

Engineering Analysis
& Problem Solving
ENG1101

Introduction to Computer Programming

Session Objectives

- Spreadsheet Tools
- Programming Basics
- Programming with VBA

Spreadsheet Formulas

- Use basic math operations
- Work directly with data on spreadsheet

Constant =	32
Temp (deg F)	Temp (C)
0	-17.78
32	0
70	21.11
100	37.78

1	B	C
2	Constant =	32
3		
4	Temp	Temp
5	(deg F)	(C)
6	0	$= (B6 - C\$2) * 5/9$
7	32	$= (B7 - C\$2) * 5/9$
8	70	$= (B8 - C\$2) * 5/9$
9	100	$= (B9 - C\$2) * 5/9$

In ENG1101

We will write small VBA programs that can be run:

1. as a spreadsheet **function**
2. as a **separate program**

Spreadsheet Functions

- Accept arguments and return a result
- Can perform complex sequences of math or other operations
- Built-In functions
 - Used for common tasks
 - Example: =average(A3:A6)
- User-Written
 - Many engineering-oriented functions not available in Excel
 - Sometimes we need to develop our own specially designed functions

Spreadsheet Macros

- Record a set of keystrokes
 - Used for repeated sequences
 - Often used to store sequences of non-mathematical operations
 - Example: adding a trendline to a graph
- Write a subprogram in Visual Basic for Applications (VBA)

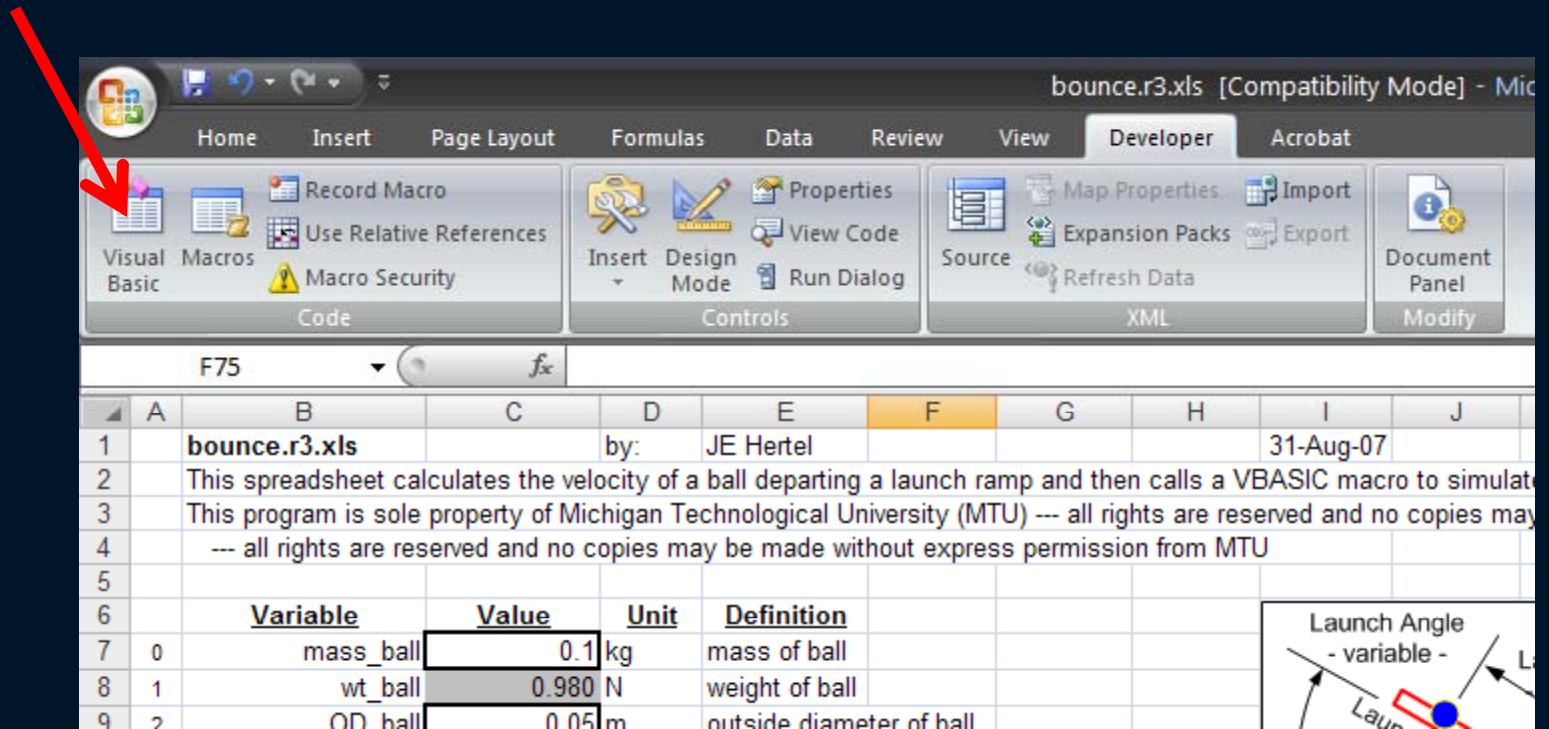
Spreadsheet VBA Programs

- Offer flexibility and power
 - Create graphical user interfaces (GUI)
 - Standard programming to control operations, perform calculations, and store values
- Used for special applications that require:
 - Speed for numerous calculations
 - Complex structure or calculation “flow control”
 - Or require GUI

