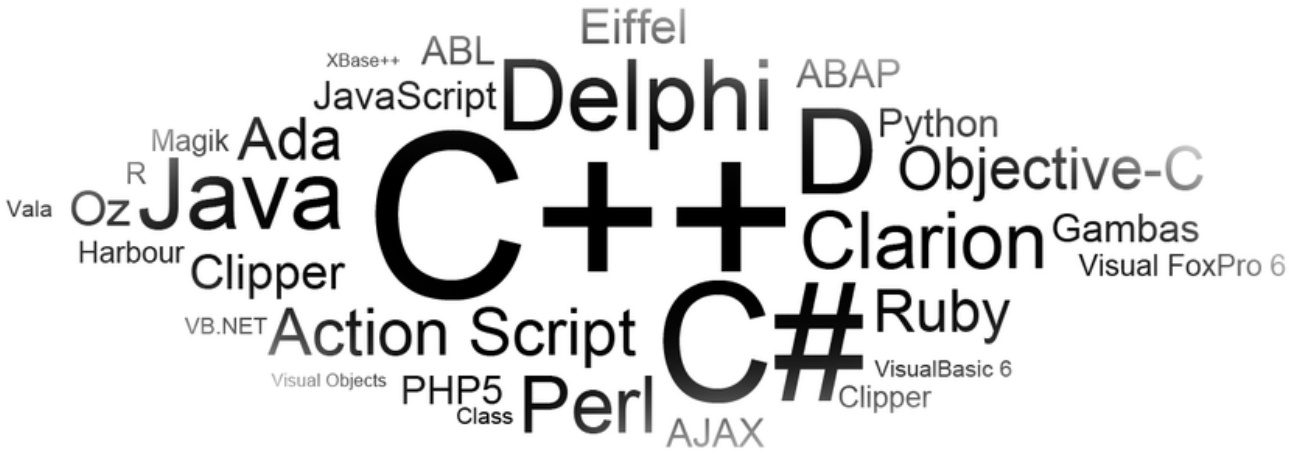


Day 1 – 19 May 2015 (3hrs)

VBA Programming for MIF

Siraprapa Watakit
Technical Instructor,
BSc(Computer),MSc(Finance,MIF#10), PhD Student(Finance)

Course Objective and Outline

- **After attending this course, participant will be able to**
  - Implement VBA programs
  - Process and clean raw financial data with VBA
- **Outline**

Date	Details
Day 1 (3hrs)	**Module 1** • Introduction to Excel/Macro Recording **Module 2 : Fundamental VBA Programming Concept** • VBA Integrated Development Environment(IDE) • Data type and Variable • Function and Subroutine • Debugging programs
Day 2 (3hrs)	**Module 2 : (Continue)** • Structural Programming • Working with Excel Objects • Event Handling in VBA

Day 3 (3hrs)	**Module 3 : Data Processing & Cleaning for Financial Research**
	• Aggregate data from multiple files
	• Data cleaning, eliminating un-wanted data
	• Data transformation(convert time-series data to panel data)
	• Data processing/cleaning tips and techniques for ...
	  ○ ***Datastream***
	  ○ ***Bloomberg***
	  ○ ***Thomson Reuters Eikon Excel***

Day 1 – Agenda

Module 1

- **Introduction to Macro**
  - Macro Basics
  - Workshops
    - Record and Call **simple macros**
    - Create **simple buttons and controls**

Module 2 – Part 1

- **Fundamental VBA Concept**
  - Fundamental of Programming
  - Workshops
    - Create **simple functions**
    - Create **simple subroutines**

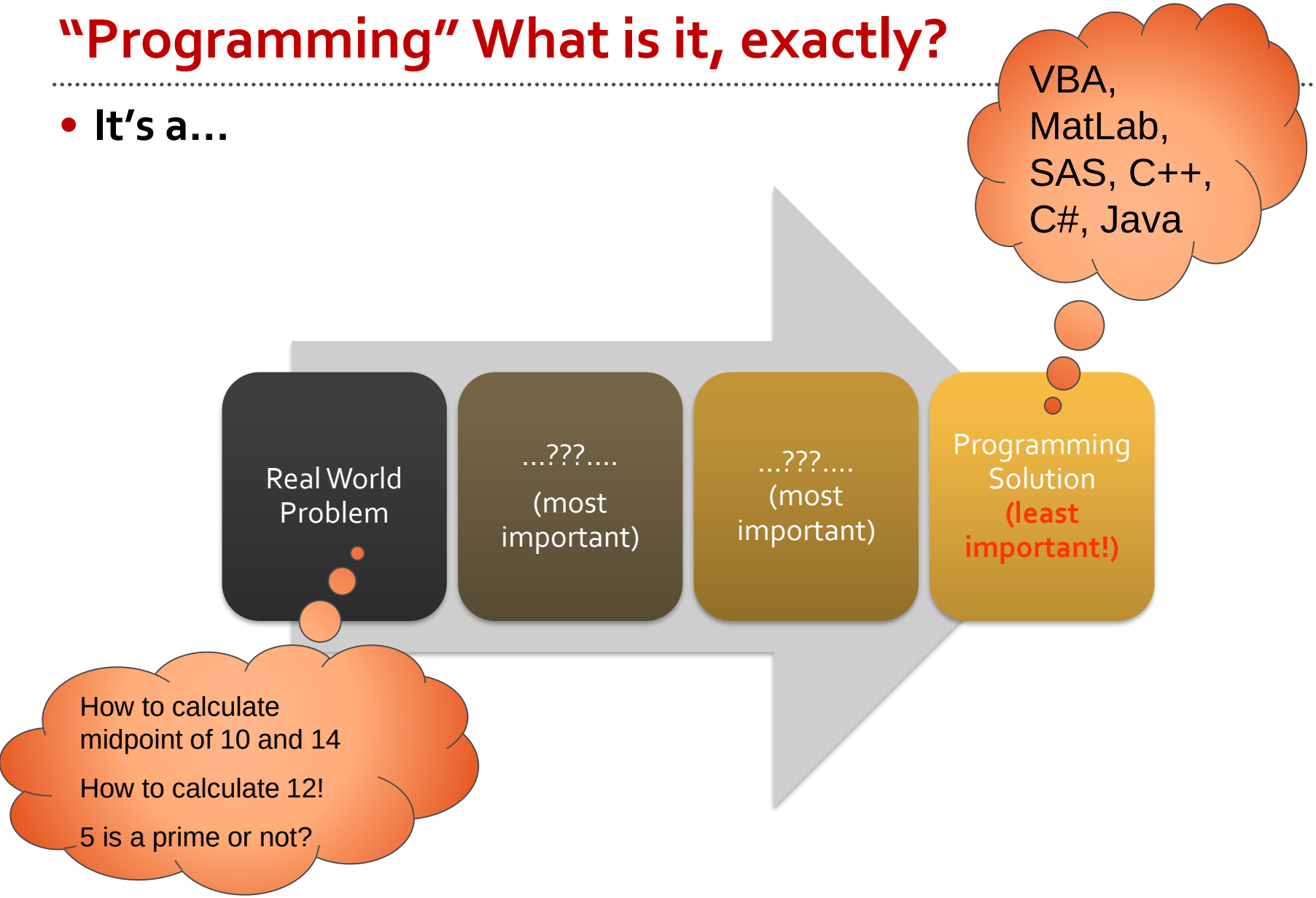
Module 1
Introduction to Macro

Objective and Agenda

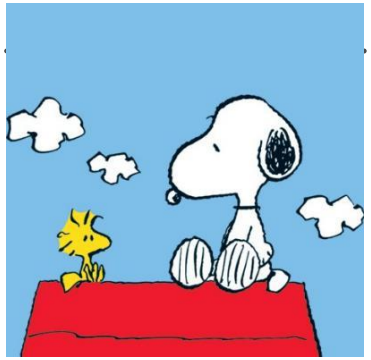
- **Objective**
  - Able to record/delete/edit/call Macros
- **Agenda**
  - Record Macros
  - View/Edit/Delete Macros
  - Export/Import Macros
  - Call Macros
  - Create simple form Buttons and MsgBox

"Programming" What is it, exactly?

