Электронный учебник
Visual Basic for Application

Уважаемые студенты! Данный учебник поможет овладеть вам программированием на Visual Basic for Application. Он содержит информацию о типах данных, математических операциях, линейных разветвляющихся, циклических структурах алгоритмов и массивах. Здесь же приведен ряд задач и их решение с пояснениями.

Успехов!

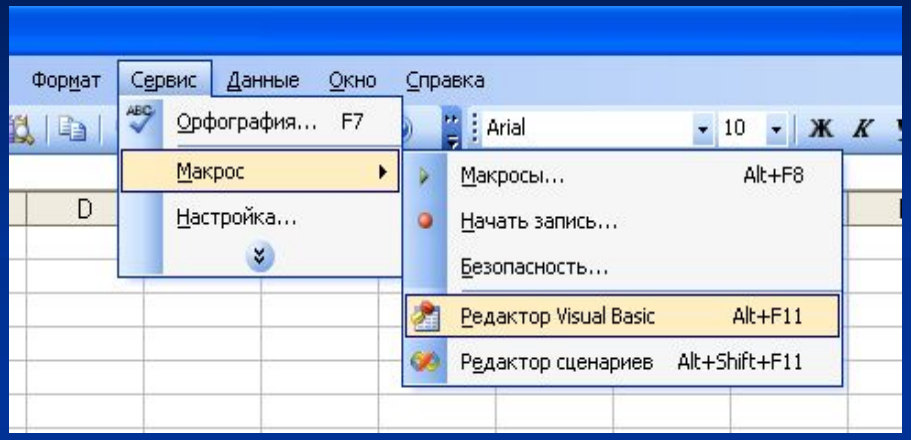
Содержание:

Вперед

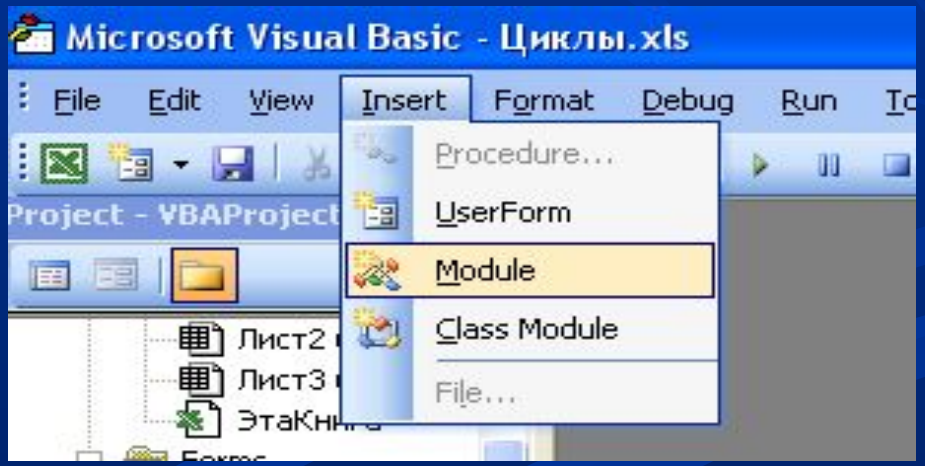
- ✔ Разветвляющаяся структура
  - Операции сравнения
  - Теория
  - Варианты задач
- ✔ Циклическая структура
  - Теория
  - Варианты задач
- ✔ Массивы
  - Теория
  - Варианты задач
- ✔ Создание пользовательской формы

Знакомство с VBA

Итак. Начнем с азов. Вы заходите в Excel, далее находите на панели управления *Сервис/Макросы/Редактор VBA*.



В появившемся окне на панели управления выбираем *Вставка/Модуль* (*Insert/Module*).



На главную

Вперед

Типы данных

В Visual Basic, как и во всех языках программирования высокого уровня, для хранения значений используются переменные и константы. Переменные могут содержать данные любых поддерживаемых типов.

Фундаментальные типы данных, поддерживаемые Visual Basic:

Тип данных	Описание	Диапазон
Byte	1-байтовое двоичное число	от 0 до 255
Integer	2-байтовое целое	от -32 768 до 32 767
Long	4-байтовое целое	от -2 147 483 648 до 2 147 483 647

На главную

Вперед

Тип данных	Описание	Диапазон
Single	4-байтовое число с плавающей точкой	от -3.402823E38 до -1.401298E-45 (отрицательные значения) от 1.401298E-45 до 3.402823E38 (положительные значения)
Double	8-байтовое число с плавающей точкой	от -1.7976931348623 1E308 до -4.940656458412 47E-324 (отрицательные значения) от 4.940656458412 47E-324 до 1.7976931348623 1E308 (положительные значения)
Currency	8-байтовое число с фиксированной десятичной точкой	от -922 337 203 685 477.5808 до 922 337 203 685 477.5807
String	строка символов	от нуля до почти двух миллиардов символов

Назад

Вперед

Тип данных	Описание	Диапазон
Variant	дата/время. число с плавающей точкой, целое, строка или объект; занимает 16 байтов плюс по 1 байту на каждый символ, если значением является строка	латы: от 1 января 100 года до 31 декабря 9999 года числовые значения: тот же диапазон, что и для Double строки: тот же диапазон, что и для String позволяет также хранить значения Error или Null
Boolean	2 байта	True или False
Date	8-байтовое значение даты/времени	от 1 января 100 года до 31 декабря 9999 года
Object	4 байта	ссылка на любой объект

Переменные

Имя переменной должно начинаться с буквы, за которой может следовать любая комбинация цифр и букв с символом подчеркивания, длиной не более 255 символов. Имена переменных в VBA не чувствительны к регистру букв, т.е. не имеет значения набрана ли буква в верхнем или нижнем регистре. В именах допускается кириллица.

Переменные в VBA создаются:

неявным их объявлением, когда ее имя появляется в выражении;

явным объявлением с помощью инструкции ***Dim*** со следующим синтаксисом

Dim <список имен переменных>.

На главную

Вперед

Все переменные, созданные таким способом, получают тип Variant. Чтобы запретить неявное объявление переменных, в начале модуля следует написать инструкцию ***Option Explicit****. Можно также запретить неявное описание переменных для всех модулей, установив флажок* ***Require Variable Declaration*** *в диалоговом окне* ***Options*** *вкладки* ***Editor****.*

Тип переменной определяется двумя способами:

1) с помощью инструкции

Dim < имя переменной> As <тип переменной>;

2) добавлением в конце имени специального символа определения типа:

! – тип Single;

@ – тип Currency;

 # – тип Double;

$ – тип String;

% – тип Integer;

& – тип Long.

Константы

Различают непоименованные и поименованные константы. Непоименованные константы появляются в тесте программы непосредственным указанием некоторого значения. В числовых константах целая часть от дробной отделяется десятичной точкой. Допускается экспоненциальная форма записи числовых констант, например запись 6.1435E2 определяет число 614.35. Строковые константы заключаются в кавычки, например, "Это строковая константа". Константы даты обрамляются знаками #, например, #18/10/04#. Введены две логические константы – True и False. Поименованные константы объявляются либо с явным указанием типа, либо без явного указания типа, соответственно инструкциями:

Const *<имя>* ***As*** *<тип>* = *<значение>*, ***Const*** *<имя>* = *<значение>*.

На главную

Вперед

Имеется также большой список предопределенных, или, как говорят, внутренних констант. Характерным для них является способ объявления имен констант, а именно:

- имена констант VBA начинается с букв vb;
- имена констант Excel начинается с букв xl;
- имена констант Word начинается с букв wd.

