

ОСНОВЫ VBA

Арифметические операторы VBA

Оператор	Описание
+	Сложение. Например, в результате вычисления выражения $4+3 = 7$.
-	Вычитание.
*	Умножение
/	Деление
^	Возведение в степень, $5^2 = 25$
\	Целочисленное деление. От результата деления отбрасывается дробная часть. Например, $10 \setminus 3 = 3$.
mod	Деление по модулю. Возвращает остаток от деления. Например, $10 \bmod 3$ равняется 1

Конкатенация

объединение двух или более объектов счётного вида с сохранением порядка следования элементов. (Списков, строк, массивов, кортежей).

Несколько ролей и у оператора +

- арифметический оператор сложения.
- оператор конкатенации строк.

Пример:

операнд₁="язык"

операнд₂="Visual Basic"

операнд₁+ операнд₂="язык" + "Visual Basic"

В качестве оператора конкатенации можно использовать и оператор **&**. Считается, что **&** использовать предпочтительнее так как он в любом случае обрабатывает операнды как строковые данные.

Пример:

```
Dim str_UserName As String  
Dim str_Result As String  
Dim num_First As Double  
Dim num_Second As Double  
Dim num_Summ As Double  
str_UserName = InputBox("Введите ваше имя")  
num_First = InputBox("Введите первое число")  
num_Second = InputBox("Введите второе число")  
num_Summ = num_First + num_Second  
str_Result = "Здравствуйте, "  
& str_UserName _  
& ". Вы ввели числа " & num_First & " и " & num_Second _ & ". Их  
сумма равняется " & num_Summ  
MsgBox (str_Result)
```

Преобразование типов данных

Val — тип String в тип Double

Val (" 12345привет") возвратит число 12345.

Val ("1 2 3") возвратит число 123

Val ("1 2 и 3") возвратит число 12.

в начале строки обязательно резервируется позиция для знака числа. Если число является положительным, возвращенная строка будет содержать пробел на месте знака.

Str — числовые типы в String

Str (12) возвратит строку «12»

Другие функции (их названия состоят из сокращенного слова «Convert»): **CBool**, **CByte**, **CCur**, **CDate**, **CDbl**, **CDec**, **CInt**, **CLng**, **CSng**, **CStr**, **CVar**.

Функции проверки типа данных

- Для того, чтобы узнать тип данных переменной, можно воспользоваться функцией **TypeName**.

Пример:

```
Dim num_MyAge as Byte
```

```
num_MyAge = 24
```

```
MsgBox (TypeName(num_MyAge))
```



- функция **IsNumeric** - проверяет, являются ли данные, хранимые в переменной типа Variant, числом
- функция **VarType** - точное определение типа данных, которые хранятся в переменной типа Variant

Встроенные математические функции

Функция	Описание
<i>Int</i>	Отбрасывает дробную часть числа и возвращает целую, получается число меньшее введенного. (Int(2.5) возвратит 2), (Int(-2.5) возвратит -3).
<i>Log</i>	Возвращает <i>натуральный логарифм</i> числа
<i>Rnd</i>	Возвращает случайное число типа Single, причем, это число находится между 0 и 1. Для инициализации генератора случайных чисел используйте директиву Randomize - ее надо вызвать до вызова <i>Rnd</i> .

Функция	Описание
Sgn	Функция предназначена для определения знака числа. Если число положительное - она возвращает 1. Для нуля функция возвратит 0, для отрицательного числа -1.
Sin	Синус
Sqr	Квадратный корень
Tan	Тангенс
Atn(x)	Возвращение арктангенса угла в радианах $\text{arctg } x$
Abs(x)	Абсолютного значения числа (по модулю) $ x $

Пример

Randomize

```
MsgBox ("Группа случайных чисел: " + _  
Str(Rnd()) + ", " + _  
Str(Rnd() * 10) + ", " + _  
Str(Rnd() * 75 + 25) + ", " + _  
Str(Int(Rnd() * 34)))
```

Использование функции **Rnd()**

1. **Rnd()** – случайное число от 0 до 1
2. **Rnd() * 10** – случайное число от 0 до 10
3. **Str(Rnd() * 75 + 25)** - случайное число от 25 до 90