- **It's a…**



What is a programmer?



- **A programmer is just like a..**
  - *we see things in **black** and **white***

- **A programmer is quite simple minded, we do one of the following things**
  - go straight on
  - given a condition, go **left** or go **right**
  - go **round** and **round,** until some condition is true

- **A programmer love to imagine, make plan, and revisits**

Pseudo Code (most important)

Flow Chart (most important)

Actually do the code! **(least important!)**

Why is it the least important?

VBA, MatLab, SAS, C++, C#, Java

Programming Solution **(least important!)**

- **A programming language is *just* a tool**
  - How many language can you say

    "I have no money"

    - Thai, English, Chinese, Japanese?
  - What is the common/different?
- **Every PL shares the same STRUCTURE but governs by different SYNTAX.**
  - Unless you want to use a special capability of a PL, any PLs can do just about the same jobs

MIF – PL Training

- ..

STATA

- **Most updated Statistical Package**
- EGARCH,MGARCH,STAR and etc..
- **a scripting language,** poor for big data

SAS

- **Expert in the Field of Transaction Processing Data(>1M record)**
- **a compiled language**
- Lots of PROC ready for utilization

VBA

- **a complied language**
- Plenty of Excel Object for utilization, **very good for data cleaning**
- **Easiest to learn**

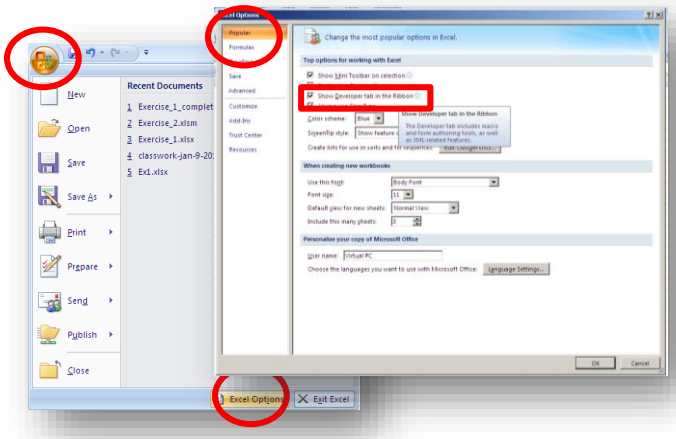
Matlab

- **Matrix manipulation**
- **Optimization**
- a scripting language, poor for big data

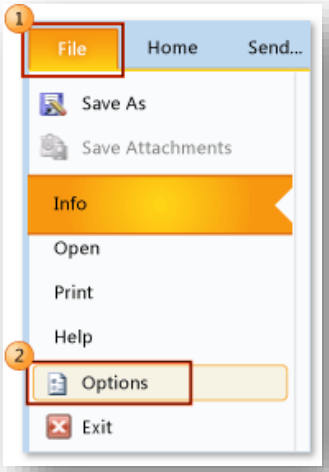
Important Notes

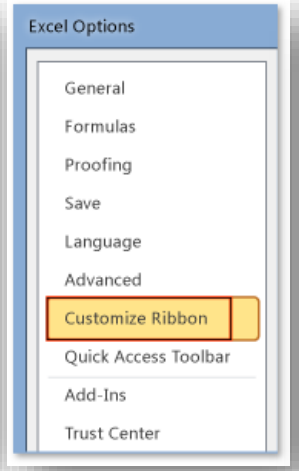
- **Always save as "XLSM" – Excel Macro-Enabled Workbook**
- **Enable "Developer Tab" on the Ribbon**

