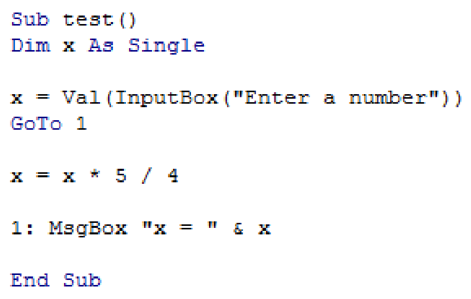
Introduction to VBA for Excel-Tutorial 5

In this tutorial, you will learn how to structure the program (code) or add some complex logic. The logic will include performing a certain sequence of operations based on the status of the process or the value of a variable in comparison to specific criteria. These are called **Decisions**. One clear example is the way we follow traffic laws while driving, i.e. you see a red light, and you stop! Notice the light status is the criteria and you breaking to stop is the decision. You perform such decisions every time you approach each traffic light, this is called **repetition** or **looping**.

Decisions:

1- ***GoTo statement***: it allows you to override the sequence in which the program is executed. However, it is easy to implement, it complicates the program very quickly, which in turn make programs extremely difficult to understand. Use it conservatively!



```
Sub test()
Dim x As Single

x = Val(InputBox("Enter a number"))
GoTo 1

x = x * 5 / 4

1: MsgBox "x = " & x

End Sub
```

When executing the above code, the user will input a value for the variable ***x***, then the program will skip over the "***x = x * 5/4***" statement because *GoTo* points the program to line number 1. In this case, the output will be the same value the user input.

2- ***If Statement***: is used to make a decision based on pre-established conditions, i.e. the traffic light is red => stop. In fact, you can use if statements to make multiple decisions based on multiple conditions. The general form is

```
If condition1 Then
   true_statement1
Elseif condition2 Then
   true_statement2
ElseIf condition3 Then
  true_statement3
  …..
  …..
  etc
Else
 true_statement4
End If
```

where, condition1, condition2, etc. are the statements that are either true or false and the true_statement1, true_statement2 , etc. are one or more statements that are executed if the condition is true.

The number of conditions and thus the number of true-statements are dependent on the process logic. To better illustrate the usage of an If statement, let's revisit our quadratic equations roots example. Previously, we were able to determine the real roots only, now let's finish the problem.

```
Option Explicit
Type complex
real As Single
imag As Single
End Type
Sub quad()
'variables declaration statements
Dim a As Single, b As Single, c As Single, d As Single, r1 As Single, r2 As Single
Dim rc As complex

'input
a = Val(InputBox("input a -Coefficient of x^2"))
b = Val(InputBox("input b -Coefficient of x"))
c = Val(InputBox("input c -Constnt"))

'echo inputs to worksheet
Worksheets(1).Activate

Cells(1, 1).Value = "a"
Cells(1, 2).Value = a
Range("A2").Value = "b"
Range("B2").Value = b
Range("A3").Value = "c"
Range("B3").Value = c

'calculate the discriminant
d = b ^ 2 - 4 * a * c

'logic
If d >= 0 Then
  r1 = (-b + Sqr(d)) / (2 * a)
  r2 = (-b - Sqr(d)) / (2 * a)
  MsgBox "Real Roots" & Chr(13) & "r1 = " & r1 & Chr(13) & "r1 = " & r1, , "Quadractic Roots"
Else
 rc.real = -b / (2 * a)
 rc.imag = Sqr(-d) / (2 * a)
 MsgBox "Imaginary Roots" & Chr(13) & "r = " & rc.real & " " & rc.imag & "i"
End If
End Sub
```

Note in this example, we used only if-Then-Else-End If statement because we had two conditions (i.e. d≥0 or d<0).

In the above example, the condition is set by comparison of the variable to a value (i.e. d≥0 or d<0) using Boolean (comparison) operators. A list is given in the table below:

Operator	Description	Example
=	Equal	x = y
<>	Not equal	x <> y
>	Greater than	x > y
>=	Greater than or equal	x > y
<	Less than	x < y
<=	Less than or equal	x <= y
Or	cond1 Or cond2	d=0 Or d>0
And	cond1 And cond2	d=0 And d>0

You can write the whole if statement in one line and in this case, you don't need to add "End If" such that

```
If d=0 Then r=4
```

Or

```
If d=3 Then r=4 Else r=5
```

3- ***Select Case Structure***: used when deciding on the basis of the value of a variable. In other words, it's used in situations where we execute particular actions on the basis of the value of a single variable. It is easy to use but may be a little intimidating in the beginning. The syntax is

