Оглавление

[VBA-Урок 1. Что такое VBA. Основные понятия. 1](#_Toc111713687)

[**Что такое VBA?** 1](#_Toc111713688)

[**Объекты (Objects)** 2](#_Toc111713689)

[**Коллекции (Collections)** 2](#_Toc111713690)

[**Свойства (Properties)** 3](#_Toc111713691)

[**Методи (Methods)** 3](#_Toc111713692)

[VBA-Урок 2. Отладчик (Debugger) 3](#_Toc111713693)

[VBA-Урок 3. Коллекция Sheets 5](#_Toc111713694)

[**Как посчитать количество листов в книге** 5](#_Toc111713695)

[**Как добавить лист в книгу** 6](#_Toc111713696)

[**Как скрыть лист** 6](#_Toc111713697)

[VBA-Урок 4. Работа с диапазонами (Range) 6](#_Toc111713698)

[**Выделяем диапазон и определяем его параметры** 6](#_Toc111713699)

[**Выделение диапазона с помощью .UsedRange** 7](#_Toc111713700)

[VBA-Урок 5. Свойства (Properties) 8](#_Toc111713701)

[1. Запись значений в ячейку 8](#_Toc111713702)

[2. Удаление значений 9](#_Toc111713703)

[3. Форматирование значений 10](#_Toc111713704)

[VBA-Урок 6.1. Tипы данных (Variables) 11](#_Toc111713705)

[**Tипы переменных** 12](#_Toc111713706)

[VBA-Урок 6.2. Tипы данных (Variables) 13](#_Toc111713707)

[**Одномерный массив** 14](#_Toc111713708)

[**Константы** 14](#_Toc111713709)

[**Создание собственного типа переменных** 16](#_Toc111713710)

[**VBA-Урок 7.1. Условия (Conditions)** 16](#_Toc111713711)

[**ElseIf** 20](#_Toc111713712)

[**Select** 21](#_Toc111713713)

[**VBA-Урок 7.2. Условия (Conditions)** 22](#_Toc111713714)

[**Условие, что базируется на типе** 22](#_Toc111713715)

[**Условие, базирующееся на типе переменной** 22](#_Toc111713716)

[**Условие, базирующееся на сравнении двух текстовых строк** 23](#_Toc111713717)

[**VBA-Урок 8.1. Циклы (Loops)** 24](#_Toc111713718)

[**WHILE** 25](#_Toc111713719)

[**DO LOOP** 25](#_Toc111713720)

[**FOR** 26](#_Toc111713721)

[**Ранний выход из цикла** 26](#_Toc111713722)

[**VBA-Урок 12.1. Пользовательские формы (UserForm)** 27](#_Toc111713723)

[**Запуск формы (UserForm)** 29](#_Toc111713724)

[**VBA-Урок 12.2. Элементы управления (Controls)** 30](#_Toc111713725)

[**Checkboxes** 32](#_Toc111713726)

[**Кнопки опций (Option Buttons)** 34](#_Toc111713727)

[**VBA-Урок 12.3. Элементы управления (Controls)** 37](#_Toc111713728)

[**Полоса прокрутки (ScrollBar)** 37](#_Toc111713729)

[**Выпадающий список и Окно со списком (ComboBox and ListBox)** 39](#_Toc111713730)

[**VBA-Урок 13.1. Использование массивов (Arrays)** 41](#_Toc111713731)

[**Почему используются массивы?** 41](#_Toc111713732)

[**Декларирование массивов** 46](#_Toc111713733)

[**Хранение данных в массиве** 47](#_Toc111713734)

[**VBA-Урок 13.2. Использование массивов (Arrays) (Продолжение)** 48](#_Toc111713735)

[**Динамический массив** 49](#_Toc111713736)

[**Ubound** 50](#_Toc111713737)

[**Сохранение данных в диапазоне элементов массива** 50](#_Toc111713738)

[**Массив (Array)** 51](#_Toc111713739)

[**Разделение (Split)** 52](#_Toc111713740)

# VBA-Урок 1. Что такое VBA. Основные понятия.

В данном разделе мы рассмотрим что такое VBA, посколько именно на нем и пишутся макросы.

**Что такое VBA?**

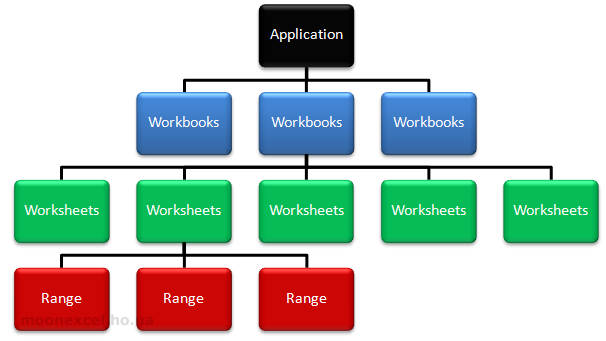
**VBA**- язык (расшифровывается как **Visual Basic for Application**) был разработан компанией **Microsoft**. Данный язык не является самостоятельным, а предназначен для автоматизации процессов в пакете **MS Office**. **VBA**широко используется в **Excel**, а также в **Access**, **Word**и других программах пакета.

**VBA** - простой язык программирования, которому может научиться любой желающий. Изучив его, вы сможет предоставлять команды **Excel**, что делать с колонками, строками, значениями в ячейках, перемещать/добавлять/сортировать листы, выводить заранее запрограммированные сообщения, писать свои формулы и функции и т.д. Суть языка заключается в оперировании **объектами**(что относит его к **объектно-ориентированному программированию**).

Чтобы работать с **VBA**кодом, нам нужен редактор, который уже установлен по умолчанию. Вы можете открыть его, нажав комбинацию клавиш "[**ALT + F11**](http://moonexcel.com.ua/tip_macros_ru)".

**Объекты (Objects)**

Давайте разберем, что же такое объект. **Объект**- это элемент, структурная частица **Excel**, а именно: книга, лист, диапазон, ячейка. Данные объекты имеют иерархию, т.е. подчиняются друг другу. Схематично структуру иерархии **Excel**можно изобразить следующим образом:



Главный объект это ***Application***, что соответствует самой программе **Excel**. Далее следует ***Workbooks***(книга), ***Worksheets***(лист), ***Range***(диапазон, или отдельная ячейка). Например, чтобы обратиться к ячейке **"A1"**на листе нам нужно будет прописать следующий путь с учетом иерархии:

***Application.Workbooks("Архив").Worksheets("Аркуш1").Range("A1")*.**

Таким образом, мы научились обращаться до наименьшего объекта в **Excel**- ячейки.

**Коллекции (Collections)**

В свою очередь объекты имеют "коллекции". **Коллекция**- это группа объектов одинакового класса. Отдельные элеметы коллекции являются также объектами. Так, объекты ***Worksheets***являются элементами коллекции объекта ***Worksheet***, который содержит также и другие коллекции и объекты:

* ***ChartObjects*** (элемент коллекции объекта **ChartObject**)
* ***Range***
* ***PageSetup***
* ***PivotTables*** (элемент коллекции объекта **PivotTable**).

**Свойства (Properties)**

Каждый объект имеет **свойства**. Например, объект ***Range***имеет свойство ***Value***или ***Formula***.

**Worksheets(“Sheet1”).Range(“A1”).Value**або ***Worksheets(“Sheet1”).Range(“A1”).Formula***

В данном примере, свойство отображает значение, которое введено в ячейку или введенную формулу.

Также, через свойство ***Formula***можно не только получить формулу, но и записать ее:

***MsgBox Range(“A1”).Formula*** - получим сообщение с формулой в ячейке "**А1**";

**Range(“B12”).Formula = “=2+6\*100”**- вписываем формулу ***=2+6\*100*** в ячейку **B12**.

**Методи (Methods)**

Теперь давайте рассмотрим, каким образом мы можем управлять содержимым диапазона или ячейки. Для этого в **VBA**существуют, так-называемые **методы**(команды "что сделать"). При написании кода **методы**отделяются от **объекта**точкой, например:

***Range("A1").Select*** или ***Cells(1, 1).Select***

Данный метод указывает выбрать (**Select**) ячейку **"A1"**.  
Далее, давайте, удалим значение в данной ячейке. Для этого напишем следующий код:

***Selection.ClearContents***

Здесь программа "берет" то, что мы выделили (***Selection***) и удаляет его содержимое (***ClearContents*** ).

# VBA-Урок 2. Отладчик (Debugger)

Иструмент в среде **VBA**для отладки кода программ. Уметь работать с **дебагером** жизненно важно каждому, кто пишет на **VBA**. Он нужен для того, чтобы проверить как работает ваш код, и чтобы найти и исправить в нем ошибки.

Откройте ***Visual Basic Editor*(Alt + F11)**. [**Создайте**](http://moonexcel.com.ua/tip_macros_ru)в нем модуль и пропишите следующий код:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/uroki-vba2-debugger_ru)

1. **Sub** LearningDebug()
2. **Dim** A **As** **Long**, B **As** **Long**, C **As** **Long**, D **As** **Long**
4. D = 0
5. A = 10
6. Debug.Print "A = " + Trim(Str(A))
7. B = 15
8. Debug.Print "B = " + Trim(Str(B))
9. C = A + B
10. Debug.Print "C = " + Trim(Str(C))
11. C = Round(C /5)
12. Debug.Print "С делим на 5: C = " + Trim(Str(C))
13. C = Round(C /D) ' здесь будет ошибка, поскольку на ноль делить нельзя (а D = 0)
14. **End** **Sub**

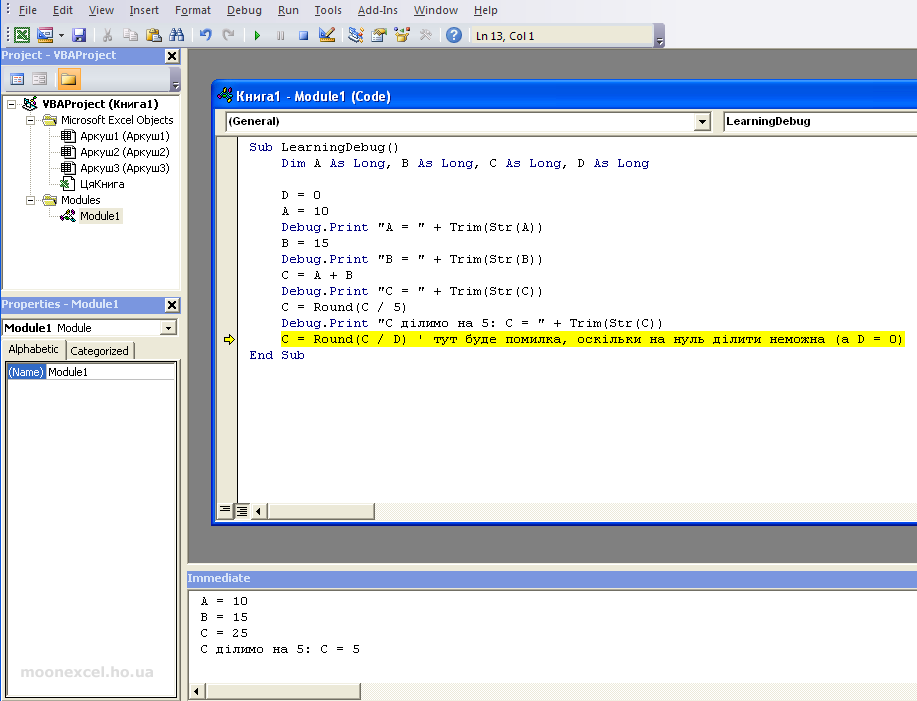
Нажмите **Ctrl + G**, при этом внизу появится окно **Immediate**. В этом окне можно в ходе работы макроса изменять значения переменных, выполнять различный код **VBA**, не написан в модуле ранее.

Если стать курсором внутрь кода модуля и нажать **F5**, он выполнится полностью. Если нажать **F8**- исполнится только одна строка кода. Нажатие **F8**выполнит следующую строку кода и т.д. можно выполнить пошагово весь код. С помощью кнопки **F9**можно создать точку остановки. Если потом вы запустите код с помощью кнопки **F5**, код будет выполнен до указанной строки и затем приостановится. Далее его можно будет продлить с помощью кнопок **F5**или **F8**, описанных ранее.

Если внимательно рассмотреть код, вы увидите, что в переменные **A, B, C, D**присваиваются числа. Строки **Debug.Print "текст"**печатают указанный нами текст в открытое ранее окно ***Immediate***, чтобы видеть, что происходит с переменной. **Str (A)**превращает число в текстовое значение. А **Trim ()**убирает с него справа и слева пробелы. Оператор **Round (С)**округляет значения по правилам арифметики до целого числа (поэтому результат вычислений мы присваиваем в переменную целого типа **Long**, значит он должен быть целым). В конце кода мы специально создали ложную ситуацию, чтобы потренироваться использовать **дебагер**.