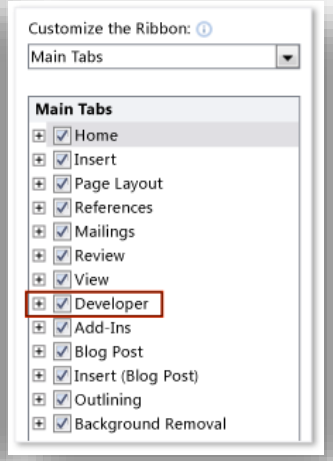
Excel2007



Excel 2010/2013



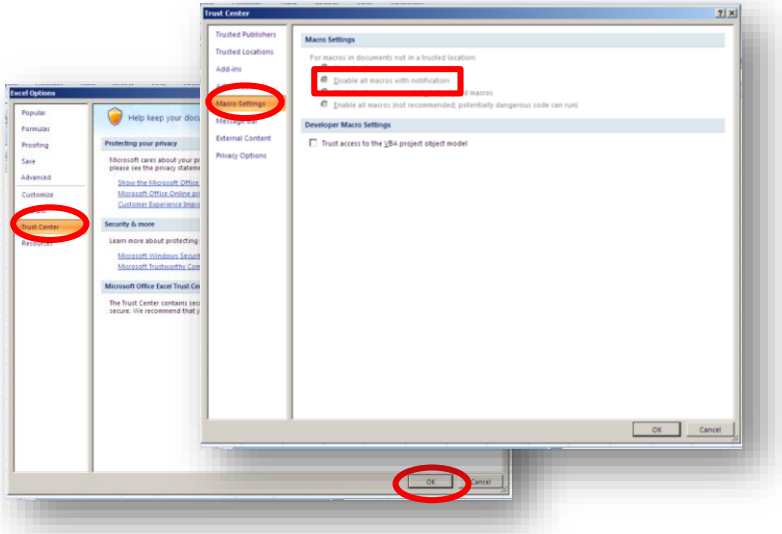




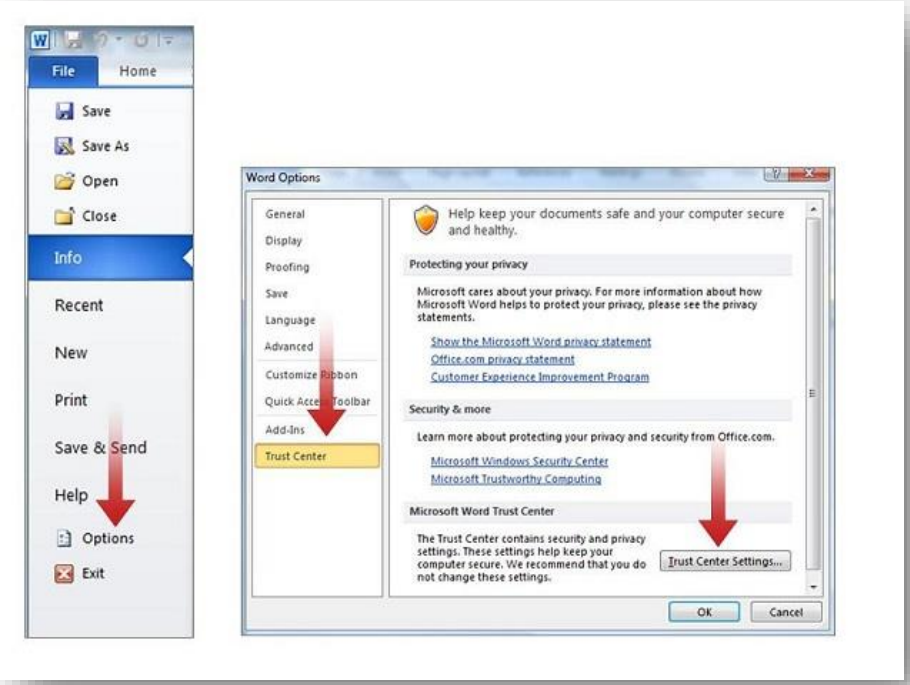
Important Notes

- **Enabled Macros – Trust Center**

Excel2007

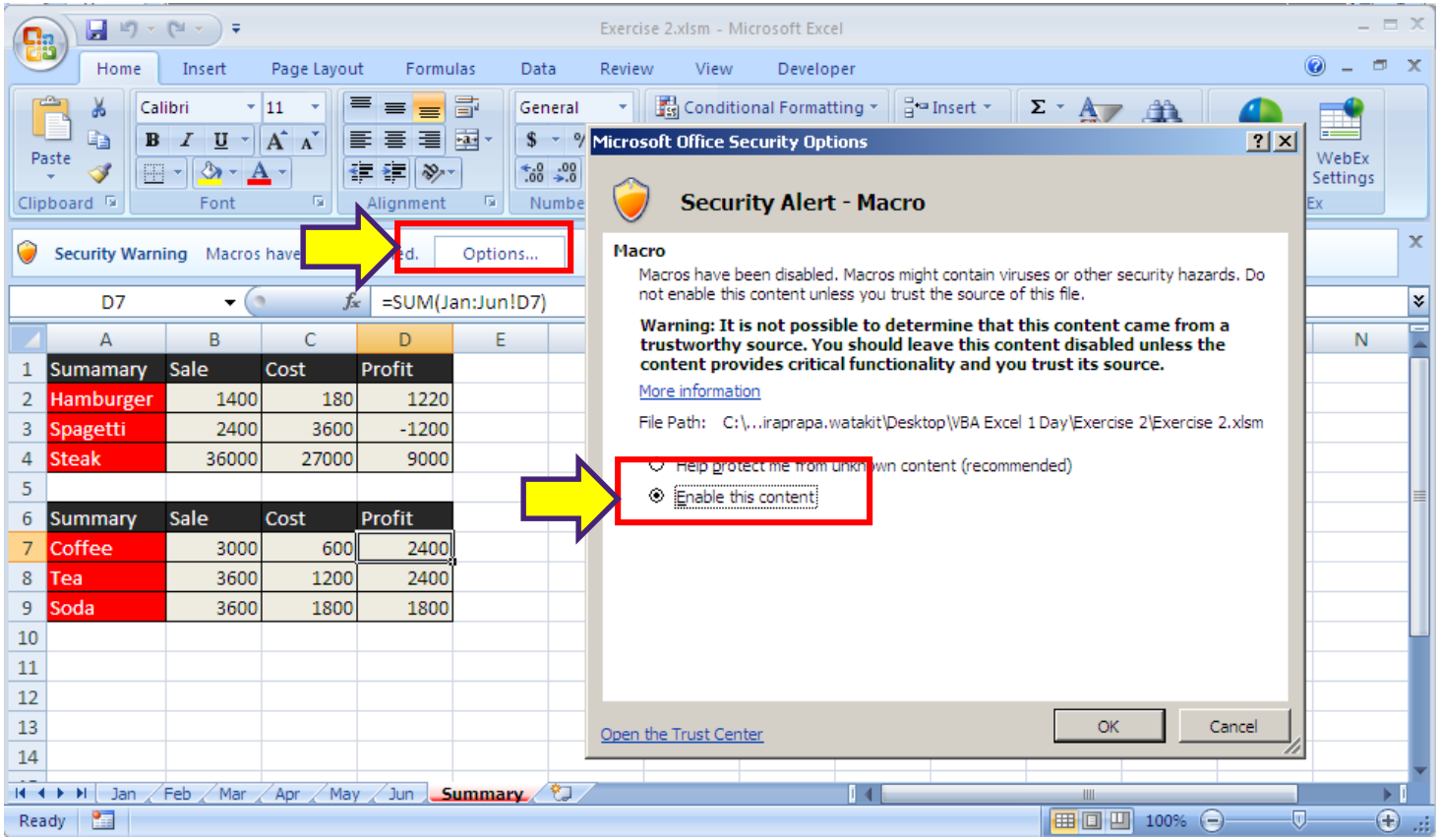


Excel 2010/2013

