Лабораторная работа VBA № 9.
Линейный алгоритм.

Цель работы: составить программу для вычисления значения функции *y(x)* при заданном значении аргумента *x*; вывести значения аргумента и функции.

Задача 1. Вычислить значения аналитического выражения (линейный алгоритм).

$$y = \frac{a \cdot x^2 - \cos^2 a \cdot x}{e^x},\ a = 1.4$$

при x=5.2

Порядок работы

1. Изучите лекционный материал *VBA* (или методические указания, см. приложение 1 стр. 10–29 «**Методические указания**»).

2. Средствами *VBA* согласно вашему варианту вычислите значения выражения (см. приложение 2 стр. 30–33). В качестве образца четыре способа решения задачи представлены на стр. 2–8. Составьте программу 4 способами, показанными ниже. Уметь ответить на вопросы по решению задачи всеми 4 способами.

3. По выполненной работе составить отчет в электронной форме (файл *Word*), куда включить титульный лист, на последующих страницах поместить содержание задания и результаты выполненного задания. Для иллюстрации выполненного задания используйте скриншот.

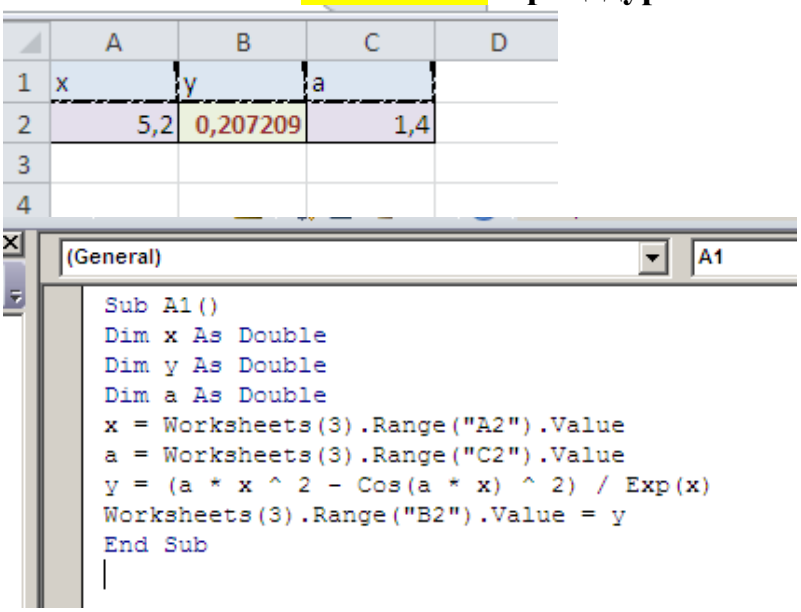
4. При защите работы проиллюстрируйте различные варианты ввода и вывода значений: оператор присваивания, считывание с ячейки, создание окна ввода – *InputBox*; создание окна вывода *MsgBox*, запись в ячейку (см. стр. 20 приложения 1).

Примеры расчета значения арифметического выражения

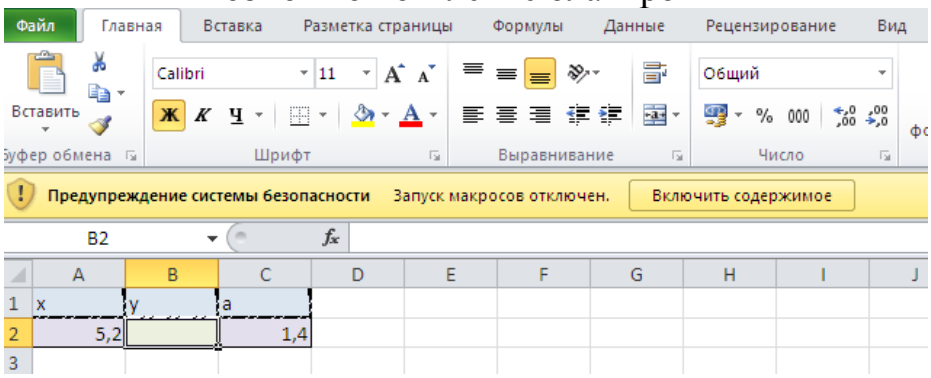
$$y = \frac{a \cdot x^2 - \cos^2 a \cdot x}{e^x}, \; a = 1.4$$

при x=5.2

Способ №1 Процедура



Возможно появление блакировки



Для активизации программы выбрать нужную опцию.

Пояснение к программе

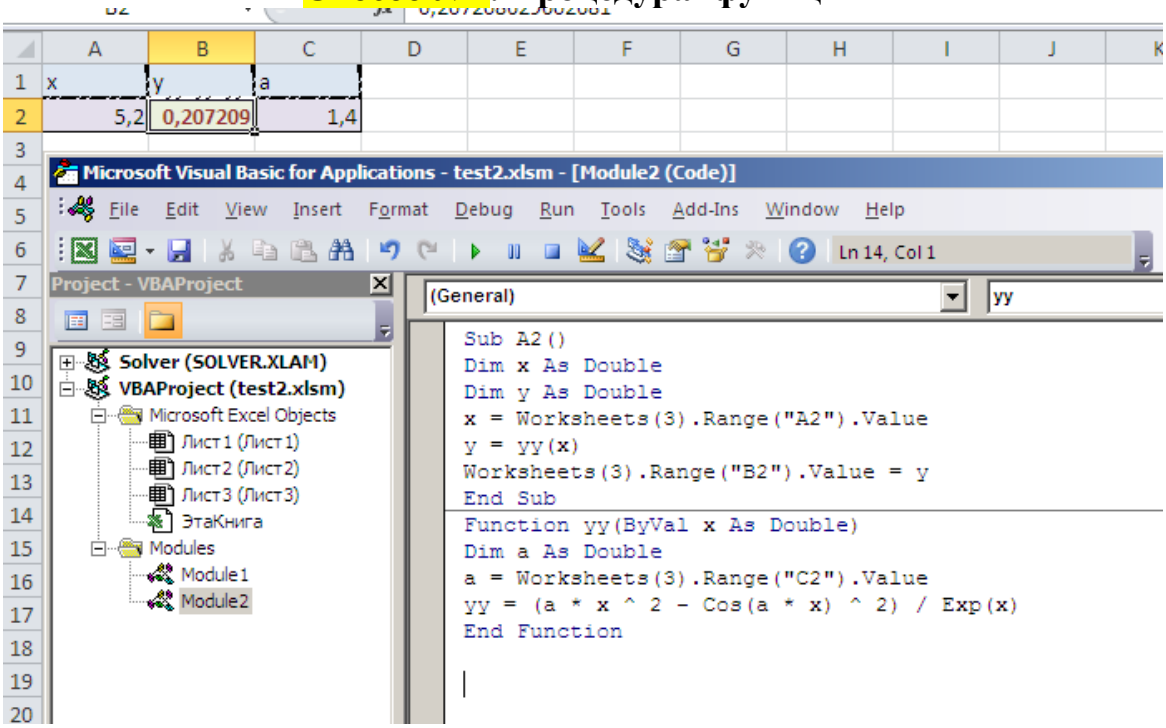
Sub A1() Sub - *назначение (начало) процедуры,* A1 - *имя процедуры,* () - *обязательные скобки, где при необходимости перечисляются передаваемые величины.*

Dim x As Double Описание переменной x с указанием типа переменной As Double (с двойной точностью). Dim – необходимый код перед началом описания переменной.

