COMP1022Q
Introduction to Computing with Excel VBA

Using For Each Loops

David Rossiter and Gibson Lam

Outcomes

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

    1. Understand that some things in Excel are organized in collections

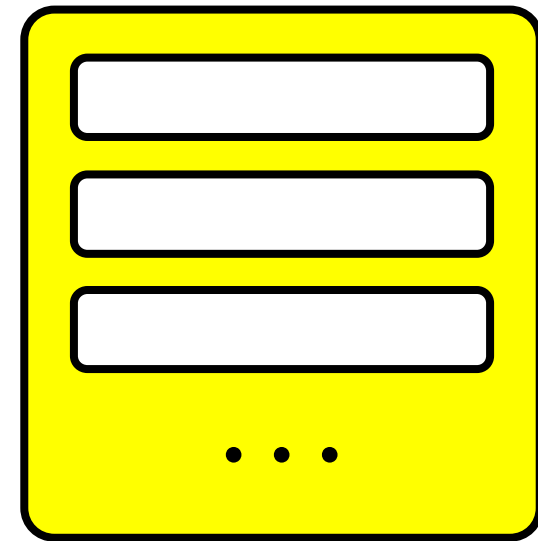   2. Use For Each loops to go through the collections

Simple Variables and Objects

- We have seen simple variables such as Integer/Long, Single/Double, String and Boolean

A simple variable with a single value

- We have also learned different Excel objects such as cells, worksheet and shapes
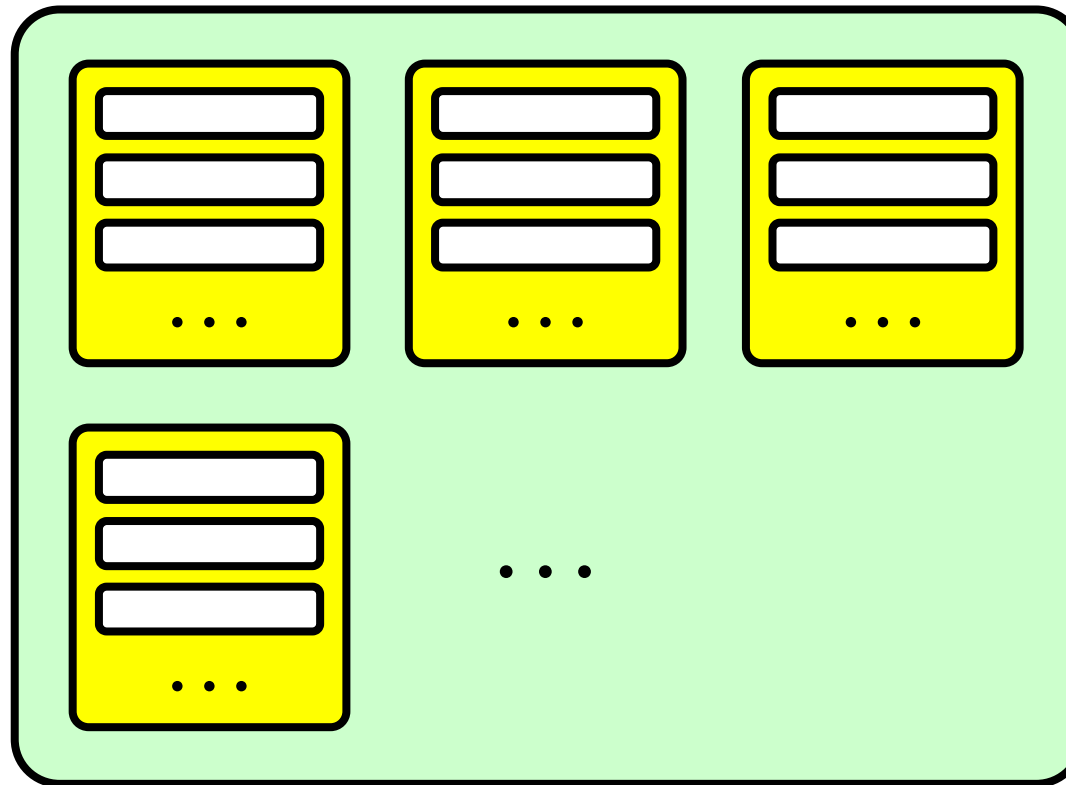
An Excel object containing different properties

