



Visual Basic Programming

Chapter 1

Visual Basic Programming

Visual Basic (VB) is a programming language for Windows applications

❖ We will use it to:

- Run our algorithms on a computer
- Assist us in making more effective use of Excel

Visual Basic Programming

- ❖ Creating an algorithm can be difficult
 - Think through the problem and solve it without using a computer

However, with an algorithm, writing the program becomes easy

Translate the instructions in the algorithm to Visual Basic instructions



Visual Basic Programming

- ❖ Problems encountered in a Visual Basic program
 - Syntax errors (Grammatical and spelling mistakes)
 - Semantic errors (Program runs but errors in logic lead to incorrect results)
 - Run time errors (Program fails when running ie. dividing by zero)

Visual Basic Editor

❖ The Visual Basic Editor

- Consists of 4 windows
 - Main, Code, Project Explorer, Properties

1- Main window

- Title bar, menu bar & standard toolbar

2- Code window

- Insert – Module
- To write code for procedures

Visual Basic Editor

3- Project Explorer window

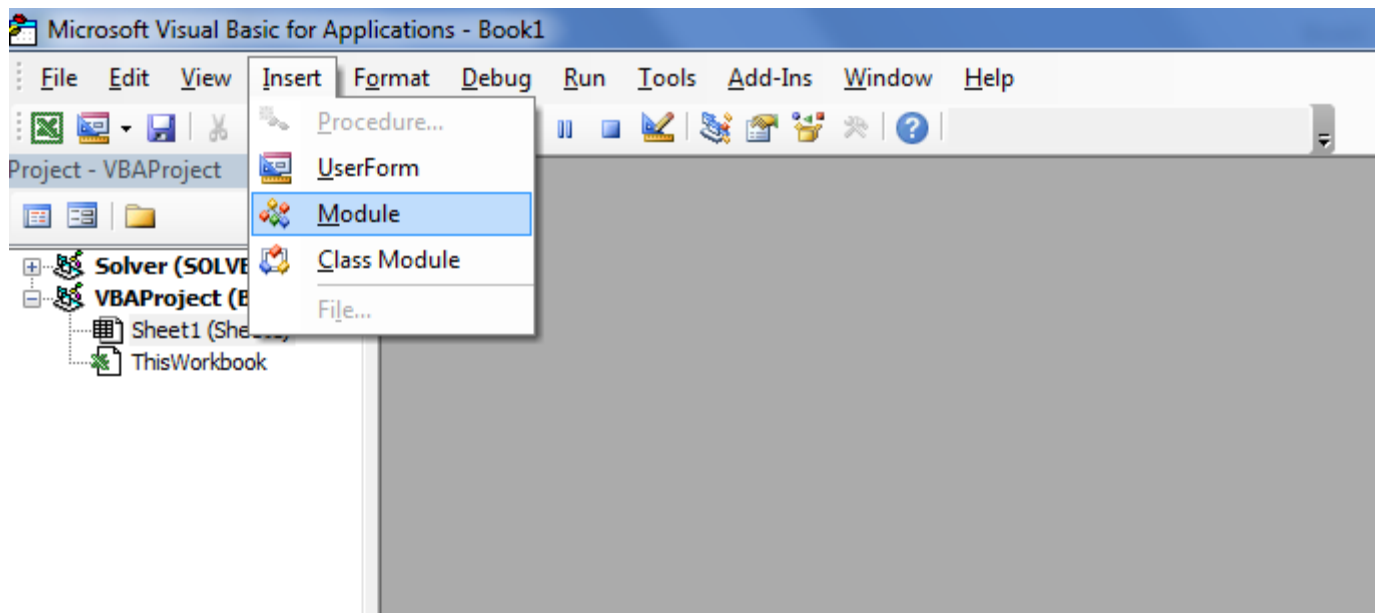
- Shows the open project (name of file containing the project)
- Shows the worksheets & modules in the project (in a format similar to Windows Explorer)
- Has 3 buttons
 - View Code: view the code in the module selected
 - View Object: view the worksheet selected
 - Toggle Folders: control the display of the folder

4- Properties window

- To change properties for any worksheet or module
- Properties control an object's appearance & behaviour

