Integer
Set r = Selection
n = r.Row
m = r.Column
k = r.Columns.Count
For i = 1 To k
Cells(n, m + i - 1).Value = i * 3
Next
End Sub
```

Задание 4

В заданном диапазоне-столбце выделить зеленым все числа, равные минимальному элементу.

```
Sub выделение_минимума()
Dim r As Range
Dim k As Integer, min As Single, n_min As Integer
Dim i As Integer, n As Integer, m As Integer
Set r = Selection
n = r.Row
m = r.Column
k = r.Rows.Count
min = Cells(n, m).Value
For i = n To n + k - 1
If Cells(i, m).Value < min Then
min = Cells(i, m).Value
End If
Next
For i = n To n + k - 1
If Cells(i, m).Value = min Then
Cells(i, m).Interior.ColorIndex = 4
End If
Next
End Sub
```

Задание 5

Найти максимальный элемент в столбце и поставить в него курсор.

```
Sub Максимум()
Dim r As Range
Dim k As Integer, max As Single, n_max As Integer
Dim i As Integer, n As Integer, m As Integer
Set r = Selection
n = r.Row
m = r.Column
k = r.Rows.Count
max = Cells(n, m).Value
n_max = n
For i = n To n + k - 1
If Cells(n + i - 1, m).Value > max Then
max = Cells(n + i - 1, m).Value
n_max = n + i - 1
End If
Next
Cells(n_max, m).Select
End Sub
```

Ввести и отладить (найти ошибку) данный макрос.

Таблица стандартных алгоритмов.

этап		Сумма **S**	Произведение **P**	Количество **k**	Минимум Min
До цикла		S=0	P=1	k=0	Min=первый элемент
Начало цикла					
В цикле	1	Проверка, надо ли данный элемент суммировать (необязательно)	Проверка, надо ли данный элемент умножать (необязательно)	Проверка, удовлетворяет ли элемент условию	Проверка, элемент меньше Min?
		S=S+элемент	P=P*элемент	k=k+1	Min:=элемент
Переход к следующему элементу. Конец цикла					

Задания для самостоятельной работы

Вариант № 1

1. Ввести дневную и ночную температуры за неделю и определить день с самым небольшим перепадом температур.
1. Сгенерировать в выделенном столбце диапазоне числа от -100 до 100. Закрасить красным цветом значение, которое максимально отличается от среднего арифметического значения в этом диапазоне.

Вариант № 2

1. Спортсмен в первый день пробежал 10 км. Каждый следующий день он увеличивал дневную норму на 10% от результата предыдущего дня. Найти какой путь пробежит спортсмен на 7 день;
2. Дан ряд-диапазон чисел. Найти все минимальные значения в этом диапазоне. Удалить эти числа и найти новое минимальное значение.

Вариант № 3

1. Ежемесячная стипендия студента составляет А руб., а расходы на проживание превышают стипендию и составляют В руб. в месяц. Рост цен ежемесячно увеличивает расходы на 3%. Составьте программу расчета суммы денег, которую необходимо попросить у родителей, чтобы можно было прожить учебный год (10 месяцев) используя только эти деньги и стипендию.

2. Дан ряд диапазон чисел. Найти максимальное значение в этом диапазоне, обнулить все максимальные значения и найти среднеарифметическое значение в этом диапазоне после обнуления.

Вариант № 4

1. Одноклеточная амеба каждые 3 часа делится на 2 клетки. Определить, сколько амеб будет через 3, 6, 9, 12, …, 24 часа.
2. Дан ряд диапазон чисел. Удалить в диапазоне все неотрицательные числа и найти максимальный элемент, среди оставшихся чисел.

Вариант № 5

1. Ввести средние месячные температуры за год и определить, сколько в году месяцев с отрицательной средней температурой.
2. Дан ряд диапазон чисел. Удалить все числа меньшие среднеарифметического и найти среди оставшихся минимальный элемент.

Вариант № 6

1. Задать закупочную цену 10 товаров и продажную цену. Найти товар, который принес самую большую прибыль.
2. Дан ряд диапазон чисел. Найти среднее арифметическое среди отрицательных элементов и умножить на это среднее арифметическое минимальный элемента в этом ряду

Вариант № 7

1. Клиент банка взял кредит в размере 500000 на год с условием, что ежемесячно, кроме долга он дополнительно выплачивает проценты в размере 10% от оставшейся суммы. Найти сумму, которую клиент выплатит банку за год.
2. Дан столбец диапазон чисел. Найти минимальное значение среди элементов, стоящих в нечетных столбцах. Найти количество чисел, равных этому минимальному значению.

Вариант № 8

1. Даны дневные и ночные температуры за неделю. Найти среднюю дневную температуру и среднюю ночную температуру за неделю, а так же разницу между этими средними температурами.
2. Дан ряд диапазон целых чисел. Удалить все нечетные числа и найти среди оставшихся чисел максимальный элемент. Просуммировать числа, стоящие после этого значения.

Вариант № 9

1. В группе 12 человек .Ввести отметки за экзамен и найти: количество двоек, количество пятерок и среднюю отметку за экзамен.
2. Дан столбец диапазон чисел. Переписать жирным цветом число максимально близкое к среднему арифметическому числу среди этих чисел.

Вариант № 10

1. Если вес пушного зверька в возрасте от 6 до 8 месяцев превышает 7 кг. То необходимо снизить дневное потребление витаминного концентрата на 125 г. Ввести возраст и вес для 10 зверьков и выяснить, на сколько килограммов в месяц снизится потребление витаминного концентрата.
2. Дан столбец диапазон чисел. Закрасить синим и просуммировать все числа стоящие перед последним минимальным значением (если несколько ячеек имеют это значение).

Вариант № 11

1. Ввести возраст и рост для 10 учеников. Сколько учеников могут заниматься в баскетбольной секции, если туда принимают детей ростом не менее 160 см., а возраст не должен превышать 13 лет.
2. Сгенерировать в выделенном столбце диапазоне числа от -100 до 100. Найти минимальное значение и количество отрицательных чисел.

Вариант № 12

1. Дано 10 блюд с указанием количества калорий в 100 г. И вес каждого блюда в граммах. Сколько блюд имеют общую калорийность менее 100 калорий.
2. Дан столбец диапазон чисел. Закрасить зеленым цветом все числа, стоящие между минимальным и максимальным значением.

Лабораторная работа № 8
Вложенные циклы

В ранее рассмотренных примерах рассматривались диапазоны, являющееся частью строки и столбца. Для работы с этими диапазонами достаточно было одного цикла, в котором перебирались номера строк или номера столбцов. В общем случае диапазон состоит из нескольких строк и столбцов.

Задание 1

В заданном диапазоне получить целые числа из интервала от -100 до 100.

```
Sub Генерация()
Dim i As Integer, j As Integer
Dim n As Integer, m As Integer
Dim k As Integer, l As Integer
Dim x As Integer
Randomize
n = Selection.Row 'начальный ряд диапазона
m = Selection.Column 'начальный столбец диапазона
k = Selection.Rows.Count 'количество рядов диапазона
l = Selection.Columns.Count 'количество колонок диапазона
For i = n To n + k - 1  'перебор строк
For j = m To m + l - 1  'перебор столбцов
x = Int(Rnd(1) * 200) - 100 'генерация числа
Cells(i, j).Value = x  'запись числа в ячейку
Next
Next
End Sub
```

Задание 2

Выделить красным цветом минимальный элемент в заданном диапазоне.

```
Sub минимум()
Dim i As Integer, j As Integer
Dim n As Integer, m As Integer
Dim k As Integer, l As Integer
Dim min As Single, rmin As Integer, cmin As Integer
n = Selection.Row
m = Selection.Column
k = Selection.Rows.Count
l = Selection.Columns.Count
min = Cells(n, m).Value
rmin = n
cmin = m
For i = n To n + k - 1
For j = m To m + l - 1
If Cells(i, j).Value < min Then
min = Cells(i, j).Value
rmin = i
cmin = j
End If
Next
Next
Cells(rmin, cmin).Font.Bold = True
Cells(rmin, cmin).Font.ColorIndex = 3
End Sub
```

В рассмотренных выше заданиях перебор ячеек выполнялся построчно. Иногда требуется изменить порядок просмотра.

Задание 3

В заданном диапазоне подсчитать количество отрицательных значений в каждом столбце, и записать это количество в конце каждого столбца. Для этого примера надо изменить порядок просмотра: при фиксированном столбце перебирать ряды.

```
Sub количество_по_столбцам()
Dim i As Integer, j As Integer
Dim n As Integer, m As Integer
Dim k As Integer, l As Integer
Dim kol As Integer
n = Selection.Row 'начальный ряд диапазона
m = Selection.Column 'начальный столбец диапазона
k = Selection.Rows.Count 'количество рядов диапазона
l = Selection.Columns.Count 'количество колонок диапазона
For j = m To m + l - 1  'перебор столбцов
kol = 0
For i = n To n + k - 1  'перебор строк
If Cells(i, j).Value < 0 Then kol = kol + 1
Next 'ряды закончились
Cells(n + k, j).Value = kol 'запись количества в конце j-го столбца
Next
End Sub
```

Цикл For Each … Next

Цикл *For Each..Next* не использует счетчик цикла. Циклы *For Each..Next* выполняются столько раз, сколько имеется элементов в определенной группе. Цикл *For Each..Next* выполняется один раз для каждого элемента в группе.

For Each <*элемент*> In <*группа*>

 <операторы>

Next

В нашем случае *элемент* – это ячейка, а *группа* – это выделенный диапазон.

Задание 4

В заданном диапазоне чисел закрасить ячейки с нулями желтым цветом.

```
Sub ячейки()
Dim cl As Range
For Each cl In Selection
 If cl.Value = 0 Then
   cl.Interior.ColorIndex = 6
   cl.Interior.Pattern = xlSolid
 End If
Next
End Sub
```

Использование цикла *For Each..Next* зачастую сокращает код программы по сравнению с вложенными циклами.

Задание 5

Приведем пример программы нахождения минимума с использованием цикла *For Each..Next.*

```
Sub минимум2()
Dim n As Integer, m As Integer
Dim min As Single, rmin As Integer, cmin As Integer
Dim cl As Range
n = Selection.Row
m = Selection.Column
min = Cells(n, m).Value
rmin = n
cmin = m
For Each cl In Selection
If cl.Value < min Then
min = cl.Value
rmin = cl.Row
cmin = cl.Column
End If
Next
Cells(rmin, cmin).Font.Bold = True
Cells(rmin, cmin).Font.ColorIndex = 3
End Sub
```

Задания для самостоятельной работы

Вариант 1

1. Дан диапазон чисел. Каждую строку переписать в обратном порядке.
2. В заданном диапазоне у всех отрицательных элементов изменить знак на противоположный.

Вариант 2

1. Дан диапазон чисел. Заменить минимальное значение в каждом столбце на максимальное значение в этом же столбце.
2. В заданном диапазоне все числа меньшие среднего арифметического закрасить красным цветом.

Вариант 3

1. Дан диапазон чисел. Найти в каждой строке сумму положительных чисел и записать в конце каждой строки это количество.
2. В заданном диапазоне все числа из интервала от 0 до 10 заменить на 0.

Вариант 4

1. Дан диапазон чисел. Найти в каждом ряду максимальное значение и записать в конце каждого ряда это значение.
2. В заданном диапазоне все числа нечетные отрицательные числа закрасить зеленым цветом.

Вариант 5

1. Дан диапазон чисел. Найти в каждой строке среднее арифметическое значение и записать его в конце каждой строки.
2. В заданном диапазоне найти количество нулевых элементов, и заменить на это количество первый элемент в диапазоне.

Вариант 6

1. Дан диапазон чисел. Заменить первое значение в каждом столбце на сумму ячеек в этом столбце
2. В заданном диапазоне все четные числа закрасить желтым цветом.

Вариант 7

1. Дан диапазон чисел. Найти в каждой строке количество чисел равных первому числу в диапазоне и записать в конце каждой строки это количество.
2. В заданном диапазоне все числа равные заданному числу выделить жирным шрифтом.

Вариант 8

1. Дан диапазон чисел. Найти в каждом ряду минимальное значение и записать в конце каждого ряда это значение.
2. В заданном диапазоне найти среднее арифметическое среди отрицательных чисел и заменить нулевые значения на это число.

Вариант 9

1. Дан диапазон чисел. Найти в каждом ряду количество четных чисел и записать в конце каждого ряда это количество..
2. В заданном диапазоне найти сумму чисел, не попадающих в интервал от 0 до 30.

Вариант 10

1. Дан диапазон чисел. Заменить минимальное значение в каждом столбце на максимальное значение в этом же столбце.
2. В заданном диапазоне все нечетные числа увеличить на 1.

Вариант 11

1. Дан диапазон чисел. Найти в каждой строке сумму чисел, значение которых меньше последнего числа в данной строке и записать в конце каждой строки эту сумму.

2. В заданном диапазоне найти число по модулю максимально близкое к 0 и заменить его на 0.

Вариант 12

1. Дан диапазон чисел. Найти в каждом ряду минимальное и максимальное значение и записать в конце каждого разность между этими значениями.
2. В заданном диапазоне найти количество чисел, равных по модулю 1 и заменить эти единицы на вычисленное количество.

Лабораторная работа № 9
Массивы

Массив - это коллекция переменных, которые имеют общие имя и базовый тип. Все элементы данных, сохраняемых в массиве, должны иметь один и тот же тип. Информация, сохраненная в массиве, может быть доступна в любом порядке.

Массив позволяет сохранять и манипулировать многими элементами данных посредством единственной переменной. Обработку массивов значительно упрощает использование циклов.

Одномерные массивы

Одномерный массив - это самый простой вариант массива, использующий обыкновенный список данных. Например:

Вася, Петя, Коля, Миша, Ваня, Слава, Игорь, Юра, Саша, Вова

Это строковый массив, состоящий из 10 элементов. Дадим ему название *My_Array*.

Нумерация элементов в массиве начинается с 0. Такая система нумерации довольно распространена в программировании и называется нумерацией с *нулевой базой*.

Для доступа к данным, хранящимся в определенном элементе массива, следует указывать имя массива с последующим числом, называемым индексом элемента. Индекс всегда заключается в круглые скобки. Например: *My_Array(3)* - этому элементу нашего массива соответствует "Миша" (не забывайте, что по умолчанию нумерация элементов массива начинается с 0).

Поскольку нумерация с нулевой базой не очень удобна (т.к. мы привыкли считать с 1, а не с 0), то в VBA имеется директива компилятора, позволяющая исправить это "неудобство": *Option Base*.

Директива компилятора имеет два варианта написания:

Option Base 0 - индексы массивов начинаются с 0 (установка по умолчанию)

Option Base 1 - индексы массивов начинаются с 1

Данная директива компилятора помещается в область объявлений модуля перед объявлениями любых переменных, констант или процедур. Нельзя помещать *Option Base* внутри процедуры. Можно иметь только один оператор Option Base в модуле, который влияет на все массивы, объявляемые в модуле.

Многомерные массивы

Одномерные массивы хорошо подходят для представления простых списков данных. Однако часто бывает необходимо представить таблицы данных в программах с организацией данных в формате строк и столбцов, подобно ячейкам в рабочих листах Excel. Для этого необходимо использовать *многомерные массивы*. Так адрес каждой ячейки листа состоит

из двух чисел, одно из которых (номер строки) является первым индексом, а второе (номер столбца) - вторым индексом массива. Такой массив называется *двумерным массивом*. Добавив еще номер листа, получим *трехмерный массив*. VBA позволяет создавать массивы, имеющие до 60 измерений.

Статические и динамические массивы

Массивы, не меняющие число своих элементов, называются *статическими массивами*. Примером такого массива может служить вышеприведенный массив *My_Array*, содержащий 10 элементов.

Однако бывают ситуации, когда изначально неизвестно количество элементов в массиве, или же, в процессе работы это количество может изменяться. Такие массивы называются *динамическими массивами*.

Динамический массив может увеличиваться или сжиматься, чтобы вмещать точно необходимое число элементов без напрасного расходования памяти.

Объявление массивов

Объявление массива с использованием оператора *Dim* имеет следующий синтаксис:

Dim *VarName*([*Subscripts*]) [As *Type*]

VarName - любое имя массива, использующее допустимый идентификатор имени;

Subscripts - измерение массива. Если размерность массива больше единицы, то Subscripts разделяются запятыми.

Оператор *Subscripts* имеет следующий синтаксис:

[*lower* To] *upper* [,[*lower* To] *upper*]..

lower - определяет нижний диапазон допустимых индексов для массива (необязательный аргумент);

upper - определяет верхний предел для индексов массива (обязательный аргумент).

Примеры правильного объявления массивов:

Dim Array_Str (1 To 10) As String - одномерный статический строковый массив, включающий 10 элементов;

Dim Array_Var() - динамический массив;

Dim Array_Mult (0 To 5, 0 To 7) As Integer - двумерный статический массив целых чисел, включающий 6*8=48 элементов.

При объявлении массивов следует помнить, что включение оператора *Subscripts* в объявлении массива создает статический массив с фиксированным числом элементов, пропуск оператора *Subscripts* в объявлении массива создает динамический массив, а установка директивы компилятора *Option Base* влияет на общее число элементов в массиве.

Изменение размерности динамического массива

Могут сложиться обстоятельства, при которых точно неизвестно, сколько элементов потребуется в массиве. В VBA имеется возможность при помощи оператора *ReDim* переопределять размерность массива, а во время объявления не указывать его размерность.

Синтаксис *ReDim*:

ReDim [Preserve] *varname*(*subscripts*) [As *Type*] [, *varname*(*subscripts*) [As *Type*]]

varname - имя существующего массива;

subscripts - размерность существующего массива;

Type - любой тип VBA. Необходимо использовать отдельный оператор As Type для каждого массива, который определяется;

Preserve - необязательный аргумент. Его использование приводит к тому, что данные уже имеющиеся в массиве, сохраняются после изменения его размерности.

Примеры правильного использования оператора *ReDim*:

Dim Array_Month() As String - одномерный строковый динамический массив

ReDim Array_Month(29) - устанавливет размерность динамического массива равную 29 элементам

ReDim Array_Month(1 To 30) - изменяет размер массива до 30 элемента

ReDim Preserve Array_Month(1 To 31) - изменяет размер массива до 31 элемента, сохраняя содержимое

Dim Array_DBL() As Single - объявляет динамический массив

ReDim Array_DBL(2, 9) - делает массив двумерным

ReDim Array_DBL(3, 7) - изменяет размер двумерного массива

ReDim Preserve Array_DBL(1 To 3, 1 To 5) - изменяет последний размер массива, сохраняя содержимое

ВАЖНО

Можно изменять только последнее измерение многомерного массива, когда используется ключевое слово *Preserve*.

Задание 1

Получить массив из 40 элементов. Элементами массива являются целые числа из интервала от 1 до 40, причем числа в массиве не повторяются.

В приведенном коде массив состоит из 100 элементов.