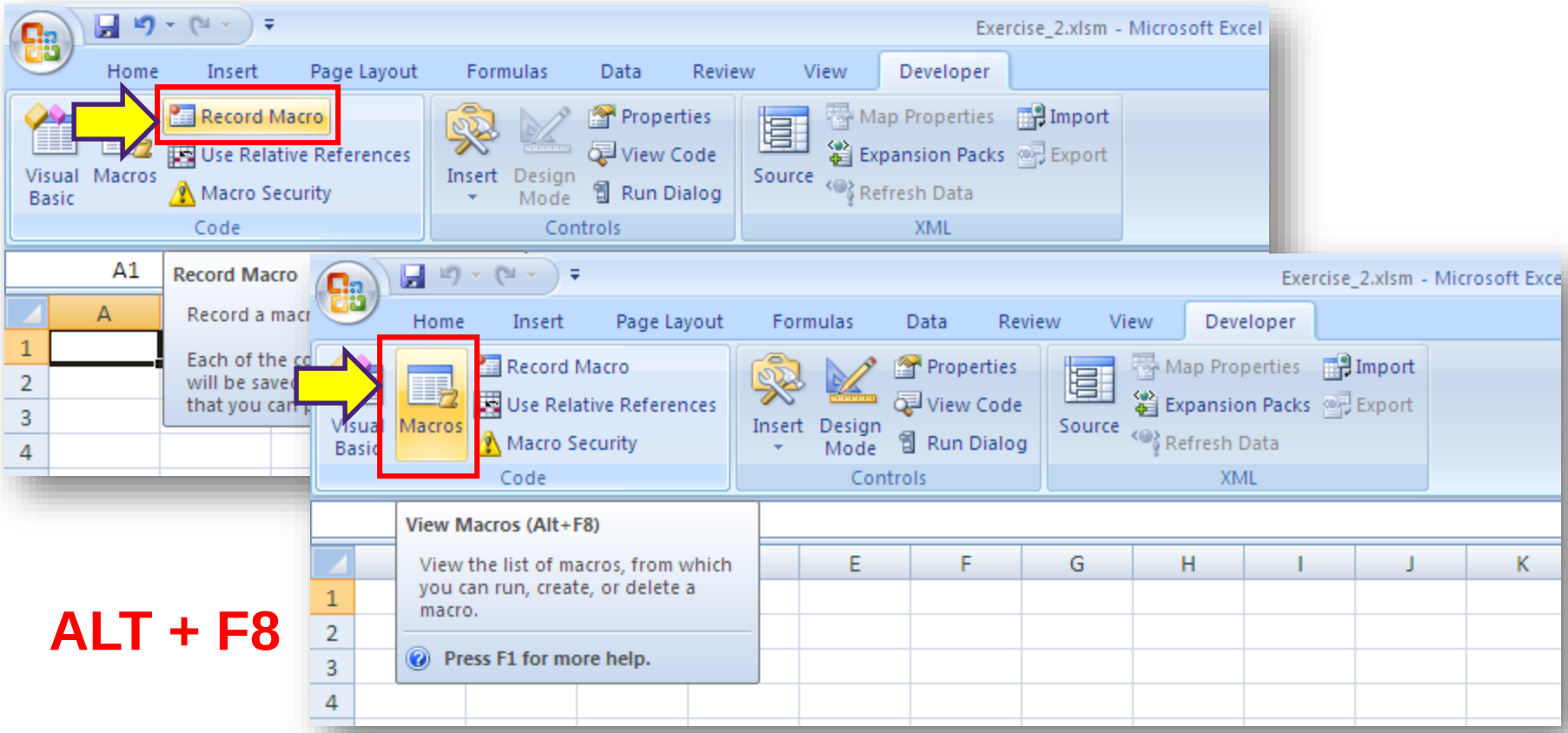


Important Notes

- **Upon opening a macro enabled file…**



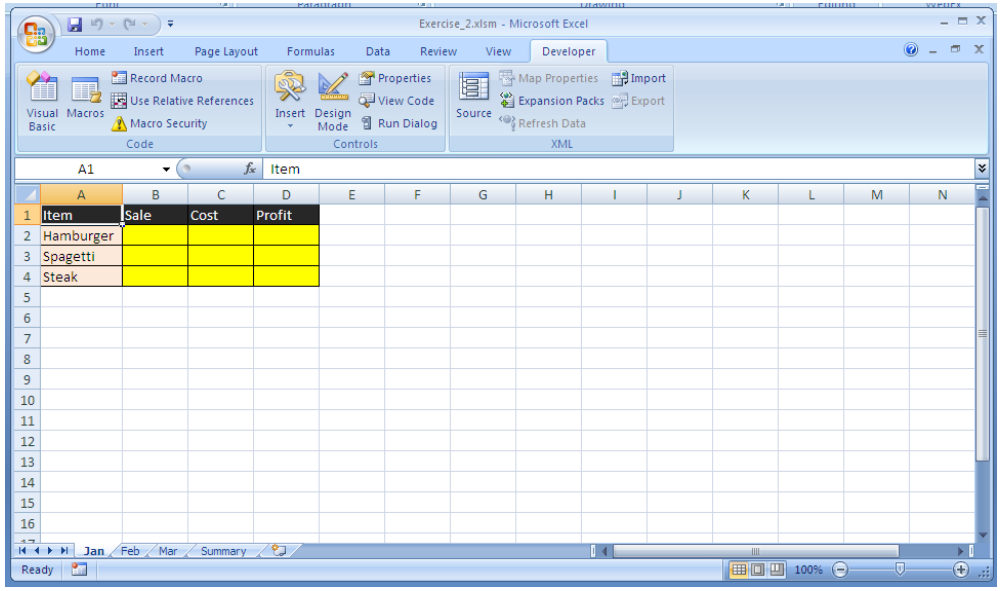
To Record/View Macros



ALT + F8

Exercise – MyTable_1

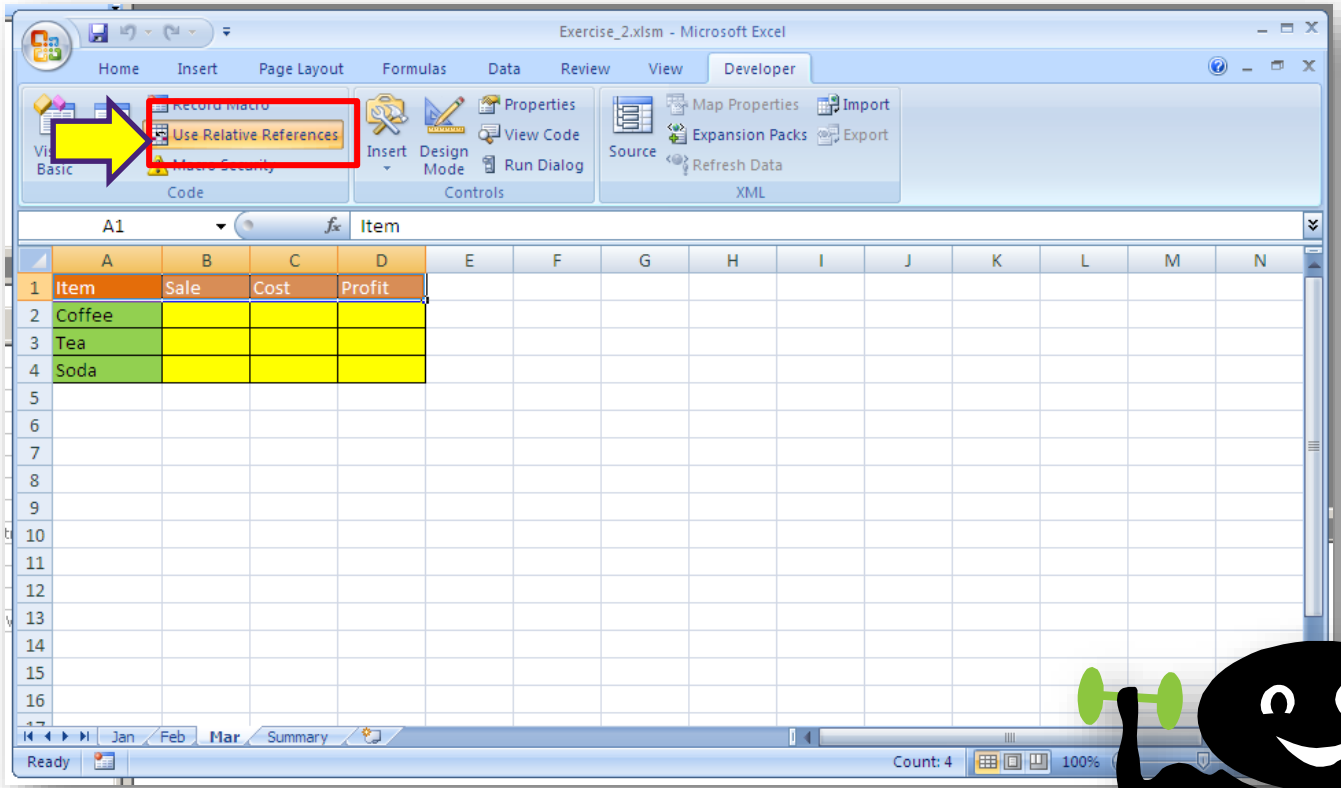
- **Create a Macro that will create a table like this..**