Let's Write a Custom Function

As a team (Excel-07)

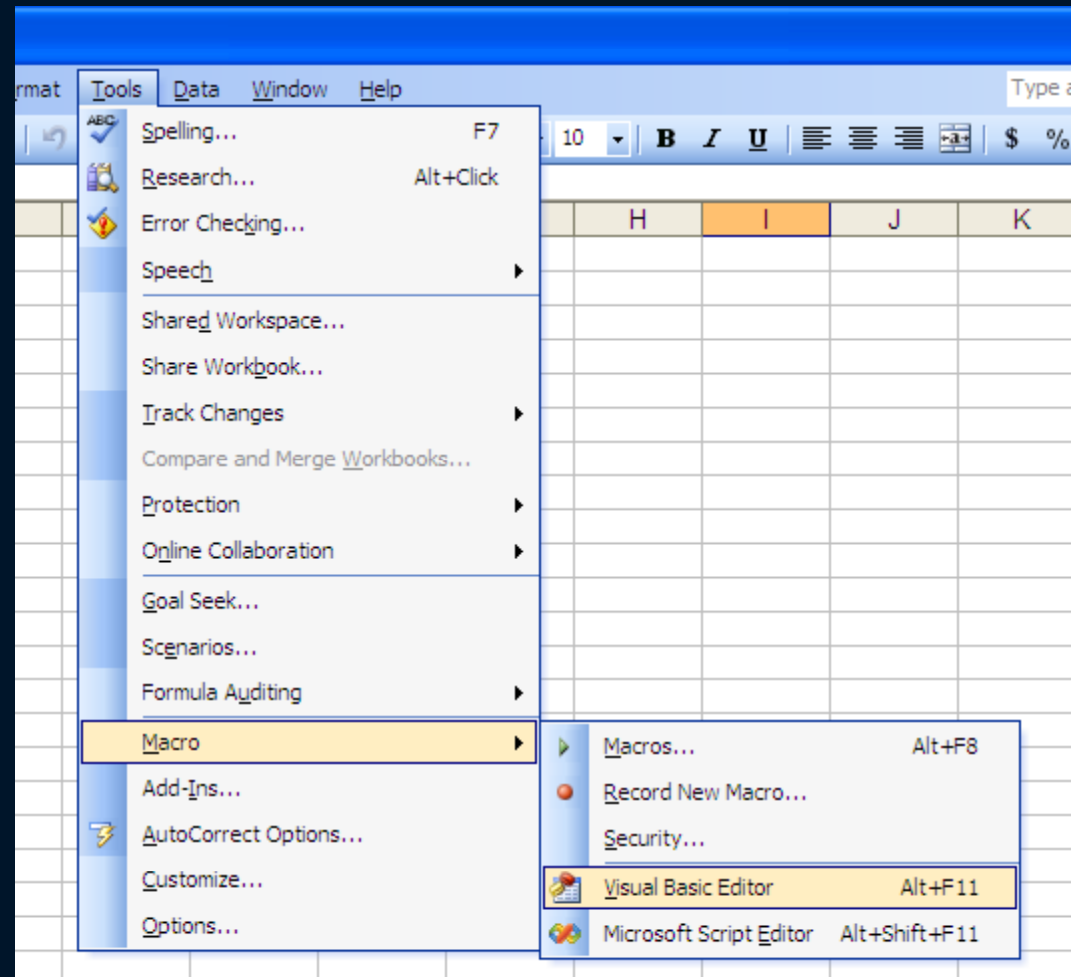
- Add in the "Developer" tab (Office button => Excel Options)
- Open the Visual Basic Editor (VBE)

