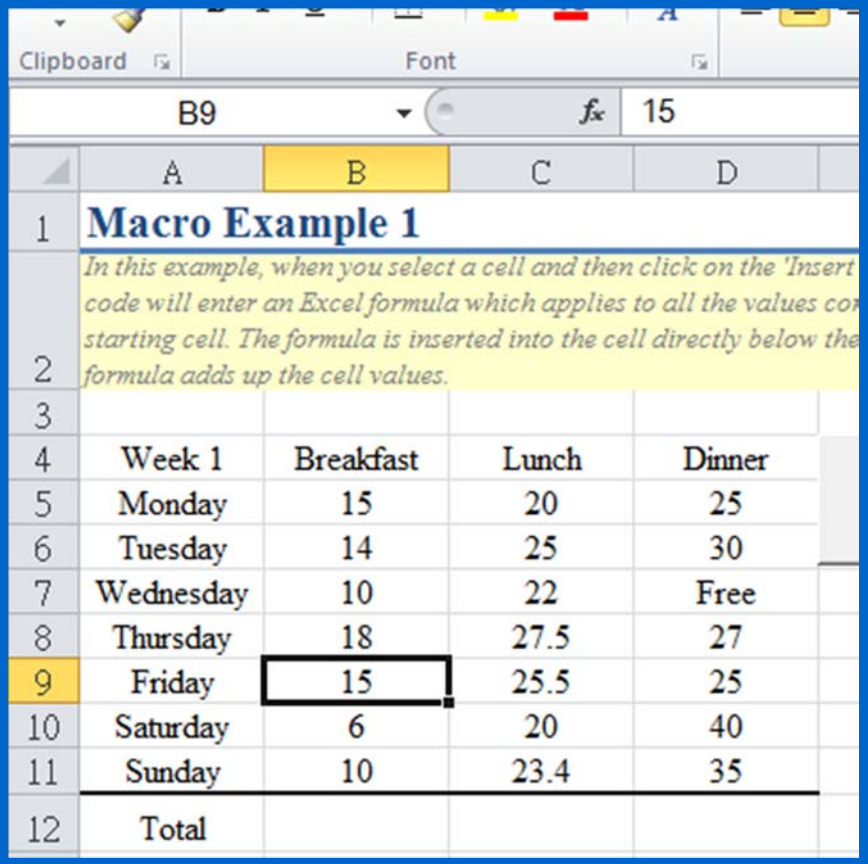
COMP1022Q
Introduction to Computing with Excel VBA