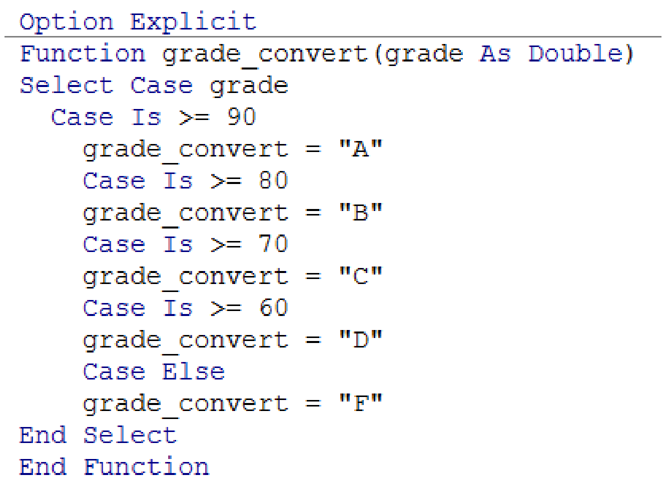
```
Select Case test_expression
    [Case expression_list1
        statements]]
    [Case Else
        [else_statements]]
End Select
```

OK! Don't be scared of the syntax, the following examples will clear everything. You will see, it is as easy as the If-statements.

Example #1:

In this example, we will translate a numerical grade to a letter grade. If the entered grade value is more than or equal 90, assign "A" as the letter grade. While grades more than or equal to 80 is considered "B" and so on. This example can also be done using If/Then/ElseIf. Yes! As you expect the Select Case and If/Then statements are interchangeable.

However in some instances If/Then is superior to the Select Case. I personally prefer If/Then statements, rarely do I use the Select Case. Refer to Example #2 to compare the two statements.



```
Option Explicit
Function grade_convert(grade As Double)
Select Case grade
  Case Is >= 90
    grade_convert = "A"
    Case Is >= 80
    grade_convert = "B"
    Case Is >= 70
    grade_convert = "C"
    Case Is >= 60
    grade_convert = "D"
    Case Else
    grade_convert = "F"
End Select
End Function
```

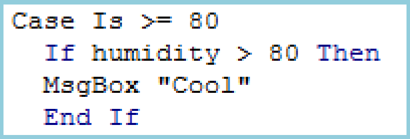
Example #2:

In this example, you are required to design control logic based on the temperature using the following rules:

- If the temperature > 85°F, display a “Cool” message so the operator turns on the air-conditioning unit.
- If the temperature > 80°F and humidity >80%, display a “Cool” message so the operator turns on the air-conditioning.
- If the temperature < 65°F, display a message “Heat” so the operator turns on the heat.

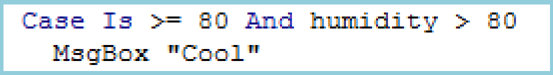
The code is shown below where the user has the choice to either use Select Case or If/Then. I wrote the code two different ways (1) the choice of using Select Case or If/Then is made using If/Then or (2) the choice is made using Select Case.

Note: in the case of using Select Case, condition #2 can not be satisfied only by using Case structure we had to use If/Then also because humidity is not the varaible in question.



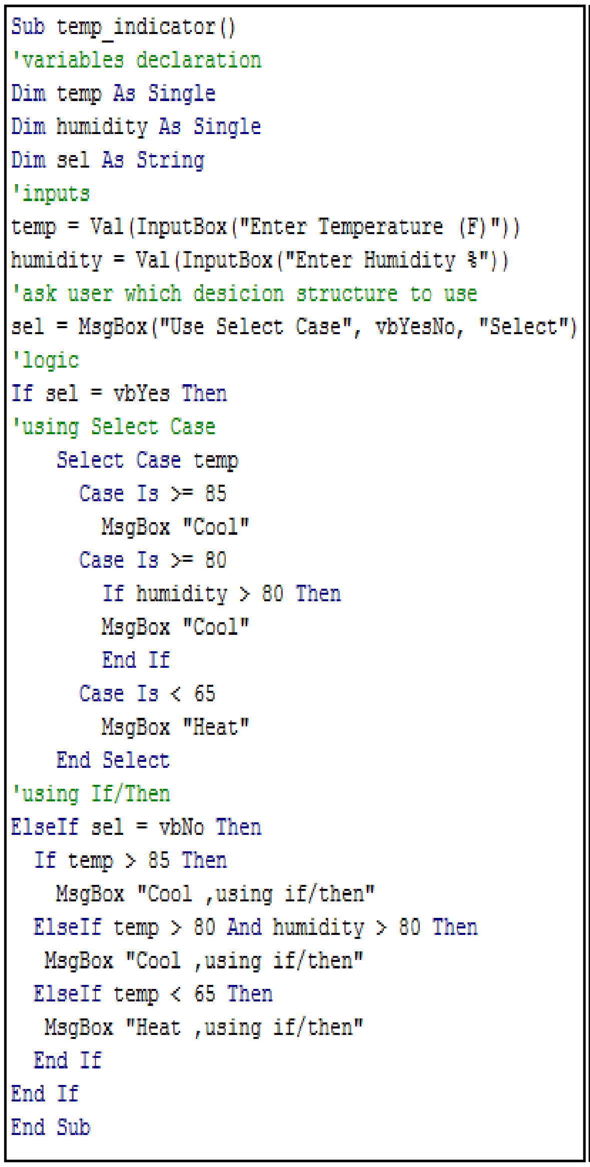
```
Case Is >= 80
  If humidity > 80 Then
  MsgBox "Cool"
  End If
```

