Module 2 Challenge

New Attempt

Due Jan 17 by 12:59am **Points** 100 **Submitting** a text entry box or a website url

Background

Steve loves the workbook you prepared for him. At the click of a button, he can analyze an entire dataset. Now, to do a little more research for his parents, he wants to expand the dataset to include the entire stock market over the last few years. Although your code works well for a dozen stocks, it might not work as well for thousands of stocks. And if it does, it may take a long time to execute.

In this challenge, you'll edit, or **refactor**, the Module 2 solution code to loop through all the data one time in order to collect the same information that you did in this module. Then, you'll determine whether refactoring your code successfully made the VBA script run faster. Finally, you'll present a written analysis that explains your findings.

Refactoring is a key part of the coding process. When refactoring code, you aren't adding new functionality; you just want to make the code more efficient—by taking fewer steps, using less memory, or improving the logic of the code to make it easier for future users to read. Refactoring is common on the job because first attempts at code won't always be the best way to accomplish a task. Sometimes, refactoring someone else's code will be your entry point to working with the existing code at a job.

What You're Creating

This new assignment consists of one technical deliverable and a written report to deliver your results. You will submit the following:

- Deliverable 1: Refactor VBA code and measure performance
  - This deliverable will include an updated workbook and a folder with PNGs of the pop-ups with script run time
- Deliverable 2: A written analysis of your results (README.md)

Files

Use the following link to download the Challenge starter code.

Download challenge_starter_code.vbs (https://2u-data-curriculum-team.s3.amazonaws.com/dataviz-online/module_2/challenge_starter_code.vbs)

Deliverable 1: Refactor VBA Code and Measure Performance (80 points)

Deliverable 1 Instructions

Use your knowledge of VBA and the starter code provided in this Challenge to refactor the `Module2_VBA_Script` so you loop through the data one time and collect all of the information. Your refactored code should run faster than it did in this module.

REWIND

For this deliverable, you've already done the following as part of Module 2:

- **Lesson 2.1.4**: Create a variable with single or long data types.
- **Lesson 2.2.3**: Write `for` loops.
- **Lesson 2.2.3**: Write `if-then` statements.
- **Lesson 2.2.3**: Use design patterns.
- **Lesson 2.2.4**: Use logical and comparison operators.
- **Lesson 2.3.2**: Use an index to access data in an array.
- **Lesson 2.3.2**: Use nested loops.
- **Lesson 2.3.3**: Reuse code.
- **Lesson 2.3.3**: Debug and comment on code.
- **Lesson 2.4.1**: Use visual and numeric formatting.
- **Lesson 2.4.2**: Use conditional formatting.
- **Lesson 2.5.3**: Measure code performance.

1. Download the `challenge_starter_code.vbs` file and rename it `VBA_Challenge.vbs`.
2. Create a folder called "Resources" to hold the run-time pop-up messages that you'll screenshot after running refactored analyses for 2017 and 2018.
3. Rename the `green_stocks.xlsm` file that you used in this module as `VBA_Challenge.xlsm`.
4. Add the `VBA_Challenge.vbs` script to the Microsoft Visual Basic editor.

5. Use the steps below to add code where indicated by the numbered comments in the starter code file.

Step 1a:

- Create a `tickerIndex` variable and set it equal to zero before iterating over all the rows. You will use this `tickerIndex` to access the correct index across the four different arrays you'll be using: the tickers array and the three output arrays you'll create in Step 1b.

Step 1b:

- Create three output arrays: `tickerVolumes`, `tickerStartingPrices`, and `tickerEndingPrices`.
  - The `tickerVolumes` array should be a `Long` data type.
  - The `tickerStartingPrices` and `tickerEndingPrices` arrays should be a `Single` data type.

Step 2a:

- Create a `for` loop to initialize the `tickerVolumes` to zero.

Step 2b:

- Create a `for` loop that will loop over all the rows in the spreadsheet.

Step 3a:

- Inside the `for` loop in Step 2b, write a script that increases the current `tickerVolumes` (stock ticker volume) variable and adds the ticker volume for the current stock ticker.
  - Use the `tickerIndex` variable as the index.