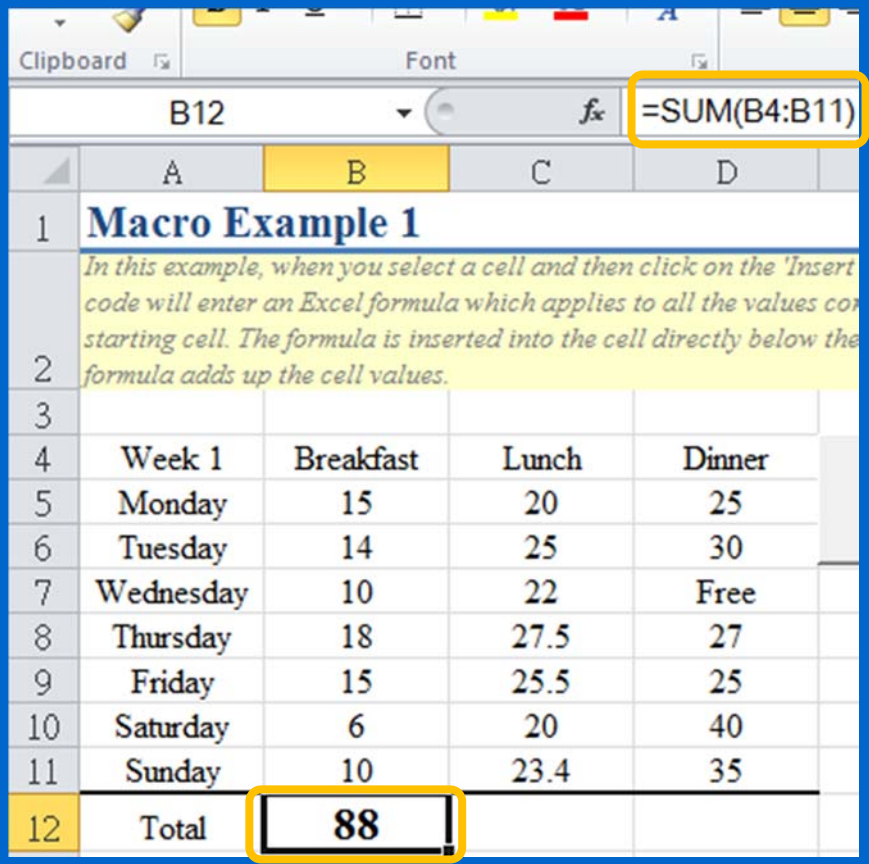
Two Example Macros

David Rossiter and Oz Lam

Two Example Macros

- In this presentation we show two macros
- The first macro inserts a formula underneath a range of continuous cells to add up all the numbers in those cells





Example Results

- The first macro works well for many different situations
- It works for both integer and decimal numbers
- It works even if there are other groups of numbers in the same column
- Non-numbers in the group are ignored

D12 | =SUM(D4:D11)

	A	B	C	D
1	**Macro Example 1**			
2	*In this example, when you select a cell and then click on the 'Insert a code will enter an Excel formula which applies to all the values cont starting cell. The formula is inserted into the cell directly below the l formula adds up the cell values.*			
3				
4	Week 1	Breakfast	Lunch	Dinner
5	Monday	15	20	25
6	Tuesday	14	25	30
7	Wednesday	10	22	Free
8	Thursday	18	27.5	27
9	Friday	15	25.5	25
10	Saturday	6	20	40
11	Sunday	10	23.4	35
12	Total	**88**	**163.4**	**182**
13				
14	Week 2	Breakfast	Lunch	Dinner
15	Monday	10	26	30
16	Tuesday	6	22	20
17	Wednesday	15	20	27.5
18	Thursday	19	20	25.5
19	Friday	20	20	30
20	Saturday	10	25	30
21	Sunday	16	18	35
22	Total	**96**	**151**	**198**

What the Macro Does

1. Check the selected cell(s)
   1.1. Stop if the user has selected more than one cell
   1.2. Stop if the selected cell is empty
2. Find the range of all non-empty cells that are continuous with the selected cell
   2.1. Locate the ‘firstRow’: start from the selected cell, keep moving up, stop when we find an empty cell or reach the top row
   2.2. Locate the ‘lastRow’: start from the selected cell, keep moving down, stop when we find an empty cell
3. Insert a formula which applies to all cells in the range ‘firstRow’ to ‘lastRow’ in the cell underneath them
   3.1. Use bold font and a larger font size to emphasize the result cell

Illustration of Step 2.1. Locate the ‘firstRow’

- Starting from the selected cell, we want to find all the non-empty cells that are continuous with the selected cell
- Initially, ‘firstRow’ points to the selected cell
- We keep moving ‘firstRow’ upwards until we meet an empty cell