Dim y As Double

Dim a As Double
x = Worksheets(3).Range("A2").Value Значение переменной x считывается с ячейки A2 Листа 3
a = Worksheets(3).Range("C2").Value
y = (a * x ^ 2 - Cos(a * x) ^ 2) / Exp(x) **Внимание!** Аргумент функции записывается в скобках (например, (a * x)- аргумент функции Cos)
Worksheets(3).Range("B2").Value = y Значение y записывается в ячейку B2 Листа 3
End Sub Конец процедуры

Способ №2. Процедура+функция



Пояснение к программе

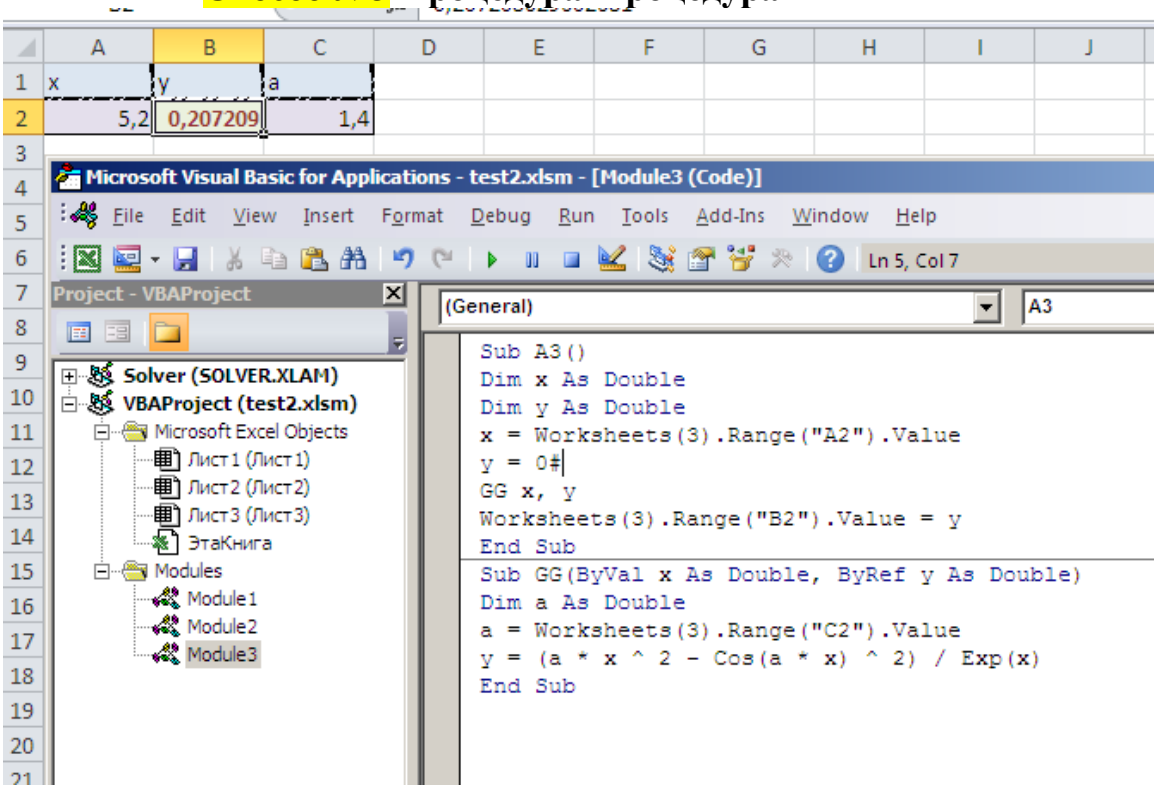
Sub A2()
Dim x As Double
Dim y As Double
x = Worksheets(3).Range("A2").Value
y = yy(x) Вызывается функция по имени yy и передается значение переменной x в функцию yy
Worksheets(3).Range("B2").Value = y
End Sub
Function yy(ByRef x As Double) Function – *назначение (начало) функции*, yy – *имя функции, в скобках указывается имя передаваемой переменной*

x, указывается способ передачи (ByRef– по ссылке)*, *указывается тип переменной x.*

* - вместо (ByRef x As Double) могли написать (x As Double) , что означало то же самой, вместо (ByRef x As Double) могли написать (ByVal x As Double) , что означает передачу по значению. В данном контексте при трех указанных записях результат будет одинаковым.

```
Dim a As Double
a = Worksheets(3).Range("C2").Value
yy = (a * x ^ 2 - Cos(a * x) ^ 2) / Exp(x) Внимание!
EndFunction
```

Способ №3 Процедура+процедура



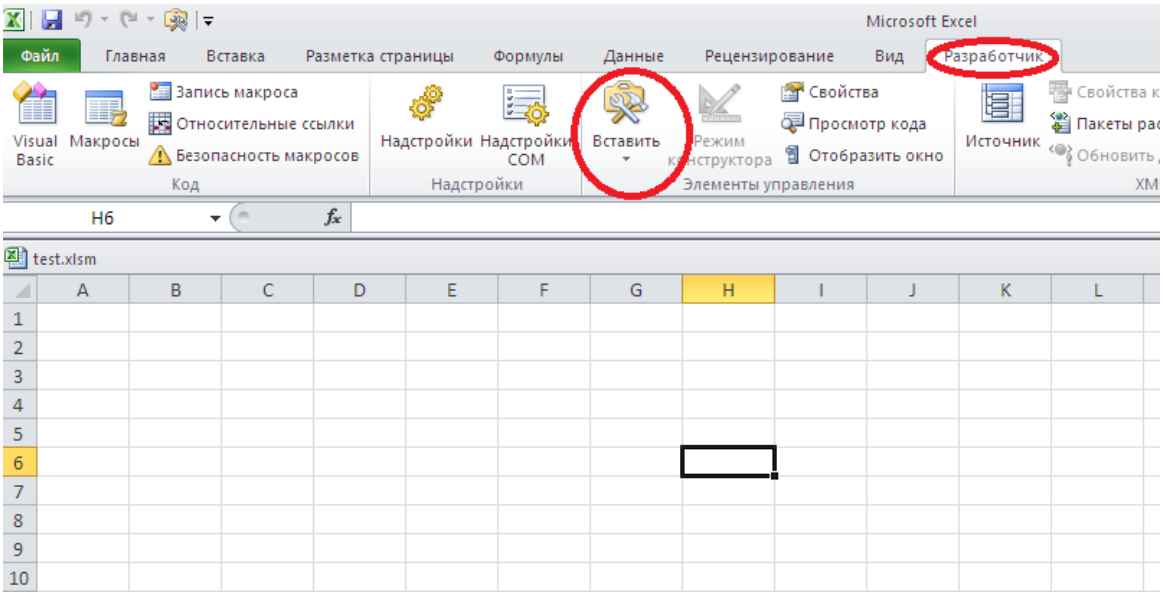
Пояснение к программе

Строка: GG x, y. Здесь вызывается процедура по имени GG и передаются значения переменных x, y

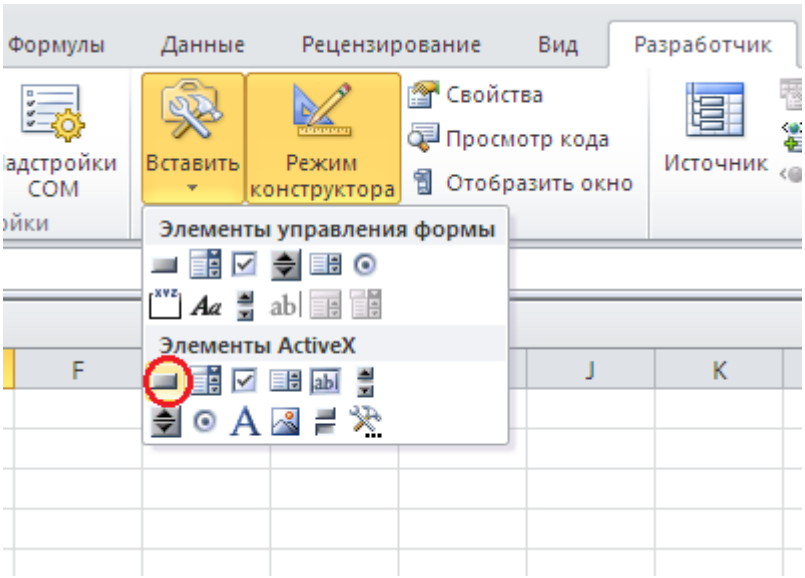
Строка:Sub GG (ByVal x As Double, ByRef y As Double). Передача значения переменной x производится по значению (ByVal x). Здесь возможен любой способ передачи. Но передача значения y по ссылке (ByRef y) здесь принципиальна, так как значение y, рассчитанное именно в процедуре должно сохраниться и передаться в главную процедуру (Sub A3()). Если

указать Sub GG (ByVal x As Double, ByVal y As Double), то после вывода значения y из Sub A3() в ячейке B2 будет 0 (значение y, рассчитанное в Sub GG не будет передаваться в процедуру Sub A3()).