If you'd like a hint on how to increase the current `tickerVolumes` by using the `tickerIndex` variable as the index, that's totally okay. If not, that's great too. You can always revisit this later if you change your mind.

SHOW HINT

Step 3b:

- Write an `if-then` statement to check if the current row is the first row with the selected `tickerIndex`. If it is, then assign the current starting price to the `tickerStartingPrices` variable.

Step 3c:

- Write an `if-then` statement to check if the current row is the last row with the selected `tickerIndex`. If it is, then assign the current closing price to the `tickerEndingPrices` variable.

Step 3d:

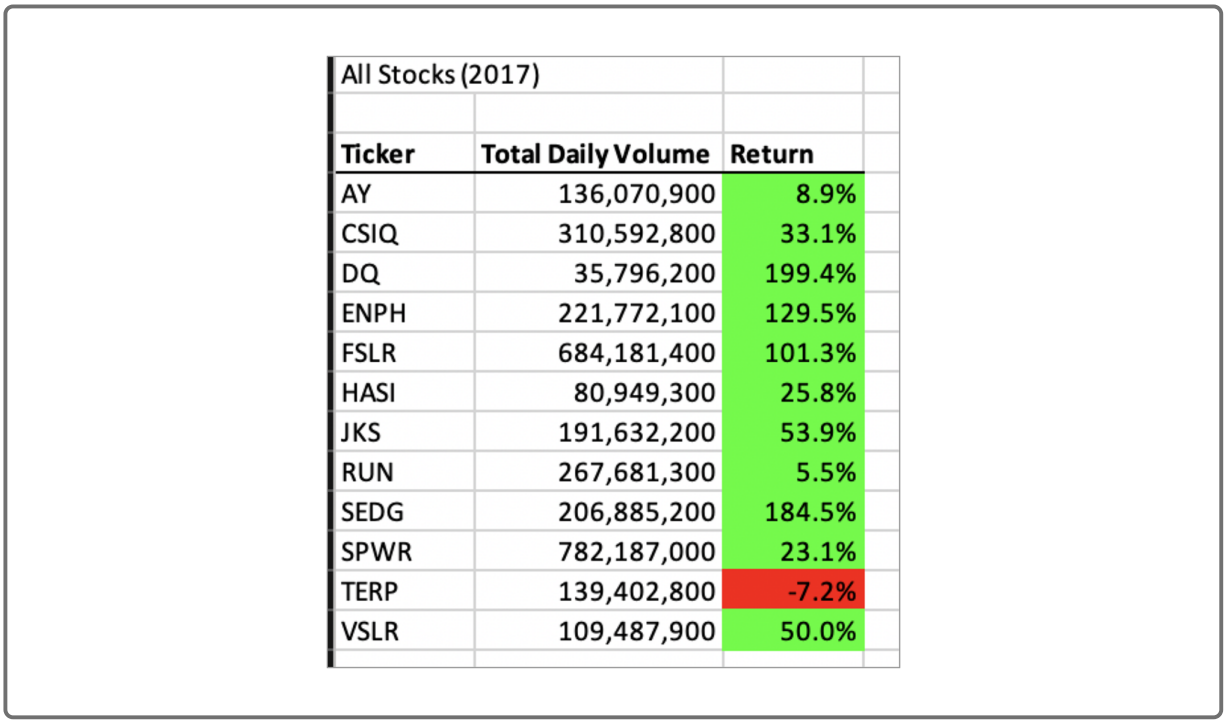
- Write a script that increases the `tickerIndex` if the next row's ticker doesn't match the previous row's ticker.

Step 4:

- Use a `for` loop to loop through your arrays (`tickers`, `tickerVolumes`, `tickerStartingPrices`, and `tickerEndingPrices`) to output the "Ticker," "Total Daily Volume," and "Return" columns in your spreadsheet.