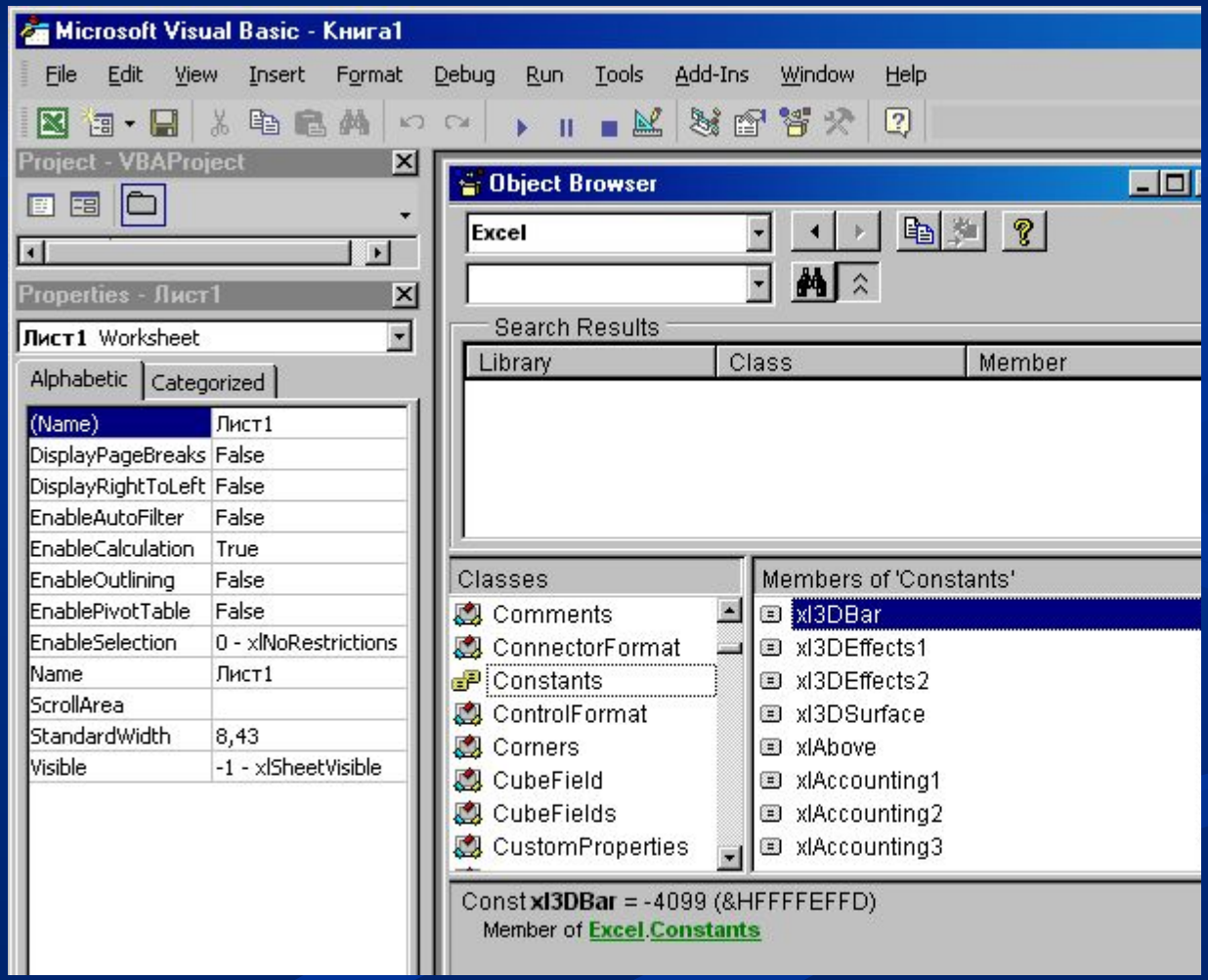
Доступ к внутренним константам осуществляется с помощью окна Object Browser по следующему алгоритму:

1. Вызовите редактор Visual Basic, нажав комбинацию клавиш *Alt+F11*.
2. Выберите в меню команду *View/Object Browser* или воспользуйтесь кнопкой *Object Browser* на панели инструментов. Редактор Visual Basic откроет одноименное диалоговое окно.

3. В списке *Project/Library* выберите библиотеку, со списком констант, например *Excel*.

4. В списке *Classes* выберите *Constants*.

5. В списке *Members of 'Constants'* выберите требуемую константу, ее имя появится в нижней части окна *Object Browser*. Дополнительную информацию о константе можно получить, щелкнув на кнопке *Help*



Назад

На главную

Математические функции

VBA содержит стандартный набор математических функций, перечень которых можно найти в окне *Object Browser*

- **Abs(x) - абсолютное значение x;**
- **Atn(x As Double) As Double - arctg(x) - в радианах;**
- **Cos(x As Double) As Double - косинус угла x;**
- **Exp(x As Double) As Double - ex;**
- **Fix(x) - возвращает целую часть числа *x*. Если x < 0, то Fix(x) > x;**
- **Int(x) - возвращает целую часть числа x. Если x < 0, то Fix(x) < x;**
- **Log(x As Double) As Double - ln(x);**
- **Rnd([x]) As Single - возвращает случайное число. Используется только после инициализации генератора случайных чисел: Sub Randomize ([Number]);**
- **Round(x, [d]) – округление числа x до *d* десятичных знаков. Если аргумент *d* опущен, производится округление x до целого числа;**

Sgn(x) – функция знака: sgn(x) = 0,если x = 0,
+1,если x>0,
-1 еслих<0.

- **Sin(x As Double) As Double – синус угла x;**
- **Sqr(x As Double) As Double - √x ;**
- **Tan(x As Double) As Double – tg(x).**

На главную

Функции преобразования данных

Часто при расчетах мы сталкиваемся с проблемой, когда нам нужно использовать вещественную переменную, которая до этого была описана как целая. Для этого используется функция конвертации. Из списка Conversion VBA выделим функции, преобразующие значение выражения x к типу, указанному после ключевого слова *As:*

- **Cbool(x) As Boolean;**
- **Cbyte(x) As Byte;**
- **Ccur(Expression) As Currency;**
- **Cdate(Expression) As Date;**
- **CDbl(Expression) As Double;**
- **Clnt(Expression) As Integer;**
- **CLng(Expression) As Long;**
- **CSng(Expression) As Single;**
- **CStr(Expression) As String.**

На главную

Функции взаимодействия с пользователем

Функции обмена данными с пользователем описаны в классе Interaction в окне *Object Browser*. Мы рассмотрим две из них (в квадратных скобках указаны необязательные аргументы)

- InputBox(Prompt, [Title], [Default], [XPos], [YPos], [HelpFile], [Context]) As String – ввод данных. Здесь обязательным аргументом является только *Prompt* – строка, используемая для подсказки о вводимой информации. Остальные аргументы могут быть опущены, так как не являются обязательными. Назначение их следующее:

  Title – заголовок диалогового окна для ввода информации;
  Default – значение сроки ввода по умолчанию;
  XPos, YPos – координаты левого верхнего угла диалогового окна, соответственно горизонтальное и вертикальные расстояния в твипах. Один твип равен 1/20 точки, которая составляет 1/72 дюйма;
  HelpFile – имя файла помощи в операционной системе Windows;
  Context – числовое выражение определяющее тематический раздел в файле помощи;

- MsgBox(Prompt, [Buttons As VbMsgBoxStyle = vbOKOnly], [Title], [HelpFile], [Context]) As VbMsgBoxResult – вывод данных. Здесь обязательным аргументом является только *Prompt* – строка выводимой информации. Остальные аргументы могут быть опущены, так как не являются обязательными. Назначение *Title, HelpFile* и *Context*– те же, что и в функции InputBox. Аргумент *Buttons* является числовым выражением, определяющим тип кнопок в диалоговом окне вывода информации. По умолчанию значение *vbOKOnly* определяет активную кнопку *ОК*.

Назад

На главную

Пользовательские функции

Функция – это своеобразная подпрограмма, которая возвращает значение. Этим и только этим функция отличается от обычной программы. Общий синтаксис функции:

```
Function <имяфункции>(переменные вводимые в функцию)
имяфункции= действия над переменными
End Function
```

Что значит, возвращает значение? Допустим у нас есть функция **КОРЕНЬ**() для вычисления квадратного корня. В скобки мы вносим число из которого хотим извлечь корень и присваиваем функции переменной с.

```
C=КОРЕНЬ(16)
```

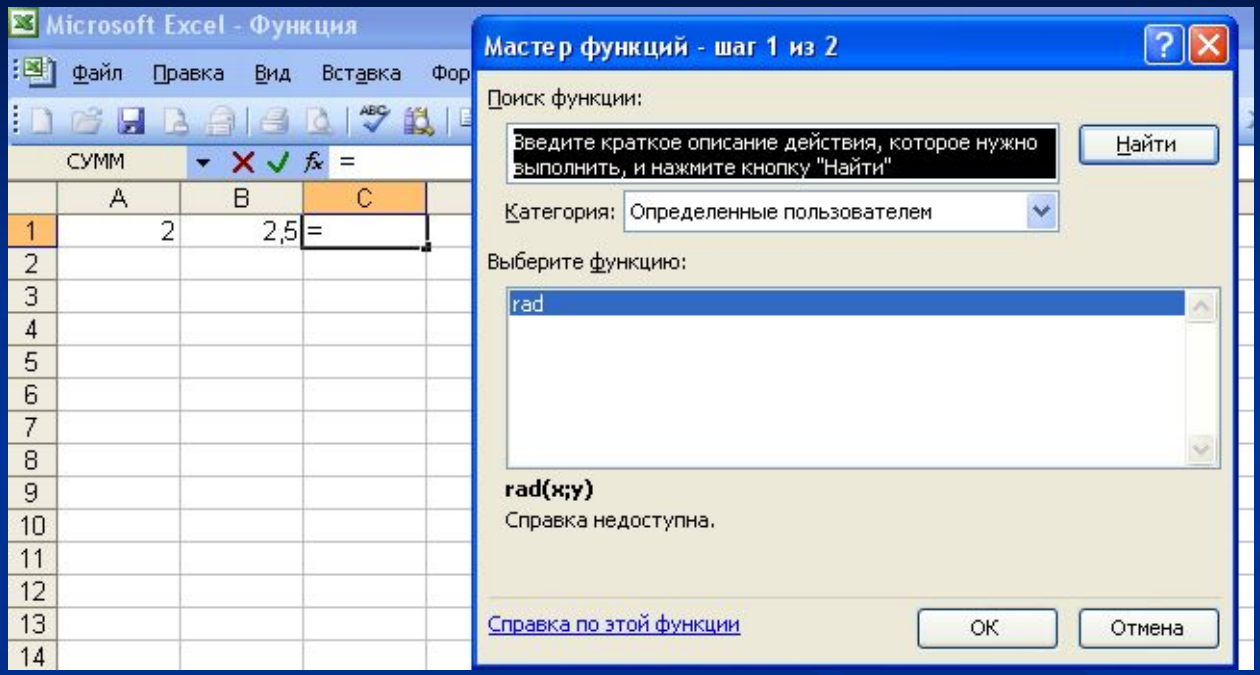
Функция извлечет корень из 16 и присвоит (вернет) переменной с значение 4. Написав функцию на VBA ее можно использовать в Excel. Вспомним задачу из предыдущего раздела про вычисление площади окружности, если радиус задан радиус-вектором. Записанная на VBA функция будет выглядеть так:

```
Function rad(x,y)
rad = 3.14*(x^2+y^2)
End Function.
```

