EXCEL WORKSHOP III: INTRODUCTION TO VBA IN EXCEL

Bruin Actuarial Society

Note: Slides will be posted on our website after this meeting

BAS.00
BRUIN ACTUARIAL SOCIETY

BACKGROUND

- You are an actuary for Bruin Health Insurance, LCC, a Los Angeles based company insuring personal health claims in Southern California
- Prior to your team's analysis of coverage and reserves, you notice that the policy data is not formatted in the desired manner
- Each membership is only listed once, regardless of how many times it has been renewed
- **Problem:** You need to be able to identify the specific 1-year term you're looking at when conducting the analyses

BAS.00
BRUIN ACTUARIAL SOCIETY

BACKGROUND

Task: Update the way the company's data is stored

- If a membership has been renewed twice (i.e. it was in force for 3 terms), there should be 3 rows, one for each term
- The listed start date should be the start date of each term (i.e. a new row is created upon renewal)

BAS.00
BRUIN ACTUARIAL SOCIETY

OBJECTIVE

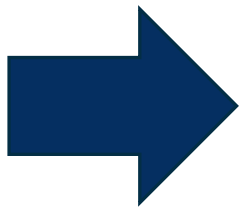
1) Write a VBA macro to accomplish this task with the short excerpt of data

2) Then, once you've ensured the macro works correctly, run it on the full set of membership data, splitting the 13,041 membership plans into 39,215 rows

BAS.00
BRUIN ACTUARIAL SOCIETY

OUR GOAL

MemberID	MembershipStart	NumberOfTerms	AvgAnnualClaims	MedicalPlan
P100458	3/17/14	3	$964.82	HMO
P105944	9/15/16	3	$692.34	HMO
P108055	9/21/17	5	$516.26	PPO
P108257	10/24/17	2	$696.74	HMO
P108429	11/20/17	1	$533.93	POS
P108776	1/15/18	2	$276.34	PPO
P108873	2/2/18	2	$786.09	HMO
P111792	6/14/19	2	$316.17	POS



MemberID	MembershipStart	AvgAnnualClaims	MedicalPlan
P100458	3/17/14	$964.82	HMO
P100458	3/17/15	$964.82	HMO
P100458	3/17/16	$964.82	HMO
P105944	9/15/16	$692.34	HMO
P105944	9/15/17	$692.34	HMO
P105944	9/15/18	$692.34	HMO
P108055	9/21/17	$516.26	PPO
P108055	9/21/18	$516.26	PPO
P108055	9/21/19	$516.26	PPO
P108055	9/21/20	$516.26	PPO
P108055	9/21/21	$516.26	PPO
P108257	10/24/17	$696.74	HMO
P108257	10/24/18	$696.74	HMO
P108429	11/20/17	$533.93	POS
P108776	1/15/18	$276.34	PPO
P108776	1/15/19	$276.34	PPO
P108873	2/2/18	$786.09	HMO
P108873	2/2/19	$786.09	HMO
P111792	6/14/19	$316.17	POS
P111792	6/14/20	$316.17	POS

