COMP1022Q
Introduction to Computing with Excel VBA

Custom Functions

Gibson Lam and David Rossiter

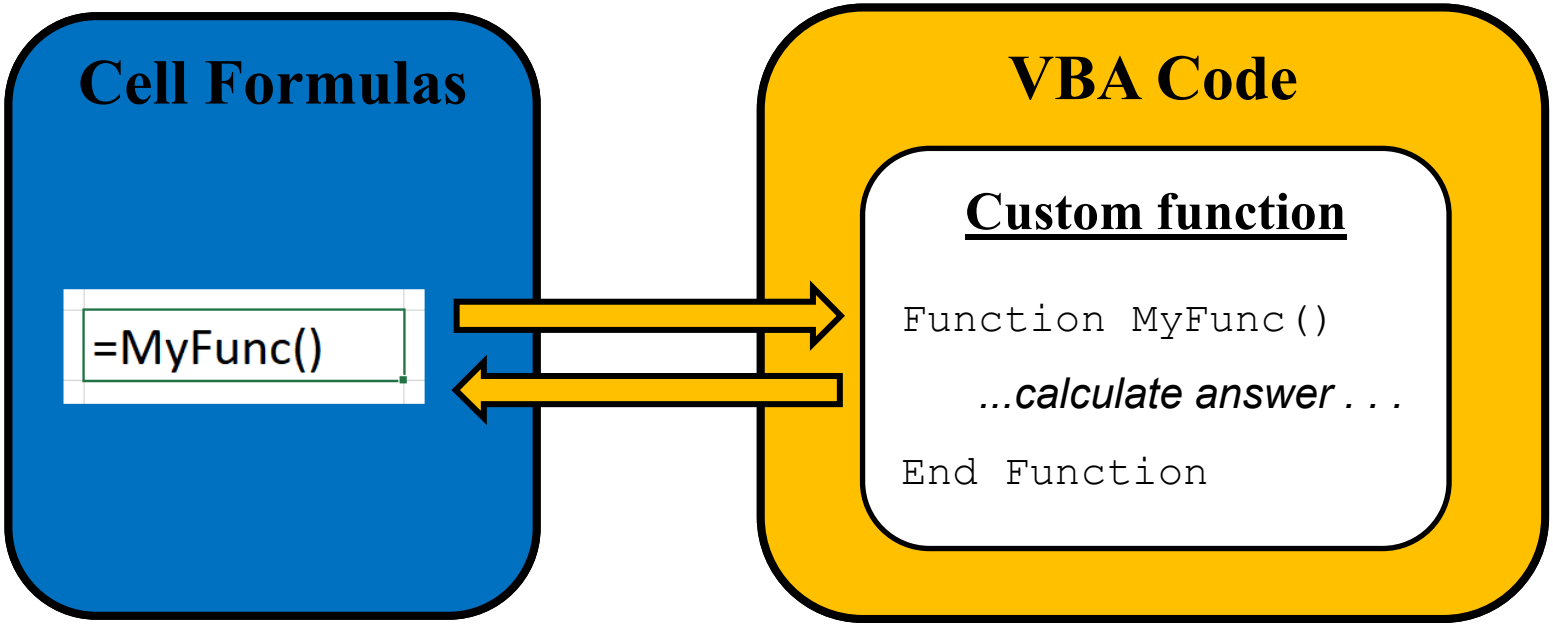
Outcomes

- After completing this presentation, you are expected to be able to:

  1. Create and use custom VBA functions in cell formulas

This Presentation

- In this presentation we will look at how to let cell formulas use a particular kind of VBA function, called a *Custom Function*