Visual Basic Editor

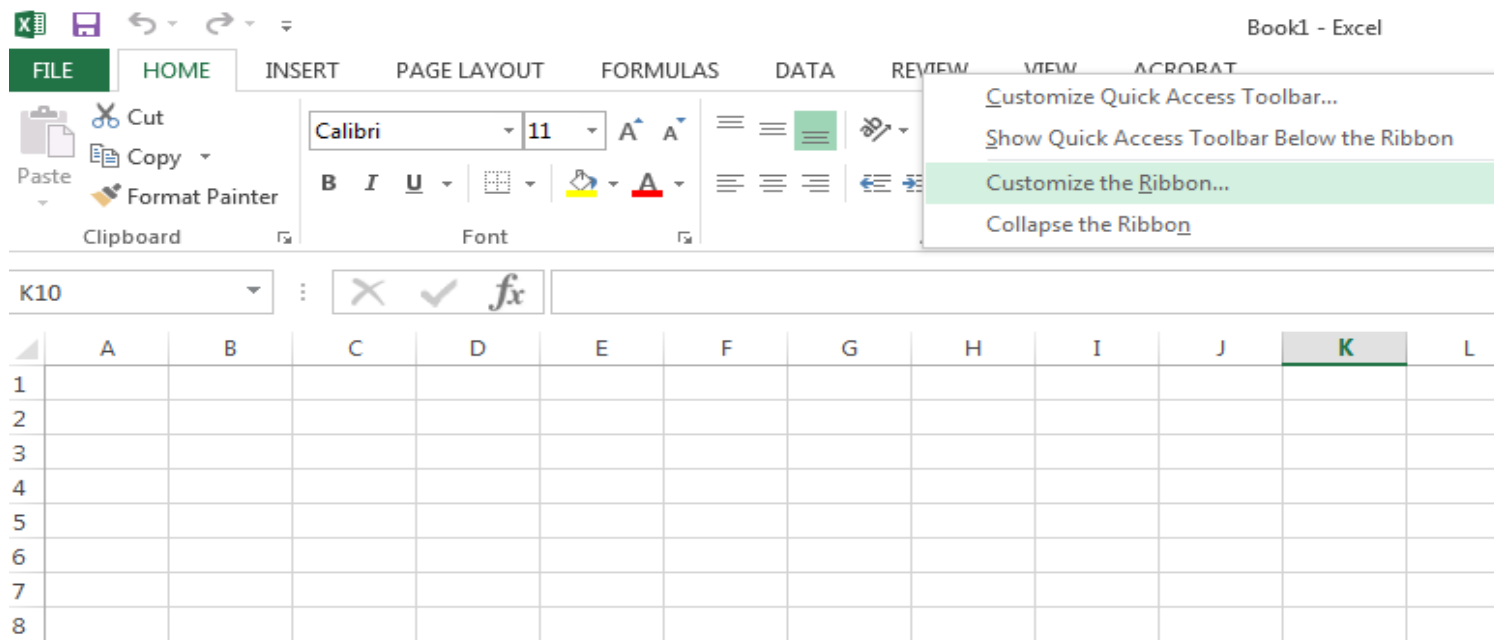
- ❖ Where do we enter the VB code?
 - Visual Basic Editor in Excel
 - You can open the Visual basic Editor in 2 ways
 - **1st method**
 - press Alt + F11 → Insert – Module → Enter the code