BAS.00
BRUIN ACTUARIAL SOCIETY

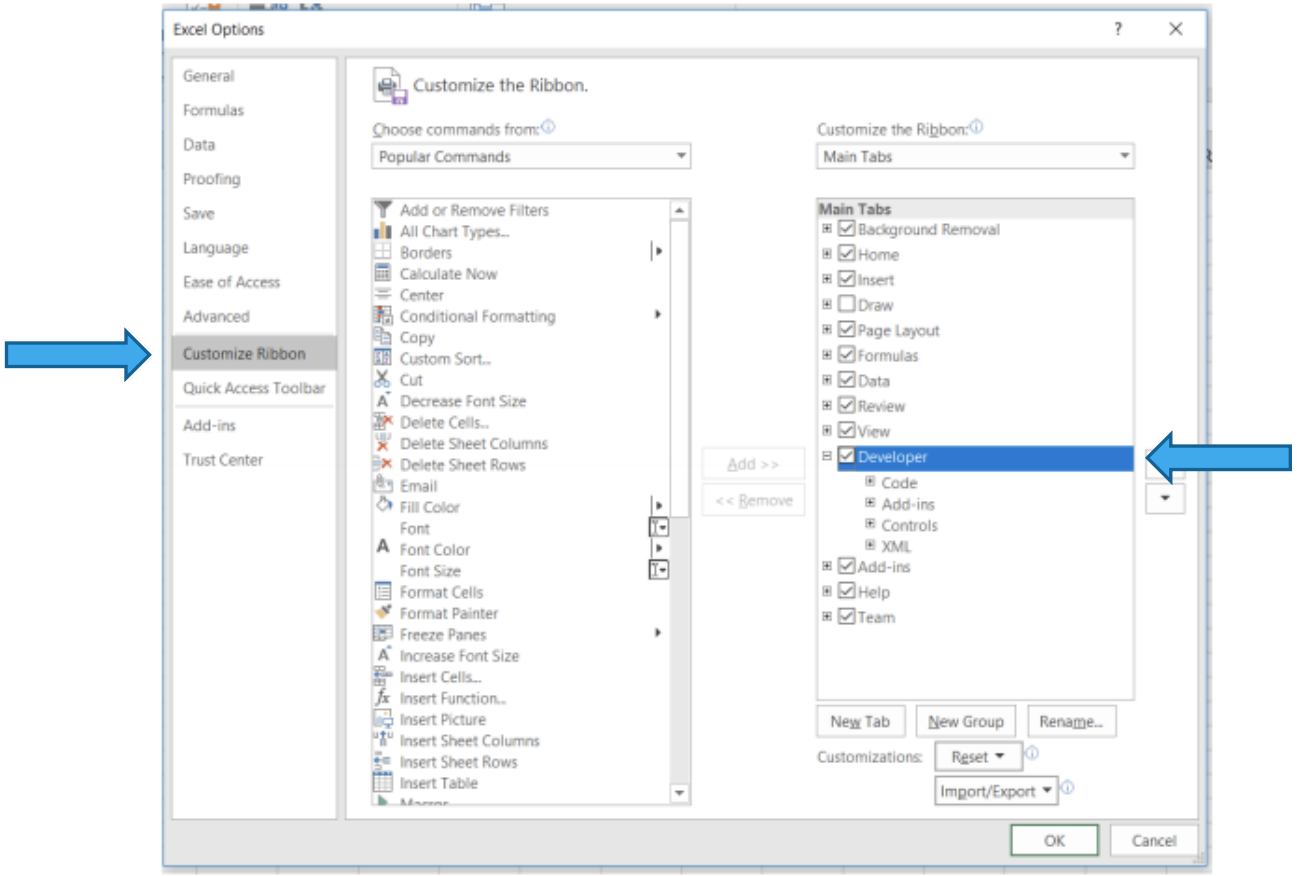
WHAT IS VBA?

- VBA stands for Visual Basic with Applications
- Closely related to Microsoft's Visual Basic
- Object-based language (similar to object-oriented languages)
- Used in Excel to simplify repetitive or complex tasks