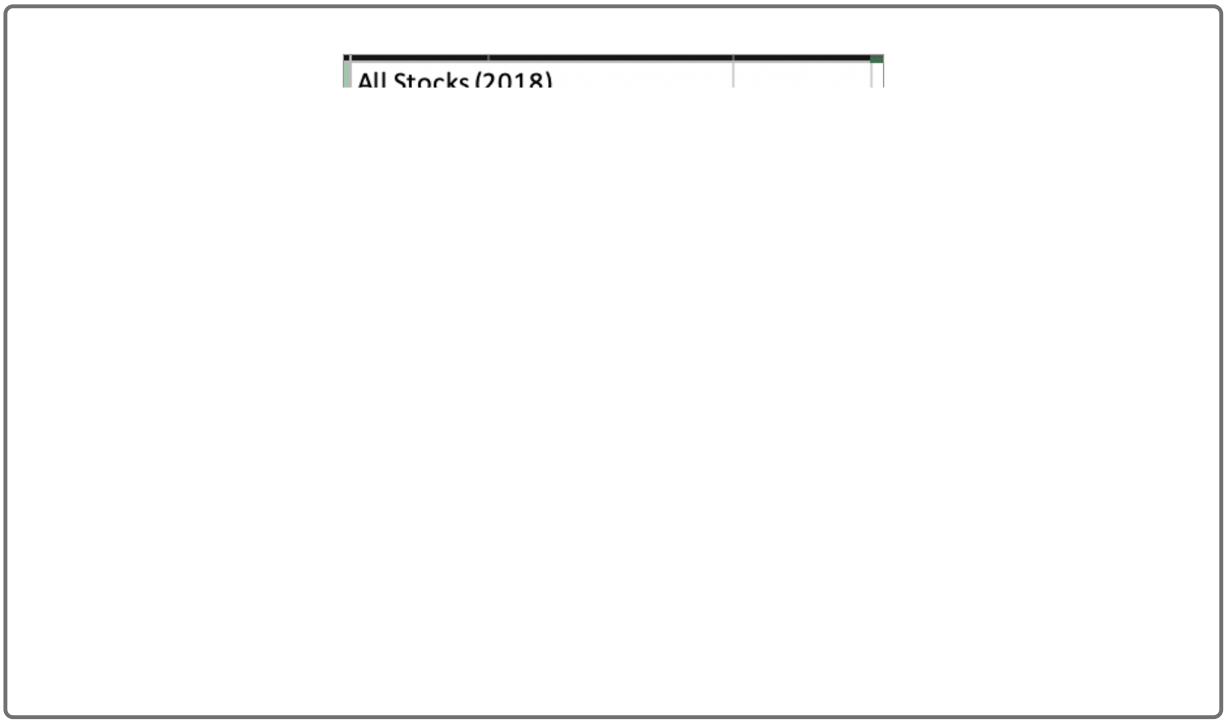
Finally, run the stock analysis, then confirm that your stock analysis outputs for 2017 and 2018 are the same as they were in the module (as

shown in the images below). In your Resources folder, save the pop-up messages showing elapsed run time for the refactored code as `VBA_Challenge_2017.png` and `VBA_Challenge_2018.png`. Then, save the changes to your workbook.



All Stocks (2017)

Ticker	Total Daily Volume	Return
AY	136,070,900	8.9%
CSIQ	310,592,800	33.1%
DQ	35,796,200	199.4%
ENPH	221,772,100	129.5%
FSLR	684,181,400	101.3%
HASI	80,949,300	25.8%
JKS	191,632,200	53.9%
RUN	267,681,300	5.5%
SEDG	206,885,200	184.5%
SPWR	782,187,000	23.1%
TERP	139,402,800	-7.2%
VSLR	109,487,900	50.0%



All Stocks (2018)

Deliverable 1 Requirements

You will earn a perfect score for Deliverable 1 by completing all requirements below:

- The `tickerIndex` is set equal to zero before looping over the rows. **(5 pt)**.
- Arrays are created for `tickers`, `tickerVolumes`, `tickerStartingPrices`, and `tickerEndingPrices` **(15 pt)**.
- The `tickerIndex` is used to access the stock ticker index for the `tickers`, `tickerVolumes`, `tickerStartingPrices`, and `tickerEndingPrices` arrays **(15 pt)**.
- The script loops through stock data, reading and storing all of the following values from each row: `tickers`, `tickerVolumes`, `tickerStartingPrices`, and `tickerEndingPrices` **(25 pt)**.
- Code for formatting the cells in the spreadsheet is working **(5 pt)**.
- There are comments to explain the purpose of the code **(5 pt)**.
- The outputs for the 2017 and 2018 stock analyses in the `VBA_Challenge.xlsm` workbook match the outputs from the AllStockAnalysis in the module **(5 pt)**.
- The pop-up messages showing the elapsed run time for the script are saved as `VBA_Challenge_2017.png` and `VBA_Challenge_2018.png` **(5 pt)**.

Deliverable 2: Written Analysis of Results (20 points)

Deliverable 2 Instructions

Initialize your repository with a README, and write your analysis of Deliverable 1.

NOTE

> This **documentation (https://help.github.com/en/articles/basic-writing-and-formatting-syntax)** provides more information about basic writing and formatting syntax for GitHub.

For your written analysis, be sure to use complete and coherent sentences. Your written analysis should contain three sections, which cover the following:

1. Overview of Project: Explain the purpose of this analysis.
2. Results: Using images and examples of your code, compare the stock performance between 2017 and 2018, as well as the execution times of the original script and the refactored script.
3. Summary: In a summary statement, address the following questions.
    1. What are the advantages or disadvantages of refactoring code?
    2. How do these pros and cons apply to refactoring the original VBA script?

Deliverable 2 Requirements

Structure, Organization, and Formatting Requirements (8 points)

The written analysis contains the following structure, organization, and formatting:

- There is a title, and there are multiple paragraphs **(2 pt)**.
- Each paragraph has a heading **(2 pt)**.

- There are subheadings to break up text **(2 pt)**.
- Links are working, and images are formatted and displayed where appropriate **(2 pt)**.

Analysis Requirements (12 points)

The written analysis has the following:

- Overview of Project
  - The purpose and background are well defined **(2 pt)**.
- Results
  - The analysis is well described with screenshots and code **(4 pt)**.
- Summary
  - There is a detailed statement on the advantages and disadvantages of refactoring code in general **(3 pt)**.
  - There is a detailed statement on the advantages and disadvantages of the original and refactored VBA script **(3 pt)**.

Submission

Once you're ready to submit, make sure to check your work against the rubric to ensure you are meeting the requirements for this Challenge one final time. It's easy to overlook items when you're in the zone!

As a reminder, upload the following deliverables to your stock-analysis GitHub repository:

- Deliverable 1: Refactor VBA code and measure performance
  - The `VBA_Challenge.xlsm` workbook

    - The Resources folder with `VBA_Challenge_2017.png` and `VBA_Challenge_2018.png`

- Deliverable 2: A written analysis of your results (README.md)

To submit your challenge assignment, click Submit, then provide the URL of your stock-analysis GitHub repository for grading. Comments are disabled for graded submissions in BootCampSpot. If you have questions about your feedback, please notify your instructional staff or the Student Success Manager. If you would like to resubmit your work for an improved grade, you can use the **Re-Submit Assignment** button to upload new links. You may resubmit up to 3 times for a total of 4 submissions.

IMPORTANT

> Once you receive feedback on your Challenge, make any suggested updates or adjustments to your work. Then, add this week's Challenge to your professional portfolio.

NOTE

> You are allowed to miss up to two Challenge assignments and still earn your certificate. If you complete all Challenge assignments, your lowest two grades will be dropped. If you wish to skip this assignment, click Next, and move on to the next Module.