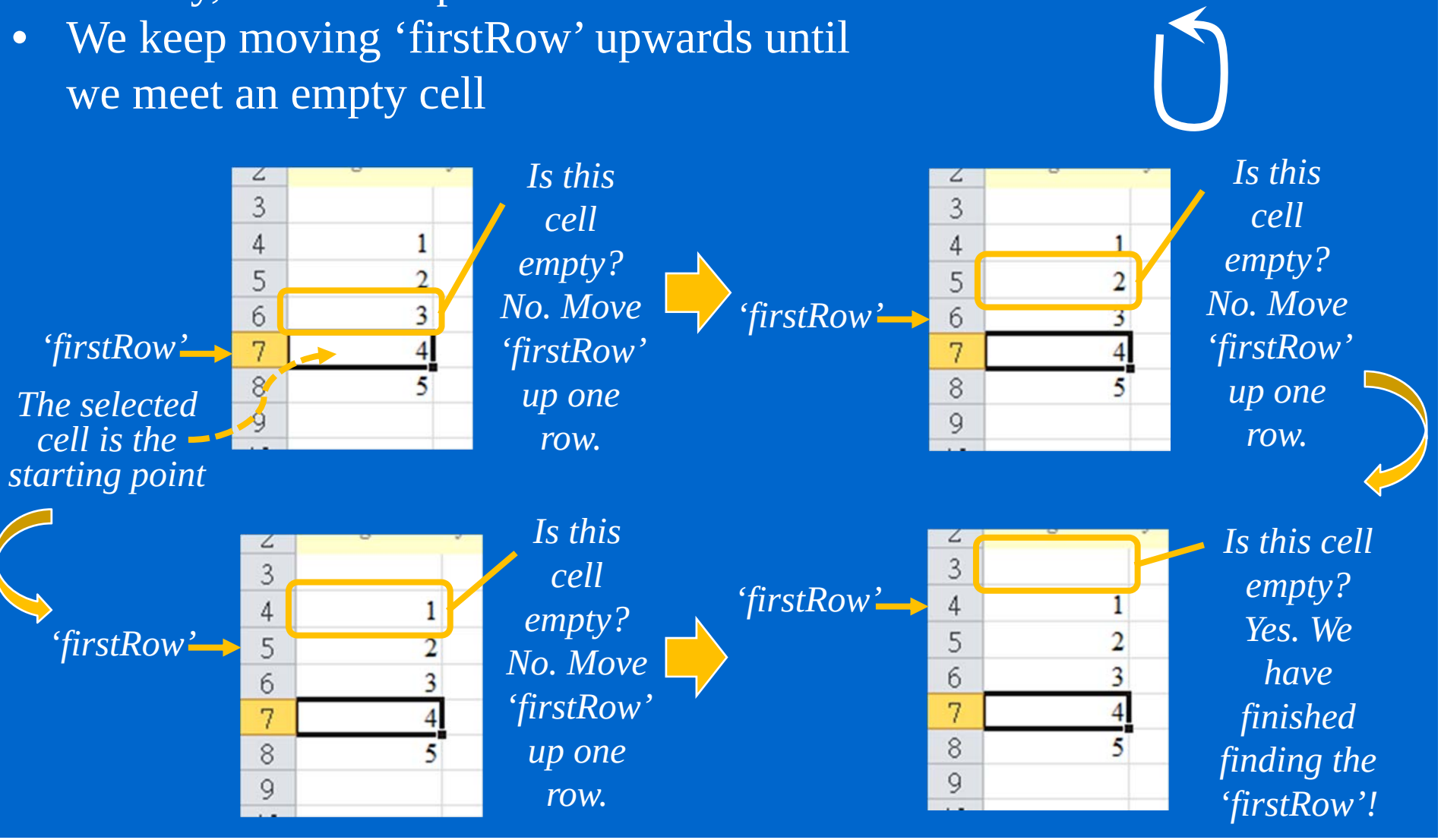