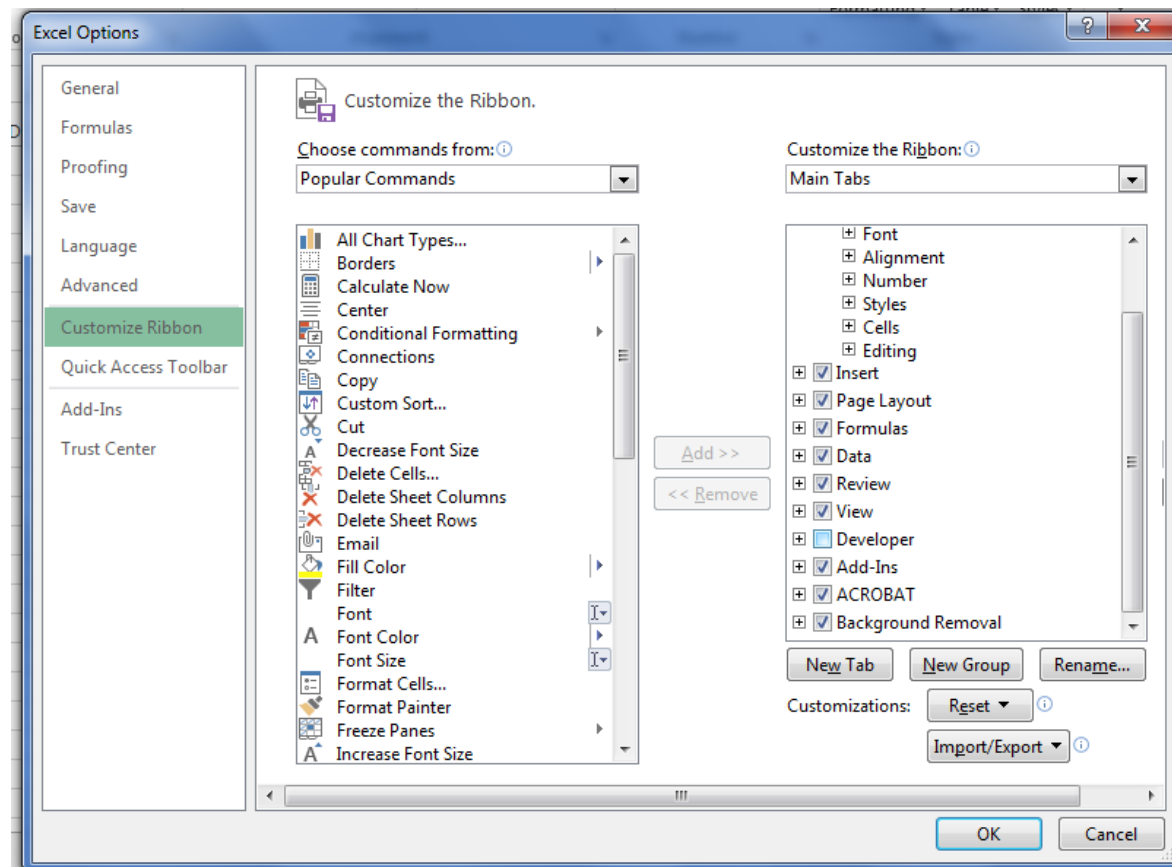
Visual Basic Editor

– 2nd method

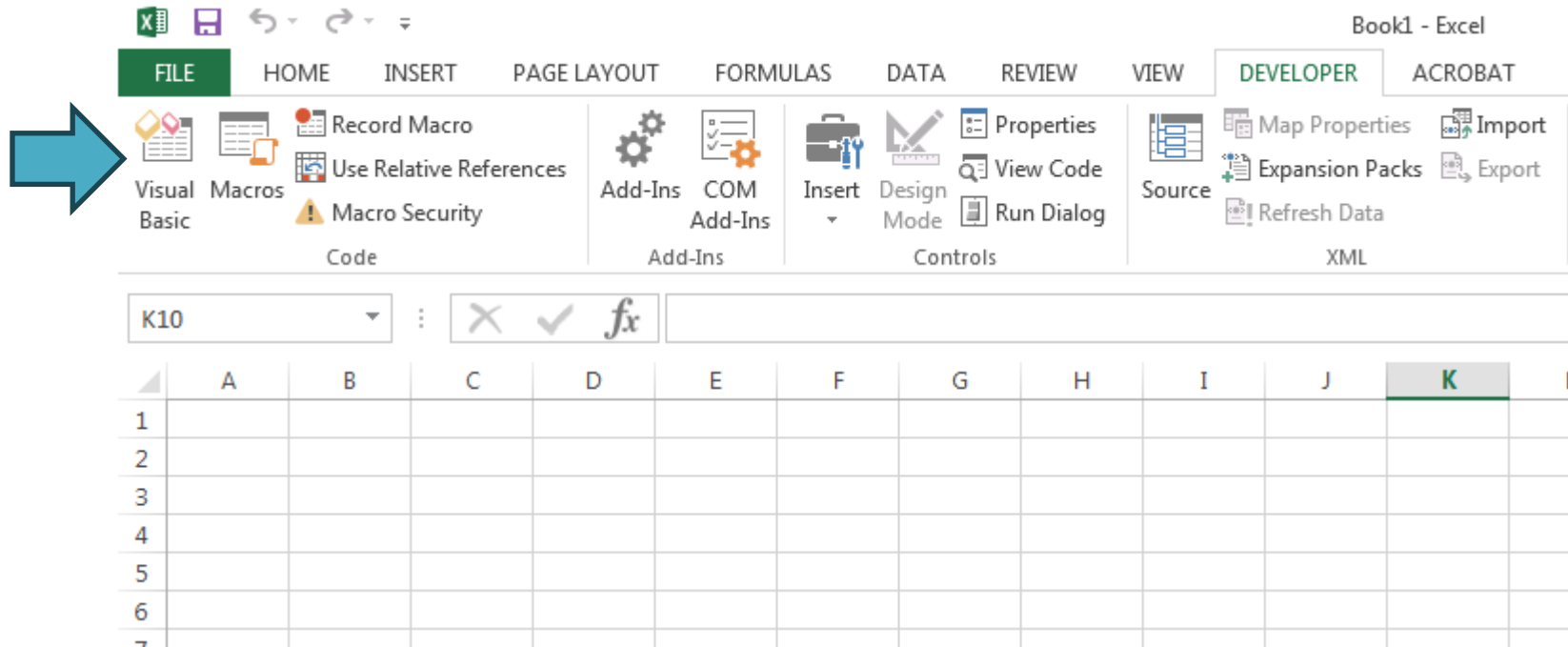
- Place the mouse over any tab button and right click as shown below
- Click on the 3rd option (Customize the Ribbon)