На главную

Вперед

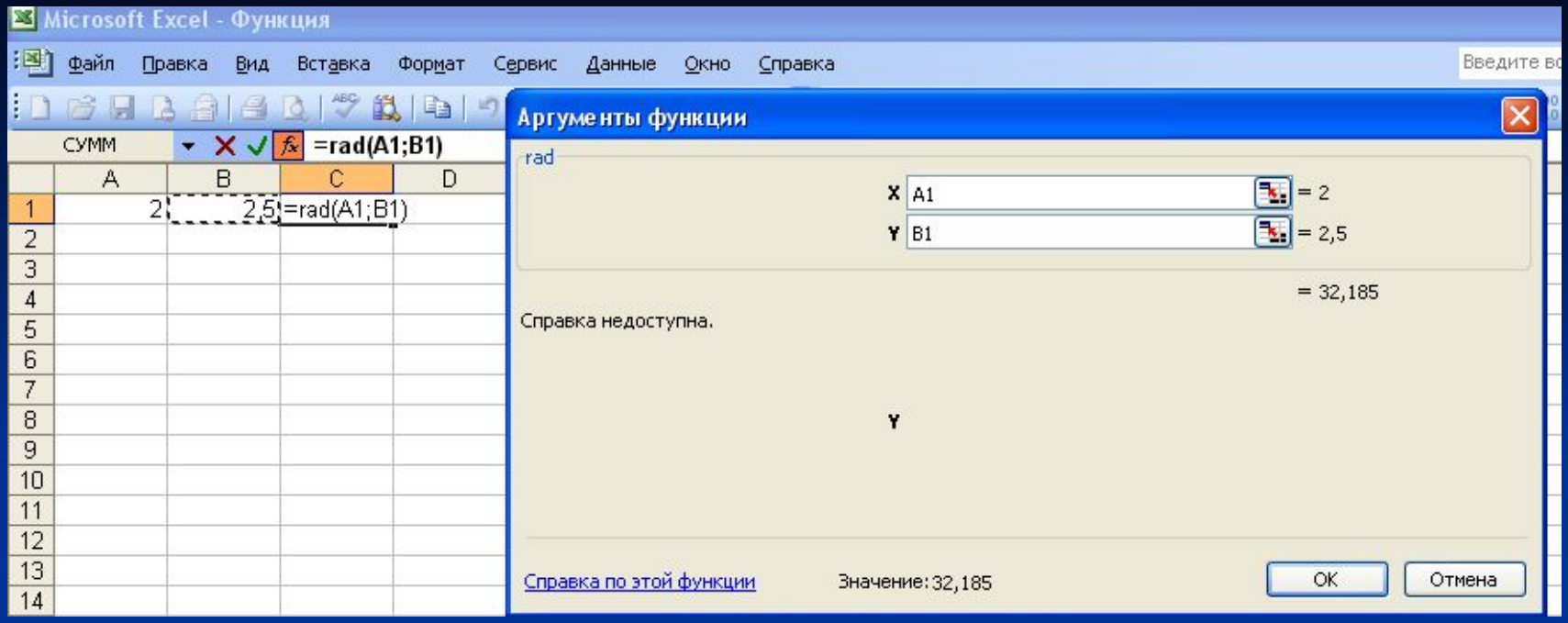
В Excel в любые две ячейки вносим необходимые нам значения x и y. В свободной ячейке пишем “=” и в категории *определенные пользователем* вызываем функцию.



После нажатия кнопки ОК выскочит окно предлагающее ввести данные в функцию. Можно щелкнуть курсором мыши по ячейке A1, потом переключиться на другое поле и щелкнуть по ячейке B1.

Назад

Вперед



Нажимаем кнопку ОК и в ячейке С1 получаем результат. Пользоваться функциями очень удобно. Несмотря на большое количество уже существующих функций, всегда лучше написать свою собственную. Для закрепления этого раздела предлагаем вам написать пользовательскую функцию для задачи о нахождении площади прямоугольного треугольника по его катетам.

Назад

На главную

Арифметические операции

Арифметические операции позволяют выполнить все стандартные арифметические действия:

\+ – операция сложения, например, *A1+A2* складывает *A1* и *A2*;

– – операция вычитания, например, *A1–A2* вычитает *A2* из *A1*;

– – унарный минус, например, *–A1* изменяет знак у *A1*;

* – операция умножения, например, *A1*A2* умножает *A1* и *A2*;

/ – операция деления, например, *A1/A2* делит *A1* на *A2*;

\ – операция целочисленного деления, например, *A1\ A2* делит *A1* на *A2*, отбрасывая дробную часть;

Mod – операция деления по модулю, например, *A1 Mod A2* делит *A1* на *A2*, возвращая остаток от деления. Остаток – целое число;

^ – операция возведения в степень, например, *A1^ A2* возводит *A1* в степень *A2*.

На главную

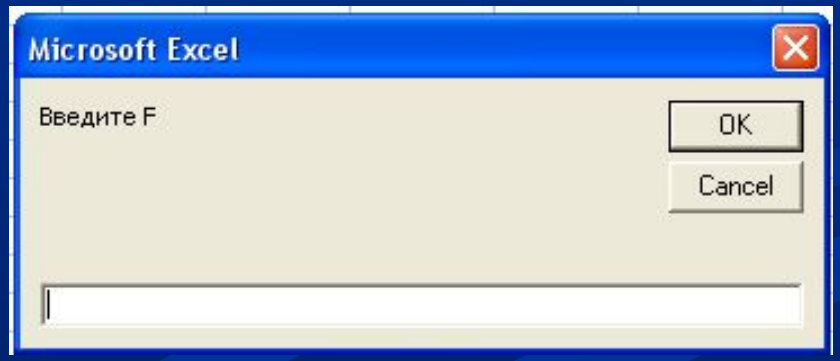
Теория

Вы уже готовы к программированию. В самой первой строке окна модуля пишем Sub <название программы> ().

В каждой программе используются переменные. Каждая переменная различного типа описывается отдельно. Переменные одного и того же типа можно описывать через запятую. Например, Dim a as integer (это значит «переменная a типа integer»). Если переменную x вы хотите описать типом Single, то придется начать со следующей строки. Кстати VBA очень ревниво следит за правильной орфографией и пунктуацией строк вашей программы, поэтому будь внимательны и осторожны, в противном случае будете получать множество сообщений об ошибках еще до запуска.

Далее. Организация ввода и вывода. Чтобы занести ваши числа в программу надо присвоить эти числа вашим переменным. Осуществляется через команду ввода inputbox(“сообщение”). Например:

F=inputbox(“введите значение F”)



На главную

Вперед

После запуска программы на выполнение перед вами выскочит окно, в котором будет написано «введите значение F». В окошко вы вводите необходимое вам число и нажимаете ОК. После чего переменной F будет присвоено численное значение.

Для вывода необходимо набрать MsgBox. Далее не надо даже скобок как в операторе ввода, просто ваша переменная. Если за переменной вы хотите вывести какое-нибудь сообщение - MsgBox m & “сообщение”.

Не стоит забывать, что VBA тесно связан с MS Excel. Ввод и вывод можно осуществлять непосредственно с листов вашей рабочей книги. Рассмотрим на примерах метод Range:

X=Worksheets(“Лист1”).Range(“B1”).Value – присваиваем переменной X значение ячейки B1 листа Лист1.

Worksheets(“Лист1”).Range(“B1”).Value=X – выводим в ячейку B1 листа Лист1 значение переменной X.

Worksheets(“Лист1”).Range(“C1”,”D6”).Value=2 – выводим в ячейки C1 и D1 листа Лист1 число 2

Worksheets(“Лист1”).Range(“B7:C9”).Value=2 – выводим в диапазон ячеек “B7:C9” листа Лист1 число 3.

Метод Cells несколько отличается от метода Range. Необходимая ячейка определяется по номеру своей строки и столбца, т. е. вместо букв A, B, C и т. д. используются цифры 1, 2, 3 и т. д.