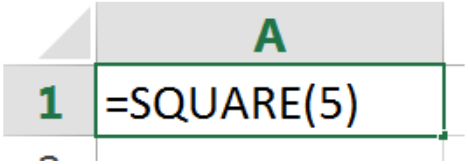


Custom Functions

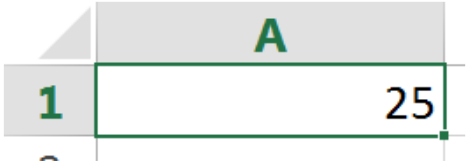
- If you create a **function** inside a VBA module, you will then be able to use the function inside a cell formula, for example:

```
Function SQUARE(Number)
    SQUARE = Number * Number
End Function
```

Create a function inside a VBA module



Use the VBA function in a cell formula



The formula is evaluated and displayed as usual

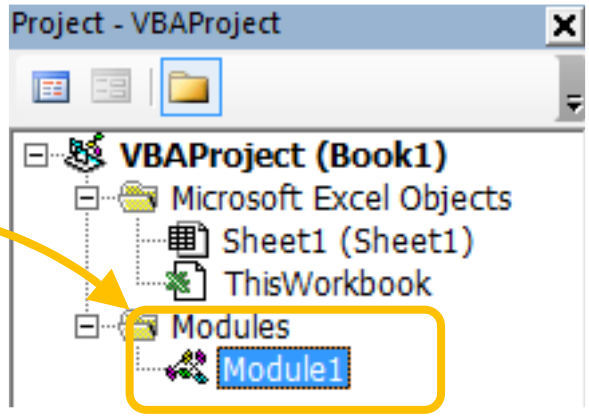
- This kind of function is called a *Custom Function*
- They are also called *User Defined Functions*

Why are Custom Functions Useful?

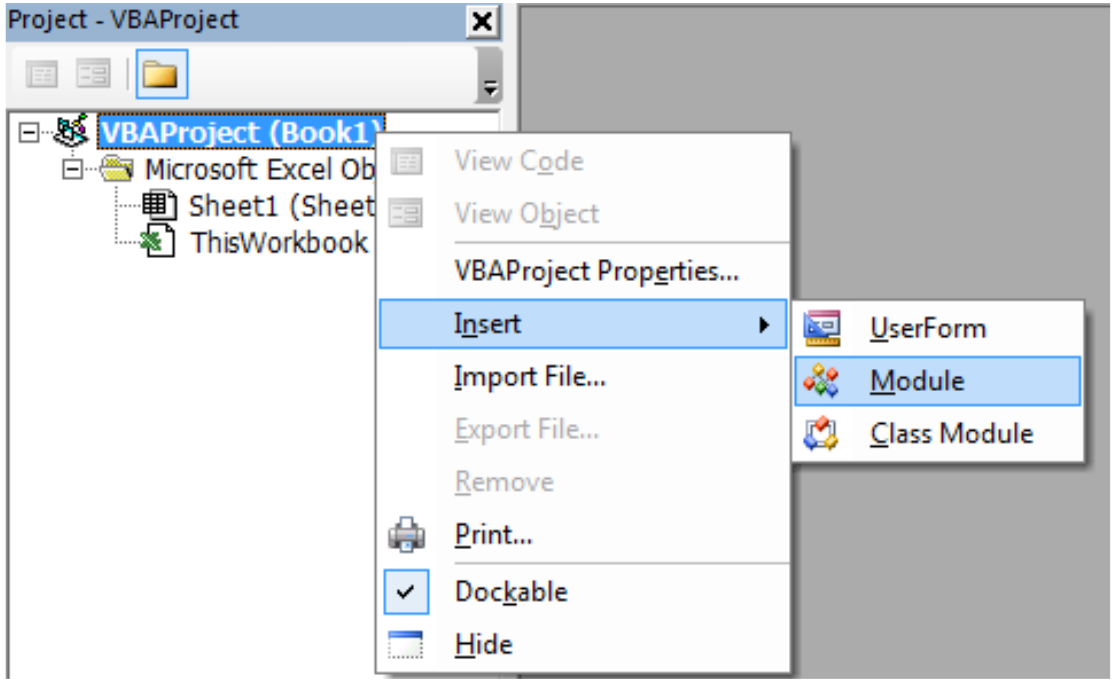
- Sometimes you may not be able to find a suitable cell function for your cell formula, even though there are hundreds of cell functions available
- If you can create your own cell function using VBA you will have a lot more flexibility when writing your cell formulas