Строковые функции

Функция	Описание
<code>Len(string)</code>	Возвращает длину строки. Например, длина строки "Добрый день" составляет 11 символов - учитывая пробел. Выходное значение имеет тип Long
<code>LCase(string)</code>	Возвращает строку, все символы которой записаны в нижнем регистре. Например, строка "Привет" превратится в "привет"
<code>UCase(string)</code>	Возвращает строку, все символы которой записаны в верхнем регистре. Например, для "Привет" мы получим "ПРИВЕТ"

Функция	Описание
<i>String(number, character)</i>	Возвращает строку, состоящую из number символов character
<i>Left(string, length)</i>	Возвращает length символов, начиная с первого левого символа строки string
<i>Right(string, length)</i>	Возвращает length символов, начиная с самого правого символа строки string
<i>LTrim(string)</i>	Возвращает строку, в которой вырезаны все пробелы слева
<i>RTrim(string)</i>	Вырезает из строки все пробелы справа
<i>Trim(string)</i>	Вырезает из строки все пробелы слева и справа
<i>Mid(string, start[, length])</i>	Вырезает из строки string с позиции start length символов
<i>Asc(string)</i>	Возвращает ASCII-код первого символа строки
<i>Chr(charcode)</i>	Возвращает символ, соответствующий коду символа

Пример

```
str_InpStr = " У Лукоморья дуб зеленый... "
```

```
str_NewStr = Mid(Trim(str_InpStr), 2, 1)
```

```
MsgBox ("Второй символ введенной строки: " + _  
str_NewStr)
```

```
str_NewStr = Mid(Trim(str_InpStr), 15, 1)
```

```
MsgBox ("Пятнадцатый символ введенной строки: " + _  
str_NewStr)
```

'Выведем 5 символов, начиная с 15 символа

```
str_NewStr = Mid(Trim(str_InpStr), 15, 5)
```

```
MsgBox ("Пять символов строки с 15-й позиции: " + _  
str_NewStr)
```

Пользовательские процедуры и функции

- функция возвращает в точку вызова некоторое значение, которое, как правило, является результатом обработки переданной функции информации.
- процедура лишь выполняет какие-либо действия, но ничего в точку вызова не возвращает.

Пример:

num_A = num_B + val(str_C) вместо выражения **val(str_C)** подставляется числовое значение переменной **str_C**, найденное благодаря функции **Val**.

num_A = InputBox("Введите число"). Здесь функция **InputBox** возвращает введенное пользователем число в переменную **num_A**.

MsgBox ("Привет") – процедура, вызывающее окно сообщения

Пользовательские процедуры

Процедуры удобно использовать для сокращения объема программы, выделяя в них часто используемые блоки операторов. И каждый раз, когда он будет нужен, вызывать его с помощью одной лишь строчки кода.

Пример: нужно опросить пятерых пользователей, обработав и записав их ответы в файл.

```
Private Sub cmd_User_Data_Click()  
Dim str_Name As String Dim byte_Age As Byte  
MsgBox ("Здравствуйте, вы пользователь № 1")  
str_Name = InputBox("Введите ваше имя")  
byte_Age = InputBox("Введите ваш возраст")  
'Здесь находится блок обработки и 'записи введенных данных  
End Sub
```

**что нужно сделать с кодом при переходе ко
второму пользователю????**

«Скелет» процедуры

```
Public Sub UserInput(UserNumber As Integer)
```

```
'пользовательская процедура
```

```
'для ввода и обработки данных
```

```
End Sub
```

UserInput - имя процедуры

(UserNumber As Integer) - объявление переменной, которую можно передать процедуре в качестве параметра

Sub – ключевое слово (процедура!)

Public – модификатор доступа. Процедуру можно вызвать из любого места проекта.

```
Public Sub UserInput(UserNumber As Integer)
```

```
MsgBox _ ("Здравствуйте, вы пользователь № " + _  
Str(UserNumber))
```

```
End Sub
```

Вызов функции: UserInput (1)

```
Public Sub UserInput(UserNumber As Integer)
```

```
    Dim str_Name As String Dim byte_Age As Byte
```

```
    MsgBox _ ("Здравствуйте, вы пользователь № " + _  
Str(UserNumber))
```

```
    str_Name = InputBox("Введите ваше имя")
```

```
    byte_Age = InputBox("Введите ваш возраст") End Sub
```

‘Код для события «Нажатие кнопки»

```
Private Sub cmd_User_Data_Click()
```

```
    UserInput (1)
```

```
    UserInput (2)
```

```
    UserInput (3)
```

```
    UserInput (4)
```

```
    UserInput (5)
```

```
End Sub
```