BAS.00
BRUIN ACTUARIAL SOCIETY

ENABLING THE DEVELOPER TAB

File → **Option** → **Customize Ribbon** → **Check "Developer"**



BRUIN ACTUARIAL SOCIETY

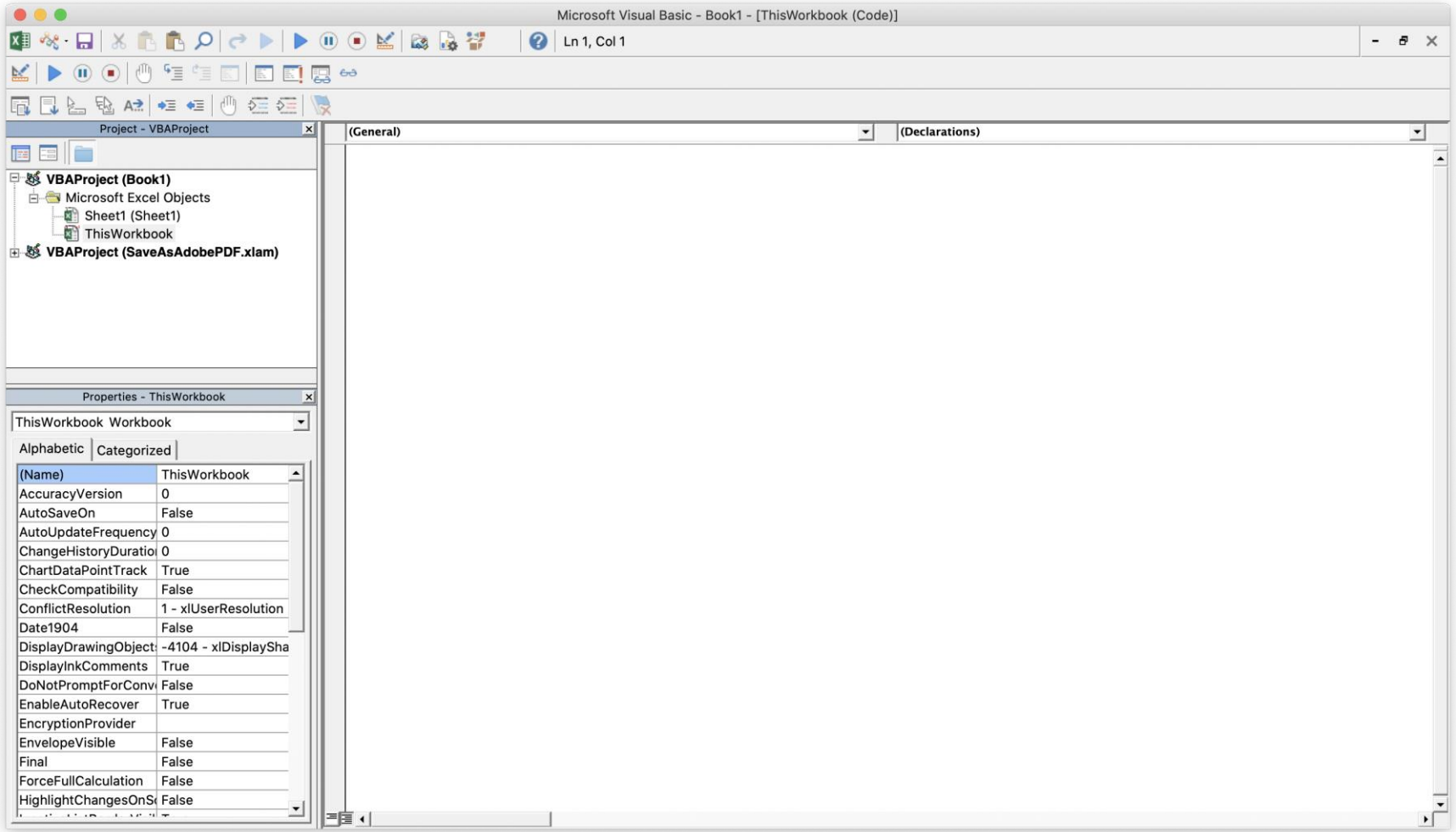
THE VBA ENVIRONMENT

Multiple ways to open the VBA Editor:

1) Go to Developer tab → Select Visual Basic
2) Keyboard shortcut:
    - Windows: Alt + F11
    - Mac: Opt + F11
3) Right-click on a worksheet → Select View Code

