Fundamentals of Programming with VBA

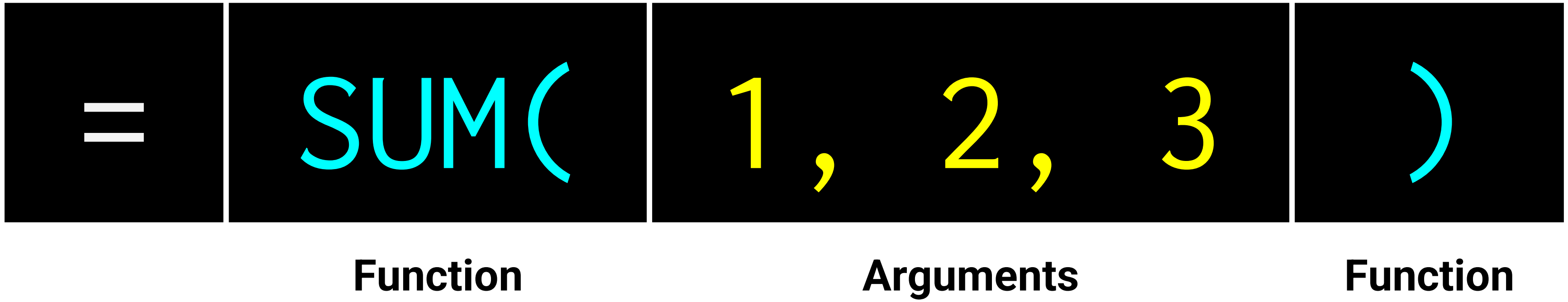
Data Boot Camp

Lesson 2.1

Intro to Programming Logic

Ooh, Coding! (Sort of...)

In a way, using Excel has introduced you to a sort of proto-programming. When writing scripts in VBA, you will rely on `functions` (methods) that do something to or with `arguments`.



```
= SUM( 1, 2, 3 )
```

Fundamental Tools of Programming

These structures are found in nearly all programming languages:

- Conditionals
- Iterations
- Functions
- Variables / Arrays

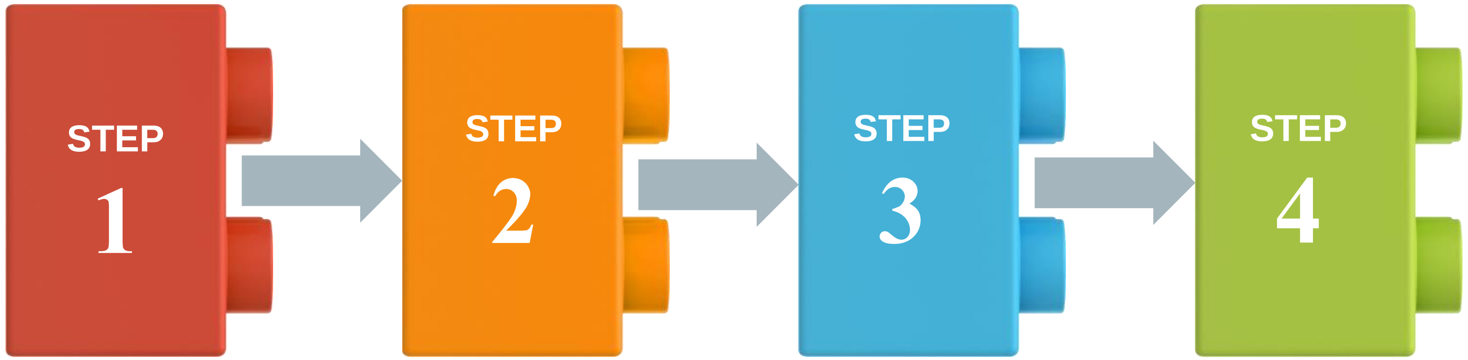
How a Computer Thinks (Procedurally)

Every problem in software development begins with a complex and abstract real-world need.

How a Computer Thinks (Procedurally)

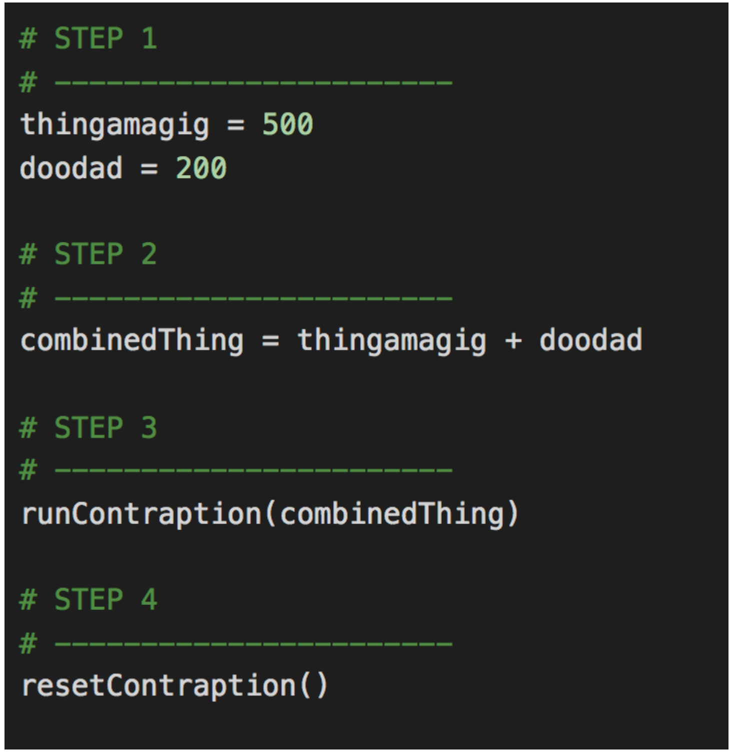
In order for a computer to interpret it, the real-world problem must be broken down into a set of procedural steps.

Complex Real-World Problem

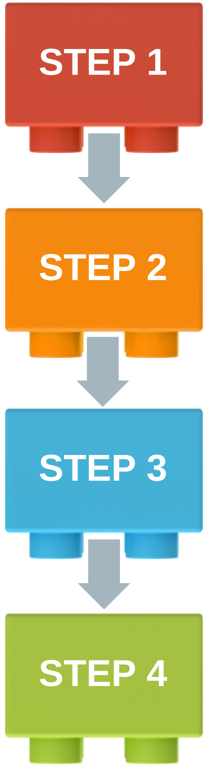


How Code Is Written (Procedurally)

Code (Python)



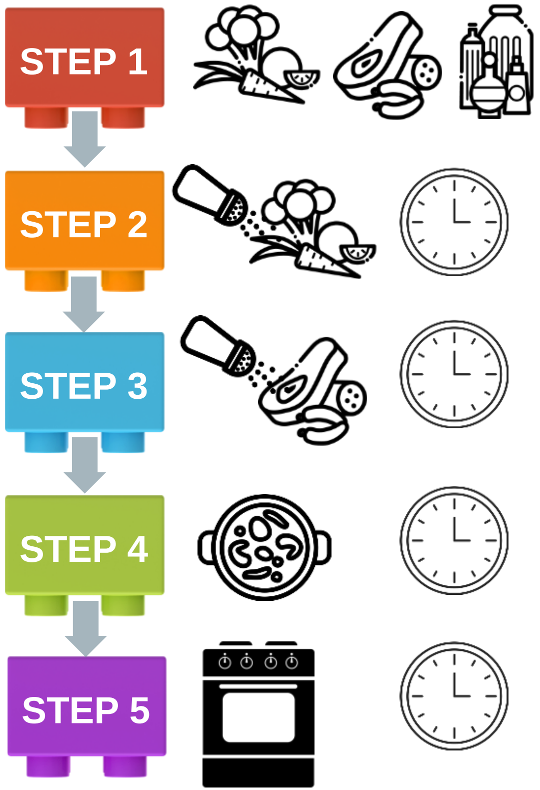
```
# STEP 1
# -----------------------
thingamagig = 500
doodad = 200

# STEP 2
# -----------------------
combinedThing = thingamagig + doodad

# STEP 3
# -----------------------
runContraption(combinedThing)

# STEP 4
# -----------------------
resetContraption()
```



When Procedures Aren't Enough... We Need More Tools!