A Collection of Objects

- Sometimes we want to do things with a *collection* of objects

Example Collections

- Here are some example collections in VBA:
  - `Range("A1:A10")` is a collection of 10 cells from cell A1 to cell A10
  - `ActiveSheet.Shapes` is a collection of shapes inside the active worksheet
  - `Worksheets` is a collection of all worksheets in the Excel file

- You can use a For Each loop to look at each object inside a collection easily

For Each…Next

`For Each` *object* `In` *collection*

. . .statement(s). . .

`Next` *object*

- A For Each loop executes the statement(s) in the loop for each *object* inside a *collection* of objects
- The For Each loop is particularly useful when you need to work with a collection of things such as some cells in a worksheet

Using For Each Loop with Cells

- You can get a collection of cells using the `Range` command, i.e. `Range("A1:A10")`

- A For Each loop can run for each cell using such collection

- For example, the following loop runs for each cell from cell B1 to B5:

```
Dim Cell As Range
```

You need to use the correct type for the variable

```
For Each Cell In Range("B1:B5")
    ...
Next Cell
```

One at a time, each cell is stored in this variable; there is no need to use `Set` here

An Example – Highlighting Cells

- In this example, a For Each loop is used to highlight cells which contain the same value as the input cell

Highlighted cells which contain the same value as the input cell

The input cell

Highlighting Cells – The Loop

- Some numbers to be highlighted by the For Each loop is in cells B4:F8, i.e. `Range("B4:F8")` in VBA

	B	C	D	E	F
4	1	2	2	5	8
5	5	6	6	8	3
6	1	4	10	3	4
7	9	8	2	4	2
8	7	3	5	2	1

- The loop therefore looks like this:

The loop first looks at B4, then C4, … up to F8

```
For Each Cell In Range("B4:F8")
          ...
Next Cell
```

Highlighting Cells – Loop Content

- The content of the loop simply compares the content of the current cell with the input cell

- If they are the same, the code will highlight the cell with yellow; otherwise, it will be white

The input cell

```
For Each Cell In Range("B4:F8")
    If Cell.Value = Range("F10").Value Then
        Cell.Interior.ColorIndex = 6
    Else
        Cell.Interior.ColorIndex = 0
    End If
Next Cell
```

Highlight the cell using a yellow background

Otherwise, no background colour

Using For Each Loop with Shapes

- The For Each loop works for any collection of things

- For example, it can also work with Excel shapes

- You can get all the shapes in the current worksheet using `ActiveSheet.Shapes`

- The following loop runs for each shape in the worksheet:
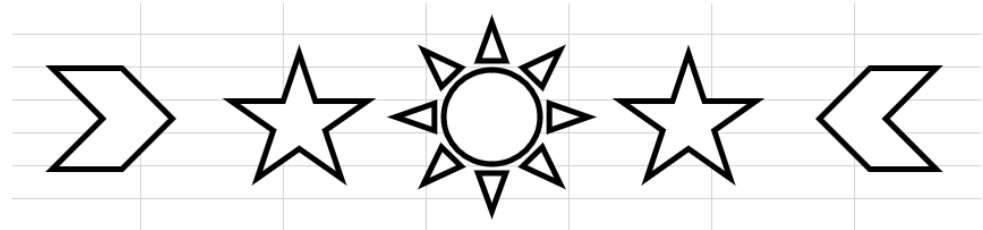
```
Dim ThisShape As Shape

For Each ThisShape In ActiveSheet.Shapes
    ...
Next ThisShape
```

ActiveSheet means 'the worksheet you are currently using'

