INSTITUTE OF ACTUARIAL & QUANTITATIVE STUDIES

Class: BSc

Subject : Application of IT- Basics and Advance Excel

Chapter: Unit 2 Chapter 3

Chapter Name: Error Trapping

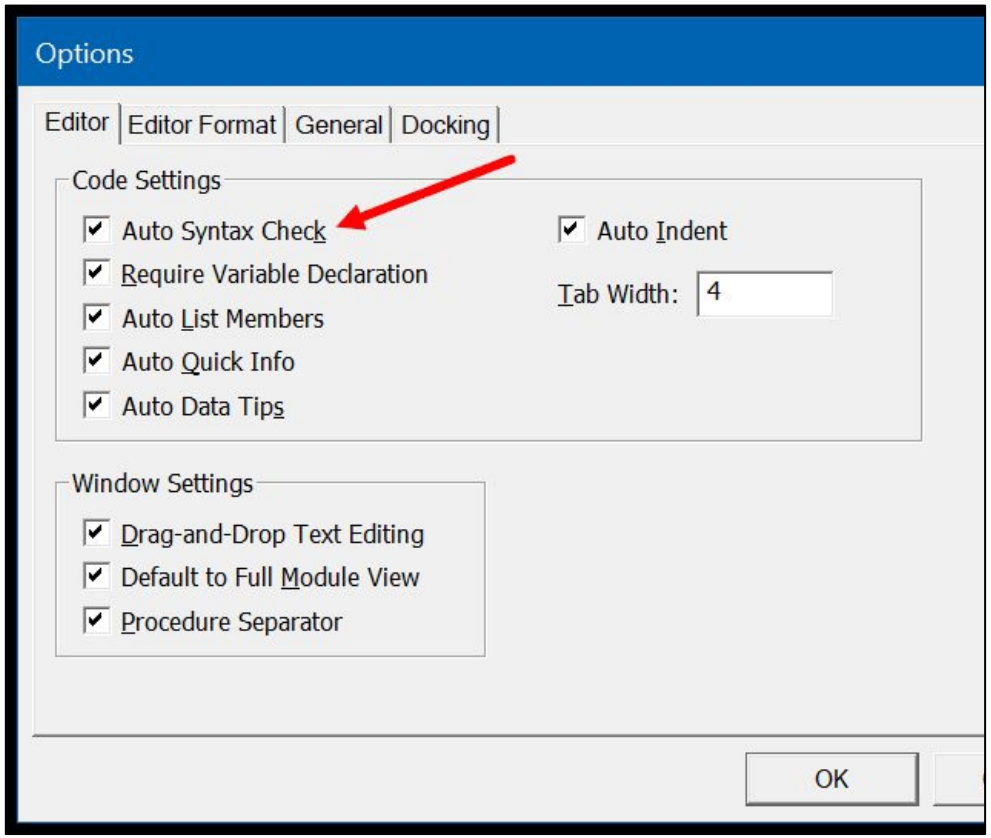
Error Handling

- *No matter how thorough you are when writing code, errors can and will happen.*
- *There are steps that developers can take to help reduce unwanted errors and this is considered just as important as the actual process of the procedure.*
- *Before understanding and applying error-handling routines, planning to avoid errors should be undertaken.*
  - *Design the procedure's process electronically or on paper – flow chart and paper test.*
  - *Using the Option Explicit statement – declaring your variables officially.*
  - *Creating smaller portions of code – snippets to be called and re-used*
  - *Syntax checking – user defined commands and functions.*
  - *Comments – remarking your code at various points.*
  - *Testing application – functional and usability.*

