

COMP1022Q

Introduction to Computing with Excel VBA

Using Cell Functions in VBA

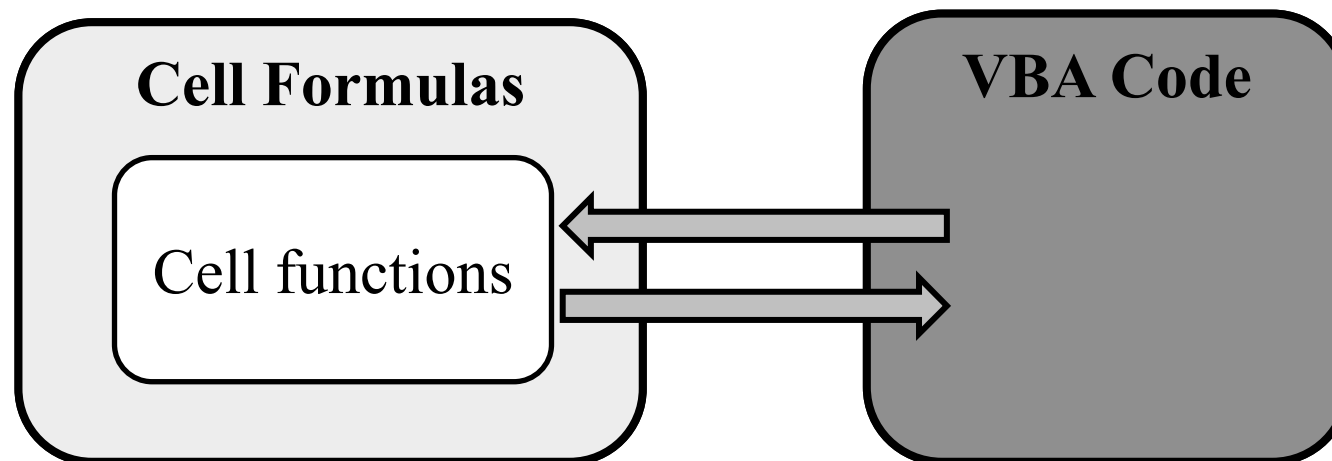
Gibson Lam and David Rossiter

Outcomes

- After completing this presentation, you are expected to be able to:
 1. Use cell functions in VBA code

Using Cell Functions in VBA Code

- So far, we've seen lots of examples of VBA code working 'by itself'
- However, VBA code can use many of the Excel cell functions:



Which Cell Functions Can You Use?

- Here are some cell functions that we have seen before which can be used in VBA:

- SUM ()
- COUNTIF ()
- AVERAGE ()
- MIN ()
- MAX ()
- FLOOR ()
- SUBSTITUTE ()
- VLOOKUP ()
- HLOOKUP ()
- ISNA ()
- IFNA ()
- AND ()
- OR ()
- and others...


Why Use Cell Functions in VBA?

- Cell functions can be very helpful
- If you already know how to do a task in a cell formula it means you don't have to work out how to do the same task in VBA
- A cell function typically (but not always) runs faster than any VBA code that you could write which does the same thing

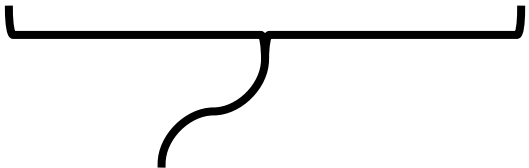
Using WorksheetFunction

- You cannot use a cell function in VBA just like what you do with a VBA function
- You need to use a cell function through *WorksheetFunction*, like this:

`WorksheetFunction.FunctionName (... input values ...)`



The name of the cell function
you want to use in the code



The input values, if there
is any, to the cell function

A Simple Example

- A cell formula to add the numbers in cells B4 to B10 can be written like this:

A diagram of an Excel cell formula bar. It is a rectangular box with a thin border. Inside the box, the text "=SUM(B4:B10)" is written in a standard font. The box is slightly offset from the center of the slide.