Adding a Custom Function 1/2

- A custom function must be created in a VBA module in your Excel file (the place where macros are stored)



- If you don't have one in your VBA project you could quickly make a macro which doesn't do anything, or add a new module by right-clicking on the *VBAProject* and then selecting *Insert > Module*



Adding a Custom Function 2/2

- When you have a VBA module you can then add your custom function inside it
- For example, you can make an `ABBREVIATION` function with one input parameter, like this:

```
Function ABBREVIATION(InputText)
```

...VBA Code of the function is shown on the next slide...

```
End Function
```

- Note that you don't have to use all capital letters for the function name; we do this here just to help emphasise that we are making a custom function

The ABBREVIATION Function

- The `ABBREVIATION` function works like this:
  - Given a piece of text, the function extracts all starting letters of words to form the abbreviation of the text
  - The function then converts the abbreviation into capital letters
- Here is an example: **B**y **t**he **w**ay ⇨ BTW
- To obtain only the starting letters, the function uses a loop to look for spaces in the text and put the letter next to a space in the abbreviation

The Code of the Function

```
Function ABBREVIATION(InputText)
    Dim Pos As Integer

    ABBREVIATION = Left(InputText, 1)

    For Pos = 2 To Len(InputText)
        If Mid(InputText, Pos - 1, 1) = " " Then
            ABBREVIATION = ABBREVIATION & _
                           Mid(InputText, Pos, 1)
        End If
    Next Pos

    ABBREVIATION = UCase(ABBREVIATION)
End Function
```

The first letter of the text is in the abbreviation, assuming it is not a space

Change the result to upper case

The loop extracts the first letters of the words, starting from the second letter

Using the Mid Function

- The example uses the `Mid` function
- The `Mid` function returns the middle part of a string given the string, the starting position and the length of the part of the string you want
- For example, `Mid("COMP1022Q", 5, 4)` gives you “1022”
- In the `ABBREVIATION` function, the `Mid` function returns a letter of `InputText` at the position `Pos-1` and the position `Pos`

Using the ABBREVIATION Function

- You can use the `ABBREVIATION` function in a cell formula like this:

```
=ABBREVIATION("Laugh out loud")
```

- The input value of the above example is "Laugh out loud" and therefore the abbreviation of the text is "LOL":

```
LOL
```

Using the Function with a Cell Reference

- You can use a custom function just like any cell function so that you can give a cell reference to the function
- Relative reference also works if you copy and paste the formula containing the function to other cells

Input Text	Abbreviation
In my humble opinion	=ABBREVIATION(B5)
Laugh out loud	
Oh my god!	
By the way	

Input Text	Abbreviation
In my humble opinion	IMHO
Laugh out loud	
Oh my god!	
By the way	

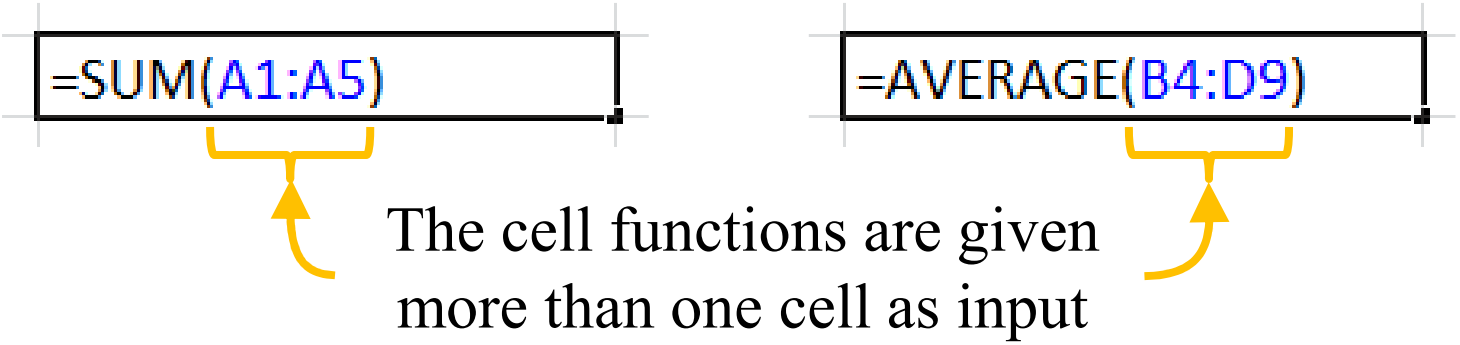
Input Text	Abbreviation
In my humble opinion	IMHO
Laugh out loud	
Oh my god!	
By the way	

