CSE 222 HOMEWORK 7:

Balanced Tree-based Stock Data Management

Gebze Technical University Computer Engineering Department

Project Report

Buket Gençer

210104004298

23.05.2024

Introduction

The goal of the homework is to manage stock data using a balanced tree data structure, specifically an AVL tree. The AVL tree helps to store, retrieve, and manipulate stock information efficiently.

Purpose and Scope of the Report

The purpose of this report is to explain the methods and results of the AVL tree-based stock data management system. This report covers:

- **The problem we are solving:** The efficient management of stock data using a balanced AVL tree structure.
- **The methods and tools we used:** Java programming language, Java Swing for the graphical user interface (GUI), and a random command generator to create test data.
- **How we collected, processed, and analyzed the data:** Generating random stock data commands, processing these commands using an AVL tree, and analyzing the time complexity of each operation.
- **The results of our analysis with visual aids:** Presenting graphs and charts that show the relationship between the size of the AVL tree and the time taken for operations like ADD, REMOVE, SEARCH, and UPDATE.
- **An interpretation of the results:** Discussing what the results mean in terms of performance and efficiency.

Problem Definition

The problem is to efficiently manage stock data using an AVL tree. Each stock has a unique symbol, a price, a trading volume, and a market capitalization. The AVL tree must stay balanced after each operation to ensure optimal performance.

Methods and Tools Used

Java Swing

Java Swing was used to create the graphical user interface (GUI). Swing provides a rich set of components for building GUIs and was suitable for visualizing the performance graphs of the AVL tree operations.

AVL Tree

An AVL tree is a self-balancing binary search tree where the difference in heights between the left and right subtrees of any node is at most one. This balance ensures that the time complexity for search, insertion, and deletion operations remains O(log n).

Random Command Generator

A custom Java program was developed to generate random stock data commands. This program creates commands like ADD, REMOVE, SEARCH, and UPDATE to simulate real-world operations on the AVL tree.

Data Collection, Processing, and Analysis

Data Collection

We generated random stock data using a custom Java program. The program creates commands like ADD, REMOVE, SEARCH, and UPDATE for stocks. This simulated data was used to test the performance of the AVL tree implementation.

Data Processing

The generated commands were processed using the following steps:

1. **Insertion (ADD):** Adding new stocks to the AVL tree.
2. **Deletion (REMOVE):** Removing stocks from the AVL tree.
3. **Search (SEARCH):** Finding stocks in the AVL tree.
4. **Update (UPDATE):** Updating stock information in the AVL tree.

Data Analysis

We measured the time taken for each operation and recorded the size of the tree after each operation. We used this data to analyze the performance of the AVL tree. The results were visualized using graphs that show the relationship between the size of the tree and the time taken for each operation.

AVL Tree and Time Complexity

AVL Tree

An AVL tree is a self-balancing binary search tree. In an AVL tree, the heights of the two child subtrees of any node differ by at most one. If at any time they differ by more than one, rebalancing is done to restore this property.

Insertion Algorithm

When inserting a new node into the AVL tree, the tree might become unbalanced. The insertion algorithm involves:

1. **Insert the node:** Follow the binary search tree insertion rule.
2. **Update the height:** Update the height of each node.
3. **Balance the tree:** Perform rotations if necessary to maintain the balance of the tree.

Deletion Algorithm

Deleting a node from an AVL tree might also unbalance the tree. The deletion algorithm involves:

1. **Delete the node:** Follow the binary search tree deletion rule.
2. **Update the height:** Update the height of each node.
3. **Balance the tree:** Perform rotations if necessary to maintain the balance of the tree.

Search Algorithm

Searching in an AVL tree follows the standard binary search tree rule. Given that the tree is balanced, the time complexity is O(log n).

Update Algorithm

Updating a node involves searching for the node, deleting it, and inserting the new node with updated values. This ensures the AVL properties are maintained.

Rotations

Rotations are used to maintain the balance of the tree:

1. **Right Rotation:** Used in the Left-Left case.
2. **Left Rotation:** Used in the Right-Right case.

3. **Left-Right Rotation:** Used in the Left-Right case.
4. **Right-Left Rotation:** Used in the Right-Left case.