- **Follow your instructor and take note your steps**

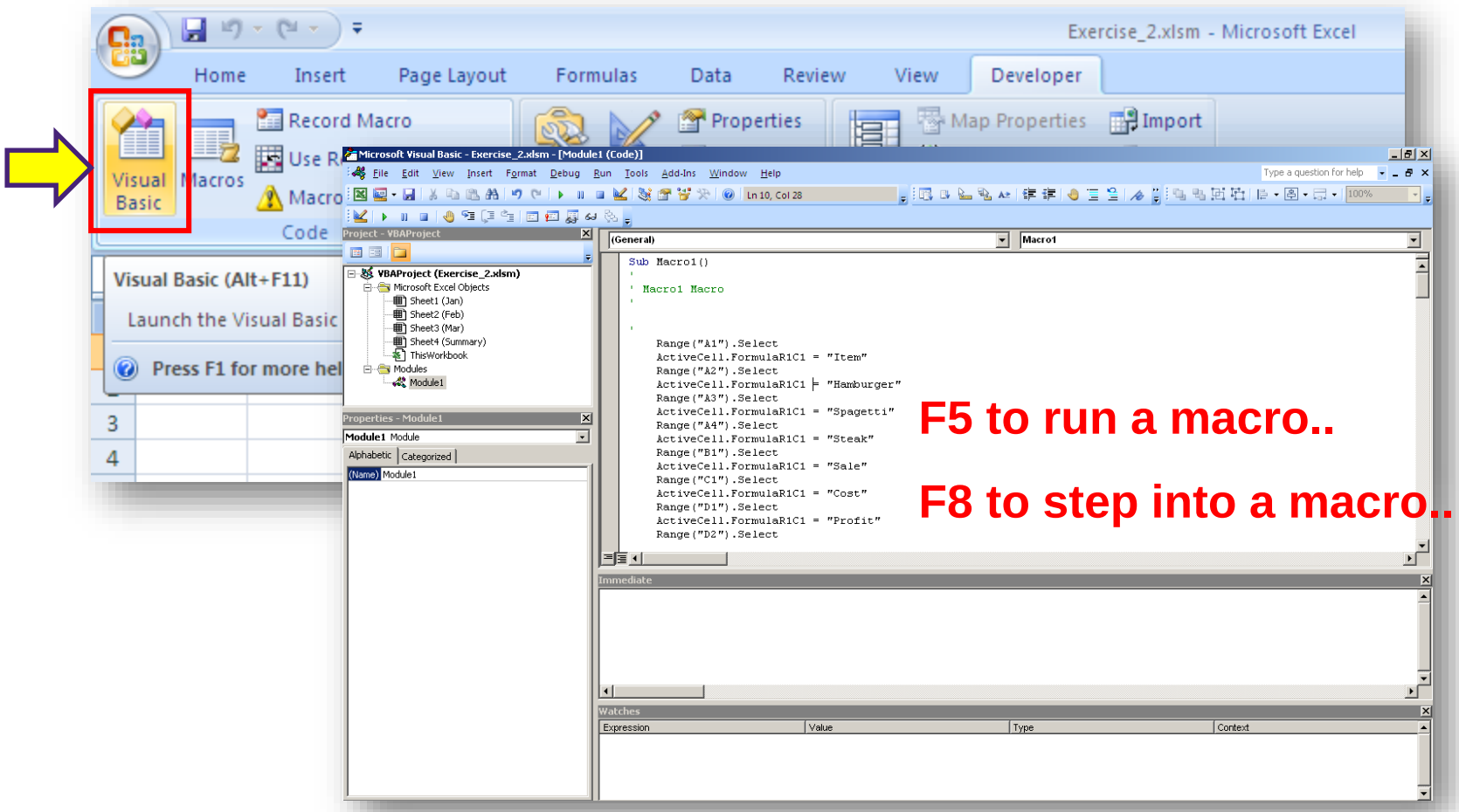
Exercise – MyTable_2

- Do the same but with Relative References enabled <u>BEFORE</u> recording



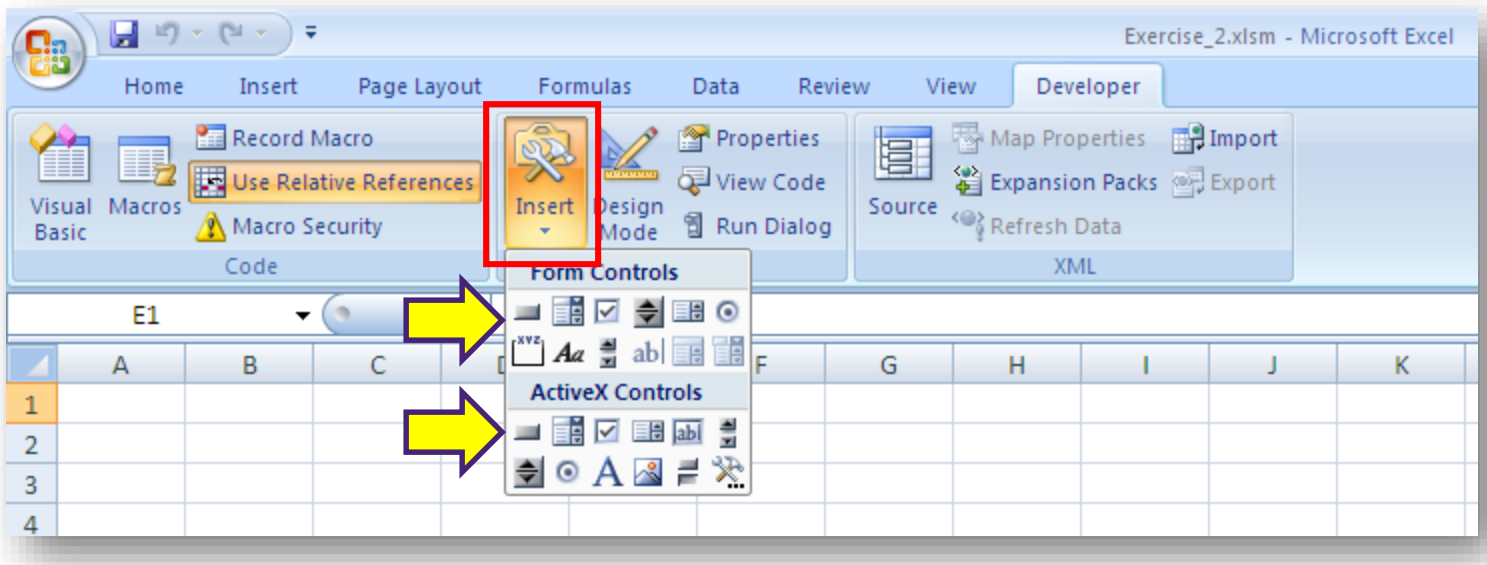
To goto VBA Editor

- **ALT + F11**



Insert Controls

- **Goto "Developer Tab" and then click insert button**
  - Button (Form Control)
  - Command Button (ActiveX Button)



Exercise - “Hello World”

a compiled language

- **Create Macro as follow**

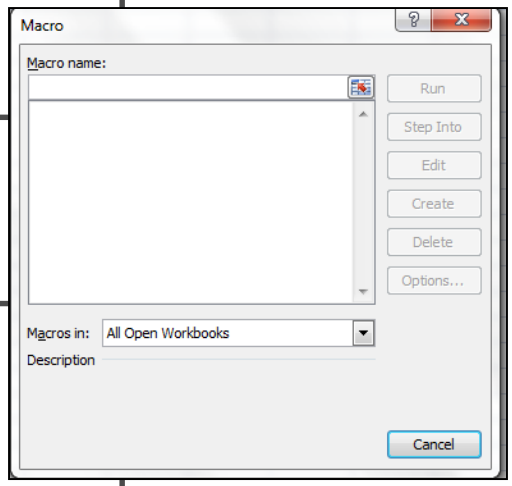
```
Sub MyHello()
    MsgBox "Hello World!!"
End Sub
```

- **Bind it with a button**
  - Button Form Control >> Right-Clicked + “Assign Macro”

How to Hide Your Macro??