Code (Python)

```
# STEP 1
# -----------------------
ingredient1 = vegetables
ingredient2 = meats
ingredient3 = spices

# STEP 2
# -----------------------
season(vegetables)

# STEP 3
# -----------------------
season(meats)

# STEP 4
# -----------------------
stirfry(vegetables)

# STEP 5
# -----------------------
roast(meats)
```



To Make a Sandwich

To Make a Sandwich

Logical Procedure:

01 Get bread, peanut butter, and jelly from pantry.

02 Lay out bread on table.

03 Open jars of peanut butter and jelly.

04 Get spreading knife.

05 Use knife to spread peanut butter.

06 Use knife to spread jelly.

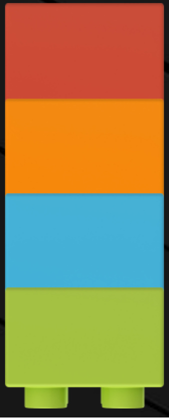
07 Combine bread to create sandwich.

Fundamental Tools Can Help Make the Sandwich

We use these tools as building blocks to make an ideal sandwich procedure:

Conditionals	If peanut butter is crunchy, use less.
Iterations	While there is more peanut butter, add more jelly.
Functions	Spread the condiment using a knife.
Variables / Arrays	The ingredients are bread, peanut butter and jelly.

VBA Building Blocks



Variables, Types, and Arrays

Variables: The Nouns of Code

- **Variables** are effectively the items in a procedure.
- They can be **physical things** (like an ingredient) or **abstractions** (like a counter).
- In VBA, items can be **declared** as variables by using `dim` followed by the type. Then they can be **assigned** a value.

Variable Declaration

```
dim ing1 as String
dim ing2 as String
dim budget as Double
```

Variable Assignment

```
ing1 = "Peanut Butter"
ing2 = "Jelly"
budget = 5.00
```

Types and typing

The **type** of a variable is super important in programming

In most languages, functions are only defined/make sense for inputs of a certain type.

“a” + “b” = “ab”   but   1 + 2 != 12   add(string, string) is different than add(int, int)

```
‘text
dim ing1 as String
‘integers (<2^16)
dim ing2 as int
‘integers (<2^32)
dim longnumber as long
‘decimals
dim budget as Double
‘true or false
dim exceeded as Boolean
```

Array: A Collection of Items

Arrays are effectively **groups** of related items. They present another way to store and reference similar pieces of information.

Item 0	**Item 1**	**Item 2**
`["Peanut Butter",`	`"Jelly",`	`"Bread"]`

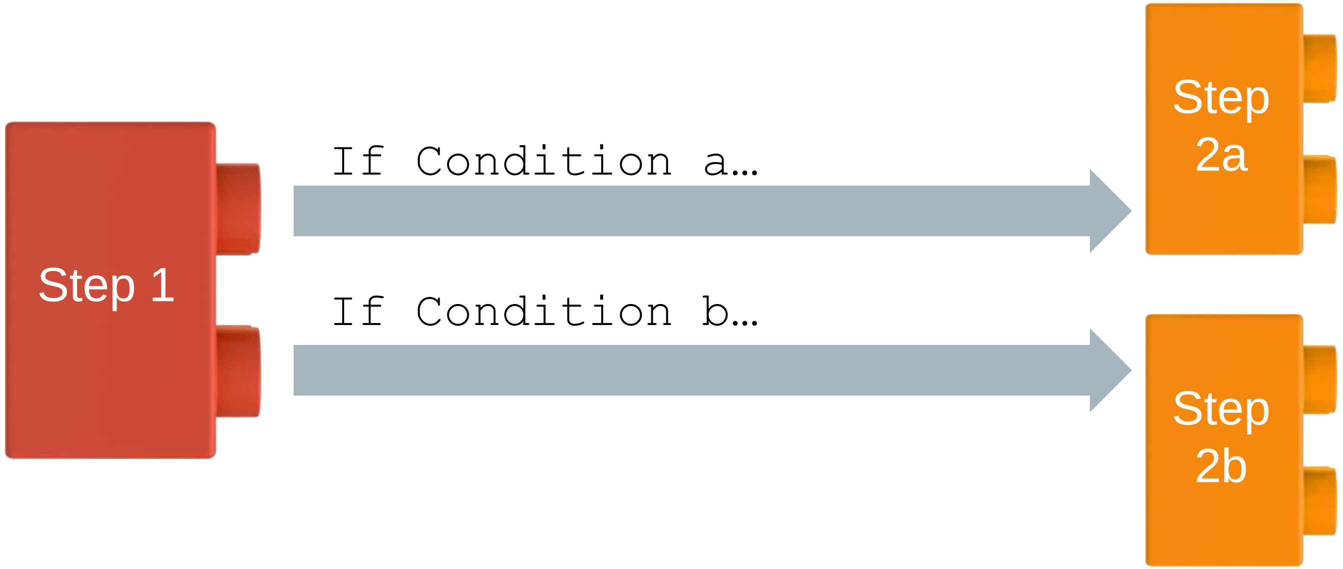
```
dim ingredients(0 to 2) as String

ingredients(0) = "Peanut Butter"
ingredients(1) = "Jelly"
ingredients(2) = "Bread"
```

Conditionals

Conditionals: If This, Then That

Conditionals can control the flow of logic based on certain conditions being met.

In most languages, you use **if/else** code for this purpose.

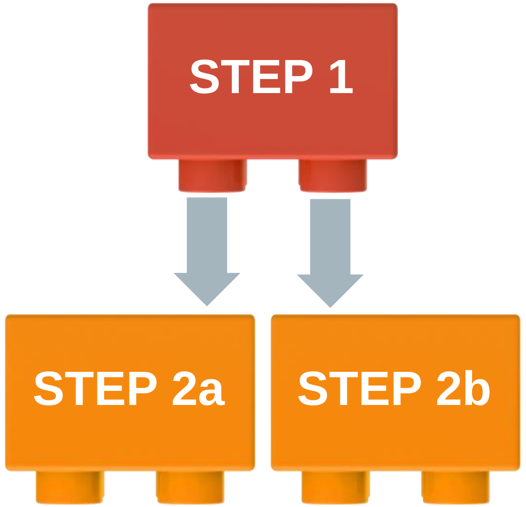


Conditionals: If This, Then That

In VBA, conditionals are declared using the keywords **If**, **Then**, **Elseif**, **Else**, and **End if**.

