SF EDUCATION

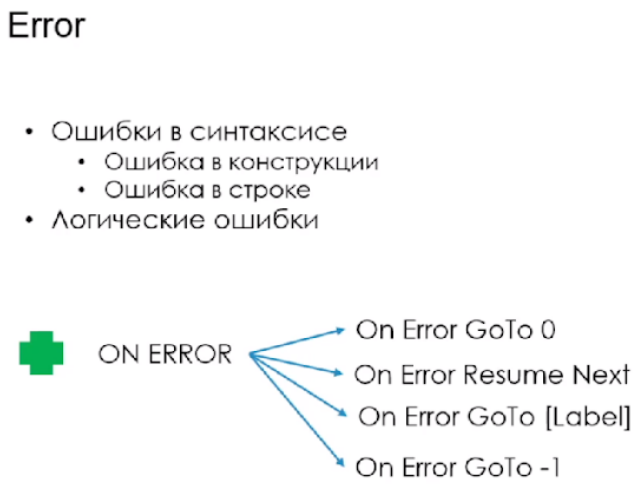
Программирование на VBA

Вводный урок. Функции и процедуры Часть 2

Содержание

1. Работа с ошибками
2. Ошибки в синтаксисе и логические
3. On Error

1. Работа с ошибками



Ошибки в Excel делятся на три типа: ошибки в синтаксисе – в конструкции и строке и ошибки логические.

Яркий пример логической ошибки – бесконечный цикл. То есть, мы заходим в какой-либо цикл и бесконечно выполняем структуры, находящиеся внутри, из которых невозможно выйти.

Такие ошибки бывают улавливаемые, например, когда при записи формулы в переменных случайно число поделили на ноль. В этом случае Excel нам укажет на ошибку.

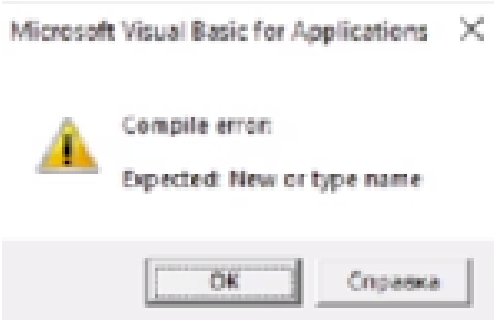
Работа с модулем ON ERROR

Для этого перейдем в Excel → Visual Basic → Format/

Самой стандартной ошибкой, которая очень часто встречается, является - **New of type name**.

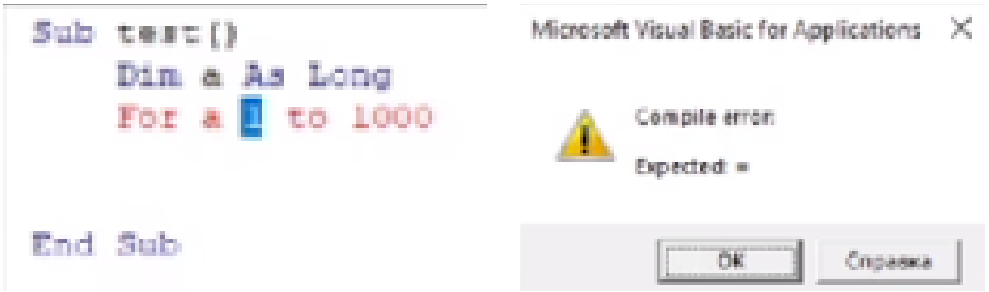
2. Ошибки в синтаксисе и логические

Например, создадим какую-нибудь процедуру, которую назовем «Test». Зададим любую переменную, которая будет появляться. И здесь мы случайно нажмем Enter. **Это ошибка в строке.**



Эта строка, где была допущена ошибка на экране подсвечивается красным. Excel сразу пытается скомпилировать строку, которую мы написали.
В этом одновременно его преимущество и недостаток.

Попробуем использовать нашу переменную от 1 до 1000. Еще одна ошибка:



Эта ошибка синтаксическая. При записи формулы мы забыли добавить «=».

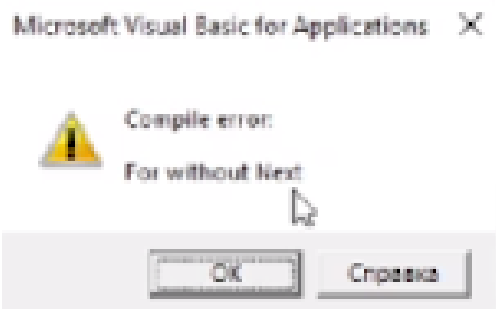
Допустим, мы хотим записать эти значения в какой-нибудь лист:

```
(General)
Sub test()
    Dim a As Long
    For a = 1 To 1000
        Worksheets(1).Range("A" & a) = a

End Sub
```

Нажав Ок, мы получаем синтаксическую ошибку, которая относится к структуре в виде цикла For. Она не завершена.

