New York University
School of Continuing and Professional Studies
Division of Programs in Information Technology

Introduction to Python

Homework Discussion, Session 7

7.1 Reading through stock_prices.csv, build a dict of lists in which the key is the stock ticker and the value is a list of closing prices for that ticker. Sort the dictionary by the difference between the highest and lowest values for each stock ticker for the year. If you have time, render the sort function as a lambda.

This solution will be very similar to previous Fama-French assignments, in particular one in which you built the 'summing' dictionary. Consider that in that assignment you maintained and updated a float sum for each year. In this assignment you'll maintain a list of values for each stock ticker.

Your dictionary considerations will be the same (specifically, whether a key is new or already exists in the dict); this time instead of pairing a new key (i.e., a key not yet in the dict) to value 0, you'll pair the new key to an empty list. Instead of adding a new value to an old key's (i.e., a key already in the dict) paired float, you'll append a new value to the key's paired list.

As you loop line by line, the operative code line is this:

```
ticker_prices[ticker].append(close)
```

where you are appending the closing price to a list of prices associated with a particular ticker (AAPL, GOOG, etc.)

However, as discussed, you will need to check the dict ahead of time to see if the ticker key is already there. If not, set the key and value in the dict for that line.

```
ticker_prices[ticker] = []
```

Please avoid looping through the file more than once. Please avoid using a separate list or set to hold any of the values. A dictionary of lists is all you need! This exercise helps you see how you can build and sort a multidimensional structure.

The sort function that sorts each key in the dictionary by the difference between the highest and lowest prices recorded. Use the max() and min() functions for this purpose. This sort function will also be very similar to the previous open/close price sort function.

Pseudocode:

```
initialize an empty dict

loop through the file line-by-line (omitting the 1st line)
    split line into a list, including 'ticker' and 'closing' element values
    if the 'ticker' is not in the dict:
        set the ticker as key and value as empty list
    append the new float value to the list for this key

loop through the keys in the dict and sort them by max-min value in its list
```