BAS.00
BRUIN ACTUARIAL SOCIETY

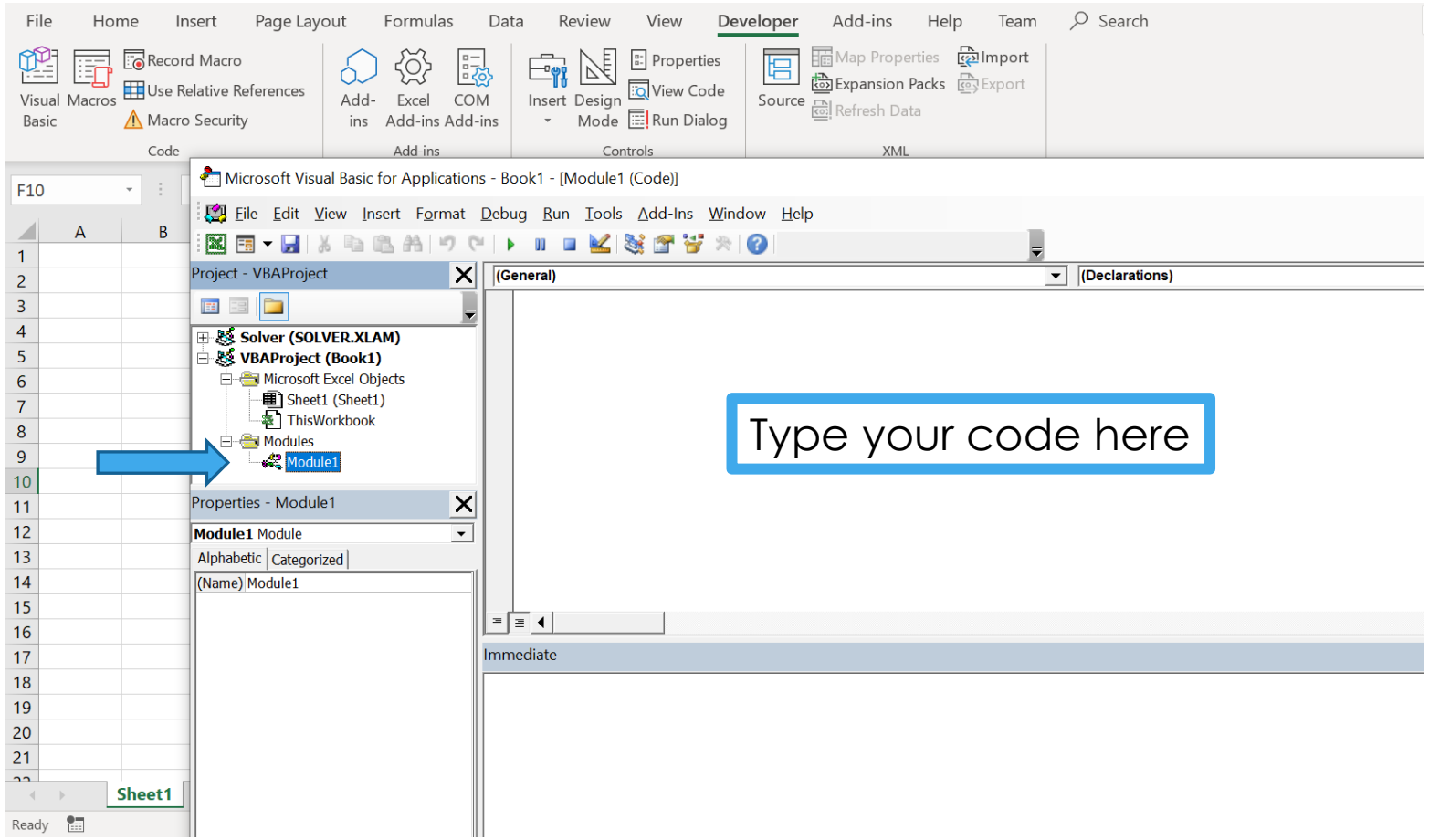
THE VBA ENVIRONMENT



BAS.00
BRUIN ACTUARIAL SOCIETY

GETTING STARTED

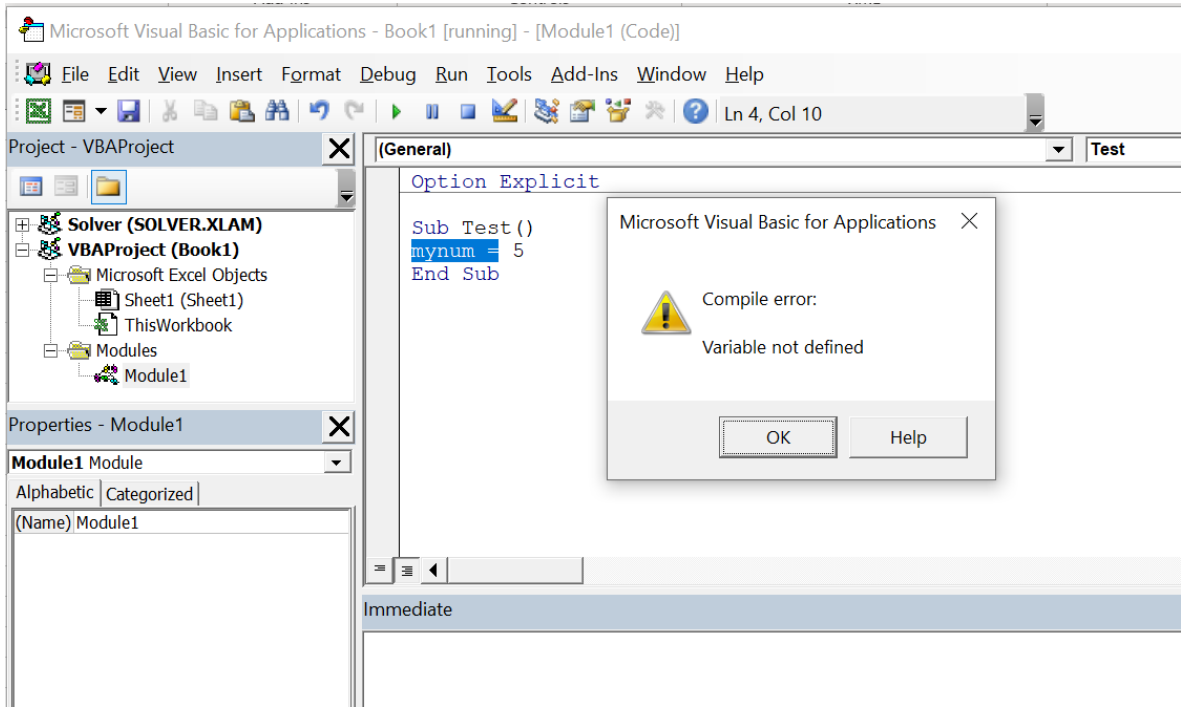
A **module** is where you can write VBA code



BAS.00
BRUIN ACTUARIAL SOCIETY

GETTING STARTED

- It is always a good idea to put `Option Explicit` in the declarations at the top of your module
- This forces you to declare all your variables
- You can either:
    1) Manually type this, or
    2) Go to Tools → Options → Check "Require Variable Declaration"



VBA IN MICROSOFT EXCEL

- We write **subroutines** (denoted Sub) to accomplish tasks that do not return values
- We write **functions** to take in inputs and return some output
- VBA objects include workbooks, worksheets, and ranges:

  Application.**Workbooks**("Book1.xlsm").**Worksheets**("Sheet1").**Range**("A1")

- Objects have properties and methods, the most important of which is:

  Range("A1").**Value**

BAS.00
BRUIN ACTUARIAL SOCIETY

VBA IN MICROSOFT EXCEL

- 2 commonly used methods in VBA: **Select** and **Activate**
- **Select** allows you to select one or more objects

```
Worksheets("Sheet1").Select

Range("A1:A3,C1:C3,E1:E3").Select
```

- **Activate** allows you to select one object. If you already have multiple objects selected, this allows you to select one object within them.

```
Range("A1").Activate
```

- **Ex:** You can select a worksheet where you want your code to run and then activate the first cell in that worksheet that you want to apply changes to.

