Lesson 22 - VBA Variables and Functions

1. VBA Variables
   - Data types: `Single, Double, Integer, Long, String, Boolean, Variant`, plus can make arrays from any of these.
   - Dim statements: `Dim a As Integer(1 to 3)` for example sets up an integer array
   - `Option Explicit` at beginning of module forces you to Dim everything, which is good programming practice

2. VBA Procedures
   - Procedures, including Subs and Functions, produce some action when they are called
   - Subroutines – can produce multiple results, some of which can be passed out through the list of arguments

```
Sub Add(a As Double, b As Double, Optional c = 1# As Double)
      a = b + c
End Function
```

     Parameters are passed *by reference*, meaning if they are changed inside the subroutine, they are changed everywhere. Put `ByVal` in front of the parameter name to pass *by value*, meaning a copy is made and passed into the subroutine.
   - Functions – produce one result which is passed out through the function name when called

```
Function Getsome(a As Double, b As Double) As Double
      Getsome = a^b
End Function
```

3. Using Excel Built-In Functions inside VBA (they are not the same!)
   `Application.WorksheetFunction.Name()`

4. Making VBA Customized Functions to use inside Excel worksheets
   `Public Function(…)`

5. Declaring Arrays
   a. Syntax
      `Dim aname(sbs) [As type]`
      where *aname* is the name of the variable, *sbs* is the number of entries in the array, and *type* is the type of variable.
   b. Examples
      i. `Dim x(4) As Double`
         1. x is an array with 5 entries
         2. Subscripts range from 0 to 4
      ii. `Dim A(1 to 10) As Double`

1. A is an array with 10 entries
2. Subscripts range from 1 to 10

iii. `Dim T(1 to 6, 1 to 6) As Integer`

1. T is a 6 by 6 array

6. Useful commands when working with arrays
   a. `Cells(row, column)` references a particular cell in a worksheet or a range.
      i. `Cells(1,1).Value = 10` is the same as `Range("A1").Value = 10`
      ii. `A = Range("B3:G10").Cells(2,3).Value` is the same as `A = Range("D4").Value`
   b. `Offset(RowOffset, ColumnOffset)` references a new range, offset from the original ranges.
      i. `ActiveCell.Offset(-1,1).Value = 10`
      ii. `ActiveCell.Offset(2,-1).Select`
   c. `Count` returns the number of cells in a range.
      i. `nrows = Selection.Rows.Count`
         1. Counts the number of rows that are currently selected

      `nrows = Range("A1:B5").Rows.Count`
         1. Counts the number of rows in the Range A1:B5

      ii. `ncols = Selection.Columns.Count`
      `ncols = Range("A1:B5").Columns.Count`

7. Dim and ReDim
   a. Sometimes, you don't know beforehand the size of the array
   b. In such times, use a Dim/ReDim
   c. Syntax

      `Dim aname() as Type`
      …
      `ReDim [Preserve] aname(sbs) [As type]`
   d. Preserve keyword keeps the existing data in the array.
8. Loops, Arrays, and Interfacing with Excel Objects
   a. Usually, you will use arrays in conjunction with loops and worksheet objects
   b. The Cells and Offset Command are needed in these applications
9. Passing Arrays to Procedures and Bubble Sorts
   a. When you pass an array to a procedure, you pass the variable name with ()'s but nothing in the ()'s.
   b. Syntax

      `Call mean(x(),answer)`
10. Multi-dimensional arrays
   a. Often, you need to declare multidimensional arrays (such as matrices)
   b. Syntax

      `Dim aname(lb To ub, lb To ub) As type`
11. Exercise 1: Determinants

The *determinant* is a property of a square matrix. One use of the determinant is to *determine* if a system of linear equations has a solution. If the determinant of the matrix is non-zero, the system of equations has a unique solution. For a 3x3 matrix

$$A = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$$

the determinants can be determined by *cofactor expansion* according to the following.

$$\det(A) = a\begin{vmatrix} e & f \\ h & i \end{vmatrix} - b\begin{vmatrix} d & f \\ g & i \end{vmatrix} + c\begin{vmatrix} d & e \\ g & h \end{vmatrix}$$

$$\det(A) = a(ei - hf) - b(di - gf) + c(dh - ge)$$

Write a function that calculates the determinant of a 3x3 matrix.

12. Exercise 2

With reference to the x,y data on the Exercise2 worksheet, write a VBA program that does the following:

a. Reads the *x* values into an array `x(i)` and the *y* values into an array `y(i)`.
b. For each relative maximum, enters "Relative Maximum" in Column C next to the *x,y* values.
c. For each relative minimum, enters "Relative Minimum" in Column C next to the *x,y* values.
d. For the absolute maximum, enters "Absolute Maximum" in Column D next to the *x,y* values
e. For the absolute minimum, enters "Absolute Minimum" in Column D next to the *x,y* values