VBA lets us create far more sophisticated conditional logic than with Excel formulas alone.

```
If (pbThickness > 1.0) Then
    stopSpreading()

Else
    spreadMore()

End if
```

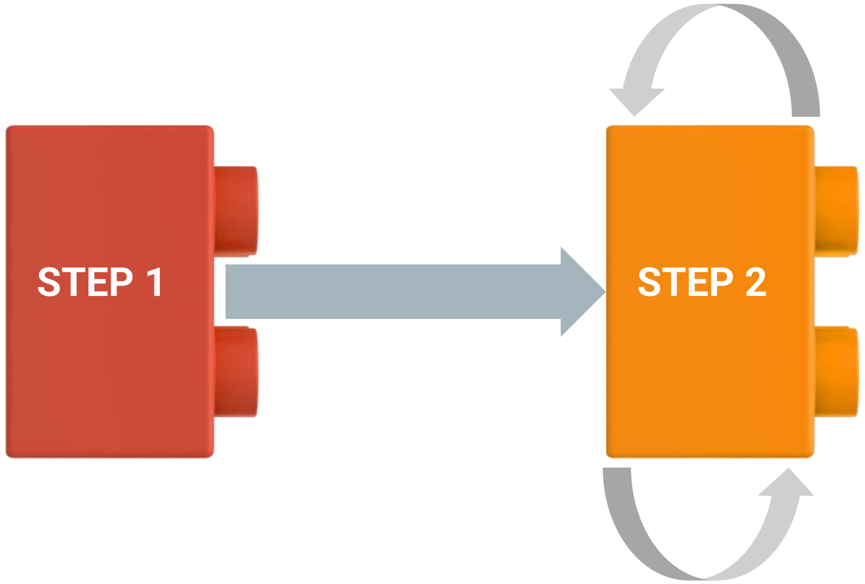


Iteration (Looping)

Iteration: Round and Round We Go!

Iteration is the concept of using loops to perform a group of tasks repeatedly a number of times.

Almost all programming languages use **for loops** and **while loops** for iteration.



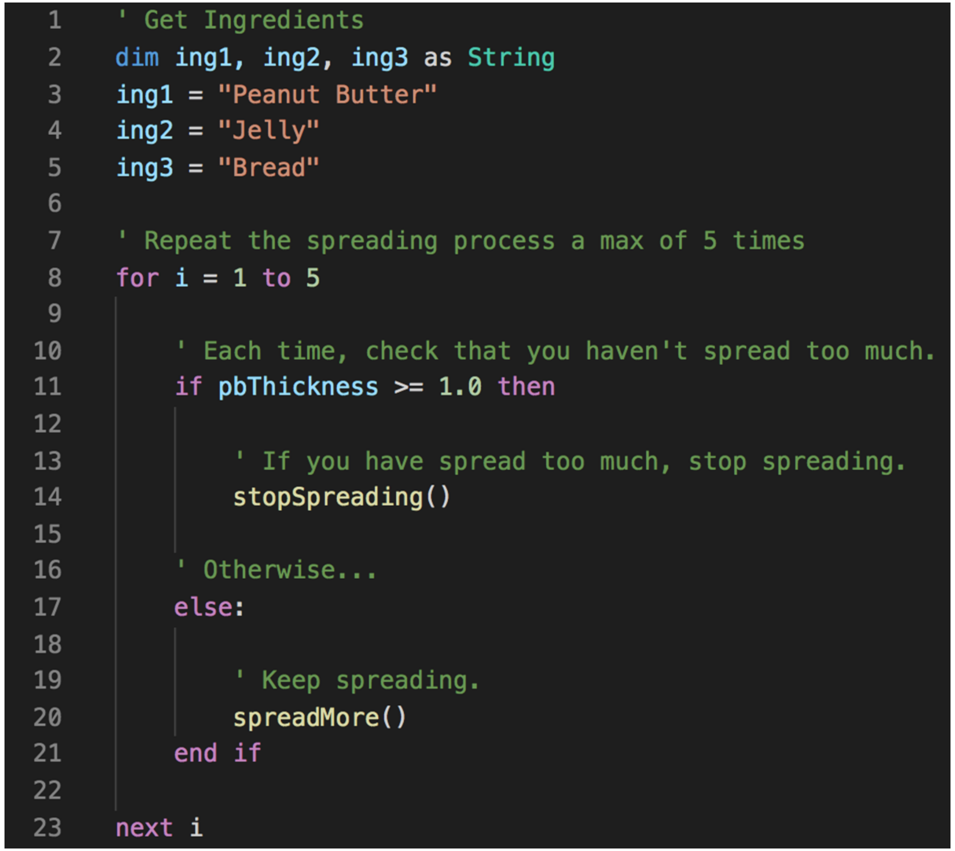
STEP (repeat X times)

Iteration: Round and Round We Go!

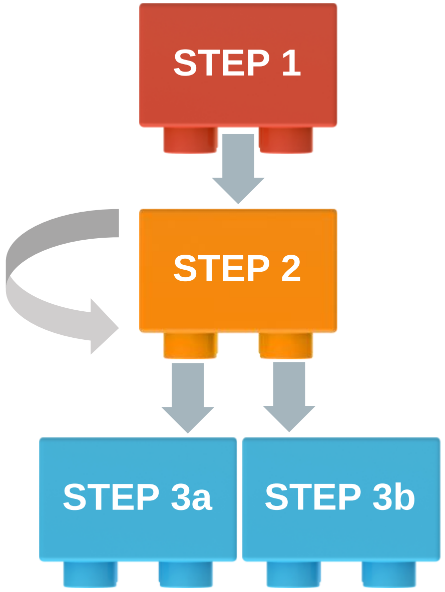
This code will make more sense later. Basically, it’s the VBA way of repeating the same block multiple times.

```
' Repeat the same step until i becomes 20
For i = 0 to 20

   ' Each time spread more
   spreadMore()

' Add one to the value of i each time
Next i
```

Build the Program!



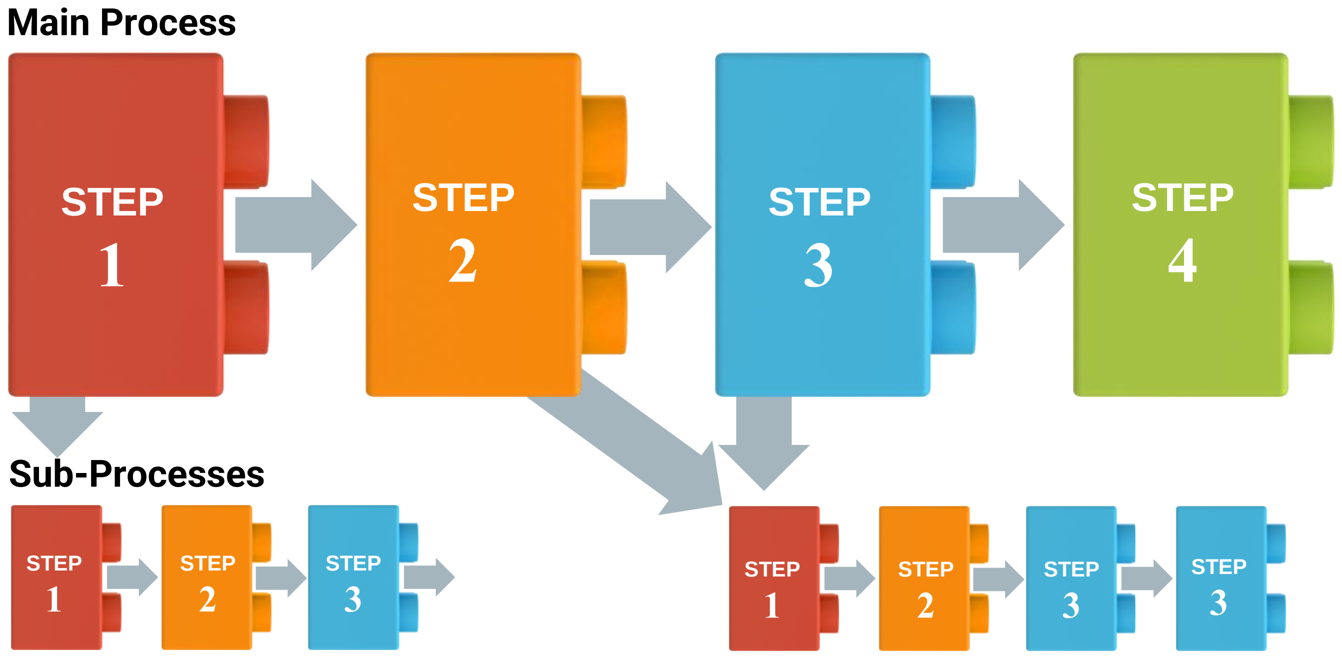
```
' Get Ingredients
dim ing1, ing2, ing3 as String
ing1 = "Peanut Butter"
ing2 = "Jelly"
ing3 = "Bread"

' Repeat the spreading process a max of 5 times
for i = 1 to 5

    ' Each time, check that you haven't spread too much.
    if pbThickness >= 1.0 then

        ' If you have spread too much, stop spreading.
        stopSpreading()

    ' Otherwise...
    else:

        ' Keep spreading.
        spreadMore()
    end if

next i
```