BAS.00
BRUIN ACTUARIAL SOCIETY

SUBROUTINES

- Most of your VBA code will be in **subroutines** (also called **macros**)
- Subroutines can be used to perform calculations, change formatting, and copy and paste among other repetitive tasks
- We enclose our code in the following:

```
Sub Myroutine()


End Sub
```

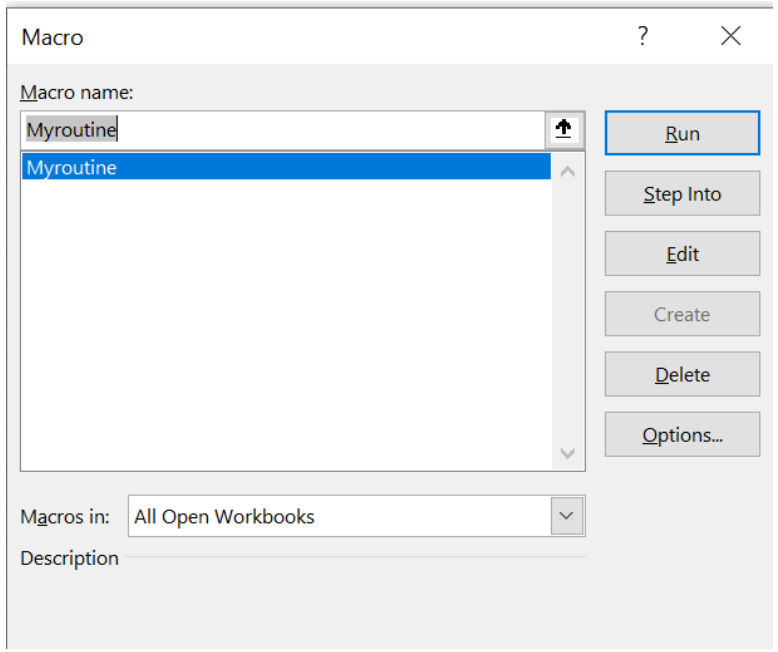
- In this case, we have named our subroutine "`Myroutine()`" (but we can name this whatever we want)

BAS.00
BRUIN ACTUARIAL SOCIETY

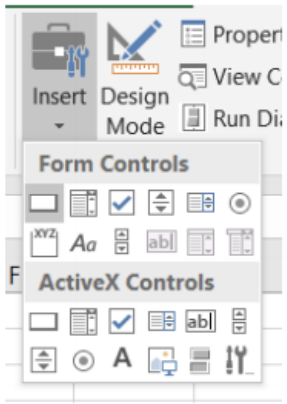
SUBROUTINES

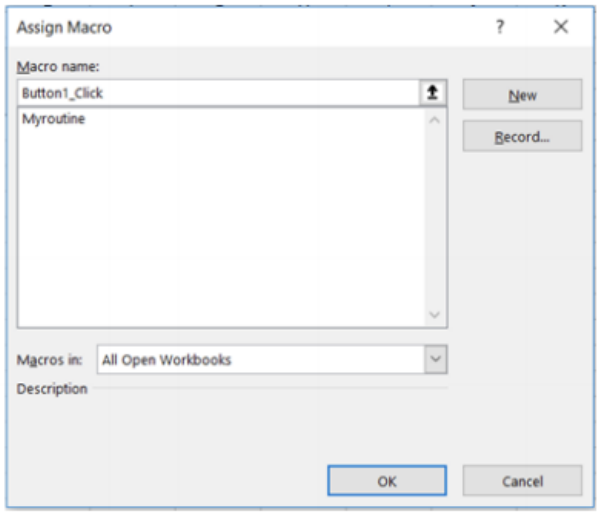
We can call our subroutine in several ways:

Clicking "Macros" in the Developer Tab



Adding a button or some tool





BAS.00
BRUIN ACTUARIAL SOCIETY

FUNCTIONS

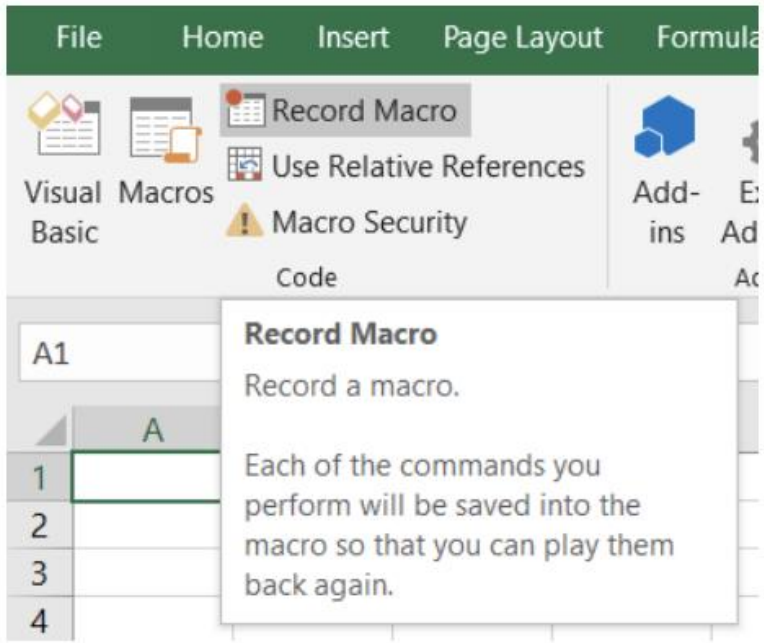
- Syntax

```
Function MyFunction(param1 As dtype, param2 As dtype,…) As dtype

    statements

    MyFunction = value

End Function
```

- Ultimately, you assign a return value by setting the name of the function to some value
- These functions can be called from workbook cells, like how you use COUNTIF(), VLOOKUP(), and other functions

BAS.00
BRUIN ACTUARIAL SOCIETY

MACRO RECORDER



- This is the easiest way to "write" a macro
- Under the Developer tab, click "Record Macro"
- All your actions will be translated into code, which you can find under Modules in the VBA Editor
- You can look at and modify this code to suit your purposes
- Often slow and clunky, but can be very powerful tool if used right (e.g. no one remembers how to filter/sort data in VBA, but this tool helps!)

BAS.00
BRUIN ACTUARIAL SOCIETY

VARIABLES

- Variable types: Integer, Double, String, Boolean, Date, Currency, and more
- We use the keyword `Dim` to declare a variable:

```
Dim mystring As String

Dim dbl As Double

Dim num As Integer

Dim rng As Range
```

- We assign them with "="

```
mystring = "Hello"

num = 5
```

- However, to assign a range, we use:

```
Set rng = Range("A1:B3")
```

Note: All highlighted words in sample code are variables that you can name whatever you want.

BRUIN ACTUARIAL SOCIETY

VARIABLES

- If the data type is not specified, the variable will be declared as a **variant**
- A variant can contain any kind of data, and the data type can change at any point
- Try to avoid these if possible (they require a lot of memory)

ARRAYS

- Arrays are created with parentheses to indicate size:

```
Dim myarray(5) as Integer
```

- The size can be changed:

```
ReDim myarray(10)
```

- But this will delete the data. To preserve the data inside, use:

```
ReDim Preserve myarray(15)
```

- Individual elements can be accessed and modified with parentheses:

```
myarray(0) = 5
```

- There's a lot more to be learned with arrays, but you can look online for more details. We'll work primarily with Workbook objects instead.

IF STATEMENTS

- Syntax

```
If condition Then
    [statements]
[ElseIf elseifcondition Then
    elseifstatements]
[Else
    elsestatements]
End If
```

- If `condition` is true, runs `statements`. If `condition` is false but `elseifcondition` is true, runs `elseifstatements`. Otherwise, runs `elsestatements`.
- There can be as many `ElseIf`s as needed

WHILE LOOPS

- Syntax

```
While condition
    statements
Wend
```

- Runs `statements` until `condition` evaluates to `FALSE`.
- Make sure that `condition` will not be `TRUE` forever, or you will have an endless while loop

FOR LOOPS

- Syntax

```
For counter = start To end [Step increment]
    statements
Next [counter]
```

- Typically, `counter` is an Integer that we increment
  - "`For i = 0 to 3`" will run 4 times (`i = 0, 1, 2, 3`)
  - "`For i = 2 to 7 Step 2`" will run 3 times (`i = 2, 4, 6`)
  - "`For i = 5 to 0 Step -3`" will run 2 times (`i = 5, 2`)