- Hide a single macro by using keyword Private

```
Private Sub MyHello()
    MsgBox "Hello World!!"
End Sub
```



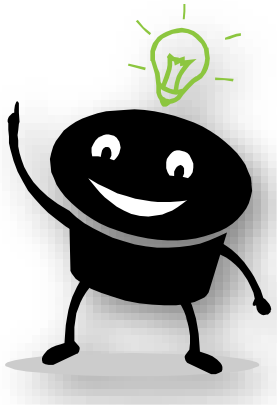
- Option Private Module

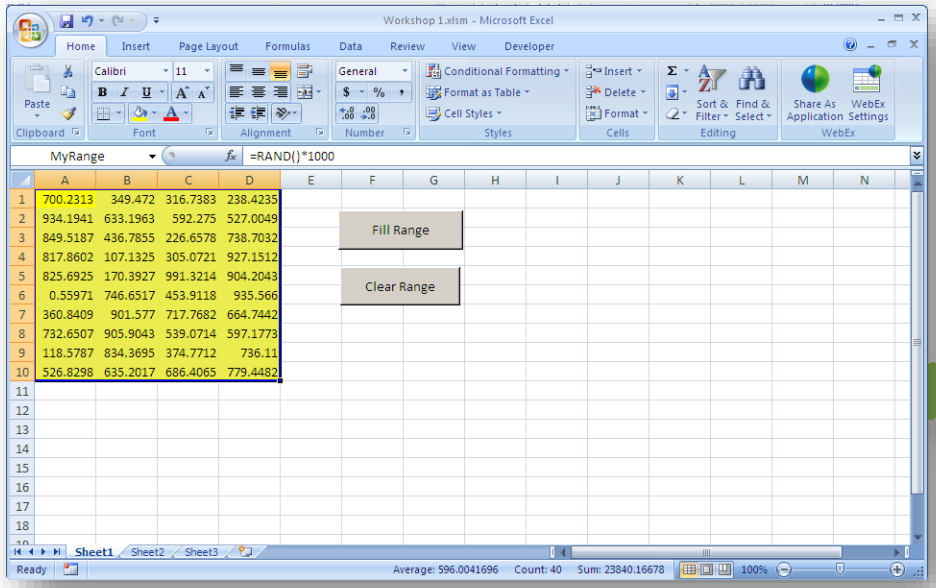
```
Option Private Module
Sub MyHello()
    MsgBox "Hello World!!"
End Sub
```

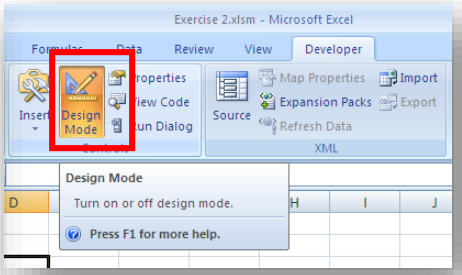
WARNING: Once done, your macros will be "invisible" from Macro Menu, and you <u>will not be able</u> to assign it to <u>Form Control</u> buttons or <u>any drawing/picture objects</u>

Workshop

- Create a Range >> MyRange
- Create a Macro "FillRange" >> Fill it with dummy data
- Create a Macro "DelRange" >>Clear all data in the range
- Assign macros onto buttons(Form Control)







Module 2

Fundamental VBA Programming Concept

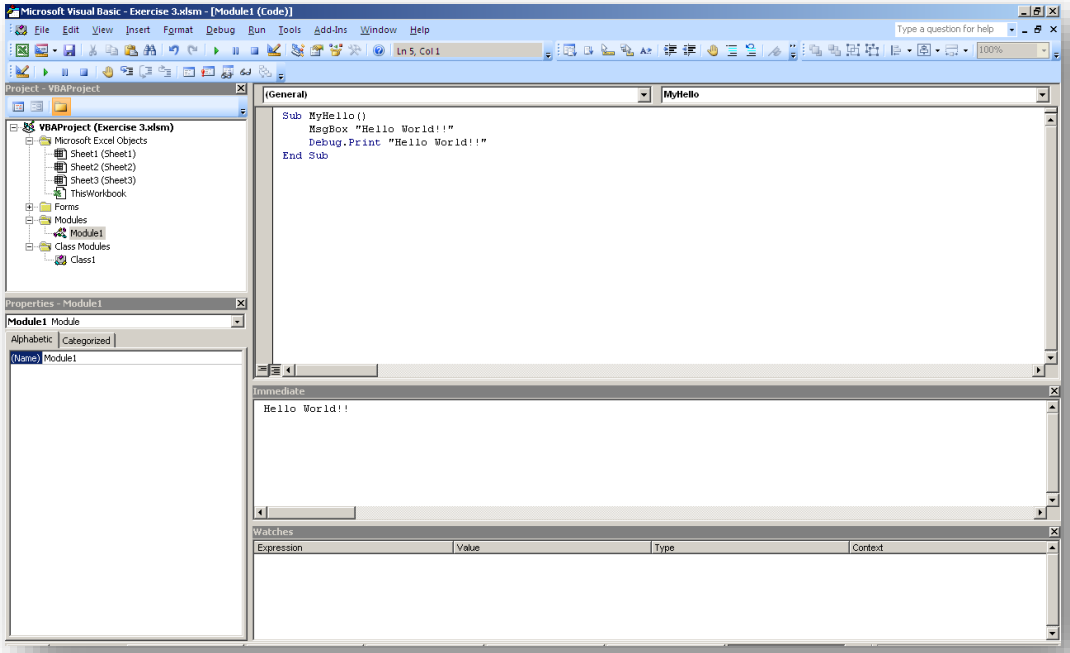
Part 1

Objective and Agenda

- **Objective**
  - Able to identify the basic programming concept/constraint of VBA
  - Able to implement basic VBA programming
  - **Able to complete all workshops**
- **Agenda**
  - Introduction to VBA Projects
  - Data type and Variable
  - Function and Subroutine
  - Structure Programming
  - Debug program using watch windows and breakpoints

VBA Projects

- **VBAProjects (ProjectName)**
  - Microsoft Excel Objects
  - Forms
  - **Modules**
  - Classes
- **Properties Window**
- **Immediate Window**
- **Watch Window**



It is recommended that you put your VBA code Sub and Function in one of the "Module".

You many rename it to anything that make sense to you.

There are 2 distinguish type of VBA programming

- Subroutine

```
Sub Greeting()
    Dim myname as string
    myname=InputBox("Input your name")
    MsgBox "Hello World" & myname
End Sub
```

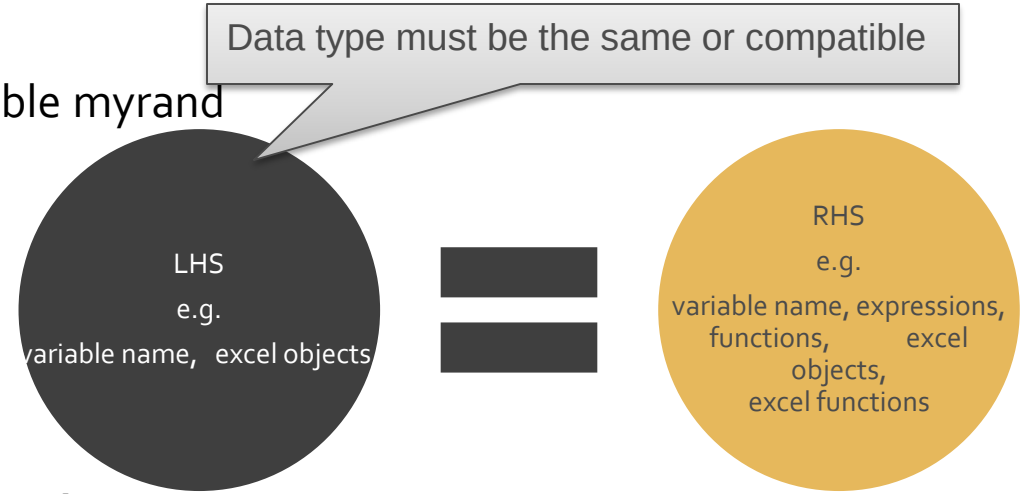
- Function

```
Function MyCal(num1,num2)
    MyCal=num1+num2
End Function
```

- The different is that Function RETURNS VALUE, but Subroutine does not..