Functions

Functions: When One Block Can’t Do It All!

In essence, **functions** are a sort of sub-process. They let you create premade, reusable blocks of code that can be called on demand.



Putting It All Together

To Make a Sandwich

Logical Procedure:

01 Get bread, peanut butter, and jelly from pantry.

02 Lay out bread on table.

03 Open jars of peanut butter and jelly.

04 Get spreading knife.

05 Use knife to spread peanut butter.

06 Use knife to spread jelly.

07 Combine bread to create sandwich.

To Make a Sandwich (Full Logic)

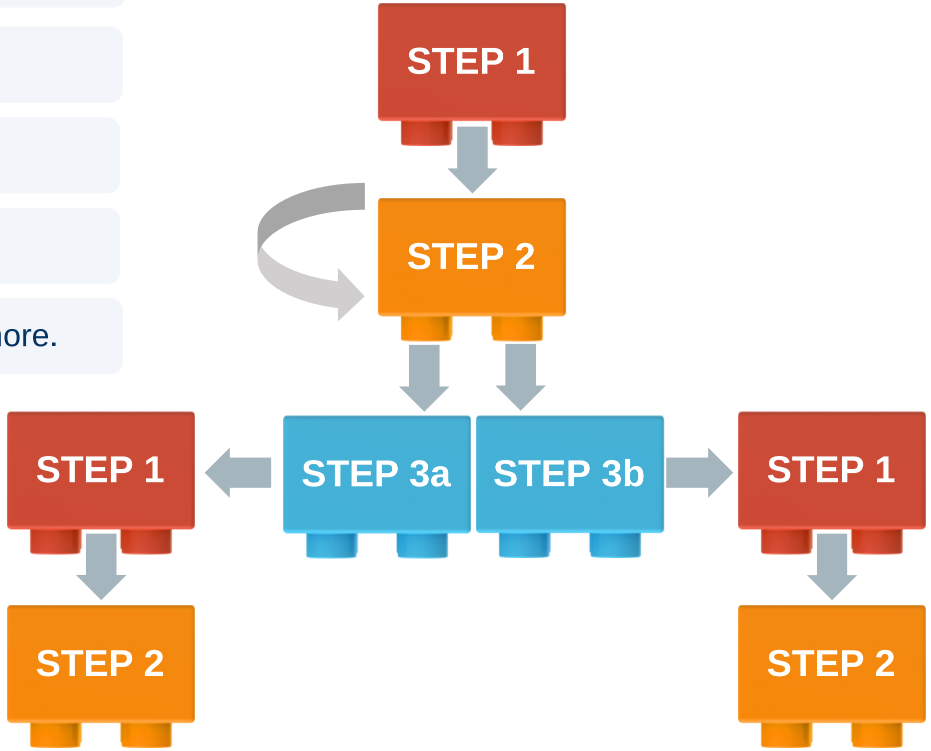
01 Get items.

02 **Repeatedly** “spread the peanut butter.”

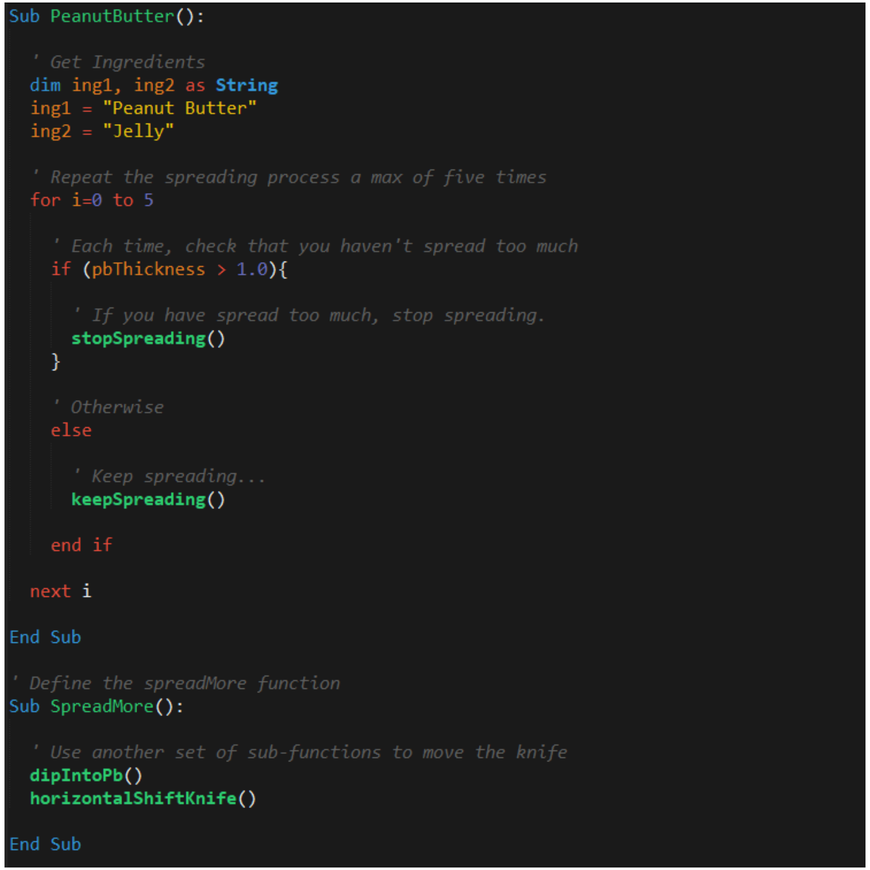
03 Check if thickness **condition** is met.

3a If thickness condition is met, run stop **function**.