Types of Errors

1. Syntax Errors:

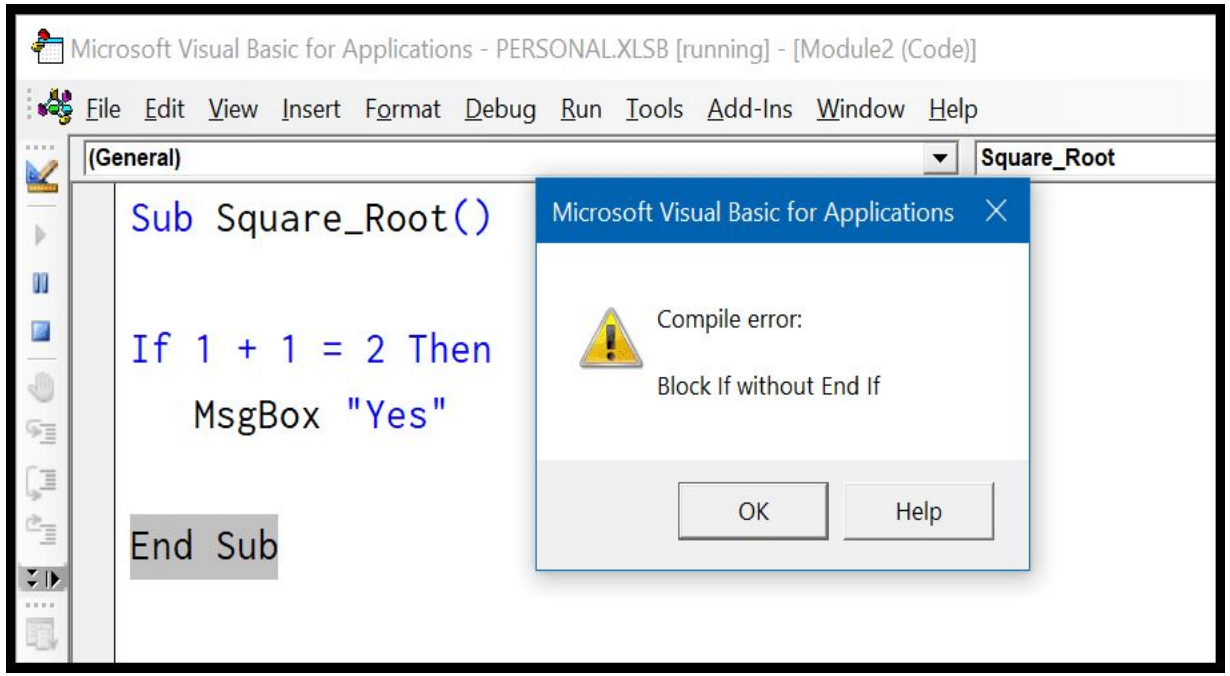
- While writing VBA code you need to follow a particular Syntax and when you skip it or don't write it in the way it should be you can face SYNTAX error (also called Language error). It's like typos that you do while writing your codes.
- Well, VBA helps you by pointing out these errors by showing an error message. You just need to make sure you have "Auto Syntax Check" activated in your VB editor.
- Go to the Tool ➤ Options and make sure to tick the "Auto Syntax Check". With this, whenever you make a SYNTAX error, VBA will show an error message.



Types of Errors

2. Compile Errors:

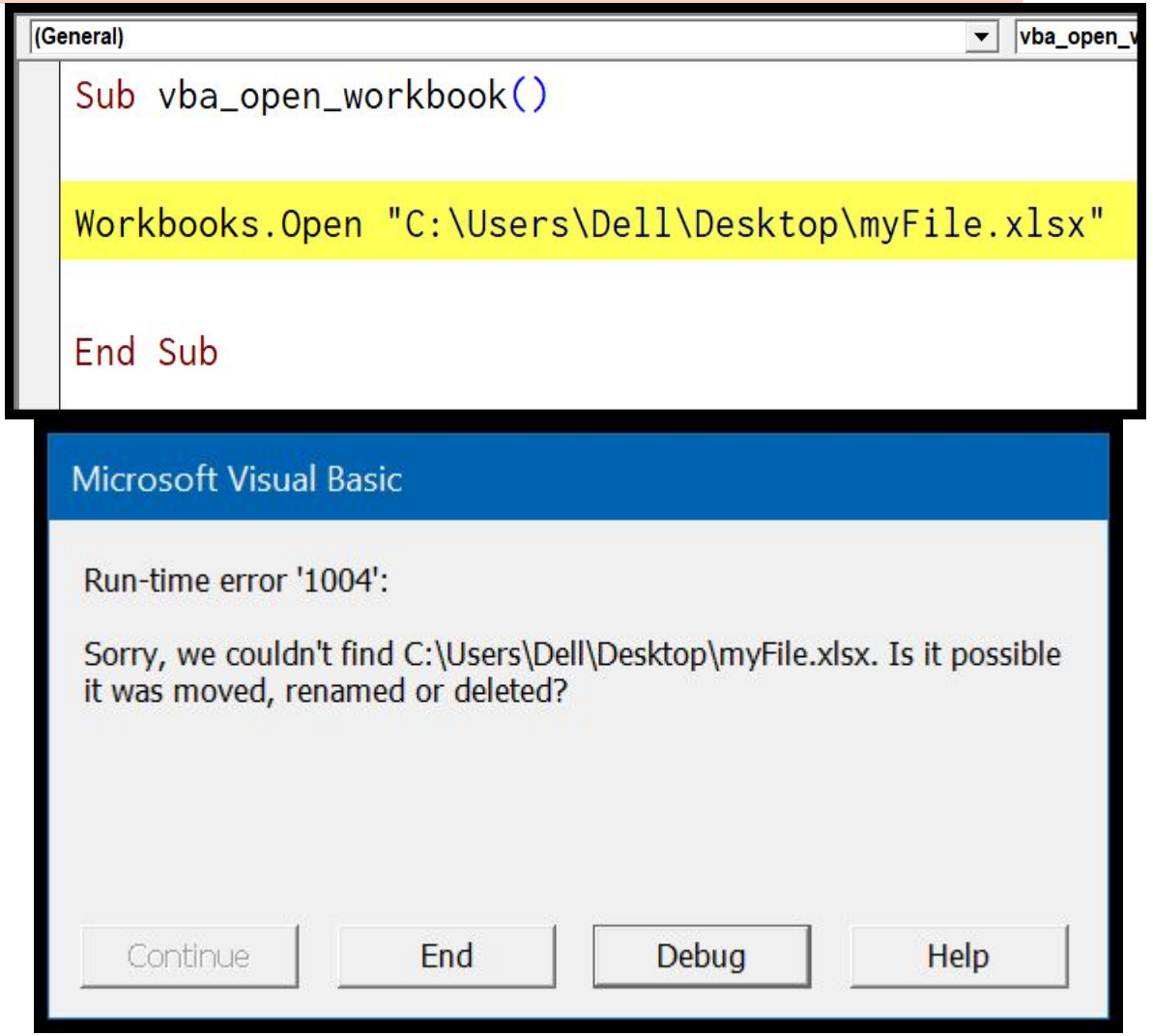
- It comes when you write code to perform an activity, but that activity is not valid or can't be performed by VBA.
- some examples of compile errors:
  - Using For without Next (For Next).
  - Select without End Select (Select Case).
  - Not Declaring a Variable when you have "Option Explicit" enabled.
  - Calling a Sub/Function that does not exist.