X=Worksheets(1).Cells(3,2).Value – переменной X присваивается значение ячейки из третьей строки и второго столбца (B3) первого листа.

Переменная получает свое значение применением к ней оператора присваивания, который обозначается символом равно "=". В общем случае синтаксис следующий

<переменная> = <выражение>.

Действие оператора присваивания состоит в том, что переменная слева от знака равенства получает значение выражения справа. Оператор присваивания требует согласования типа выражения с типом переменной, записанной слева от него. Например, для получения данных от пользователя используются функция ввода данных InputBox, которая возвращает строку ввода. Если требуется числовой ввод от пользователя, то применяют функцию преобразования, например, *CSng*.

В линейной структуре все операторы исполняются друг за другом, строка за строкой. Никогда не уклоняясь от последовательности. Рассмотрим пример решения задачи.

Пример 1. *Составьте программу на языке VBA для вычисления площади прямоугольного треугольника по двум катетам.*

Входные (вводные) данные, как понятно из условия задачи, это длины двух катетов. Формула для вычисления площади прямоугольного треугольника (0,5*a*b), где a и b – катеты треугольника. Наша программа будет иметь следующий вид:

1. Название программы
2. Описание переменных a, b, s (где s – это площадь треугольника)
3. Ввод переменных a и b
4. Нахождение s по формуле s=0.5*a*b
5. Вывод “площадь треугольника равна” & s
6. Конец программы

В VBA эта запись будет выглядеть так:

```
Sub z1_v3()
Dim a, b, s As Single
a = InputBox("введите a")
b = InputBox("введите b")
s = 0.5 * a * b
MsgBox " площадь треугольника равна  " & s
End Sub
```

Назад

Вперед

Пример 2. *Составьте программу на языке VBA для вычисления площади круга по радиусу, если радиус задан вектором.*

Логическая запись:

1. Название программы
2. Описание переменных x, y, s (где s – это площадь круга; x, y – координаты радиус-вектора)
3. Ввод переменных x и y
4. Нахождение s по формуле s= 3.14*(x^2+y^2) (значок ^ - обозначает возведение в степень)
5. Вывод “площадь круга равна” & s
6. Конец программы

Запись на VBA:

```
Sub zad1_v4()
Dim x, y, s As Double
x = InputBox("введите x")
y = InputBox("введите y")
s = 3,14 * (x ^ 2 + y ^ 2)
MsgBox s & " –площадь круга"
End Sub.
```

Назад

На главную

Варианты задач

Составьте блок-схему и программу на VBA для задач:

1. Вычислить значения функции

f(x,y,z) = (x^2 – y^2)/(1 + z + x^2)

Примечание. ^ означает степень.

Исходные данные Результат

x=1 y=2 z=3.9 f(x,y,z) = -0.50847457627

2. Вычислить значения функции

f(x,y,z) = (x + y + z)/(x^2 + y^2 + z^2)

Примечание. ^ означает степень.

Исходные данные Результат

x=5 y=3.3 z=2.6 f(x,y,z) = 0.25556858148

3. Вычислить значения функции

f(x,y) = x/(1 + y) + y/(1 + x) + 1/(x + y)

Исходные данные Результат

x=4 y=3.2 f(x,y) = 1.7312698413

На главную

Вперед

4. Вычислить площадь куба по его стороне.

Исходные данные	Результат
a = 3	S = 54

5. Вычислить значения функции

f(x,y,z) = (x + y + z)/(xyz)

Исходные данные	Результат
x=7.1 y=1 z=2	f(x,y,z) = 0.71126760563

6. Вычислить вектор по двум его концам.

Исходные данные	Результат
x1=2 y1=4 x2=3 y2=-1	Вектор равен (1,-5)

7. Вычислить значения функции

f(a,b,c,x) = ax^2 + bx + c

Примечание. ^ означает степень.

Исходные данные	Результат
a=2.4 b=0.7 c=9.1 x=3	f(a,b,c,x) = 32.8

Назад

Вперед

8. Вычислить процентное отношение двух чисел (сколько процентов составляет величина первого от величины второго).

Исходные данные	Результат
Первое число 5.7	
Второе число 8.7	65.517241379%

9. Вычислить значения функции

f(x,y,z) = x/y/z + z/y/x + y/x/z

Исходные данные	Результат
x=5.9 y=2.1 z=2.8	f(x,y,z) = 1.3565087052

10. Вычислить дискриминант квадратного уравнения.

Исходные данные	Результат
a=5.9 b=8.9 c=0.7	Дискриминант D = 62.69

11. Вычислить значения функции

f(x,y) = (x + y)(x^2 + y^2)(x^3 + y^3)

Примечание. ^ означает степень.

Исходные данные	Результат
x=1.9 y= 5.4	f(x,y) = 39309.512383

12. Вычислить объем шара по заданному радиусу.

Исходные данные Результат

r = 5.78 Объем шара V = 808.85770076

13. Вычислить объем цилиндра по заданным радиусу и высоте.

Исходные данные Результат

r=5.1 h=1.9 Объем цилиндра V = 155.25436735

14. Вычислить значения функции

f(x,y,z) = (xyz)/(x + y^2 + z^3)

Примечание. ^ означает степень.

Исходные данные Результат

x=1.1 y=2 z=4.6 f(x,y,z) = 0.09879339295

15. Вычислить длину главной диагонали параллелепипеда.

Исходные данные Результат

a=4 b=3 c=6 Длина диагонали L = 7.8102496759

Назад

На главную

Операции сравнения

Операции сравнения иногда называют также *операциями отношения*. Результат операции сравнения имеет тип *Boolean*:

= – равно. Синтаксис: *E1=E2*. True, если E1 равно E2, False – в противном случае;

< – меньше. Синтаксис: *E1<E2*. True, если E1 меньше чем E2, False – в противном случае;

<= – меньше или равно. Синтаксис: *E1 <= E2*. True, если E1 меньше чем E2 или рано E2, False – в противном случае;

> – больше. Синтаксис: *E1>E2*. True, если E1 больше чем E2, False – в противном случае;

<= – больше или равно. Синтаксис: *E1 >= E2*. True, если E1 больше чем E2 или рано E2, False – в противном случае;

<> – не равно. Синтаксис: *E1<>E2*. True, если E1 не равно E2, False – в противном случае;

На главную

Теория

По сравнению с линейной структурой, разветвляющаяся предоставляет большую свободу действий. Программе можно предоставить выбор, какой следующий оператор выполнить. Для этого формулируется логическое условие, по которому и происходит выбор. Если условие даёт истинный результат, то выполняется один оператор, иначе выполняется другой.

Оператор ветвления записывается следующим образом:

```
If  условие Then
   Операторы
Else: операторы
EndIf
```

Например, если x равен нулю, то прибавить к x десять, иначе прибавить один

```
If x = 0 Then
   x = x + 10
Else: x = x + 1
EndIf
```

Часто решение определенной задачи требует соблюдение нескольких условий сразу, либо одного из нескольких. Для этого используются логические операции, основные из которых *конъюнкция* и *дизъюнкция*.

Конъюнкция. Разберем для начала жизненный пример:

Если *на улице тепло* **и** *солнечно*

то *пойду гулять*

иначе *останусь дома.*

Не трудно заметить, что при не соблюдении хотя бы одного из условий все выражение становиться ложным. Если несколько условий объединены словом *и* (в VBA это слово ***And***), выражение истинно только в том случае, когда истины все условия. ***Дизъюнкция***. Представьте себе выражение, составленное из множества условий. Если хотя бы одно из них - верно, то и все выражение верно. Это и есть явление дизъюнкции. Условия в дизъюнкции соединяются с помощью служебного слова *или* (в VBA *Or*).

Теперь поговорим немного о синтаксисе разветвляющейся структуры. Существуют так называемые одно- и многовариантные процессы. После вашего условия вы можете указать, что надо делать программе, если оно истинно, но при этом умолчать, что ей делать, когда оно ложно. Это и есть одновариантный процесс. Записывается он следующим образом:

If <условие> Then <оператор>

Подобная форма записи допустима лишь в том случае, если у вас предусмотрено выполнение только одного оператор, если таких операторов несколько, то запись будет выглядеть вот так:

If<условие>Then
<оператор1>
<оператор2>
<операторN>
EndIf

Многовариантные процессы состоят из двух ветвей. Пример такого процесса вы могли видеть в самом начале данной главы. Рассмотрим теперь решение нескольких задач.

Пример 1. *Составьте программу на языке VBA для задачи: Определить, попадает ли точка (x,y) в круг с радиусом R, лежит на окружности с тем же радиусом или вне ее. Ввести радиус и точку.*

Из условия задачи понятно, что в программе нам надо использовать три переменные и несколько условий. Заметим, что здесь удобно воспользоваться так называемым вложенным ветвлением, то есть когда вместо оператора идет еще одна разветвляющаяся структура. Решение будет выглядеть так:

```
Function okr(x, y, r)
If (y ^ 2 + x ^ 2) > r ^ 2 Then
okr= "точка лежит вне круга"
ElseIf (y ^ 2 + x ^ 2) < r ^ 2 Then
okr= "точка лежит в круге"
Else
okr= "точка лежит на окружности"
End If
End Function
```

Отметим, что при написании программы мы сделали только две проверки: «точка лежит вне круга» и «точка лежит в круге». В случае если первые две проверки ложны, то третья проверка становится просто бессмысленной и можно сразу выводить сообщение о том, что точка находится на окружности.

