CSC 212 Project

Analyzing Stock Data

تحليل بيانات الاسهم

لا يغني عن ملف
Project

الهدف منه

تطبيق تراكيب البيانات والاستفادة من بيانات الأسهم عن طريق:

* تنظيمها
* وتحليلها
* واستقرائها

مجموعة البيانات (Dataset)

جداول بيانات لأسهم الشركات، كل جدول لشركة موجود بملف .CSV (Comma Separated Values)

Table 1: Sample stock data for Google (company code: GOOGL)

Date	Open	High	Low	Close	Volume
2004-08-19	2.50	2.60	2.40	2.51	893181924
2004-08-20	2.53	2.73	2.52	2.71	456686856
2004-08-23	2.77	2.84	2.73	2.74	365122512
2004-08-24	2.78	2.79	2.59	2.62	304946748
...	...	...	...	...	...

6 أعمدة للجدول:

Date

Open

High

Low

Close

Volume

مجموعة البيانات (Dataset)

Table 1: Sample stock data for Google (company code: GOOGL)

Date	Open	High	Low	Close	Volume
2004-08-19	2.50	2.60	2.40	2.51	893181924
2004-08-20	2.53	2.73	2.52	2.71	456686856
2004-08-23	2.77	2.84	2.73	2.74	365122512
2004-08-24	2.78	2.79	2.59	2.62	304946748
...	...	...	...	...	...

- **Date**: The specific trading day for which the data is recorded.
- **Open**: The price at which the first trade was made at the beginning of the trading day.
- **High**: The highest price that the stock reached during the trading day.
- **Low**: The lowest price that the stock reached during the trading day.
- **Close**: The price at which the last trade was made at the end of the trading day.
- **Volume**: The number of shares of the stock that were traded during the trading day.

مجموعة البيانات (Dataset)

.CSV (Comma Separated Values)

Table 1: Sample stock data for Google (company code: GOOGL)

Date	Open	High	Low	Close	Volume
2004-08-19	2.50	2.60	2.40	2.51	893181924
2004-08-20	2.53	2.73	2.52	2.71	456686856
2004-08-23	2.77	2.84	2.73	2.74	365122512
2004-08-24	2.78	2.79	2.59	2.62	304946748
...	...	...	...	...	...

المحتوى عند عرضه عن طريق
Excel

Listing 1: Sample of Google stock data stored in a CSV file named "GOOGL.csv"

```
Date,Open,High,Low,Close,Volume
2004-08-19,2.502502918243408,2.6041040420532227,2.4014010429382324,2.5110108852386475,893181924
2004-08-20,2.527777910232544,2.7297298908233643,2.515014886856079,2.7104599475860596,456686856
2004-08-23,2.771522045135498,2.8398399353027344,2.728979110717734,2.7377378940582275,365122512
2004-08-24,2.7837839126586914,2.792793035507202,2.59184193611145,2.6243739128112793,304946748
```

المحتوى داخل الملف
Source Code

مجموعة البيانات (Dataset)

يوجد مجلدين للبيانات المجدولة في ملف الموارد

Real: you will find real data files
Examples: you will find simple example data to debug your code

Terminology المصطلحات

- **Data Point**: A single piece of data recorded at a specific time. In our dataset, a data point could be the open, high, low, and close prices and volume for a stock on a particular day or a combination of these.
- **Time Series**: A sequence of data points representing the same information over time. In the context of stock data, a time series could be the daily closing prices of a stock over a period.
- **Moving Average**: A calculation used to analyze time series by creating a series of averages of subsets of the full data set. It helps smooth out price data over a specified period, making it easier to identify trends.

Example 1. *Consider a simple scenario where we have the daily closing prices of a stock over five days as follows:*

- *Day 1: $10*
- *Day 2: $12*
- *Day 3: $11*
- *Day 4: $13*
- *Day 5: $14*

To calculate a 3-day moving average (period = 3), we average the prices over three consecutive days, then move the window one day forward and repeat the process. The calculations are as follows:

1. *First 3-day moving average (Days 1-3):* $\frac{\$10 + \$12 + \$11}{3} = \$11$
2. *Second 3-day moving average (Days 2-4):* $\frac{\$12 + \$11 + \$13}{3} = \$12$
3. *Third 3-day moving average (Days 3-5):* $\frac{\$11 + \$13 + \$14}{3} = \$12.67$

The resulting time series will then be:

- *Day 3: $11*
- *Day 4: $12*
- *Day 5: $12.67*

Terminology المصطلحات

- **Price Increase**: Refers to a rise in the price of a stock from one time period to the next
  A price increase between two different dates is defined as the difference between the closing prices on those two days.
  A Single Day Price Increase (SDPI) is defined as the difference between the closing and opening price on that day.