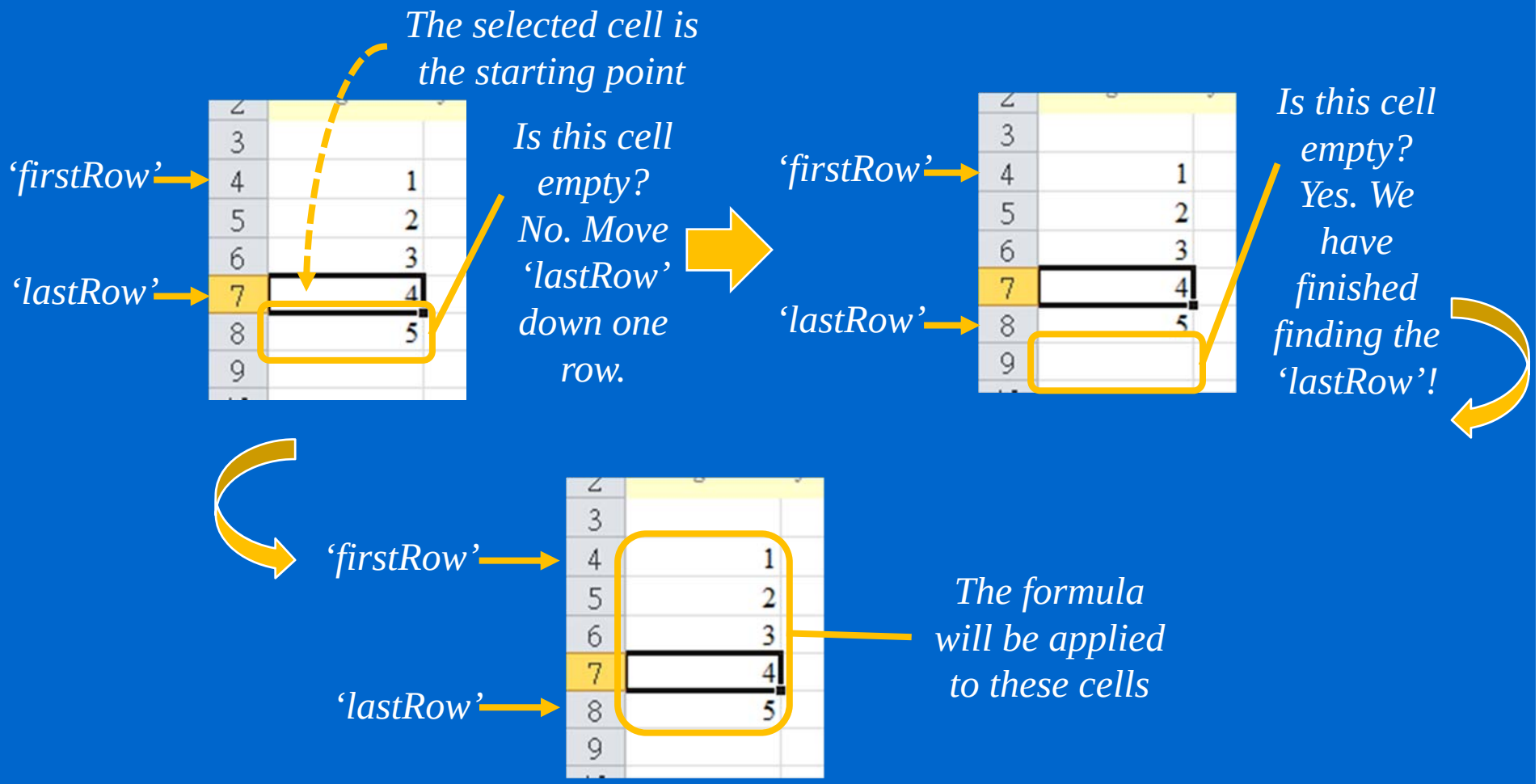