Пример 2. *Составьте программу на языке VBA для задачи: Решить квадратное уравнение, заданное своими коэффициентами. При D < 0 вывести 'Нет действительных корней'.*

Принцип написания программы такой же, как и в предыдущей задаче. Если дискриминант не больше нуля и не равен нулю, тогда выводим сообщение «действительных корней нет». Решение:

```
Function disk(a, b, c)
Dim d, x1, x2 As Double
d = b * b - 4 * a * c
If d > 0 Then
x1 = (-b - Sqr(d)) / (2 * a)
x2 = (-b + Sqr(d)) / (2 * a)
disk=”X1=” & x1 & ",X2=" &  x2
ElseIf d = 0 Then
x1 = -b / (2 * a)
disk=”X1=” & x1
Else
disk= "нет действительных корней"
End If
End Function
```

Назад

На главную

Варианты задач

Составить блок-схему и программу наVBA для задачи:

1. Сравнить три введенных числа, наибольшее и наименьшее вывести.

Исходные данные	Результат
1. a=5 b=7 c=3	max = 7 min = 3
2. a=9 b=-1 c=4	max = 9 min = -1
3. a=0 b=0 c=19	max = 19 min = 0

2. Ввести три числа A,B,C и проверить, могут ли являться эти числа значениями углов треугольника, и если да, то является ли этот треугольник прямоугольным.

Исходные данные	Результат
1. A=70 B=90 C=20	A,B,C являются углами треугольника Треугольник прямоугольный
2. A=35 B=80 C=65	A,B,C являются углами треугольника
3. A=90 B=-70 C=160	A,B,C не являются углами треугольника

На главную

Вперед

3. Ввести число X и проверить, что больше: sin(X), cos(X) или tg(X). X исчисляется в радианах.

	Исходные данные	Результат
1.	x=2.9	sin(x) больше
2.	x=0.5	cos(x) больше
3.	x=1.5	tg(x) больше

Назад

Вперед

4. Определить, принадлежит ли точка (x,y):

1) прямой $ax + by + c = 0$

2) прямой $ax + 2by + 3c = 0$

Исходные данные	Результат
1. a=2 b=3 c=1 x=2.5 y=-2	Точка (2.5,-2) принадлежит первой прямой Точка (2.5,-2) не принадлежит второй прямой
2. a=2 b=3 c=1 x=3 y=-1.5	Точка (3,-1.5) не принадлежит первой прямой Точка (3,-1.5) принадлежит второй прямой
3. a=2 b=4 c=1 x=0.5 y=-0.5	Точка (0.5,-0.5) принадлежит первой прямой Точка (0.5,-0.5) принадлежит второй прямой
4. a=2 b=4 c=1 x=0 y=0	Точка (0,0) не принадлежит первой прямой Точка (0,0) не принадлежит второй прямой

Назад

Вперед

5. Определить, какая из двух точек (a,b) и (c,d) больше удалена от центра координат. Если они находятся на равном расстоянии, сообщить об этом.

Исходные данные	Результат
1. a=2 b=1 c=1 d=3	(1,3) дальше (2,1)
2. a=1 b=6 c=2 d=2	(1,6) дальше (2,2)
3. a=2 b=0 c=0 d=2	(2,0) и (0,2) на равном расстоянии

6. Определить, является ли X n-ным членом арифметической прогрессии, если ее первый член равен A, а разность - D. A n-ным членом арифметической прогрессии с разностью 2D?

Исходные данные	Результат
. n=3 A=5 D=6 X=17	X является 3-им членом первой ариф. прогр. X не является 3-им членом второй ариф. прогр.
2. n=3 A=5 D=6 X=29	X не является 3-им членом первой ариф. прогр. X является 3-им членом второй ариф. прогр.
3. n=3 A=5 D=6 X=23	X не является 3-им членом первой ариф. прогр. X не является 3-им членом второй ариф. прогр.

Назад

Вперед

7. Ввести X. Вычислить F(X) по формуле:

$$F(X) = \begin{cases} X/2, & \text{если } X \text{ - четное} \\ (X + 1)/2, & \text{если } X \text{ - нечетное} \end{cases}$$

Получившееся число (т.е. F(X)) проверить на четность. Вывести X, F(X) и результаты проверки F(X) на четность.

Исходные данные	Результат
1. x = 8	X = 8 F = 4 F четное
2. x = 9	X = 9 F = 5 F нечетное
3. x = 11	X = 11 F = 6 F четное

8. Определить корень квадратный из числа X (с проверкой на допустимость его извлечения!), и вывести:

а) сам корень, если он целый

б) дробную часть корня, если он вещественный.

Исходные данные	Результат
1. x=25	Корень = 5
2. x=31	Дробная часть корня = 0.5677643628
3. x=-4	Неверные исходные данные

9. Найти сумму n первых членов геометрической прогрессии с первым членом A и знаменателем q. A должно быть ненулевым, а q - больше единицы.

Исходные данные	Результат
1. n=5 A=2 q=2	Сумма 5-ти членов равна 62
2. n=5 A=0 q=0.5	Неверные исходные данные
3. n=-2 A=3 q=1.5	Неверные исходные данные

10. Определить, в каком квадранте находится точка (x,y) и вывести номер квадранта (или определяющие его условия на координаты).

Исходные данные	Результат
1. x=3 y=1	I-ый квадрант
2. x=-3 y=1	II-ый квадрант
3. x=-3 y=-1	III-ый квадрант
4. x=3 y=-1	IV-ый квадрант

Назад

Вперед

11. Ввести 3 стороны прямоугольного треугольника A,B,C. Определить, какая из них является гипотенузой. Учесть, что введенные числа могут и не являться сторонами треугольника.

Исходные данные	Результат
1. A=3 B=4 C=5	Гипотенуза C = 5
2. A=5 B=3 C=4	Гипотенуза A = 5
3. A=4 B=5 C=3	Гипотенуза B = 5
4. A=-3 B=2 C=1	Введенные числа не явл. сторонами пр. треугольника

Назад

Вперед

12. Определить, может ли существовать треугольник со сторонами A,B,C.

Исходные данные	Результат
1. A=3 B=4 C=2	Треугольник со сторонами 3,4,2 существует
2. A=5 B=1 C=3	Треугольник со сторонами 5,1,3 не существует
3. A=-1 B=5 C=3	Неверные исходные данные

13. Вычислить значение функции:

$$f(x) = \begin{cases} \text{натуральный логарифм из } x, & \text{если } x > 0 \\ 0, & \text{если } x = 0 \\ \text{квадрат } x, & \text{если } x < 0. \end{cases}$$

Исходные данные	Результат
1. x=15	f(x) = 2.7080502011
2. x=0	f(x) = 0.0
3. x=-7	f(x) = 49.0

Назад

Вперед

14. Проверьте, возрастает или убывает прямая ax + by + c = 0.
Замечание. Любая прямая монотонна на всей оси абсцисс. Если прямая параллельна оси абсцисс (т.е. оси Ox), выдать соответствующее сообщение.

Исходные данные	Результат
1. a=5 b=3 c=7	Прямая убывает
2. a=2 b=0 c=4	Прямая параллельна оси Ox
3. a=-6 b=5 c=1	Прямая возрастает
4. a=0 b=0 c=1	Прямая не существует

15. Определить, пересекаются две прямые y = ax + b и y = cx + d, параллельны или совпадают.

Исходные данные	Результат
1. a=7 b=4 c=7 d=11	Прямые параллельны
2. a=3 b=4 c=4 d=11	Прямые пересекаются
3. a=5 b=1 c=5 d=1	Прямые совпадают

Назад

На главную

Теория

Некоторые задачи требуют при решении повторения однотипных действий. Что бы в программах одно и тоже действие не записывать отдельными операторами, применяют операторы цикла. Они являются программным представлением такой алгоритмической структуры, как итерация.

Мы рассмотрим три варианта операторов цикла – со счётчиком, с пост условием, с предусловием.

Циклы For-Next. Они же циклы со счетчиком. Используются когда вам известно конечное число повторений. Форма записи выглядит вот так:

For <имя счетчика>=<начальное значение счетчика> To <конечное значение счетчика>

<операторы>

Next <имя счетчика>

Чаще всего переменную-счетчик обозначают i. Если нам необходимо сто раз вывести сообщение «Все будет хорошо», наша программа примет следующий вид:

```
For i=1 To 100
MsgBox ‘Все будет хорошо’
Next i
```

На главную

Вперед

Вместо числа 100 мы могли поставить какую-либо переменную, единственное условие – эта переменная и счетчик должны быть описаны целочисленным типом. Кроме того, существует возможность задать шаг цикла. Например, вам надо, чтобы ваш цикл выполнялся через раз. Это можно сделать поместив в цикле лишний оператор i=i+1, а можно написать после конечного значения счетчика Step 2, где два это величина шага.