```
Sub генерация()
Dim mas(100) As Integer, n As Integer
Dim flag As Boolean
Randomize
k = 1
mas(1) = Int(Rnd(1) * 100) + 1
Do While k < 100
n = Int(Rnd(1) * 100) + 1
flag = False
For i = 1 To k
If n = mas(i) Then flag = True
Next
If flag = False Then
k = k + 1
mas(k) = n
End If
Loop
For i = 1 To 100
Cells(i, 1).Value = mas(i)
Next
End Sub
```

Задание 2

Даны температуры за неделю. Найти день с самой низкой температурой.

```
Sub температура_недели()
Dim T(7) As Integer, i As Integer
Dim imin As Integer
Dim d, n As Integer
d = Array("Понедельник", "Вторник", "Среда", "Четверг", "Пятница", _
"Суббота", "Воскресение")
For i = 1 To 7
T(i) = InputBox("Введите температуру за " + d(i))
Next
imin = 1
For i = 1 To 7
If T(i) < T(imin) Then imin = i
Next
MsgBox "Самая низкая температура в " + d(imin) + " " + CStr(T(imin) + " град")
End Sub
```

Задание 4

Имеется список участников кастинга с указанием полученных баллов.

Получить список допущенных к конкурсу. Это участники набравшие баллы, больше среднего арифметического от всех баллов.

```
Sub конкурс()
Dim FioX() As String, BallX() As Integer
Dim FioY() As String, BallY() As Integer
Dim n As Integer, i As Integer, k As Integer
Dim s As Integer, sr As Single
i = 1
Do While Cells(i, 1).Text <> ""
i = i + 1
Loop
n = i - 2
ReDim FioX(n), BallX(n)
For i = 1 To n
FioX(i) = Cells(i + 1, 1).Value
BallX(i) = Cells(i + 1, 2).Value
Next
s = 0
For i = 1 To n
s = s + BallX(i)
Next
sr = s / n
k = 0
For i = 1 To n
If BallX(i) > sr Then
k = k + 1
ReDim Preserve FioY(k)
ReDim Preserve BallY(k)
FioY(k) = FioX(i)
BallY(k) = BallX(i)
End If
Next
k = UBound(FioY)
For i = 1 To k
Cells(i + 1, 4).Value = FioY(i)
Cells(i + 1, 5).Value = BallY(i)
Next
End Sub
```

Задания для самостоятельной работы

Вариант 1

1. Дан список дневных и ночных температур за неделю. Определить день, с максимальной разницей дневных и ночных температур.
2. Дан список студентов с отметками по информатике. Получить список отличников.

Вариант 2

1. Дано количество абитуриентов, поступавший последние 10 лет определить год с самым большим количеством абитуриентов.
2. Дан список поваров с указанием их цены. Получить список товаров с ценой ниже средней арифметической цены.

Вариант 3

1. Даны среднемесячные температуры за год. Определить, сколько месяцев среднемесячная температура была отрицательной.
2. Дан список учащихся с указанием их роста. Получить список учащихся, которых рост отличается от максимального роста не более чем на 5 сантиметров.

Вариант 4

1. Дано количество выпускников с красным дипломом за последние 10 лет. Определить год с самым низким количеством красных дипломов.
2. Дан список городов и указанием количества их жителей. Получить список городов с населением более миллиона.

Вариант 5

1. В группе 25 студентов. Дан возраст каждого студента. Определить, сколько в группе студентов моложе 21 года.
2. Дан список коробок конфет с указанием их цены. Получить список конфет, которые отличаются от самой дорогой коробки на 50 руб.

Вариант 6

1. Дано количество осадков выпавших за сутки в неделю. Найти день с максимальным количеством осадков.
2. Дан список автомобилей с указанием автопробега. Получить список с автопробегом менее 1000 км.

Вариант 7

1. Даны прибыли магазина в течение года. Найти месяц, в котором прибыль была минимальна.
2. Дан список пациентов с указанием их возраста. Получить список детей поциентов.

Вариант 8

1. Дано количество комнат в квартирах 20-квартирного дома. Найти количество однокомнатных квартир в этом доме.
2. Дан список блюд с указанием их калорийности. Найти блюда, у которых калорийности отличается от блюда с минимальной калорийностью на 20.

Вариант 9

1. Даны среднемесячные температуры за год. Определить в какое полугодие средняя температура за полугодие была больше.
2. Дан список дней с указанием температуры. Найти дни с температурой равной минимальной температурой за этот период.

Вариант 10

1. Дано количество отличников в классе за каждый год обучения. Найти в какой год обучения количество отличников было максимально.
2. Дан список товаров с указанием первоначального количества и количества проданного товара. Получить список товаров, у которых количество оставшегося товара равно 0.

Вариант 11

1. Даны результаты соревнований для 10 спортсменов. Определить результат спортсмена, занявшего второе место.
2. Дан список спектаклей с указанием количества проданных билетов. Подучить список спектаклей, у которых количество проданных билетов превышает 300 штук.

Вариант 12

1. Дано количество осадков, выпавших за сутки во второй декаде. Найти день с максимальным количеством осадков.
2. Дан список спортсменов. участвующих в соревнованиях. Получить список спортсменов женщин.

Лабораторная работа № 10
Объекты и коллекции

VBA является идеальным инструментом для изучения основ объектно-ориентированного программирования, так как имеет встроенные объекты (рабочие книги и листы, ячейки, документы и выделенные фрагменты текста), их свойства и методы.

Объектно-ориентированное приложение организует данные и выполняемые операторы программного кода в связанные объекты, что облегчает разработку, организацию и работу со сложными структурами данных и действиями, выполняемыми над этими данными (или с ними). Каждый объект в программном приложении содержит данные и операторы, связанные вместе и образующие единый элемент. Большинство приложений содержат много различных типов объектов.

В Excel объектами являются рабочие книги, листы, диапазоны данных, таблицы, графические объекты, диалоговые окна и само приложение Excel.

Как и объекты реального мира, объекты VBA имеют различные присущие им качества или *свойства (properties).* Объекты в Excel имеют свойства, определяющие их вид и поведение: рабочий лист Excel может быть видимым или нет, строки в рабочем листе или таблице имеют свойство высоты, столбцы имеют свойство ширины и так далее.

Свойства регулируют вид и поведение объекта. Чтобы изменить вид и поведение объекта, необходимо изменить его свойства. В Excel можно изменить поведение рабочего листа, изменяя свойство вычисления с автоматического на ручное, или можно изменить вид рабочего листа, задав новый цвет для текста или графики в листе.

Чтобы узнать о текущем виде и поведении объекта, необходимо узнать о его свойствах. Вы можете определить диск, папку и имя рабочей книги Excel рассмотрев свойство **FullName** этой рабочей книги или документа.

Некоторые свойства объекта можно изменять, некоторые — нет. Например, в рабочей книге Excel можно изменить имя автора книги, изменяя свойство **Author,** но нельзя изменить свойство **Name** рабочей книги. (Свойство **Name** рабочей книги содержит имя файла на диске и не может быть изменено без создания нового дискового файла или переименования файла рабочей книги вне Excel).

Некоторые объекты имеют свойства с одними и теми же или подобными именами: объекты **Application, Workbook** и **Worksheet** в Excel — все имеют свойство **Name.** Каждый объект рабочего листа сохраняет данные для своих свойств внутри самого рабочего листа вместе с пользовательскими данными. [*Пользовательские данные (user data)* — это любые данные, сохраняемые объектом, которые поступают непосредственно от пользователя подобно данным, содержащимся в ячейках рабочего листа.]

Объекты реального мира почти всегда имеют тот тип присущего им поведения или действия, который они могут выполнить. Объекты VBA также

имеют поведение и возможности, называемые *методами (methods).* Объект рабочей книги Excel, например, имеет встроенную способность добавлять к себе новый рабочий лист — *метод* добавления рабочих листов (называемый **Add).** Другой пример: документ Word имеет встроенную способность проверять орфографию текста в документе — *метод* проверки орфографии (называемый, как вы могли догадаться, **CheckSpelling).**

Методы изменяют значения свойств объектов; методы выполняют также действия с данными (или над данными), сохраняемыми объектом. Методы во многом похожи на процедуры VBA, с которыми вы уже знакомы, но связаны с объектом; к методам объекта можно обратиться, только используя объект. Это может казаться сложным, но на деле все проще. Для вызова метода объекта (функции обработки данных объекта) указывается не только наименование метода, но и объект (имя), которому принадлежит метод.

Одной из причин того, что объектно-ориентированное программирование стало популярной техникой проектирования, является то, что оно позволяет разработчикам программного обеспечения создавать более эффективные программы с совместно используемым кодом. Вместо сохранения отдельной копии кода для каждого метода каждого объекта VBA-объекты одного и того же типа совместно используют код своих методов.

Несмотря на то, что объекты одного и того же типа совместно используют код для своих методов, метод считается частью объекта; когда вы вызываете определенный метод для конкретного объекта, метод воздействует только на объект, посредством которого вы обращаетесь к этому методу.

После того как объект создан (если это — не встроенный объект), нет необходимости беспокоиться о том, как работают его методы и как объект сохраняет или манипулирует данными. Все, что нужно знать, — как указать объект и как указать метод, который необходимо использовать (или определенное свойство, которое нужно выбрать и изменить). Встроенный код для объекта VBA выполняет остальную работу автоматически.

Операторы программ VBA, использующие объекты, обычно выполняют одно или несколько из следующих действий:

- определяют текущее состояние или статус объекта путем выборки значения, сохраняемого в определенном свойстве;
- изменяют состояние или статус объекта установкой значения, сохраненного в определенном свойстве;
- используют один из методов объекта, обеспечивая выполнение объектом одной из его встроенных задач.

Например, можно определить имя активного в данный момент рабочего листа в Excel, выполняя выборку строки, сохраняемой в свойстве **Name** рабочего листа. (Свойство **Name** рабочего листа содержит имя рабочего листа, как показано на ярлычке листа.) Чтобы изменить имя рабочего листа, необходимо присвоить новую строку свойству **Name** этого рабочего листа.

Для добавления рабочего листа в рабочую книгу используется метод **Add** рабочей книги.

Прежде чем применять свойства и методы некоторого объекта, их следует сначала определить.

В операторах VBA используется следующий общий синтаксис для определения свойства или метода объекта:

<Имя обьекта>.<свойство или метод>

В таблице 1 приведены несколько из наиболее важных объектов (с точки зрения программиста VBA) в Excel. В таблице показано имя объекта и краткое описание этого объекта. (Excel содержит намного больше объектов, чем перечислено в табл 1**.)**

Таблица 1

Объект	Описание
Application	Само приложение Excel (host-приложение)
Chart	Диаграмма в рабочей книге
Font	Этот объект содержит атрибуты шрифта и стиля для текста, отображаемого в рабочем листе
Name	Заданное имя для диапазона ячеек рабочего листа
Range	Диапазон ячеек (одна или более) или именованный диапазон в рабочем листе
Window	Любое окно в Excel; окна используются для отображения рабочих листов, диаграмм и т.д.
Workbook	Открытая рабочая книга
Worksheet	Рабочая таблица в книге

Использование свойств объектов

Свойства объектов можно использовать только двумя способами: *получать* значение свойства или *устанавливать* его. Следует отметить, что не все свойства объекта изменяемы. Свойства объектов, которые нельзя изменять, называют *свойствами, доступными только на чтение (read-only);* свойства, которые можно устанавливать, называют *свойствами, доступными на чтение/запись (read-write).*

Свойства обычно содержат численные, строковые, значения типа Boolean, хотя некоторые свойства могут возвращать значения типа Object или другие типы данных.

Свойства используются в выражениях так же, как любое другое значение переменной или константы. Можно присваивать значение свойства переменной, использовать свойства объектов в выражениях как аргументы к функциям и процедурам или как аргументы для методов какого-либо объекта.

Чтобы присвоить некоторой переменной значение свойства объекта, используйте следующий синтаксис:

Переменная=Обьект.Свойство

Можно также использовать свойство объекта непосредственно в каком-либо выражении или в качестве аргумента функции или процедуры.

Пример использования свойств объекта

AnyStr = aSheet.Name

В этом примере строка, сохраняемая в свойстве **Name** рабочего листа Excel, на которую ссылается объектная переменная **aSheet,** присваивается переменной **AnyStr:**

MsgBox InstBook.FullName

В приведенном примере **InstBook** — это переменная, заданная для ссылки на объект открытой рабочей книги. Если **InstBook** ссылается на рабочую книгу с именем **Sales.xls** в папке **My Documents,** то окно сообщения, вызываемое приведенным выше оператором, отображает строку "C:\My Documents\SALES.XLS".

Чтобы задать свойство объекта, надо присвоить свойству новое значение, используя следующий синтаксис:

ИмяОбъекта.Свойство=Выражение

Пример изменения свойств объекта

InstSheet.Name = "Первый квартал"

В данном примере изменяется имя рабочего листа, на который ссылается объектная переменная **InstSheet,** присваивая значение свойству **Name** листа:

В таблице 2 перечислены некоторые из наиболее употребительных или полезных свойств объектов в Excel. В таблице представлено имя свойства, тип и значение, а также объекты, которые имеют это свойство.

Таблица 2

Свойство	Тип/Что означает	Где найти
ActiveCell	Object: активная ячейка в рабочем листе	Application, Window
ActiveChart	Object: активная диаграмма	Application, Window, Workbook
ActiveSheet	Object: активный лист	Application, Window, Workbook
Address	Возвращает координаты ячейки указанного объекта	Range
Cells	Диапазон объекта Range	Application, Range, Worksheet
Count	Integer: число объектов в коллекции	Все объекты коллекции
Formula	String: формула для ячейки рабочего листа	Диапазон
Name	String: имя объекта	Application, Workbook и в других объектах
Path	String: драйвер и каталог, в	Addln, Application,

	котором сохранен объект	Workbook
Saved	Boolean: сохранялась ли рабочая книга после последних изменений	Workbook
Selection	Object: текущий выделенный фрагмент	Application, Window
StatusBar	String: сообщение в статусной строке	Application
ThisWorkBook	Object: рабочая книга, из которой выполняется текущая процедура	Application
Visible	Boolean: отображается или нет объект на экране	Application, Worksheet, Range и в других объектах
Value	(варьируется): действительное значение, отображаемое в ячейке	Range

Использование методов объекта

Методы объекта используются в операторах VBA так, как использовались бы любые встроенные процедуры VBA.

Метод объекта имеет следующий синтаксис:

ИмяОбъекта.Метод

Для методов объектов, имеющих обязательные и необязательные аргументы, используется такой синтаксис:

ИмяОбъетк.Метод Аргумент1, Аргумент2,…

Во второй строке синтаксиса `Аргумент1, Аргумент2` и так далее представляют аргументы в списке аргументов метода. Необходимо перечислять аргументы метода в определенном порядке, отделяя каждый аргумент в списке запятой и включая *отмечающие запятые {place holding commas)* для пропущенных необязательных аргументов. Метод может иметь один или несколько аргументов в своем списке или не иметь их совсем; аргументы метода могут быть обязательными или необязательными.

Еще один способ вызова метода заключается в использовании *именованных аргументов* и является, пожалуй, наиболее простым и очевидным. При вызове метода указываются названия аргументов, за которыми следует оператор := и сами передаваемые значения:

ИмяОбъекта.Метод Аргумент1:=Значение1, Аргумент2:=Значение2, …

Вызов метода с использованием именованных аргументов имеет два преимущества. Прежде всего, это наглядность кодов программы. Второе преимущество – в его простоте. С этом случае указывать необходимо только те аргументы, значения которых определяются вызывающей программой.

Примеры использования методов объекта

InstBook.Activate

ActiveWorkbook.SaveAs Filename:="D:\VBA2000\NEWFILE.xls", FileFormat:=xlNormal

Многие объекты имеют методы, которые возвращают значения так же, как это делает функция. Чтобы использовать значение, возвращаемое методом, необходимо поместить список аргументов метода в круглые скобки и включить вызов метода в оператор присваивания или другое выражение, точно так же, как при использовании функции. Можно также игнорировать результат, возвращаемый методом. Чтобы игнорировать результат метода (если он имеет результат), вызовите метод без круглых скобок вокруг списка аргументов, как если бы метод не возвращал результата.

Примеры использования возвращаемых значений методов объекта

MsgBox myRange.Address

MsgBox myRange.Address(, , xlRlCl)

Метод `Address` в Excel (который принадлежит объекту `Range`**)** возвращает адрес диапазона ячеек в рабочем листе как строку. Если переменная `myRange` ссылается на первую ячейку в рабочем листе, то оператор MsgBox в приведенной выше строке примера отображает строку $A$1.

Хотя в этом примере метод `Address` вызывается без каких-либо аргументов, он, на самом деле, имеет несколько необязательных аргументов. Эти необязательные аргументы определяют стиль адреса ячеек рабочего листа, возвращаемого методом, а также, являются ли координаты ячеек абсолютными или относительными. Во втором примере показан вызов метода `Address` с его третьим необязательным аргументом (который определяет стиль возвращаемых координат ячеек):

В таблице 3 приведены некоторые из наиболее употребительных или полезных методов в Excel. В таблице представлено имя, краткое описание метода и объекты, имеющие этот метод.

Таблица 3

Метод	Назначение	Имеется в объектах
Activate	Активизирует объект	Window, Workbook, Worksheet, Range и в других объектах
Calculate	Выполняет вычисления в открытой рабочей книге, рабочем листе или диапазоне	Application, Range, Worksheet
Clear	Удаляет данные, сохраненные в указанном объекте	Range
Close	Закрывает указанный объект	Window, Workbook, Workbooks
Save	Сохраняет файл рабочей книги	Application, Workbook
SaveAs	Сохраняет указанный объект в другом файле	Workbook, Worksheet
Select	Выбирает указанный объект	Range, Sheets,

		Worksheets

Многие методы имеют большое количество аргументов. Чтобы получить список параметров метода, можно воспользоваться системой `Auto Quick Info`. Эта система работает следующим образом: как только вы наберете строку, например