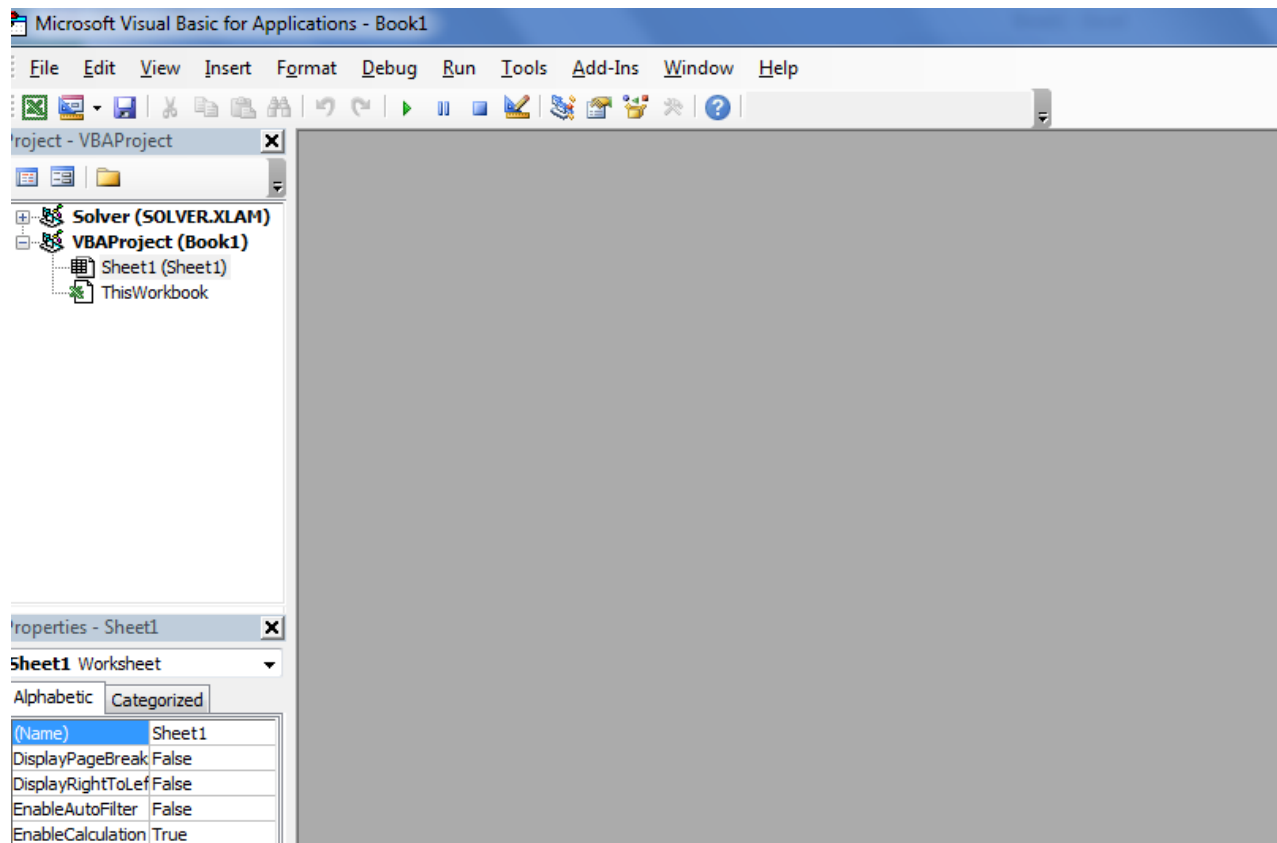
- Put a check mark next to Developer and press OK