The "=" Operator means?

- An ***ASSIGNMENT*** operator
  - Assign a string **Adam Smith** to a variable called **myname**
    - **`myname="Adam Smith"`**
  - Assign a value from a variable myname to myname_2
    - **`myname_2=myname`**
  - Assign a random number to a variable myrand
    - **`myrand=Rnd()`**
  - Assign a value 1000 to cell A1
    - **`Range("A1")=1000`**



- A ***LOGICAL*** operator
  - `If (myname="Adam Apple") then..`
  - `If (myrand>0.03) then…`
  - `If myrand then…`

In any programming languages 0 is False, other than 0 is always True, including negative number or a character

There are 2 distinguish type of VBA Data Type

- **Primitive Type**
  - **Numeric:** Integer, Long, Single, Double, Byte,Boolean
  - **Alphabetic:** String, Char
  - **Alphanumeric**: Variant

These type come with lots of built-in function and subroutine which can be utilized in your source code

- **Objects Type**
  - **Excel Objects:** Workbook, Worksheets, and etc.
  - **Class Objects:** Defined/Construct by Class Module
- **Example of variable declaration**
  - Dim myname as **String**
  - Dim aRange as **Range**

Note that variable name must be starting with _ or a character, and must not be a reserved word or a sequence of number only or contain special character such as #@!

- Correct: a123, _123 ,myDate, myString, myInteger
- Incorrect: 446, 123a, String, Date, Integer, ag@io

Data Type: Primitive

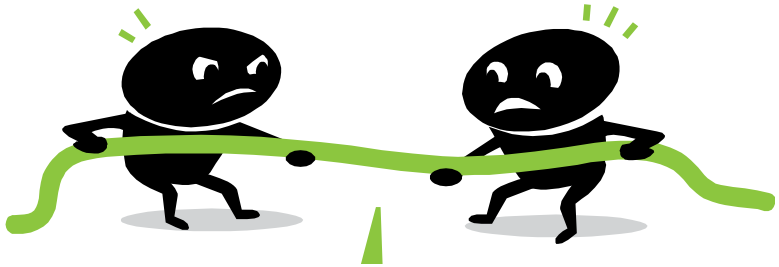
Data Type	Size	Range of Value
Byte	1	(0.....255)
Boolean	2	(True, False)
Integer(%)	2	(-32,768.....32,767)
Long(&)	4	(-2,147,483,648.....2,147,483,648)
Single	4	- Real (-3.402823E38....-1.401298E-45) +Real (+1.401298E-45.....3.402823E38)
Double	8	- Real (-1.79769313486231E308...-4.94065645841247E-324) +Real (+4.94065645841247E-324...+1.79769313486231E308)
Variant(Number)	16	Same as Double
Variant(Char)	22	0....2^{31} Characters

F1 - Data Type Summary

Implement VBA /wo properly declaring variable can be very dangerous..

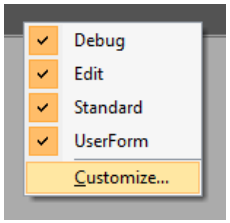
```
Function CallNums(a,b)
   c=6
   d=7
   CallNums=a*b+c*d*e
End Function
```

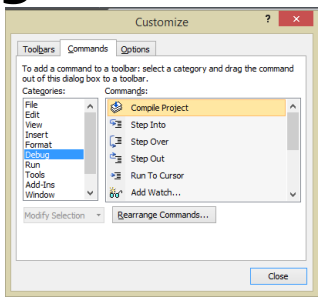
- **try this at immediate window, what's wrong?**
  - ?CallNums(10,20)



VBA Good Practices..

- **There are 3 types of error in all kind of programming**
  - Compile time errors
  - Runtime errors
  - **Logical errors**





- **There are several ways to prevent such incidents**
  - Always **declare (dim)** variable name and type appropriately
  - Always **document(comment)** your code
  - Keep your source code clean, short and **neat**
  - Give a **meaningful** sub/function/variable name
  - Learn to use **breakpoint** and watch windows
  - Learn to use excel add-ins tools: **Smart Indenter**

Data Type: Primitive

- Use watch windows to debug each variable carefully

```
Sub MyVarTest()
    Dim MyInt1, MyInt2 As Integer
    Dim MyVar

    MyInt2=10/3
    MsgBox "10 divide by 3 is " & MyInt2

End Sub
```

What is the data type of MyInt1,MyInt2,MyVar?

What is the display value is MsgBox? Why?

What will happened if you add this line→ MsgBox "MyInt3=" & MyInt3

Data Type: Primitive
Working with Array

- Assign values to array

```
Sub DeclareArray()
    Dim MyArray1D(2) As Integer
    Dim MyArray2D(1 to 2,1 to 2) As String
    MyArray1D(0)=10
    MyArray1D(1)=20
    MyArray1D(2)=30
    MyArray2D(1,1)="Maggie"
    MyArray2D(1,2)="Alice"
    MyArray2D(2,1)="Mark"
    MyArray2D(2,2)="Joe"
End Sub
```

Data Type: Primitive
Working with Array and FOR Loop

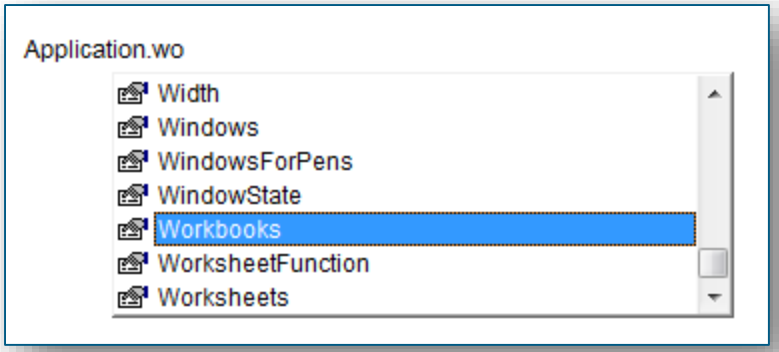
- **Now, how about array with 100 or more values?**

```
Sub DeclareArray_Loop()
   Dim i as Integer
   Dim j as Integer
   Dim MyArray1D(1 to 10) As Integer
   Dim MyArray2D(1 to 10, 1 to 10) As Integer

    For i=1 to 10
        MyArray1D(i)=i
    Next

    For i=1 to 10
       For j=1 to 10
           MyArray2D(i,j)=i*j
       Next
    Next
End Sub
```

Exercise, paste “MyArray2D” into a Worksheet

Programming Notes

- It is most recommended to use **Option Explicit**
- If you want all array to start with index 1, use **Option Base 1**
- Be very careful with **Data Type**
- VBA programming is **Case-Insensitive**
- Ctrl+Spacebar for **Popup Box**



Some useful functions/operators

Function/Operators	Description
=	Assignment e.g. a=100 Equal e.g. if (a=b) then..
< <= >= <> Is like	less than, less or equal, greater or equal, greater than, not equal, **is, like** e.g. If (a is "Yes") then.. e.g. If(a Like "[Y,y][E,e][S,s]") then..
+ - * / \ Mod ^	Add, Subtract, Multiply, DivReal, DivInt, Mod, Power
Ubound(array) Lbound(array)	Find the upper /lower bound of an array e.g Dim a(1 to 2, **2** to 3, 1 to **4**) As integer Ubound(a,3) is equal to **4** Lbound(a,2) is equal to **2**
UCase(str), LCase(str)	Turn str to upper/lower case
CInt(x), CLong(x)	Convert x to integer/long
Len(str)	Return length of string input
Split(str)	Split string into array of variant