Illustration of Step 2.2. Locate the ‘lastRow’

- Finding the ‘lastRow’ is similar, but this time we move downwards until we find an empty cell



The Macro Code

- This is the complete code for the first macro
- It is shown and discussed in the following two slides

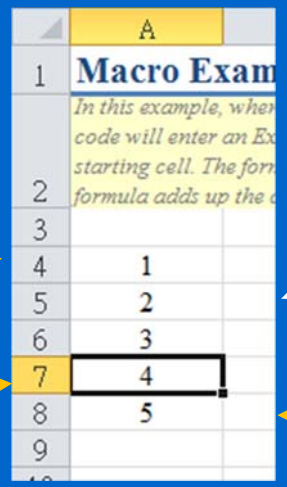
The first loop:

The second loop:

```
Option Explicit ' This forces the code to declare a variable before it can be used

' This function is executed when the button 'Insert a Formula' is clicked
Sub Insert_Formula_Click()
    On Error Resume Next ' Ignore any errors

    ' Test if the current selection is an empty cell
    If Selection = "" Then
        Exit Sub
    End If

    ' Get the current selected column
    Dim thisColumn As Long
    thisColumn = Selection.column

    ' Find the 'firstRow' of the continuous non-empty cells in the same column
    ' If the selected row is not the first row and the cell above the selected cell
    ' not an empty cell, then search upwards and locate the end point of these
    ' continuous non-empty cells
    Dim firstRow As Long
    firstRow = Selection.Row
    Do While firstRow > 1
        If Cells(firstRow - 1, thisColumn) <> "" Then
            firstRow = firstRow - 1
        Else
            Exit Do
        End If
    Loop

    ' Find the 'lastRow' of the continuous non-empty cells in the same column
    ' If the cell below the selected cell is not an empty cell, then search downwards
    ' and locate the end point of these continuous non-empty cells
    Dim lastRow As Long
    lastRow = Selection.Row
    Do While Cells(lastRow + 1, thisColumn) <> ""
        lastRow = lastRow + 1
    Loop

    ' Make a range which includes all the continuous cells
    Dim cellsToInclude As Range
    Set cellsToInclude = Range(Cells(firstRow, thisColumn), Cells(lastRow, thisColumn))

    ' Insert a formula in the cell directly below the range
    ' Set the font style of that cell to bold and a bigger font size
    With Cells(lastRow + 1, thisColumn)
        ' Get the relative address by passing two False as the parameters of Address()
        .Formula = "=SUM(" & cellsToInclude.Address(False, False) & ")"
        .Font.Bold = True ' Set the cell to bold
        .Font.Size = 16 ' Set the font size to 16
    End With

End Sub
```

The Macro Code in Detail 1/2

```
' This function is executed when the button
' 'Insert a Formula' is clicked
Sub Insert_Formula_Click()
    On Error Resume Next ' Ignore any errors

    ' Test if the current selection is an empty cell
    If Selection = "" Then
        Exit Sub
    End If

    ' Get the current selected column
    Dim thisColumn As Long
    thisColumn = Selection.column

    ' Find the 'firstRow' of the continuous
    ' non-empty cells in the same column
    Dim firstRow As Long
    firstRow = Selection.Row
    Do While firstRow > 1
        If Cells(firstRow - 1, thisColumn) <> "" Then
            firstRow = firstRow - 1
        Else
            Exit Do
        End If
    Loop
```



'firstRow'

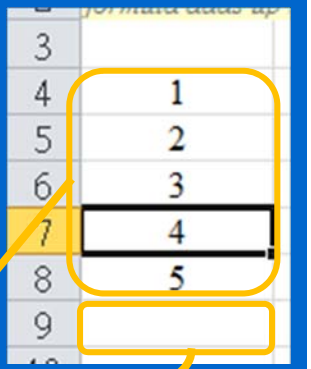
Search upwards to find the 'firstRow' of the group of cells

'lastRow'

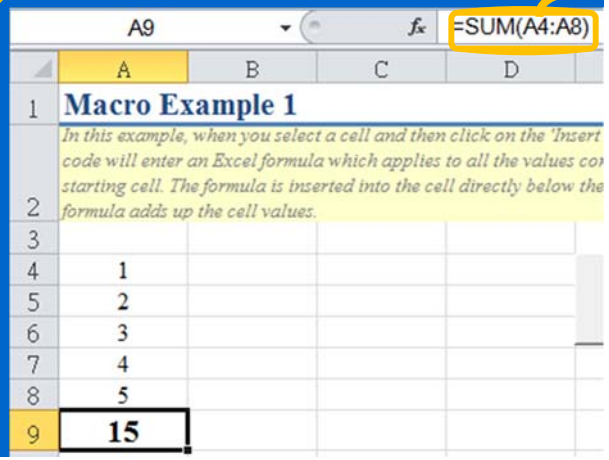
1. *If the row is not the top row*
2. *and the cell above is not an empty cell*
3. *move upwards*

The Macro Code in Detail 2/2

```
    ' Find the 'lastRow' of the continuous
    ' non-empty cells in the same column
    Dim lastRow As Long
    lastRow = Selection.Row
    Do While Cells(lastRow + 1, thisColumn) <> ""
        lastRow = lastRow + 1
    Loop

    ' Make a range which includes all the continuous cells
    Dim cellsToInclude As Range
    Set cellsToInclude = Range(Cells(firstRow, thisColumn),_
                               Cells(lastRow, thisColumn))

    ' Insert a formula in the cell directly
    ' below the range
    With Cells(lastRow + 1, thisColumn)
        .Formula = "=SUM(" & cellsToInclude.Address(False,False) & ")"
        .Font.Bold = True
        .Font.Size = 16
    End With
End Sub
```