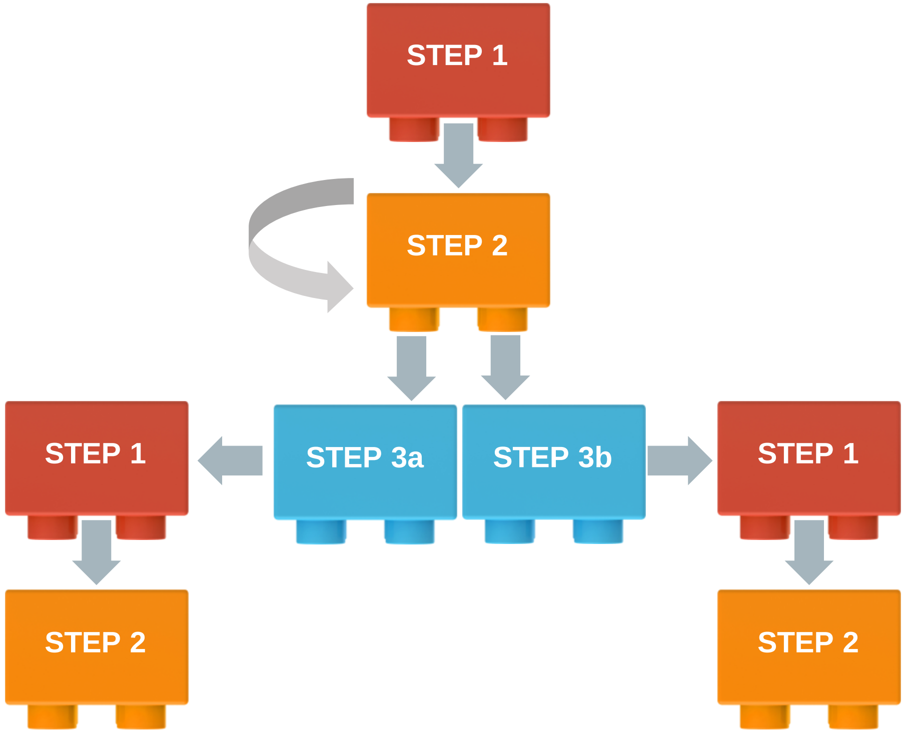
3b If thickness condition is **not** met, then spread more.



To Make a Sandwich (in Code)



```
Sub PeanutButter():

  ' Get Ingredients
  dim ing1, ing2 as String
  ing1 = "Peanut Butter"
  ing2 = "Jelly"

  ' Repeat the spreading process a max of five times
  for i=0 to 5

    ' Each time, check that you haven't spread too much
    if (pbThickness > 1.0){

      ' If you have spread too much, stop spreading.
      stopSpreading()
    }

    ' Otherwise
    else

      ' Keep spreading...
      keepSpreading()

    end if

  next i

End Sub

' Define the spreadMore function
Sub SpreadMore():

  ' Use another set of sub-functions to move the knife
  dipIntoPb()
  horizontalShiftKnife()

End Sub
```



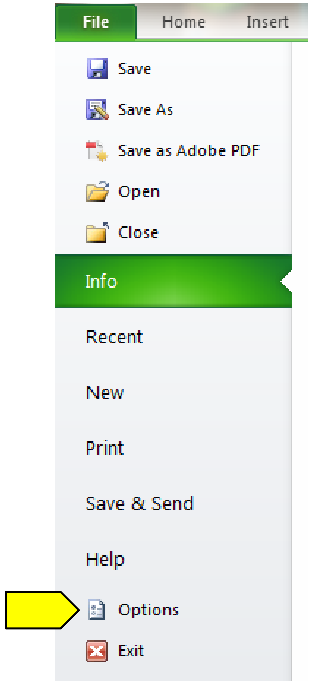
Big Picture!

