Assignment #2
CSCI 201 Spring 2021
5% of Course Grade

Title
SalStocks

Topics Covered
Locks Programming
Semaphore Programming
Multi-Threaded Programming Design

Introduction
Introducing the groundbreaking, definitely original, non-USC backed, nor affiliated startup, SalStocks! SalStocks aims to hire stock brokers 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 2021 CSCI 201 students to design a cutting-edge system that supports the scheduling of stock brokers and trades.

Assignment
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 generally the same as the one we used in the previous assignment, so feel free to reuse the parser you've built. However, you will need to alter the classes, as we have modified the JSON object, by adding an additional name/value pair. Here's a sample JSON:

```
{
  "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"
    },
    {
      "name": "Apple Inc",
      "ticker": "AAPL",
      "startDate": "1980-12-12",
      "stockBrokers": 1,
      "description": "Apple Inc. (Apple) designs, manufactures and
markets mobile communication and media devices, personal computers,
and portable digital music players, and a variety of related software,
services, peripherals, networking solutions, and third-party digital
content and applications.",
      "exchangeCode": "NASDAQ"
    },
    {
      "name": "Microsoft Corporation",
      "ticker": "MSFT",
```

```
      "startDate": "1986-03-13",
      "stockBrokers": 2,
      "description": "Microsoft (Nasdaq "MSFT" @microsoft) enables
digital transformation for the era of an intelligent cloud and an
intelligent edge. Its mission is to empower every person and every
organization on the planet to achieve more.",
      "exchangeCode": "NASDAQ"
    },
    {
      "name": "Advanced Micro Devices",
      "ticker": "AMD",
      "startDate": "1972-12-29",
      "stockBrokers": 2,
      "description": "Advanced Micro Devices, Inc. engages in the
provision of semiconductor businesses. It operates through the
following segments: Computing \u0026 Graphics, and Enterprise,
Embedded and Semi-Custom.",
      "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 stock broker exactly **one (1) second to complete a trade**. You can assume that each individual stock broker will only carry out trades for a single company. The number of stock brokers “allocated” to each company to trade exclusively their stock will be listed in the `assignment2.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 stock brokers. For example, one stock broker 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.

Example output with timestamps bolded for clarity (you do not have to bold the timestamps for your execution of the program):

```
What is the name of the file containing the company information?
assignment2.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.

Note: If the program crashes or does not terminate at any point, -0.5 will be deducted.

File I/O (1.0%)
0.25% - JSON File I/O
0.25% - Text File I/O
0.5% - Checking for invalid user inputs

SalStocks Program Execution (4%)
1.5% - Trade start/completion print statements
2.5% - Semaphores and locks implementation