```
ActiveSheet.SaveAs(
```

система подсказки выдаст на экран список всех параметров метода `SaveAs`, как это обычно делается для обычных функций.

Согласно выданному системой `Auto Quick Info` всплывающему окну метод `SaveAs` имеет довольно много аргументов, хотя все они — необязательные. Если интуитивно вам непонятно назначение параметров какой-либо функции в окне `Auto Quick Info`, обратитесь за помощью к справочной системе. Для этого либо выделите, например, строку `ActiveDocument.SaveAs` и нажмите клавишу F1.

В справочной системе очень подробно приведено описание каждого метода, и следует почаще прибегать к ее услугам. Впрочем, Редактор VBA постоянно предлагает эту помощь в процессе вашей работы с ним.

Объектные переменные

В дополнение к типам Byte, Integer, Long, Single, Double и String VBA также имеет тип `Object`. Переменные или выражения типа `Object` ссылаются на объект VBA или на объект, принадлежащий приложению, например Excel-объекты `Workbook, Worksheet и Range.`

Как и в случае с другими типами VBA, можно объявлять переменные в модулях, процедурах и функциях с определенным типом `Object`, что показано в следующем операторе:

```
Dim myObject As Object
```

Можно задавать переменную `myObject`, создаваемую предшествующим оператором **Dim,** чтобы она содержала ссылку на любой объект VBA или объект приложения. Если вы собираетесь использовать переменную типа `Object` для некоторых специфических типов объектов, можно также объявлять объектную переменную для этого специфического типа объекта:

```
Dim InstBook As Workbook
```

Объектную переменную `InstBook`, создаваемую этим оператором `Dim`, можно использовать только для сохранения ссылок на объекты `Workbook`; при попытке присвоить переменной `InstBook` ссылку на объект Range или `Worksheet` VBA отображает сообщение об ошибке несовпадения типов. Аналогично, следующее предложение объявляет объектную переменную, которая может сохранять только объекты `Document`:

```
Dim InstDoc As Document
```

Объектное выражение (object expression) — это любое выражение VBA, которое определяет отдельный объект. Все объектные выражения должны вычисляться до единственной объектной ссылки {ссылки на объект);

объектные выражения используются с единственной целью — создание ссылок на специфические объекты в ваших программах VBA.

Объектное выражение может состоять из объектных переменных, объектных ссылок или объектного метода или свойства, которое возвращает объект. Нельзя использовать переменные типа `Object` или объектные выражения в арифметических, логических или операциях сравнения. Объектная ссылка, созданная с помощью объектного выражения или сохраненная в объектной переменной, в действительности, является только адресом, указывающим место в памяти компьютера, где сохранен объект, на который выполняется ссылка. Поскольку объектная ссылка — это адрес памяти, арифметические, логические операторы и операторы сравнения не имеют смысла.

Перед использованием объектной переменной для ссылки на объект необходимо задать эту переменную, чтобы она содержала ссылку на нужный объект. Присваивание объектной ссылки объектной переменной отличается от присваиваний других переменных; чтобы присвоить объектную ссылку объектной переменной, используйте оператор **`Set`.**

Оператор **Set** имеет следующий синтаксис:

Set Переменная=Объект

`Переменная` — это любая объектная переменная или переменная типа `Variant`**.** `Объект` — любая допустимая объектная ссылка; это может быть другая объектная переменная или объектное выражение. Если `Переменная` — переменная, объявленная с каким-либо определенным типом (например, `Range` или `Workbook`**),** этот тип должен быть совместим с объектом, на который ссылается `Объект`.

Пример использования объектных переменных

```
Dim InstSheet As Worksheet
Set InstSheet = Application.ActiveSheet
```

Чтобы задать отдельный объект в выражении или объектную переменную для ссылки на этот объект, используйте методы и свойства, возвращающие объекты, такие как свойства **ActiveWorkbook** и **ActiveSheet** объекта **Application** или метод **Cells** объекта **Worksheet** (в Excel). Аналогичные принципы применимы в Word: используйте свойство **ActiveDocument** объекта **Application** для получения ссылки на текущий документ и так далее.

Ссылка на объекты с помощью With...End With

При написании кода программы можно встретить ссылку на один и тот же объект в нескольких операторах. VBA предоставляет особую структуру — структуру `With...End With`**,** позволяющую ссылаться.на свойства или методы, которые принадлежат одному и тому же объекту, без задания всей объектной ссылки каждый раз.

Структура `With...End With` имеет следующий синтаксис:

```
With  Object
```

```
' операторы, использующие свойства и методы Object
End With
```

`Object` — это любая допустимая объектная ссылка.

Пример использования структуры With...End With.

```
Dim FName As String     'имя файла-копии
With ActiveWorkbook
' Сформировать новое имя файла из исходного
FName - Left(.Name, InStr(.Name, ".") - 1) & "_bp2.xls"
FName = .Path & "\" & FName
.SaveCopyAs Filename:=FName
End With
End Sub
```

Сравните этот же код без использования структуры `With...End With`

```
Dim FName As String       ' имя файла-копии
' Сформировать новое имя файла из исходного
FName = Left(ActiveWorkbook.Name, InStr(ActiveWorkbook.Name, ".") -
1) _
 & "_bp.xls"
FName = ActiveWorkbook.Path & "\" & FName 22
ActiveWorkbook.SaveCopyAs Filename:=FName
End Sub
```

Работа с коллекциями объектов и контейнерами объектов

Коллекция (collection) объектов — это группа связанных объектов, таких как все рабочие листы в рабочей книге или все символы в параграфе. Объект в коллекции называется *элементом (element)* этой коллекции.

Сама коллекция является объектом; коллекции имеют собственные свойства и методы. Каждая коллекция, например, имеет свойство **Count,** которое возвращает число элементов в коллекции. Если в активной рабочей книге имеется 16 рабочих листов, то следующее выражение вычисляется до числа 16:

Application.ActiveWorkbook.Worksheets.Count

В этом выражении `Worksheets` — коллекция всех рабочих листов в рабочей книге, `ActiveWorkbook` — свойство Excel-объекта `Application`**,** возвращающее активную рабочую книгу, а `Count` — свойство коллекции Worksheets, возвращающее общее число рабочих листов в коллекции.

Это простое выражение помогает проиллюстрировать то, что одни объекты содержат другие объекты.. *Контейнер (container)* — это любой объект, содержащий один или несколько других объектов. В данном примере `Application` содержит объект, на который ссылается `ActiveWorkbook`**,** содержащий в свою очередь, коллекцию объектов `Worksheets`**.** Все контейнерные объектные ссылки соединяются вместе с помощью точки-разделителя (.) для образования одного объектного выражения.

Объект Application включает в себя коллекцию (семейство) объектов Книга (Workbook). Семейство книг называется Workbooks (в конец добавляется буква “s” в английском языке это обозначает множественное

число). Обратиться с конкретной книге в приложении можно указав название семейства – Workbooks и в скобках имя или номер книги. Например, Workbooks(“Книга.xls”) или Workbooks(1). Объект Workbook является частью семейства Workbooks, хотя семейство Workbooks тоже является объектом. Кроме того, существует еще объект ActiveWorkbook – активная книга, который тоже является свойством объекта Application. Объекты и ActiveWorkbook являются одиночными объектами. Объект Workbooks - семейство.

Объект Workbook содержит набор свойств объектов, которые будут перечислены ниже:

- объект семейство (коллекция) Worksheets – это рабочие листы конкретной рабочей книги.
- объект семейство Sheets – это рабочие листы в активной рабочей книге.
- объект семейство Windows – это все окна в конкретной рабочей книге.
- объект ActiveSheet - это активный рабочий лист в активной рабочей книге.

Объект Worksheet является рабочим листом - элементом семейства Worksheets, но чтобы обратиться к конкретному листу из семейства надо указать имя или номер рабочего листа в коллекции. Например Worksheets(”Лист1”) или Worksheets(2).

Объект APPLICATION

Объект Application - это главный (корневой) объект в иерархии объектов Excel, представляющий само приложение Excel. Он имеет огромное число свойств и методов, позволяющих установить общие параметры приложения Excel. Кроме того, объект Application через свойство WorksheetFunction предоставляет возможность использовать в коде все встроенные функции рабочего листа. Это семейство возвращает WorksheetFunction объект, являющийся контейнером всех функций рабочего листа. Например, в следующем примере находится максимальное знчение из диапазона A1:A4:

```
Макс=Application.WorksheetFunction. Max(Range("A1:A4"))
```

Функции рабочего листа можно включать в код непосредственно через объект Application, опуская свойство WorksheetFunction. Например, в следующем примере переменной Pi присваивается значение Пи, а переменной Сумма присваивается значение суммы из диапазона A1:A4:

```
Pi = Application.Pi()
Сумма = Application.Sum(Range("A1:A4"))
```

Свойства объекта APPLICATION

Свойство	Описание
ActiveWorkbook, ActiveSheet, ActiveCell, Activechart	Возвращают активную рабочую книгу, лист ячейку, диаграмму. ActiveCell содержится в ActiveSheet, а ActiveSheet и ActiveChart в ActiveWorkbook. В следующем примере в активной ячейке устанавливается

	полужирный шрифт и в нее вводится строка текста "Отчет за май": *With ActiveCell* *.Font Bold = True* *.Value = "Отчет за май"* *End With*
ThisWorkbook	Возвращает рабочую книгу, содержащую выполняющийся в данный момент макрос Может возвращать рабочую книгу отличную от возвращаемой свойством ActiveWorkbook, т. к. выполняемый макрос может находиться в неактивной книге.
Calculation	Устанавливает режим вычислений. Допустимые значения: - xlCalculationAutomatic (автоматически); - xlCalculationManual (вручную); - xlCalculationSemiAutomatic (автоматически, кроме таблиц);
Caption	Возвращает или устанавливает текст из заголовка главного окна Excel. Установка значения свойства равным Empty возвращает заголовок, используемый по умолчанию. В следующем примере первая инструкция устанавливает в качестве заголовка окна приложения текст "Отчет за 2000 год", а вторая возвращает окну имя, используемое по умолчанию, т. е. Excel Application.Caption = "Отчет за 2000 год" Application.Caption = Empty
DisplayScrollBars	Логическое свойство, регулирующее отображение полос прокрутки.
DisplayStatusBar	Логическое свойство, регулирующее отображение строки состояния..
Height и Width	Высота и ширина окна приложения в пунктах.
Left и Top	Расстояние в пунктах от левой границы окна приложения до левого и верхнего края экрана.
Right	Расстояние в пунктах от правой границы окна приложения до правого края экрана.
StatusBar	Возвращает или устанавливает текст, выводимый в строке состояния. В данном примере в строке состояния выводится "Ввод данных..." Application.DisplayStatusBar = True Application.StatusBar = "Ввод данных..."
WindowState	Устанавливает размер окна. Допустимые значения: - xlMaximized (максимальный);

	- xlMinimized, (минимальный); - xlNormal (нормальный).

Методы объекта APPLICATION

Метод	Описание
Calculate	Вызывает принудительное вычисление во всех открытых рабочих книгах, или специфицированном рабочем листе или диапазоне. В следующем примере первая инструкция приводит к перерасчету во всех открытых книгах, вторая инструкция - к перерасчету на указанном рабочем листе активной рабочей книге, а третья - в указанном диапазоне: - Application.Calculate - Worksheets("Отчет").Calculate - Worksheets("Отчет ").Range("A1:C10").Calculate
Help	Отображает справку. Синтаксис: Help(HelpFile, HelpContextID) - HelpFile - имя HLP-файла. Если значение этого параметра не указано, то отображается файл справки Microsoft Excel; - HelpContextID - номер раздела справки. Если значе ние этого параметра не указано, то отображается оглав ление справки. Application.Help "Notepad.hip"
Volatile	Вызывает перевычисление функции пользователя при изменениях в ячейках рабочего листа. Например, Function Квадрат(x) Application.Volatile Квадрат = x^2 End Function
OnKey	Устанавливает выполнение специфицированной процедуры при нажатии заданной комбинации клавиш.

Объект WORKBOOK и семейство WORKBOOKS

В иерархии Excel объект **Workbook** идет сразу после объекта **Application** и представляет файл рабочей книги. Рабочая книга хранится либо в файлах формата XLS (стандартная рабочая книга) или XLA (полностью откомпилированное приложение). Свойства и методы рабочей книги позволяют работать с файлами.

Свойства объекта WORKBOOK и семейства WORKBOOKS

Свойство	Описание

ActiveSheet	Возвращает активный лист книги. В следующем примере устанавливается имя активного рабочего листа: ActiveSheet.Name = "Отчет"
Sheets	Возвращает семейство всех листов книги.
Worksheets	Возвращает семейство всех рабочих листов книги.
Name	Возвращает или устанавливает имя книги.
Path	Возвращает полное имя папки, в которой находится книга.
FullName	Возвращает полное имя книги, включая путь. Например, Имя = ActiveWorkbook. FullNaine.
Saved	Логическое свойство, которое устанавливает, не производились ли изменения в книге со времени ее последнего сохранения.
WriteReserved	Логическое свойство, которое устанавливает, закрыта ли книга для записи.

Методы объекта WORKBOOK и семейства WORKBOOKS

Метод	Описание
Activate	Активизирует рабочую книгу так, что ее первый рабочий лист становится активным. Например, Workbooks("Отдел кадров").Activate
Add	Создает новый объект в семействе Workbooks. Синтаксис: Add (Template) где Template - необязательный. Задает шаблон, на основе которого создается новая рабочая книга. Допустимые значения: - xlWBATChart; - xlWBATExcel4IntlMacroSheet; - xlWBATExcel4MacroSheet; - xlWBATWorksheet. Если аргумент Template опущен, то создается новая рабочая книга с количеством листов, заданных свойством SheetsInNewWorkbook.
Protect	Защищает рабочую книгу от внесения в нее изменений. Синтаксис: - Protect (Password, Structure, Windows) - Password - необязательный. Строка, используемая в качестве пароля для защиты книги. Если параметр опущен, то книга защищена без пароля; - Structure - необязательный. Логический параметр, который устанавливает, защищена ли структура книги, т.

	е. взаимное расположение листов; - Windows - необязательный. Логический параметр, который устанавливает, защищено ли окно книги. В следующем примере устанавливается защита для активной рабочей книги: ActiveWorkbook.Protect Password:= "ВинниПух"
Unprotect	Снятие защиты с рабочей книги. Синтаксис: Unprotect(Password) где Password - необязательный. Строка, используемая в качестве пароля для защиты листа. В следующем примере снимается защита с активной книги: ActiveWorkbook.Unprotect Password:= "ВинниПух"
Close	Закрытие книги.
Open	Открытие существующей книги. Синтаксис: Open(FileName, Readonly, Password, Converter, AddToMRU) - FileName - обязательный. Имя открываемого файла; - Readonly - необязательный. Логический параметр, задающий открытие файла в режиме, доступном только для чтения; - Password - необязательный. Строка с паролем для защищенной книги; - Notify - необязательный. Логический параметр, задающий, надо ли извещать пользователя о том, что файл доступен в режиме только для чтения; - AddToMRU - необязательный. Логический параметр, задающий, надо ли добавить открываемый файл в список недавно использованных файлов. Пример:Workbooks.Open "000 Рога и Копыта"
Save	Сохраняет книгу. Пример: ActiveWorkbook.Save.
SaveAs	Сохраняет книгу в другой файл. Синтаксис: SaveAs (Filename, FileFormat, Password, WriteResPassword, ReadOnlyRecornmended, ConflictResolution, AddToMru, TextCodePage, TextVisualLayout) - D Filename - строка, указывающая имя файла, в который будет сохранена рабочая книга; - FileFormat - необязательный. Задает формат файла;

	- Password - необязательный. Строка с паролем для защищенной книги; - WriteResPassword - необязательный. Строка с паролем для сохранения защищенной книги; - ReadOnlyRecommended - необязательный. Логический параметр, указывающий, надо ли отображать сообщение, что файл доступен только для чтения; - ConflictResolution - необязательный. Задает режим разрешения конфликтов при многопользовательском доступе к файлу; - TextCodePage и TextVisualLayout - необязательные. Используются только в локальных версиях для задания кодовой страницы. Пример: ActiveWorkBook.SaveAs Filename:="Новая жизнь"
SaveAsCopy	Сохранить рабочую книгу в другой файл, оставляя рабочую книгу в памяти с прежним именем. Синтаксис: SaveAs(Filename) где Filename - строка, указывающая имя файла, в который будет сохранена рабочая книга. В следующем примере активная рабочая книга сохраняется в файл с именем "ЗапаснаяВерсия": ActiveWorkbook.SaveCopyAs "C:\ЗапаснаяВерсия".

Объект Worksheet и семейство Worksheets

В иерархии Excel объект worksheet идет сразу после объекта workbook и представляет рабочий лист.

Приведем несколько наиболее часто используемых свойств и методов объекта Worksheet.

Свойства объекта WORKSHEET и семейства WORKSHEETS

Свойства	Описание
Name	Возвращает имя рабочего листа: Worcsheets(1).Name=”Итоги”
Visible	True (False) – рабочий лист видим (невидим) на экране.
Range	Возвращает ссылку на указанный диапазон ячеек. Например: ActiveSheet.Range("B1")
UsedRange	Возвращает диапазон ячеек рабочего листа.
ActiveCell	Возвращает активную ячейку рабочего листа.

Методы объекта WORKSHEET и семействаWORKSHEETS

Методы	Описание

Activate	Активизирует рабочий лист: Worksheet(2).Activate
Add	Создает новый рабочий лист. Параметры: Before – лист, перед которым будет размещен новый лист; After – лист после которого будет помещен новый лист; Count – число добавляемых листов; Type – тип добавляемого листа. Например, ActiveWorkbook.Worksheets.Add
Delete	Удаляет рабочий лист: Worksheets(1).Delete
Evaluate	Преобразует текстовую строку в объект Excel или значение. Используется, например, для ввода ссылок на ячейки: MyCell = InputBox("Введите имя ячейки") Evaluate(myCell).Value = "Новое значение"
Copy	Копирование активного рабочего листа в другое место рабочей книги: Worksheets(“Лист2”). _ Copy After:= Worksheets(“Лист3”)
Move	Перемещение активного рабочего листа в другое место рабочей книги: Worksheets(“Лист2”). _ Move After:= Worksheets(“Лист3”)

Задание 1

Написать макрос, который открывает книгу “Сотрудники”, создает новую книгу “Копия” и копирует из книги “Сотрудники” лист “Штат” после первого листа книги “Копия”.

Предполагается, что книга “Сотрудники” находится в той же папке, что и файл с книгой, содержащей макрос.

```
Sub открыть_книгу()
Dim n As Integer, i As Integer
m = 0
n = Workbooks.Count 'определение количества открытых книг
'Проверка, открыты ли книга Сотрудники
For i = 1 To n
If Workbooks(i).Name = "Сотрудники" Then
m = i
End If
Next
'Отрытие книги Сотрудники, если она еще не открыта
If m = 0 Then
Workbooks.Open "Сотрудники" 'Книга должна находится в текущем каталоге
End If
Workbooks.Add 'Создание новой книги
n = Workbooks.Count
Workbooks(n).SaveAs "Копия" 'Сохранение новой книги
'Копирование листа Штат
Workbooks(n - 1).Sheets("Штат").Copy Before:=Workbooks(n).Sheets(2)
End Sub
```

Задание 2

Переставить листы в обратном порядке.

```
Sub переставить_листы()
Dim k As Integer, i As Integer
k = Sheets.Count
For i = 1 To k - 1
Sheets(k).Move before:=Sheets(i)
Next
End Sub
```

Обработчики событий

Событие — это действие, распознаваемое объектом, для которого можно запрограммировать отклик.

Например, в качестве события можно использовать открытие или закрытие документа, щелчок мыши, нажатие клавиши.

Набор действий или повторяющихся явлений, которые можно сопоставить с кодом VBA, называется *событиями*, а специальный тип процедуры, которая выполняется при возникновении события, называется *обработчиком событий*.

Обработать можно события следующих обьектов Excel:

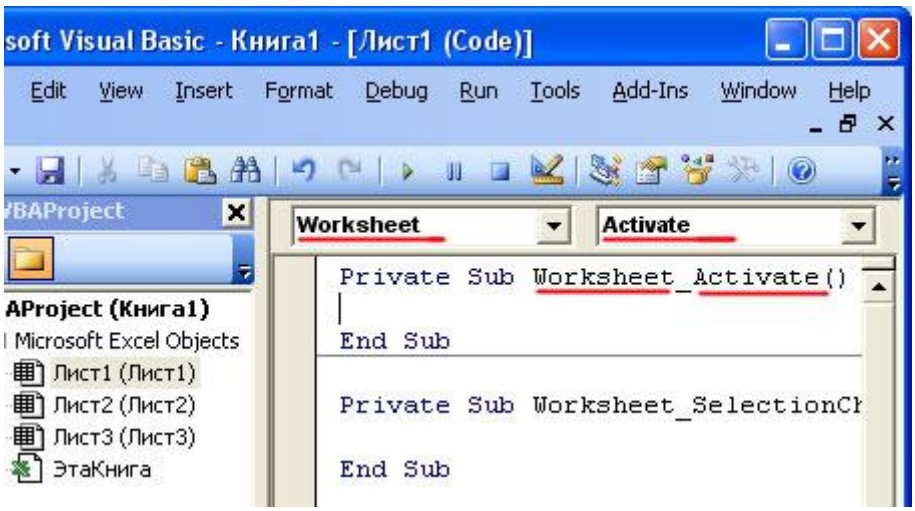
- Application
- WorkBook
- WorkSheet
- Chart

Обработчики событий дают возможность привязать свой код к действиям пользователя, например к открытию или закрытию книги, активации таблицы, сохранению документа ... Обработчики событий создаются с модулях лисов или книги (в зависимости от того, с каким объектом будет связано это событие. Чтобы создать процедуру обработки события, откройте редактор Visual Basic (Alt + F11), выберите например Лист1 и из двух раскрывающихся списков сверху выберите объект и событие. Редактор автоматически создаст процедуру для обработки события. Вам остается только написать в ней свой код (см. рисунок).

Однако в некоторых ситуациях события для объектов не появляются в окне редактора кода (например, это справедливо для очень важного объекта Application). В этом случае необходимо явно объявить этот объект с событиями — при помощи ключевого слова WithEvents, например так:

```
Public WithEvents App As Word.Application
```

Делается это в области объявлений модуля (Declarations). После этого в редакторе кода Visual Basic появляется новый объект App со всеми необходимыми событиями.



Многие события имеют параметры. Это выглядит вот так:

```
Private Sub Worksheet_BeforeRightClick(ByVal Target As Excel.Range, Cancel
As                                                                  Boolean)
......
End Sub
```

Здесь Target - диапазон ячеек, подвергшийся процедуре правого клика, Cancel - параметр, позволяющий отменить событие, если установить его в True.

События объекта Application

Событие	Описание
NewWorkbook	При создании новой рабочей книги
WorkbookActivate	При активизации рабочей книги
WorkbookBeforeClose	Перед закрытием рабочей книги
WorkbookBeforeSave	Перед сохранением рабочей книги
WorkbookDeactivate	Когда активная книга теряет фокус
WorkbookNewSheet	При добавлении нового листа в рабочую книгу
WorkbookOpen	При открытии рабочей книги

События объекта Workbook

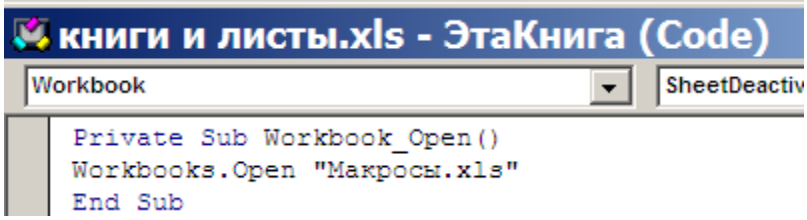
Событие	Описание
BeforeClose	При закрытии рабочей книги
BeforeSave	Перед сохранением рабочей книги
Deactivate	Когда рабочая книга теряет фокус
NewSheet	При добавлении нового листа
Open	При открытии рабочей книги
SheetActivate	При активизации любого рабочего листа
SheetDeactivate	Когда рабочий лист теряет фокус

События объекта Worksheet

Событие	Описание
BeforeClose	При закрытии рабочей книги
BeforePrint	Перед печатью рабочей книги
BeforeSave	Перед сохранением рабочей книги
Deactivate	Когда рабочая книга теряет фокус
NewSheet	При добавлении нового листа
Open	При открытии рабочей книги
SheetActivate	При активизации любого рабочего листа
Sheet Deactivate	Когда рабочий лист теряет фокус

Задание 3

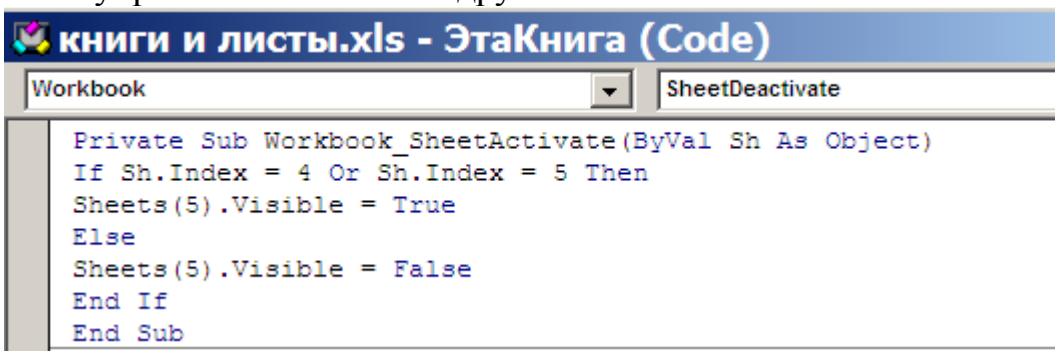
При открытии книги открывать файл Макросы.xls.



Задание 4

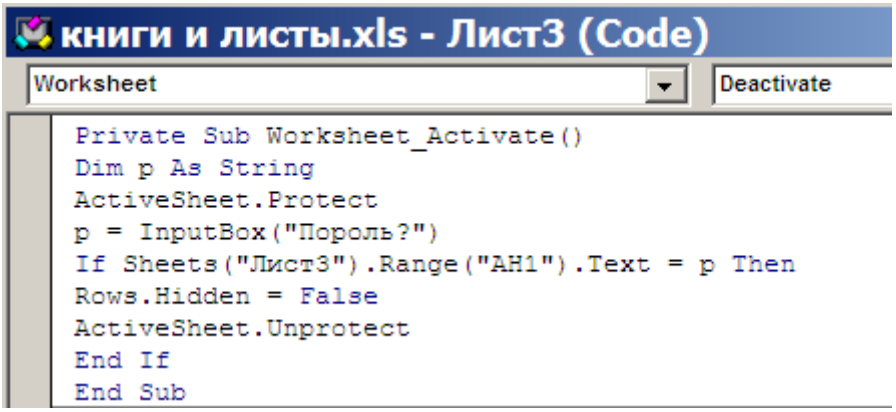
При активации “Лист4” и “Лист5” сделать видимым “Лист5”. При активации других листов “Лист5” становится невидимым.

Для этой задачи надо выбрать событие активации листа но для книги, а не для Листа 4 и Листа 5, чтобы можно было анализировать состояние одних листов и управлять свойствами других листов.



Задание 5

При активации “Листа 3” запрашивать пароль. Если пароль правильный, строки отображаются и с листа снимается защита. Если пароль не совпадает со значением в ячейке AH1 этого листа, оставить скрытыми все строки и защищенным лист.



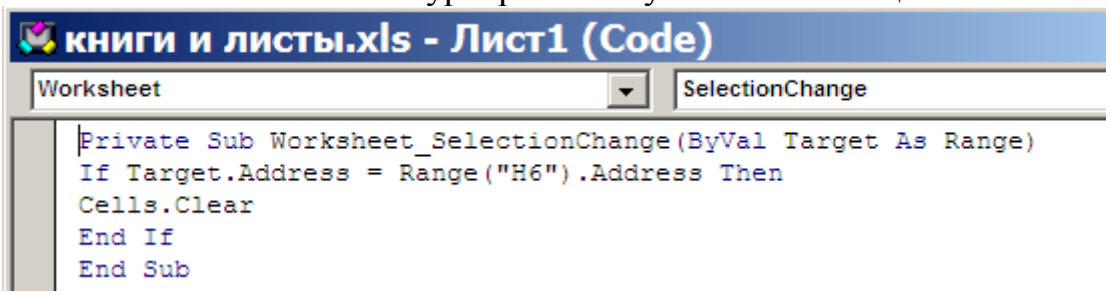
Проверить корректность работы программы и исправить ошибки.

После работы с листом надо привести Лист 3 в первоначальное состояние, т.е. скрыть строки и защитить лист. Для этого надо использовать событие деактивации листа.