One at a time, each shape in the worksheet is stored in this variable

An Example with Shapes

- In this example,
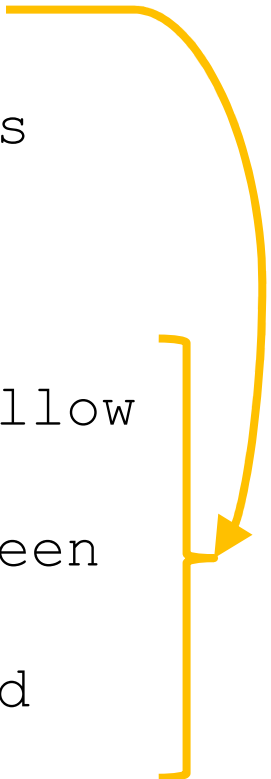 a few shapes have
  been inserted in
  a worksheet

- When the macro in the example is run, the shapes
  are changed to different colours randomly
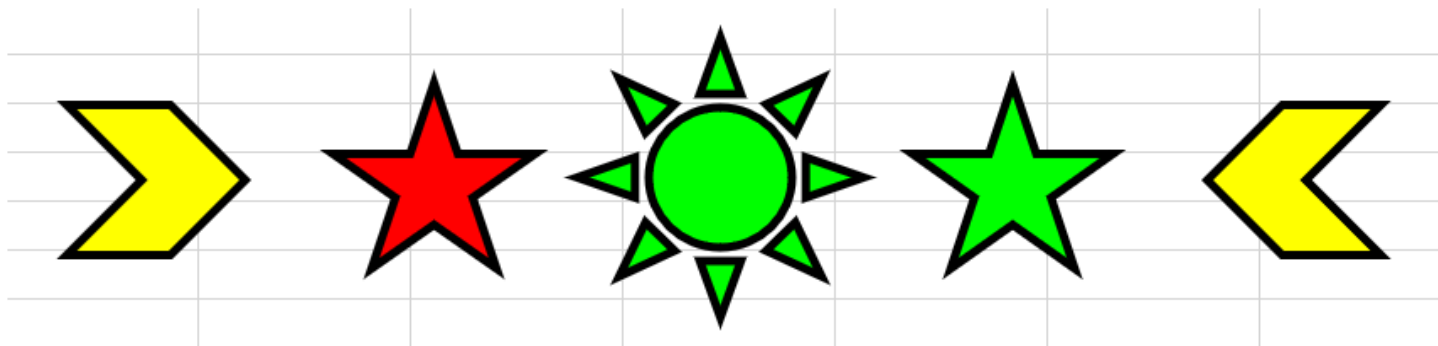
The For Each Loop With Shapes

- Here is the loop of the example

- It changes each shape using a random colour:

```
For Each ThisShape In ActiveSheet.Shapes
   RandomValue = Int(Rnd() * 3) + 1

   If RandomValue = 1 Then
     ThisShape.Fill.ForeColor.RGB = vbYellow
   ElseIf RandomValue = 2 Then
     ThisShape.Fill.ForeColor.RGB = vbGreen
   ElseIf RandomValue = 3 Then
     ThisShape.Fill.ForeColor.RGB = vbRed
   End If
Next ThisShape
```