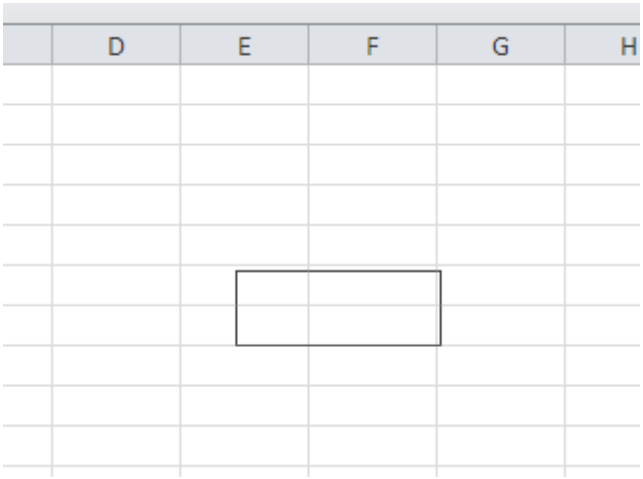
Способ №4 Кнопка+Процедура
Установка кнопки для подключения программы и проведения расчета



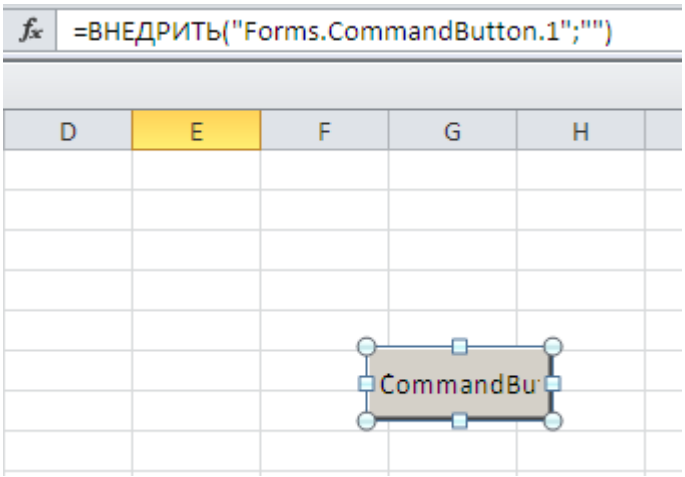
В **Разработчике** выбрать опцию **Вставить**



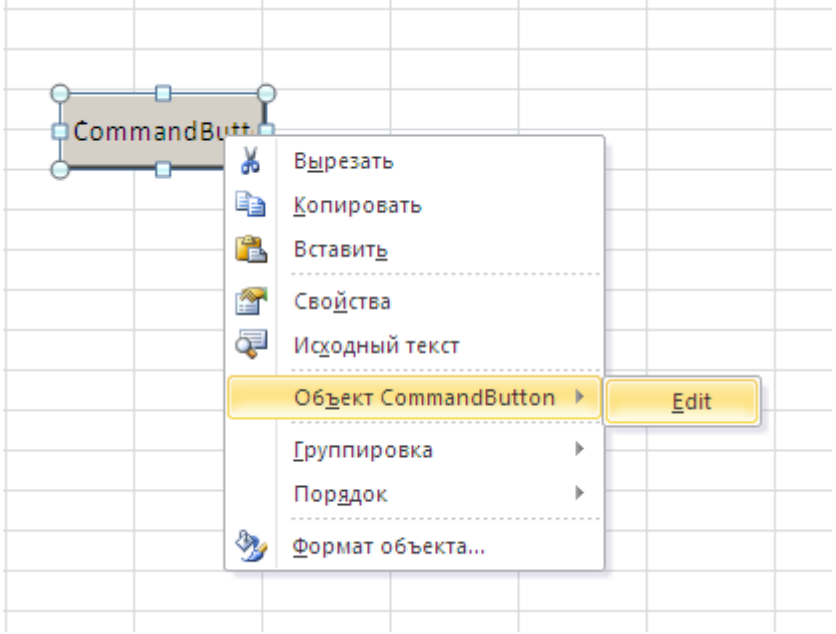
Из списка выбрать **кнопку**



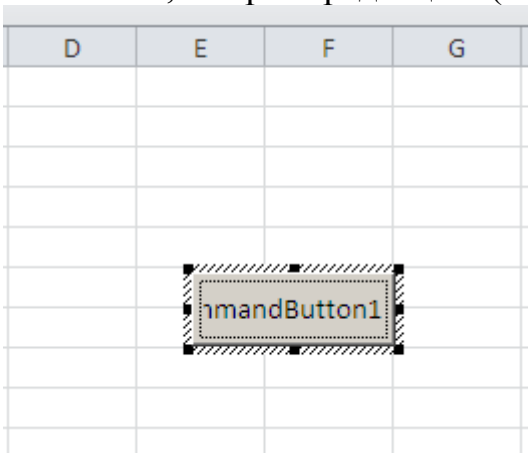
Указать место (используя мышь), где будет располагаться кнопка



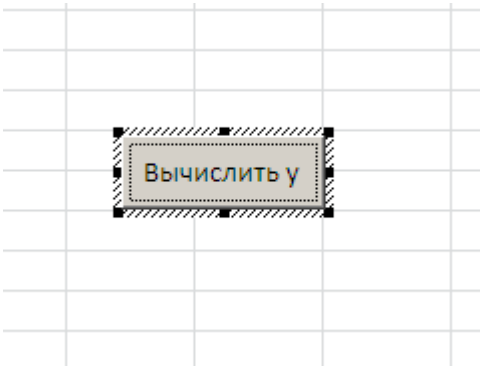
Появится кнопка, которую следует отредактировать.



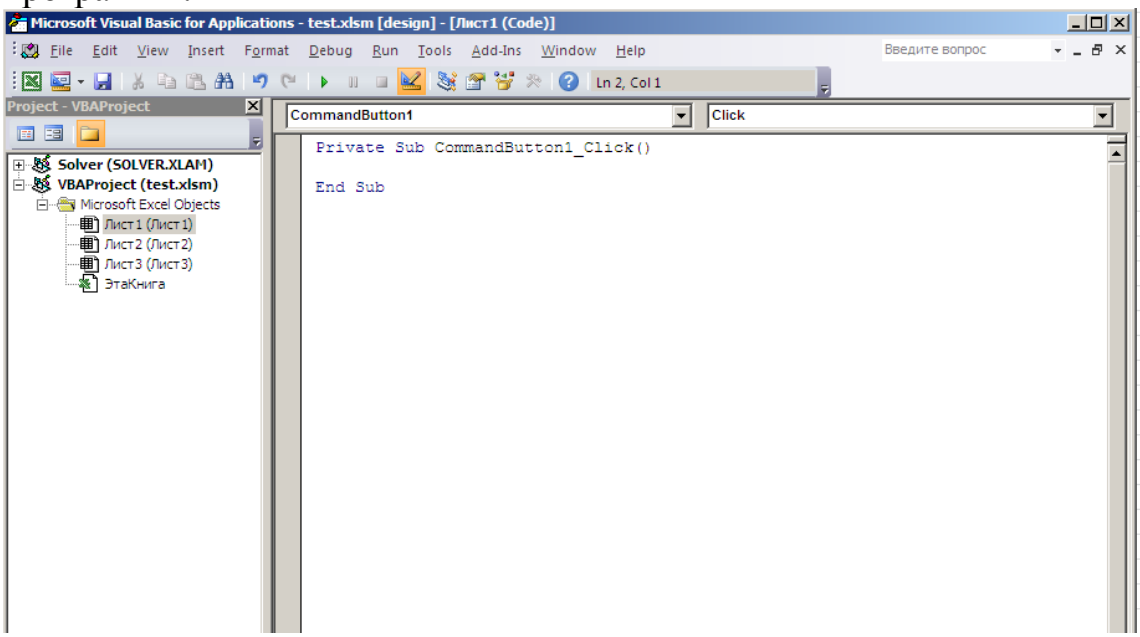
Вызвать контекстное меню, выбрать редакцию (**Edit**)

