CS 101
Spring 2016
Program Assignment # X - Stock Market
Algorithm Due : Sunday, April 5, 2016
Program Due : Sunday, April 10, 2016

Stock Market Data

People have been trying to beat the stock market ever since the beginning of the market. It's intriguing to look back at stocks that made it big to think "what if". You've been provided with a file that has a list of our available stock indexes for this project, along with the stock history for each company. You are tasked with writing a program that asks the user what stock index they want a history for, the date the stock was purchased, the date the stock was sold and how many stocks were initially purchased. The purchase price will be the open price and the sell price will be the closing price on the end date. Stocks can also split, if you have 100 stocks and it splits 2:1 you end up with 200 stocks for that company. This is a pretty large program with lots of pieces begging to be split into functions that make it easier to test each portion. Divide and conquer.

stocklist.csv

Stock	Name
AAPL	Apple Inc. (AAPL)
ACN	Accenture plc.
BIDU	Baidu, Inc.
CGI	Celadon Group, Inc.
GOOGL	Google Inc.
INFN	Infinera Corporation
MSFT	Microsoft Corporation
TSLA	Tesla Motors, Inc.

The stocklist.csv file provided is a comma separated value file. The header row identifies the columns. The first column is the stock index, and the second is the company name. The company name will be used in our final output, but the stock index name is how we will lookup the stock. Each company has a csv file that begins with the index name and then .csv.

{index}.csv - Sample shown is aapl.csv

Date	Open	High	Low	Close	Volume	Ex-Dividend	Split Ratio
12/12/1980	28.75	28.87	28.75	28.75	2093900	0	1
12/15/1980	27.38	27.38	27.25	27.25	785200	0	1
12/16/1980	25.37	25.37	25.25	25.25	472000	0	1
12/17/1980	25.87	26	25.87	25.87	385900	0	1
12/18/1980	26.63	26.75	26.63	26.63	327900	0	1
12/19/1980	28.25	28.38	28.25	28.25	217100	0	1

There is a csv file for each company index. AAPL, GOOGL, etc. The header row contains the column information. The date column is the date for this record. Open is the price for the stock when the market opened, high is the highest price of the day, low is the lowest price of the day, the close is the price it closed at. Volume and Ex-Dividend will not be used. High and Low actually won't' be used for our purposes either. The Split Ratio is the stock split. If the split ratio is 2 then the stock doubles prior to opening. If it's 5 then you get 5 stocks for each stock you own. If it's 0.5 then you lose half your stocks.

Examples

If you look at the BIDU data for the date around 5/12/2010 you'll see these rows.

5/7/2010	666.89	671.23	630	639.49	1705800	0	1
5/10/2010	667.45	694.79	667.44	694.78	1721000	0	1
5/11/2010	684.98	716.67	681.22	714.17	2093300	0	1
5/12/2010	74.26	78.36	72.82	78.21	41843500	0	10
5/13/2010	80.92	82.29	75	75.64	58873900	0	1
5/14/2010	73.92	74.81	72.28	73.98	19574000	0	1
5/17/2010	74.14	75.1	71.64	73.18	15382100	0	1

You'll notice that the stock split 10:1 on 5/12/2010. This occurred before the open of the market. So if you bought one stock on 5/12/2010 and sold it on 5/13/2010 you would get it at

$74.26. That one stock would still be one stock on 5/13/2010 which you'd sell at the close price of $75.64. Which would net you a profit of $1.38. Not bad for one day.

If you purchase that one stock of BIDU on 5/11/2010 and sold on 5/12/2010, you would purchase it at $684.98. Before the market opens on 5/12 the stock splits 10 to 1, so you'd end up having 10 stocks, which sold at the closing price of $78.21 per share would make you $97.12.

If you purchase one BIDU stock on 5/11/2010 and sell it on 5/13/2010, the purchase price would be $684.98 and the sell price would be $75.63, but the stock would have split on the 12th. On a large date range your stock may split several times and you'll have to keep track of that as you read through the file.