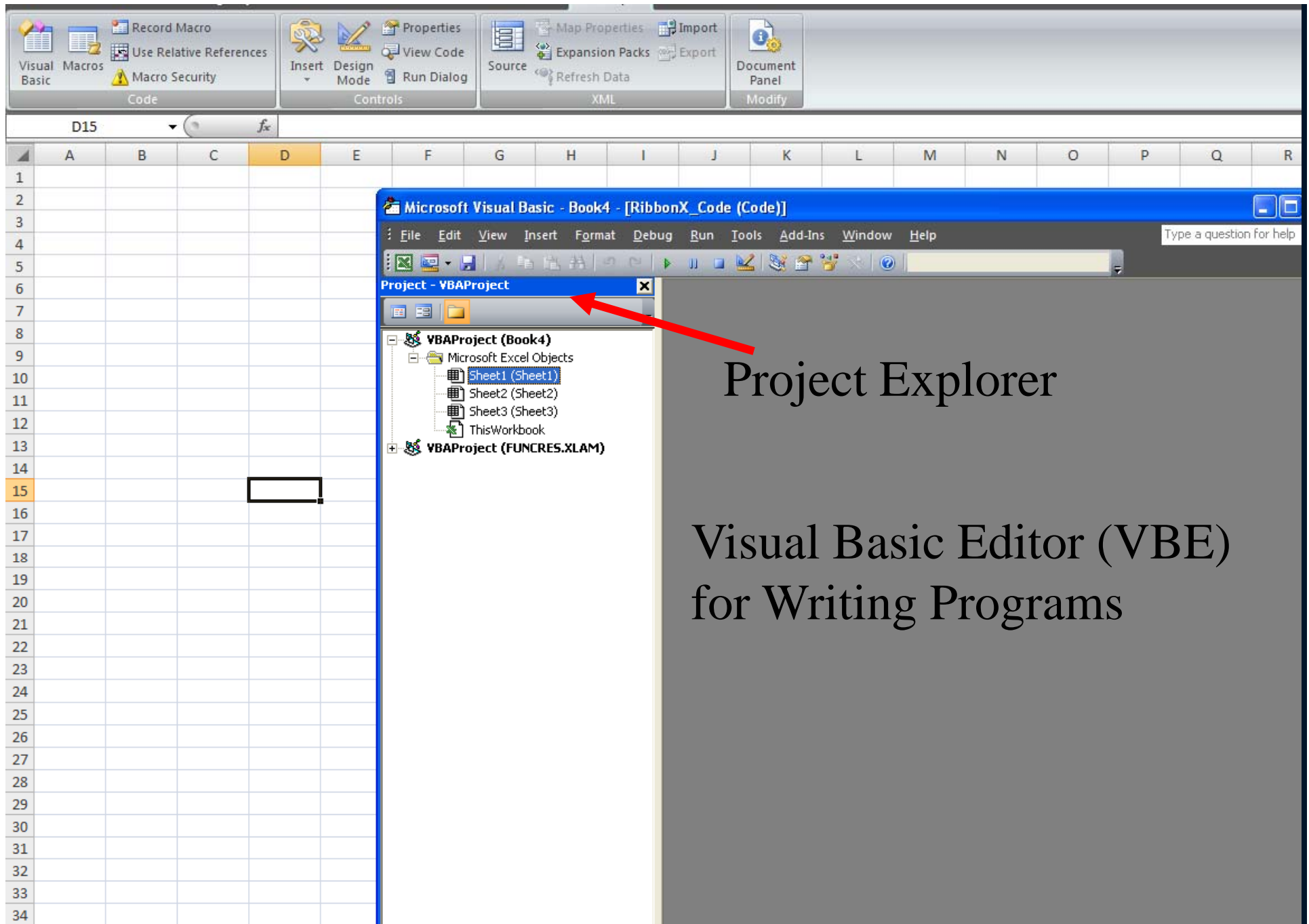


Let's Write a Custom Function

As a team (Excel-03)

