Homework Notes:

* First look at what the spreadsheet looks like
* Summarize all of this data up until G
* Have to create a summary in VBA coding
* Yearly change – taking the opening value for that stock for the first day, find the closing value for the last day of the stock –>
* Yearly change, percent change
* (N8-N7)/N7 \* 100
  + If your starting value is 0, and you try to divide by 0, you will get an error
* Stock value is sum of all column G
* You will also find the stock with greatest percent and decrease increase in column K, find ticket associated with the value, and the total greatest volume of the associated ticker
* Do intial testing and coding in smaller test data file
* Submit as a google drive link and github submission

**Intro to Programming Logic**

* Conditionals = If this, then that
  + If peanut butter is crunchy, use less
* Iterations = Looping
  + While there is more PB, add more jelly (keep looping thru until a condition is satisfied)
* Functions = Contained pieces of code that can do a lot
  + Spread the PB with a knife
* Variables / Arrays = Base ingredients
  + Bread, pb, and jelly
* Variables
  + The items in our procedure, or the core objects.
  + In VBA, we have to **declare** a variable by using *dim* followed by the type. Then they can be assigned a value.
  + When we declare a variable, the computer is creating an open memory slot that we insert a value into
  + Variable declaration:
    - Dim ing1 as String
      * String = type of variable
      * String = word or words, text
      * Integer = rounded to a single, whole number
    - Dim budget as double
      * Double = a decimal type value, you are allowing for a remainder
  + Variable assignment
    - Ing1 = “peanut butter”
    - Ing2 = “jelly”
    - We designate strings with quotations
    - Budget = 5.00
* Array: A Collection of Items
  + Designate array with square brackets
  + Group of related items. They present another way to store and reference similar pieces of information.
    - [“Peanut Butter”, “Jelly”, “Bread”]
      * PB = item 0, jelly = item 1, bread = item 2
    - VBA starts at 0
    - Dim ingredients (0 to 2) as String
      * Declared an array, then we assign the value to the string with ingredients
      * Creating a 3 slot list or array, and then the dim is declared
      * Ingreditents(0) = “peanut butter”
      * Ingredients(1) = “jelly”
      * Ingredients(2) = “bread”
* Conditionals
  + They can control the flow of logic based on certain conditions being met
  + Most languages = if/else
  + If (pbThickness > 1.0) Then stopSpreading()

Else

stopMore()

EndIf

* Iteration: Concept of using loops to perform a group of tasks repeatedly a number of times
  + For loops = designate the number of times you loop thru something
  + While loops = while condition X is true, keep going with this loop. Once condition is false, it jumps out of the loop and we keep going.
  + Syntax = grammar of the language
  + ‘ Repeat the same step until I become 20

For I = 0 to 20

‘Each time spread more

SpreadMore()

‘Add one to the value of I each time

Next i

* + The first time we run thru this loop, I = 0, the second time, I = 1, and so on. We spread more until I is greater than 20, and it will jump out of this loop and keep going with the code below it.
  + Use “next I” to increment our loop
  + Best practice is to use *I* for iteration
* Comment out your code and take notes with a single apostrophe = ‘
* Functions
  + Sub process of code. They let you create premade, reusable blocks of code.
    - If you want to use a code 3 times.
  + We can create mini programs inside of this major program
  + Function takes an input and spits out an output.

**02 Basic\_For\_Loop**

* Loop iterator would be Dim I as Integer
* For i = 1 to 20
  + We will go thru the loop 20 times
  + We will then go thru the loop until i = 20
  + The number 1 will populate in the column up until the number 20
* Cells(1, i).Value = 5 🡪 First row will have all 5s
* Cells (I, 1).Value = 1 🡪 Column A up until Row 20 will have all ones
  + We stay in Column A and assign values of all 1 for 20 iterations
* Cells(i + 1, 2).Value = i + 1
  + First row will start counting as you go down
  + Start in Column B, take the iteration + 1, and take the value + 1

**03 Chicken Nuggets**

* First step is to set our iterator
* Want to go thru this ten times, so we set I = 10
* First time thru, I = 1
  + Second time thru, I =1, so we iterate down Column A
* In Column B, it will take the iteration, and add 10 to it, and store the value
* In column C, we set all the values in third column for each iteration

**04 Loop Conditionals**

* Modulus = Remainder after you divide = Mod
  + Will tell us if something is evenly divisible by a certain number
* Conditional Loop = repeat steps, ask if something is T/F, if T we do this, if F we do something else

**05 FizzBuzz**

* Check if divisible by 3 AND 5
  + If (num Mod 3 = 0 And num Mod 5 = 0) Then
  + Have to add if its BOTH first, otherwise the loop will break
* Check if number is divisible by just 3
  + ElseIf (num Mod 3 = 0) Then
  + If so, print Fizz
    - Cells(i, 2).Value = “Fizz”
* Check if Number is divisible by just 5
  + ElseIf (num Mod 5 = 0) Then
  + If so, print “Buzz”
    - Cells(i, 2).Value = “buzz”
  + End with: EndIf
  + End with: Next i

**06 Student Lotto**

* Create first and last name strings to make the message box work
* Declare variables as Longs because integers over 35000 VBA will have trouble
  + Dim first\_place as long
  + Dim runner1 as long
* Establish the winning ticket values
  + First\_place = 3957481
* Loop through each of the lotto tickets
  + i = 1 to 1001
    - iterate thru all rows
* Check if the lotto. Number matches the first place winner
  + Set your i as your row value for cells, so you will start at row 1 and change each time
* If so, create a message box specifying the first place win
  + “Congrats” + cells(i, 1).Value
* Retreive the values associated with the winner and put in appropriate cells