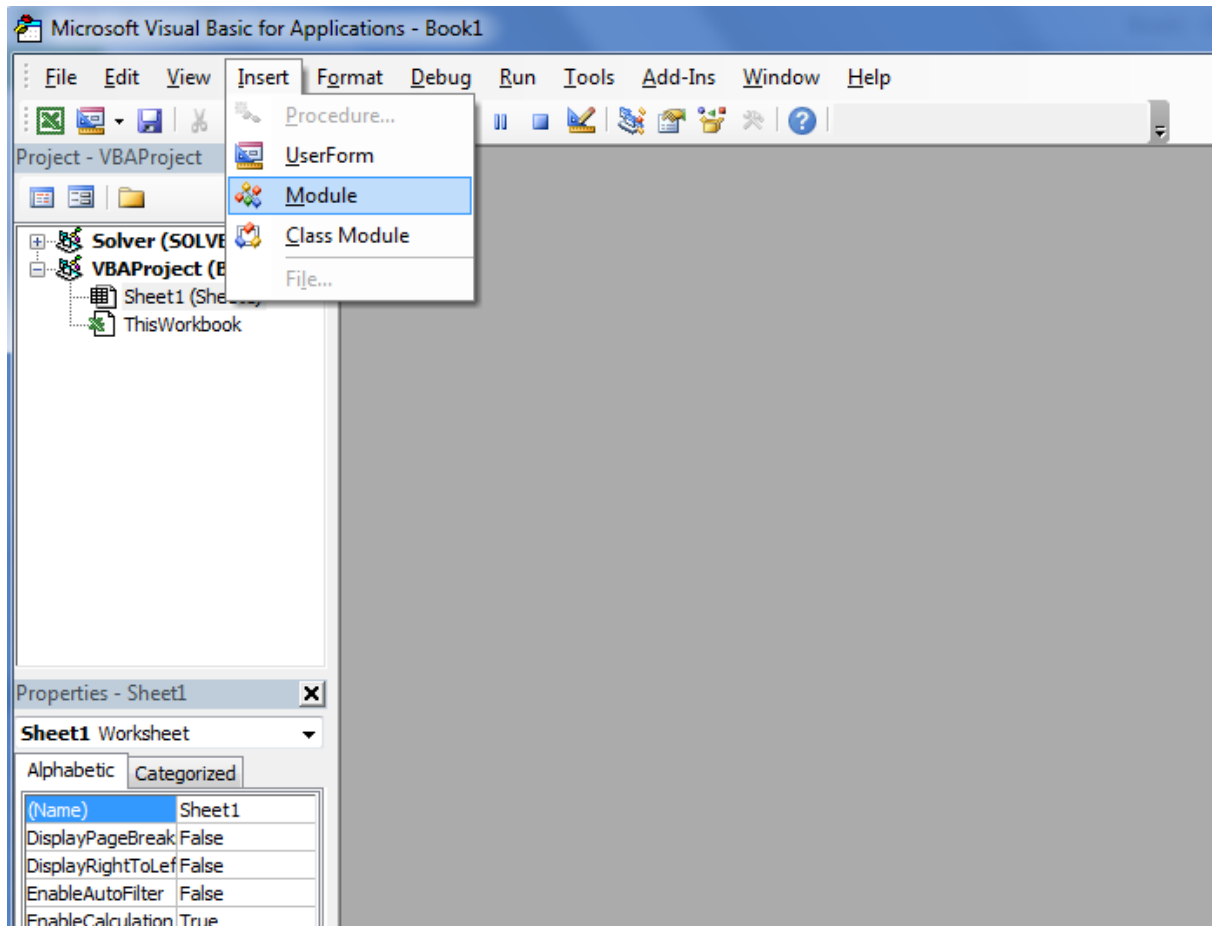
- Now click on the Developer tab, then click on Visual Basic