Вторая структура циклов с фиксированным числом повторений – это цикл **For Each…Next**, синтаксис которого следующий.

For Each <элемент> In <группа>

<операторы >

Next [<элемент>]

Алгоритм работы этой конструкции приспособлен для перебора всех элементов из указанной группы. Именно, он выполняется ровно столько раз, сколько есть элементов в группе.

Циклы Do While – Loop. Цикл с предусловием или неопределенный цикл. Очень похоже на одновариантное ветвление. В самом начале вы задаете условие и ваши операторы будут повторятся до тех пор пока условие истинно. Кстати если перевести с английского языка этот цикл мы получим «делать пока <условие>». Такие циклы требуют больше внимания при написании программы, чем циклы со счетчиком. Условие может быть всегда истинным, и ваши операторы будут повторяться бесконечно. Форма записи:

Do While <логическое условие>
<операторы>
Loop

Аналогично **Do While** работает цикл **Do Until**, общий синтаксис которого такой:

Do Until <логическое выражение>
    <операторы тела цикла>
Loop

Цикл **Do Until** работает как цикл с предусловием. Отличие в алгоритме работы цикла **Do Until** состоит только в том, что он завершает свою работу, когда <логическое выражение> оказалось после его вычисления истинным, т.е. приобрело значение True.

Оператор **Do…Loop While** создает цикл с постусловием. Общий синтаксис его такой:

Do

<операторы тела цикла>

Loop While <логическое выражение>

Алгоритм работы следующий. В первый раз <операторы тела цикла> выполняются до достижения ключевого слова Loop. Затем вычисляется <логическое выражение>, указанное после ключевого слова While. Если оно истинно, т.е. есть True, то происходит переход в начало цикла и повторяется выполнение операторов тела цикла.

После чего снова проверяется, сохранилось ли у <логическое выражение> значение True. Если да, то цикл повторяется снова, и так до тех пор, пока значение не изменится на False. После чего происходит выход из цикла с передачей управления инструкции, записанной после ключевого слова Loop.

Существует еще один оператор **Do…Loop Until**, который организует цикл с постусловием. Общий синтаксис его такой:

Do

<операторы тела цикла>

Loop Until <логическое выражение>

Отличие в алгоритме работы цикла **Do…Loop Until** от алгоритма работы цикла **Do…Loop While** состоит только в том, что цикл **Do…Loop Until** завершает свою работу, когда <логическое выражение> оказалось после его вычисления истинным, т.е. приобрело значение True.

Пример 1. *Составьте программу на языке VBA для задачи: Найти сумму n членов последовательности*

F(i) = sqr(F(i-1)) + 1

n и F(0) задаются при вводе.

Примечание. Указана математическая запись выражения, к VBA она неприменима.

Решение:

```
Function posl(n, f)
Dim i As Integer
Dim s As Double
If f > 0 And n > 0 Then
For i = 1 To n
f = Sqr(f) + 1
s = s + f
Next i
posl= s
Else
posl="неверные исходные данные"
End If
End Function
```

Назад

Вперед

Пример 2. *Составьте программу на языке VBA для задачи: Найти сумму n членов ряда*

m/2 + (m+1)/2^2 + (m+2)/2^3 + (m+3)/2^4...

m и n вводятся, m должно быть не равно 0, а n > 0.

Примечание. ^ означает степень.

Решение:

```
Function sum(m, n)
Dim i As Integer
Dim a, s As Double
If n > 0 And m <> 0 Then
For i = 1 To n
a = m / (2 ^ i)
s = s + a
m = m + 1
Next i
sum=s
Else
posl="неверные исходные данные"
End If
End Function
```

Назад

На главную

Варианты задач

Составить блок-схему и программу на VBA для задачи:

1. Подсчитать число положительных и отрицательных чисел во введенной последовательности из n (n задается и должно быть больше 0) целых чисел. Нулевые числа при подсчете не учитывать.

Исходные данные	Результат
1. n = -8	Неверные исходные данные
2. n = 10	Положительных чисел 5
4 0 -7 5 0 4 -1 5 0 9	Отрицательных чисел 2

2. Возвести число m в целую степень n, не используя операции возведения в степень. m и n вводятся.

Примечание. Учтите, что степень может быть отрицательной.

Исходные данные	Результат
1. m=5 n=3	5 в степени 3 равно 125
2. m=4 n=-4	4 в степени -4 равно 0.00390625
3. m=-7 n=3	-7 в степени 3 равно -343.0
4. m=-5 n=-3	-5 в степени -3 равно -0.008

На главную

Вперед

3. Найти сумму ряда

-1 + 1/2 - 1/3 + 1/4 - ....

с точностью e (это означает, что вычисления следует прекратить, когда вычисленная сумма будет отличаться от реальной не более чем на e. Или, что то же самое, когда модуль последнего вычисленного члена меньше e).

Замечание. e - величина порядка 0.00001

Исходные данные	Результат
1. e = -0.01	Неверные исходные данные
2. e = 0.0005	S = -0.69339724304

4. Найти процент положительных значений функции

f(x) = 2*sin(x) - cos(x)

среди вычисленных от a до b с шагом h. a,b,h вводятся. Введенные данные необходимо проверить на допустимость (a < b, h > 0).

Исходные данные	Результат
1. a=2 b=5 h=0	Неверные исходные данные
2. a=7 b=5 h=0.2	Неверные исходные данные
3. a=2 b=7 h=0.2	Положительных значений 42.307692308%

5. Найти сумму n членов ряда

m/2 + (m+1)/2^2 + (m+2)/2^3 + (m+3)/2^4...

m и n вводятся, m должно быть не равно 0, а n > 0.

Примечание. ^ означает степень.

Исходные данные	Результат
1. m=0 n=-5	Неверные исходные данные
2. m=6 n=10	S = 6.9833984375

6. Вычислить сумму n членов ряда 1/1! + 2/2! + 4/3! + 8/4!...,n вводится и должно быть больше 0. Примечание.m!=1*2*3*....*(m-1)*m

Исходные данные	Результат
1. n=-1	Неверные исходные данные
2. n=10	S = 3.1944973545

7. Определить, является ли число n простым или нет. n вводится и должно быть больше 0.

Замечание. Число называется простым, если оно делится только на 1 и само на себя.

Исходные данные	Результат
1. n = -1	Неверные исходные данные
2. n = 37	37 является простым числом
3. n = 39	39 не является простым числом

Назад

Вперед

8. Найти сумму цифр целого числа n. n должно быть больше 0 (проверить).
Пример. n = 257 ответ 2 + 5 + 7 = 14

Исходные данные	Результат
1. n = -1	Неверные исходные данные
2. n = 257	Сумма цифр равна 14

9. Вычислить n! (n факториал). n вводится и должно быть >= 0.
Примечание. Формула вычисления факториала: n! = 1*2*3*....*(n-1)*n. Предусмотреть особый случай n = 0, т.к. 0! = 1.

Исходные данные	Результат
1. n = -1	Неверные исходные данные
2. n = 0	Факториал 0 равен 1
3. n = 5	Факториал 5 равен 120

10. Найти сумму n членов последовательности
F(i) = (F(i-1))2 + 1
n и F(0) задаются при вводе.

Исходные данные	Результат
1. n = 7 F(0) = -7	Неверные исходные данные
2. n = -5 F(0) = 17	Неверные исходные данные
3. n=4 F(0)=4	S=7073062610

11. Найти n-ое число Фибоначчи, если первые два члена равны a1=1 и a2=1. Числа Фибоначчи находятся по правилу an=a(n-1)+a(n-1).

Исходные данные	Результат
1. n = -1	Неверные исходные данные
2. n = 8	21

12. Определить сколько счастливых билетов попадают в диапазон от m до n(m>99999, n<1000000, m<n).

Исходные данные	Результат
1. m=307 n=298766	Неверные исходные данные
2. m =287993 n=345678	3086

13. Сумма в R рублей положена в банк под P% ежегодно. Через сколько лет сумма достигнет M рублей (M>R)?

Исходные данные	Результат
1. R=100 P=50 M=225	2 года
2. R=200 P=20 M=8000	21 год

14. Имеется два сосуда. В первом находится C1 литров воды, а во втором C2 литров воды. Из первого сосуда переливают половину во второй. Потом из второго половину в первый. Это считается как одно переливание. Сколько воды окажется в каждом сосуде после K переливаний?

Исходные данные	Результат
1. C1=10 C2=8 K=1	C1=11,5 литров C2=6,5 литров
2. C1=30 C2=56 K=4	C1=57,22 литров C2=28,77 литров

15. Определите на натуральном отрезке [c ; d] число сумма цифр которого минимальна.

