CS 8, Winter 2015
Homework Assignment #? (draft)

Assignment Overview

The goal of this project is to gain more practice with file I/O, dictionary, lists and functions. It also introduces you to the advantage of code reuse.

Project Specifications

1. A file of Apple's daily stock's prices will be given to you, whose name is table.csv. The daily “open”, “high”, “low”, “close”, “volume” and “adj close” of Apple stock from 1984 to 2013 are stored. This file could be opened by Notepad or a text editor, and is delimited by commas. If you open it with Excel, it will show you the data as a spreadsheet. Be warned that this is a fairly big file with more than 7000 lines!

2. You want to perform similar operations as what you did with the sunspots data: construct a database, extract items of a specific period from the database, tabulate and print the extracted data, and compute average of the extracted data.

3. Certainly, these two data sets are not exactly the same. Sunspots data have only one attribute (sunspots), while Apple data have six attributes (open, high, low, close, volume, and adj close). Apple data are delimited by comma while sunspot data by space. Furthermore, Apple data are acquired daily while sunspots data are acquired monthly.

4. However, these differences are not critical and, with some appropriate data conversion, you can match the format of these two data sets to allow bulk of your sunspot codes to be reused. In fact, with proper design, init_dictionary, input_query_period, extract_data, and aveage_data can all be reused.

Deliverables

The deliverable for this assignment is the following file:

apple.py – the source code for your Python program

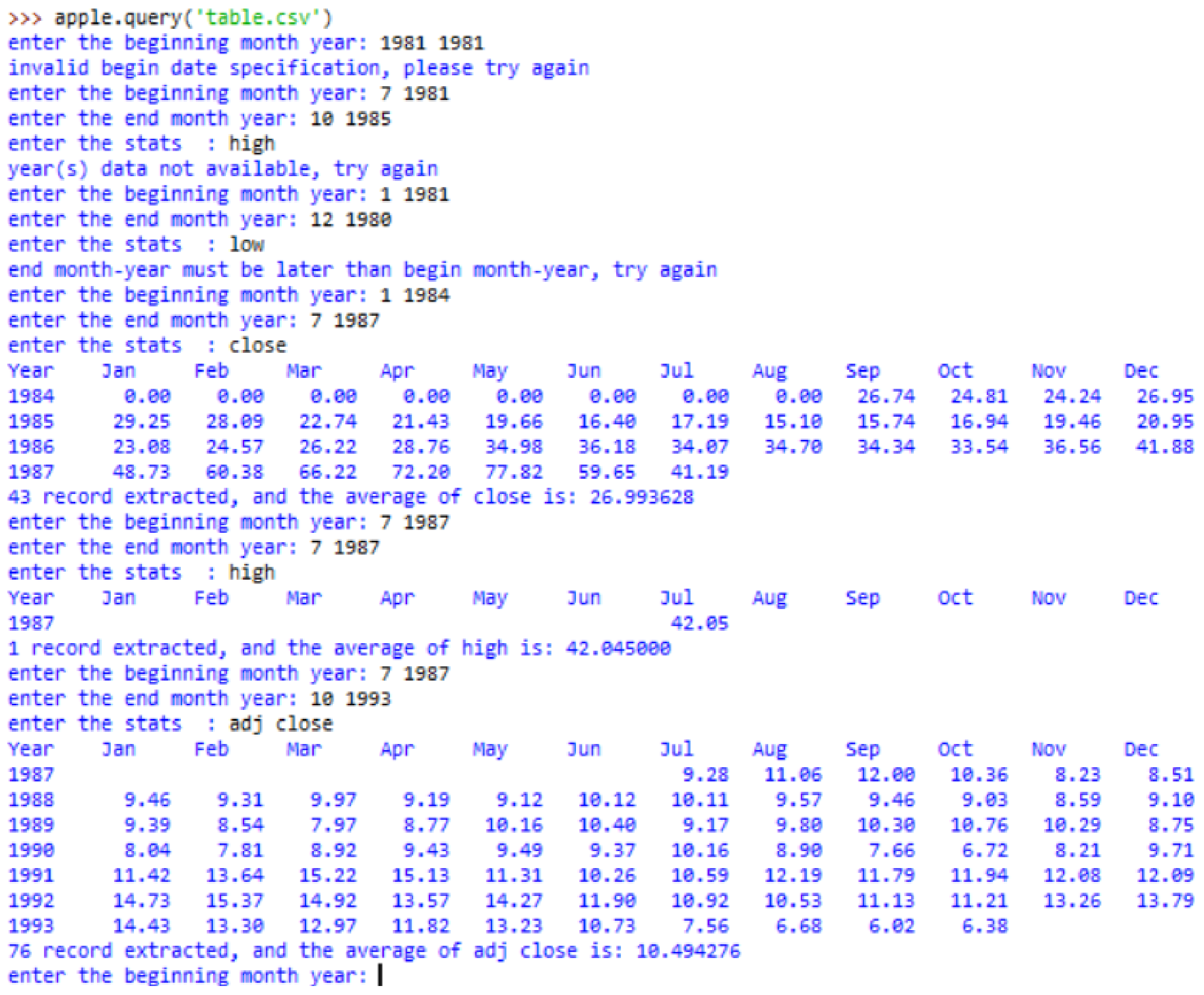
Be sure to use the specified file name and submit it for grading via the **turnin** system before the project deadline.