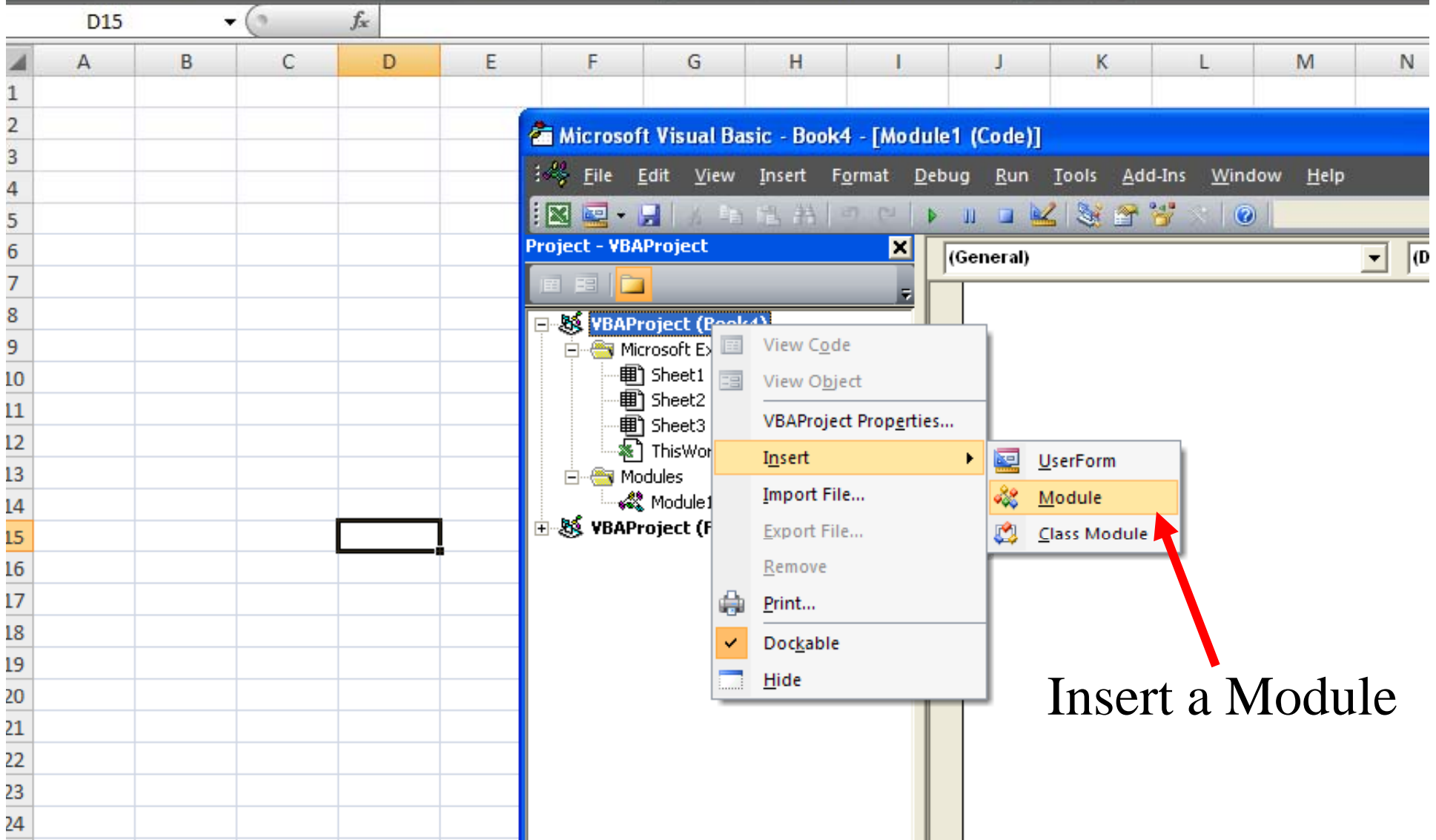
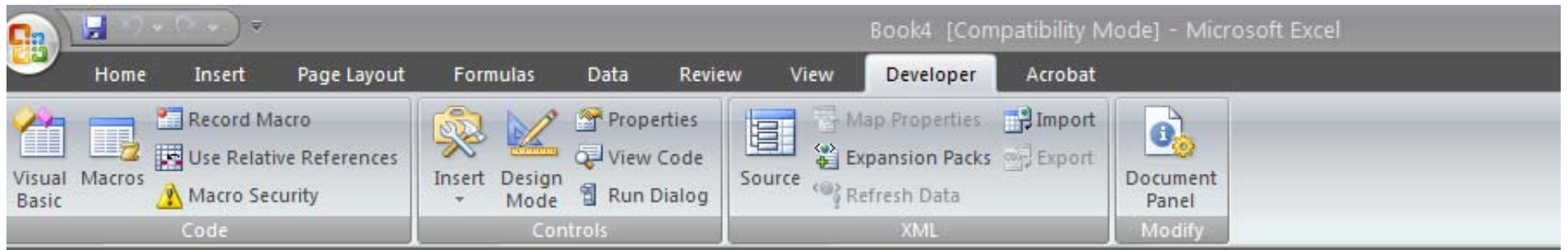
- Open Excel
- Open the Visual Basic Editor (VBE)
 - Alt-F11
 - **OR:** Tools, Macro, Visual Basic Editor





Project Explorer

Visual Basic Editor (VBE)
for Writing Programs



Insert a Module

Writing Custom Functions in VBA

- Excel uses radians to evaluate trig functions
- If your data is in degrees . . .

`=SIN(A3*PI()/180)`

- Let's write a user-written function to convert from radians to degrees

Using VBE

- In the Project Explorer window:
 - Click on: “VBA Project (Book 1)”
 - Select from the menu bar: Insert, Module
- Write the following code in the code window...
- Save and try it out! In your spreadsheet...

=sind(30)

```
Function sind(angle)
'Function to compute the sin of an
'angle in degrees

'set the value for pi
Pi = 3.14159
'calculate sine
sind = Sin(angle * Pi / 180)
End Function
```

Note: **sind** is the “Function” or “Program” Name

What is going on??

Function sind(angle)

'Function to compute the sine of
'an angle in degrees

'set the value for Pi

Pi = 4 * Atn(1)

'calculate sine

sind = Sin(angle * Pi / 180)

End Function

All functions start with
a line defining the function
"Function name(arguments)"

Calculations

One calculation must be named
the same as your function.
The function outputs this value
when the function is used.

VBE editor puts this here
automatically

What is going on??

Function sind(angle)

'Function to compute the sine of
'an angle in degrees

'calculate the value for pi

'Pi = 3.14159

Pi = 4 * Atn(1)

'calculate sine

sind = Sin(angle * Pi / 180)

End Function

“Comment Out” a line to keep it from executing.

Try using the arctan function for getting a more accurate value for Pi.

Call sind as a function

The image shows a screenshot of the Microsoft Visual Basic editor and an Excel spreadsheet. The spreadsheet on the left has a grid with rows 5 to 16. Cell B14 contains the formula `=sind(30)`. A red arrow points from the text 'Call sind as a function' to this cell. The Visual Basic editor on the right is titled 'Microsoft Visual Basic - Book4 - [Module1 (Code)]'. The Project Explorer shows a project named 'VBAPROJECT (FUNCREX.XLAN)' with a module named 'Module1'. The Properties Window shows the 'sind' function. The code in the Properties Window is as follows:

```
Function sind(angle)  
'Function to compute the sin of an an  
'angle in degrees  
'angle=angle, degrees  
  
'set the value for pi  
Pi = 3.14159  
'calculate sine  
sind = Sin(angle * Pi / 180)  
  
End Function
```