Coding = creating building blocks and putting them together

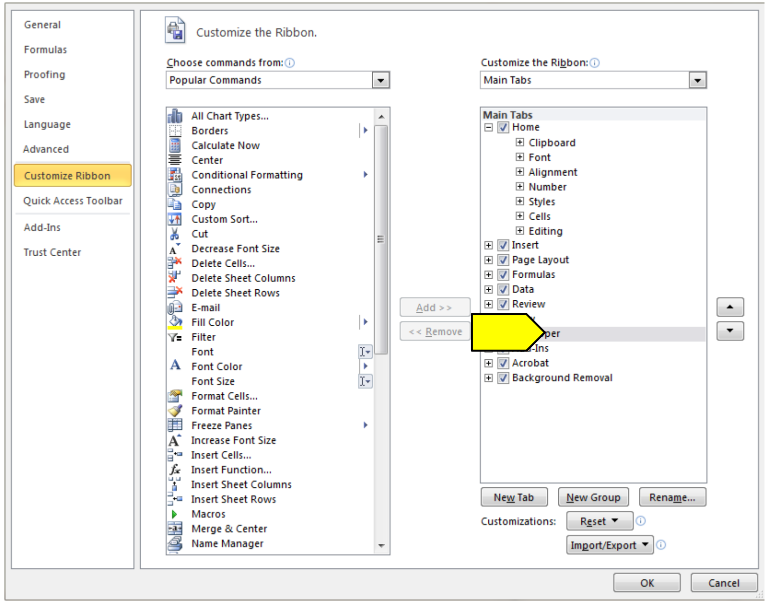
Activity One – Pop Up Messages

Add Developer Tools: Windows

01 Go to **File > Excel Options**.

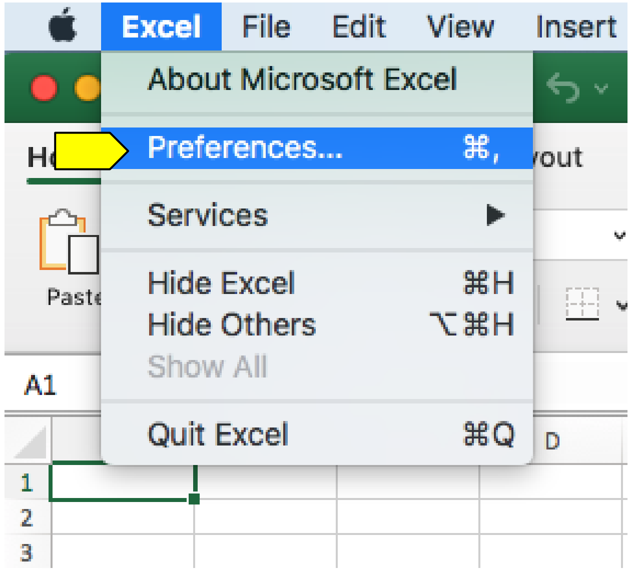


02 Then go to **Customize Ribbon**, choose **Main Tabs** in the right pane, and make sure **Developer** is checked.

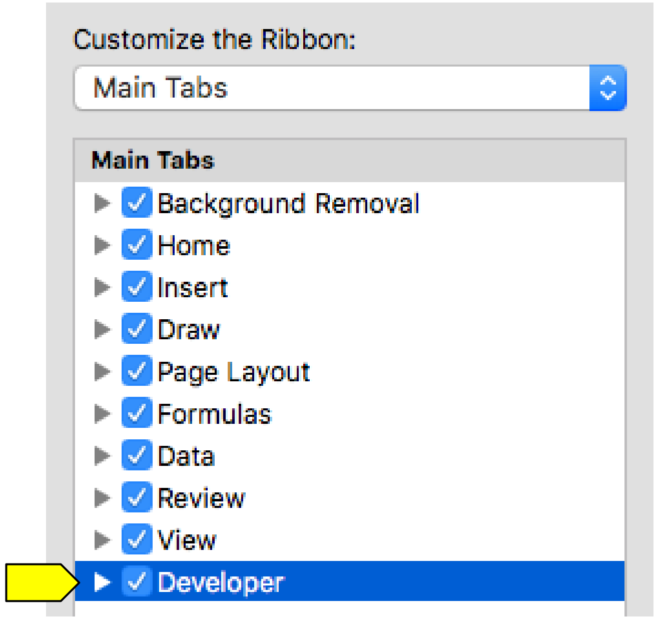


Add Developer Tools: Mac

01 Go to **Excel > Preferences**.



02 Then go to **Ribbon & Toolbar**, select **Main Tabs** in the right pane, and make sure **Developer** is checked.

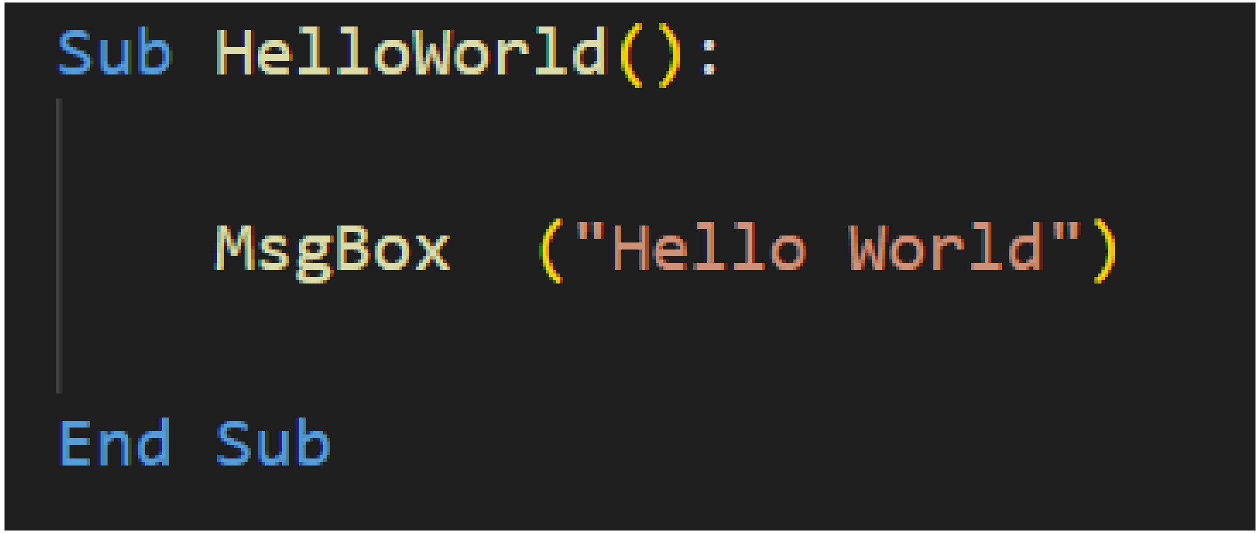

Customize the Ribbon: Main Tabs

Main Tabs
- ✓ Background Removal
- ✓ Home
- ✓ Insert
- ✓ Draw
- ✓ Page Layout
- ✓ Formulas
- ✓ Data
- ✓ Review
- ✓ View
- ✓ Developer

Questions?

Questions?

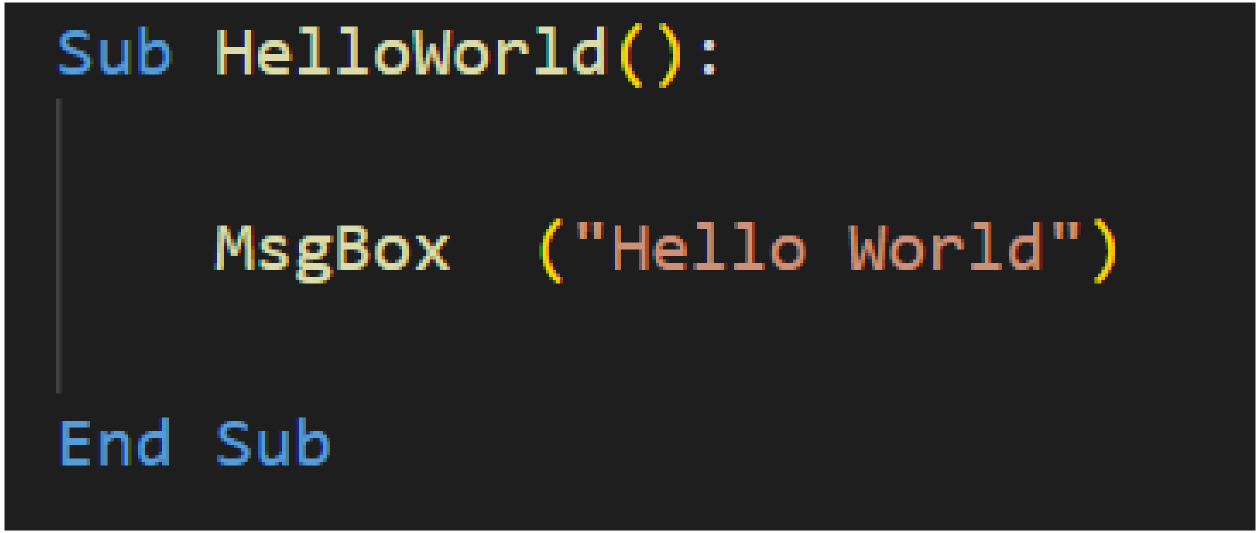
Visual Studio Subroutine Example



```
Sub HelloWorld():


    MsgBox  ("Hello World")


End Sub
```

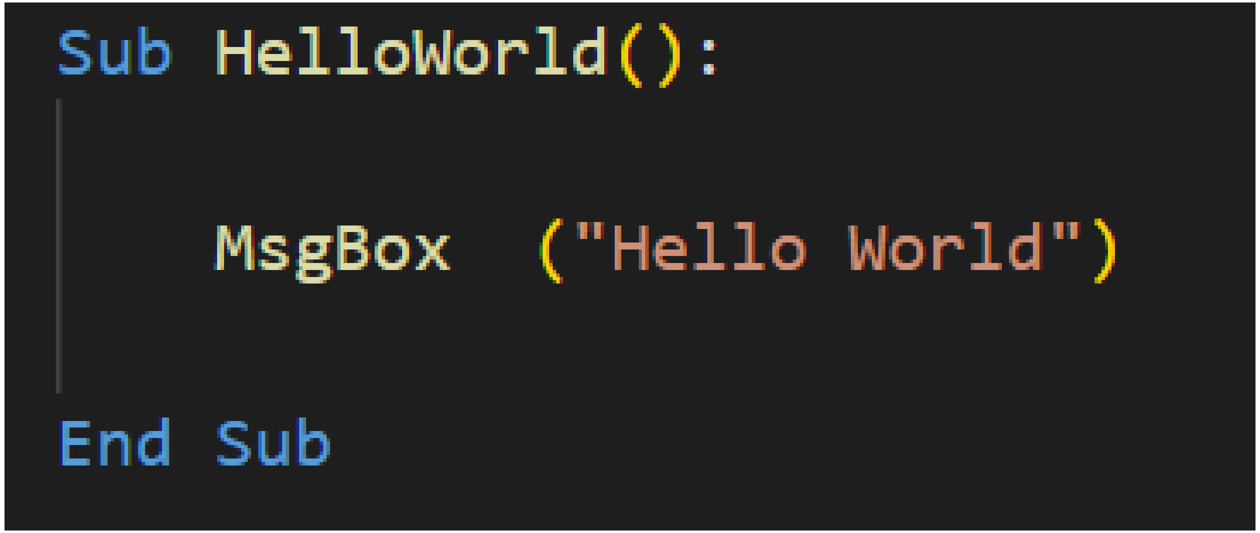
Visual Studio Subroutine Example

Define Function Start and End


```
Sub HelloWorld():


    MsgBox  ("Hello World")


End Sub
```