Below shows the program results for the previous examples

```
Enter the name of the stock purchased.  Enter quit to exit ==> bidu
Enter the stock purchase date ==> 5/12/2010
Enter the date you sold the stock ==> 5/13/2010

How many stocks were purchased on start date ==> 1

Our Baidu, Inc. (bidu) Portfolio
Action            Date         Shares          Price    Total Price
=================================================================
Buy         05/12/2010            1.0          74.26          74.26
Sold        05/13/2010            1.0          75.64          75.64
=================================================================
                                                              1.38

Enter the name of the stock purchased.  Enter quit to exit ==> bidu
Enter the stock purchase date ==> 5/11/2010
Enter the date you sold the stock ==> 5/12/2010

How many stocks were purchased on start date ==> 1

Our Baidu, Inc. (bidu) Portfolio
Action            Date         Shares          Price    Total Price
=================================================================
Buy         05/11/2010            1.0         684.98         684.98
Sold        05/12/2010           10.0          78.21         782.10
=================================================================
                                                             97.12
```

```
Enter the name of the stock purchased.  Enter quit to exit ==> bidu
Enter the stock purchase date ==> 5/11/2010
Enter the date you sold the stock ==> 5/13/2010

How many stocks were purchased on start date ==> 1

Our Baidu, Inc. (bidu) Portfolio
Action            Date         Shares          Price    Total Price
=================================================================
Buy         05/11/2010            1.0         684.98         684.98
Sold        05/13/2010           10.0          75.64         756.40
=================================================================
                                                             71.42
```

Requirements

- Your program will let the user continue to choose stocks until they enter ***quit*** for the name of the stock and then the program will exit.
- If the stock index entered by the user is invalid (we can't find it in the stocklist.csv), warn the user and allow them to reenter until they give a valid stock or quit.
- If the purchase date is not a valid date, then warn the user and let them re-enter the date until correct
- If the sell date of the stock is invalid, warn the user and let them re-enter until they get it correct.
- If the purchase date is greater than the sell date it obviously isn't possible for our program to use that, so warn the user and allow them to enter both dates again.
- Allow the user to enter the number of stocks to buy on the purchase date. The number must be an integer and greater than 0.
- When looking for the purchase date or the sell date in the history file for the company, if either of those aren't found, let the user that no data was recorded for that date, and restart.

Development notes

- Validating dates and times can be very hard to do. Luckily like many tough problems, python has a module to help. The datetime module can handle most of your problems.

```
>>> import datetime
>>> valid_date = "01/01/2015"
>>> invalid_date1 = "01/35/2015"
>>> invalid_date2 = "notadate"
>>> my_date = datetime.datetime.strptime(valid_date, "%m/%d/%Y")
>>> type(my_date)
<class 'datetime.datetime'>
```

```
>>> my_date
datetime.datetime(2015, 1, 1, 0, 0)
>>>
```

Notice that the strptime method takes a format. We're going to use the MM/DD/YYYY format. The string %m/%d%Y tells strptime to format the data in that manner for validation.

When you try to create an invalid date even one in a valid date format, but one that can't exist, like 01/35/2015 you get an exception thrown, a ValueError. Luckily you know how to catch those now. You will also get a ValueError when you try to convert a string into a date that is not in the correct format.

```
>>> my_date = datetime.datetime.strptime(invalid_date1, "%m/%d/%Y")
Traceback (most recent call last):
  File "<pyshell#7>", line 1, in <module>
    my_date = datetime.datetime.strptime(invalid_date1, "%m/%d/%Y")
  File "C:\Python34\lib\_strptime.py", line 500, in _strptime_datetime
    tt, fraction = _strptime(data_string, format)
  File "C:\Python34\lib\_strptime.py", line 337, in _strptime
    (data_string, format))
ValueError: time data '01/35/2015' does not match format '%m/%d/%Y'
>>>
```

- You can use a datetime object and compare it to other datetime objects. If it's less than another datetime object then it is a date and time prior to the other.

```
>>> d1 = datetime.datetime.strptime("01/01/2015", "%m/%d/%Y")
>>> d2 = datetime.datetime.strptime("01/02/2015", "%m/%d/%Y")
>>> d1 < d2
True
>>> d1 > d2
False
>>>
```

- You can also output a datetime object in The format you want. Again, we'll use MM/DD/YYYY.

```
>>> d1.strftime("%m/%d/%Y")
'01/01/2015'
>>>
```

- We'll be using the MM/DD/YYYY format that we see in these examples, but you can play around with different formats.
- The csv module is used to make reading comma separated value files easier.