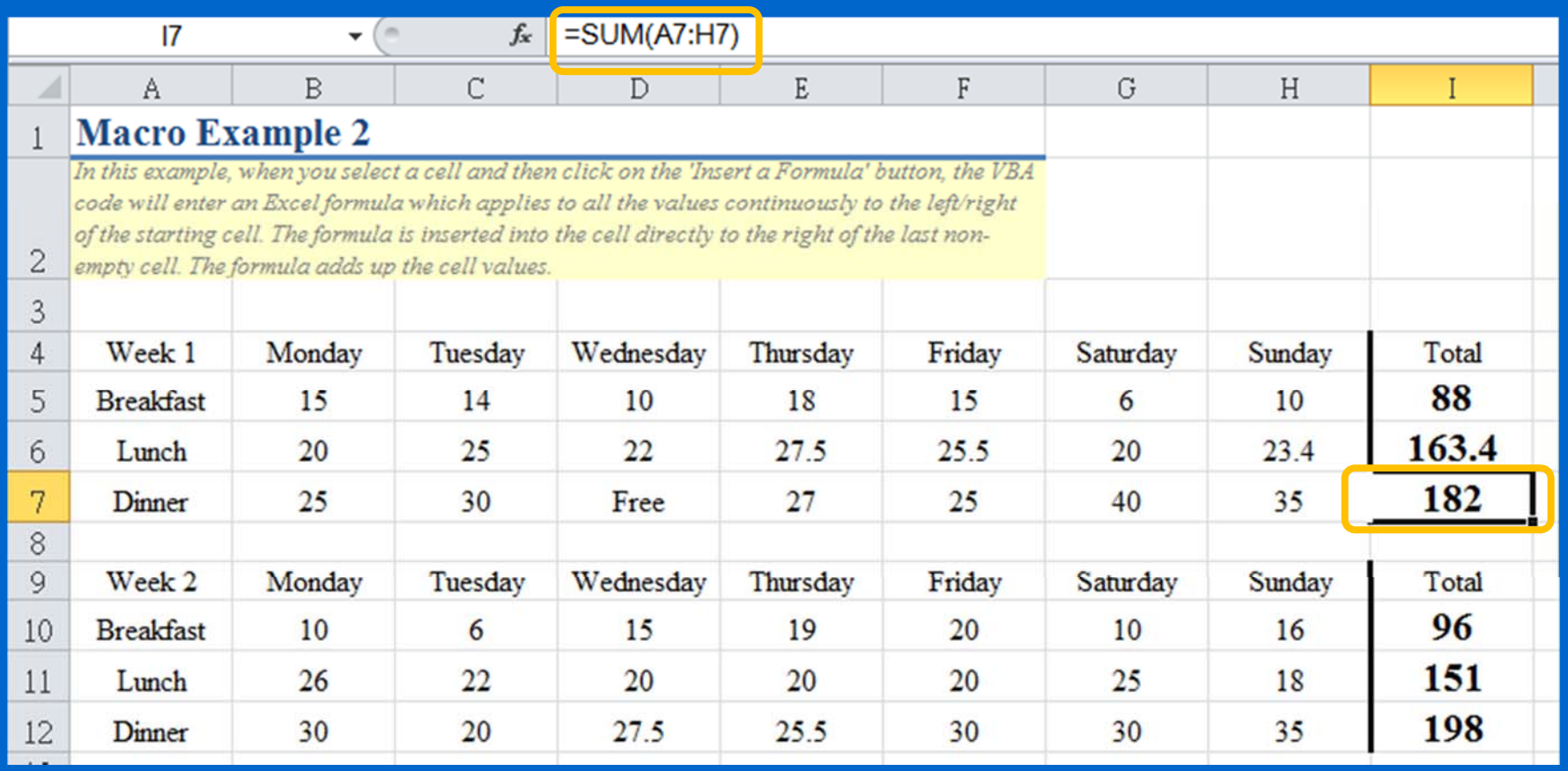


We don't want to use absolute addresses ($)