Copy and paste the cell to the rest of the column

Input Text	Abbreviation
In my humble opinion	IMHO
Laugh out loud	LOL
Oh my god!	OMG
By the way	BTW

Giving a Range as Input

- The `ABBREVIATION` function assumes the input is a single value, i.e. a string or a cell
- In some situations, you need to process more than one cell
- `SUM` and `AVERAGE` are examples of cell functions which process more than one cell



An Example Using Range as Input

- In this example, an Excel worksheet has a class of students and their end-of-semester total
- Their course grades are then automatically assigned based on a grade distribution table

	A	B
4	**Student**	**End-of-semester Total**
5	Barbara	82.51
6	Charles	76.16
7	David	69.66
8	Dorothy	82.16

⋮

There are 20 students in total

	E	F
4	**Grade**	**Percentage**
5	A	20
6	B	30
7	C	35
8	D	10
9	F	5

- To assign the grades a custom function is created, next slide:

The Custom Function

The first parameter will be in a variable called *Total* which is a *Single* type

The second parameter will be in a variable called *Class* which is a *Range* type (for cells)

```
Function ASSIGNGRADE( Total As Single , _
                      Class As Range , _
                      Table As Range )
    . . .
End Function
```

The third parameter will be in a variable called *Table* which is a *Range* type (for cells)

- The function takes the total of a student, the total of the entire class and the grade distribution as inputs
- It returns the grade of the student

An Example Use of the Function

- Let's use the function for the first student of the class:

	A	B	C
4	**Student**	**End-of-semester Total**	**Grade**
5	Barbara	82.51	=ASSIGNGRADE(B5, $B$5:$B$24, $E$5:$F$9)

⋮

	E	F
4	**Grade**	**Percentage**
5	A	20
6	B	30
7	C	35
8	D	10
9	F	5

	A	B	C
4	**Student**	**End-of-semester Total**	**Grade**
5	Barbara	82.51	B

Using a Range Input

- Both `Class` and `Table` inputs are ranges of cells
- We can use the properties of a range object to know the location of the cells, for example, for the grade distribution table:

first row `Table.Row`

first column `Table.Column`

how many rows

`Table.Rows.Count`

how many columns

`Table.Columns.Count`

	E	F
4	Grade	Percentage
5	A	20
6	B	30
7	C	35
8	D	10
9	F	5

Using the Function Over the Class

- We can then assign the grades of the entire class using the function on all students:

	A	B	C
4	**Student**	**End-of-semester Total**	**Grade**
5	Barbara	82.51	=ASSIGNGRADE(B5,$B$5:$B$24,$E$5:$F$9)
6	Charles	76.16	=ASSIGNGRADE(B6,$B$5:$B$24,$E$5:$F$9)
7	David	69.66	=ASSIGNGRADE(B7,$B$5:$B$24,$E$5:$F$9)
8	Dorothy	82.16	=ASSIGNGRADE(B8,$B$5:$B$24,$E$5:$F$9)
9	Elizabeth	81.4	=ASSIGNGRADE(B9,$B$5:$B$24,$E$5:$F$9)
10	James	80.87	=ASSIGNGRADE(B10,$B$5:$B$24,$E$5:$F$9)
11	Jennifer	67.02	=ASSIGNGRADE(B11,$B$5:$B$24,$E$5:$F$9)
12	John	86.53	=ASSIGNGRADE(B12,$B$5:$B$24,$E$5:$F$9)
13	Joseph	84.19	=ASSIGNGRADE(B13,$B$5:$B$24,$E$5:$F$9)
14	Linda	66.56	=ASSIGNGRADE(B14,$B$5:$B$24,$E$5:$F$9)
15	Margaret	96.39	=ASSIGNGRADE(B15,$B$5:$B$24,$E$5:$F$9)
16	Maria	72.33	=ASSIGNGRADE(B16,$B$5:$B$24,$E$5:$F$9)
17	Mary	76.81	=ASSIGNGRADE(B17,$B$5:$B$24,$E$5:$F$9)
18	Michael	100	=ASSIGNGRADE(B18,$B$5:$B$24,$E$5:$F$9)
19	Patricia	91.8	=ASSIGNGRADE(B19,$B$5:$B$24,$E$5:$F$9)
20	Richard	79.47	=ASSIGNGRADE(B20,$B$5:$B$24,$E$5:$F$9)
21	Robert	94.28	=ASSIGNGRADE(B21,$B$5:$B$24,$E$5:$F$9)
22	Susan	87.81	=ASSIGNGRADE(B22,$B$5:$B$24,$E$5:$F$9)
23	Thomas	82.08	=ASSIGNGRADE(B23,$B$5:$B$24,$E$5:$F$9)
24	William	67	=ASSIGNGRADE(B24,$B$5:$B$24,$E$5:$F$9)

C
Grade
B
C
C
B
C
C
D
B
B
F
A
C
C
A
A
C
A
B
B
D

The Procedure

- The function uses two steps to find the grade of a student:
  1. Finding the rank of the student within the class and then expressing the rank as a percentage
  2. Based on the above percentage, finding where the student lies within the grading distribution table

Preparing the Variables

- The function starts by creating the necessary variables used inside it:

```
Option Explicit

Function ASSIGNGRADE( Total As Single, _
                      Class As Range, _
                      Table As Range )

    Dim Row As Integer
    Dim Rank As Integer, Percentage As Single
    Dim GradeCutOff As Single
```

Require the code to create all variables before using them

Continued on the next slide…

Finding the Rank As a Percentage

Continued from the previous slide…

First find the rank of the student (`Rank`) using a loop

```
Rank = 1
For Row = Class.Row To _
          Class.Row + Class.Rows.Count - 1
    If Cells(Row, Class.Column).Value > Total Then
        Rank = Rank + 1
    End If
Next Row

Percentage = _
    Rank / Class.Rows.Count * 100
```

Change the rank number to a percentage

Number of students in the class

Continued on the next slide…

Finding the Grade

Continued from the previous slide…

Accumulated percentrage for the cutoff of a grade

```
    GradeCutOff = 0
    For Row = Table.Row To Table.Row + _
              Table.Rows.Count - 1
        GradeCutOff = GradeCutOff + _
            Cells(Row, Table.Column + 1).Value

        If Percentage <= GradeCutOff Then
            ASSIGNGRADE = _
                Cells(Row, Table.Column).Value
            Exit For
        End If
    Next Row
End Function
```

Loop through the grade table

Assign the grade and exit the loop

Limitations of Custom Functions

- A custom function can only be used to process data such as numbers and text and then return a value
- It cannot be used to perform some ‘actions’ such as selecting a cell or changing the font size of a cell
- For example, the following code (changing the text in all selected cells to a large size) does NOT work inside a custom function:

```
Selection.Font.Size = 50
```

Change the text in all selected cells to a large size

- A normal function can do anything, it’s only a custom function which can’t do visual things