```
Private Sub Worksheet_Deactivate()
ActiveSheet.Unprotect
If Rows.Hidden = False Then
Rows.Hidden = True
End If
ActiveSheet.Protect
End Sub
```

Задание 6

Если на листе поместить курсор в ячейку H6 Лист очищается.



Задания для самостоятельной работы

Вариант № 1

1. Создать книгу под именем Данные.xls. В этой книге создать таблицу с 2 столбцами – ФИО, Должность и 4 строками. Написать макрос, который открывает эту книгу и после последней строки дописывает Вашу фамилию.
2. Создать обработчик, в котором при активизации листа, лист следующий за ним скрывается, а при уходе с этого листа, снова открывается.

Вариант № 2

1. Написать макрос, который создает новую книгу. Запрашивает имя этой книги и дает такое же имя первому листу этой книги.
2. Создать обработчик в котором при открытии книги все листы сортируются в обратном порядке.

Вариант № 3

1. Написать макрос, который создает новую книгу под именем Итоги.xls и изменяте названия ее листов на 1, 2 и 3.
2. Создать обработчик, в котором при изменении данный в ячейке F3 этот лист перемещается в конец.

Вариант № 4

1. Создать книгу под именем Пример.xls. В этой книге создать таблицу с 2 столбцами – Должность и Оклад , заполнить в этой таблице 3 строки. Написать макрос, который открывает эту книгу и копирует лист с этой таблицей в книгу, содержащую данный макрос.
2. Создать обработчик, в котором при изменении данных в последней заполненной ячейке столбца В эта ячейка окрашивается черным цветом. После выхода из этой ячейки возвращается в прежнее состояние.

Вариант № 5

1. Написать макрос, который создает новую книгу. Запрашивает имя этой книги , а затем добавляет в конец этой книги еще 3 листа.
2. Создать обработчик, в котором при перемещении на лист 4 создается новый лист и в ячейке А1 нового листа появляется слово «Привет».

Вариант № 6

1. Написать макрос, который создает новую книгу под именем Листы.xls и удаляет 1 и 3 листы этой книги и добавляет в начало новый лист.
2. Создать обработчик, в котором при создании нового листа этот новый лист становится первым.

Вариант № 7

1. Создать книгу под именем Проба.xls. В этой книге создать таблицу с 2 столбцами – Должность и Оклад , заполнить в этой таблице 3 строки. Написать макрос, который открывает эту книгу и копирует лист с этой таблицей в новую книгу.
2. Создать обработчик, в котором при создании нового листа этот лист получает в названии индекс этого листа.

Вариант № 8

1. Создать книгу под именем Данные.xls. В этой книге создать таблицу с 2 столбцами – ФИО, Должность и 4 строками. Написать макрос, который открывает эту книгу и после последней строки дописывает Вашу фамилию.
2. Создать обработчик, в котором при открытии книги первый и второй лист скрываются, а при активизации Листа 3 они снова отображаются.

Вариант № 9

1. Написать макрос, который создает новую книгу. Запрашивает имя этой книги и дает такое же имя первому листу этой книги.
2. Создать обработчик, в котором на Листе 3при изменении данных ячейке A1 весь лист становится черным, при уходе с Листа 3 все возвращается в первоначальное состояние.

Вариант № 10

1. Написать макрос, который создает новую книгу под именем Итоги.xls и изменяет названия ее листов на 1, 2 и 3.
2. Создать обработчик, для Листа 2в котором при перемещении в любую ячейку столбца D этот столбец скрывается. При уходе с листа столбец открывается

Вариант № 11

1. Создать книгу под именем Пример.xls. В этой книге создать таблицу с 2 столбцами – Должность и Оклад , заполнить в этой таблице 3 строки. Написать макрос, который открывает эту книгу и копирует лист с этой таблицей в книгу, содержащую данный макрос.
2. Создать обработчик, в котором при активации листа ширина столбцов с A до H становится равным 1 символ. При уходе с листа ширина этих столбцов становится равной 8 символов.

Вариант № 12

1. Написать макрос, который создает новую книгу. Запрашивает имя этой книги, а затем добавляет в конец этой книги еще 3 листа.
2. Создать обработчик, в котором при изменении данных в 10 ряду. Шрифт во всем ряду становится белого цвета. При ухода курсора из этого ряда шрифт становится черным.

Лабораторная работа № 11
Пользовательские формы

Практически во всех приложениях Office используются пользовательские диалоговые окна. Диалоговые окна в VBA называются формами (объект UserForms). Каждому объекту UserForm присущи определенные свойства, методы и события, которые он наследует от класса объектов UserForms. Диалоговые окна (формы) и элементы управления составляют основу современного визуального интерфейса. Все элементы управления и технология работы с ними в основном стандартизованы и похожи для разных платформ и программных сред. Эти объекты помещены в специальную библиотеку MSForms.

Выделим основные моменты, которые следует иметь в виду при создании визуального интерфейса.

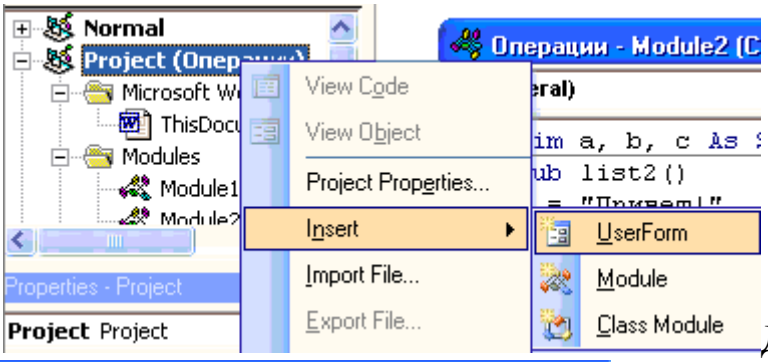
- Все загруженные диалоговые окна представляют коллекцию UserForms со стандартными методами и свойствами. Элемент коллекции – объект класса UserForm – задает отдельное окно.
- Для каждого типа элементов управления в библиотеке msforms имеется класс объектов, имя которого совпадает с именем элемента управления (его типа). Например, есть классы SpinButton и TextBox.
- Диалоговые окна создаются, как правило, не программно, а визуально. Вначале создается само окно, а затем оно наполняется элементами управления при помощи соответствующей панели элементов. Этот этап называется этапом проектирования, и его следует отличать от этапа выполнения, когда приложение выполняется и конечный пользователь взаимодействует с приложением, в частности через диалоговые окна и их элементы управления. Как только создается диалоговое окно и помещается в него тот или иной элемент управления, в этот же самый момент автоматически в программе появляется объект соответствующего класса, с которым можно работать, вызывая его методы и изменяя его свойства.

На этапе проектирования, используя окно свойств, можно задать большинство свойств как самого диалогового окна, так и всех элементов управления, помещенных в него, кроме этого, программно необходимо прописать все обработчики событий.

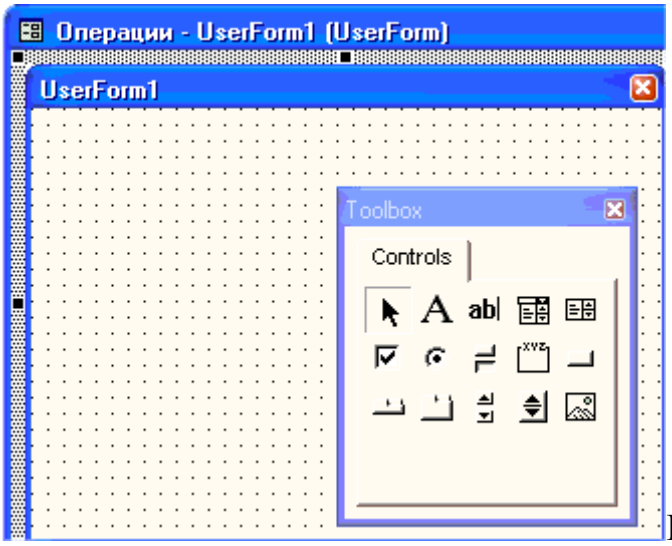
- Последний момент – отладка. Для ведения отладки нужно предварительно откомпилировать приложение и затем перейти в режим выполнения приложения.

Для того чтобы в разрабатываемое приложение можно было добавить форму, необходимо выполнить следующие действия:

- запустить редактор VBA;
- выделить правой кнопкой мыши объект Project, выполнить команду Insert + UserForm, после чего появляются новая форма и панель элементов Toolbox.



Добавление формы



Новая форма

Форма как объект имеет некоторые встроенные свойства, и их можно устанавливать или программным образом, или в Properties Window (окне свойств) редактора VBA (таблица. Наиболее часто используемые свойства объектов UserForm).

Наиболее часто используемые свойства объектов UserForm

Свойство	Описание
ActiveControl	Возвращает объектную ссылку на элемент управления, находящийся в фокусе в данный момент. Только для чтения
BackColor	Целое типа Long определяет цвет фона формы
Caption	Текст, выводимый в качестве заголовка формы
Controls	Возвращает коллекцию всех элементов управления формы
Cycle	Определяет, должно ли нажатие клавиши табуляции вызывать последовательный выбор всех элементов управления во всех группах и на каждой странице многостраничных элементов управления или только в пределах текущей группы или страницы. Может содержать одну из двух встроенных констант: fmCycleAllForms или fmCycleCurrentForm
Enabled	Содержит значение типа Boolean, указывающее, доступна ли форма. Если его значение равно False, ни один из элементов управления формы не доступен

Font	Возвращает ссылку на объект Font, посредством которого можно выбрать параметры шрифта формы или элемента управления
ForeColor	То же самое, что и свойство BackColor, но устанавливает цвет, используемый для переднего плана (обычно это цвет текста) объекта формы

Методы объекта UserForm

Всякий раз, создавая в проекте новый объект UserForm, одновременно создается новый подкласс объекта UserForm. Любые процедуры или функции, написанные в разделе General (общий) модуля класса, относящегося к форме, становятся дополнительными методами для отдельного подкласса объекта (таблица. Наиболее часто используемые методы для объектов UserForm).

Наиболее часто используемые методы для объектов UserForm

Метод	Назначение
Copy	Копирует выделенный в элементе управления текст в буфер обмена Windows
Cut	Вырезает выделенный в элементе управления текст и помещает его в буфер обмена Windows
Hide	Скрывает UserForm, не выгружая ее из памяти, сохраняя значения элементов управления формы и всех переменных, объявленных в модуле класса формы
Paste	Вставляет содержимое буфера обмена Windows в текущий элемент управления
PrintForm	Выводит на используемый в Windows по умолчанию принтер изображение формы, включая все данные, введенные в элементы управления
Repaint	Перерисовывает форму, выведенную на экран. Используется этот метод, если необходимо перерисовать форму, не ожидая, когда она будет перерисована через обычный период времени
Show	Выводит форму на экран. Если форма еще не загружена в память, то данный метод сначала ее загружает. Синтаксис метода Show:FormName.Show

События объекта UserForm

Событие - это что-то, что может произойти с диалоговым окном или элементом управления диалогового окна (таблица. События объектов UserForm).

Событийные процедуры следует записывать в модуль класса, который является частью User Form. При этом такие процедуры должны иметь имена в виде

ObjectName_EventName,

где ObjectName - имя формы или элемента управления, а EventName - имя события, с которым идет работа. Такой формат имени позволяет VBA сопоставлять заданному событию требуемую процедуру.

События объектов UserForm

Событие	Синтаксис заголовка процедуры	Описание
Activate	Private Sub object_Activate()	Инициируется всякий раз, когда окно формы становится активным. Используйте это событие для обновления содержимого диалоговых элементов управления, чтобы отразить любые изменения, которые произошли, пока окно было неактивным
Click	Private Sub object_Click()	Инициируется всякий раз, когда по форме (любой ее части, не занятой элементами управления) щелкают мышью
DblClick	Private Sub object_DblClick()	Инициируется всякий раз, когда по форме (любой ее части, не занятой элементами управления) дважды щелкают мышью
Deactivate	Private Sub object_Deactivate()	Инициируется всякий раз, когда форма перестает быть активной
Initialize	Private Sub object_Initialize()	Инициируется всякий раз, когда форма впервые загружается в память посредством выполнения оператора Load или с помощью метода Show. Используйте это событие для инициализации элементов управления формы при ее появлении на экране
Resize	Private Sub object_Resize()	Инициируется при изменении размеров формы
Terminate	Private Sub object_ Terminate()	Инициируется всякий раз, когда форма выгружается из памяти. Используйте это событие для осуществления любых специальных служебных задач, которые необходимо выполнить прежде, чем переменные формы будут выгружены

Элементы управления

Объект UserForm может содержать те же элементы управления, что и находящиеся в диалоговых окнах Word, Excel или других приложений Windows (таблица. Стандартные элементы управления, включенные в VBA). Элементы управления - это элементы диалогового окна, позволяющие пользователю взаимодействовать с программой. Они включают в себя кнопки-переключатели, текстовые поля, линейки прокрутки, командные кнопки и так далее

Стандартные элементы управления, включенные в VBA

Элемент управления	Назначение
Label (надпись, метка)	Позволяет создавать заголовки элементов управления, которые не имеют собственных встроенных заголовков
TextBox (текстовое поле)	Окно редактируемого текста свободной формы для ввода данных. Может быть одно- или многострочным
ComboBox (поле со списком)	Этот элемент управления объединяет окно редактирования и окно списка. Используйте, когда хотите предложить пользователю выбрать значение, но при этом дать ему возможность ввести данные, отсутствующие в списке
ListBox (список)	Отображает список значений, из которых пользователь может сделать выбор. Окна списка можно использовать, чтобы дать возможность пользователю выбрать только одно значение или же несколько
CheckBox (флажок)	Стандартный флажок (квадратное окно, содержащее, если элемент выбран, галочку). Используйте флажки для выбора вариантов, которые не являются взаимоисключающими
OptionButton (переключатель)	Стандартная кнопка-переключатель (круглое окно, при выборе в центре него находится черная точка). Используйте OptionButton, когда пользователю необходимо сделать выбор между положениями «включено/выключено» или «истина/ложь». Кнопки-переключатели, как правило, объединяются вместе при помощи рамки для создания группы переключателей
ToggleButton (выключатель)	Выключатели служат для той же цели, что и флажки, но выводят установки в виде кнопки, находящейся в «нажатом» или «отжатом» состоянии
Frame (рамка)	Визуально и логически объединяет некоторые элементы управления (особенно флажки, переключатели и выключатели)

CommandButton (кнопка)	Используйте кнопки для выполнения таких действий, как Cancel (Отмена), Save (Сохранить), ОК и так далее Когда пользователь щелкает по кнопке, выполняется VBA-процедура, закрепленная за данным элементом управления
TabStrip (набор вкладок)	Этот элемент управления состоит из области, в которую вы помещаете другие элементы управления (такие как текстовые поля, флажки и так далее) и полосы кнопок табуляции. Используйте элемент управления TabStrip для создания диалоговых вкладок, отображающих одни и те же данные в различных категориях
MultiPage (набор страниц)	Этот элемент управления состоит из нескольких страниц. Вы можете выбрать любую из них, щелкнув по соответствующей вкладке. Используйте элемент управления MultiPage для создания диалоговых окон с вкладками
ScrollBar (полоса прокрутки) и SpinButton (счетчик)	Элемент управления ScrollBar позволяет выбирать линейное значение аналогично тому, как это можно сделать при помощи счетчика. Элемент управления SpinButton является специальной разновидностью текстового поля
Image (рисунок)	Элемент управления Image позволяет вывести на форме графическое изображение. Используйте Image для вывода графических изображений в любом из следующих форматов: *.bmp, *.cur, *.gif, *.ico, *.jpg или *.wmf

Обращение к элементам управления происходит в основном через их свойства и с помощью процедур обработки событий, написанных для каждого элемента (таблица. Свойства стандартных элементов управления).

Свойства стандартных элементов управления

Свойство	Где применяется	Описание
Accelerator	CheckBox, Tab, CommandButton, Label, Page, OptionButton, ToggleButton	Содержит символ, используемый в качестве быстрой клавиши вызова, элемента управления, при нажатии Alt + <клавиша быстрого вызова> происходит выбор элемента управления
BackColor	Все элементы	Число, представляющее определенный цвет фона элемента управления
Caption	CheckBox, CommandButton,	Для надписи - текст, отображаемый элементом управления. Для других

	Frame, Label, OptionButton, ToggleButton, Page, Tab, UserForm	элементов управления - надпись, которая появляется на кнопке или вкладке или рядом с рамкой, флажком или переключателем
Cancel	CommandButton	Задает кнопку отмены диалогового окна. При нажатии на эту кнопку или клавишу Esc диалоговое окно исчезает. Только одна кнопка формы может иметь данное свойство
ControlTip-Text	Все элементы управления	Устанавливает текст, который отображается в виде всплывающей подсказки (ControlTip, называемой также ToolTip), когда указатель мыши помещается на элемент управления
Default	CommandButton	Определяет заданную по умолчанию кнопку. Когда пользователь нажимает в процессе диалога клавишу Enter, эта кнопка ведет себя так, как если бы по ней щелкнули мышью
Enabled	Все элементы управления	Хранит значение типа Boolean, определяющее, доступен или нет элемент управления. Если Enabled имеет значение False, то элемент управления продолжает отображаться в диалоговом окне, но не может быть выбран
ForeColor	Все элементы управления	То же самое, что и BackColor, но устанавливает цвет для переднего плана элемента управления, как правило, символов текста
List	ComboBox	Массив типа variant (одно- или многомерный), представляет список, содержащийся в элементе управления
Max	ScrollBar, SpinButton	Переменная типа Long, определяющая максимальное значение счетчика, или значение, при котором полоса прокрутки находится в самом верху (для вертикальной полосы) или справа (для горизонтальной)
Min	ScrollBar, SpinButton	Переменная типа Long, определяющая минимальное значение счетчика, или значение, при котором полоса прокрутки находится в самом низу (для

		вертикальной полосы) или слева (для горизонтальной)
Name	Все элементы управления	Содержит имя элемента управления. Вы можете установить данное свойство только с помощью Properties Window
RowSource	ComboBox	Задает источник, из которого ListBox берет список объекта. В Excel VBA RowSource обычно использует диапазон рабочего листа
Selected	ListBox	Возвращает массив значений типа Boolean для списка, который допускает множественный выбор. Каждый элемент массива содержит по одному элементу, соответствующему каждому пункту списка. Если значение элемента в массиве selected равно True, то соответствующий пункт списка выбран
TabIndex	Все элементы управления	Число, указывающее положение элемента управления в порядке табуляции (может иметь значение от 0 до значения, равного количеству элементов управления на форме)
TabStop	Все элементы управления	Значение типа Boolean, указывающее, может ли элемент управления быть выбран клавишей Tab. Если значение TabStop равно False, вы тем не менее можете щелкнуть на элементе и таким образом его выбрать
Value	Все элементы управления	Значение текущих установок элемента управления: текст в текстовом поле, какие выбраны флажки и переключатели, индекс выбранного раздела списка или число, указывающее текущее положение полосы прокрутки или счетчика
Visible	Все элементы управления	Значение типа Boolean, указывающее, является ли элемент управления видимым

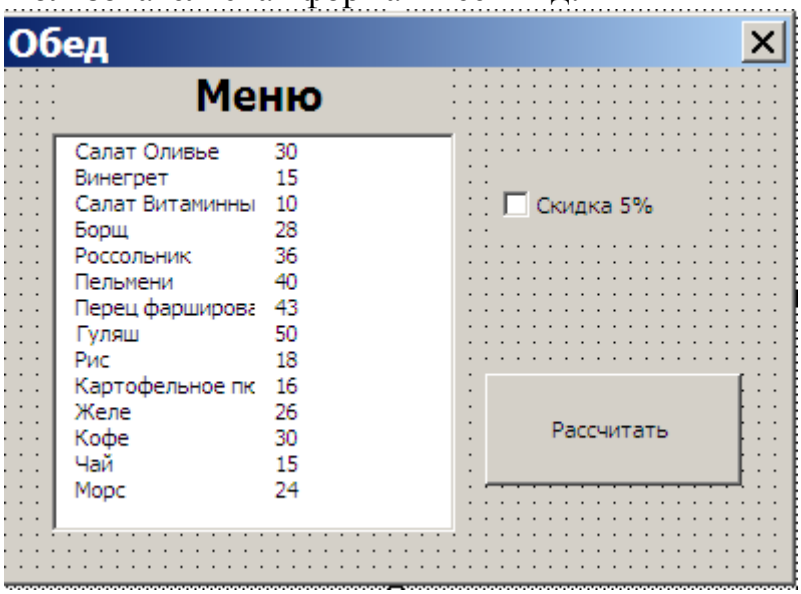
Задание 1

Создать приложение Меню для выбора блюд из списка и вывода результата выбора на лист Excel. Можно одновременно выбрать несколько

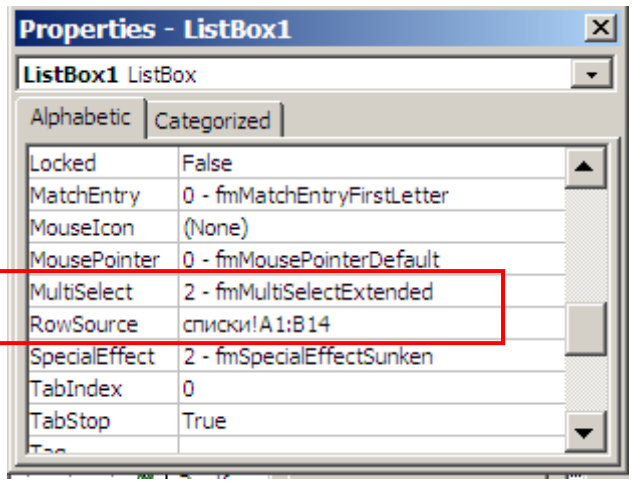
блюд. Предусмотрено наличие скидки 5%. Данные о блюдах (название блюда и его цена) находятся на листе Excel с именем «списки».

	A	B	C
1	Салат Оливье	30	
2	Винегрет	15	
3	Салат Витаминный	10	
4	Борщ	28	
5	Россольник	36	
6	Пельмени	40	
7	Перец фаршированный	43	
8	Гуляш	50	
9	Рис	18	
10	Картофельное пюре	16	
11	Желе	26	
12	Кофе	30	
13	Чай	15	
14	Морс	24	
15			

Пользовательская форма имеет вид:

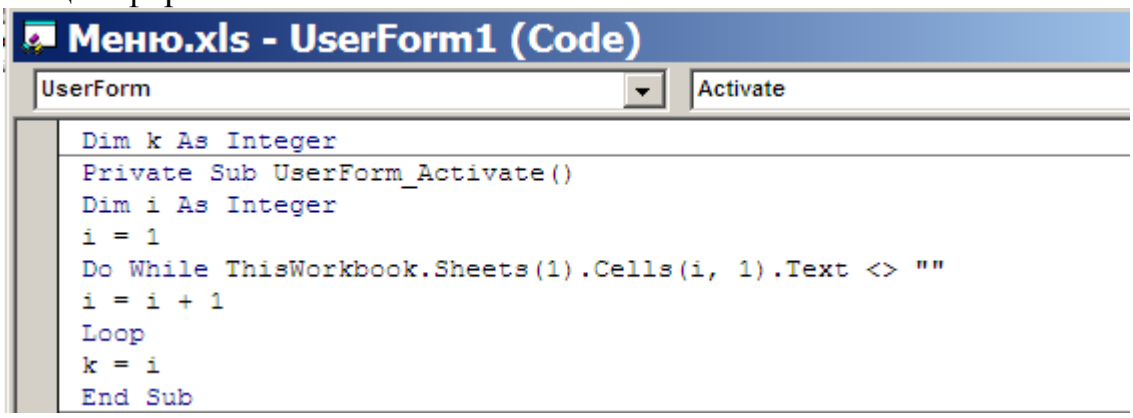


Свойства элемента ListBox1:



На Лист1 записываются результаты выбора. Но при каждом сеансе данные должны заноситься в конец списка. Поэтому каждый раз необходимо

определять последнюю заполненную строчку на листе. Это делается при активации формы.