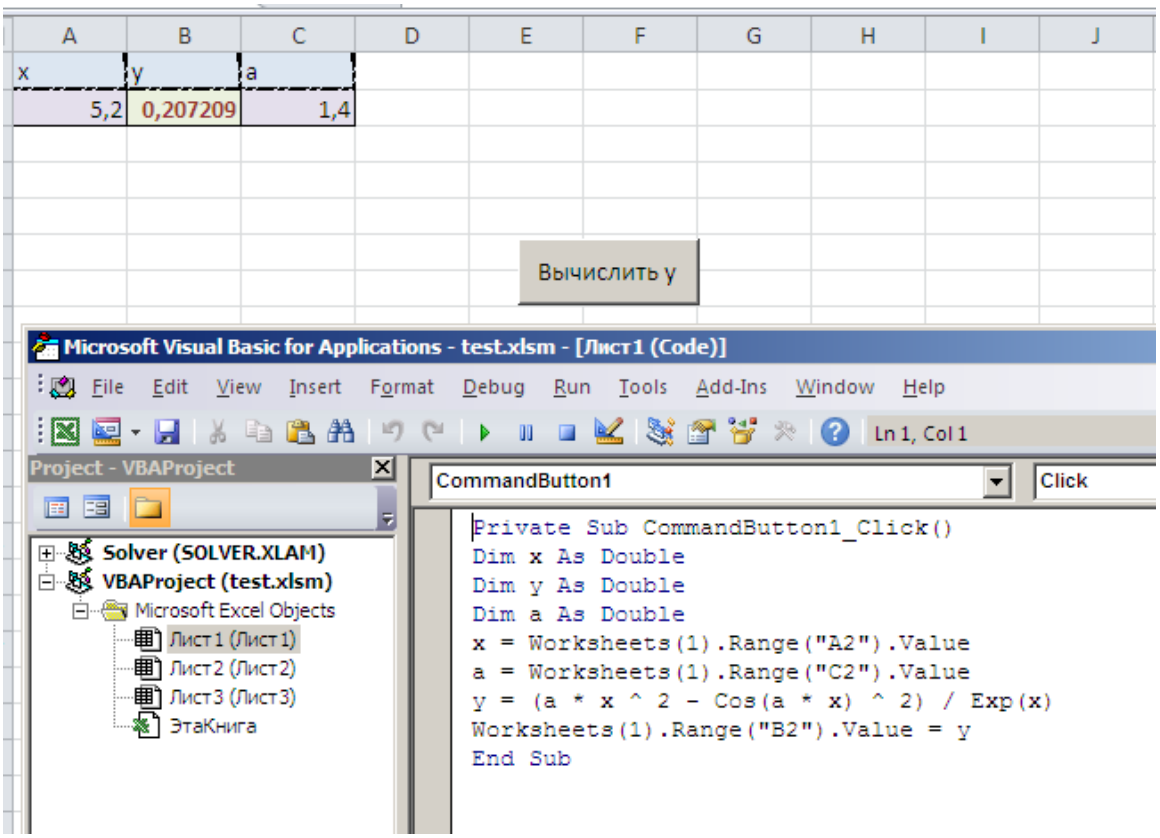


Изменить название кнопки

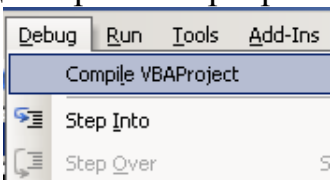


«Кликнуть» кнопку, в появившемся рабочем поле написать код программы.

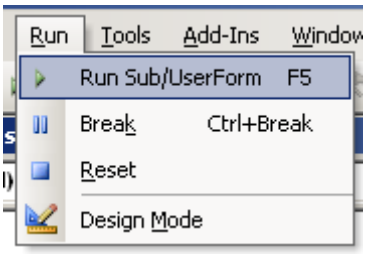




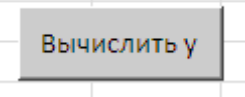
При необходимости отредактировать программу и выбрать



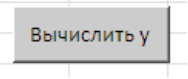
Затем отправить на счет



Результат расчета появится (согласно коду программы) в ячейке B2 (Лист 1). Последующие запуски программы на счет осуществляются при нажатии кнопки



Допуск к тексту программы осуществляется двойным нажатием кнопку



Пример. Вычислить значение выражения

$$y(x)=\frac{\sqrt{a\cdot b\cdot c}}{2.4}-\frac{0.7\cdot a\cdot b\cdot c\cdot x}{(a+b)^2}-e^{a+b}$$

при c=1.5, d=0.6, x=0.64.

Решение.

Программа и результат расчетов представлены на рис. 1.

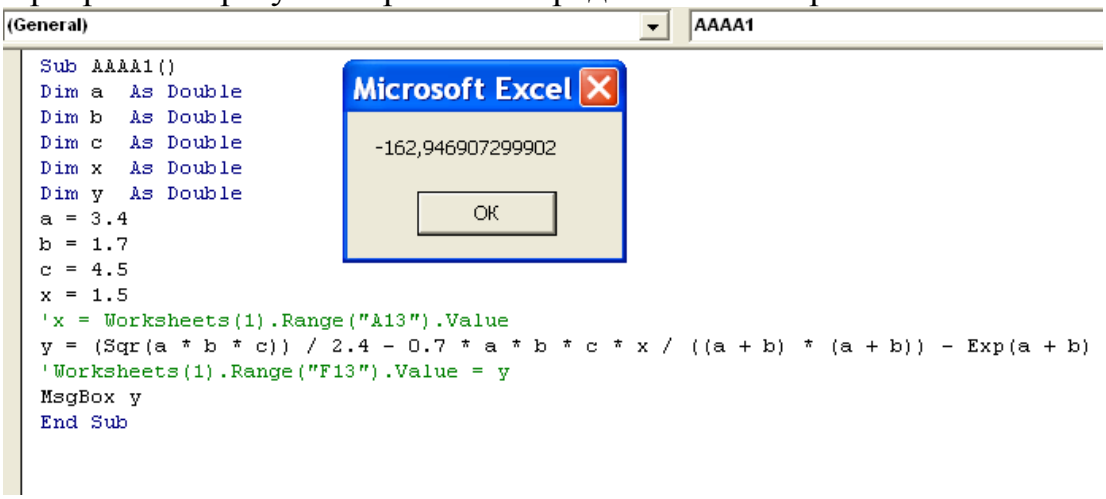


Рис. Листинг программы для расчета значения выражения с результатом расчета

Замечание: первый символ строки:

`'x=Worksheets(1).Range("A13").Value`

есть апостроф (′). После символа ′ (см. на клавиатуре клавишу с буквой Э) записываются комментарии, не участвующие в работе программы.

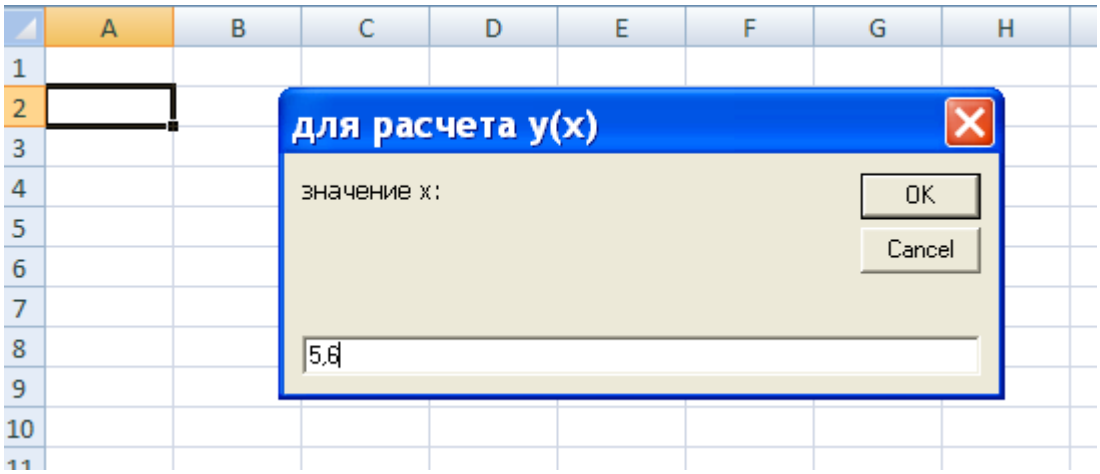


Рис. Результат выполнения оператора ввода `x=InputBox("значение x","для расчета y(x)")`

Приложение 1.

Методические указания.
Введение в *VBA*

Создание приложений на языке *Visual Basic for Applications* (*VBA*)

Visual Basic for Applications (*VBA*) — среда визуального объектно-ориентированного программирования для создания прикладных программ в среде *Microsoft Office*.

С помощью *VBA*:

- создаются объекты управления графического интерфейса пользователя;
- задаются и изменяются свойства объектов;
- подключается соответствующий программный код.