Exercise

- **Give an example of correct and incorrect variable name**

- **Write a program that does the following bits..**
  1. define a variable name **MyString**
  2. assign **Maggie Q** into the variable
  3. display it on the **message box**
  4. display it at the **immediate window**

Full signature of Subroutine

- **Subroutine**

```
Sub MySub1()
End Sub
```

```
Sub MySub2(a)
End Sub
```

```
Sub MySub3(ByRef a As Integer, _
           ByRef b as Integer, _
           Optional c As Integer=999)

      Debug.Print a;b;c

End Sub
```

All sub/fnc name must be unique

ByRef is the default identifier, alternatively you can also use ByVal

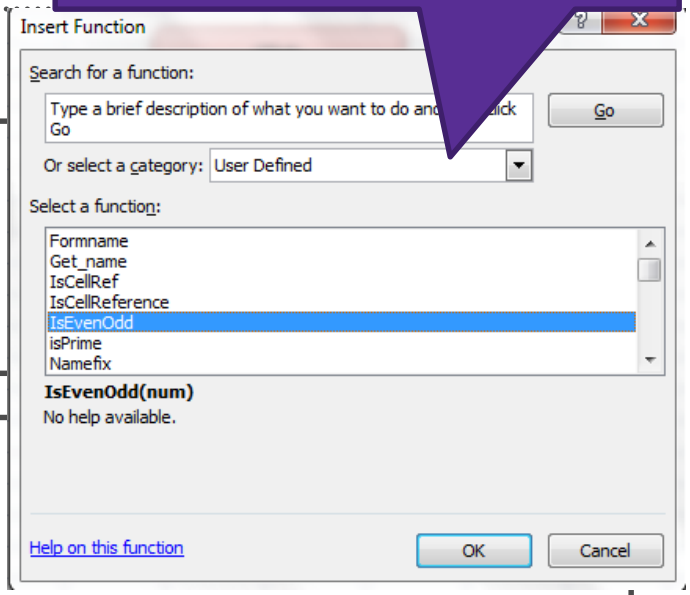
Full signature of Function

Note that your functions will be under UDF categories

- Function

```
Function MyFnc1()
      MyFnc1 = <expression>
End Function
```

```
Function MyFnc2(a)
      MyFnc2 = <expression>
End Function
```




```
Function MyFnc3(ByRef a As Integer, ByRef b as Integer, _
Optional c As Integer=999) As Double

      MyFnc3 = <expression>
End Function
```

Calling Subroutine is easy!

```
Sub MySub_Hello()
     MsgBox "Hello"
End Sub
```

```
Sub MySub_GoodBye()
     MsgBox "GoodBye"
End Sub
```

```
Sub MySub_Hello_GoodBye()
     MySub_Hello
     MySub_GoodBye
End Sub
```

Calling Function is just a bit different!

```
Function MyFnc_1 ()
     MyFnc_1="Michael"
End Function
```

```
Function MyFnc_2()
     MyFnc_2="Jackson"
End Function
```

```
Sub MyFnc_3()
 Dim MyVar as String
 MyVar=MyFnc_1 & " " & MyFnc_2
End Sub
```

Called function must be on the right

The different between– ByRef and ByVal

- Example of passing by reference and by value

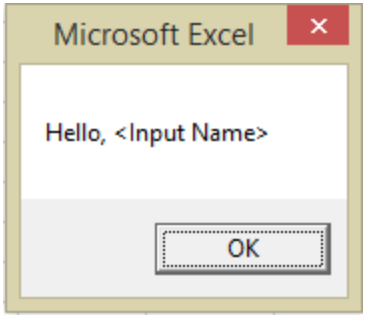
```
Sub CalledProcedure(ByRef X As Long, ByVal Y As Long)
    X = 321
    Y = 654
End Sub

Sub CallingProcedure()
    Dim A As Long
    Dim B As Long
    A = 123
    B = 456
    Debug.Print "BEFORE CALL = A: " & A, "B: " & B
    CalledProcedure A, B
    Debug.Print "AFTER CALL =  A: " & A, "B: " & B
End Sub
```

Test your knowledge

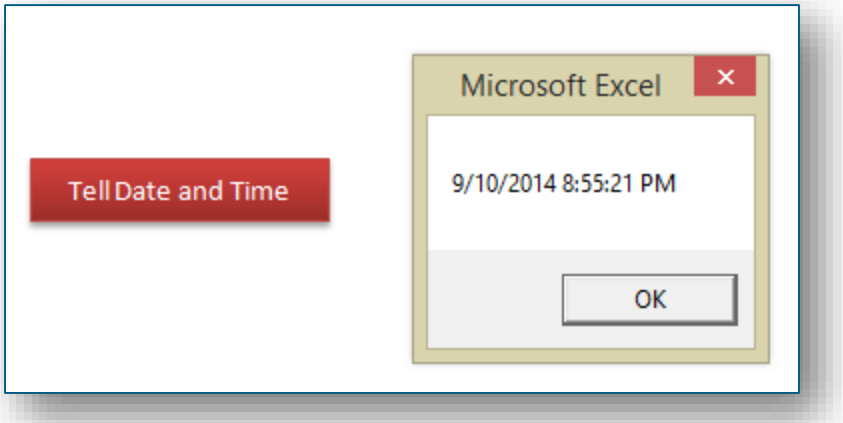
- [T/F] Dim without designated datatype will result in "INTEGER" by default
- [T/F] variable name is case-insensitive in VBA
- What is the difference between subroutine and function?

Homework – Day 1

- **Create a SUBROUTINE that ask for your name, then display a message box like this**



- **Create a button, when you click this button, it will tell you the current Date and Time**
  - Hint: Type Date, Time at immediate window



Homework – Day 1

- **Create a FUNCTION that accepts 3 parameters: *discount*, *price* and *quantity*. Then calculate and return *netprice* as**
  - netprice= (1-discount)*price*quantity
- **Create a FUNCTION that accepts 1 parameter: *r*. Then calculate and return the area of a circle as**
  - area=πr^2

Homework – Day 1

- **Create Black-Scholes CALL/PUT valuation FUNCTION.**
  - Input parameter: S, X, rf, T, sigma
  - Hint N(d1) = NORMSDIST(d1)
  - E.g for S=100,X=95,rf=7%,T=3/12,vol=20%
    - Call=8.0559, Put=1.40792

$$d_1 = \frac{\ln\left[\frac{S_t}{X}\right] + \left(r + \frac{1}{2}\sigma^2\right)\tau}{\sigma\sqrt{\tau}}$$

$$d_2 = d_1 - \sigma\sqrt{\tau}$$

$$C_{B-S} = SN(d_1) - Xe^{-r\tau}N(d_2)$$

$$P_{B\text{-}S} = Xe^{-r\tau} - S + C_{B\text{-}S}.$$

Preparation for Day 2

- **Write down the Pseudo Code for the following problem**
  - How to calculate 10!
  - How to calculate the summation of number between 2 and 15
  - How to determine a prime number, for example 5 is a prime number because 5 can only be divided by 1 and 5
  - Guess a magic number game. A random number between 1-99 is selected at the beginning, player will have to guess the magic number until the game ends. The ends condition of the game are one of the following
    - Player guess correctly
    - Player give up by submitting **a negative number**
- **Think! programmatically.**
  - How to accept parameters? Sub or Function?