Example

```
Enter the name of the stock purchased.  Enter quit to exit ==>
Could not find the stock .  Please enter another

Enter the name of the stock purchased.  Enter quit to exit ==> badindex
Could not find the stock badindex.  Please enter another

Enter the name of the stock purchased.  Enter quit to exit ==> AAPL
Enter the stock purchase date ==> djskl
You entered an incorrect date, please re-enter in the format MM/DD/YYYY
Enter the stock purchase date ==> 01/01/1960
Enter the date you sold the stock ==> 9229
You entered an incorrect date, please re-enter in the format MM/DD/YYYY
Enter the date you sold the stock ==> 01/40/1970
You entered an incorrect date, please re-enter in the format MM/DD/YYYY
Enter the date you sold the stock ==> 01/01/1970

How many stocks were purchased on start date ==> 10
Could not locate the start date of 1960-01-01 00:00:00
Could not locate the end date of 1970-01-01 00:00:00

Enter the name of the stock purchased.  Enter quit to exit ==> aapl
Enter the stock purchase date ==> 01/04/2005
Enter the date you sold the stock ==> 01/04/2015

How many stocks were purchased on start date ==> 10
Could not locate the end date of 2015-01-04 00:00:00

Enter the name of the stock purchased.  Enter quit to exit ==> aapl
Enter the stock purchase date ==> 01/04/2005
Enter the date you sold the stock ==> 01/05/2015

How many stocks were purchased on start date ==> 10

Our Apple Inc. (AAPL) (aapl) Portfolio
Action            Date         Shares          Price    Total Price
=================================================================
Buy         01/04/2005           10.0          63.80         638.00
Sold        01/05/2015          140.0         106.25       14875.00
=================================================================
                                                          14237.00
```

```
Enter the name of the stock purchased.  Enter quit to exit ==> googl
Enter the stock purchase date ==> 8/19/2004
Enter the date you sold the stock ==> 2/11/2016

How many stocks were purchased on start date ==> 100

Our Google Inc. (googl) Portfolio
Action            Date         Shares          Price    Total Price
=================================================================
Buy         08/19/2004          100.0         100.00       10000.00
Sold        02/11/2016          100.0         706.36       70636.00
=================================================================
                                                           60636.00

Enter the name of the stock purchased.  Enter quit to exit ==> msft
Enter the stock purchase date ==> 3/13/1986
Enter the date you sold the stock ==> 8/17/2015

How many stocks were purchased on start date ==> 85

Our Microsoft Corporation (msft) Portfolio
Action            Date         Shares          Price    Total Price
=================================================================
Buy         03/13/1986           85.0          25.50        2167.50
Sold        08/17/2015        24480.0          47.32     1158393.60
=================================================================
                                                         1156226.10

Enter the name of the stock purchased.  Enter quit to exit ==> tsla
Enter the stock purchase date ==> 11/1/2010
Enter the date you sold the stock ==> 4/17/2015

How many stocks were purchased on start date ==> -0
You must enter a value greater than zero

How many stocks were purchased on start date ==> kdls
Enter an integer only.

How many stocks were purchased on start date ==> 200

Our Tesla Motors, Inc. (tsla) Portfolio
Action            Date         Shares          Price    Total Price
=================================================================
```

```
Buy         11/01/2010          200.0           21.94         4388.00
Sold        04/17/2015          200.0         206.79        41358.00
=============================================================
                                                            36970.00

Enter the name of the stock purchased.  Enter quit to exit ==> quit
>>>
```

References

- Python Date Time module https://docs.python.org/3.5/library/datetime.html
- Python csv module https://docs.python.org/3.5/library/csv.html
- Stock Data provided by https://www.quandl.com/data/WIKI

(Side information) - not part of the project.
Many websites have API's (Application Programming Interfaces) so that you can programmatically access the data. For instance, with quandl you can download the data straight from them programmatically. You can check the the docs at quandl for specifics, but if the data can be read from https://www.quandl.com/api/v3/datasets/WIKI/{company}.csv where {company} is the index of the company you want historical data for. eg. MSFT.

One of the easiest ways to interact with the internet with python is by using the urllib library.

```
>>> import urllib.request
>>> web_handle = urllib.request.urlopen("http://www.google.com")
>>> data = web_handle.read()
>>> print(data[:200])
b'<!doctype html><html itemscope="" itemtype="http://schema.org/WebPage"
lang="en"><head><meta content="Search the world\'s information, including
webpages, images, videos and more. Google has many speci'
>>>
```

This code downloads the google home page and then displays the first 200 bytes of it. You can find more documentation about urllib at https://docs.python.org/3/library/urllib.html

There are lots of API's and helper modules for python, including some to help you automate tweets and posts in other social media apps. Many of the API's deliver csv, xml, json and other formats.