Пользовательские функции

Пример: создание и использование функции, которая возводит переданное ей число во вторую степень.

```
Public Function Square(num_One As Double) As Double
```

```
    Square = num_One ^ 2
```

```
End Function
```

Public Function Square – объявление функции (доступ, функция, имя)

Square(num_One As Double) – объявление параметров, передаваемых в функцию

As Double - тип возвращаемого значения

После того, как найдено значение, которое функция должна вернуть, в тексте кода функции нужно присвоить это значение переменной, которая имеет то же имя, что и функция.

```
Private Sub cmd_UserCalc_Click()  
    Dim num_Res As Double  
    Dim num_Input As Double  
    num_Input = CDb1(InputBox("Введите число"))  
    num_Res = Square(num_Input)  
    MsgBox (Str(num_Input) + " во 2-й степени = " + _  
Str(num_Res))  
End Sub
```

Массивы и циклы

Массив

Массив - это именованный набор индексированных ячеек. Ячейки так же называют элементами массива или индексированными переменными.

Используются для обработки больших объемов однотипной информации.

У каждого массива 5 основных характеристик:

- имя,
- размерность,
- число элементов,
- номер первого элемента
- тип элементов.

Характеристики массивов

Имя - правила именования массивов аналогичны правилам именования переменных.

Размерность:

- одномерные массивы напоминают одну строку таблицы, каждая ячейка которой содержит какие-то данные.
- многомерные массивы имеют больше измерений. Их можно сравнивать с таблицами, имеющими множество строк и столбцов и с наборами таблиц.

Нумерация элементов :

- По умолчанию нумерация элементов массива начинается с 0. Первый по счету элемент получит индекс 0, второй - 1 и т.д.
- В объявлении отдельного массива можно явно указать индекс его первого и последнего элемента, разделив их ключевым словом To.
- Если вы хотите, чтобы индексы всех массивов начинались с 1, добавьте в раздел объявлений модуля (вне процедур, функций и обработчиков событий) команду Option Base 1.

Тип - подчиняется тем же правилам, которые мы ранее рассмотрели для переменных. Если пользователь не указал его явно, массив получит тип по умолчанию - Variant. Это требует больше системных ресурсов, но позволяет обрабатывать значения различных типов.

Одномерные массивы

Для объявления массивов используют оператор **Dim**.

Объявить массив можно двумя способами.

- указать общее количества элементов.

Пример:

```
Dim MyArrayA(30) As Single
```

Объявленный массив MyArrayA содержит 31 элемент (с индексами от 0 до 30) типа Single.

Поскольку нумерация явно не задана, элементы получают индексы по обычным правилам.

- задать границы нумерации

```
Dim MyArrayB(1 To 25)
```

Массив MyArrayB содержит 25 элементов. Границы нумерации заданы явно - первый элемент получит индекс 1, второй - 2 и т.д. Тип не указан - в массиве можно хранить любые данные.

Пример

1. Объявить одномерный массив на 3 элемента
2. Внести в первый элемент число 5 в программе, во второй - запросив значение с помощью окна ввода
3. Вычислить в третьем элементе массива произведение значений, хранящихся в первом и втором элементах.
4. Вывести полученное значение в окне сообщения.

Dim A(2)

A(0) = 5

*A(1) = InputBox("Введите значение второго
элемента")*

*A(2) = A(0) * A(1)*

MsgBox A(2)

Индекс	0	1	2
Значение	5	2	10

Виды циклов

Название цикла	Вид
For - Next	С фиксированным количеством повторов. Выполняется заданное количество раз
While - Wend	С предусловием. Если не верно условие, заданное на входе в цикл, может не выполниться ни разу
Do - Loop	С постусловием. Выполняется по меньшей мере один раз.

Цикл For - Next

Пример: вывести цифры от 1 до 10 в окнах сообщений.

Можно решить и без использования циклов, написав 10 строк вида: MsgBox ("1") и т.д.

```
For i = 1 To 10  
    MsgBox (i)  
Next i
```

For - ключевое слово, задает начало цикла.

Переменная **i**, увеличивается при каждом проходе цикла.

Начальное значение счетчика устанавливается при входе в цикл.

Конечное значение переменной задается после ключевого слова **To**.

Ключевое слово **Next** с указанием переменной, к которой оно относится, закрывает цикл.

В качестве первого и последнего значения счетчика цикла можно использовать какую-нибудь переменную. Она может быть определена в ходе выполнения программы, но до входа в конструкцию **For - Next**.