- **Stock Performance:** An evaluation of how the price of a particular stock has changed over time. Performance can be measured in various ways, but in our case, we will focus on the relative change in price.

Example 2. *Given the following closing prices:*

- *Day 1: \$10*
- *Day 2: \$12*
- *Day 3: \$11*
- *Day 4: \$13*
- *Day 5: \$14*

The performance of this stock between Day 2 and Day 4, for example, is $\frac{13-12}{12} = 0.083$, *or 8.3%, where 12 in the denominator is the price of Day 2.*

الاكواد

15 Java files:

- 8 interfaces
- 7 classes (2 of them is optional)

الاكواد

- Interfaces should not be changed.
- Some classes are given to you, whereas you should implement other classes.

Java file	Description	To-Do
Interface DLL	This is the interface of a doubly linked list seen in class, augmented with the method size	Implement this interface in the class DLLImp
Interface DLLComp	This is the interface of a doubly linked list with comparable data	Implement this interface in the class DLLCompImp [Requires Chapter on Sorting]
Class Pair	A generic class used to store a pair of objects	do not modify
Class CompPair	A generic class used to store a pair of objects. The pair is comparable on the second element	do not modify
Class DataPoint	A generic class that represents a data point	do not modify
Interface TimeSeries	A generic interface that represents a time series	Implement this interface in the class TimeSeriesImp
Interface NumericTimeSeries	An interface that specializes and extends TimeSeries for Double data	Implement this interface in the class NumericTimeSeriesImp
Class StockData	A class used to store daily stock data	do not modify
Interface StockHistory	An interface that represents the stock history of a single company	Implement this interface in the class StockHistoryImp (use TimeSeries to store data)
Class Map	A map interface that is generic in both key and data	Implement this interface in the class BST [Requires Chapter on BST]
Interface StockHistoryDataSet	An interface that represents the stock histories of several companies	Implement this interface in the class StockHistoryDataSetImp (use Map and StockHistory to store data) [Requires Chapter on BST]
Interface StockDataLoader	An interface used to load stock data from a file	Implement this interface in the class StockDataLoaderImp
Interface StockDataSetAnalyzer	An interface used to analyze stock data	Implement this interface in the class StockDataSetAnalyzerImp
Class StockAnalyzerCLI	This class includes a main method that runs some examples (not all code and cases are covered) You may find the output of this program in the file: resources/Output.txt.	The use of this class is optional
Class StockAnalyzerGUI	This is a GUI interface that uses other classes (not all code and cases are covered). You can find this class in the directory extra.	The use of this class is optional

Deliverable and Rules

You must deliver:

1. You have to submit the following classes (and any other classes you need) in a zipped file:
   - `DLLImp.java`
   - `DLLCompImp.java`
   - `TimeSeriesImp.java`
   - `NumericTimeSeriesImp.java`
   - `StockHistoryImp.java`
   - `BST.java`
   - `StockHistoryDataSetImp.java`
   - `StockDataLoaderImp.java`
   - `StockDataSetAnalyzerImp.java`

   Notice that you should **not upload** the interfaces and classes given to you.

You have to read and follow the following rules:

1. The specification given in the assignment (**class and interface names, and method signatures**) must not be modified. Any change to the specification results in compilation errors and consequently the mark zero.
2. All data structures used in this assignment **must be implemented** by the student. The use of Java collections or any other data structures library is strictly forbidden.
3. Posting the code of the assignment or a link to it on public servers, social platforms or any communication media including but not limited to Facebook, Twitter or WhatsApp will result in disciplinary measures against any involved parties.
4. The submitted software will be evaluated automatically.
5. All submitted code will be automatically checked for similarity, and if plagiarism is confirmed penalties will apply.
6. You may be selected for discussing your code with an examiner at the discretion of the teaching team. If the examiner concludes plagiarism has taken place, penalties will apply.