CSCI-201: Principles of Software Development

Spring 2022

PA3: SalStocks

Concepts Covered:

- Locks Programming
- Semaphore Programming
- Multi-Threaded Programming Design

Introduction:

Introducing the ground-breaking, definitely original, non-USC backed, nor affiliated startup, SalStocks! SalStocks aims to hire stockbrokers to complete stock trades between USC and various companies. As the Executive Officer of SalStocks, I am here to inform you that the Council of SAL has decided to commission all Spring 2022 CSCI 201 students to design a cutting-edge system that supports the scheduling of stockbrokers and trades.

What To Do:

In this assignment, you will first read in a JSON file containing various information regarding public companies and their stock. The JSON file format is as below but you can reuse the parser you've built in the previous assignments with bit of modifications as we have modified the JSON object.

```
{
  "data": [
    {
      "name": "Tesla Inc",
      "ticker": "TSLA",
      "startDate": "2010-06-29",
      "stockBrokers": 3,
      "description": "Tesla Motors, Inc. (Tesla) designs, develops, manufactures and sells electric vehicles and advanced electric vehicle powertrain components.",
      "exchangeCode": "NASDAQ"
    }
]}
```

Aside from the JSON file, you will need to read in a second file, in CSV format. Below is a layout of the second file, which contains information for the stock trades:

0, AAPL, 2
0, MSFT, 5
1, TSLA, -3
5, AAPL, -1
18, TSLA, 10
18, MSFT, -1
25, MSFT, 3

On each line of the CSV file, you will have the following fields:

- The first field indicates when the stock trade is to be initiated, measured in seconds from the start of trading.
- The second field indicates the ticker which corresponds to the public company whose stocks are being bought or sold.
- The third field indicates how many stocks are being bought or sold, where a positive number indicates a buy, and a negative number indicates a sale.

It takes each stockbroker exactly **one (1) second to complete a trade**. You can assume that each individual stockbroker will only carry out trades for a single company. The number of stockbrokers "allocated" to each company to trade exclusively their stock will be listed in the PA3.json file, as the value of the stockBrokers name. Stockbrokers that facilitate trading in a specific stock are called "market makers" in the finance industry.

Just like the previous assignment, you should prompt the user to enter a filename and check for the file's existence and validity when the program initially runs. You will need to utilize locks and conditions to settle the availability of the stockbrokers. For example, one stockbroker cannot make two trades at the same time. The first trade must be completed before starting the second trade. The program should output when a trade is initiated and when it is completed.

Sample Output:

```
What is the name of the file containing the company information?
PA3.json

What is the name of the file containing the schedule information?
schedule.csv

Starting execution of program...
```

```
[00:00:00.00] Starting purchase of 2 stocks of AAPL
[00:00:00.00] Starting purchase of 5 stocks of MSFT
[00:00:01.00] Finished purchase of 2 stocks of AAPL
[00:00:01.00] Finished purchase of 5 stocks of MSFT
[00:00:01.00] Starting sale of 3 stocks of TSLA
[00:00:02.00] Finished sale of 3 stocks of TSLA
[00:00:05.00] Starting sale of 1 stock of AAPL
[00:00:06.00] Finished sale of 1 stock of AAPL
[00:00:18.00] Starting purchase of 10 stocks of TSLA
[00:00:18.00] Starting sale of 1 stock of MSFT
[00:00:19.00] Finished purchase of 10 stocks of TSLA
[00:00:19.00] Finished sale of 1 stock of MSFT
[00:00:25.00] Starting purchase of 3 stocks of MSFT
[00:00:26.00] Finished purchase of 3 stocks of MSFT
All trades completed!
```

Grading Criteria:

You will be graded based on the correctness of scheduling as well as the order and duration of trades. We will run your code on our input CSV files. These files follow the same format as it's described in the assignment.

File I/O (20 points)

- Start the program.
    - The program should immediately prompt you for the name of the public company information file and reads in the file. **(5 points)**
- Enter the name of the JSON file.
    - The program should immediately prompt you for the name of the schedule file and reads in the file. **(5 points)**
- Enter missing.csv
    - The program should print out a relevant error message. **(4 points)**
    - The program should prompt you again for another schedule file. **(2 points)**
- If the program starts over from the beginning, take off an additional **(2 points)**
- Enter any of the well-formatted files.
    - The program should proceed. **(2 points)**

Locks Implementation (20 points)

- Concurrent implementation (locks, semaphores, thread pools, etc.) must be used.

Scheduler Execution (60 points)

- The program prints out a message to indicate the start of the execution. **(10 points)**
- The printouts should match the solution on our testing CSV files. **(20 points)**
- The program should not crash during execution. **(10 points)**
- Trader scheduling must be correct, e.g. a trader only taking the next trade after completing the previous one. **(10 points)**
- At the end of execution, the program prints out a message to indicate all trades are completed. **(10 points)**

Submission:

Export your Eclipse project and submit it on DEN. It is of paramount importance that you export correctly. **If we cannot import your project or if it contains errors after importing, you will get a zero.**

Some notes:

- You MUST include the Gson JAR file as part of your Eclipse project. Without this, we will not be able to run your submission and you will get a zero.
- It is up to you if you wish to include JSON files you create as part of your Eclipse project. We have our own files we will test you against, so we will ignore any files of yours.