Станьте внутрь кода и нажмите **F8**четыре раза. Желтым будет отмечено строку, который будет выполнен следующим. Наведите курсор на различные переменные. При этом появится всплывающее окошко, в котором увидите значение. Переменные, которым еще не были присвоены значения равны нулю.  
Нажмите еще раз **F8**. В окне ***Immediate***появится строка:  
**А = 10**  
Теперь мы можем изменить значение переменной **A**сразу в окне ***Immediate***(например, изменим его на **8**) . Для этого в новой строке окна ***Immediate***напишите:  
**А = 8**  
и нажмите **Enter**. Теперь наведите на переменную **A**курсор и увидите, что его значение равно **8**. чтобы увидеть значение, не приводя курсор, можно заставить его появиться в окне ***Immediate***. Делается это так, введите в пустой строке окна ***Immediate***:  
**?A**  
и нажмите **Enter**. Знак в окне ***Immediate***означает то же самое, что и **Debug.Print**в коде. Просто так удобнее и короче писать. Сразу вы увидите число - результат вашего запроса.

Станьте курсором на последнюю строку кода **(C = Round (C /D))**и нажмите **F9**. Появится точка остановки. Той же кнопкой ее можно убрать, но мы ее убирать пока не будем. Нажмите **F5**, программа выполнит все строки и остановится на последней. В окне ***Immediate***появятся, сообщение о присвоении значений переменным. Мы добрались до последней строки кода. Он должен вызвать ошибку, так как содержит деления на ноль. Нажмите **F8**и убедитесь в этом. В появившемся окне нажмите ***Debug***. Если вы нажмете ***End***, выполнение программы остановится, а мы хотим довести ее до конца.



Исправить ситуацию можно, изменив значение переменной **D**. Сейчас она равна нулю. В окне ***Immediate***введите в пустой строке текст:  
**D = 2**  
и нажмите **Enter**. Теперь последней строкой кода мы делим не на ноль, а на **2**, таким образом избегая ошибки. Нажмите **F5**и программа завершит свою работу без ошибок.

Попробуйте изменить код потренируйтесь в использовании отладчика. Поверьте, он вам очень облегчит жизнь.

# VBA-Урок 3. Коллекция Sheets

Данная коллекция представляет собой набор листов (***Sheets***) в книге (***Workbooks***). Давайте посмотрим, какие действия мы можем делать над листами.

**Как посчитать количество листов в книге**

Сначала попробуем узнать сколько листов имеет наша книга:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/uroki-vba3-sheets_ru)

1. **Sub** Test()
2. MsgBox (Str(Application.Workbooks.Item("Test.xls").Sheets.Count))
3. **End** **Sub**

Данным кодом мы вызвали сообщения на экран (***MsgBox***), которое отобразило количество листов (***Sheets.Count***) в книге (***Workbooks***) "**Test.xls**".

Под листом понимается не только ячейки, но и диаграммы. Также, как и лист для расчета, диаграмма будет включена в подсчет листов.

**Как добавить лист в книгу**

В коллекции листов также есть возможность добавлять свои листы, для этого существует метод **Add**. Этот метод имеет 4 параметра **Add (*Before, After, Count, Type*)**. Все эти параметры необязательны. Первые два отвечают за место вставки листа. Далее, количество листов, вставляемых ***Count***и тип листа ***Type***. Типы могут быть, например, такие **xlWorkSheet**для расчетного листа и **xlChart**для диаграммы. Если местоположение не указывать, то лист будет вставляться относительно текущего листа.

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/uroki-vba3-sheets_ru)

1. **Sub** Test()
2. Sheets.Add After:=Worksheets("Лист3"), Count:=4
3. **End** **Sub**

Таким образом мы вставили 4 листа (**Count: = 4**) после листа **"Лист3"**.

Также можно вставить лист в самый конец книги:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/uroki-vba3-sheets_ru)

1. **Sub** Test()
2. Worksheets.Add
3. ActiveSheet.Move After:=Sheets(ActiveWorkbook.Sheets.Count)
4. **End** **Sub**

**Как скрыть лист**

Если у Вас есть желание, то некоторые листы можно скрыть. Это бывает полезно, если у Вас есть константы или расчеты, которые Вы не хотите чтобы видели на экране в виде листов. Для этого можно использовать метод ***Visible***. Устанавливая это свойство в **TRUE**или **FALSE**вы можете убрать или отобразить необходимый лист.

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/uroki-vba3-sheets_ru)

1. **Sub** Test()
2. ActiveWorkbook.Sheets("Лист3").Visible = **False**
3. **End** **Sub**

# VBA-Урок 4. Работа с диапазонами (Range)

Рассмотрим каким образом мы можем работать с диапазоном. Для этого в **Excel** существует объект **Range**, включающий в себя как диапазоны ячеек, так и одну отдельную ячейку. Давайте посмотрим, какие действия мы можем делать над диапазоном.

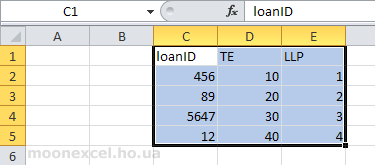
**Выделяем диапазон и определяем его параметры**

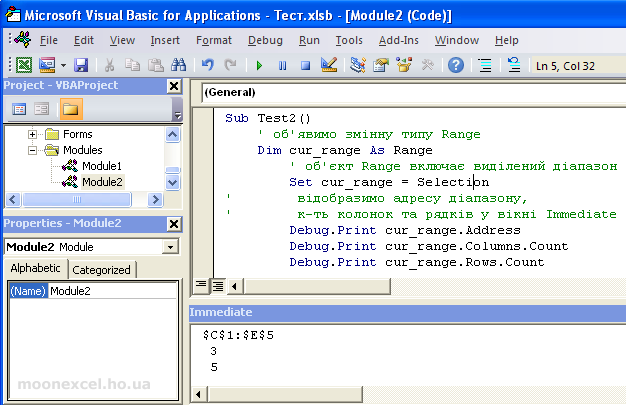
Сначала попробуем выделить диапазон и определить его параметры:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/uroki-vba4-range_ru)

1. **Sub** Test2()
2. 'moonexcel.com.ua
3. 'объявим переменную типа Range
4. **Dim** cur\_range **As** Range
5. 'объект Range включает выделенный диапазон
6. **Set** cur\_range = Selection
7. 'отобразим адрес диапазона, кол-во колонок и строк в окне Immediate
8. Debug.Print cur\_range.Address
9. Debug.Print cur\_range.Columns.Count
10. Debug.Print cur\_range.Rows.Count
11. **End** **Sub**

Данным кодом мы присвоили нашему диапазону **cur\_range**выделены ячейки. Далее, с помощью функции **Debug.Print**отобразили параметры диапазона в окне предварительного просмотра значений ***Immediate***.





Видим, что адрес диапазона **$C$1:$E$5**, кол-во колонок - **3**, кол-во строк - **5**.

**Выделение диапазона с помощью .UsedRange**

Давайте рассмотрим как можно выделить наш диапазон другим способом. Для этого воспользуемся **.UsedRange**

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/uroki-vba4-range_ru)

1. **Sub** Test()
2. 'moonexcel.com.ua
3. **Dim** cur\_range **As** Range
4. **Set** cur\_range = ActiveSheet.UsedRange
5. Debug.Print cur\_range.Address
6. **End** **Sub**

Преимущество данного способа в том, что Вам не нужно выделять диапазон вручную, за Вас это сделает **Excel**, который проанализирует какие ячейки имеются заполненные в листе, и выберет только их.

# VBA-Урок 5. Свойства (Properties)

В данной статье мы будем учиться писать VBA код, чтобы управлять содержимым ячеек, самими ячейками и листами.

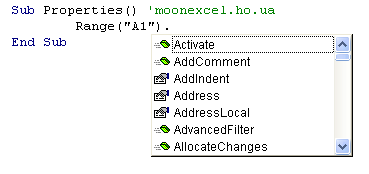
## 1. Запись значений в ячейку

Для начала откроем редактор, добавим модуль, скопируем туда этот макрос:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba5-%D1%81%D0%B2%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%B0-properties_ru)

1. **Sub** Properties() 'moonexcel.com.ua
2. Range ("A1")
3. **End** **Sub**

Мы обратились к ячейке **A1**. Теперь давайте попробуем управлять данной ячейкой. Чтобы увидеть, что мы можем сделать, добавим точку после **Range ("A1")**.



Выберите значение Value и нажмите Tab. Получим такой код:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba5-%D1%81%D0%B2%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%B0-properties_ru)

1. **Sub** Properties() 'moonexcel.com.ua
2. Range ("A1").Value
3. **End** **Sub**

Значение Value отображает содержимое ячейки.

Теперь давайте запишем значение 35 в ячейку A1:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba5-%D1%81%D0%B2%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%B0-properties_ru)

1. **Sub** properties() 'moonexcel.com.ua
2. Range("A1").Value = 35
3. 'Значение ячейки A1 равно 35
4. **End** **Sub**

Попробуем теперь записать текст в ячейку (когда присваиваем текстовое значение, его нужно брать в двойные кавычки ""):

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba5-%D1%81%D0%B2%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%B0-properties_ru)

1. **Sub** properties() 'moonexcel.com.ua
2. Range("A1").Value = "Тут є якийсь текст"
3. **End** **Sub**

Заметим, что макрос будет отображать значение в том листе, который вы последний раз открывали. Поэтому, чтобы управлять содержимым ячейки на любом листе книги, нам нужно будет прописать полный путь к ячейке, а именно добавить впереди кода название листа, например:

**Вариант 1.**Обращаемся по названию листа - Sheets("Sheet2").

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba5-%D1%81%D0%B2%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%B0-properties_ru)

1. **Sub** properties() 'moonexcel.com.ua
2. Sheets("Sheet2").Range("A1").Value = "Тут є якийсь текст"
3. 'Означает: открыть лист 2, выбрать ячейку A1 и в ее значение записать текст
4. **End** **Sub**

**Вариант 2.**Обращаемся не по названию листа, а по его порядковому номеру - Sheets(2).

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba5-%D1%81%D0%B2%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%B0-properties_ru)

1. **Sub** properties() 'moonexcel.com.ua
2. Sheets(2).Range("A1").Value = "Здесь есть какой-то текст"
3. **End** **Sub**

Аналогично, если мы хотим обратиться к ячейке в другой книге, нам нужно в начале кода прописать название книги:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba5-%D1%81%D0%B2%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%B0-properties_ru)

1. **Sub** properties() 'moonexcel.com.ua
2. Workbooks("Book2.xlsx").Sheets("Sheet2").Range("A1").Value = "Здесь есть какой-то текст"
3. **End** **Sub**

Хотя мы указываем параметр Value в наших примерах, на самом деле его можно не использовать, поскольку он стоит по умолчанию. То есть, эти две строки кода будут эквивалентными:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba5-%D1%81%D0%B2%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%B0-properties_ru)

1. **Sub** properties() 'moonexcel.com.ua
2. Range("A1").Value = 35
3. Range("A1") = 35
4. **End** **Sub**

## 2. Удаление значений

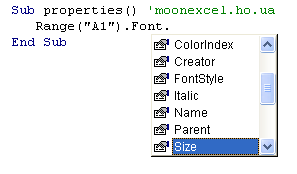
Удалим значение 35 из ячейки A1, которое мы записали в начале урока:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba5-%D1%81%D0%B2%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%B0-properties_ru)

1. **Sub** properties() 'moonexcel.com.ua
2. Range("A1").Clear
3. 'Означает: выбрать ячейку A1 и очистить ее
4. **End** **Sub**

## 3. Форматирование значений

Если вы выберете значение **Font**, то появится список параметров, которые вы сможете применить к ячейке:



#### Форматирование: изменение размера текста

Давайте присвоим ячейке значение 35 и уменьшим размер шрифта до 8:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba5-%D1%81%D0%B2%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%B0-properties_ru)

1. **Sub** properties() 'moonexcel.com.ua
2. Range("A1") = 35
3. Range("A1").Font.Size = 8
4. **End** **Sub**

http://moonexcel.com.ua/images/vba5_ua_3.png

#### Форматирование: сделать текст жирным

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba5-%D1%81%D0%B2%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%B0-properties_ru)

1. **Sub** properties() 'moonexcel.com.ua
2. Range("A1").Font.Bold = **True**
3. **End** **Sub**

Забрать выделение полужирным:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba5-%D1%81%D0%B2%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%B0-properties_ru)

1. **Sub** properties() 'moonexcel.com.ua
2. Range("A1").Font.Bold = **False**
3. **End** **Sub**

#### Форматирование: сделать текст наклонным

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba5-%D1%81%D0%B2%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%B0-properties_ru)

1. **Sub** properties() 'moonexcel.com.ua
2. Range("A1").Font.Italic = **True**
3. **End** **Sub**

#### Форматирование: подчеркнуть текст

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba5-%D1%81%D0%B2%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%B0-properties_ru)

1. **Sub** properties() 'moonexcel.com.ua
2. Range("A1").Font.Underline = **True**
3. **End** **Sub**

#### Форматирование: установить тип шрифта

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba5-%D1%81%D0%B2%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%B0-properties_ru)

1. **Sub** properties() 'moonexcel.com.ua
2. Range("A1").Font.Name = "Arial"
3. **End** **Sub**

#### Форматирование: закрасить ячейку

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba5-%D1%81%D0%B2%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%B0-properties_ru)

1. **Sub** properties() 'moonexcel.com.ua
2. Range("A1").Interior.ColorIndex = 6
3. **End** **Sub**

http://moonexcel.com.ua/images/vba5_ua_4.png

# VBA-Урок 6.1. Tипы данных (Variables)

С помощью переменных мы можем записывать все типы информации, такие как: числа, текст, дата и т.д.

