* Object properties
  + Object properties = Something that object has
  + Properties come after the object hierarchy
  + Every object in excel has predefined sets of properties (e.g. properties for cars are: color, size, type, engine, etc)
    - Properties themselves can return objects. E.g. shoes.heel.materials – heel is property of shoes, but it’s also an object. Excel example: Range(“A2”).Interior.Color
  + Property can be ready only or Write:
    - Read only: Range(“A10”).Address
    - Read & Write: Range(“A1”).Value
      * E.g.: ActiveCell.Address
    - Read & Write: Range(“A2”).Font.Color
      * E.g.: vbRed, vbBlue, etc.
* Object methods
  + Object methods = Something that object does. E.g. cars can start, stop, crash, etc.
  + Methods can have additional info: start slowly, stop quickly, etc.
    - E.g.: Car.Stop Quickly; Car.Stop StopStyle:=Quickly
  + Methods can change object properties
  + Methods can have arguments – meaning they can handle more specific instructions on what to do. E.g.:
    - Range(“A2”).Clear 🡪 no further arguments needed
    - Range(“A2”).Delete 🡪 1 optional no argument (shift cells up, down, left, right)
    - Sheet1.Copy 🡪 2 optional arguments (copy to before or after)
* How to find the object, property & method
  + Don’t expect to remember everything. Here’s where you can find properties and methods:
    - Use macro recorder to get object names, properties, and methods
    - Use the object library (F2)
    - Use IntelliSense: VBA suggests the right properties and methods (similar to auto complete)
    - Use complete word in VBA to get suggestions (Ctrl + space)
    - Use the “Immediate Window” to query or test code
    - Google for info
* Key Takeaways: Object Model
  + You refer to an object through its position in the object hierarchy. The dot (.) is used as a separator. If you do not specify the parent, excel assumes it’s the active object.
  + You don’t need to select objects to manipulate them. The macro recorder “selects” but to write code, it’s more efficient not to refer directly to objects & properties (exceptions apply).
  + Objects have specific properties & methods.
  + Properties can return a reference to another object. For example Range(“A2”).Interior.Color: The interior property returns an interior object which has the color property.
  + Macro and VBA code is kept inside Sub Procedures (for functions it’s Function Procedures)
  + To find the correct object, property or method you can record macros, use the object library, use MSDN help, Intellisense, google, and immediate window to query and test code.

Section 4: Referencing ranges, worksheets & workbooks

* Overview: referencing ranges, worksheets, & workbooks
* Referring to ranges & writing to cells in VBA
  + Different methods to write to cells – using rows, columns, range referencing – examples:
    - *‘start sub procedure*

Sub ReferToCells()

*‘select cells in active worksheet as object; clear cells is method*

Cells.Clear

*‘select range A1 in active worksheet as object; enter value as “1st” is method. ‘Another way to select A1 is by “Cells(1,1)”*

Range("A1").Value = "1st" 'Cells(1, 1) = "1st"

*‘select range of cells*

Range("A2:C2").Value = "2nd"

Range("A3:C3,E3:F3").Value = "3rd"

Range("a" & 6, "C" & 6) = "6th"

*‘Within the range of A4:C7, refer to the 4th row, 2nd column*

Range("A4:C7").Cells(4, 2).Value = "7th"

*‘Starting in cell A1, refer to +7th row down (row 8) and +2nd column over (col C)*

Range("A1").Offset(7, 2).Value = "8th"

*‘This treats B1 as the end of the range from which the offset occurs. So starting in B1, refer to 8th row down and +1 column over. When range is selected with offset, I think it replicates the range wherever the offset is being entered. In this example, you wind up with “9th” in cells B9:C9.*

Range("A1:B1").Offset(8, 1).Value = "9th"

*‘Referencing using named ranges (“LastOne” in this example)*

Range("LastOne").Value = "10th"

*‘referencing rows and columns to change height and widths:*

Rows("12:14").RowHeight = 30

Range("16:16,18:18,20:20").RowHeight = 30

Columns("E:F").ColumnWidth = 10

Range("H:H,J:J").ColumnWidth = 10

Range(Columns(1), Columns(3)).ColumnWidth = 5

*‘select entire active workbook as object, columns as object, autofit as method.*

Cells.Columns.AutoFit

End Sub

* Most useful range properties & methods