Assignment Notes:

1. To enable code reuse, you must perform two kinds of data conversions: (1) aggregate daily data into monthly data, and (2) separate a dictionary with six attributes into six dictionaries of one attribute each. This way, your Apple data will become very similar to the sunspot data.

2. Again, regardless of how you perform the operations internally, there should be a single function called “query” that takes a filename as input. This function should invoke all other functions appropriately in a way very similar to what you did with the sunspots assignment (with the extra step of database conversion).

Sample Outputs:



```
>>> apple.query('table.csv')
enter the beginning month year: 1981 1981
invalid begin date specification, please try again
enter the beginning month year: 7 1981
enter the end month year: 10 1985
enter the stats  : high
year(s) data not available, try again
enter the beginning month year: 1 1981
enter the end month year: 12 1980
enter the stats  : low
end month-year must be later than begin month-year, try again
enter the beginning month year: 1 1984
enter the end month year: 7 1987
enter the stats  : close
Year    Jan     Feb     Mar     Apr     May     Jun     Jul     Aug     Sep     Oct     Nov     Dec
1984      0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00   26.74   24.81   24.24   26.95
1985     29.25   28.09   22.74   21.43   19.66   16.40   17.19   15.10   15.74   16.94   19.46   20.95
1986     23.08   24.57   26.22   28.76   34.98   36.18   34.07   34.70   34.34   33.54   36.56   41.88
1987     48.73   60.38   66.22   72.20   77.82   59.65   41.19
43 record extracted, and the average of close is: 26.993628
enter the beginning month year: 7 1987
enter the end month year: 7 1987
enter the stats  : high
Year    Jan     Feb     Mar     Apr     May     Jun     Jul     Aug     Sep     Oct     Nov     Dec
1987                                                     42.05
1 record extracted, and the average of high is: 42.045000
enter the beginning month year: 7 1987
enter the end month year: 10 1993
enter the stats  : adj close
Year    Jan     Feb     Mar     Apr     May     Jun     Jul     Aug     Sep     Oct     Nov     Dec
1987                                                      9.28   11.06   12.00   10.36    8.23    8.51
1988      9.46    9.31    9.97    9.19    9.12   10.12   10.11    9.57    9.46    9.03    8.59    9.10
1989      9.39    8.54    7.97    8.77   10.16   10.40    9.17    9.80   10.30   10.76   10.29    8.75
1990      8.04    7.81    8.92    9.43    9.49    9.37   10.16    8.90    7.66    6.72    8.21    9.71
1991     11.42   13.64   15.22   15.13   11.31   10.26   10.59   12.19   11.79   11.94   12.08   12.09
1992     14.73   15.37   14.92   13.57   14.27   11.90   10.92   10.53   11.13   11.21   13.26   13.79
1993     14.43   13.30   12.97   11.82   13.23   10.73    7.56    6.68    6.02    6.38
76 record extracted, and the average of adj close is: 10.494276
enter the beginning month year: 
```