Структура программного кода

Программа на языке *VBA* имеет модульную структуру, в составе которого вложенные модули, содержащие одну или более вложенных процедур. Каждая переменная имеет сферу действия (уровень видимости). В *VBA* переменные имеют три уровня видимости:

- уровень процедуры;
- уровень модуля;
- уровень проекта (общий уровень)

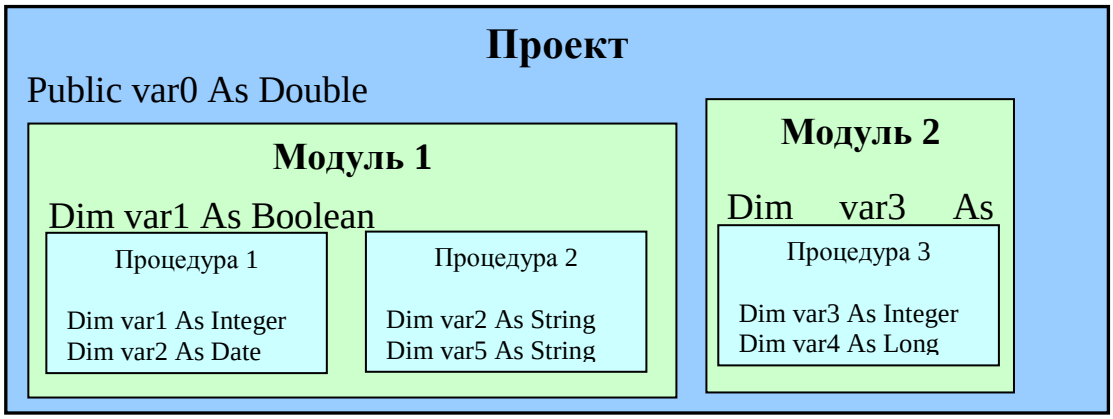


Рис. 11. Пример общей структуры программы

Процедуры и функции

Программные модули *VBA* состоят из одной или более подпрограмм (процедур или функций). Процедура − минимальный

модуль в составе прикладной программы на языке *VBA*. Процедуры имеют стандартное оформление:

Sub <имя_процедуры> (аргументы)
тело процедуры (операторы)
End Sub

Оператор ***Sub*** - объявление процедуры, задается имя, указывается состав аргументов, передаваемых при вызове процедуры из программы. Каждому оператору ***Sub*** обязательно соответствует ***End Sub***.

Функция − отличается от процедуры следующим:

- начинается ключевым словом ***Function*** и заканчивается ключевыми словами ***End Function***;
- функцию можно вызвать из формулы, введенной в ячейку;
- функция может возвращать значение в вызывающую программу или формулу.

Вызов процедуры общего назначения выполняется по имени:

<Имя>(<Фактические аргументы>)

При вызове процедуры фактические аргументы подставляются на место формальных и управление выполнением передается процедуре. Аргументы могут быть входными, выходными или модифицируемыми. Через входные аргументы процедура получает данные при обращении к ней. Выходные аргументы возвращают результаты выполнения процедуры. Модифицируемые аргументы являются одновременно входными и выходными.

Функция общего назначения построена также как процедура, однако, результат работы функции передается (возвращается) через ее имя. Поэтому, как и в математике, обращения к функциям можно использовать внутри арифметических и логических выражений.

Область видимости процедур и функций

У процедур и функций два уровня видимости:

- уровень модуля;
- уровень проекта.

Служебные слова ***Private*** и ***Public*** задают область видимости процедур и функций. ***Private*** делает объект доступным только внутри данного модуля. ***Public*** делает объект доступным из другого модуля. Для того чтобы перевести процедуру или функцию на уровень модуля, необходимо объявить ее с ключевым словом ***Private.*** Такие программы

не могут запускаться сами по себе. Их можно только вызвать из других программ. Рассмотрим пример.

```
Sub Proc()
   var1=GetRand
   ModuleLevelProcedure var1
End Sub

Private Sub ModuleLevelProcedure(ByVal var1)
   MsgBox var1*GetRand
End Sub

Private Function GetRand()
   GetRand=Int(100*Rnd())
End Function
```

Сначала процедура ***Proc()*** вызывает функцию ***GetRand*** для генерации случайного целого числа в диапазоне от 0 до 99, а затем передает его в процедуру ***ModuleLevelProcedure***. Там этот аргумент умножается на результат еще одного обращения к функции ***GetRand***. Их произведение выводится в информационном окне.

Переменные

Переменная − поименованная область в памяти компьютера во время выполнения программы. Переменная предназначена для хранения и изменения значений во время выполнения программы. Переменная требует явного объявления своего имени. Имя образуется из алфавитно-цифровых символов и знака подчеркивания **_.** Имя всегда начинается с буквы и представляет собой непрерывную последовательность символов, но не более 254; пробелы в имени не допускаются.

Итак, в языке *Visual Basic* действуют следующие соглашения на имена процедур, переменных и констант:

- должны начинаться с буквы;
- могут включать буквы, цифры и символы подчеркивания;
- не должны включать знаки препинания или пробелы;
- не должны совпадать с ключевыми словами языка *Visual Basic*.

Описание переменных

Модуль, тело процедуры или функции обычно начинаются с раздела описаний. Он содержит определения переменных и констант, которые используются в модуле и процедурах. С помощью переменных

в процедуры передаются аргументы, в ходе выполнения процедур сохраняются рабочие промежуточные значения, осуществляется обмен данными между процедурами. Переменные существуют только внутри модулей, процедур или функций. Каждая переменная имеет имя. Основной инструкцией для явного описания переменных является инструкция ***Dim*.** При определении переменной для нее указывается тип данных.

Формат инструкции описания переменной:

Dim <Имя переменной> [***As*** <Тип данных>]

Следующая инструкция создает переменную *x* и указывает для нее текстовый (строковый) тип данных ***String***: ***Dim x As String.*** Если разместить данную инструкцию внутри процедуры, то переменная *x* может быть использована только внутри этой процедуры. Если поместить данную инструкцию в раздел описаний модуля, то переменная *x* будет доступна для любых процедур в данном модуле, но недоступна для процедур в других модулях. Для того чтобы сделать данную переменную доступной для всех процедур в базе данных, следует описать ее как общую с помощью инструкции ***Public***: ***Public x As String***

Основными типами данных, используемыми при описании переменных, являются:

Integer – целое число (2 байта) (например, ***Dim X As Integer, Y As Integer, Z As Integer*)**;

- ***Long*** – длинное целое число (4 байта);
- ***Single*** – десятичное число одинарной точности (4 байта);
- ***Double*** – десятичное число двойной точности (8 байтов);
- ***Currency*** – десятичное число с фиксированной точкой (8 байтов);
- ***String*** – строка текста (до 65400 символов);
- ***Byte*** – целое от 0 до 255 (1 байт);
- ***Boolean*** – логическое значение ***True*** или ***False*** (2 байта);
- ***Date*** – дата и время (8 байтов);
- ***Object*** – экземпляр класса (4 байта);
- ***Variant*** – любой из перечисленных выше типов (16 байтов + 1 байт/символ). Указание типа данных в инструкции описания не является обязательным. Если тип данных не указан, по умолчанию переменная получит тип ***Variant.***

Массивы

Объявление массива

- Локальный массив:

Dim <имя массива> **(**размерность**)** ***As*** <тип элементов массива>

• Глобальный массив:

Public <имя массива> **(**размерность**)** ***As*** <тип элементов массива>

Примеры объявления массива

Одномерный массив, состоящий из 10 элементов –

```
Dim NumberArray(10) As Integer
```

2-х мерный массив, состоящий из $10 \times 20 = 200$ элементов –

```
Dim TableArray(10,20) As String
```

3-х мерный массив –

```
Dim BigArray(5,50,100) As Variant
```

Нумерация элементов массива задается инструкцией ***Option Base*** в начале модуля: ***Option Base*** **0** или ***Option Base*** **1.** По умолчанию нумерация элементов массива начинается с 0.

Использование массива

Пример использования массива типа ***Integer***:

```
Option Base 1
```

```
Sub UsingArray()
        Dim Vals(3) As Integer
        Vals(1)=Int(100*Rnd())
        Vals(2)=Int(100*Rnd())
        Vals(3)=Int(100*Rnd())
        MsgBox "Lottery numbers: " & Vals(1) & ", "
& Vals(2) & "," & Vals(3)
End Sub
```

Имеются ряд специальных функций для работы с массивами.