– The VB editor will look like this

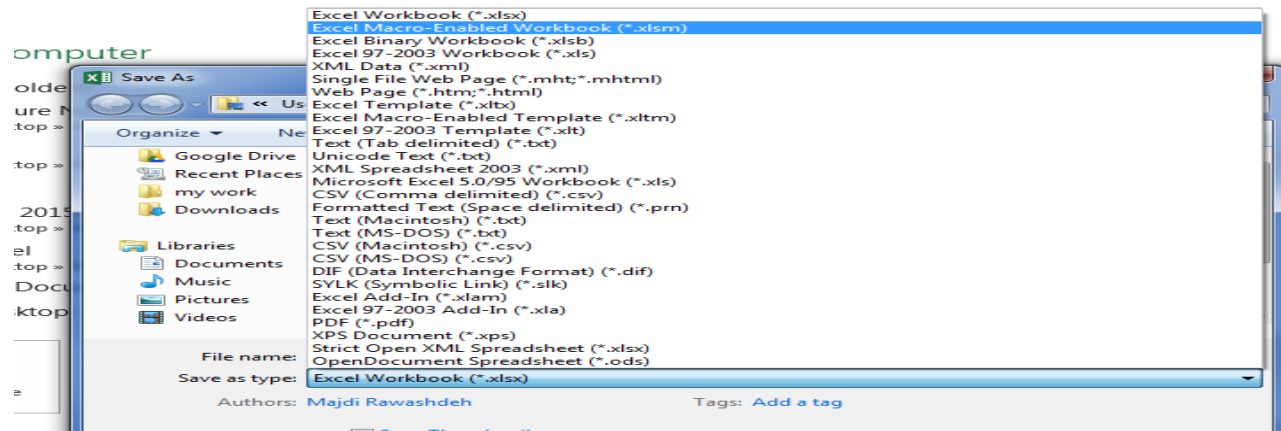


– Click on **Insert** and choose **Module** to insert your code



Remember the VB code has to be inserted in the **Module** window

- ❖ Make sure you **Save** your excel file **As** **Excel Macro-Enabled Workbook** the file name will have **.xlsm** extension otherwise the VB code will not work.



❖ Variable

- Is an address in RAM that has a name and contains a value
- The value can change as the code is executed
- Can hold only one value at a time

❖ Constant

- Has a value that does not change as the code is executed

❖ Names

- Begin with a letter
- Contain only letters, numbers and _ (underscore)
- Not case sensitive
- Cannot be a Visual Basic reserved word

❖ Declaring variables and constants in a VB program

- Done with a DIM statement for a variable
 - Also specify the data type with the As clause
 - Advantages
 - Code is easier to read
 - Speeds up execution of code
 - Conserves memory space
- Done with a CONST statement for a constant

❖ Option Explicit must be used in the first line

- Forces all variables and constants to be declared
- (also useful to detect misspelled variable and constant names)

- ❖ Enter `OPTION EXPLICIT` on the first line
- ❖ All programs begin with `SUB` and end with `END SUB`
- ❖ Comments are lines that are ignored by the computer.
 - We use them to make our code easier to read
 - Everything to the right of a ' (single quote) is a comment

Write a VB program to find the average of three numbers?