Рассмотрим первый пример:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba6-%D1%82%D0%B8%D0%BF%D1%8B-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85-variables_ru)

1. **Sub** variables()
2. 'Декларируем переменную как целое число
3. **Dim** my\_variable **As** **Integer**
4. 'Присваиваем переменной определенное значение
5. my\_variable = 12
6. 'Отображаем значение переменной в окне сообщения
7. MsgBox my\_variable
8. **End** **Sub**

Первая строка кода декларирует переменную:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba6-%D1%82%D0%B8%D0%BF%D1%8B-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85-variables_ru)

1. **Dim** my\_variable **As** **Integer**

* - **Dim**: определение переменной
* - **my\_variable**: имя переменной
* - **As**: определение типа переменной
* - **Integer**: тип переменной

В общем, декларирование переменных не является обязательным и код будет работать, однако это настоятельно рекомендуется. Это облегчает работу с ними, их поиск, позволяет решать разного рода проблемы. В принципе, декларирование переменных является хорошей привычкой при программировании.

Тип переменной указывает на природу ее содержания (текст, число, дата и т.д.).

Вторая строка присваивает значение переменной:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba6-%D1%82%D0%B8%D0%BF%D1%8B-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85-variables_ru)

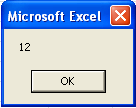
1. my\_variable = 12

И наконец третья строка отображает значение переменной в окне сообщения:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba6-%D1%82%D0%B8%D0%BF%D1%8B-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85-variables_ru)

1. MsgBox my\_variable

Результат выполнения кода будет следующий:



**Tипы переменных**

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Тип | Описание | Символ |
| Byte | Число | Целое число от 0 до 255 |  |
| Integer | Число | Целое число от -32'768 до 32'767 | % |
| Long | Число | Целое число от - 2'147'483'648 до 2'147'483'647 | & |
| Currency | Число | Фиксированное десятичное число между -922'337'203'685'477.5808 до 922'337'203'685'477.5807 | @ |
| Single | Число | Десятичное число с плавающей запятой между -3.402823E38 до 3.402823E38 | ! |
| Double | Число | Десятичное число с плавающей запятой между -1.79769313486232D308 до 1.79769313486232D308 | # |
| String | Текст | Текст | $ |
| Date | Дата | Дата и время |  |
| Boolean | Булево | True (ПРАВДА) или False (ЛОЖЬ) |  |
| Object | Объект | Microsoft Object |  |
| Variant | Другой | Любой вид данных (тип по умолчанию, если переменная не определена) |  |

Рассмотрим несколько примеров с различными типами переменных:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba6-%D1%82%D0%B8%D0%BF%D1%8B-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85-variables_ru)

1. 'Целое число
2. **Dim** nbInteger **As** **Integer**
3. nbInteger = 12345
5. 'Десятичное число
6. **Dim** nbComma **As** **Single**
7. nbComma = 123.45
9. 'Текст
10. **Dim** varText **As** **String**
11. varText = "moonexcel.com.ua"
13. 'Дата
14. **Dim** varDate **As** **Date**
15. varDate = "24.08.2012"
17. 'Булево значение True/False
18. **Dim** varBoolean **As** **Boolean**
19. varBoolean = **True**
21. 'Объект (Рабочий лист как тип переменной)
22. **Dim** varSheet **As** Worksheet
23. **Set** varSheet = Sheets("Sheet2") 'Set => присвоение значения для переменной типа "объект"
25. 'Пример использования переменной типа "объект": активирование листа
26. varSheet.Activate

Символы, указанные в таблице, могут использоваться для более краткого декларирования переменных:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba6-%D1%82%D0%B8%D0%BF%D1%8B-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85-variables_ru)

1. **Dim** example **As** **Integer**
2. **Dim** example%

Эти две записи являются идентичными.

Комментарий : Можно также заставить пользователя декларировать переменные принудительно, прописав команду **Option Explicit**в начале модуля. В таком случае будет появляться сообщение об ошибке, если было введено незадекларированную переменную.

# VBA-Урок 6.2. Tипы данных (Variables)

На предыдущем уроке мы рассматривали переменные, в которые можно записывать определенные данные, однако они не могут содержать более одного значения. Чтобы обойти это ограничение, в VBA существуют массивы, в которые можно записывать множество значений.

Приведем несколько примеров различных массивов:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba6-2-%D1%82%D0%B8%D0%BF%D1%8B-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85-variables_ru)

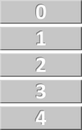
1. **Sub** variables()
2. 'Пример декларирования переменной
3. **Dim** var1 **As** **String**
5. 'Пример декларирования 1-но мерного массива
6. **Dim** array1(4) **As** **String**
8. 'Пример декларирования 2-х мерного массива
9. **Dim** array2(4, 3) **As** **String**
11. 'Пример декларирования 3-х мерного массива
12. **Dim** array3(4, 3, 2) **As** **String**
13. **End** **Sub**

**Одномерный массив**

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba6-2-%D1%82%D0%B8%D0%BF%D1%8B-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85-variables_ru)

1. **Dim** array1(4) **As** **String**

Одномерный массив можно представить как таблицу, которая состоит из одного столбца. Массив array1 (4) является одномерным, в который можно поместить 5 записей. Почему 5? Потому, что номера записей в массиве начинаются с нуля (0, 1, 2, 3, 4).

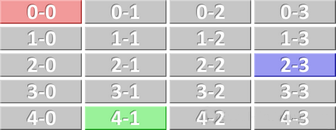


Теперь давайте рассмотрим 2-х мерный массив:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba6-2-%D1%82%D0%B8%D0%BF%D1%8B-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85-variables_ru)

1. **Dim** array2(4, 3) **As** **String**

Это уже будет выглядеть, как таблица с 5 строками и 4 столбцами:



'Присвоение значений для закрашенных ячеек

array2(0, 0) = "Значение в красной ячейке"

array2(4, 1) = "Значение в зеленой ячейке"

array2(2, 3) = "Значение в синей ячейке"

**Константы**

Как и переменные, константы могут использоваться для сохранения значений, однако отличие в том, что значения не могут изменяться. Мы можем добавить константу, чтобы избежать повторения значение, например,***13.14***:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba6-2-%D1%82%D0%B8%D0%BF%D1%8B-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85-variables_ru)

1. **Sub** const\_example()
2. Cells(1, 1) = Cells(1, 2) \* 13.14
3. Cells(2, 1) = Cells(2, 2) \* 13.14
4. Cells(3, 1) = Cells(3, 2) \* 13.14
5. Cells(4, 1) = Cells(4, 2) \* 13.14
6. Cells(5, 1) = Cells(5, 2) \* 13.14
7. **End** **Sub**

Это делает код легче для восприятия и редактирования. Также позволяет достаточно просто изменить само значение константы:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba6-2-%D1%82%D0%B8%D0%BF%D1%8B-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85-variables_ru)

1. **Sub** const\_example()
2. 'Декларирование константы + присвоение значения
3. **Const** ANNUAL\_RATE **As** **Double** = 13.14
5. Cells(1, 1) = Cells(1, 2) \* ANNUAL\_RATE
6. Cells(2, 1) = Cells(2, 2) \* ANNUAL\_RATE
7. Cells(3, 1) = Cells(3, 2) \* ANNUAL\_RATE
8. Cells(4, 1) = Cells(4, 2) \* ANNUAL\_RATE
9. Cells(5, 1) = Cells(5, 2) \* ANNUAL\_RATE
10. **End** **Sub**

Область видимости переменных

Если переменная декларируется в начале процедуры (Sub), то она может быть использована только в данной процедуре. Значение переменной не будет больше доступно после выполнения процедуры.

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba6-2-%D1%82%D0%B8%D0%BF%D1%8B-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85-variables_ru)

1. **Sub** procedure1()
2. **Dim** var1 **As** **Integer**
3. ' => Переменная действительна только в этой процедуре
4. **End** **Sub**
6. **Sub** procedure2()
7. ' => var1 не может быть использована здесь
8. **End** **Sub**

Для того, чтобы использовать переменную в любой процедуре модуля, нам нужно лишь задекларировать ее в начале модуля. И если вы задекларируете переменную таким образом, она будет доступна вплоть до закрытия робочей книги.

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba6-2-%D1%82%D0%B8%D0%BF%D1%8B-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85-variables_ru)

1. **Dim** var1 **As** **Integer**
3. **Sub** procedure1()
4. ' => var1 может быть использована здесь
5. **End** **Sub**
7. **Sub** procedure2()
8. ' => var1 может быть также использована и здесь
9. **End** **Sub**

Если вы захотите использовать одну и ту же переменную во всех модулях книги, стоит только в предыдущем примере заменить **Dim** на **Global**:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba6-2-%D1%82%D0%B8%D0%BF%D1%8B-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85-variables_ru)

1. Global var1 **As** **Integer**

Чтобы использовать переменную после выполнения процедуры, в которой она появилась, замените **Dim** на **Static**:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba6-2-%D1%82%D0%B8%D0%BF%D1%8B-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85-variables_ru)

1. **Sub** procedure1()
2. **Static** var1 **As** **Integer**
3. **End** **Sub**

Чтобы использовать значения всех переменных в процедуре, добавьте **Static** перед **Sub**:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba6-2-%D1%82%D0%B8%D0%BF%D1%8B-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85-variables_ru)

1. **Static** **Sub** procedure1()
2. **Dim** var1 **As** **Integer**
3. **End** **Sub**

**Создание собственного типа переменных**

Вот краткий пример, как можно создать свой собственный тип:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba6-2-%D1%82%D0%B8%D0%BF%D1%8B-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85-variables_ru)

1. 'Создание типа переменной
2. Type customers
3. last\_name **As** **String**
4. first\_name **As** **String**
5. **End** Type
7. **Sub** variables()
8. 'Объявление переменной
9. **Dim** cust1 **As** customers
11. 'Присвоение значения для cust1
12. cust1.last\_name = "Smith"
13. cust1.first\_name = "John"
15. 'Пример использования
16. MsgBox cust1.last\_name & " " & cust1.first\_name

**VBA-Урок 7.1. Условия (Conditions)**

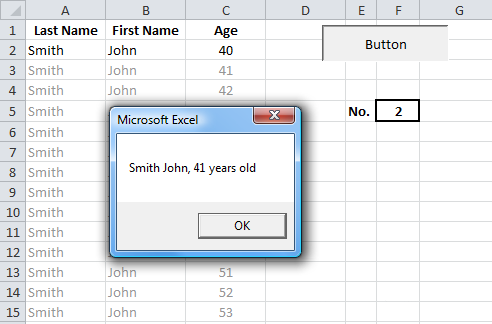
Условия являются очень полезными при программировании, поскольку позволяют нам выполнять действия, в зависимости от установленных критериев (используется такой же принцип как и в **IF**функции Excel).

Наиболее важной функцией, которая задает условие является **IF**и сейчас мы посмотрим, как она работает:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba7-%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%B8%D1%8F-conditions_ru)

1. **If** [УСЛОВИЕ ЗДЕСЬ] **Then** '=> ЕСЛИ условие верно, ТОГДА
2. 'Инструкции, если "правда"
3. **Else** '=> В ПРОТИВНОМ СЛУЧАЕ
4. 'Инструкции, если "ложь"
5. **End** **If**

Давайте перейдем к практике и вернемся к примеру, который мы использовали в уроке с переменными. Цель этой процедуры была в том, чтобы открыть диалоговое окно, которое бы содержало значение из строки, указанного в ячейке **F5**:



Если вы введете букву в ячейку **F5**, это повлечет ошибку. Мы хотим предотвратить это.

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba7-%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%B8%D1%8F-conditions_ru)

1. **Sub** variables ()
2. 'Декларирование переменных
3. **Dim** last\_name **As** **String**, first\_name **As** **String**, age **As** **Integer**, row\_number **As** **Integer**
5. 'Присвоение значений переменным
6. row\_number = Range("F5") + 1
7. last\_name = Cells(row\_number, 1)
8. first\_name = Cells(row\_number, 2)
9. age = Cells(row\_number, 3)
11. 'Диалоговое окно
12. MsgBox last\_name & " " & first\_name & "," & age & " лет"
13. **End** **Sub**

Давайте добавим условие, которое будет проверять - является ли введенное значение в ячейку **F5**числом, перед тем, как код будет выполнен.