Примеры функций для работы с массивами

1. Функция ***Array*** – позволяет создавать массив в ходе выполнения программы, без предварительного описания.

Пример:

```
Sub CreateArray()
    DataCA=Array("Kate",43,#4/15/1962#)
```

```
    MsgBox DataCA(0) & ",age" & DataCA(1) & ", born
    " & DataCA(2)
End Sub
```

2. Функция ***Erase*** – используется для удаления данных, хранимых в элементах массива. Если массив фиксированного размера – очищается содержимое массива, а память, выделенная массиву, остается за ним.

Пример:

```
    Option Base 1
    ____________________________________________
Sub EraseArray()
    Dim EA(2) As Integer
    EA(1) = Int(100 * Rnd())
    EA(2) = Int(100 * Rnd())
    MsgBox "Lottery numbers: " & EA(1) & " ," &
EA(2)
    Erase EA
    MsgBox "Lottery numbers: " & EA(1) & " ," &
EA(2)
End Sub
```

3. Функция ***IsArray*** – позволяет проверить, является ли переменная массивом. Функция имеет один аргумент и возвращает ***True***, если переменная является массивом, и ***False***, если переменная не массив. Функция полезна, если необходимо проверить, возвращает ли вызываемая функция массив или обычную переменную.
Пример:

```
Sub IsArrayFunction()
    Dim Arr(2) As Integer
    Dim ArrayBool As Boolean
    ArrayBool = IsArray(Arr)
    If ArrayBool = True Then
```

```
    MsgBox "Arr is an array."
    End If
End Sub
```

4. Функции ***LBound*** и ***UBound*** – определяют верхнюю и нижнюю границы индексов элементов массива.
Пример:

```
Sub LBoundAndUBound()
    Dim Data(4 To 15) As Integer
    MsgBox "The lower bound is " & LBound(Data) &
"."
    MsgBox "The upper bound is " & UBound(Data) &
"."
End Sub
```

Перед вызовом функций ***LBound*** и ***UBound*** рекомендуется проверить переменную функцией ***IsArray***.

Передача данных при вызове подпрограммы

Передача аргументов из одной программы в другую осуществляется двумя способами:

- по ссылке (***by reference***);
- по значению (***by value***).

Выбор способа передачи – с помощью ключевых слов ***ByRef*** и ***ByVal*:**

по ссылке – передается сама переменная (имя переменной), поэтому ее значение в подпрограмме можно изменить;

по значению – передается только значение переменной; изменить это значение в вызванной подпрограмме нельзя.

При отсутствии в описании ключевых слов передача осуществляется по ссылке.

Пример передачи переменной по ссылке

```
Sub PassArgumentByReference()
```

```
    Dim Username As String
    Username = "Mike"
    ChangeName Username
    MsgBox Username
End Sub

_____________________________________
Sub ChangeName(ByRef Username)
    Username = "Kate"
End Sub
```

В процедуре *PassArgumentByReference()* переменная *Username* передается в процедуру *ChangeName* по ссылке. Это означает, что в *ChangeName* используется имя переменной *Username*, а именно – переменной *Username* присваивается новое значение, которое затем возвращается в процедуру.

Пример передачи переменной по значению

```
Sub PassArgumentByVal()
    Dim Username As String
    Username="Mike"
    ChangeName Username

    MsgBox Username
End Sub
```

Mike

```
_____________________________________
Sub ChangeName(ByVal Username)
    Username="Kate"
End Sub
```

Операторы, используемые в выражениях

Таблица 1

Оператор	Пример использования	Описание
+	a + b	Сложение двух чисел
-	a - b	Вычитание
*	a * b	Умножение
/	a / b	Деление

\	a \ b	Целочисленное деление
Mod	a Mod b	Возвращается остаток от деления
^	a ^ b	Возведение в степень
And	выражение1 And выражение2	Логическое “И”
=	выражение1 = выражение2	Оператор эквивалентности
>	выражение1 > выражение2	Оператор сравнения “больше”
>=	выражение1 >= выражение2	Оператор сравнения “больше или равно”
<>	выражение1 <> выражение2	Оператор сравнения “не равно”
<	выражение1 < выражение2	Оператор сравнения “меньше”
<=	выражение1 <= выражение2	Оператор сравнения “меньше или равно”

Основные математические функции VBA

Таблица 2

Функция	Пример использования	Описание	Тип возвращаемого результата
Abs	Abs(x)	Модуль числа x	Совпадает с типом числа x
Atn	Atn(x)	Арктангенс числа *x*	Double
Cos	Cos(x)	Косинус числа *x*	Double
Exp	Exp(x)	Экспоненциальная функция (в степени *x*)	Double

Fix	Fix(x)	Возвращает целую часть числа *x*. Если *x* - отрицательное, то возвращаемое значение округляется в <u>большую</u> сторону (напр., если *x*=-1.5, возвращается -1).	Integer
Int	Int(x)	Возвращает целую часть числа *x*. Если *x* - отрицательное, то возвращаемое значение округляется в меньшую сторону (напр., если *x*=-1.5, возвращается -2).	Integer
Log	Log(x)	Натуральный логарифм числа x	Double
Rnd	Rnd или Rnd(x)	Генерируется случайное число. Если аргумент не указан – генерируется число в диапазоне от 0 до 1	Single
Sgn	Sgn(x)	Знак числа (1 0 -1)	Integer
Sin	Sin(x)	Синус числа *x*	Double
Sqr	Sqr(x)	Квадратный корень из *x*	Double
Tan	Tan(x)	Тангенс числа *x*	Double

Дополнение к основным математическим функциям *VBA*:

- $Log_n(x) = Log(x) / Log(n)$
- Для перевода x из градусов в радианы: $x = x * \pi / 180$
- ***IsNumeric*** *()* – встроенная функция *VBA* – возвращает *True*, если ее аргумент является числом (строка), и *False* – в противном случае.
- ***Val*** *()* **–** встроенная функция *VBA* - преобразует переданную ей строку в число.
- ***InputBox*** *()* – выдает запрос на ввод значения; возвращает строковое значение.

Примеры ввода и вывода значений

Ввод:

- `x=5.7` (оператор присваивания);
- `x = Worksheets(1).Range("A1").Value` (значение считывается из ячейки A1);
- `x = InputBox("Enter Number: ", "Calculate Factorial")` (создается окно ввода значений переменной *x* и указываются соответствующие комментарии).

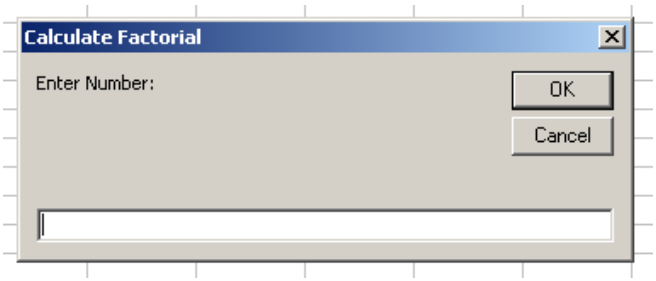


Рис. 12. Окно ввода значений

Вывод:

- `Cells(1,2).Value=5` (присваивает ячейке B1 текущего рабочего листа активной рабочей книги значение 5);
- `Worksheets(1).Range("A1:B2").Value = 10` (присваивание блоку ячеек значения, равного 10);
- `Range("D15").Value="Test"` (присваивание ячейке текущего рабочего листа активной рабочей книги значения *Test)*;
- `MsgBox (x)` (создание окна сообщений, пример результата см. рис. 13).