- You can do the same thing in VBA using the same cell function, as shown below:

```
Total =                       
WorksheetFunction.Sum(Range("B4:B10"))
```



The result is put in a variable Total in this example

Cell Function Names in VBA


- As you know, the name of a cell function uses all capital letters in a cell, e.g. `COUNTIF ()`
- However, in VBA, the name of a cell function is capitalized only at the beginning of each word, e.g. `CountIf ()`
- When you type the name of a cell function either way inside the VBA editor, VBA will automatically convert the name to use the VBA way

Giving Cells As Input to a Cell Function

- When an input value to a cell function is a cell reference, you cannot directly write the cell reference as an input in VBA
- You need to use `Range()` or `Cells()` when you pass cells to a cell function, like the example we have shown before:

```
Total = _  
WorksheetFunction.Sum ( Range ("B4:B10") )
```

`Range ("B4:B10")` is used instead of `B4:B10`



A Bigger Example

- Let's look at a bigger example
- Sometimes people who buy the Mark 6 lottery ticket don't know what numbers to put down
- Let's help them by randomly generating the numbers
- So, we need to generate 6 numbers – but we have to make sure the same number is not used twice



Using RANDBETWEEN and COUNTIF

- We will use these two cell functions:

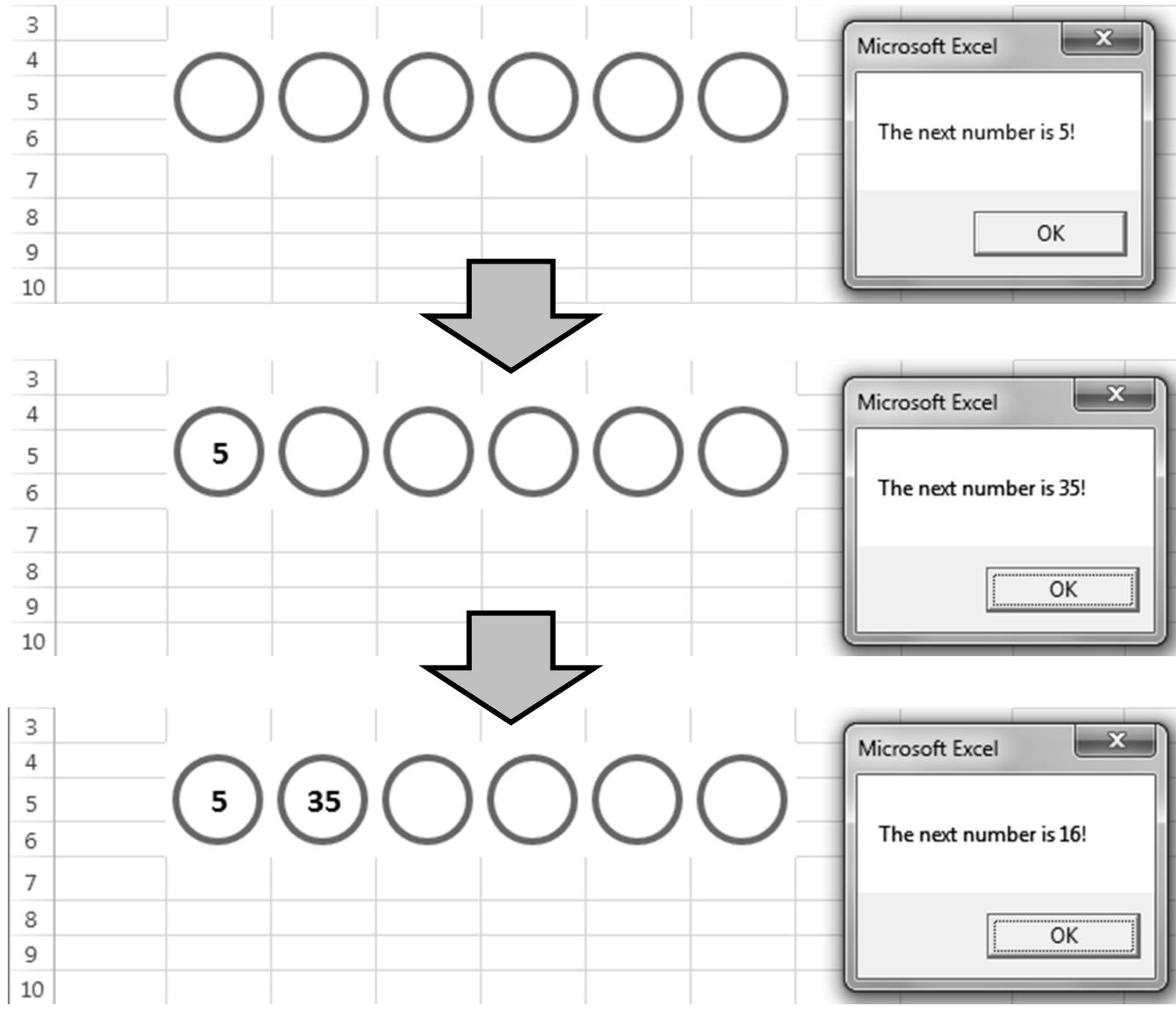
`RANDBETWEEN (a, b)`

- We use `RANDBETWEEN (1, 49)` to generate a random number between 1 and 49 inclusive

`COUNTIF (range, number)`

- Counts how many times `number` is inside `range`
- For example, `COUNTIF (B5 : G5, 3)` counts how many times the number 3 appears in those cells
- We have used `COUNTIF` earlier in the course