Types of Errors

3. Runtime Errors

- *A runtime error occurs at the time of executing the code.*
- *When a runtime error occurs while running code, it stops the code and shows you the error dialog box and that error box talks about the nature of the error you have.*
- *Let's say you have written a code that opens a workbook from the location which you have specified but now that workbook is relocated or deleted by someone.*
- *So, when you run the code, VBA will show you a runtime error as it can't find that file on that location.*



Types of Errors

4. Logical Error

- *It's not an error but a mistake while writing code. These types of errors sometimes can give you nuts while finding them and correcting them.*
- *Let's say you write code and while declaring a variable you used the wrong data type, or you have used the wrong calculation steps. In this case, your code will work fine, and you won't find this error easily.*
- *The best way to deal with this kind of problem is to run each line of code one by one. To do this you can use F8.*

Error Handling

1. On Error Resume Next:

- *On Error Resume Next tells VBA to skip any lines of code containing errors and proceed to the next line.*
- *A great time to use On Error Resume Next is when working with objects that may or may not exist.*
- *For example, you want to write some code that will delete a shape, but if you run the code when the shape is already deleted, VBA will throw an error. Instead you can use On Error Resume Next to tell VBA to delete the shape if it exists.*

```
On Error Resume Next
    ActiveSheet.Shapes("Start_Button").Delete
On Error GoTo 0
```

Note: *On Error Resume Next does not fix an error, or otherwise resolve it. It simply tells VBA to proceed as if the line of code containing the error did not exist. Improper use of On Error Resume Next can result in unintended consequences.*

Error Handling

2. On Error GoTo 0:

- *On Error GoTo 0 is VBA's default setting. You can restore this default setting by adding the following line of code:*

```
On Error GoTo 0
```

- *When an error occurs with On Error GoTo 0, VBA will stop executing code and display its standard error message box.*
- *Often you will add an On Error GoTo 0 after adding On Error Resume Next error handling. Notice in the revious slide we added On Error GoTo 0. This was done to reset error handling*

Error Handling

3. On Error GoTo LabelName:

- *On Error GoTo LabelName branches to the portion of the code with the label LabelName('LabelName' must be a text string and not a value).*
- *These commands are usually placed at the beginning of the procedure and when the error occurs,*
- *the macro will branch to another part of the procedure and continue executing code or end, depending on the instruction given.*
- *myHandler is a user defined label (must not use known keywords) which listens for any errors that may occur. When an error is detected, the procedure jumps to a bookmark of the same label with a colon (:) (myHandler:) and executes from that point forward.*

```
'Simple Error handler with Err Object
Sub ErrorTestOne()
    On Error GoTo myHandler

    Dim intDay As Integer
    intDay = "Monday"
    MsgBox intDay
    Exit Sub

myHandler:
    MsgBox "Error Number: " & Err.Number & vbNewLine _
                            & "Description: " & Err.Description
End Sub
```