Visual Studio Subroutine Example

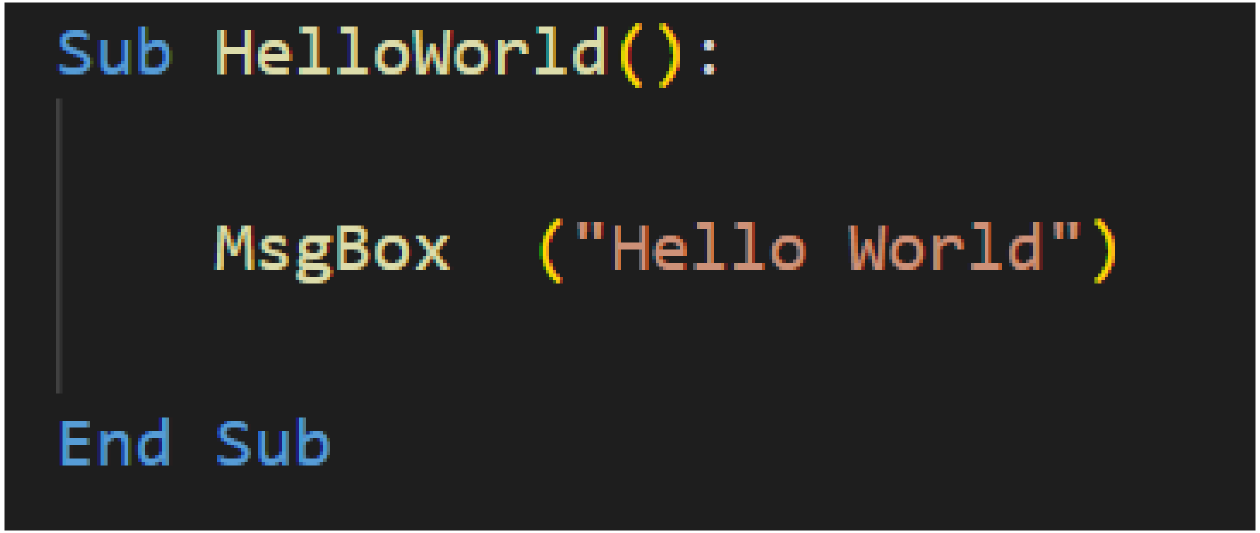
Define Name of the Function



```
Sub HelloWorld():

    MsgBox  ("Hello World")

End Sub
```

Visual Studio Subroutine Example

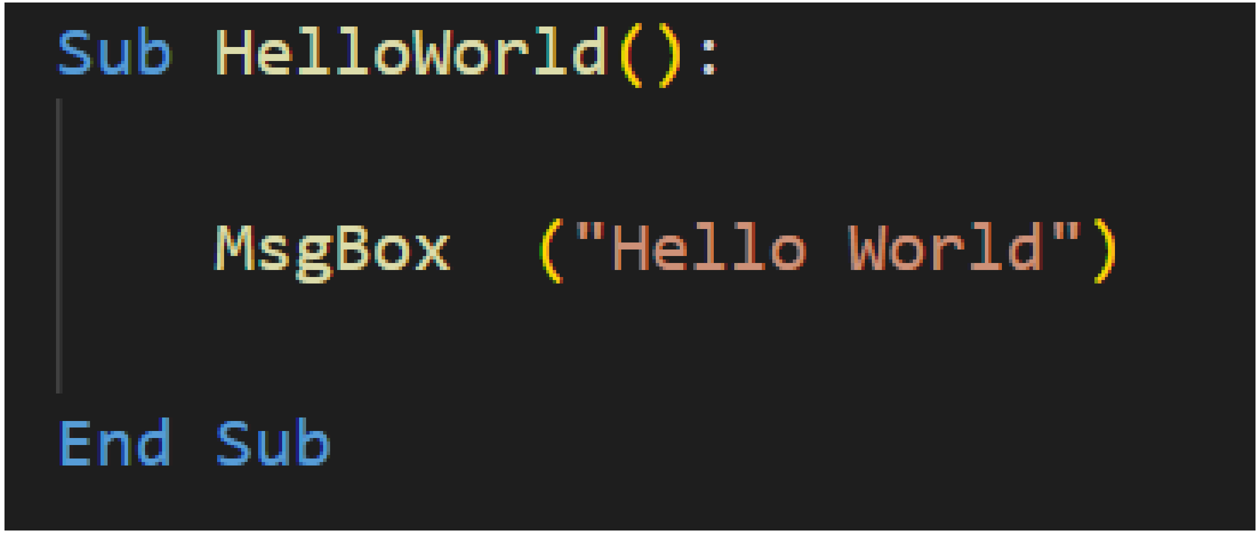
Define Arguments of the Function (None)



```
Sub HelloWorld():

    MsgBox  ("Hello World")

End Sub
```

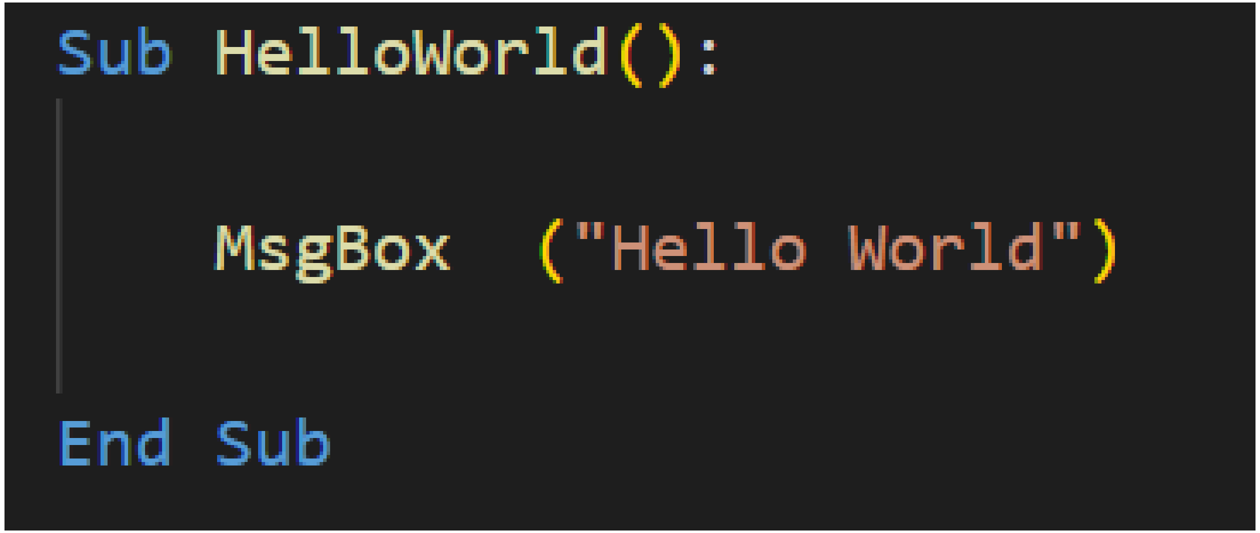
Visual Studio Subroutine Example

Body of the function – Call an inbuilt function called MsgBox



```
Sub HelloWorld():

    MsgBox  ("Hello World")

End Sub
```

Visual Studio Subroutine Example

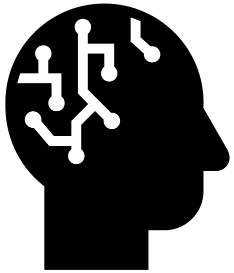


```
Sub HelloWorld():

    MsgBox  ("Hello World")

End Sub
```

With the argument **"Hello World"**

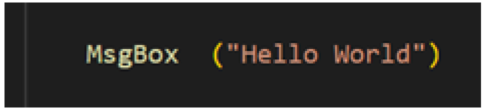
Function Documentation

What does MsgBox do and how do I use it?