Мы воспользуемся функцией **IsNumeric**для проверки условия:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba7-%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%B8%D1%8F-conditions_ru)

1. **Sub** variables ()
3. 'Если значение в скобках (ячейка F5) является числовым (И ПОЭТОМУ УСЛОВИЕ IF ЕСТЬ ВЕРНЫМ) тогда
4. 'выполнить инструкции, которые следуют после THEN
5. **If** IsNumeric (Range ("F5")) **Then**
7. 'Декларирование переменных
8. **Dim** last\_name **As** **String**, first\_name **As** **String**, age **As** **Integer**, row\_number **As** **Integer**
9. 'Присвоение значений переменным
10. row\_number = Range ("F5") + 1
11. last\_name = Cells (row\_number, 1)
12. first\_name = Cells (row\_number, 2)
13. age = Cells (row\_number, 3)
14. 'Диалоговое окно
15. MsgBox last\_name & " " & first\_name & "," & age & " лет"
17. **End** **If**
19. **End** **Sub**

Нам также нужно прописать инструкции, если поставленное нами условие не выполнится:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba7-%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%B8%D1%8F-conditions_ru)

1. **Sub** variables ()
3. **If** IsNumeric (Range ("F5")) **Then** 'Если условие выполняется
5. 'Декларирование переменных
6. **Dim** last\_name **As** **String**, first\_name **As** **String**, age **As** **Integer**, row\_number **As** **Integer**
7. 'Присвоение значений переменным
8. row\_number = Range ("F5") + 1
9. last\_name = Cells (row\_number, 1)
10. first\_name = Cells (row\_number, 2)
11. age = Cells (row\_number, 3)
12. 'Диалоговое окно
13. MsgBox last\_name & " " & first\_name & "," & age & " лет"
15. **Else** 'Если условие не выполняется
17. 'Диалоговое окно: предупреждение
18. MsgBox "Введенное значение " & Range ("F5") & " не является верным!"
19. 'Удаление содержимого ячейки F5
20. Range ("F5").ClearContents
22. **End** **If**
24. **End** **Sub**

Теперь нечисловое значения не повлечет никаких проблем.

Работая с нашим массивом, который содержит 16 строк данных, наш следующий шаг будет в проверке является ли переменная row\_number: "больше чем или равно 2" и "меньше чем или равно 17".

Но сначала взглянем на операторы сравнения:

|  |  |
| --- | --- |
| = | равно |
| <> | не равно |
| < | менше чем |
| <= | меньше или равно |
| > | больше чем |
| >= | больше или равно |

и эти полезные операторы:

|  |  |
| --- | --- |
| AND | [условие1] AND [условие2] Два условия должны быть выполнены |
| OR | [условие1] OR [условие2] Не менее 1 из 2 условий должно быть выполнено |
| NOT | NOT [условие1] Условие не должно выполниться |

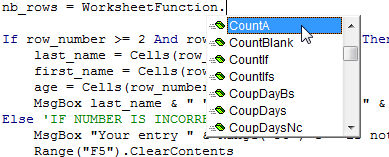
Теперь давайте добавим одно из выше указанных условий **AND**между операторов сравнения:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba7-%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%B8%D1%8F-conditions_ru)

1. **Sub** variables ()
2. **If** IsNumeric (Range ("F5")) **Then** 'Если числовое значение
3. **Dim** last\_name **As** **String**, first\_name **As** **String**, age **As** **Integer**, row\_number **As** **Integer**
4. row\_number = Range ("F5") + 1
6. **If** row\_number> = 2 **And** row\_number <= 17 **Then** 'Если верное число
7. last\_name = Cells (row\_number, 1)
8. first\_name = Cells (row\_number, 2)
9. age = Cells (row\_number, 3)
10. MsgBox last\_name & " " & first\_name & "," & age & " лет"
11. **Else** 'Если число не является корректным
12. MsgBox "Введенное число" & Range ("F5") & " не является корректным!"
13. Range ("F5").ClearContents
14. **End** **If**
16. **Else** 'Если это не числовое значение
17. MsgBox "Введенное значение " & Range ("F5") & " не является верным!"
18. Range ("F5").ClearContents
19. **End** **If**
20. **End** **Sub**

Если мы хотим сделать наш макрос более практичным, мы можем заменить **17**на переменную, которая бы содержала количество строк. Это бы позволило нам добавлять и удалять строки из массива без необходимости изменять этот лимит каждый раз.

Для того, чтобы сделать это, мы должны создать переменную **nb\_rows**и добавить эту функцию.



В этом случае, мы используем функцию **WorksheetFunction.CountA**, которая является аналогом функции **COUNTA**в самом Excel.

Мы хотим, чтобы эта функция подсчитала количество непустых ячеек в первой колонке по записала полученное значение в переменную **nb\_rows**:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba7-%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%B8%D1%8F-conditions_ru)

1. **Sub** variables ()
2. **If** IsNumeric (Range ("F5")) **Then** 'ЕСЛИ ЧИСЛО
3. **Dim** last\_name **As** **String**, first\_name **As** **String**, age **As** **Integer**, row\_number **As** **Integer**
4. **Dim** nb\_rows **As** **Integer**
6. row\_number = Range ("F5") + 1
7. nb\_rows = WorksheetFunction.CountA (Range ("A:A")) 'Функция подсчета количества строк
9. **If** row\_number > = 2 **And** row\_number < = nb\_rows **Then** 'ЕСЛИ ВЕРНОЕ ЧИСЛО
10. last\_name = Cells (row\_number,1)
11. first\_name = Cells (row\_number,2)
12. age = Cells (row\_number,3)
13. MsgBox last\_name & " " & first\_name & "," & age & " лет"
14. **Else** 'ЕСЛИ ЧИСЛО НЕКОРРЕКТНО
15. MsgBox "Введенное число " & Range("F5") & " не является корректным!"
16. Range("F5").ClearContents
17. **End** **If**
19. **Else** 'ЕСЛИ НЕ ЕСТЬ ЧИСЛОМ
20. MsgBox "Введенное значение " & Range ("F5") & " не является верным!"
21. Range("F5").ClearContents
22. **End** **If**
23. **End** **Sub**

**ElseIf**

**ElseIf** дает возможность добавлять дополнительные условия после IF команды:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba7-%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%B8%D1%8F-conditions_ru)

1. **If** [УСЛОВИЕ 1] **Then** '=> ЕСЛИ условие 1 верно, ТОГДА
2. 'Инструкции 1
3. **ElseIf** [УСЛОВИЕ 2] **Then** '=> ЕСЛИ условие 1 неверно, но условие 2 верно, ТОГДА
4. 'Инструкции 2
5. **Else** '=> ИНАЧЕ
6. 'Инструкции 3
7. **End** **If**

Если *УСЛОВИЕ 1*выполняется, **Инструкция 1**будет выполнена и покинет оператор **IF**(который начинается с **IF**и заканчивается **End If**). Если *УСЛОВИЕ 2*принимает значение "***ложь***", тогда будет выполнена **Инструкция 2**, и если она в свою очередь возвращает "***ложь***", тогда **Инструкция 3**(под **Else**) будет выполнена.

Далее есть пример с оценками от 1 до 6 в ячейке **A1**и комментарием к этим оценкам в ячейке **B1**:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba7-%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%B8%D1%8F-conditions_ru)

1. **Sub** scores\_comment ()
2. 'Переменные
3. **Dim** note **As** **Integer**, score\_comment **As** **String**
4. note = Range ("A1")
6. 'Комментарии, основанные на полученной оценке
7. **If** note = 6 **Then**
8. score\_comment = "Великолепный бал!"
9. **ElseIf** note = 5 **Then**
10. score\_comment = "Хороший бал"
11. **ElseIf** note = 4 **Then**
12. score\_comment = "Удовлетворительный бал"
13. **ElseIf** note = 3 **Then**
14. score\_comment = "Неудовлетворительный бал"
15. **ElseIf** note = 2 **Then**
16. score\_comment = "Плохой бал"
17. **ElseIf** note = 1 **Then**
18. score\_comment = "Ужасный бал"
19. **Else**
20. score\_comment = "Нулевой бал"
21. **End** **If**
23. 'Комментарий в ячейке B1
24. Range ("B1") = score\_comment
25. **End** **Sub**

**Select**

Существует альтернатива использованию **If**со многими **ElseIf**инструкциями, а именно команда **Select**, которая больше подходит к такого рода ситуаций.

Рассмотрим пример макроса с оператором **Select**:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba7-%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%B8%D1%8F-conditions_ru)

1. **Sub** scores\_comment ()
2. 'Переменные
3. **Dim** note **As** **Integer** , score\_comment **As** **String**
4. note = Range ("A1")
6. 'Комментарии , основанные на полученной оценке
7. **Select** **Case** note '< = оценка теста (баллы)
8. **Case** **Is** = 6 '< = если значение = 6
9. score\_comment = "Великолепный бал!"
10. **Case** **Is** = 5 '< = если значение = 5
11. score\_comment = "Хороший бал"
12. **Case** **Is** = 4 '< = если значение = 4
13. score\_comment = "Удовлетворительный бал"
14. **Case** **Is** = 3 '< = если значение = 3
15. score\_comment = "Неудовлетворительный бал"
16. **Case** **Is** = 2 '< = если значение = 2
17. score\_comment = "Плохой бал"
18. **Case** **Is** = 1 '< = если значение = 1
19. score\_comment = "Ужасный бал"
20. **Case** **Else**   '< = если значение не является равным любому из вышеуказанных
21. score\_comment = "Нулевой бал"
22. **End** **Select**
24. 'Комментарий в ячейке B1
25. Range ("B1") = score\_comment
26. **End** **Sub**

Стоит отметить, что мы также могли использовать и другие операторы сравнения:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba7-%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%B8%D1%8F-conditions_ru)

1. **Case** **Is** > = 6 'если значение> = 6

Примеры с различными значениями:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba7-%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%B8%D1%8F-conditions_ru)

1. **Case** **Is** = 6, 7 'если значение = 6 или 7
2. **Case** **Is** <> 6, 7 'если значение не равно 6 или 7

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba7-%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%B8%D1%8F-conditions_ru)

1. **Case** 6 **To** 10 'если значение = любому числу от 6 до 10

**VBA-Урок 7.2. Условия (Conditions)**

**Условие, что базируется на типе**

**IsNumeric**(функция, которую мы использовали в предыдущем уроке) возвращает **ПРАВДА**(**TRUE**), если значение является числом, и **ЛОЖЬ**(**FALSE**), если - не является числом:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba7-2-%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%B8%D1%8F-conditions_ru)

1. **If** IsNumeric (Range ("A1")) = **True** **Then** 'ЕСЛИ ЗНАЧЕНИЕ ЯВЛЯЕТСЯ ЧИСЛОМ ...

Следующий код дает такой же эффект, как и предыдущий (мы не должны включать = **True**, так как сама конструкция является проверкой условия):

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba7-2-%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%B8%D1%8F-conditions_ru)

1. **If** IsNumeric (Range ("A1")) = **True** **Then** 'ЕСЛИ ЗНАЧЕНИЕ ЯВЛЯЕТСЯ ЧИСЛОМ...

Если мы хотим проверить является ли значение не числом, то мы можем это сделать двумя способами:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba7-2-%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%B8%D1%8F-conditions_ru)

1. **If** IsNumeric (Range ("A1")) = **False** **Then** 'ЕСЛИ ЗНАЧЕНИЕ НЕ ЯВЛЯЕТСЯ ЧИСЛОМ ...

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba7-2-%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%B8%D1%8F-conditions_ru)

1. **If** **Not** IsNumeric (Range ("A1")) **Then** 'ЕСЛИ ЗНАЧЕНИЕ НЕ ЯВЛЯЕТСЯ ЧИСЛОМ ...

Давайте рассмотрим еще несколько подобных **IsNumeric**функций:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba7-2-%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%B8%D1%8F-conditions_ru)

1. **If** IsDate (Range ("A1")) **Then** 'ЕСЛИ ЗНАЧЕНИЕ ЯВЛЯЕТСЯ ДАТОЙ ...

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba7-2-%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%B8%D1%8F-conditions_ru)

1. **If** IsEmpty(Range("A1")) **Then** 'ЕСЛИ ПУСТО ...

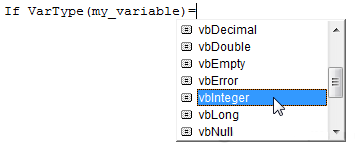
[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba7-2-%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%B8%D1%8F-conditions_ru)

1. **If** var\_object **Is** **Nothing** **Then** 'ЕСЛИ ОБЪЕКТ НЕ ОПРЕДЕЛЕН...

**Условие, базирующееся на типе переменной**

Чтобы выполнить команды, основанные на типе переменной (**Variant**), нам потребуется использовать функцию **VarType**.

Список типов переменных появится как только мы введем знак "**=**":



[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba7-2-%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%B8%D1%8F-conditions_ru)

1. **If** VarType(my\_variable) = vbInteger **Then** 'ЕСЛИ my\_variable является типом переменной Integer...

Значения констант:

|  |  |
| --- | --- |
| **Константа** | **Значение** |
| vbEmpty | 0 |
| vbNull | 1 |
| vbInteger | 2 |
| vbLong | 3 |
| vbSingle | 4 |
| vbDouble | 5 |
| vbCurrency | 6 |
| vbDate | 7 |
| vbString | 8 |
| vbObject | 9 |
| vbError | 10 |

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba7-2-%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%B8%D1%8F-conditions_ru)

1. **If** VarType(my\_variable) = vbInteger **Then** 'ЕСЛИ my\_variable является типом переменной Integer ...
2. 'Является идентичним до:
3. **If** VarType (my\_variable) = 2 **Then** 'ЕСЛИ my\_variable является типом переменной Integer ...

**Условие, базирующееся на сравнении двух текстовых строк**

Немного раньше мы использовали следующий фрагмент кода:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba7-2-%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%B8%D1%8F-conditions_ru)

1. my\_variable = "Example 12345"
3. **If** my\_variable = "Example 12345" **Then** ' => ПРАВДА (TRUE)

В этом случае две строки одинаковы, но если мы хотим проверить содержит ли переменная значение "12345" без учета других символов, то нам следует использовать команду **Like**и оператор **\***(звездочка) перед и после значения, что мы ищем.