Algorithm

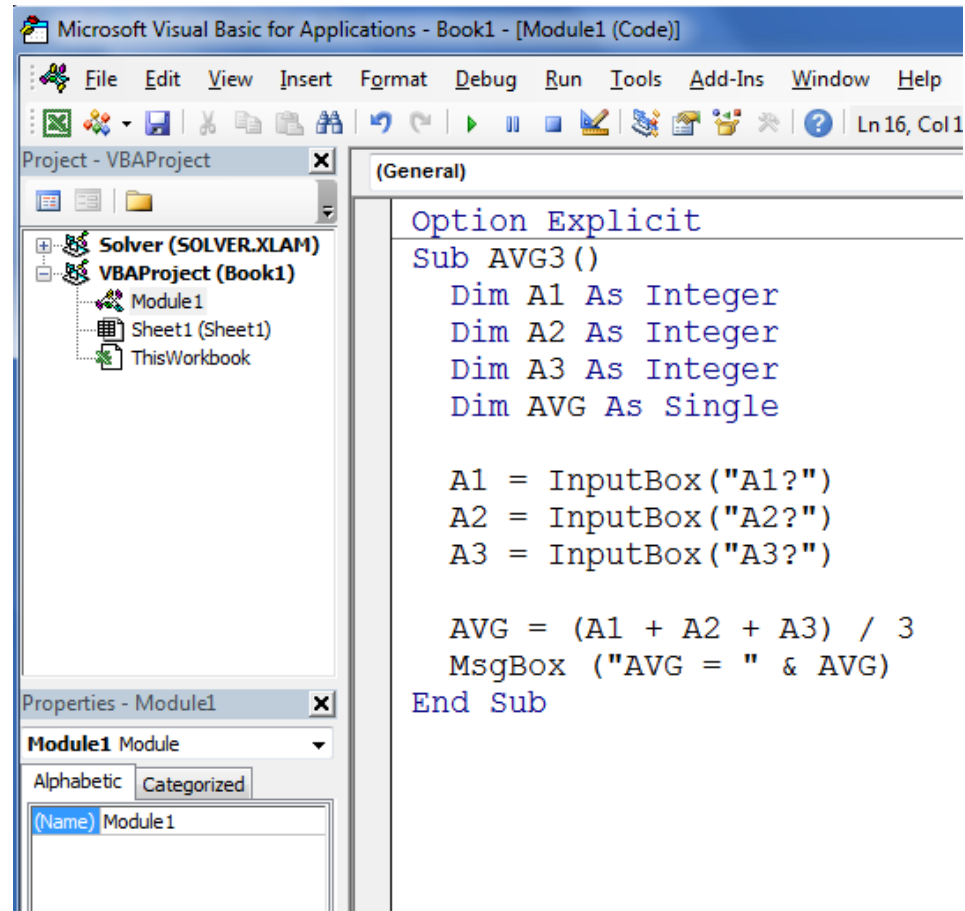
NAME: AVG3

INPUTS: A1, A2, A3

RESULTS: AVG METHOD:

- Input A1
- Input A2
- Input A3

- $AVG = (A1 + A2 + A3) / 3$
- Output AVG



The screenshot shows the Microsoft Visual Basic for Applications environment. The main window displays the code for a subprocedure named AVG3. The code includes variable declarations for A1, A2, A3, and AVG, followed by input prompts and a calculation of the average. The Project Explorer on the left shows the structure of the VBAProject (Book1), including Module1, Sheet1 (Sheet1), and ThisWorkbook. The Properties window at the bottom shows the name of the module as Module1.

```
Microsoft Visual Basic for Applications - Book1 - [Module1 (Code)]
File Edit View Insert Format Debug Run Tools Add-Ins Window Help
Ln 16, Col 1
Project - VBAProject
  Solver (SOLVER.XLAM)
  VBAProject (Book1)
    Module1
    Sheet1 (Sheet1)
    ThisWorkbook
Properties - Module1
Module1 Module
Alphabetic Categorized
(Name) Module1

Option Explicit
Sub AVG3 ()
    Dim A1 As Integer
    Dim A2 As Integer
    Dim A3 As Integer
    Dim AVG As Single

    A1 = InputBox("A1?")
    A2 = InputBox("A2?")
    A3 = InputBox("A3?")

    AVG = (A1 + A2 + A3) / 3
    MsgBox ("AVG = " & AVG)
End Sub
```