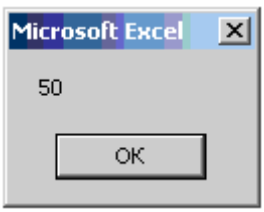


Рис. 13. Результат выполнения оператора вывода MsgBox (x)

Примеры на использование обращений к функции и процедуре

Пример 1:

```
Sub Call_Function()
    Dim var1 As Integer
    Dim var2 As Integer
    Dim var3 As Integer
    var1 = 5
    var2 = 10
    var3 = Multiply(var1,var2)
    MsgBox (var3)
End Sub
```

```
Function Multiply(ByVal var1 As Integer, ByVal var2 As Integer)
    Multiply = var1 * var2
End Function
```

Значение строк программы (пример 1)

Таблица 3

Строка программы	***Назначение строки программы***
`Sub Call_Function()`	начало процедуры с указанием имени процедуры

`Dim var1 As Integer`
`Dim var2 As Integer`
`Dim var3 As Integer`
	блок описания (представляется переменная с именем *var1* и указывается тип переменной *var1* как целочисленный)

`var1 = 5` `var2 = 10`	блок ввода (присваивается переменной *var1* значение 5)
`var3 = Multiply(var1, var2)`	переменной по имени *var3* присваивается значение. Причем это значение появляется в

	результате вызова функции *Multiply* и передачи ей значений переменных *var1*, *var2*, равных соответственно 5 и 10
MsgBox (var3)	оператор вывода (вызывается окно сообщений, где представляется значение переменной *var3*)
End Sub	конец процедуры
_____	граница между процедурой и функцией одного модуля
Function Multiply(**ByVal** var1 **As Integer, ByVal** var2 **As Integer)**	начало функции с указанием имени функции (*Multiply*), в скобках указывается способ передачи значений переменных (*ByVal* – по значению, имя переменной –*var1*, тип переменной – *As Integer*; подобным образом, через запятую, указывается информацию о другой переменной – *var2*)
Multiply = var1 * var2	Результат умножения значений переменных *var1* на *var2* является значением функции. Внимание: имя функции и имя объекта, куда записывается расчетное значение функции – одинаковое.
End Function	Конец функции

Пример 2:

```
Sub DDDD()
Dim x As Double
Dim y As Double
Dim z As Double
x = 1.1
y = 2.2
MsgBox (x)                                    x=1.1
MsgBox (y)                                    y=2.2
TTTT x,y
```

вызываем процедуру по имени TTTT и передаем значения переменных x и y

```
MsgBox (y)                                    y=3.3
MsgBox (x)                                    x=1.1
z= FFF(x)
MsgBox (z)
End Sub
```

```
Function FFF(ByVal x As Double)
FFF = 1/x^2
```

$$FFF = \frac{1}{x^2} \xrightarrow{x=1.1} \frac{1}{1.1^2}$$

```
End Function
```

```
Sub TTTT(ByVal x As Double, ByRef y As Double)
```

процедуре TTTT передаются значения x=1.1 ***«по значению»*** *(способные не изменяться при выходе из процедуры), y=2.2* ***«по ссылке»*** *(способные изменяться при выходе из процедуры)*

```
MsgBox (y)                                    y=2.2
y=x+y                                         y=1.1+2.2=3.3
MsgBox (y)                                    y=3.3
x=5.5
MsgBox (x)                                    x=5.5
End Sub
```

Управляющие структуры

Управляющие структуры предназначены для изменения порядка выполнения инструкций.

If - Then - Else – выполняет группу инструкций, если соблюдено некоторое условие;

For - Next – выполняет группу инструкций заданное число раз;

While - Wend – выполняет группу инструкций, пока соблюдается некоторое условие;

Do - Loop – выполняет группу инструкций, пока соблюдается или не соблюдается некоторое условие;

Select Case – в зависимости от значения некоторой переменной или результата проверки условия выполняет одну или несколько возможных групп инструкций;

For - Each - Next – выполняет действия над каждым объектом семейства или элементом массива.

Примеры использования некоторых управляющих структур

Управляющая инструкция *If - Then - Else*

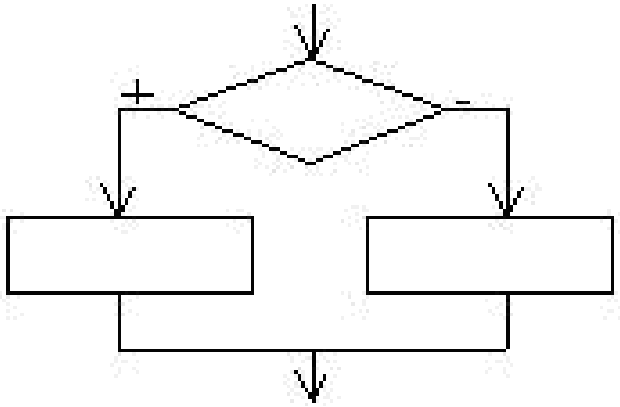
Условная инструкция *If - Then - Else* изменяет порядок выполнения инструкций в зависимости от результатов проверки некоторого условия.

Пример.

$$y = \begin{cases} \cos(x), & x < 7, \\ e^x, & x \geq 7. \end{cases}$$

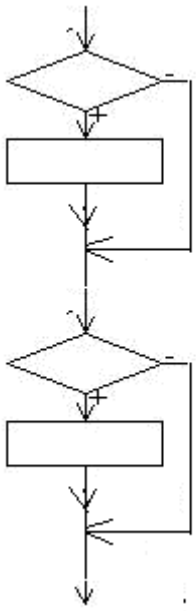
Решение (способ 1).

```
Sub AAAAA()
   Dim x As Double
   Dim y As Double
   x=1.5
   y=0
   If x>= 7 Then
y=exp(x)
   Else y=cos(x)
   End If
End Sub
```



Решение (способ 2).

```
Sub AAAAA()
   Dim x As Double
   Dim y As Double
   x=1.5
   If x< 7 Then  y=cos(x)
   If x>= 7 Then y=exp(x)
   MsgBox y
End Sub
```



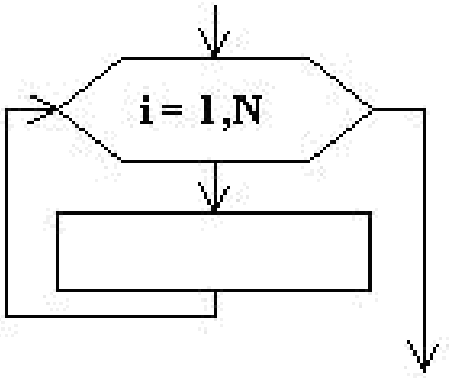
Управляющая инструкция *For - Next*

Управляющая инструкция ***For - Next*** позволяет выполнять несколько команд заданное число раз.

Пример.

$$\sum_{i=1}^{5} V_i = S, \quad V_i = i, \quad S = ?$$

```
Option Base 1
Sub BBBB()
   Dim V(5) As DOUBLE
   Dim S As DOUBLE
   Dim i As Integer
   Dim n As Integer
   n=5
   S = 0
   For i = 1 To n Step 1
   V(i) =i
   S = S + V(i)
   MsgBox V(i)
   Next
   MsgBox S
End Sub
```



Step – любое целое число, определяющее шаг приращения счетчика.

Пример. Дан массив *B*, содержащий 15 элементов. Указанный массив сформирован на основе встроенной функции *Rnd()*. Вывести массив *B* и элементы массива *B*, которые стоят на четных местах и превышают по абсолютному значению 2.5. Записать в массив *T* найденные элементы. Вывести массив *T*.

Решение. Рабочая область листа и программный код показаны на рис. 14 и 15.

(General) CommandButton1_Click

```
Private Sub CommandButton1_Click()
Worksheets(1).Range("A1:C15").ClearContents
Dim arrb(15) As Double
Dim arrt(15) As Double
Dim k, i As Integer
k = 0
For Count = 1 To 15 Step 1
Randomize
arrb(Count) = 10 * Rnd()
Worksheets(1).Cells(Count, 1).Value = arrb(Count)
If (Count Mod 2 = 0) And (Abs(arrb(Count) > 2.5)) Then
Worksheets(1).Cells(Count, 2).Value = arrb(Count)
k = k + 1
arrt(k) = arrb(Count)
End If
Next
For i = 1 To k Step 1
Worksheets(1).Cells(i, 3).Value = arrt(i)
Next
End Sub
```

Рис. 14. Программный код к решению примера на тему «Управляющая инструкция ***For – Next***»

	A	B	C	D	E	F	G	H	I
1	1,225582		4,167703		Работа с элементами одномерного массива				
2	0,050635		5,66309						
3	3,575475		3,526824						
4	0,345165								
5	4,72451								
6	1,586473								
7	3,656377								
8	4,167703	4,167703							
9	3,42001								
10	5,66309	5,66309							
11	8,933851								
12	1,777359								
13	3,246835								
14	3,526824	3,526824							
15	5,34265								
16									
17									
18									
19		Вычислить							
20									

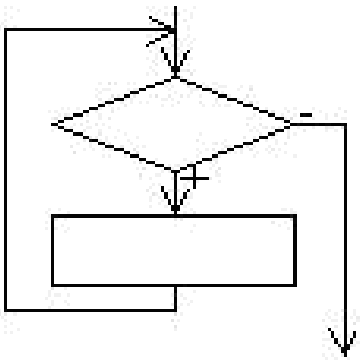
Рис. 15. Рабочая область листа *Excel* к решению примера на тему «Управляющая инструкция ***For – Next***» к рис. 14

Управляющая инструкция *While - Wend*

Управляющая инструкция ***While - Wend*** выполняется до соблюдения определенного условия.