Macro Example 2

- The second macro is similar to the first macro
- However, instead of summing numbers in a column, it works in rows



I7 | fx | =SUM(A7:H7)

	A	B	C	D	E	F	G	H	I
1	**Macro Example 2**								
2	*In this example, when you select a cell and then click on the 'Insert a Formula' button, the VBA code will enter an Excel formula which applies to all the values continuously to the left/right of the starting cell. The formula is inserted into the cell directly to the right of the last non-empty cell. The formula adds up the cell values.*								
3									
4	Week 1	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday	Total
5	Breakfast	15	14	10	18	15	6	10	**88**
6	Lunch	20	25	22	27.5	25.5	20	23.4	**163.4**
7	Dinner	25	30	Free	27	25	40	35	**182**
8									
9	Week 2	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday	Total
10	Breakfast	10	6	15	19	20	10	16	**96**
11	Lunch	26	22	20	20	20	25	18	**151**
12	Dinner	30	20	27.5	25.5	30	30	35	**198**

What the Second Macro Does

1. Check the selected cell(s)
   1.1. Stop if the user has selected more than one cell
   1.2. Stop if the selected cell is empty
2. Find the range of all non-empty cells that are continuous with the selected cell
   2.1. Locate the ‘firstColumn’: start from the selected cell, keep moving leftwards, stop when we find an empty cell or we reach the leftmost column
   2.2. Locate the ‘lastColumn’: start from the selected cell, keep moving rightwards, stop when we find an empty cell
3. Insert a formula which applies to all cells in the range ‘firstColumn’ to ‘lastColumn’ in the cell on the right
   3.1. Use bold font and a larger font size to emphasize the result cell

The Macro Code

- This is the complete code for the second macro

The first loop:

The second loop:

- It is shown and discussed in the following two slides

```
Option Explicit ' This forces the code to declare a variable before it can be used

' This function is executed when the button 'Insert a Formula' is clicked
Sub Insert_Formula_Click()
    On Error Resume Next ' Ignore any errors

    ' Test if the current selection is an empty cell
    If Selection = "" Then
        Exit Sub
    End If

    ' Get the current selected row
    Dim thisRow As Long
    thisRow = Selection.Row

    ' Find the 'firstColumn' of the continuous non-empty cells in the same row
    ' If the selected column is not the first column, and the cell to the left of the
    ' selected cell is not an empty cell, then search leftwards and locate the end
    ' point of these continuous non-empty cells
    Dim firstColumn As Long
    firstColumn = Selection.Column
    Do While firstColumn > 1
        If Cells(thisRow, firstColumn - 1) <> "" Then
            firstColumn = firstColumn - 1
        Else
            Exit Do
        End If
    Loop

    ' Find the 'lastColumn' of the continuous non-empty cells in the same row
    ' If the cell to the right of the selected cell is not an empty cell, then search
    ' rightwards and locate the end point of these continuous non-empty cells
    Dim lastColumn As Long
    lastColumn = Selection.Column
    Do While Cells(thisRow, lastColumn + 1) <> ""
        lastColumn = lastColumn + 1
    Loop

    ' Make a range which includes all the continuous cells
    Dim cellsToInclude As Range
    Set cellsToInclude = Range(Cells(thisRow, firstColumn), Cells(thisRow, lastColumn))

    ' Insert a formula in the cell directly to the right of the range
    ' Set the font style of that cell to bold and a bigger font size
    With Cells(thisRow, lastColumn + 1)
        ' Get the relative address by passing two False as the parameters of Address()
        .Formula = "=SUM(" & cellsToInclude.Address(False, False) & ")"
        .Font.Bold = True ' Set the cell to bold
        .Font.Size = 16 ' Set the font size to 16
    End With

End Sub
```