FOR EACH LOOPS

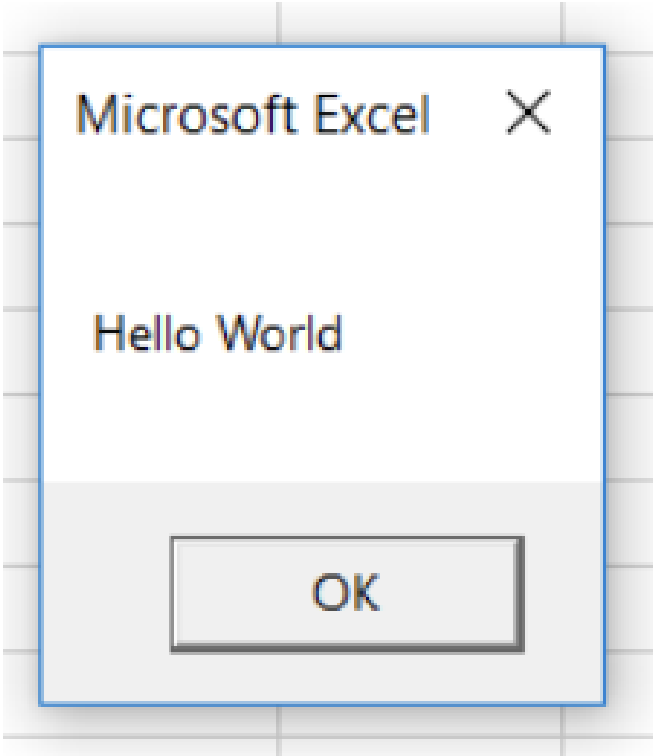
- Syntax

```
For Each cell In range
    statements
Next
```

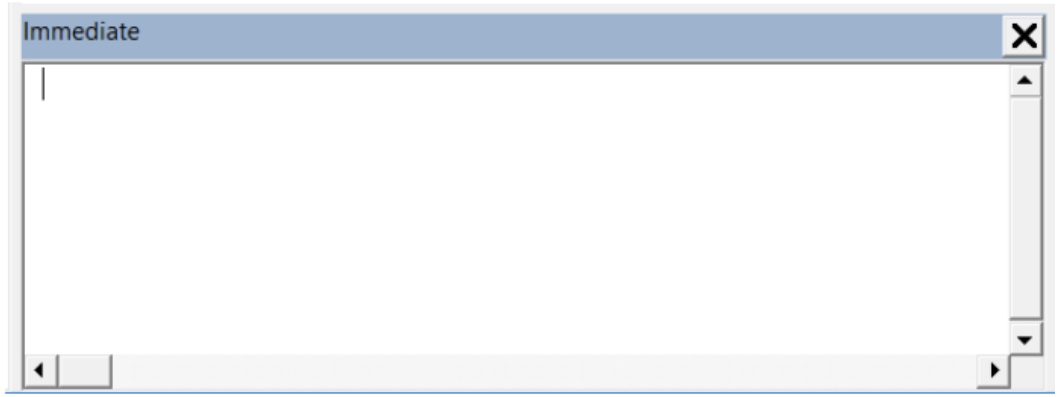
- A quick and simple way to loop through all the cells in a range
- `cell` and `range` must both be Range objects
- `range` should be initialized to some Range, `cell` need not be

MESSAGE BOXES

- Syntax: `MsgBox("Prompt")`
- You can replace "`Prompt`" with whatever you want to appear in a message box
- Can be very useful in debugging to display the value of variables



DEBUGGING TOOL: IMMEDIATE WINDOW



- Short lines of code can be run here
- You can also ask questions:

```
?Worksheets.Count

?Range("B2").Value
```

- If you use `Debug.Print` in your code, the output goes to this window

BAS.00
BRUIN ACTUARIAL SOCIETY

WORD OF CAUTION

Once you run a macro, you can't undo it

Make sure to save your workbook before running a macro to avoid losing your work in case the macro doesn't work as intended

BAS.00
BRUIN ACTUARIAL SOCIETY