Running the Example

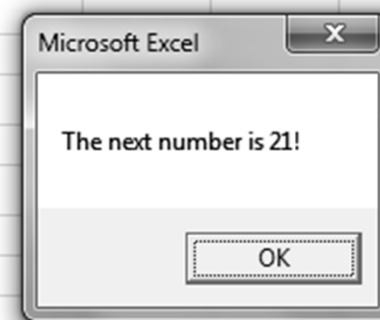




3					
4					
5	5	35	16		
6					
7					
8					
9					
10					



3					
4					
5	5	35	16	15	
6					
7					
8					
9					
10					



3					
4					
5	5	35	16	15	21
6					
7					
8					
9					
10					



3					
4					
5	5	35	16	15	21
6					6
7					

```

Private Sub Workbook_Open()
    Dim Ball As Integer, Number As Integer

    ' Clear the current numbers
    Range("B5:G5").ClearContents

    ' Generate a total of 6 numbers
    Ball = 1
    While Ball <= 6
        ' Generate a random number between 1 to 49
        Number = WorksheetFunction.RandBetween(1, 49)

        ' If the number is a unique number (CountIf returns 0),
        ' put the number in the worksheet and move to
        ' the next round
        If WorksheetFunction.CountIf(Range("B5:G5"), Number) = 0 Then
            ' Show the number in a message box
            MsgBox "The next number is " & Number & "!"

            ' Put it in a cell
            Cells(5, Ball + 1) = Number

            ' Move on to generate the next ball number
            Ball = Ball + 1
        End If
    Wend
End Sub

```

Which Cell Functions **Can't** You Use?

- You can't use all cell functions when you use `WorksheetFunction`
- For example, you cannot use the `IF ()` cell function
- One reason it can't be used is because the same thing can already be done in VBA

Available Cell Functions

- How can you know exactly which cell functions can be used inside VBA?
 1. From some reference pages on the web, e.g.:
<http://msdn.microsoft.com/en-us/library/office/ff822194.aspx>
 2. From the list which is shown when you type “WorksheetFunction.” inside the VBA editor:

WorksheetFunction.