Пример. Выделить значения из последовательности случайных чисел, когда *M* = 7

```
Sub CCCC()
   Dim M As Integer
   Dim n As Integer
   M = 0
   n=0
   Randomize
   While M <> 7
   M = Int(10 * Rnd())
   n = n + 1
   Wend
   MsgBox "n="&n
End Sub
```



Управляющая инструкция *Do – Loop*

Условие завершения цикла может задаваться не только в начале цикла, но и в конце. Условие в конце цикла гарантирует, что он будет выполнен хотя бы один раз. Кроме того, условие можно сделать критерием, как выполнения цикла, так и его завершения.

Пример. Выделение определенного значения (равного 7) из последовательности случайных чисел.

Вариант 1.

```
Sub DoWhileLoop()
   Dim Number As Integer
   Number = 0
   Do While Number <> 7
   Number = Int(10 * Rnd())
   Loop
   MsgBox "Your number is " & Number & " ."
End Sub
```

Вариант 2.

```
Sub DoUntilLoop()
   Dim Number As Integer
   Number = 0
   Do Until Number = 7
   Number = Int(10 * Rnd())
   Loop
   MsgBox "Your number is " & Number & "."
End Sub
```

Управляющая инструкция *For – Each - Next*

Количество повторений операций цикла определяется количеством элементов массива.

Пример 1:

```
Option Base 1
Sub StructureFEN()
   Dim Countries(5) As String
   Dim Country As Variant
   Countries(1) = "India"
   Countries(2) = "Peru"
   Countries(3) = "Greece"
   Countries(4) = "Canada"
   Countries(5) = "Kenya"
   For Each Country In Countries
        MsgBox Country
    Next
End Sub
```

Приложение 2

Варианты задачи 1

1) $y = \dfrac{\sqrt{abc}}{2.4} - \dfrac{0.7abc}{(a+b)^2}x - e^{(a+b)}$

$a = 3.4; b = 1.7; c = 4.5; x = 1.7$

2) $y = \dfrac{\sqrt{c} + d \cdot x^3 - 0.41}{|a-b|} + \dfrac{\ln(a-b)}{(\sqrt{c} + d \cdot x^2)^2} - e^{(a-b)}$

$a = 4.5; b = 1.2; c = 4.3; d = 3.1; x = 1.4$

3) $y = \dfrac{\sqrt{a^2 + b^2}}{x^2 - a} - 1.7\dfrac{(a^2 + b^2) \cdot 10^{-2}}{a+b} - \sin(a^2 - b^2)$

$a = 1.4; b = 27; x = 1.6$

4) $y = \dfrac{abx + tga \cdot bx}{|a-b| + 0.4x} - 10^{-2} \ln x$

$a = 1.4; b = 27; x = 1.6$

5) $y = \dfrac{(ax-b)^3 + |a-b| + e^{ax}}{(a-b)^2 + 10^3 b^2 + \ln(ax)}$

$a = 1.5; b = 2; x = 4.7$

6) $y = \dfrac{\arcsin(a-b)}{2.5x} + \sqrt{x} - \dfrac{(a-b)^2}{x}$

$a = 4.5; b = 2.5; x = 4.7$

7) $y = \dfrac{a^3 x - b|d|}{|d| + bc} - 10^{-2}\cos^2 x + \sqrt{a^3 x}$

$a = 5.2; b = 3.1; c = -1.5; d = 0.07; x = 0.5$

8) $y = 1.2\dfrac{(a-b)^3 - \dfrac{c}{x^2}}{(|a|-b)(a-b)}e^{x^2} + 10^{-1}tg(a-b)$

$a = 7.4; b = 4.5; c = -0.75; x = 1.5$

9) $y = 1.1\dfrac{\sqrt{(a+b)^3 + (b-c)^2}}{ax^3 + |b-c|} - 3^{-2}ad^2\sin^2(b-c)$

$a = 0.07; b = 1.7; c = 2.6; d = 0.27; x = -0.72$

10) $y = 3.1\dfrac{\sqrt{ac^2} - |a+b|}{|a|+|b|} + \ln(\sin^2 x)$

$a = 4.5; b = -1.7; c = 2.74; x = 1.57$

11) $y = 2.7\dfrac{\sqrt{ax+cd}}{(x+a)^2} - e^{-3x}\sin(ax^2 + cd) + e^{(x+a)}$

$a = 5.5; d = 5.25; c = -0.2; x = 7.2$

12) $y = 4.1\dfrac{a^3x - |b|}{(a+b)^2} - tg\dfrac{a+b}{(b-c)^2} - e^{(x+b)}$

$a = 2.4; b = -3.2; c = 5.7; x = 0.75$

13) $y = 0.5\dfrac{a^2x + |d|}{a + \ln b} - \sqrt{\dfrac{a^3x^2}{a-b}} - \cos(a-b)$

$a = 4.7; b = 2.4; d = -0.01; x = 3.5$

14) $y = 10^{-2}\dfrac{\sqrt{c-d} + |a+b|}{b^2 + c^2x} + e^{(c-d)} + 4.1tg(c-d)^2$

$a = 2.2; b = -18; c = 7.7; d = 4.5; x = 0.12$

15) $y = \dfrac{a^2x + |d|}{(b-c)^2 - a^2 + x} - 2.5\ln(c-b) + 10^{-2}\dfrac{(b-c)^2}{\sin x}$

$a = 4.5; b = 2.5; c = 3.7; d = -1.7; x = -1.2$

16) $y = 1.7\dfrac{|a+b|-|c-d|}{bx^2+c^2d} - \sqrt{\dfrac{(a+b)^2}{c-d}} - 10^{-2}\cos^2(c-d)$

$a = -4.5; b = 3.7; c = 7.4; d = 1.5; x = -0.7$

17) $y = 2.5\dfrac{axd^2}{2} + 10^{-1}\dfrac{\sqrt{x-a}}{ax} - \cos^3(x^2-a^2)$

$a = 3.4; d = 1.2; x = 7.5$

18) $y = 3.2\dfrac{tg(x-a)}{(x-a)^2} - 10^{-2}(x-a) + tgx^2 \cdot e^{(x-a)}$

$a = 5.1; x = 7.7$

19) $y = 1.5\dfrac{|bx|-\sqrt{acd}}{\lg|acd|} \cdot e^{acd} - 10^{-3}\arcsin^2(bx)$

$a = 3.2; b = 1.7; c = 2; d = 3.7; x = 7.1$

20) $y = \sqrt{cx} - 2.7\dfrac{|c|+|x|}{c^2x^2} \cdot e^{cx} + \cos\dfrac{(a+b)^2}{cx-b}$

$a = 3.7; b = 0.07; c = 1.5; x = 5.75$

21) $y = 4.5\dfrac{(a+b)^2}{(a-b)^2} - \sqrt{(a+b)(a-b)} + 10^{-1}\dfrac{\ln(a-b)}{\ln(a+b)} \cdot e^{x^2}$

$a = 7.5; b = 1.2; x = 0.5$

22) $y = 2.4\left|\dfrac{x^2+b}{a}\right| + (a-b)\sin^2(a-b) + 10^{-2}(x-b)$

$a = 5.1; b = 0.7; x = -0.05$

23) $y = \dfrac{ax-\sqrt{b}}{5.7(x^2+b^2)} - \dfrac{|x+b|-a^2}{x^2}tg^2b$

$a = 0.1; b = 2.4; x = -0.3$

24) $y = \sqrt{\frac{c - dx^2}{x}} + \frac{\ln(x^2 + c)}{0.7x + ad} - \frac{10^{-2}}{c - dx^3}$

$a = 4.5; c = 7.4; d = -2.1; x = 0.15$

25) $y = \arcsin\frac{x^2}{a} - \frac{ax^3 - \sqrt{b}}{\sqrt{a + b^2}} + 0.05\frac{e^{x^2}}{x^2}$

$a = 2.1; b = 3.12; x = 1.14$