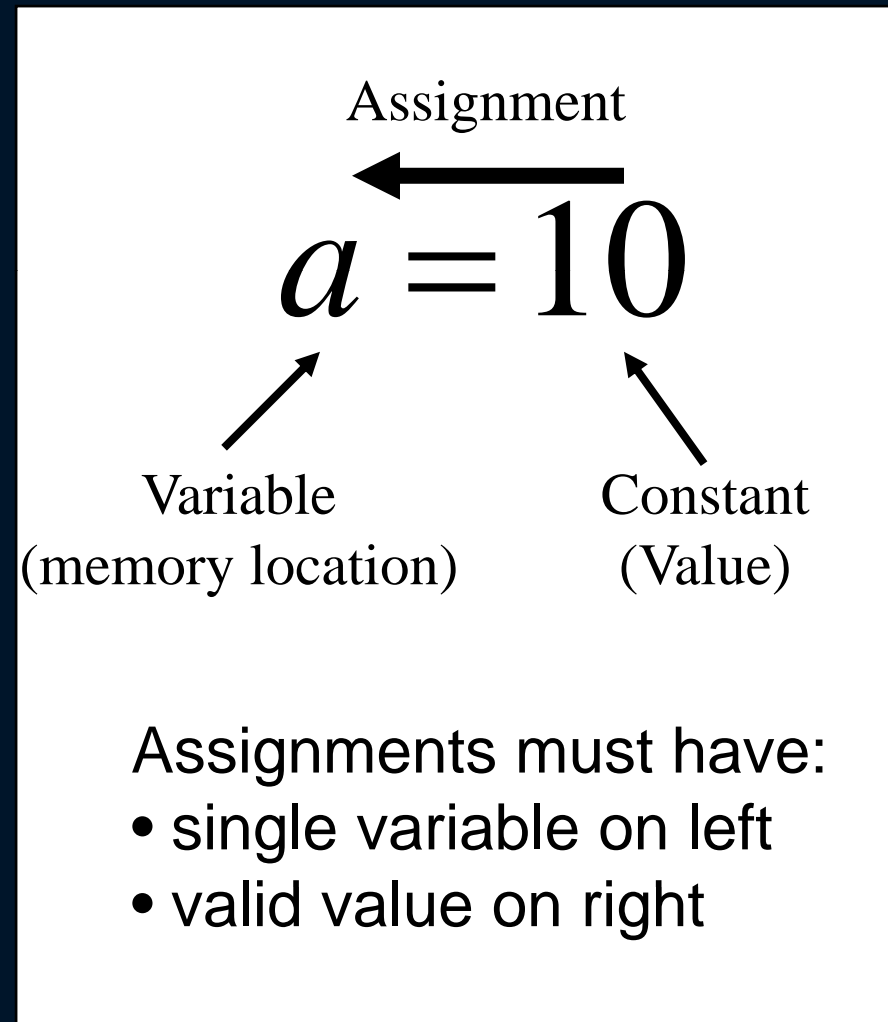
Programming Basics

- Program is a sequence of unambiguous instructions, similar to driving directions
- Instructions are called statements
- Logic and Order of statements must be correct
- Computers only do what you tell it to do

The Concept of Assignment

Information within program is stored:

- Directly as *constant* (example, 9.81), whose value does not change
- Symbolically as a *variable* ($a=9.81$), whose value can be changed



Equal Sign

- In programming the equal sign means:
 - “Is replaced by”
 - “Is assigned the result of”
 - Valid assignment examples:
 $c=a+b$, $x=x+1$
- Equal sign does **NOT** mean equality as in algebra
 - Invalid assignment examples:
 $a+b=c$; $3=x+y+99$

As a Team

- Which of the following are valid assignments?

A) $m=5+2$

B) $9.8=g$

C) $n=m/3$

D) $2=x*t+3$

E) $s=-1.5$

F) $y=y+2$

As a Team

- Which of the following are valid assignments?

A) Valid: $m=5+2$

B) **Invalid:** $9.8=g$

C) Valid (as long as m has been assigned): $n=m/3$

D) **Invalid:** $2=x*t+3$

E) Valid: $s=-1.5$

F) Valid: $y=y+2$

What is going on??

Function `sind(angle)`

'Function to compute the sine of
'an angle in degrees

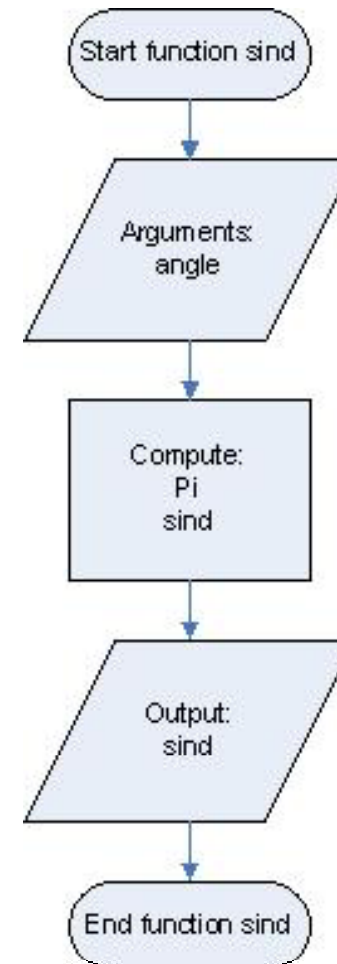
'calculate the value for pi

$$\text{Pi} = 4 * \text{Atn}(1)$$

'calculate sine

$$\text{sind} = \text{Sin}(\text{angle} * \text{Pi} / 180)$$

End Function



What is going on??

```
Function sind(angle)
'Function to compute the sine of
'an angle in degrees
'angle=angle, degrees

'calculate the value for pi
Pi = 4 * Atn(1)

'calculate sine
sind = Sin(angle * Pi / 180)

End Function
```

These are comments.
They are important.