Переменная цикла меняется с приращением

приращение можно задать в явном виде с помощью ключевого слова **Step**.

```
For i = 1 To 10 Step 2
```

```
    MsgBox (i)
```

```
Next i
```

С помощью ключевого слова Step можно создать не только возрастающий, но и убывающий счетчик. Для этого в Step надо указать отрицательное число и проследить за тем, чтобы первое значение переменной цикла было больше последнего.

Пример: *For i=10 to 1 Step -1.*

Первое значение переменной в таком цикле будет равняться 10, второе - 9 и т.д. - до 1.

Пример: использование массивов и цикла

Программа, которая предлагает пользователю ввести 10 фамилий, сохраняет их в массиве, а потом выводит в окнах сообщений.

```
Dim MyArray(9)
```

```
For i = 0 To 9
```

```
MyArray(i) = InputBox("Введите фамилию №" & i + 1)
```

```
Next i
```

```
For i = 0 To 9 'Начало еще одного цикла
```

```
MsgBox ("Фамилия №" & (i + 1) & " " & MyArray(i))
```

```
Next i 'Конец цикла
```


Многомерные массивы

Двумерный массив - матрица

```
Dim MyArrayA(10, 1) As Single
```

Массив MyArrayA содержит 11 строк и 2 столбца типа Single.

Можно в явном виде задать границы размерностей:

```
Dim MyArrayB(1 To 25, 1 To 5)
```

Массив MyArrayB содержит 25 строк и 5 столбцов. Границы нумерации заданы явно. Тип не указан - в массиве можно хранить любые данные.

Пример

программы, которая объявляет двумерный массив 5x2 и предлагает ввести в него фамилии и номера телефонов сотрудников

```
Dim MyArray(1 To 5, 1 To 2)
```

```
For i = 1 To 5
```

```
    MyArray(i, 1) = InputBox("Введите фамилию №" & i)
```

```
    MyArray(i, 2) = InputBox("Введите Телефон №" & i)
```

```
Next i
```

Индекс	1	2
1	Иванов	898989898
2	Петров	343434343
3	Сидоров	565656565
4	Александров	121111212
5	Маринин	545454544

Вложенные циклы For-Next

```
Dim MyArray(1 To 10, 1 To 10)
```

```
For i = 1 To 10
```

```
    For j = 1 To 10
```

```
        MyArray(i, j) = Int(Rnd(1) * 10)
```

```
    Next j
```

```
Next i
```

Динамические массивы

```
Dim MyArray()
```

```
ArraySize = InputBox("Введите количество  
сотрудников")
```

```
ReDim MyArray(1 To ArraySize, 1 To 2)
```

Для добавления двух дополнительных столбцов в динамический массив нужно использовать такую команду:

```
ReDim Preserve MyArray(1 To ArraySize, 1 To 4)
```

Дополнительные команды работы с массивами

Array (Список аргументов) - позволяет быстро заполнять массив. Например:

```
Dim MyArray
```

```
MyArray = Array(1, 2, 6, 9, 19)
```

```
MsgBox MyArray(0)
```

IsArray (Имя переменной) - возвращает True если переменная является массивом.

```
Dim MyArray(10)
```

```
If IsArray(MyArray) Then _
```

```
MsgBox ("Переменная MyArray - массив") _
```

```
Else MsgBox ("Переменная MyArray - не массив")
```

ФУНКЦИИ

LBound (Имя Массива, Размерность) - возвращает нижнюю границу для указанной размерности массива.

UBound (Имя Массива, Размерность) - возвращает верхнюю границу для указанной размерности массива.

Пример

```
Dim MyArray()  
ReDim MyArray(Int(Rnd * 5 + 5), Int(Rnd * 5 + 5))  
MsgBox ("Двумерный массив MyArray:" + Chr(13) + _  
"Первая размерность:" + Str(LBound(MyArray, 1)) + « - » _ +  
Str(UBound(MyArray, 1)) + Chr(13) + _  
"Вторая размерность:" + Str(LBound(MyArray, 2)) + « - » _  
+ Str(UBound(MyArray, 2)))  
For i = LBound(MyArray, 1) To UBound(MyArray, 1)  
    For j = LBound(MyArray, 2) To UBound(MyArray, 2)  
        MyArray(i, j) = Int(Rnd * 100)  
    Next j  
Next i
```


Erase Имя_массива - очистить массив. Элементы обычных массивов, содержащих числовые данные, обнуляются.