```
Dim k As Integer
Private Sub UserForm_Activate()
Dim i As Integer
i = 1
Do While ThisWorkbook.Sheets(1).Cells(i, 1).Text <> ""
i = i + 1
Loop
k = i
End Sub
```

Номер первой пустой строки записывается в переменную k. Так как переменная k будет использоваться в других обработчиках событий, эта переменная описывается в первой строчке модуля формы.

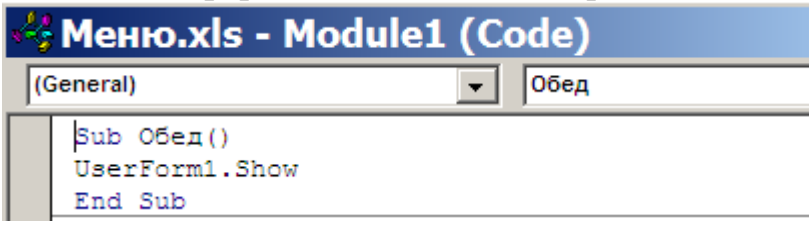
При щелчке по кнопке Рассчитать будут выполняться следующие действия:

```
Private Sub CommandButton1_Click()
Dim s As Integer, i As Integer
Dim itog As Currency, skidka As Single
s = 0
skidka = 0
'Суммирование цены в выделенных строках
For i = 0 To ListBox1.ListCount - 1
If ListBox1.Selected(i) = True Then
s = s + Val(ListBox1.List(i, 1))
'вывод цены и названия выделенных блюд в k строку
ThisWorkbook.Sheets(1).Cells(k, 1) = ListBox1.List(i, 0)
ThisWorkbook.Sheets(1).Cells(k, 2) = ListBox1.List(i, 1)
k = k + 1
End If
Next
'Вывод суммы на Лист1
ThisWorkbook.Sheets(1).Cells(k, 1) = "Сумма"
ThisWorkbook.Sheets(1).Cells(k, 2) = s
k = k + 1
'Вычисление скидки
If CheckBox1.Value = True Then
skidka = s * 0.05
End If
'Вывод итоговых результатос и форматирование чисел
ThisWorkbook.Sheets(1).Activate
Cells(k, 1) = "Скидка"
Cells(k, 2) = skidka
Cells(k, 2).Style = "Currency"
k = k + 1
Cells(k, 1) = "Итого"
Cells(k, 2) = s - skidka
Cells(k, 2).Style = "Currency"
```

После вывода очередного сеанса надо отделить результаты от новых данных.

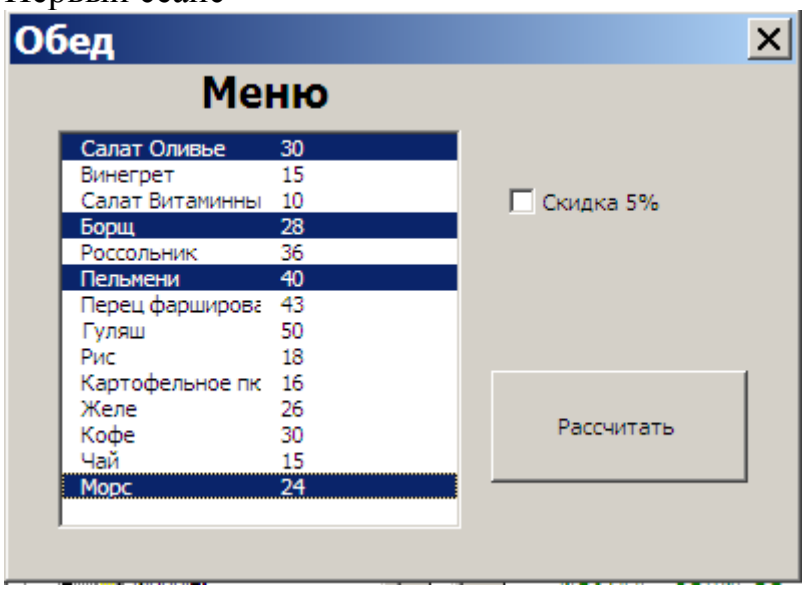
```
'Вывод итоговых результатос и форматирование чисел
ThisWorkbook.Sheets(1).Activate
Cells(k, 1) = "Скидка"
Cells(k, 2) = skidka
Cells(k, 2).Style = "Currency"
k = k + 1
Cells(k, 1) = "Итого"
Cells(k, 2) = s - skidka
Cells(k, 2).Style = "Currency"
'отделение чертой результата от следующего
Range(Cells(k, 1), Cells(k, 2)).Select
Selection.Borders(xlDiagonalDown).LineStyle = xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    Selection.Borders(xlEdgeLeft).LineStyle = xlNone
    Selection.Borders(xlEdgeTop).LineStyle = xlNone
    With Selection.Borders(xlEdgeBottom)
        .LineStyle = xlContinuous
        .Weight = xlThick
        .ColorIndex = xlAutomatic
    End With
    Selection.Borders(xlEdgeRight).LineStyle = xlNone
    Selection.Borders(xlInsideVertical).LineStyle = xlNone
k = k + 1
End Sub
```

Для вывода формы надо создать макрос

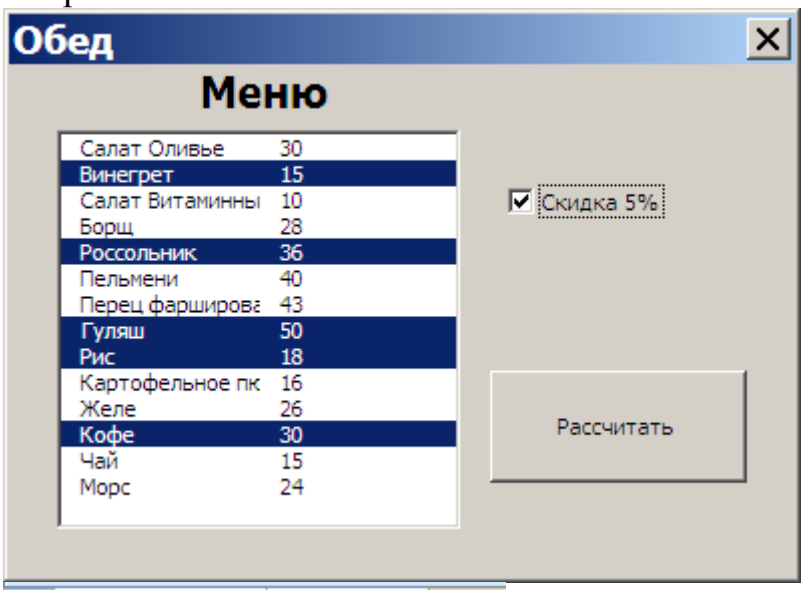


Результаты будут иметь вид:

Первый сеанс



Второй сеанс



	A	B	C
1	Блюдо	Цена	
2	Салат Оливье	30	
3	Борщ	28	
4	Пельмени	40	
5	Морс	24	
6	Сумма	122	
7	Скидка	- р.	
8	Итого	122,00р.	
9	Винегрет	15	
10	Россольник	36	
11	Гуляш	50	
12	Рис	18	
13	Кофе	30	
14	Сумма	149	
15	Скидка	7,45р.	
16	Итого	141,55р.	
17			

В приведенном задании список listbox заполняется с помощью свойства RowSource и разрешается выбрать одновременно несколько строк с помощью клавиши Ctrl.

В следующем задании можно выбрать из списка только один элемент. Список заполняется во время работы программы

Задание 2

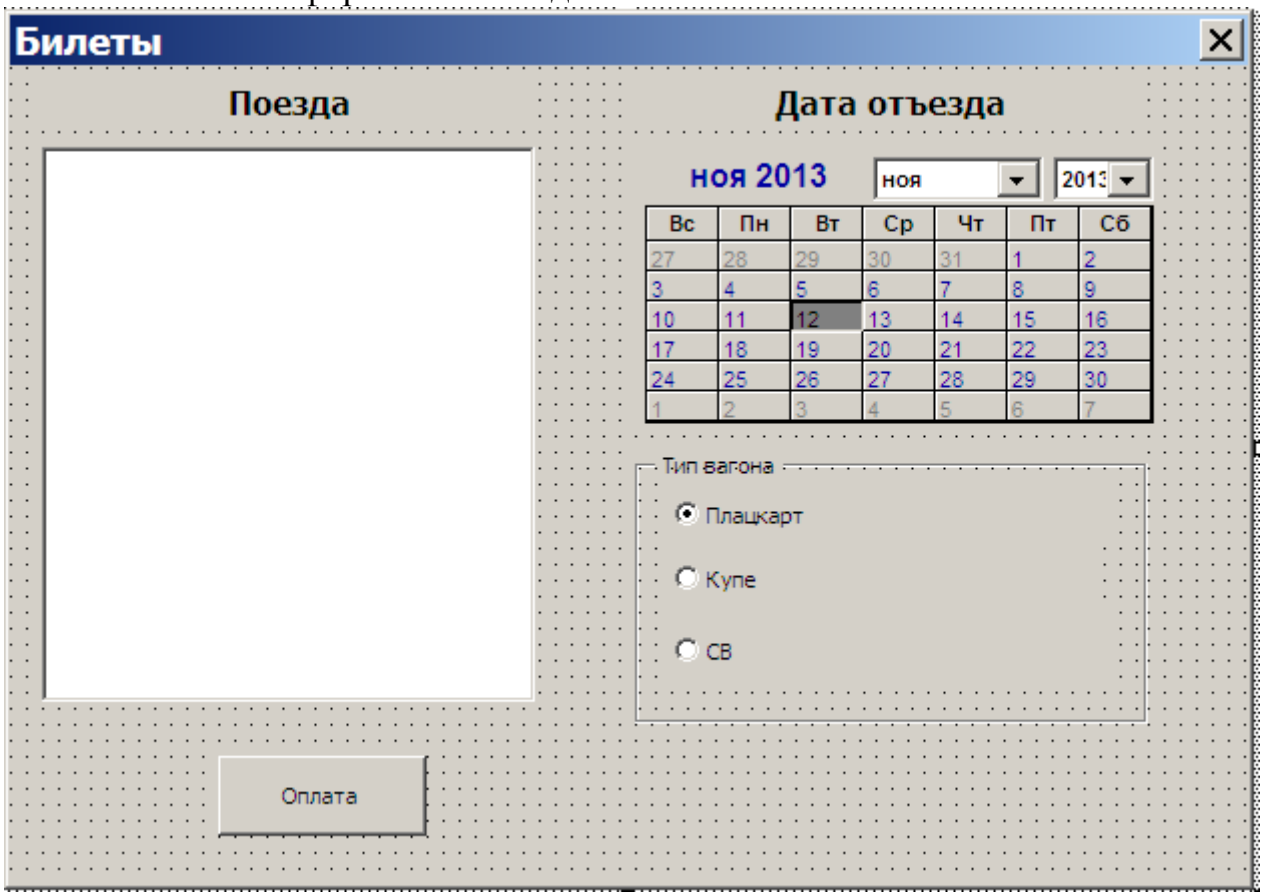
Создать приложение рассчитывающее стоимость железнодорожного билета в зависимости от направления вида вагона и сезона. Летом стоимость увеличивается на 20%, зимой - уменьшается на 10%.

Данные о стоимости билетов по-прежнему располагаются на листе Excel.

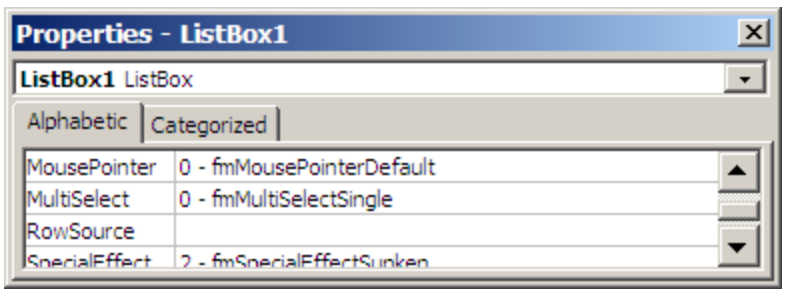
	A	B	C	D	E	F
1	№	Направление	Плацкарт	Купе	СВ	
2	128	Екатеринбург-Челябинск	2000	1500	1000	
3	130	Екатеринбург-Пермь	2500	1800	1300	
4	21	Екатеринбург-Москва	5000	3500	3000	
5	36	Екатеринбург-Москва	4500	3200	2800	
6	40	Екатеринбург-Москва	4800	3300	3000	
7	45	Екатеринбург-С.Петербург	5200	3800	3200	
8	56	Екатеринбург-С.Петербург	5100	4000	3500	
9	150	Екатеринбург-Уфа	2400	1700	1200	
10	160	Екатеринбург-Тюмень	1500	1000	800	
11						

Лист1 | список | Лист3

Пользовательская форма имеет вид:



У элемента ListBox1 свойство



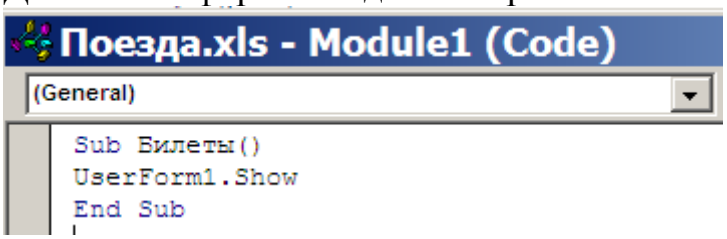
Список заполняется при инициализации формы. Количество элементов в списке определяются во время работы программы, а не во время создания формы, как это было в предыдущем задании.