Excel информирует нас об ошибке, сообщая, что в нашей формуле отсутствует значение **Next**. **Эта синтаксическая ошибка в конструкции.** Исправив ее, макрос начнет работать.



Пример логической ошибки

Имеем формулу:

```
Sub test ()
Dim a As Long
For a = 0 to 249
    For j = 1 to 1000
    Worksheets (1). Range (“A” & a) = a
    a = a - 1
Next a
End sub
```

Запускаем макрос. Видим, что Цикл стал бесконечным, программа не будет завершена. Для выхода из этой ситуации есть два пути: **Esc** или **Break.**
Иногда логические ошибки трудно сразу заметить, потому что Excel о них не сигнализирует.

Но есть такие логические ошибки, о которых Excel сигнализирует.

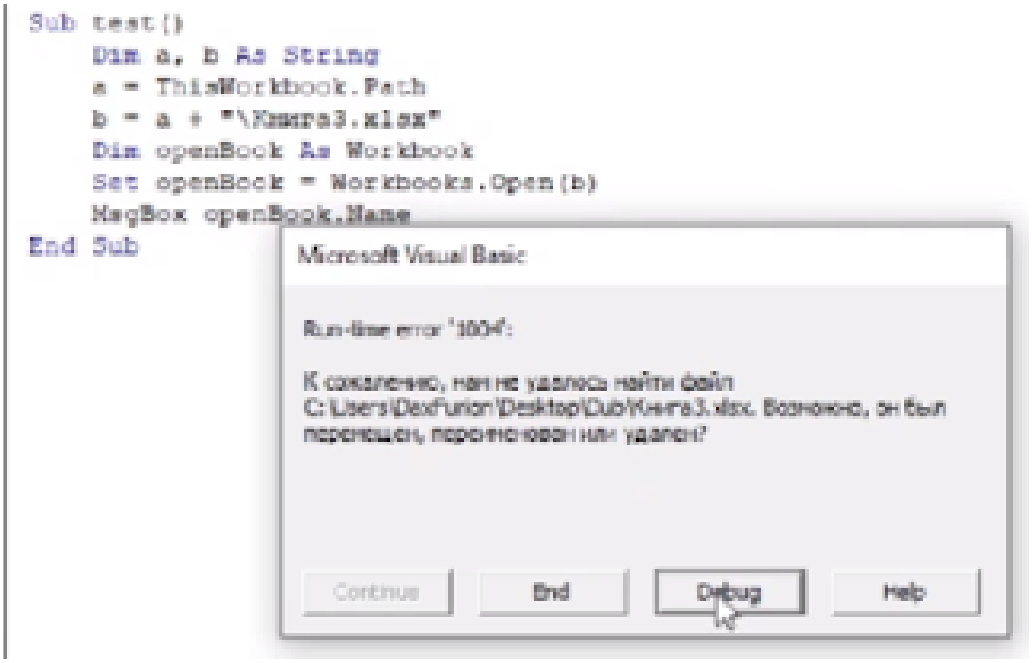
Введем строковую переменную Dim a As String. И возьмем какую-нибудь переменную «b», которая будет являться объединением переменной «a», плюс книга, которая будет находиться на нашем пути – Книга 2.xlsx.

Откроем наш макрос:

```
Sub test()
    Dim a, b As String
    a = ThisWorkbook.Path
    b = a + "\Книга2.xlsx"
    Dim openBook As Workbook
    Set openBook = Workbooks.Open(b)
    MsgBox openBook.Name
End Sub
```

А теперь давайте введем название книги, которой не в нашей папке, например – книга 3.

Мы получили путь и видим, что книгу 3 не удалось найти:



Excel предлагает два варианта – либо исправить текст, либо исправить ошибку и запустить макрос повторно.

3. ON ERROR

ON ERROR бывает 4-х типов:

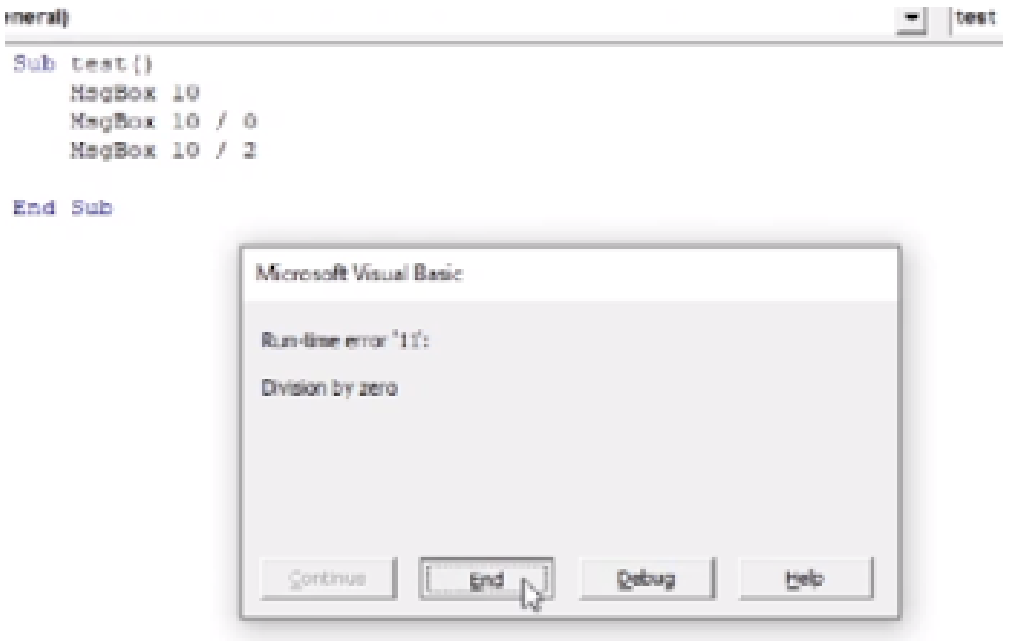
- On Error Goto 0;
- On Error Resume Next;
- On Error Goto [Label];
- On Error Goto – 1

On Error Goto 0 – дефолтное состояние

Попробуем в MagBox вывести некоторые значения

```
Sub test ()
    MagBox  10
    MagBox  10 / 0 – очевидно, что ошибка должна быть на этой строке
    MagBox  10 / 2
End sub
```

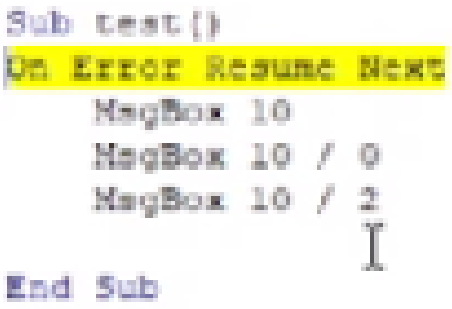
Эту ошибку мы и получим после того, как начнем пошагово производить наш макрос. Excel выдает нам надпись:



Excel, указав нам на ошибку, использовал дефолтное состояние Goto 0.

Особенность этих 4-х состояний заключается в том, что мы можем переходить в них из одного состояния в другое.

Иногда важно не допустить синтаксических ошибок, подобных делению на 0. В некоторых случаях мы можем допустить эту ошибку, так как это вычисление будет не столь значимым. Как пропустить выполнение ошибки.
Для этого достаточно в формуле перед текстом написать On Error Resume next.



Теперь пошагово выполняем макрос.
Мы получили ответ на первое значение. Второе мы пропустили, т.к. была ошибка.

По сути структура On Error Resume next пропускает ошибки по всем строкам действия

Свойство структуры **On Error Resume next** пропускать ошибки не всегда оказывается рациональным. Есть некоторые ошибки, которые пропускать нельзя. Поэтому нам надо вернуться к стандартному дефолтному состоянию **On Error Goto 0**, чтобы мы получили эту ошибку.
Это можно сделать, дописав в формуле эту строчку:

```
Sub test()
On Error Resume Next
    MsgBox 10
    MsgBox 10 / 0
    MsgBox 10 / 2
On Error GoTo 0
    MsgBox 10 / 0
End Sub
```

Таким образом, когда нам нужно пропустить какие-то математические действия, которые нам не столь важны, мы можем использовать Resume next. Но при этом мы можем вернуться назад к дефолтному состоянию, чтобы выборочно замечать и исправлять ошибки.

On Error Goto [Label] – следующий тип структуры ON ERROR.

Нам достаточно прописать строчку в формулу:

```
Sub test ()
On Error GoTo oshibka
        MagBox  10
        MagBox  10 / 0
        MagBox  10 / 2
Oshibka:
        MagBox " Присутствует ошибка"
End sub
```

Запускаем макрос. Программа отлавливает допущенные математические ошибки.

Если ошибок нет, то выполнение макроса будет производиться в стандартном режиме, но при этом программа все равно будет выдавать надпись «наличие ошибки».

С этим бороться можно следующим способом – переписываем конструкцию формул – добавим строчку **Exit Sub**, **которая заканчивает выполнение процедуры в принципе:**

```
Sub test ()
On Error GoTo oshibka
        MagBox  10
```

```
    ‘ MagBox  10 / 0
    MagBox  10 / 2
    Exit Sub
Oshibka:
    MagBox “ Присутствует ошибка”
End sub
```

On Error Goto – 1 – эта процедура используется для того, чтобы воспользоваться функцией On Error Goto [Label] повторно.

При этом каждый раз, когда мы используем функцию **On Error Goto [Label]**, нам заранее нужно будет использовать функцию **On Error Goto – 1.**