VBA has some
functions (like Atn*
and Sin) but doesn't
have others (like Pi)

*Atn is arctangent

Some Built-In Numeric VBA Functions

Purpose	VBA Function	Excel Function
Absolute value	Abs(x)	ABS(x)
Truncate to integer	Int(x)	INT(x)
Round x to n digits after decimal	Round(x,n)	ROUND(x)
Square root	Sqr(x)	SQRT(x)
Exponential, e	Exp(x)	EXP(x)
Natural log	Log(x)	LN(x)
Base-10 log	-	LOG10(x)
Base-b log	-	LOG(x,b)
Value of pi	-	PI()
Sine	Sin(x)	SIN(x)
Cosine	Cos(x)	COS(x)
Tangent	Tan(x)	TAN(x)
Arcsine	-	ASIN(x)
Arccosine	-	ACOS(x)
Arctangent	Atn(x)	ATAN(x)
Arctangent (4 quadrant)	-	ATAN2(x,y)
Degrees to radians	-	RADIANS(x)
Radians to degrees	-	DEGREES(x)
X modulo y	X Mod y	MOD(x,y)
Random number	Rnd()	RAND()

Bold indicates functions that differ between VBA and Excel.

Table from: Chapra, S.C. Power Programming With VBA/EXCEL, pg. 93.

Using Excel Functions in VBA

- Some functions can be called from Excel to use in your VBA code.
- Calling an Excel function in VBA:

`Application.WorksheetFunction.functionname(argument)`

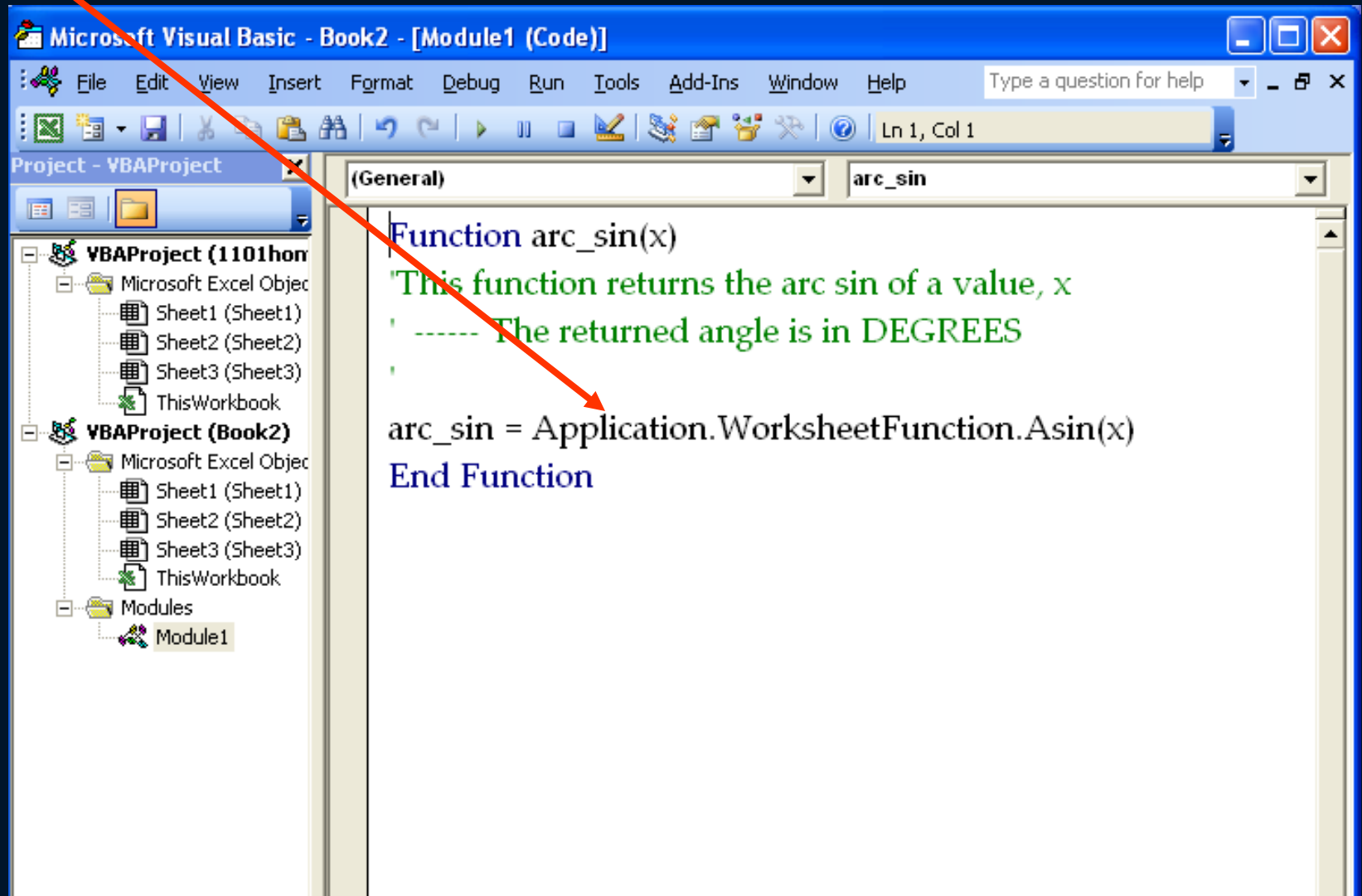
- Example: inverse cosine of 0.5

`Application.WorksheetFunction.Acos(0.5)`

- Do a help search on “Visual Basic Functions”
 - “List of Worksheet Functions Available to Visual Basic”
 - “Using Microsoft Excel Worksheet Function in Visual Basic”

Using Excel Functions in VBA

- Asin () in Excel



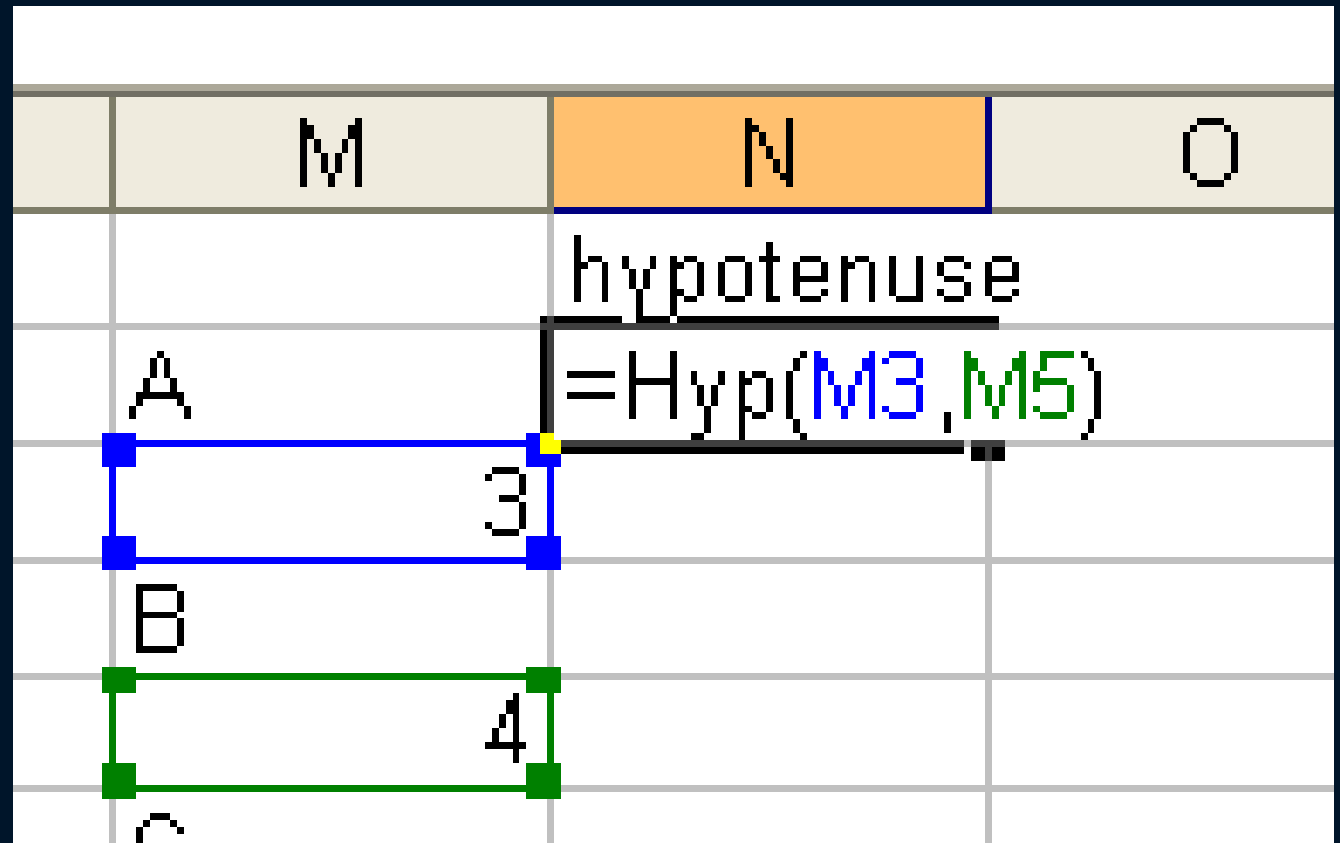
The screenshot shows the Microsoft Visual Basic Editor window for 'Book2 - [Module1 (Code)]'. The interface includes a menu bar (File, Edit, View, Insert, Format, Debug, Run, Tools, Add-Ins, Window, Help), a toolbar, and a Project Explorer on the left. The Project Explorer shows a tree view of the VBAProject (Book2) with folders for Microsoft Excel Objects (Sheet1, Sheet2, Sheet3, ThisWorkbook) and Modules (Module1). The main editor area displays the code for a function named 'arc_sin'. The code includes a function signature, a comment in green text explaining the function's purpose and that the returned angle is in degrees, and the function body which calls the 'Asin' method of the 'WorksheetFunction' object. A red arrow points from the 'Asin' text in the code to the 'Asin ()' text in the bullet point above.

```
Function arc_sin(x)  
    'This function returns the arc sin of a value, x  
    '----- The returned angle is in DEGREES  
    '-----  
    arc_sin = Application.WorksheetFunction.Asin(x)  
End Function
```

