# Homework 4: Stock Market Analysis (Google)

**Total points: 50pts**

Time due: November 30th. Tuesday.

Topics include: file I/O, lists, functions, reading data from Internet.

No copy solutions from the Internet.

# Introduction

In this project, we want to do some preliminary data mining (<https://en.wikipedia.org/wiki/Data_mining>) to the prices of Google stock. Your program will calculate the monthly average prices of Google from 2014-2018, and tell us the 6 best and 6 worst months for Google in each of the period.

# Project Specifications

1. You can download the historical stock data of Google from this link:

<https://www.stock-data-solutions.com/kb/how-to-load-historical-prices-from-google-finance-to-csv.htm#Downloading_Historical_Prices_from_Google_Finance_using_Market_csv>

You can download the .csv file or use the tools suggests in the link above.

We will try to do some DYI in this project. You can choose to directly work with internet data using Python libraries or directly download the .csv to your disk.

You can also use webcsv:

<https://www.stock-data-solutions.com/docs/webcsv.htm>

If you don’t want to mess up with the internet, for this assignment, a historical table.csv is given to you as Google stock 2004-2008. It is your job to mine the data for 2014-2018. You can find the Google stock between 2014-2018 here:

<https://finance.yahoo.com/quote/GOOG/history?p=GOOG>

You need to go to Historical Data and Then specify the time range using the dropdown menu:

Graphical user interface

Description automatically generated

1. A template program will be given to you, which is used for your program's frame. There are three functions with their simple descriptions, and you are required to pad them to make them work correctly.

a. get\_data\_list(FILE\_NAME)

In this function, you are required to read the file of stock's historical prices. You should use FILE\_NAME instead of hard code “table.csv” in this function, that way if we wanted to use a different table at any time we could just change the call to the function and not have to change the function itself. After reading each line, you will split it into a list, and append this list to another main list, suppose its name is “data\_list”. So, data\_list is a list of lists, a.k.a. 2-D list. At the end of this function,

return data\_list.

b. get\_monthly\_averages(data\_list)

In this function, you will use the data\_list generated by the first function as the parameter. Use Date, Volume, and Adj Close to calculate the average monthly prices.

How to calculate the average price? Suppose one day's volume and close price are V1 and C1 respectively, then that day’s total sales equals V1 \* C1. We will use the “Volume” column for the day’s volume and the “Adj. Close” column for the day’s close. Now suppose another day's volume and close price are V2 and C2. The average of these 2 days is the sum of the total sales divided by the total volume.

So, the average price of these two days is calculated in this way:

average price = (V1\*C1 + V2\*C2) / (V1 + V2)

To average a whole month you just add up the total sales (V\*C) for each day and divide by the sum of all the volumes (V1 + V2 + … + Vn)

For each month create a tuple with 2 items, the average for that month, and the date (you only need the month and year). Append the tuple for each month to a list (e.g. monthly\_averages\_list), and after calculating all the monthly averages, return this list. We use tuples here because once these values are calculated we don’t want to accidentally change them!

c. print\_info(monthly\_averages\_list)

In this function, you need to use the list of monthly averages calculated in the 2nd function. Here you will need to find and print (to a file) the 6 best (highest average price) and 6 worst (lowest average price) months for Google stock. You will print to a file named “monthly\_averages.txt”. You will first print a header like “6 best months” and then print the 6 best months, 1 month per line, from highest price to lowest, in the following format: month-year, average\_price (to 2 decimal places).

You will then print a blank line and then another header like “6 worst months” and print the 6 worst months, 1 month per line from lowest price to highest, in the same format as for the best months.

Sample output can be found below.

This function does not return anything

3. If you don't call these functions, they are useless. Thus, you should write codes to call them. The steps are already given in the template program.

## Bonus (10pts)

1. (5pts) Use the same method to output the same results for your favorite stock (e.g. Apple or S&P500). You can download from yahoo, finance. Findout the worst 6 months and best 6 months of Appl and S&P 500 between November 2020-Noveber 2021.

You can find the historical data here:

<https://finance.yahoo.com/quote/AAPL/history?p=AAPL>

<https://finance.yahoo.com/quote/%5EGSPC/history/>

2. (5tps) Using Python library numpy/matplotlib libraries to plot the trending of the stocks in the last 6 months. Tutorials on Matplotlib is here. You have to check if you have these packages installed. If you use PyCharm, you just have to link these libraries \*(ask me if you don’t know how). If you have installed Anaconda, you should have all of them.

<https://matplotlib.org/users/pyplot_tutorial.html>



# Deliverables

1. Your code should output the time/year and value of the 6 best and 6 worst stocks for Goolge in the period 2014-2018/10
2. Submit your .py and .csv or .txt file and zip it to Canvas.