Оператор **\***(звездочка) расшифровывается как: любой символ или набор символов:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba7-2-%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%B8%D1%8F-conditions_ru)

1. my\_variable = "Example 12345"
3. **If** my\_variable **Like** "\*12345\*" **Then** ' => ПРАВДА (TRUE)

Оператор **#** (решетка) расшифровывается как: любой числовой единичный символ от 0 до 9:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba7-2-%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%B8%D1%8F-conditions_ru)

1. my\_variable = "Example 12345"
3. **If** my\_variable **Like** "Example 12###" **Then** ' => ПРАВДА (TRUE)

Оператор **?** (вопросительный знак) расшифровывается как: любой единичный символ:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba7-2-%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%B8%D1%8F-conditions_ru)

1. my\_variable = "Example 12345"
3. **If** my\_variable **Like** "?xample?1234?" **Then** ' => ПРАВДА (TRUE)

Мы также можем использовать определенные символы или набор символов таким же образом:

* **[abc]** расшифровывается как: любой один из следующих символов: ***a b c***
* **[a-g]** расшифровывается как: любой один из следующих символов:***a b c d e f g***
* **[369]** расшифровывается как: любой один из следующих символов: ***3 6 9***
* **[2-5]** расшифровывается как: любой один из следующих символов: ***2 3 4 5***
* **[?\*#]** расшифровывается как: любой один из следующих символов: ***? \* #***

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba7-2-%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%B8%D1%8F-conditions_ru)

1. my\_variable = "Example 12345"
3. **If** my\_variable **Like** "[DEF]xample 1234[4-7]" **Then** ' => ПРАВДА (TRUE)

Оператор **!** (восклицательный знак) добавлен после знака **[** будет означать: любой символ, не включенный в квадратные скобки:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba7-2-%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%B8%D1%8F-conditions_ru)

1. my\_variable = "Example 12345"
3. **If** my\_variable **Like** "[!GHIJ]xample 1234[!6-9]" **Then** ' => ПРАВДА (TRUE)

Комментарий : В данном контексте, символы с большой буквы не являются эквивалентными таким же символам с маленькой буквы. Если вы не хотите делать различия между большими и маленькими символами, тогда просто пропишите команду **Option Compare Text**в начале модуля.

**VBA-Урок 8.1. Циклы (Loops)**

Циклы осуществляют повторение инструкций несколько раз, и могут сэкономить много времени.

Следующий код вставляет последовательность чисел в каждую ячейку колонки A (от 1 до 12):

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba8-%D1%86%D0%B8%D0%BA%D0%BB%D1%8B-loops_ru)

1. **Sub** while\_loop()
3. Cells(1, 1) = 1
4. Cells(2, 1) = 2
5. Cells(3, 1) = 3
6. Cells(4, 1) = 4
7. Cells(5, 1) = 5
8. Cells(6, 1) = 6
9. Cells(7, 1) = 7
10. Cells(8, 1) = 8
11. Cells(9, 1) = 9
12. Cells(10, 1) = 10
13. Cells(11, 1) = 11
14. Cells(12, 1) = 12
16. **End** **Sub**

Этот код очень повторяющийся...

Представьте, если бы мы должны были вставлять последовательные числа в сотни ячеек вместо только 12... Теперь вы понимаете почему циклы являются очень полезными.

**WHILE**

Приведем пример пустого цикла **While**:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba8-%D1%86%D0%B8%D0%BA%D0%BB%D1%8B-loops_ru)

1. **Sub** while\_loop()
3. **While** [условие]
4. 'Инструкции
5. Wend
7. **End** **Sub**

Пока условие будет действительным, инструкции в цикле будут продолжать выполняться (но будьте осторожны, чтобы не сделать бесконечный цикл).

Ниже есть повторяющийся макрос, который был рассмотрен сначала, преобразован в **While**цикл:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba8-%D1%86%D0%B8%D0%BA%D0%BB%D1%8B-loops_ru)

1. **Sub** while\_loop ()
3. **Dim** num **As** **Integer**
4. num = 1 'Начальное число (в данном случае, это номер строки и значение, которое проставляется)
6. **While** num <= 12 'Пока переменная num будет <= 12, инструкции будут выполняться
7. Cells (num, 1) = num 'Нумерация
8. num = num + 1 'Число увеличивается на 1 при каждом проходе цикла
9. Wend
11. **End** **Sub**

Используя этот цикл в макросе, все чтобы мы должны были бы сделать, если бы нам надо было 400 строк вместо 12, это заменить 12 на 400.

**DO LOOP**

Другой способ написать цикл, который работает так же, как **While Wend**(выполняется так долго, пока действительно определенное условие, содержащееся в **While**команде):

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba8-%D1%86%D0%B8%D0%BA%D0%BB%D1%8B-loops_ru)

1. **Sub** do\_while\_loop()
3. **Do** **While** [условие]
4. 'Инструкции
5. **Loop**
7. **End** **Sub**

В следующем варианте показано, что условие может быть прописано в конце цикла **Do Loop**, что означает, что инструкции будут точно выполнены по меньшей мере один раз:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba8-%D1%86%D0%B8%D0%BA%D0%BB%D1%8B-loops_ru)

1. **Sub** do\_while\_loop()
3. **Do**
4. 'Инструкции
5. **Loop** **While** [условие]
7. **End** **Sub**

Вместо того, чтобы повторять цикл пока выполняется условие, можно выйти из цикла по условию, если мы заменим **While** на **Until**:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba8-%D1%86%D0%B8%D0%BA%D0%BB%D1%8B-loops_ru)

1. **Sub** do\_while\_loop()
3. **Do** **Until** [условие]
4. 'Инструкции
5. **Loop**
7. **End** **Sub**

**FOR**

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba8-%D1%86%D0%B8%D0%BA%D0%BB%D1%8B-loops_ru)

1. **Sub** for\_loop()
3. **For** i = 1 **To** 5
4. 'Инструкции
5. **Next**
7. **End** **Sub**

Цикл **For**будет повторяться здесь 5 раз.

При каждом повторении этого цикла, переменная **i**автоматически увеличивается на 1:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba8-%D1%86%D0%B8%D0%BA%D0%BB%D1%8B-loops_ru)

1. **Sub** for\_loop()
3. **For** i = 1 **To** 5
4. MsgBox i
5. **Next**
7. **End** **Sub**

**Ранний выход из цикла**

Возможно выйти из цикла **For**заранее, воспользовавшись следующей инструкцией:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba8-%D1%86%D0%B8%D0%BA%D0%BB%D1%8B-loops_ru)

1. **Exit** **For** 'Выход из цикла For

Давайте рассмотрим пример, который наглядно покажет:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba8-%D1%86%D0%B8%D0%BA%D0%BB%D1%8B-loops_ru)

1. **Sub** for\_loop ()
2. **Dim** max\_loops **As** **Integer**
3. max\_loops = Range ("A1") 'В A1: мы указали лимит количества повторений
5. **For** i = 1 **To** 7 'Ожидаемое количество циклов: 7
6. **If** i> max\_loops **Then** 'Если A1 является пустой или содержит число < 7, уменьшить количество повторений
7. **Exit** **For** 'Если условие ПРАВДА, выход из цикла For loop
8. **End** **If**
10. MsgBox i
11. **Next**
13. **End** **Sub**

Другие операторы **Exit**:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba8-%D1%86%D0%B8%D0%BA%D0%BB%D1%8B-loops_ru)

1. **Exit** **Do** 'Выход из цикла Do Loop

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba8-%D1%86%D0%B8%D0%BA%D0%BB%D1%8B-loops_ru)

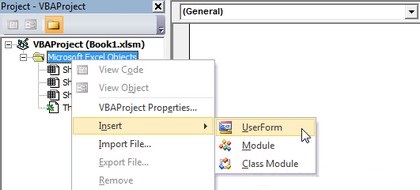
1. **Exit** **Sub** 'Выход из процедуры

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba8-%D1%86%D0%B8%D0%BA%D0%BB%D1%8B-loops_ru)

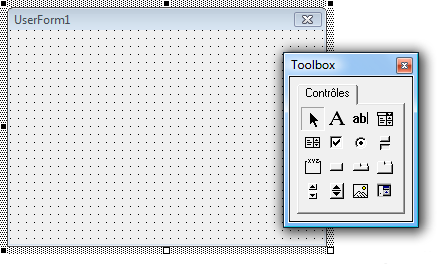
1. **Exit** **Function** 'Выход из функции

**VBA-Урок 12.1. Пользовательские формы (UserForm)**

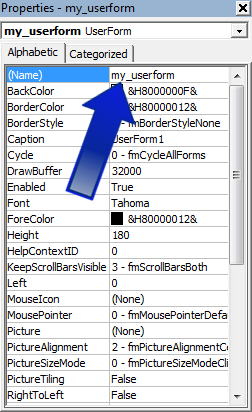
Чтобы добавить UserForm, нужно сделать то же самое, когда мы добавляем новый модуль:



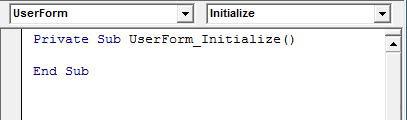
После этого появится форма (UserForm) и панель инструментов (Toolbox):



Если вы не видите окно **Свойства**(**Properties**), убедитесь, что оно отображено и тогда начните с изменения названия **Формы**(так, чтобы вы ее легко могли найти позже):

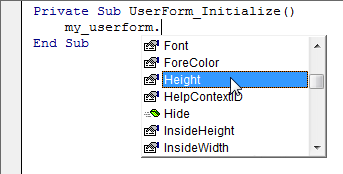


**Вид**(**UserForm**) имеет свои собственные события, подобно тому, как имеет рабочая книга или лист. Чтобы добавить событие, дважды щелкните по **Форме** (**UserForm**).



Теперь давайте создадим два события, чтобы посмотреть как это работает. Первое событие будет определять начальный размер **Формы**, и второе - увеличивать ее размеры на 50 пикселей, когда пользователь щелкает.

Событие **UserForm\_Initialize**будет срабатывать когда **Форма**запускается:



[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba12-1-%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D1%81%D0%BA%D0%B8%D0%B5-%D1%84%D0%BE%D1%80%D0%BC%D1%8B-userform_ru)

1. **Private** **Sub** UserForm\_Initialize ()
2. my\_userform.Height = 100
3. my\_userform.Width = 100
4. **End** **Sub**

Для упрощения кода, мы можем использовать **Me**вместо названия формы (так как этот код есть в **Форме**, с которой мы работаем):

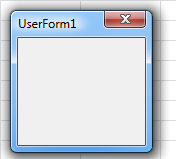
[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba12-1-%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D1%81%D0%BA%D0%B8%D0%B5-%D1%84%D0%BE%D1%80%D0%BC%D1%8B-userform_ru)

1. **Private** **Sub** UserForm\_Initialize()
2. **Me**.Height = 100
3. **Me**.Width = 100
4. **End** **Sub**

Второе событие будет возникать когда пользователь будут кликать по **Форме**:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba12-1-%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D1%81%D0%BA%D0%B8%D0%B5-%D1%84%D0%BE%D1%80%D0%BC%D1%8B-userform_ru)

1. **Private** **Sub** UserForm\_Initialize()
2. **Me**.Height = 100
3. **Me**.Width = 100
4. **End** **Sub**
6. **Private** **Sub** UserForm\_Click()
7. **Me**.Height = **Me**.Height + 50
8. **Me**.Width = **Me**.Width + 50
9. **End** **Sub**



**Запуск формы (UserForm)**

Чтобы запустить **Форму**в процедуре, используйте **Show**:

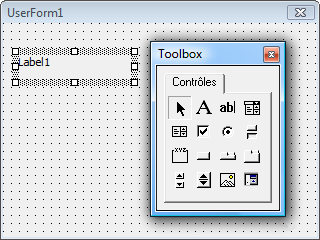
[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba12-1-%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D1%81%D0%BA%D0%B8%D0%B5-%D1%84%D0%BE%D1%80%D0%BC%D1%8B-userform_ru)

1. **Sub** show\_userform()
2. my\_userform.Show
3. **End** **Sub**

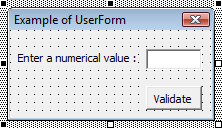
**VBA-Урок 12.2. Элементы управления (Controls)**

Элементы управления имеют все виды свойств и много событий связанных с ними, но сейчас мы лишь рассмотрим несколько из них, которые используются в **VBA**программировании:

Давайте начнем с добавления 3-х элементов управления - **Label**, **TextBox**и **CommandButton**:



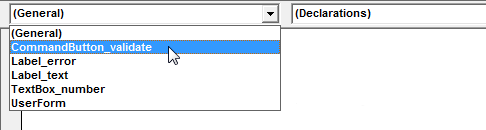
Давайте сейчас отредактируем название и свойства этого элемента управления (используя свойство **Caption**, которое содержит текст). Мы хотим получить следующий результат:



Сейчас, когда мы введем номер и нажмем ОК - ничего не произойдет.

Чтобы создать какое-то событие, мы начнем добавлять событие, которое введет значение из текстового окна в ячейки **А1**и закроет Форму.