Исходные данные	Результат
1. c=33 d=86	40
2. c=48 d=54	50

Назад

На главную

Теория

Их ещё называют списками. Итак, что же такое массивы? Массив (вектор) - это набор однотипных переменных, объединенных одним именем и доступных через это имя и порядковый номер переменной в наборе. Количество элементов массива теоретически может быть бесконечным, ограничения накладываются конкретными языком программирования и операционной системой. Элементы массива обладают непрерывной нумерацией определённого диапазона. В программировании массивы используются довольно часто. Представьте, что вам нужно использовать тысячу переменных. Согласитесь – описывать каждую переменную напрасная трата времени и сил. Проще обратиться к массивам.
В Visual Basic массивы определяются следующим образом:

```
Dim myArray (10) As Long
```

Как вы могли заметить, определение массива отличается от определения обычной переменной только индексом, указанным в скобках. Этот индекс указывает размерность массива. В данной случае массив myArray будет содержать 11 элементов. Почему 11? Потому что нижняя граница массива начинается с нуля. [0,1,2.....9,10]. Чтобы задать определённую размерность можно использовать зарезервированное слово To:

```
Dim myArray (5 To 10) As Long
```

На главную

Вперед

Здесь определяется массив, размерность которого 6 элементов (5,6,7,8,9,10).

Итак, массив определён. Теперь необходимо узнать - как же можно добраться к элементам этого массива. Очень просто! К элементам массива нужно обращаться по индексу, к примеру, чтобы изменить нулевой элемент массива myArray нужно написать:

myArray(0)=1234.

Как вы уже, наверное, догадались заполнять заданный массив проще всего через цикл For – Next.

```
For i=1 To 10
myArray(i)=<какое-то значение>
Next i
```

Чтобы не заполнять бесконечные окна ввода можно воспользоваться методом Cells (он описан в разделе «общие сведения»). Поля со значениями элементов массива нужно заполнять заранее, до запуска программы

Пример 1. *Составьте программу на языке VBA для задачи:*
Ввести вещественный массив длины n (n > 0) и подсчитать сумму и произведение его элементов. Нули при подсчете игнорировать.

```
Sub ar1()
Dim a(100) As Single
Dim s, p As Single
Dim i, n As Integer
n = InputBox("введите n")
If n > 0 Then
For i = 1 To n
a(i) = Worksheets(1).Cells(i, 1).Value
Next i
p = 1
For i = 1 To n
If a(i) <> 0 Then
s = s + a(i)
p = p * a(i)
End If
Next i
```

Назад

Вперед

```
Worksheets(1).Cells(1, 4).Value = "сумма= " & s
Worksheets(1).Cells(2, 4).Value = "произведение= " & p
Else
MsgBox "неверные исходные данные"
End If
End Sub
```

Если вы все сделали правильно то при введение в ячейки A1:A6 при n=6. Должно получиться следующее:

E5 f_x

	A	B	C	D	E
1	2			сумма= 2,1	
2	0			произведение= -29,52	
3	1,2				
4	-4,1				
5	3				
6	0				

Назад

Вперед

Пример 2. *Составьте программу на языке VBA для задачи: Ввести целый массив длины n (n > 0) и вычислить сумму*

$F(X1, ..., Xn) = -1*X1 + 2*X2 - 3*X3 + ...$

считая, что Xi - элементы массива.

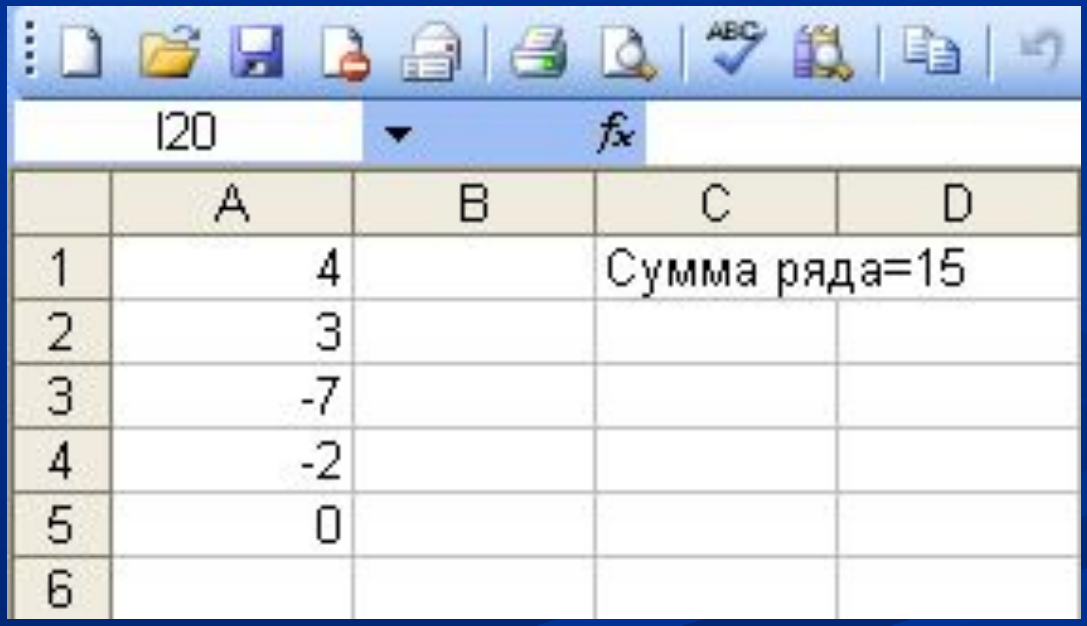
```
Sub ar2()
Dim a(100) As Integer
Dim i, n, s As Integer
n = InputBox("введите n")
If n > 0 Then
For i = 1 To n
a(i) = Worksheets(1).Cells(i, 1).Value
Next i
For i = 2 To n Step 2
s = s + i * a(i)
Next i
For i = 1 To n Step 2
s = s - i * a(i)
Next i
```

Назад

Вперед

```
Worksheets(1).Cells(1, 3).Value = "Сумма ряда равна=" & s
Else
MsgBox "Неверные исходные данные"
End If
End Sub
```

При n=5 и вводе чисел указанных на рисунке должно получиться так:



	A	B	C	D
1	4		Сумма ряда=15	
2	3			
3	-7			
4	-2			
5	0			
6				

Назад

На главную

Варианты задач

Составьте блок-схему и программу на VBA для задач:

1. Ввести целый массив длины n (n > 1) и найти самый длинный его сплошной отрезок, элементы которого являются членами арифметической прогрессии с разностью 1. Вывести длину отрезка и первый элемент.

Примечание. Минимальная длина массива - 2.

Исходные данные	Результат
1. n = -1	Неверные исходные данные
2. n = 10, массив: 2 7 8 2 3 4 5 2 0 1	Длина 4, первый элемент 2 (отрезок 2 3 4 5)

2. Ввести целый массив длины n (n > 0), найти и вывести

а) минимальный элемент и его номер

б) максимальный элемент и его номер

Исходные данные	Результат
1. n = -1	Неверные исходные данные
2. n = 9, массив: 3 -7 6 0 5 11 9 -2 7	Максимальный элемент 11 (номер 6) Минимальный элемент -7 (номер2)

На главную

Вперед

3. Ввести целый массив длины n (n > 0) и найти самую большую и самую малую (по модулю) разности между двумя его соседними элементами.

Исходные данные	Результат
1. n = -1	Неверные исходные данные
2. n = 7, массив: 1 8 2 -15 5 3	Максимальная разность 20 Минимальная разность 4

4. Ввести целый массив длины n (n > 0), расположить его элементы от меньшего к большему и вывести результат.

Исходные данные	Результат
1. n = -1	Неверные исходные данные
2. n = 7, массив: 9 3 7 1 0 2 3	0 1 2 3 3 7 9

5. Ввести вещественный массив длины n (n > 0) и подсчитать количество элементов вне, внутри и на круге с радиусом r. Координатами считать последовательные пары элементов. n должно быть четным!

Исходные данные	Результат
1. n=1 r=-2	Неверные исходные данные
2. n=-2 r=4	Неверные исходные данные
3. n=8 r=4, массив: 0.5 0 1.5 8 3 3 2 0	1 внутри, 1 на круге, 2 вне круга

6. Ввести целый массив длины n (n > 0) и, считая, что это n-мерный вектор, найти его длину.

Исходные данные Результат

1. n = -1 Неверные исходные данные
2. n = 5, массив:
   3 0 6 7 2 Длина вектора равна 9.8994949366

7. Ввести целый массив длины n (n > 0), подсчитать и вывести
   а) среднее арифметическое всех элементов
   б) элемент, наиболее близкий к числу а)

Исходные данные Результат

1. n = -1 Неверные исходные данные
2. n = 7, массив: Среднее арифметическое
   8.7142857143
   41 2 -7 9 3 7 6 Самый близкий к нему элемент 9

8. Ввести вещественный массив длины n (n > 0) и преобразовать его по следующему правилу:

n-ный эл. = ((n-1)-ый + n-ный)/2

Первый элемент не изменяется. Выдать преобразованный массив.

Исходные данные	Результат
1. n = -1	Неверные исходные данные
2. n = 7, массив: 7.4 1.8 3.5 5.3 2 0 4.1	7.4 4.6 2.65 4.4 3.65 1.0 2.05

9. Ввести целый массив длины n (n > 0) и вывести номера всех элементов, являющихся степенями двойки (проверять до 15 степени).