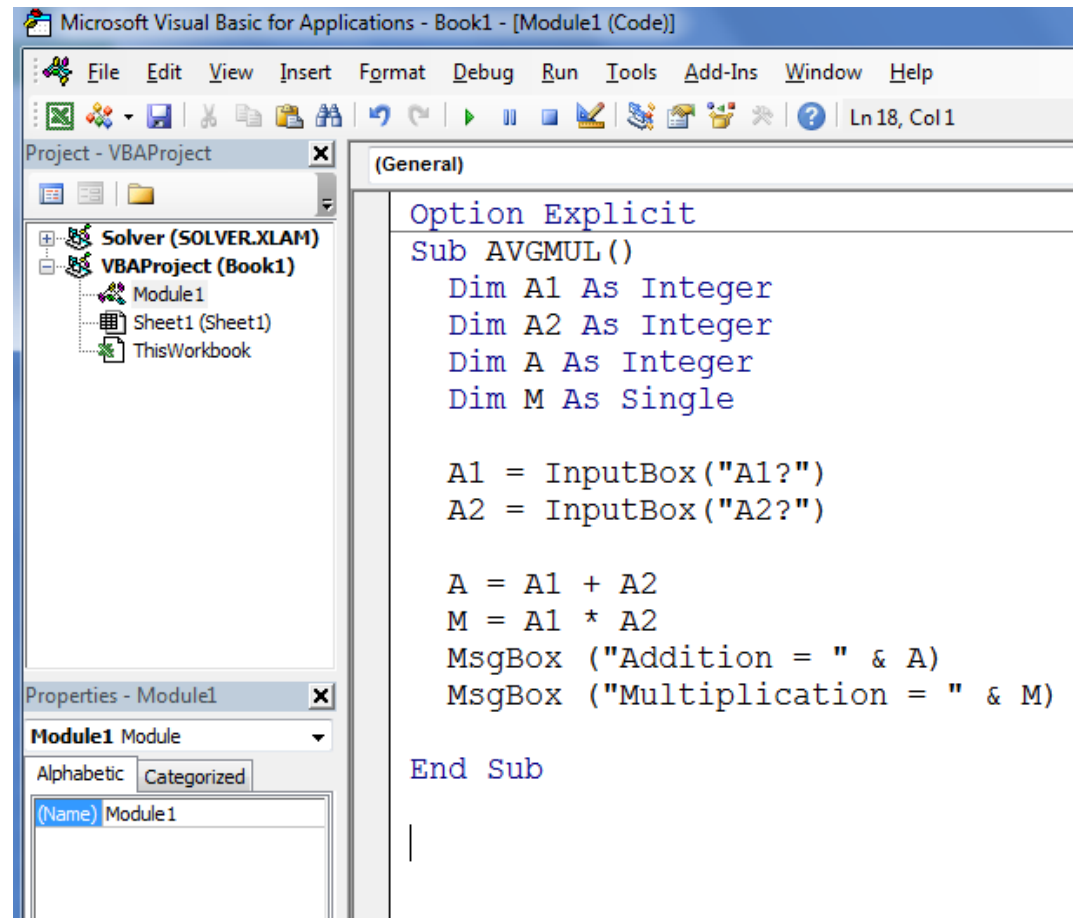
Write a VB program to find the addition and multiplication of 2 numbers.

Algorithm

- NAME: AVGMUL
- INPUTS: A1, A2
- RESULTS: A, M
- METHOD:
 - Input A1
 - Input A2

 - $A = A1 + A2$
 - $M = A1 * A2$

 - Output A
 - Output M



```
Microsoft Visual Basic for Applications - Book1 - [Module1 (Code)]
File Edit View Insert Format Debug Run Tools Add-Ins Window Help
Ln 18, Col 1
Project - VBAProject
  Solver (SOLVER.XLAM)
  VBAProject (Book1)
    Module1
    Sheet1 (Sheet1)
    ThisWorkbook
Properties - Module1
Module1 Module
Alphabetic Categorized
(Name) Module1
(General)
Option Explicit
Sub AVGMUL ()
  Dim A1 As Integer
  Dim A2 As Integer
  Dim A As Integer
  Dim M As Single

  A1 = InputBox ("A1?")
  A2 = InputBox ("A2?")

  A = A1 + A2
  M = A1 * A2
  MsgBox ("Addition = " & A)
  MsgBox ("Multiplication = " & M)

End Sub
```