Another Example

- Here is another example that changes the position of some shapes in a worksheet
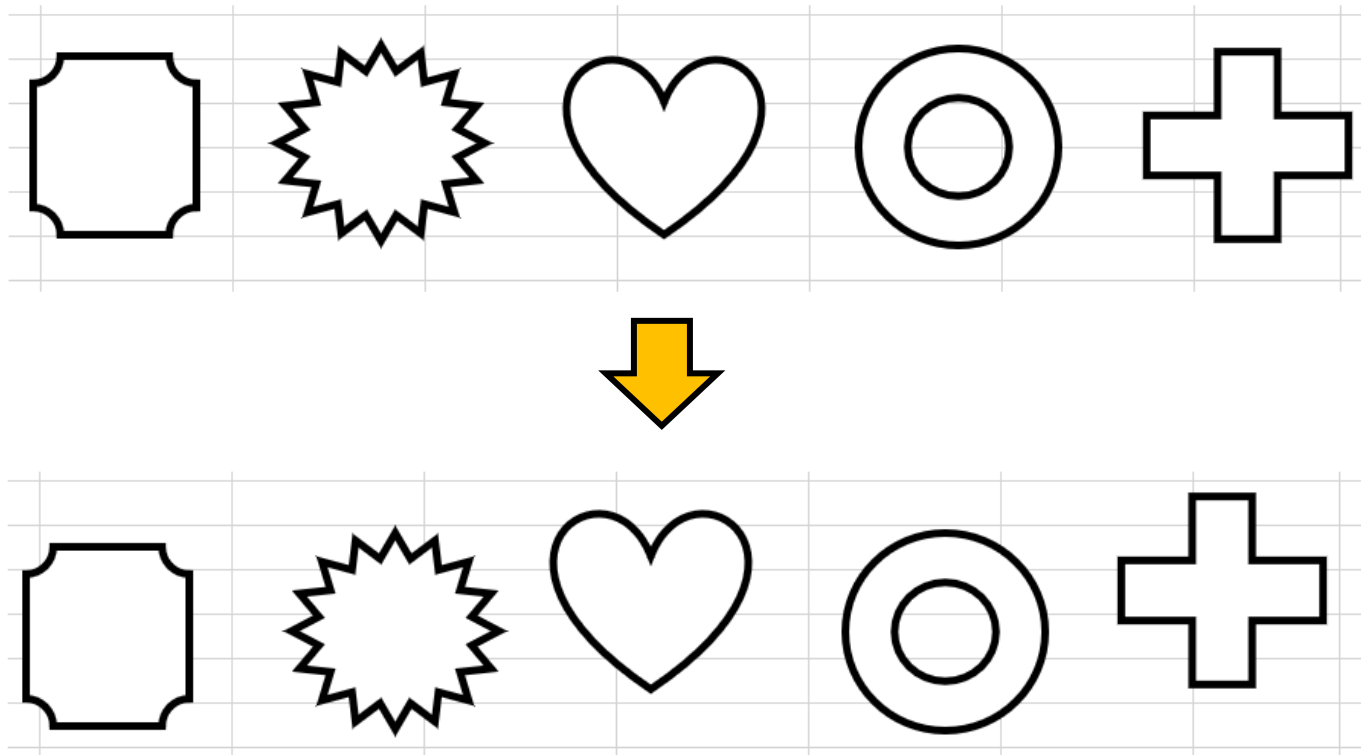
Moving the Shapes

- The For Each loop goes through the shapes and moves them randomly

```
For Each ThisShape In ActiveSheet.Shapes
    RandomX = Int(Rnd() * 11) - 5
    RandomY = Int(Rnd() * 11) - 5

    ThisShape.Left = _
        ThisShape.Left + RandomX
    ThisShape.Top = _
        ThisShape.Top + RandomY
Next ThisShape
```
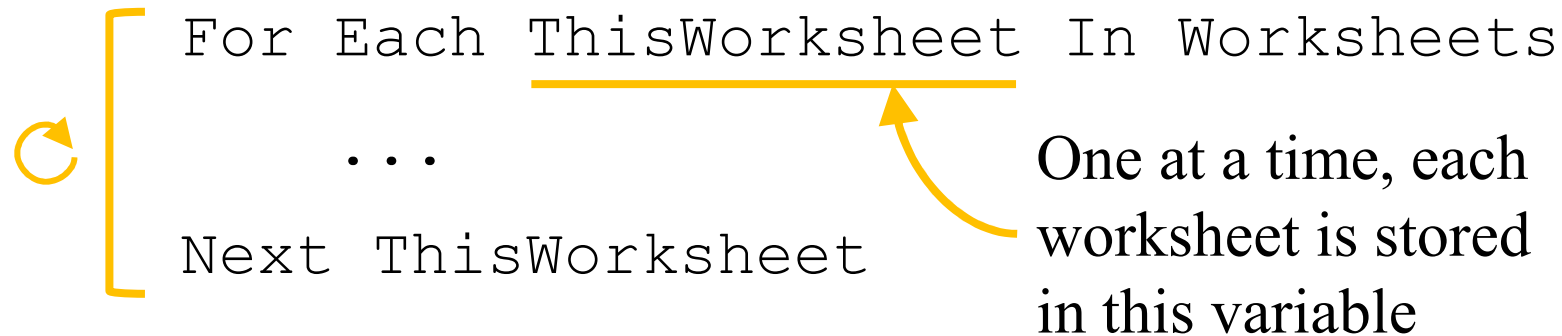
Generate random numbers from -5 to 5