Если мы применим команду Erase к массиву строк - каждый его элемент будет хранить строку нулевой длины ("").

Применяя команду Erase к динамическому массиву, мы очищаем память, выделенную этому массиву командой ReDim. Для повторного использования динамического массива, придется снова устанавливать его размерности.

Цикл с предусловием

Цикл с предусловием **While - Wend** выполняется до тех пор, пока условие, указанное на входе, верно.

A = 1

While A < 10

A = Int(Rnd() * 20)

MsgBox A

Wend

Цикл с постусловием

Цикл **Do-Loop While** выполняется до тех пор, пока значение на выходе из цикла верно. Подобные циклы используют, например, для проверки правильности ввода каких-либо данных пользователем. Если данные введены неверно - цикл выполняется снова.

Аналогично действует цикл **Do-Loop Until** - он будет выполняться до тех пор, пока условие цикла неверно (то есть равно False).

```
Dim var_A
```

```
Do
```

```
var_A = InputBox("Введите число")
```

```
Loop Until IsNumeric(var_A)
```

Пользователю предлагается ввести какое-нибудь число. Если введено не число (то есть функция `IsNumeric` возвратит `False`), программа выведет окно ввода снова.

Оператор Like

Like используется для сравнения строк с шаблонами.

Шаблон - это особым образом записанная последовательность символов.

При построении шаблонов используются специальные символы, приведенные в таблице.

Символы для построения шаблонов

Символы	Описание
?	Любой одиночный символ
*	Любое количество любых символов
#	Любая одиночная цифра
[СПИСОК СИМВОЛОВ]	Любой одиночный символ, входящий в список символов
[! СПИСОК СИМВОЛОВ]	Любой одиночный символ, не входящий в список

Пример

Узнать, есть ли в строке прописные и заглавные буквы латинского алфавита.

Проверить, состоит ли введенное слово из четырех символов (цифр или букв)

Проверить, состоит ли введенная последовательность из двух любых символов (цифр или букв) и двух цифр

Проверить, нет ли во введенной строке русских букв "а" и "о"

Если введенная строка начинается двумя буквами "d" и заканчивается тремя буквами "f", сообщить об этом

Dim str_Inp As String

str_Inp = InputBox("Введите строку")

'Есть ли латинские буквы в строке

If str_Inp Like "[a-z]" Or str_Inp Like "[A-Z]*" _*

Then MsgBox ("В строке есть латинские буквы")

'Состоит ли введенное слово из 4-х символов

If str_Inp Like "????" Then _

MsgBox ("Введенное слово состоит из 4-х символов")

'Состоит ли введенная последовательность

'из 2-х любых символов и 2-х цифр

If str_Inp Like "??##" Then _

MsgBox ("Введены два любых символа и две цифры")

'Проверка на отсутствие букв "a", "o"

If str_Inp Like "[!a]" And _

str_Inp Like "[!o]" Then

MsgBox ("В строке нет букв " + Chr(34) + _

"a" + Chr(34) + " и " + Chr(34) + "o" + Chr(34))

End If

'Проверка на наличие в начале введенной 'строки двух букв "d", а в конце трех "f"

*If str_Inp Like "dd*fff" _*

*Then MsgBox ("Строка имеет вид: dd*fff")*

Работа с внешними файлами

```
var _Doc = Dir("C:\*.*.*)  
  Do While var _Doc <> ""  
    MsgBox var _Doc  
    var _Doc = Dir()  
  Loop
```

Поиск всех файлов в корневом каталоге диска С.

Функция Dir возвращает строку, содержащую имя файла, используя путь, заданный при вызове

Dir

По умолчанию функция ищет лишь обычные файлы, не обращая внимания на папки, скрытые и системные файлы. Чтобы функция нашла по заданному пути не только файлы, но и папки, ее нужно вызвать так:

```
var _Doc = Dir("C:\*.*", vbDirectory)
```

Обратите внимание на то, что после пути и маски указан параметр `vbDirectory` - он указывает функции, что она должна включить в поиск и директории.

ChDir

Помимо Dir полезной может оказаться команда ChDir. Она позволяет перейти в указанную при ее вызове директорию, которая будет использоваться в качестве директории по умолчанию. Такая конструкция, предшествующая циклу из предыдущего примера позволит найти все файлы в папке "Документы", которая расположена по пути "C:\Документы":

```
ChDir ("C:\Документы")
```

```
var _doc = Dir("*.*")
```