However, I could wrote it such as

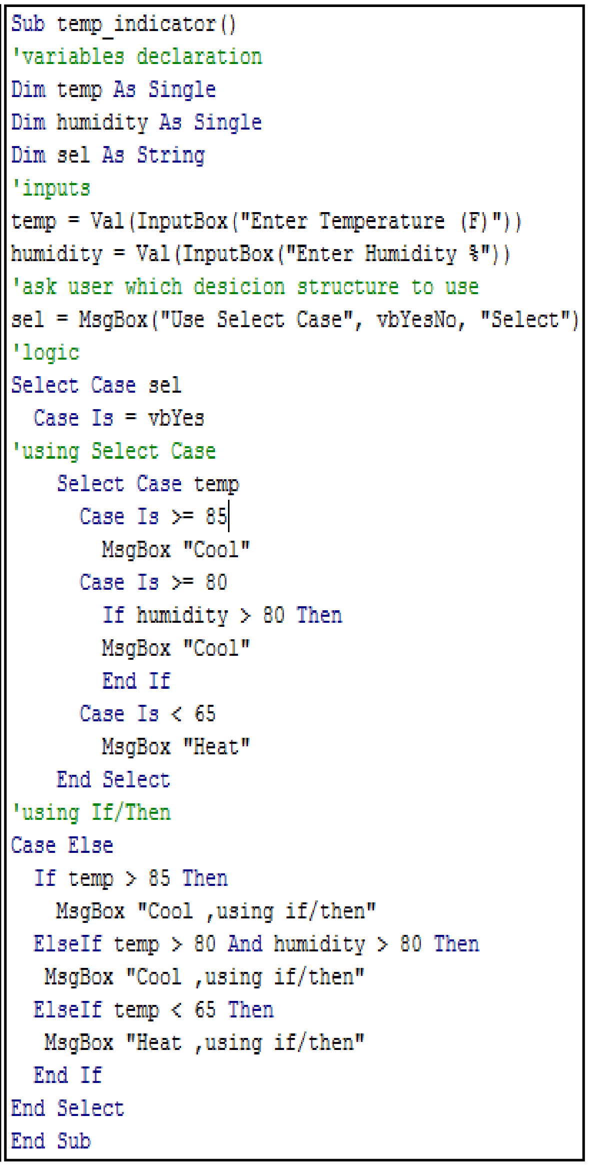


```
Case Is >= 80 And humidity > 80
  MsgBox "Cool"
```

Both ways would work fine. In this example as you see, using If/Then or Select Case would yield the same results. Which would you use? It is entirly up to you unless the question otherwise indicate to use sepcific structure.



```
Sub temp_indicator()
'variables declaration
Dim temp As Single
Dim humidity As Single
Dim sel As String
'inputs
temp = Val(InputBox("Enter Temperature (F)"))
humidity = Val(InputBox("Enter Humidity %"))
'ask user which desicion structure to use
sel = MsgBox("Use Select Case", vbYesNo, "Select")
'logic
If sel = vbYes Then
'using Select Case
    Select Case temp
      Case Is >= 85
        MsgBox "Cool"
      Case Is >= 80
        If humidity > 80 Then
        MsgBox "Cool"
        End If
      Case Is < 65
        MsgBox "Heat"
    End Select
'using If/Then
ElseIf sel = vbNo Then
  If temp > 85 Then
    MsgBox "Cool ,using if/then"
  ElseIf temp > 80 And humidity > 80 Then
   MsgBox "Cool ,using if/then"
  ElseIf temp < 65 Then
   MsgBox "Heat ,using if/then"
  End If
End If
End Sub
```



```
Sub temp_indicator()
'variables declaration
Dim temp As Single
Dim humidity As Single
Dim sel As String
'inputs
temp = Val(InputBox("Enter Temperature (F)"))
humidity = Val(InputBox("Enter Humidity %"))
'ask user which desicion structure to use
sel = MsgBox("Use Select Case", vbYesNo, "Select")
'logic
Select Case sel
  Case Is = vbYes
'using Select Case
    Select Case temp
      Case Is >= 85
        MsgBox "Cool"
      Case Is >= 80
        If humidity > 80 Then
        MsgBox "Cool"
        End If
      Case Is < 65
        MsgBox "Heat"
    End Select
'using If/Then
Case Else
  If temp > 85 Then
    MsgBox "Cool ,using if/then"
  ElseIf temp > 80 And humidity > 80 Then
   MsgBox "Cool ,using if/then"
  ElseIf temp < 65 Then
   MsgBox "Heat ,using if/then"
  End If
End Select
End Sub
```