Вы можете получить доступ к опциям, которые показаны ниже, дважды щелкнув по элементу управления:



Данный выпадающий список содержит различные элементы управления и эту Форму.

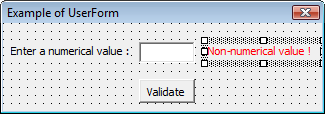
Выберите кнопку (Button) и событие **Click**:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba12-2-%D1%8D%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82%D1%8B-%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F-controls_ru)

1. **Private** **Sub** CommandButton\_validate\_Click()
3. Range("A1") = Textbox\_number.Value
4. 'Textbox\_number является названием текстового окна (text box)
5. 'Value является свойством, содержащим значения текстового окна
7. Unload **Me**
8. 'Unload закрывает форму (UserForm)
9. 'Мы используем Me вместо названия формы (потому что, этот код есть в середине UserForm, которую мы хотим закрыть)
10. **End** **Sub**

Введенное значение сейчас уже будет сохранено в ячейке **А1**перед закрытием этой **Формы**(**UserForm**).

Добавьте вторую подпись (Label) и измените следующие свойства: **Caption**, **Forecolor**(color: red) and **Visible**(False, чтобы скрыть элемент управления по умолчанию) :



Теперь давайте добавим событие, которое будет запускаться, когда значение в поле будет изменено. Событие отобразит сообщение об ошибке, если значение не является цифровым.

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba12-2-%D1%8D%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82%D1%8B-%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F-controls_ru)

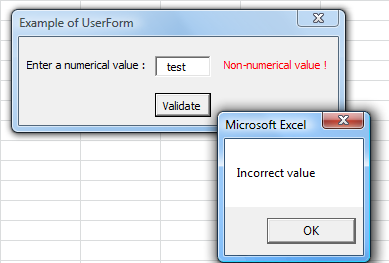
1. **Private** **Sub** Textbox\_number\_Change ()
2. **If** IsNumeric (Textbox\_number.Value) **Then** 'ЕСЛИ является цифровым значением ...
3. Label\_error.Visible = **False** 'Название (Label) скрытое
4. **Else** 'иначе ...
5. Label\_error.Visible = **True** 'Название отображается
6. **End** **If**
7. **End** **Sub**

Введенное значение будет тестироваться каждый раз при вводе очередного символа.

Нам еще нужно предотвратить валидацию формы, если значение не является числом.

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba12-2-%D1%8D%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82%D1%8B-%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F-controls_ru)

1. **Private** **Sub** CommandButton\_validate\_Click ()
2. **If** IsNumeric (Textbox\_number.Value) **Then** 'ЕСЛИ является цифровым значением ...
3. Range ("A1") = Textbox\_number.Value 'копируем в A1
4. Unload **Me** 'закрывает
5. **Else** 'иначе ...
6. MsgBox "Incorrect value" "Неправильное значение
7. **End** **If**
8. **End** **Sub**



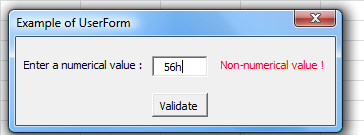
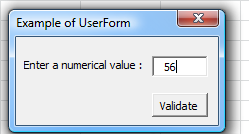
Для того, чтобы не оставлять правую сторону формы пустой, когда там нет сообщения об ошибке, мы можем уменьшить её размер, откорректировав свойство **Width**этой формы:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba12-2-%D1%8D%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82%D1%8B-%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F-controls_ru)

1. **Private** **Sub** Textbox\_number\_Change ()
2. **If** IsNumeric (Textbox\_number.Value) **Then** 'ЕСЛИ является цифровым значением ...
3. Label\_error.Visible = **False** 'Название (Label) скрытое
4. **Me**.Width = 156 'Ширина формы
5. **Else** 'иначе ...
6. Label\_error.Visible = **True** 'Название отображается
7. **Me**.Width = 244 'Ширина формы
8. **End** **If**
9. **End** **Sub**

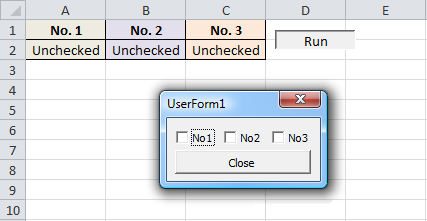
Вы можете скачать Excel файл с примером: [**userform1.xls**](http://moonexcel.com.ua/docs/userform1.xls)

Вот результат:



**Checkboxes**

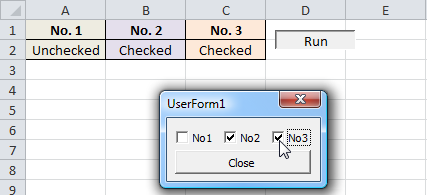
Далее пример как использовать **CheckBox**:



Когда чекбокс отмеченный/неотмеченный, значение связанной ячейки может быть изменено из-за использования события **Click**:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba12-2-%D1%8D%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82%D1%8B-%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F-controls_ru)

1. **Private** **Sub** CheckBox1\_Click () 'Номер 1
2. **If** CheckBox1.Value = **True** **Then** 'ЕСЛИ выбрано ...
3. Range ("A2") = "Checked"
4. **Else** 'ЕСЛИ не выбрано ...
5. Range ("A2") = "Unchecked"
6. **End** **If**
7. **End** **Sub**
9. **Private** **Sub** CheckBox2\_Click () 'Номер 2
10. **If** CheckBox2.Value = **True** **Then** 'ЕСЛИ выбрано ...
11. Range ("B2") = "Checked"
12. **Else** 'ЕСЛИ не выбрано ...
13. Range ("B2") = "Unchecked"
14. **End** **If**
15. **End** **Sub**
17. **Private** **Sub** CheckBox3\_Click () 'Номер 3
18. **If** CheckBox3.Value = **True** **Then** 'ЕСЛИ выбрано ...
19. Range ("C2") = "Checked"
20. **Else** 'ЕСЛИ не выбрано ...
21. Range ("C2") = "Unchecked"
22. **End** **If**
23. **End** **Sub**



В этом примере, чекбокс стартует неотмеченным, когда Форма открывается впервые.

Чтобы проверить бокс когда значения соответствующих ячеек является & quot; Отмечено & quot;, мы запустим проверку когда форму активировано, используя **UserForm\_Initialize:**

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba12-2-%D1%8D%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82%D1%8B-%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F-controls_ru)

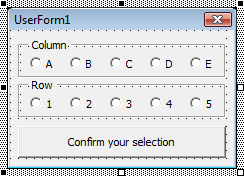
1. **Private** **Sub** UserForm\_Initialize() 'Если поставлена "галочка"
2. **If** Range("A2") = "Checked" **Then**
3. CheckBox1.Value = **True**
4. **End** **If**
6. **If** Range("B2") = "Checked" **Then**
7. CheckBox2.Value = **True**
8. **End** **If**
10. **If** Range("C2") = "Checked" **Then**
11. CheckBox3.Value = **True**
12. **End** **If**
13. **End** **Sub**

Здесь вы можете посмотреть пример в готовом Excel файле: [**userform2.xls**](http://moonexcel.com.ua/docs/userform2.xls)

**Кнопки опций (Option Buttons)**

Пользователь может выбрать только одну кнопку-опцию на "группу" в отличие от чекбоксов.

Чтобы создать "группу", сначала вставьте Frame, а затем **OptionButton**:



Здесь вы можете посмотреть пример в готовом Excel файле: [**userform3.xls**](http://moonexcel.com.ua/docs/userform3.xls)

Как только форма была отправлена, мы введем данные в ячейку, которая связана с названием колонки и row\_value chosen.

Для того, чтобы знать какие опции были выбраны, мы могли бы сделать то же самое что и в предыдущем примере (с чекбоксами), но мы используем цикл, чтобы уменьшить размер кода.

Мы собираемся использовать цикл **For Each**, тип цикла, который мы еще до этого не рассматривали. Этот цикл позволяет выполнять инструкции для каждого объекта в "группе объектов":

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba12-2-%D1%8D%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82%D1%8B-%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F-controls_ru)

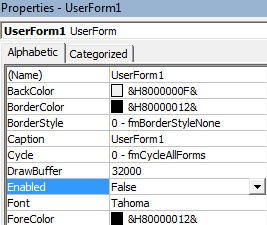
1. **Private** **Sub** CommandButton1\_Click ()
2. **Dim** column\_value **As** **String**, row\_value **As** **String**
4. 'Цикл для каждого элемента управления Frame\_column
5. **For** **Each** column\_button **In** Frame\_column.Controls
6. "Если значение элем. управления = **True** (тогда, если выбрано) ...
7. **If** column\_button.Value **Then**
8. 'Переменная "column\_value" принимает значение текста кнопки
9. column\_value = column\_button.Caption
10. **End** **If**
11. **Next**
13. 'Цикл для другого блока
14. **For** **Each** row\_button **In** Frame\_row.Controls
15. **If** row\_button.Value **Then**
16. row\_value = row\_button.Caption
17. **End** **If**
18. **Next**
20. Range (column\_value & row\_value) = "Cell chosen!" 'Ячейка выбрана
21. Unload **Me**
22. **End** **Sub**

Сейчас эта форма вводит значение "Ячейка выбрана!" в ячейку, которая была выбрана.

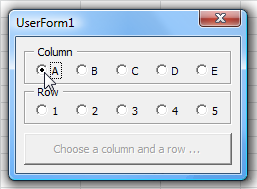
Чтобы избежать баги, нам необходимо проверить , что пользователь выбрал правильно с двух наборов опций.

В этом примере, когда форма еще не завершена, кнопка "Подтверждение" ("Confirm") появится в сером (будет деактивирована). Это не самое простое решение, но это хороший пример того, почему функции/процедуры является Полезные внутри формы (UserForm).

Измените текст и свойство **Enabled**, чтобы деактивировать кнопку.



Результат будет следующий:



В предыдущем коде мы использовали два For Each цикла, чтобы получить значение опционных кнопок (option buttons). Сейчас нам нужно использовать те же значения для кнопки "Подтверждение" ("Confirm") и событие **Click**для десяти опций.

Для этого нам не нужно копировать циклы для каждого события, мы вызовем их через функцию.

Начнем с предыдущего кода и модифицируя его, мы достигнем этого результата:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba12-2-%D1%8D%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82%D1%8B-%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F-controls_ru)

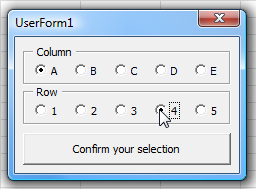
1. **Private** **Function** column\_value()
2. 'Функция, которая возвращает текстовое значение для выбранной кнопки (column\_value)
3. **For** **Each** column\_button **In** Frame\_column.Controls
4. **If** column\_button.Value **Then**
5. column\_value = column\_button.Caption
6. **End** **If**
7. **Next**
8. **End** **Function**
10. **Private** **Function** row\_value()
11. 'Функция, которая возвращает текстовое значение для выбранной кнопки (row\_value)
12. **For** **Each** row\_button **In** Frame\_row.Controls
13. **If** row\_button.Value **Then**
14. row\_value = row\_button.Caption
15. **End** **If**
16. **Next**
17. **End** **Function**
19. **Private** **Sub** CommandButton1\_Click () 'Действие, происходит при нажатии "Confirm your selection" ("Подтвердите свой выбор")
20. Range (column\_value & row\_value) = "Cell chosen!"
21. 'column\_value и row\_value являются значениями, которые возвращаються этими функциями
22. Unload **Me**
23. **End** **Sub**

Все что нам осталось сделать, это создать процедуру, которая проверяет , что кнопки опций были выбраны корректно (через вызов двух функций), и которые активируют эту кнопку когда нужно.

Опять же, проверка выполняется в отдельной процедуре, чтобы избежать копирования кода 10 раз для каждого события каждого опционной кнопки:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba12-2-%D1%8D%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82%D1%8B-%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F-controls_ru)

1. **Private** **Sub** activate\_button ()
2. 'Активация кнопки, если условие успешно проверено
3. **If** column\_value <> "" **And** row\_value <> "" **Then**
4. 'column\_value и row\_value являются значениями, которые возвращаються этими функциями
5. CommandButton1.Enabled = **True**
6. CommandButton1.Caption = "Confirm your selection"
7. **End** **If**
8. **End** **Sub**
10. **Private** **Sub** OptionButton11\_Click ()
11. activate\_button 'Запускаем процедуру "activate\_button"
12. **End** **Sub**
13. **Private** **Sub** OptionButton12\_Click()
14. activate\_button
15. **End** **Sub**
16. **Private** **Sub** OptionButton13\_Click()
17. activate\_button
18. **End** **Sub**
19. **Private** **Sub** OptionButton14\_Click()
20. activate\_button
21. **End** **Sub**
22. **Private** **Sub** OptionButton15\_Click()
23. activate\_button
24. **End** **Sub**
25. **Private** **Sub** OptionButton16\_Click()
26. activate\_button
27. **End** **Sub**
28. **Private** **Sub** OptionButton17\_Click()
29. activate\_button
30. **End** **Sub**
31. **Private** **Sub** OptionButton18\_Click()
32. activate\_button
33. **End** **Sub**
34. **Private** **Sub** OptionButton19\_Click()
35. activate\_button
36. **End** **Sub**
37. **Private** **Sub** OptionButton20\_Click()
38. activate\_button
39. **End** **Sub**



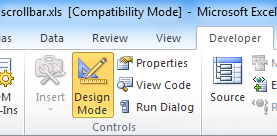
Здесь вы можете посмотреть пример в готовом Excel файле: [**userform3b.xls**](http://moonexcel.com.ua/docs/userform3b.xls)

**VBA-Урок 12.3. Элементы управления (Controls)**

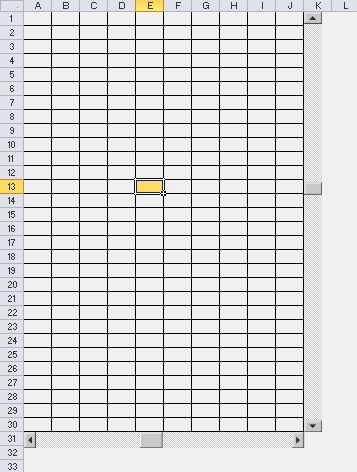
**Полоса прокрутки (ScrollBar)**

Элементы управления также могут использоваться вне форм. В следующем примере мы используем элементы управления прямо на рабочем листе.