Исходные данные	Результат
1. n = -1	Неверные исходные данные
2. n = 9, массив: 4 1 12 16 5 128 1024 234 78	Степенями двойки являются 1,2,4,6,7 элементы

Назад

Вперед

10. Ввести целый массив длины n (n > 0) и преобразовать его так, чтобы сначала шли отрицательные элементы, потом нулевые, потом положительные. Вывести результат.

Исходные данные	Результат
1. n = -1	Неверные исходные данные
2. n = 9, массив: 1 6 0 -1 5 -7 0 5 -8	-1 -7 -8 0 0 1 6 5 5

11. Ввести целый массив длины n (n > 0) и инвертировать его (перевернуть так, чтобы первый элемент стал n, второй - (n-1) и т.д.)

Исходные данные	Результат
1. n = -1	Неверные исходные данные
2. n = 7, массив: 5 2 -7 5 0 2 1	1 2 0 5 -7 2 5

12. Ввести целый массив длины n (n > 0) подсчитать, сколько его элементов
a) делятся на 2
b) делятся на 3

Исходные данные	Результат
1. n = -1	Неверные исходные данные
2. n = 8, массив: 5 9 12 5 15 1 6 14	3 элемента делятся на 2 4 элемента делятся на 3

Назад

Вперед

13. Ввести целый массив длины n (n > 0) и вывести номера и значения всех его нечетных и ненулевых элементов, которые находятся в диапазоне -n < a[i] < n

Исходные данные	Результат
1. n = -1	Неверные исходные данные
2. n = 9, массив:	элемент 5 (номер 1)
5 -7 0 8 8 11 1 -9 15	элемент -7 (номер 2)
	элемент 1 (номер 7)

14. Ввести целый массив длины n (n > 0) и сжать его, выбросив все нулевые и отрицательные элементы.

Исходные данные	Результат
1. n = -1	Неверные исходные данные
2. n = 9, массив:	
2 -5 2 0 5 9 0 -7 54	2 2 5 9 54

15. Ввести целый массив длины n (n > 0) и определить максимальную сумму двух подряд идущих элементов. Вывести сумму и номера элементов, ее составляющих.

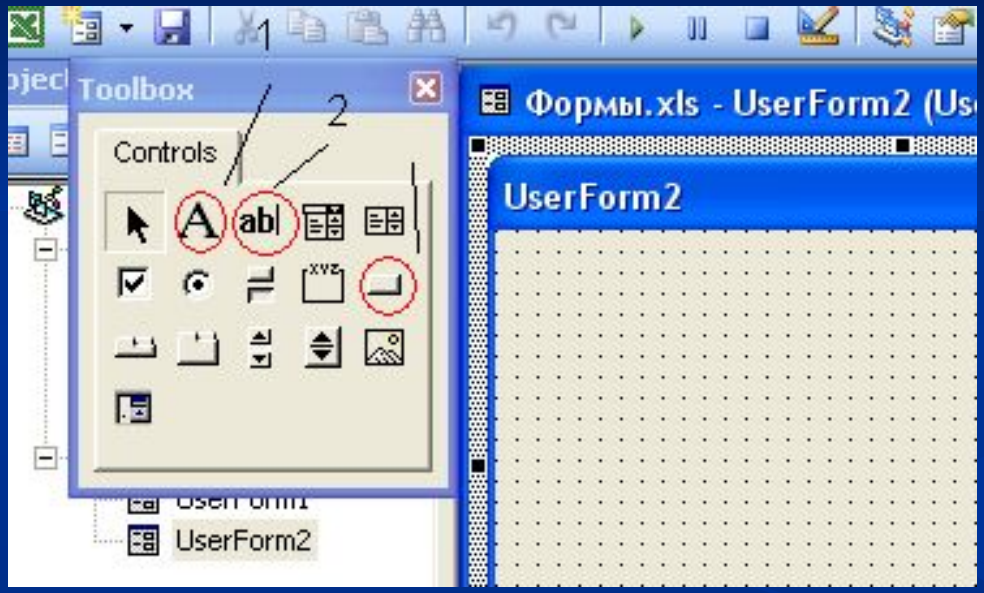
Исходные данные	Результат
1. n = -1	Неверные исходные данные
2. n = 8, массив:	
5 -2 0 2 4 8 2 1	Максимальная сумма: 12
	(5 и 6 лементы)

Назад

На главную

Создание пользовательской формы

Форма – это диалоговое окно, на котором размещены элементы управления. С помощью VBA вы сами можете создавать пользовательские формы. Новая форма добавляется выбором команды ***Insert/UserForm.***

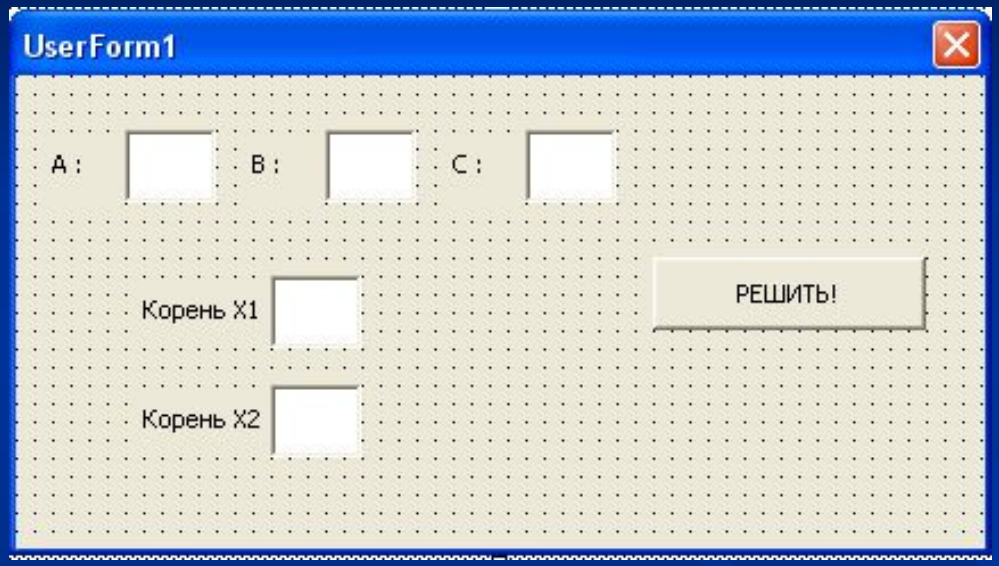


Вы можете видеть перед собой панель инструментов (Toolbox). Красными кружками обведены те, которые нам понадобятся для создания простейшей формы. Под номером один элемент управления *надпись (label)* просто поясняющая надпись и ничего больше. Под вторым номером *поле (TextBox)* в поля мы будем вводить данные и выводить результаты. И наконец под номером три *кнопка (CommandButton)* при нажатии которой будет решаться задача.

На главную

Вперед

В качестве примера разберем решение квадратного уравнения через его коэффициенты. Нам необходимо пять полей (три для коэффициентов и два для корней), пять надписей (соответственно) и одна кнопка. После размещения элементов управления и замены их имен выглядеть все будет следующим образом:



Теперь нужно написать программу, благодаря которой форма будет действовать. Для этого дважды щелкните курсором на кнопке, и перед вами появится окно для ввода программы. Мы приведем ее в готовом виде, в фигурных скобках указаны комментарии:

```
Private Sub CommandButton1_Click()
 If IsNumeric(TextBox1.Text) And IsNumeric(TextBox2.Text) And
IsNumeric(TextBox3.Text) Then
  {проверка являются ли введенные значения числами}
  a = CDbl(TextBox1.Text)
  b = CDbl(TextBox2.Text)
  c = CDbl(TextBox3.Text)
  {в поля мы вводим текст, а работать нам надо с числами поэтому применяем
функцию перевода из строкового типа в числовой}
  d = b * b - 4 * a * c
{находим дискриминант}
If d > 0 Then
TextBox4.Text = (-b - Sqr(d)) / (2 * a)
TextBox5.Text = (-b + Sqr(d)) / (2 * a)
{выводим результат}
ElseIf d = 0 Then
TextBox4.Text = -b / (2 * a)
TextBox5.Text = " "
```

Назад

Вперед

```
Else
TextBox4.Text = " "
TextBox5.Text = " "
MsgBox "нет действительных корней"
End If
Else
MsgBox "Коэффициент должен быть числом"
TextBox1.SetFocus
{устанавливаем курсор в поле 1}
End If
End Sub

Private Sub userform_initialize()
UserForm1.Caption = "Вычисление корней уравнения"
{меняем заголовок формы}
TextBox4.Enabled = False
TextBox5.Enabled = False
{делаем поля 4 и 5 недоступными для редактирования пользователем}
End Sub
```

Вот в принципе и все. Количество и комбинации элементов управления на вашей форме варьируются, что делает пользовательскую форму очень удобным подспорьем, когда нужно проводить много тестов в задаче