```
Private Sub UserForm_Initialize()
Dim i As Integer
i = 1
ListBox1.ColumnWidths = "20;140"
With ThisWorkbook.Sheets("список")
Do While .Cells(i, 1).Text <> ""
ListBox1.AddItem .Cells(i, 1).Value
ListBox1.List(i - 1, 1) = .Cells(i, 2).Value
i = i + 1
Loop
End With
End Sub
```

При щелчке по кнопке Оплата выполняются операторы:

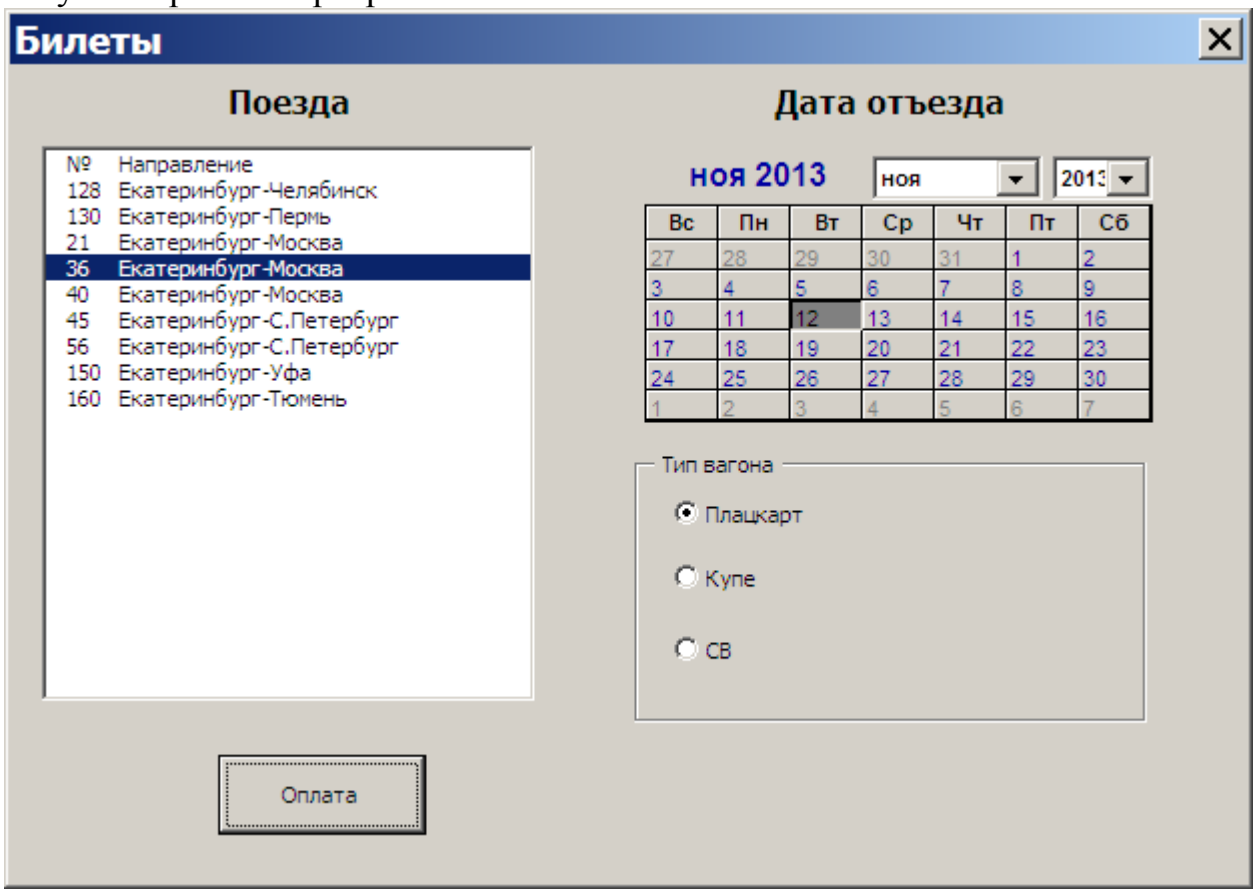
```
Private Sub CommandButton1_Click()
Dim s As Integer, cena As Single
Dim i As Integer, k As Integer
Dim m As Integer, pr As Single
'Определение номера выделенной строки
i = ListBox1.ListIndex
'Определение номера столбца с категорией вагона
If OptionButton1.Value = True Then k = 3
If OptionButton2.Value = True Then k = 4
If OptionButton3.Value = True Then k = 5
'Определение цены билета
s = ThisWorkbook.Sheets("список").Cells(i, k).Value
'Определение месяца поездки
m = Calendar1.Month
'Определение процента скидки/надбавки
Select Case m
Case 1, 2, 12:
pr = 0.9
Case 6, 7, 8:
pr = 1.2
Case Else
pr = 1
End Select
'Вычисление и вывод цены.
cena = s * pr
MsgBox "цена билета = " + CStr(cena)
End Sub
```

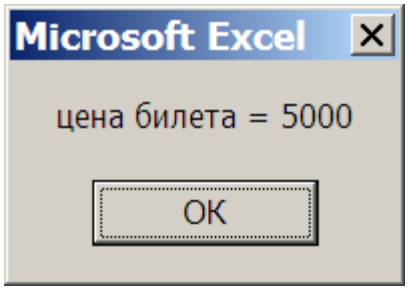
Для вызова формы создаем макрос Билеты:




```
Sub Билеты()
UserForm1.Show
End Sub
```

Результат работы программы:





Задания для самостоятельной работы

В приведенных задания используются списки. Необходимо определить какой режим для работы со списками надо задать: выпор нескольких вариантов или один вариант.

Данные для списков считываются из листа Excel. Результаты надо заносить в таблицу Excel/

Вариант № 1

Книжный интернет-магазин:

Вывести название книг с ценой. Указать варианты доставки: курьерская (постоянная цена 800 руб), наложенным платежом (зависит от кол-ва книг и от общей стоимости) и оплата через банк (зависит от стоимости книг).

Написать приложение для выбора книг и подсчета общей стоимости.

Вариант № 2

Ателье мод:

Вывести перечень изделий с ценой. Указать дополнительные услуги: сложность (процент от стоимости изделия), срочность (процент от стоимости и сложность), доставка на дом (конкретная сумма).

Написать приложение для заказа изделий и подсчета общей стоимости.

Вариант № 3

Хозяйственный магазин:

Вывести перечень товаров с указанием цены. Указать вид оплаты: наличные, карточка Visa (скидка 5%), карточка MasterCard (скидка 3%).

Написать приложение для покупки товара и подсчета общей стоимости.

Вариант № 4

Магазин с оплатой за валюту:

Вывести перечень товаров с указанием цены в рублях. Указать в какой валюте будет оплата с указанием курса.

Написать приложение для покупки товара и подсчета общей стоимости.

Вариант № 5

Покупка туристической путевки в Москву:

Вывести названия гостиниц для проживания с ценой. Указать дополнительную доплату: дорога, питание, все включено (процент от стоимости гостиницы), экскурсии.

Написать приложение для покупки путевки и подсчета общей стоимости.

Вариант № 6

Гостиница:

Вывести категории номеров с ценой за день. Указать дополнительные услуги: все включено (зависит от категории номера), 2-х разовое питание (за 1 день), 3-х разовое питание (за один день), пользование бассейном (разовый взнос).

Написать приложение для покупки путевки и подсчета общей стоимости при проживании недель.

Вариант № 7

Компьютерный магазин:

Вывести список компьютеров с мониторами и указать их цену. Перечислить лицензионной программное обеспечение, которое нужно установить на компьютере.

Написать приложение для покупки компьютера и подсчета общей стоимости

Вариант № 8

Магазин стиральных машин:

Вывести список стиральных машин с указанием цены. Указать дополнительные услуги: увеличение гарантийного срока, дополнительные аксессуары, бесплатное подключение.

Написать приложение для покупки стиральной машины и подсчета общей стоимости

Вариант № 9

Магазин корпусной мебели:

Вывести список изделий из мебели (комод, шифоньер, тумбочка,…) с ценой изготовления. Указать вид материала.

Написать приложение для покупки набора мебели и подсчета общей стоимости. Общая стоимость вычисляется как проценты от цены изготовления.

Вариант № 10

Покупка кухни:

Вывести список изделий для кухни (плита, посудомоечная машина, разделочный стол, стол, пенал, навесной шкаф) с ценой. Вывести варианты материала для столов и шкафов.

Написать приложение для покупки набора для кухни и подсчета общей стоимости. Стоимость столов и шкафов зависит от выбранного материала.

Вариант № 11

Подключение телефона:

Вывести список тарифов с ценой. Указать дополнительные услуги.

Написать приложение для подключения телефона.

Вариант № 12

Тренажерный зал:

Вывести список тренажеров, которые имеются в тренажерном зале с ценой. Указать время посещения зала (утром, днем, вечером, выходные дни).

Написать приложение для посещения зала с подсчетом общей стоимость. Общая стоимость зависит от времени, для утра и дня - скидки. Самое дорогое время – выходные, самое дешевое – утро.

Лабораторная работа № 12
Отбор данных

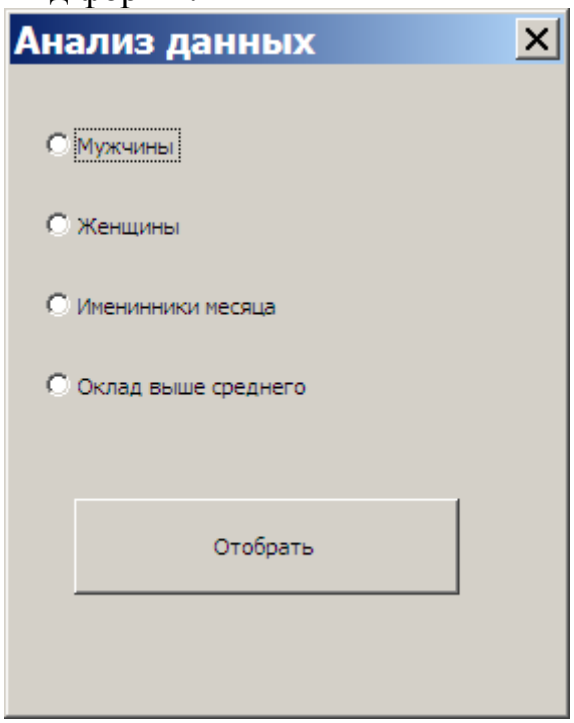
При работе с большими таблицами часть требуется отбирать данные, удовлетворяющие определенным условиям. Это обычно делается с помощью инструмента «Фильтр».

Но с помощью макросов так же можно отбирать данные, удовлетворяющие часто встречающимся условиям и делать это можно с помощью формы.

Задание 1

Создать макрос в котором для файла *Сотрудники* с помощью формы можно выполнять отбор данных, переписав отобранные данные на новый лист.

Вид формы:



Код обработчика события Щелчок по кнопке будет иметь вид:

Создание новых листов для отбора данных.

```
Private Sub CommandButton1_Click()
Dim n As Integer, k As Integer
Dim i As Integer, j As Integer
Dim sum As Long, sr As Single
Sheets("Сотрудники").Activate
Sheets.Add before:=Sheets(1)
k = 1
```

В переменной n вычисляется количество рядов в таблице. Переменная k нужна для нумерации рядов на новом листе. Переменные i и j номера строк и столбцов соответственно.

При выборе пункта Мужчины рассматриваются отчества в 4 столбце. Если отчество заканчивается на «ч», то это мужчина, если на «а», то это женщина.

```
If OptionButton1.Value Then
Sheets(1).Name = "мужчины"
Sheets("Сотрудники").Activate
n = Range("A2").CurrentRegion.Rows.Count
For j = 1 To 9
Sheets(1).Cells(1, j).Value = Cells(1, j).Value
Next
For i = 2 To n
If Right(Cells(i, 4).Text, 1) = "ч" Then
k = k + 1
For j = 1 To 9
Sheets(1).Cells(k, j).Value = Cells(i, j).Value
Next
End If
Next
End If
If OptionButton2.Value Then
Sheets(1).Name = "Женщины"
Sheets("Сотрудники").Activate
n = Range("A2").CurrentRegion.Rows.Count
For j = 1 To 9
Sheets(1).Cells(1, j).Value = Cells(1, j).Value
Next
For i = 2 To n
If Right(Cells(i, 4).Text, 1) = "а" Then
k = k + 1
For j = 1 To 9
Sheets(1).Cells(k, j).Value = Cells(i, j).Value
Next
```

При отборе именинников месяца определяется месяц рождения и текущий месяц.

```
If OptionButton3.Value Then
Sheets(1).Name = "Именинники"
Sheets("Сотрудники").Activate
n = Range("A2").CurrentRegion.Rows.Count
For j = 1 To 9
Sheets(1).Cells(1, j).Value = Cells(1, j).Value
Next
For i = 2 To n
If Month(Cells(i, 7).Value) = Month(Date) Then
k = k + 1
For j = 1 To 9
Sheets(1).Cells(k, j).Value = Cells(i, j).Value
Next
End If
Next
End If
```

Отбор сотрудников с окладов выше среднего.

```
If OptionButton4.Value Then
Sheets(1).Name = "Больше среднего"
Sheets("Сотрудники").Activate
n = Range("A2").CurrentRegion.Rows.Count
For j = 1 To 9
Sheets(1).Cells(1, j).Value = Cells(1, j).Value
Next
sum = 0
For i = 2 To n
sum = sum + Cells(i, 9).Value
Next
sr = sum / (n - 1)
For i = 2 To n
If Cells(i, 9).Value > sr Then
k = k + 1
For j = 1 To 9
Sheets(1).Cells(k, j).Value = Cells(i, j).Value
Next
End If
Next
End If

End Sub
```

Задания для самостоятельной работы.

Вариант 1

В файле Горные лыжи.xls выполнить отбор:

- Товары категории Одежда проданные в феврале
- Товары фирмы Atomic.
- Товары стоимостью больше 20000.
- Проданное снаряжение со скидкой 5%.

Вариант 2

В файле Кадры.xls выполнить отбор:

- Клиенты женщины из сферы промышленность
- Клиенты моложе 30 лет
- Клиенты мужчины без высшего образования
- Клиенты безработные.

Вариант 3

В файле Склад.xls выполнить отбор:

- Товара огурцы
- Товара со стоимостью ниже средней.
- Товар заказчика Атлант количество меньше 5000
- Товары поступившие в июле со скидкой 10%.

Вариант 4

В файле Курсы.xls выполнить отбор:

- Учащиеся из Магнитогорска
- Учащиеся с высшим образованием
- Учащиеся старше 40 лет
- Учащиеся инженеры моложе 35

Вариант 5

В файле Сетевые продажи.xls выполнить отбор:

- Продажи, выполненные летом
- Продажи, дороже 1000 выполненные продавцом, у которого спонсора женщина
- Продажи весной, товар дороже средней цены
- Продажи товара содержащего слово крем.

Вариант 6

В файле Сетевые продажи.xls выполнить отбор:

- Продажи товаров Форевер.
- Продажи за последние 2 месяца товаров с итоговой ценой дороже 1000
- Продажи товаров с количеством больше среднего.
- Продажи Елагина.

Вариант 7

В файле Клининговая компания.xls выполнить отбор:

- Уборки у клиента Топоногова
- Уборки, выполненные первой бригадой в марте.
- Уборки с площадью выше средней.
- Весенние уборки в процентом скидок больше 10.

Вариант 8

В файле Языковый центр.xls выполнить отбор:

- Учащиеся, изучающие немецкий язык в продвинутой группе
- Учащиеся в малочисленных группах номером группы
- Учащиеся мужчины
- Учащиеся изучающие французский язык без с кидок

Вариант 9

В файле Автосалон.xls выполнить отбор:

- Автомобили доставленные менее чем за 15 дней

- Японские автомобили стоимостью более 700000
- Автомобили из Токио со сроком выполнения заказа более 6 дней.
- Автомобили со стоимостью меньше средней.