**07 NestedForLoops**

* Is are rows and Js are columns
* Loop through rows
  + For i = 1 To 3
* Loop through columns
  + j = 1 to 5

**08 Hornets Nest**

* Count the number of hornets found
  + Do a For Loop Nested because we are going thru rows and columns
  + Intialize the counter to 0 🡪 Dim HornetsCount As Integer
    - HornetsCount = 0
    - We initialize this count outside of our loop.
  + Loop through all rows, then columns
    - For i = 1 To 6
    - For j = 1 to &
    - If Cells(i, j).Value = “Hornets” Then
      * Add to the hornets counter via:
        + HornetsCount = HornetsCount + 1
* Then, we replace Hornets with Bugs
  + Cells(i, j).Value = “bugs”
* Then, show the number of hornets found
  + MsgBox(HornetsCount & “Hornets Found”)
* Number of bugs and bees available and take from that bin instead of incrementing
  + Pull the value out of the cell with BugsCount = Range(“L2”).Value
    - If the equal sign was on the other side, we would be assigning the value, rather than pulling out the value

**Lesson 2.3 Scripting Practice in VBA**

**01 Stars Counter**

* For a nested loop, you need to iterate across rows and columns
* StarCounter = 0
  + Prior to this, the counter was outside of the loop. It is now inside of the loop, which will reset the star counter to 0 every time it goes to a new row
* Every row will check if the value is equal to full star, and if it is, we increment the star counter by 1. If not, we jump out of the End If.
* Every time we jump out of this loop, we take the value of our Star Counter and set the value for it using the (i, 9) reference
* **BONUS**
  + When we want to count the number of rows, we can use:
    - Lastrow = Cells(Rows.Count, 1).End(xlUp).Row
    - The End xlUp 🡪 goes from the bottom of the worksheet and counts the number of rows up to the first value that it finds. Calculates backwards to find where the last value in a row is.
    - For i = 2 to lastrow

**02 Formatter**

* Interior = the color fill
* ColorIndex = the color

**03 Gradebook**

* Based on Column B, we conditionally format C and D
* >= 90 and <100 would create bounds if you wanted

**04 Checkerboard**

* Need to set up two For Loops
* If you are on an even row, you’re not changing it, but if you are on an odd number row, it would be changed

1. **Next Cells**- Purpose is to iterate over a column and determine when we encounter a new value. We are trying to identify boundaries between similar values.The cell will remain in the same color until it reaches a new value.

* For homework: When ticker changes the value
* Specify a column and iterate through a single column
* Check if the value + 1 in our column, or the next value, is not equal to the value of the current iteration.
* Checking the one ahead of it and the current i, and will flag with i + 1 <> i

1. **Credit Card Checker**

* Think about the variables you need to keep track of
  + Current card
  + Current total on the card
  + Number to indicate which row you want to print results on
  + I need to keep track of the total per CC brand, so I need to somehow use this next cell value to determine where the boundary is.
  + Keep accumulating the sum until a condition is met (check for the CC boundary – the final CC brand)
  + You also have to create a summary table
* Intialize brand total to 0
* Create a variable to tell us where we are in the summary table
* Every time we don’t find a change, we add to the value of the current iteration

**07 Wells Fargo 01**

* We can reference each worksheet by calling For Each ws in Worksheets
* Split(WorksheetName, “\_”) creates an array from a string, and then we have to reference the first element of the array
* This is where you can figure out how to find the last column. This is how you find the boundary of our table

**08 Wells Fargo 02**

**Extra**

**Homework Notes:**

1. Activity 6 shows ways of doing cell referencing (Range, Cells)
2. Activity 7 shows an example of using the Long data type.  There is also a link in TTh about the data types and their limits.  We said an integer max value goes a bit above 32,000.  If you go over 32,000 in your code then it will give you an error about data size.
3. Activity 8 shows how to get values from the sheets and set them as a variable and it shows how to write values/calculations/strings back to a cell in a sheet.
4. Activity 12 shows if statements so it tells you how to compare two values and then execute code only for that situation.
5. Also when dealing with decimals there is the data type Single and Double.  Double works for a larger set of numbers.  If you use single then you could get an error if you exceed its limit.
6. I posted a link in TTh about Range(...).Interior.ColorIndex = #  color codes. I think there is an example of this in Activity 6 where the chessboard black and white squares are created.
7. Please note:  all vba code needs to be saved to a .xlsm or your vba script you wrote will be deleted if saved as another format like .xls.

 Create a script that will loop through all the stocks for one year and output the following information.

* The ticker symbol.
* Yearly change from opening price at the beginning of a given year to the closing price at the end of that year.
* The percent change from opening price at the beginning of a given year to the closing price at the end of that year.
* The total stock volume of the stock.

 You should also have conditional formatting that will highlight positive change in green and negative change in red.

**CHALLENGES**

1. Your solution will also be able to return the stock with the "Greatest % increase", "Greatest % decrease" and "Greatest total volume". The solution will look as follows:
2. Make the appropriate adjustments to your VBA script that will allow it to run on every worksheet, i.e., every year, just by running the VBA script once.

**Other Considerations**

* Use the sheet alphabetical\_testing.xlsx while developing your code. This data set is smaller and will allow you to test faster. Your code should run on this file in less than 3-5 minutes.
* Make sure that the script acts the same on each sheet. The joy of VBA is to take the tediousness out of repetitive task and run over and over again with a click of the button.