Sending Multiple Arguments to a Function

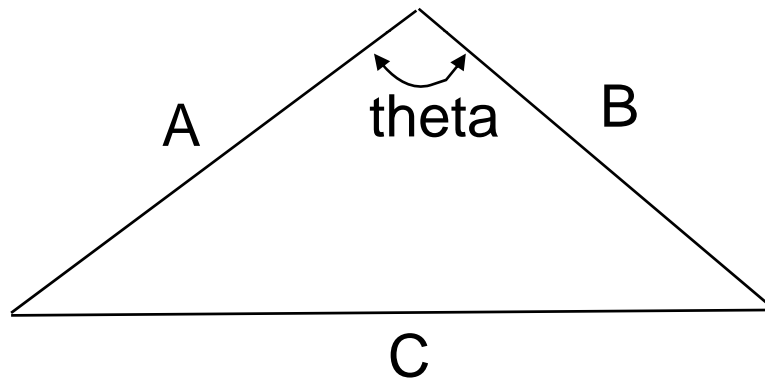
- $A^2+B^2=C^2$
- VBA Function Hyp(A,B)

	M	N	O
		hypotenuse	
A		=Hyp(M3,M5)	
	3		
B			
	4		
C			



As a Team...

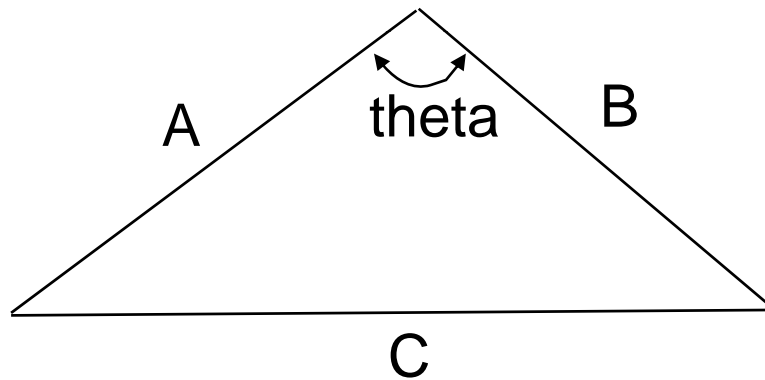
- Write a function that uses A , B , and C as **arguments** to compute the angle theta for a general triangle shown below.



- Hint:
 - Use the Law of Cosines
 - (What **is** the Law of Cosines?...)

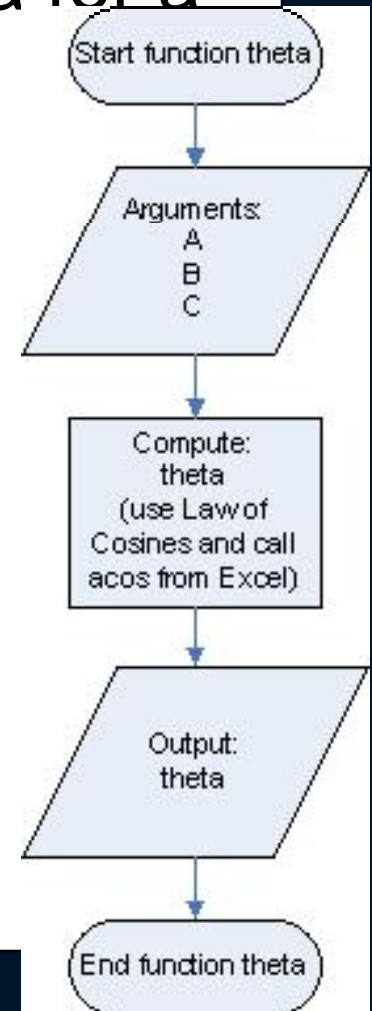
As a Team...

- Write a function that uses A, B, and C as arguments to compute the angle theta for a general triangle shown below.



Hint:

- $C^2 = A^2 + B^2 - 2AB\cos(\text{theta})$
- `Application.WorksheetFunction.Acos(argument)`



Units?

- What are the units of theta?
- Convert theta to degrees if you haven't done so.
 - Hint: `Application.WorksheetFunction.Pi()`
 - 180 degrees = π * radians

Debugging in VBE

- If you make a mistake and your code will not run
 - VBE debugger will highlight the line with the problem
 - Fix the error.
 - *** Click on the “reset” (blue box) or “run” (blue arrow) icons.
 - OR to trouble shoot use F8 to step through the program
- If your function/macro just “hangs”...
 - Hit the “escape” key
- If you update an already working function and want your spreadsheet to update...
 - Hit F9