The Left and Top properties set the position of a shape

Using For Each Loop with Worksheets

- All worksheets are stored together in a collection

- You can access all of them using `Worksheets`

- The following loop runs through each worksheet:
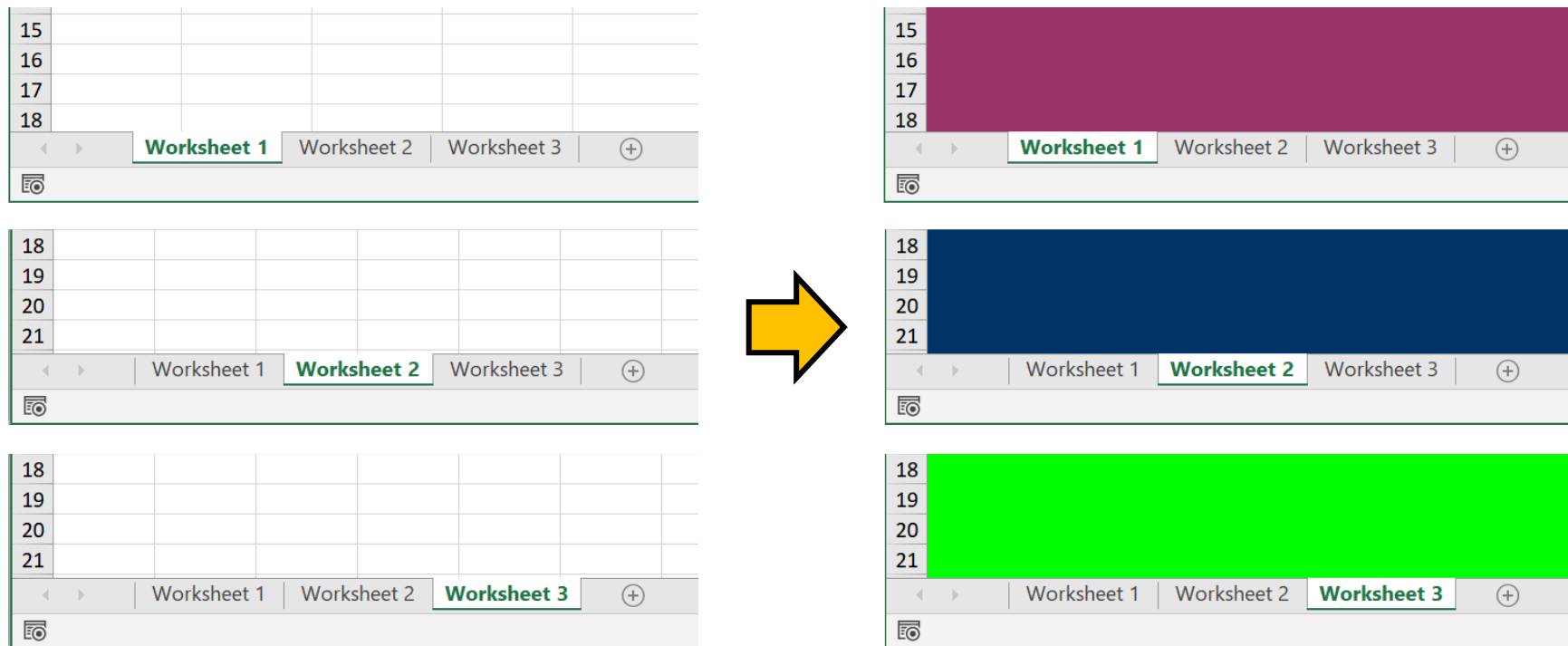
```
Dim ThisWorksheet As Worksheet

For Each ThisWorksheet In Worksheets

      ...

Next ThisWorksheet
```