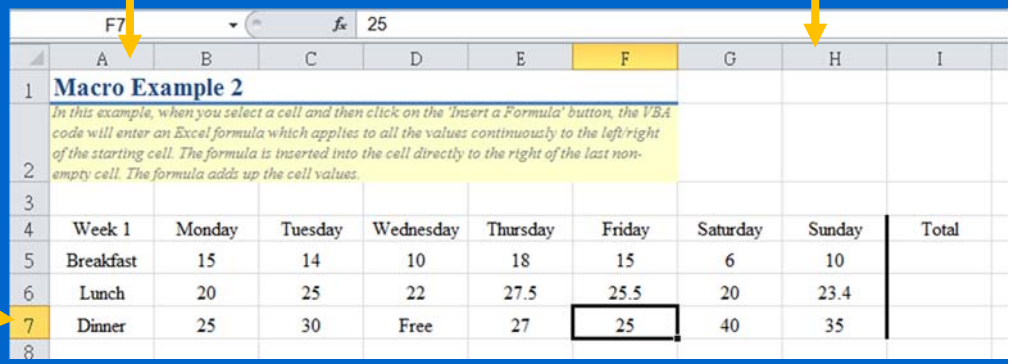
The Macro Code in Detail 1/2

```
' This function is executed when the button
' 'Insert a Formula' is clicked
Sub Insert_Formula_Click()
    On Error Resume Next ' Ignore any errors

    ' Test if the current selection is an empty cell
    If Selection = "" Then
        Exit Sub
    End If

    ' Get the current selected row
    Dim thisRow As Long
    thisRow = Selection.Row
```

Unlike the previous example, we start by finding the selected row instead of the column

'firstColumn'

'lastColumn'



Search leftwards to find the 'firstColumn' of the group of cells

```
    ' Find the 'firstColumn' of the continuous
    ' non-empty cells in the same row
    Dim firstColumn As Long
    firstColumn = Selection.column
    Do While firstColumn > 1
        If Cells(thisRow, firstColumn - 1) <> "" Then
            firstColumn = firstColumn - 1
        Else
            Exit Do
        End If
    Loop
```

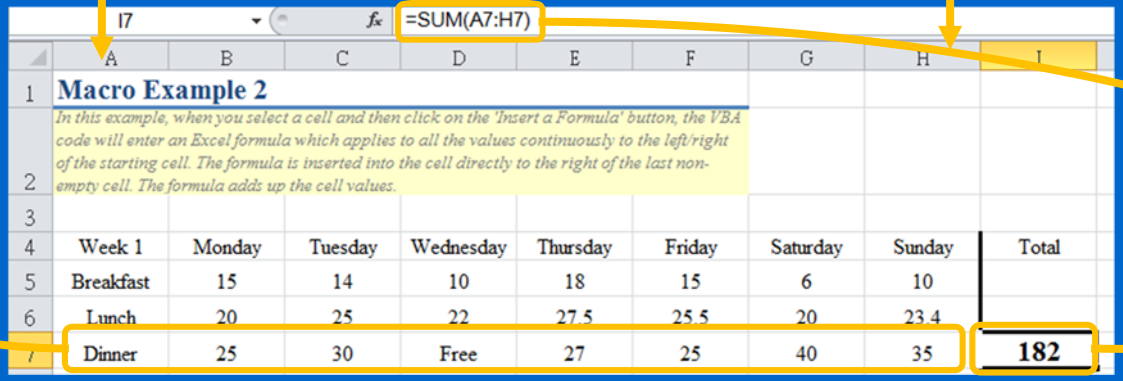
Similar to the previous example, we start from the selected cell looking for the first empty cell. In this example, we search horizontally instead of vertically.

The Macro Code in Detail 2/2

```
    ' Find the 'lastColumn' of the continuous
    ' non-empty cells in the same row
    Dim lastColumn As Long
    lastColumn = Selection.column
    Do While Cells(thisRow, lastColumn + 1) <> ""
        lastColumn = lastColumn + 1
    Loop

    ' Make a range which includes all the continuous cells
    Dim cellsToInclude As Range
    Set cellsToInclude = Range(Cells(thisRow, firstColumn),_
                               Cells(thisRow, lastColumn))
```

'firstColumn'

'lastColumn'




```
    ' Insert a formula in the cell directly on the right of the range
    With Cells(thisRow, lastColumn + 1)
        .Formula = "=SUM(" & cellsToInclude.Address(False,False) & ")"
        .Font.Bold = True
        .Font.Size = 16
    End With
End Sub
```

We don't want to use absolute addresses ($)