Заметьте, что "**Режим дизайнера**" ("Design mode") должен быть активирован, для того чтобы изменять элемент управления, который расположен на рабочем листе (и так же должен быть деактивирован, для того чтобы использовать этот элемент управления).



Перед рассмотрением этого примера, давайте посмотрим на это:



Сейчас, мы хотим добавить цвет фона ячеек и выбрать её основываясь на позиции ползунка в определенной площади 30 строк на 10 колонок.

Свойства вертикального ползунка для прокрутки экрана.

* **Min**1
* **Max**30 (30 строк)
* **Value**: позиция ползунка (в данном случае, между 1 и 30)

Горизонтальный ползунок такой же, за исключением **Max**: 10.

Далее идет код, который будет запускаться каждый раз, когда идет изменение значения (**Value**) вертикального ползунка:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba12-3-%D1%8D%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82%D1%8B-%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F-controls_ru)

1. 'Серый фон в ячейках
2. Cells.Interior.Color = RGB (240, 240, 240)
4. 'Применяем цвет и выбираем ячейку
5. **With** Cells (ScrollBar\_vertical.Value, ActiveCell.Column) 'Определяем ячейку, используя Значение (Value)
6. .Interior.Color = RGB (255, 220, 100) 'Применяем оранжевый цвет
7. .**Select** 'Выбираем ячейку
8. **End** **With**

Этот код запускается когда события **Change**и **Scroll**возникают и запустят инструкции в не зависимости от того, по какой части ползунка мы щелкнули.

Далее код для вертикального ползунка:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba12-3-%D1%8D%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82%D1%8B-%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F-controls_ru)

1. **Private** **Sub** vertical\_bar ()
2. 'Применяем серый фон для ячеек
3. Cells.Interior.Color = RGB (240, 240, 240)
5. 'Применяем задний фон и выбираем ячейку
6. **With** Cells (ScrollBar\_vertical.Value, ActiveCell.Column)
7. .Interior.Color = RGB (255, 220, 100) 'Оранжевый
8. .**Select** 'Выбираем ячейку
9. **End** **With**
10. **End** **Sub**
12. **Private** **Sub** ScrollBar\_vertical\_Change()
13. vertical\_bar
14. **End** **Sub**
16. **Private** **Sub** ScrollBar\_vertical\_Scroll()
17. vertical\_bar
18. **End** **Sub**

А тут код для горизонтального ползунка:

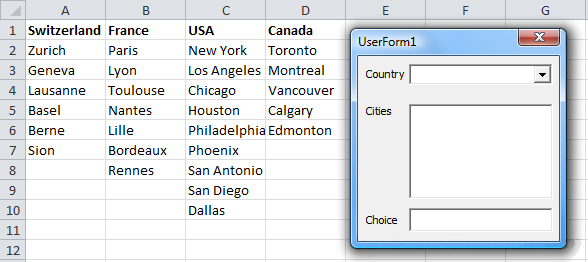
[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba12-3-%D1%8D%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82%D1%8B-%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F-controls_ru)

1. **Private** **Sub** horizontal\_bar ()
2. 'Применяем серый фон для ячеек
3. Cells.Interior.Color = RGB (240, 240, 240)
5. 'Применяем задний фон и выбираем ячейку
6. **With** Cells (ActiveCell.Row, ScrollBar\_horizontal.Value)
7. .Interior.Color = RGB (255, 220, 100) 'Оранжевый
8. .**Select** 'Выбираем ячейку
9. **End** **With**
10. **End** **Sub**
12. **Private** **Sub** ScrollBar\_horizontal\_Change()
13. horizontal\_bar
14. **End** **Sub**
16. **Private** **Sub** ScrollBar\_horizontal\_Scroll()
17. horizontal\_bar
18. **End** **Sub**

Вы можете скачать Excel файл с примером: [**scrollbar.xls**](http://moonexcel.com.ua/docs/scrollbar.xls)

**Выпадающий список и Окно со списком (ComboBox and ListBox)**

Это начальная точка нашего следующего примера:



Вы можете скачать Excel файл с примером: [**userform4.xls**](http://moonexcel.com.ua/docs/userform4.xls)

Когда форма запускается, мы хотим, чтобы 4 страны были загружены в выпадающий список (используя **AddItem** метод):

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba12-3-%D1%8D%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82%D1%8B-%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F-controls_ru)

1. **Private** **Sub** UserForm\_Initialize ()
2. **For** i = 1 **To** 4 '=> чтобы заполнить 4 страны
3. ComboBox\_Country.AddItem Cells (1, i) 'Добавляем значение ячеек от A1 до A4 используя цикл
4. **Next**
5. **End** **Sub**

Когда значение выпадающего списка меняется, мы хотим добавить города выбранной страны, используя цикл, подобен предыдущему.

Для того, чтобы сделать это, нам нужно номер колонки и номер строки.

Свойство **ListIndex** содержит порядковый номер выбранного элемента в выпадающем списке (в отличие от **Value**, которое содержит значение элемента списка). Заметьте, что **ListIndex** начинается с номера 0.

Номер колонки получается из:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba12-3-%D1%8D%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82%D1%8B-%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F-controls_ru)

1. column\_number = ComboBox\_Country.ListIndex + 1

Чтобы получить количество строк в выбранной колонке определенной страны, мы можем найти номер последней непустой ячейки:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba12-3-%D1%8D%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82%D1%8B-%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F-controls_ru)

1. rows\_number = Cells(1, column\_number).**End**(xlDown).Row

Используя эту информацию, теперь возможно создать цикл, чтобы добавить города в список:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba12-3-%D1%8D%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82%D1%8B-%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F-controls_ru)

1. **Private** **Sub** ComboBox\_Country\_Change ()
2. 'Очищаем список (в противном случае, города будут добавлены сразу)
3. ListBox\_Cities.Clear
5. **Dim** column\_number **As** **Integer**, rows\_number **As** **Integer**
7. 'Порядковый номер выбранного элемента (ListIndex начинается с 0):
8. column\_number = ComboBox\_Country.ListIndex + 1
9. 'Количество строк в выбранной колонке со странами:
10. rows\_number = Cells(1, column\_number).**End**(xlDown).Row
12. **For** i = 2 **To** rows\_number '=> наполнение списка городами
13. ListBox\_Cities.AddItem Cells(i, column\_number)
14. **Next**
15. **End** **Sub**

Примечание: Мы могли бы сократить код, но это бы сделало его менее читабельным:

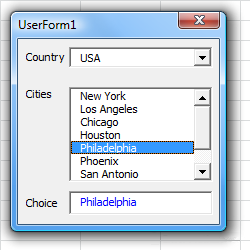
[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba12-3-%D1%8D%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82%D1%8B-%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F-controls_ru)

1. **Private** **Sub** ComboBox\_Country\_Change()
2. ListBox\_Cities.Clear
3. **For** i = 2 **To** Cells(1, ComboBox\_Country.ListIndex + 1).**End**(xlDown).Row
4. ListBox\_Cities.AddItem Cells(i, ComboBox\_Country.ListIndex + 1)
5. **Next**
6. **End** **Sub**

Город, который мы выберем, будет введен в текстовое поле:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba12-3-%D1%8D%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82%D1%8B-%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F-controls_ru)

1. **Private** **Sub** ListBox\_Cities\_Click()
2. TextBox\_Choice.Value = ListBox\_Cities.Value
3. **End** **Sub**



Здесь вы можете посмотреть пример в готовом Excel файле: [**userform4b.xls**](http://moonexcel.com.ua/docs/userform4b.xls)

**VBA-Урок 13.1. Использование массивов (Arrays)**

Массив - это по сути переменная, которая может хранить много значений. Мы уже рассматривали это в [**VBA-Урок 6.1. Типы данных**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba6-%D1%82%D0%B8%D0%BF%D1%8B-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85-variables_ru), но давайте сейчас окунемся еще глубже ...

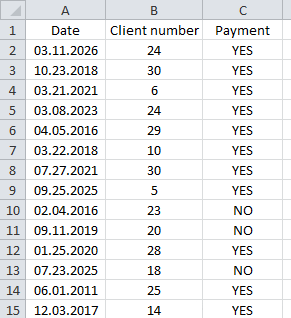
**Почему используются массивы?**

Представьте, что вы пытаетесь написать процедуру, в которой вам необходимо сохранять до 500 значений. Если вы должны будете создать 500 отдельных переменных, то это будет очень тяжело. В внутри массива, хранить и работать с этими значениями будет значительно легче.

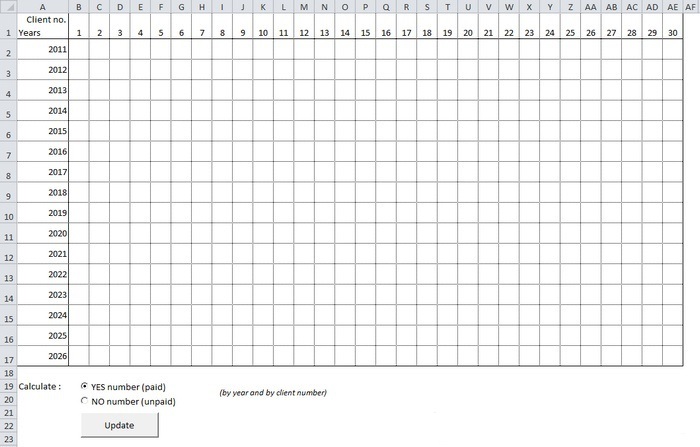
Вторая причина, чтобы использовать массивы - это их скорость. Занимает значительно меньше времени на считывание данных из массивов, чем из таблиц (созданных из ячеек) на рабочем листе Excel.

Итак, здесь есть пример, который покажет это нагдяно...

На первом рабочем листе ("DS") есть набор данных: 5000 строк на 3 столбца:

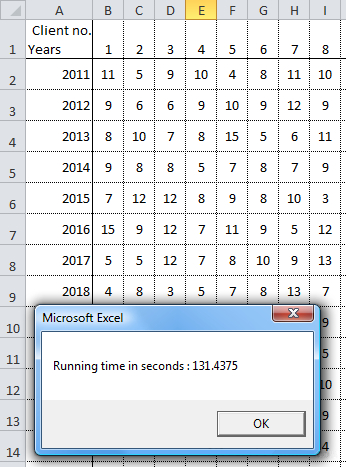


На втором листе вы найдете итоговую таблицу, которая учитывает все ответы "Да" ("YES") в разрезе лет и клиентов:

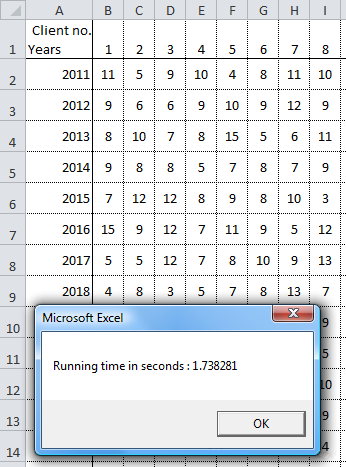


В данном случае, процедура будет использовать цикл, чтобы обработать набор данных и записать количество ответов "Да" для каждого года и каждого номера клиента, а затем введет эти данные в соответствующие ячейки.

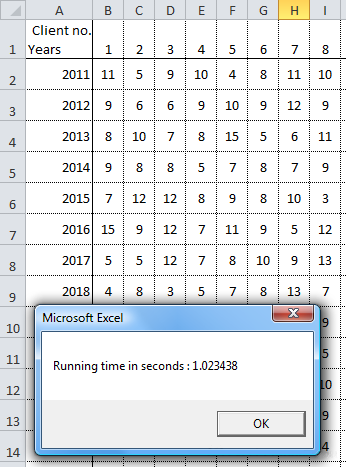
Без использования массивов, это бы заняло Excel **131.44 секунды**, чтобы выполнить эту процедуру:



Но для первого сохранения данных (с рабочего листа "DS") в массив и затем, осуществление таких же расчетов (используя массивы вместо набора данных с рабочего листа "DS"), займет лишь **1.74 секунды**:



Если мы решим оптимизировать нашу процедуру, сохраняя в массиве только данные, содержащие ответы "YES" (которых примерно 3/4 от всех данных), то это бы заняло лишь **1.02 секунды**:



Это хороший пример того, как использование массивов позволяет выполнять процедуру быстрее в 128 раз. Результат нашей оптимизации был бы даже лучшим, если бы мы работали с многими наборами данных одновременно.