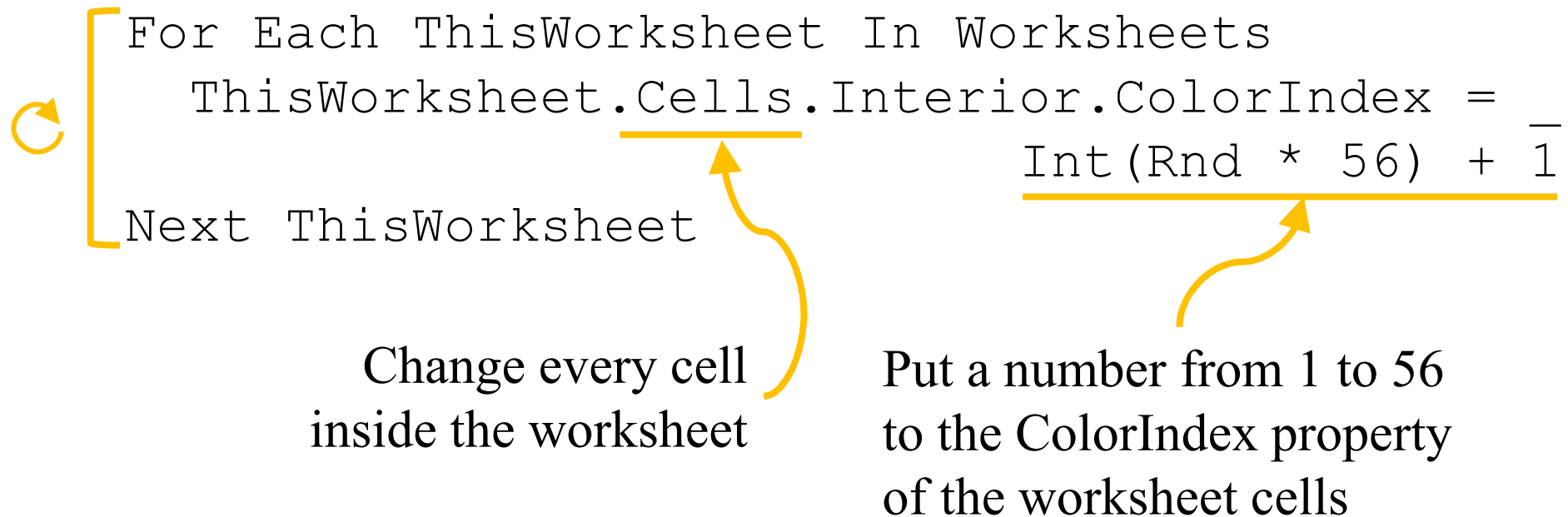One at a time, each worksheet is stored in this variable

Changing Worksheet Colours

- In the following example, the macro changes the colour of the three worksheets randomly

The Loop with Worksheets

- The For Each loop simply goes through all worksheets and changes their colours to a random colour index from 1 to 56

```
For Each ThisWorksheet In Worksheets
    ThisWorksheet.Cells.Interior.ColorIndex = _
                          Int(Rnd * 56) + 1
Next ThisWorksheet
```

Change every cell
inside the worksheet

Put a number from 1 to 56
to the ColorIndex property
of the worksheet cells