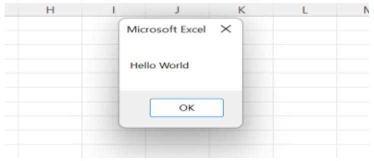


- MsgBox(string_parameter) takes a string parameter and, when run, creates a pop-up window where the body of the text shows string_parameter

- Example



```
MsgBox ("Hello World")
```



Microsoft Excel

Hello World

Student Activity 2: Three Pop-ups in VBA

You have seen how to create a function in VBA that creates a popup in an Excel spreadsheet. Open a new Excel spreadsheet and Create and execute a VBA script that generates three pop-up messages containing any text of your choice.

Suggested Time:

5 minutes

Activity 3 - Buttons

Student Activity 4: Three Pop-ups in VBA

Create an Excel file with two buttons.

For each button, create a different VBA subroutine that will trigger a unique pop-up message when clicked.

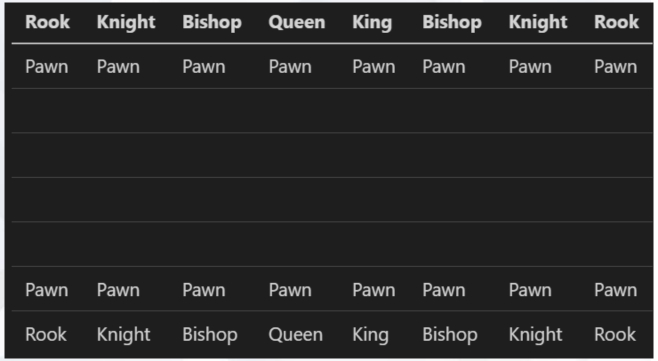
Suggested Time:

10 minutes

Activity 5 – Populating Cells

Student Activity 6: Populate Chessboard with VBA

Open Unsolved Activity 6 file.
Using a VBA Script, load the following values into the cells. Use Range() for the first two rows (the black pieces) and Use Cells() for the last two rows (the white pieces)



Rook	Knight	Bishop	Queen	King	Bishop	Knight	Rook
Pawn	Pawn	Pawn	Pawn	Pawn	Pawn	Pawn	Pawn
Pawn	Pawn	Pawn	Pawn	Pawn	Pawn	Pawn	Pawn
Rook	Knight	Bishop	Queen	King	Bishop	Knight	Rook

Suggested Time:

15 minutes

Break

Activity 7 – Declaring Variables

Student Activity 8: Playing with Variables

Open Unsolved Activity 8 excel sheet and VBA script.
Using a VBA Script, define the appropriate variables for Price, Tax, Quantity, and Total.

Then populate these variables with the appropriate values so that the script works as intended

Suggested Time:

15 minutes

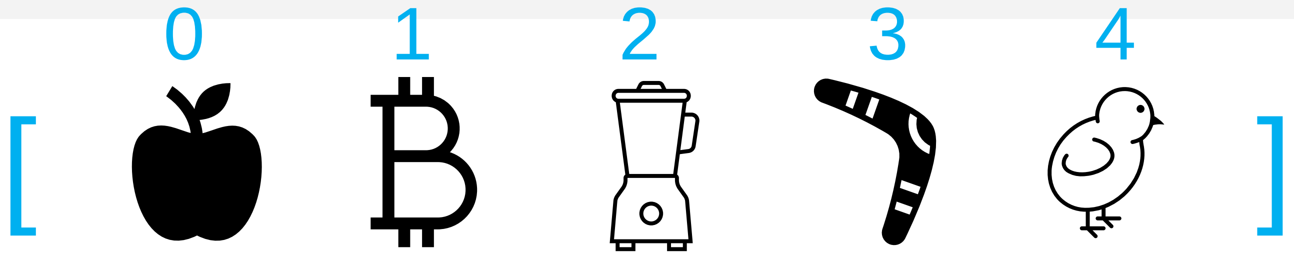
Activities 9 + 10 - Arrays

Arrays are 0 indexed

Arrays are basically **lists of values.** In VBA, arrays must contain values of the same type

Arrays in nearly all programming languages start counting at 0. We call this **0-indexing.**

Rebel Children (1-Indexed): MATLAB, Julia, Mathematica, ***R***

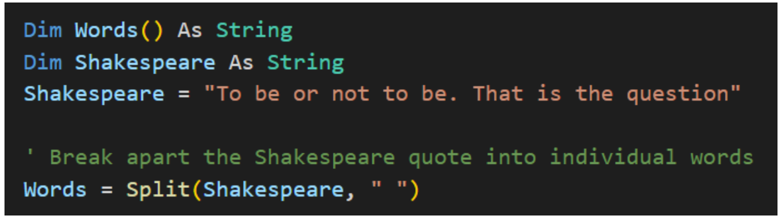


The Split Function

Split(wholeString, delimiter)

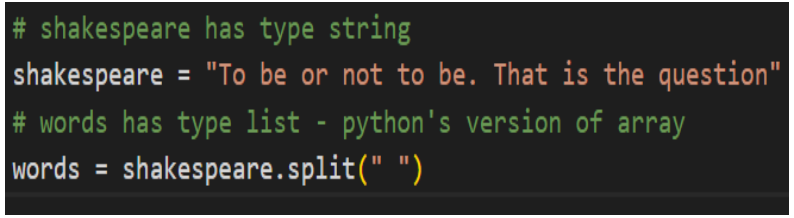
- Splitting is a very common function in most languages that takes an input string and a delimiter and returns an array of substrings chopped up according to that delimiter

VBA Syntax



```
Dim Words() As String
Dim Shakespeare As String
Shakespeare = "To be or not to be. That is the question"

' Break apart the Shakespeare quote into individual words
Words = Split(Shakespeare, " ")
```

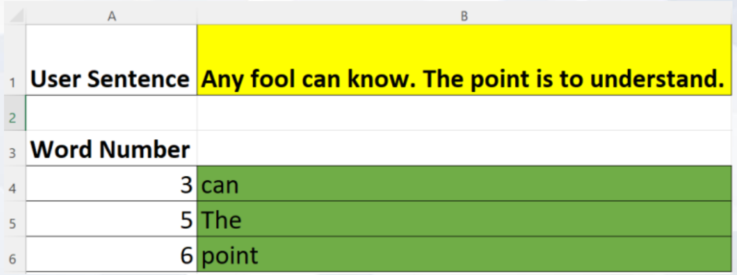
Python Syntax



```
# shakespeare has type string
shakespeare = "To be or not to be. That is the question"
# words has type list - python's version of array
words = shakespeare.split(" ")
```

Student Activity 11: Sentence Breaker

Open the excel workbook and the VBA file in Activity 11/Unsolved. The User will enter a Sentence in the Cell shown in yellow and three "word numbers" in A4, A5, and A6 The script should populate B4:B6 with the nth word in the user sentence as described by word numbers



	A	B
1	User Sentence	Any fool can know. The point is to understand.
2		
3	Word Number	
4	3	can
5	5	The
6	6	point

Suggested Time:

20 minutes

Activity 12 - Conditionals