Мы вернемся к деталям нашего примера в конце урока.

**Декларирование массивов**

Ниже есть несколько примеров декларирования массивов (если первые два не ясны вам, [**перечитайте это**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D1%8B-vba6-2-%D1%82%D0%B8%D0%BF%D0%B0-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85-variables_ru)):

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba13-1-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%BE%D0%B2-arrays_ru)

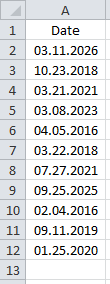
1. 'Пример декларирования одномерного массива
2. **Dim** array1 (4)
4. 'Пример декларирования двумерного массива
5. **Dim** array2 (6, 1)
7. 'Пример декларирования динамического массива
8. **Dim** array3()

Если вы не имеете возможности ввести фиксированные значения при декларировании массивов (так как они зависят от, например, размера набора данных), оставьте скобки пустыми.

Вы не должны декларировать тип данных (string, long и т.д.), хотя во многих случаях это будет замедлять выполнение вашей процедуры ...

**Хранение данных в массиве**

Давайте попробуем сохранить некоторые данные в массиве:



Мы хотим сохранить 11х1 значений в этом случае, поэтому нам необходимо создать одномерный массив:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba13-1-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%BE%D0%B2-arrays_ru)

1. 'Декларирование
2. **Dim** array\_example(10)

Не забывайте, что **нумерации элементов в массиве начинается с 0**(это стандарт в программировании, поэтому стоит сразу взять себе такую привычку, хотя, на самом деле, можно изменить этот подход в VBA).

Каждый элемент в массиве сейчас получит свое значение:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba13-1-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%BE%D0%B2-arrays_ru)

1. 'Сохранение значений в массив
2. array\_example(0) = Range("A2")
3. array\_example(1) = Range("A3")
4. array\_example(2) = Range("A4")
5. array\_example(3) = Range("A5")
6. array\_example(4) = Range("A6")
7. array\_example(5) = Range("A7")
8. array\_example(6) = Range("A8")
9. array\_example(7) = Range("A9")
10. array\_example(8) = Range("A10")
11. array\_example(9) = Range("A11")
12. array\_example(10) = Range("A12")

Вы можете работать или менять каждый элемент массива, будто это бы была бы обычная переменная.

Далее пример, в котором мы используем array\_example (8):

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba13-1-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%BE%D0%B2-arrays_ru)

1. **Sub** example ()
2. 'Декларирование
3. **Dim** array\_example (10)
5. 'Сохранение значений в массив
6. array\_example(0) = Range("A2")
7. array\_example(1) = Range("A3")
8. array\_example(2) = Range("A4")
9. array\_example(3) = Range("A5")
10. array\_example(4) = Range("A6")
11. array\_example(5) = Range("A7")
12. array\_example(6) = Range("A8")
13. array\_example(7) = Range("A9")
14. array\_example(8) = Range("A10")
15. array\_example(9) = Range("A11")
16. array\_example(10) = Range("A12")
18. 'Тест 1
19. MsgBox array\_example(8) '=> возвращает: 02.04.2016
21. 'Изменение одного из значений
22. array\_example (8) = Year (array\_example (8))
24. 'Тест 2
25. MsgBox array\_example (8) '=> возвращает : 2016
26. **End** **Sub**

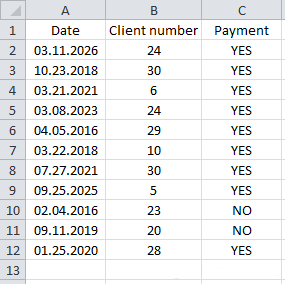
Цикл **For** был бы лучшим вариантом, чтобы сохранить массив быстрее:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba13-1-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%BE%D0%B2-arrays_ru)

1. 'Декларирование
2. **Dim** array\_example (10)
4. 'Сохранение значений в массив
5. **For** i = 0 **To** 10
6. array\_example(i) = Range("A" & i + 2)
7. **Next**

**VBA-Урок 13.2. Использование массивов (Arrays) (Продолжение)**

Для того, чтобы сохранить более одного столбца данных, нам необходима другая размерность массива. например:



Сохранение данных в двумерный массив:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba13-2-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%BE%D0%B2-arrays_ru)

1. 'Декларирование
2. **Dim** array\_example (10, 2) '11 x 3 "определенный" массив
4. 'Сохранение данных в массиве
5. **For** i = 0 **To** 10
6. array\_example(i, 0) = Range("A" & i + 2)
7. array\_example(i, 1) = Range("B" & i + 2)
8. array\_example(i, 2) = Range("C" & i + 2)
9. **Next**

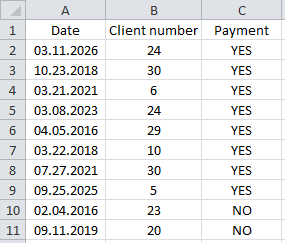
Далее есть несколько примеров работы с этими значениями:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba13-2-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%BE%D0%B2-arrays_ru)

1. MsgBox array\_example (0, 0) '=> возвращает: 03.11.2026
2. MsgBox array\_example (0, 1) '=> возвращает: 24
3. MsgBox array\_example (9, 2) '=> возвращает: NO
4. MsgBox array\_example (10, 2) '=> возвращает: YES

**Динамический массив**

Давайте представим на минутку, что нам нужно обновлять данные в нашем массиве на регулярной основе, и поэтому мы не можем присвоить фиксированные значения при декларировании ...



Чтобы узнать номер строки последней непустой ячейки, или другими словами, последнюю строку нашей базы данных, мы используем следующую формулу:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba13-2-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%BE%D0%B2-arrays_ru)

1. last\_row = Range("A1").**End**(xlDown).Row

Excel не принимает переменные в декларации.

Вместо этого, задекларируем динамический массив (используя пустые скобки), затем определим его размер используя Redim:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba13-2-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%BE%D0%B2-arrays_ru)

1. **Dim** array\_example ()
2. **ReDim** array\_example (last\_row - 2, 2)

Используя следующую процедуру, вы можете сохранить все строки вашего набора данных (таблицы) в нашем массиве:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba13-2-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%BE%D0%B2-arrays_ru)

1. **Dim** array\_example()
2. **ReDim** array\_example(last\_row - 2, 2)

**Ubound**

В предыдущем примере, последний номер в нашем массиве был last\_row - 2:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba13-2-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%BE%D0%B2-arrays_ru)

1. **For** i = 0 **To** last\_row - 2

Другой способ, чтобы определить последний номер в нашем массиве, мог бы быть через использование **Ubound**:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba13-2-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%BE%D0%B2-arrays_ru)

1. **For** i = 0 **To** UBound (array\_example)

Эта функция возвращает наибольший номер в массиве для выбранного измерения (первое измерение есть по-умолчанию).

Далее есть несколько примеров, которые прояснят это подробнее:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba13-2-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%BE%D0%B2-arrays_ru)

1. **Sub** example()
2. **Dim** array\_example(10, 2)
4. MsgBox UBound (array\_example) '=> возвращает: 10
5. MsgBox UBound (array\_example, 1) '=> возвращает: 10
6. MsgBox UBound (array\_example, 2) '=> возвращает : 2
7. **End** **Sub**

**Сохранение данных в диапазоне элементов массива**

Есть возможным заполнить массив значениями из диапазона ячеек на рабочем листе даже без использования цикла:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba13-2-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%BE%D0%B2-arrays_ru)

1. 'Декларирование
2. **Dim** array\_example (10, 2) '11 x 3 "определенный" массив
4. 'Сохранение данных в массиве
5. **For** i = 0 **To** 10
6. array\_example(i, 0) = Range("A" & i + 2)
7. array\_example(i, 1) = Range("B" & i + 2)
8. array\_example(i, 2) = Range("C" & i + 2)
9. **Next**

Предыдущий код может быть эффективно заменен этим:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba13-2-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%BE%D0%B2-arrays_ru)

1. 'Декларирование
2. **Dim** array\_example ()
4. 'Сохранение данных в массиве
5. array\_example = Range("A2:C12").Value

Хотя второй метод кажется более привлекательным, чем первый, будьте осторожны, что в большинстве случаев он может вам стоить больше времени на выполнение, чем первый.

Если вы сохраните данные в вашем массиве таким способом, первый номер будет 1, а не 0, что может привести к недоразумению ...

Далее в процессе увеличения кода, если вы решите сохранить только данные, которые соответствуют определенным критериям поиска в массиве (или проводить совсем другую операции), вам придется полностью переписать код, используя другую функцию цикла ...

Но этот второй метод является весьма полезным, если вам нужно сохранить все содержимое большого набора данных, потому что это быстрее, чем циклом (экономит примерно 0,2 секунд на каждые 15 000 записей).

**Массив (Array)**

Но если вам нужно создать массив, который имеет "фиксированное" содержание.

Одним из решений могло бы быть прописать значение строчка за строчкой:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba13-2-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%BE%D0%B2-arrays_ru)

1. **Dim** en(5)
3. en(0) = "IF"
4. en(1) = "VLOOKUP"
5. en(2) = "SUM"
6. en(3) = "COUNT"
7. en(4) = "ISNUMBER"
8. en(5) = "MID"

К счастью, вы можете упростить этот код, используя массив (**Array**):

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba13-2-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%BE%D0%B2-arrays_ru)

1. en = Array ("IF", "VLOOKUP", "SUM", "COUNT", "ISNUMBER", "MID")

Вот демонстрация использования функции **Replace**(это поможет вам понять следующий пример):

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba13-2-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%BE%D0%B2-arrays_ru)

1. **Sub** replace\_example()
2. **Dim** var\_translate **As** **String**
4. 'Текстовая строчка для этого примера
5. var\_translate = "Hello World!"
7. 'Замена "World" на "you" в текстовой строке
8. var\_translate = Replace (var\_translate, "World", "you")
10. 'Строка после замены
11. MsgBox var\_translate '=> возвращает "Hello you!"
12. **End** **Sub**

Теперь, если мы хотим заменить ряд значений другим набором данных, использование массивов и **Array**функции будет чрезвычайно полезным:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba13-2-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%BE%D0%B2-arrays_ru)

1. **Sub** translate () 'Упрощенный пример перевода формул с английского на французский
2. **Dim** var\_translate **As** **String**
4. 'Текстовая строчка для этого примера
5. var\_translate = "Formula to translate: SUM (IF (ISNUMBER (A1: E1), A1: E1,0))"
7. 'Два набора значений
8. en = Array ("IF", "VLOOKUP", "SUM", "COUNT", "ISNUMBER", "MID")
9. fr = Array ("SI", "RECHERCHEV", "SOMME", "NB", "ESTNUM", "STXT")
11. 'Замена" SI "на" IF "," RECHERVEV "на" VLOOKUP "и т.д.
12. **For** i = 0 **To** UBound (en)
13. var\_translate = Replace (var\_translate, en (i), fr (i))
14. **Next**
16. 'Строка после замены
17. MsgBox var\_translate '=> возвращает "Formula to translate : SOMME(SI(ESTNUM(A1:E1),A1:E1,0))"
18. **End** **Sub**

**Разделение (Split)**

Функция **Split**позволяет нам превратить символьную строку в массив.

Чтобы превратить строку в массив, сделайте следующее:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba13-2-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%BE%D0%B2-arrays_ru)

1. variable = "IF/VLOOKUP/SUM/COUNT/ISNUMBER/MID"

Используйте функцию **Split**и укажите разделитель:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba13-2-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%BE%D0%B2-arrays_ru)

1. en = Split (variable, "/")

Массив en вернет следующие значения:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba13-2-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%BE%D0%B2-arrays_ru)

1. MsgBox en(0)  '=> возвращает: IF
2. MsgBox en (1) '=> возвращает: VLOOKUP
3. MsgBox en (2) '=> возвращает: SUM
4. MsgBox en (3) '=> возвращает: COUNT
5. MsgBox en (4) '=> возвращает: ISNUMBER
6. MsgBox en (5) '=> возвращает: MID

Следующие 3 массивы также вернут те же значения:

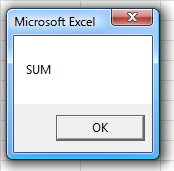
[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba13-2-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%BE%D0%B2-arrays_ru)

1. en = Array("IF", "VLOOKUP", "SUM", "COUNT", "ISNUMBER", "MID")
2. en = Split("IF,VLOOKUP,SUM,COUNT,ISNUMBER,MID", ",")
3. en = Split("IF VLOOKUP SUM COUNT ISNUMBER MID", " ")

Следующий пример возвращает третье значение в строке:

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba13-2-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%BE%D0%B2-arrays_ru)

1. MsgBox Split("IF,VLOOKUP,SUM,COUNT,ISNUMBER,MID", ",")(2) '=> возвращает : SUM



Обратной к **Split**является функция **Join**.

Эта функция собирает значение массива в строку.

[**view plaincopy to clipboardprint?**](http://moonexcel.com.ua/%D1%83%D1%80%D0%BE%D0%BA%D0%B8-vba13-2-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2%D0%BE%D0%B2-arrays_ru)

1. MsgBox Join (Array (1, 2, 3, 4, 5), "") '=> возвращает: 12345