Module-2 Rubric

Criteria	Ratings					Pts
Deliverable 1: Refactor VBA Code and Measure Performance.	**80 to >71.0 pts Demonstrating Proficiency** ✓The tickerIndex is set to equal to zero before looping over the rows. ✓Arrays are created for tickers and ALL output arrays. ✓The tickerIndex is used to access the stock ticker index for the tickers array and ALL THREE output arrays. ✓The script loops through stock data, reading and storing ALL of the following values from each row: tickers, volumes, starting prices, ending prices. ✓Code for formatting the cells in the spreadsheet is working. ✓The output for the 2017 and 2018 stock analyses match the outputs from the AllStockAnalysis. ✓The pop-up messages are saved as PNGs.	**71 to >67.0 pts Approaching Proficiency** ✓The tickerIndex is set to equal to zero before looping over the rows. ✓Arrays are created for tickers and TWO of the THREE output arrays. ✓The tickerIndex is used to access the stock ticker index for the tickers array and ALL THREE output arrays. ✓The script loops through stock data, reading and storing THREE of the following values from each row: tickers, volumes, starting prices, ending prices. ✓Code for formatting the cells in the spreadsheet is working. ✓The output for the 2017 and 2018 stock analyses match the outputs from the AllStockAnalysis. ✓The pop-up messages are saved as PNGs.	**67 to >59.0 pts Developing Proficiency** ✓The tickerIndex is set to equal to zero before looping over the rows. ✓Arrays are created for tickers and TWO of the THREE output arrays. ✓The tickerIndex is used to access the stock ticker index for the tickers array and TWO output arrays. ✓The script loops through stock data, reading and storing TWO of the following values from each row: tickers, volume, starting prices, ending prices. ✓Code for formatting the cells in the spreadsheet is working. ✓The output for the 2017 and 2018 stock analyses match the outputs from the AllStockAnalysis. ✓The pop-up messages are saved as PNGs.	**59 to >0.0 pts Emerging** ✓The tickerIndex is set to equal to zero after looping over the rows. ✓Arrays are created for tickers and ONE of the THREE output arrays. ✓The tickerIndex is used to access the stock ticker index for the tickers array and ONE output array. ✓The script loops through stock data, reading and storing ONE of the following values from each row: tickers, volume, starting prices, ending prices. ✓Code for formatting the cells in the spreadsheet is working. ✓The output for the 2017 and 2018 stock analyses match the outputs from the AllStockAnalysis. ✓The pop-up messages are saved as PNGs.	**0 pts Incomplete**	80 pts

Criteria	Ratings					Pts
Deliverable 2: Structure, Organization, and Formatting.	**8 to >7.0 pts Demonstrating Proficiency** ✓The written analysis has a title and multiple paragraphs. ✓Each paragraph has a heading and subheadings. ✓Links are working, images and code are correct and displayed where appropriate.	**7 to >6.0 pts Approaching Proficiency** ✓The written analysis has a title and multiple paragraphs. ✓Each paragraph has a heading and subheadings. ✓Some links are not working, AND some code is displayed where appropriate.	**6 to >4.0 pts Developing Proficiency** ✓The written analysis has a title and multiple paragraphs. ✓Each paragraph has a heading, and there may be subheadings for each paragraph, OR links are working, images and code are displayed where appropriate.	**4 to >0.0 pts Emerging** ✓The written analysis has a title and multiple paragraphs. ✓There may be a heading OR subheadings for each paragraph, OR all links are working, images and code are displayed where appropriate.	**0 pts Incomplete**	8 pts
Deliverable 2: Analysis	**12 to >11.0 pts Demonstrating Proficiency** ✓The Deliverable Fulfills "Emerging" Required Criteria AND has the following: ✓There is a detailed summary of the pros and cons of refactoring code. ✓There is a detailed summary of the pros and cons of the original and refactored VBA script.	**11 to >8.0 pts Approaching Proficiency** ✓The Deliverable Fulfills "Emerging" Required Criteria AND has at least ONE of the following: ✓There is a detailed summary of the pros and cons of refactoring code OR there is a key point missing. ✓There is a detailed summary of the pros and cons of the original and refactored VBA script OR there is a key point missing.	**8 to >6.0 pts Developing Proficiency** ✓The Deliverable Fulfills "Emerging" Required Criteria AND has at least ONE of the following: ✓The summary of refactoring code is missing key points. ✓The summary of the original and refactored VBA script is missing key points.	**6 to >0.0 pts Emerging** ✓The purpose and background are well defined. ✓The analysis is well described with screenshots and code.	**0 pts Incomplete**	12 pts
						Total Points: 100