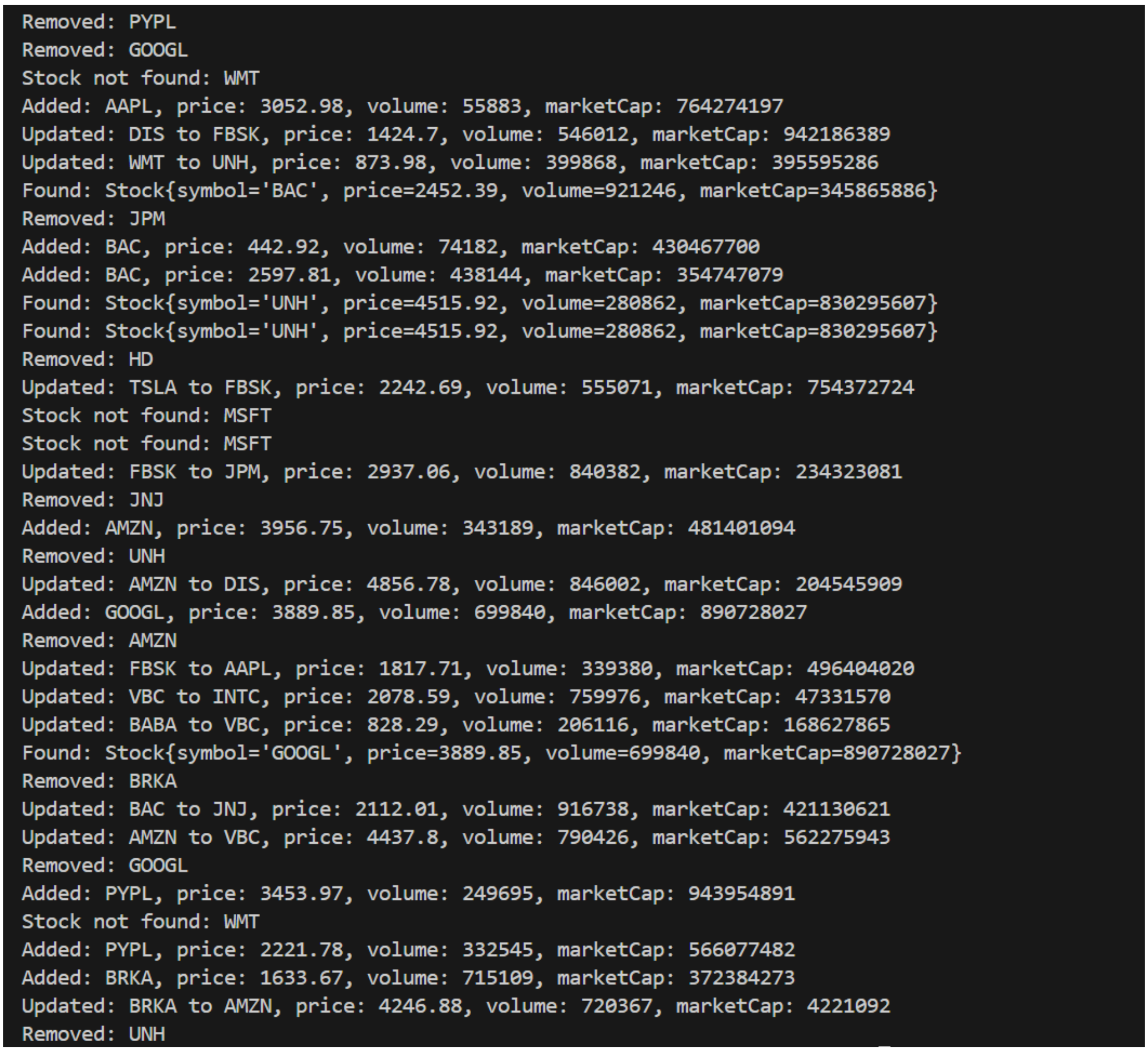
Time Complexity

The time complexity for insertion, deletion, and searching operations in an AVL tree is O(log n), where n is the number of nodes in the tree. This ensures efficient performance for managing stock data.

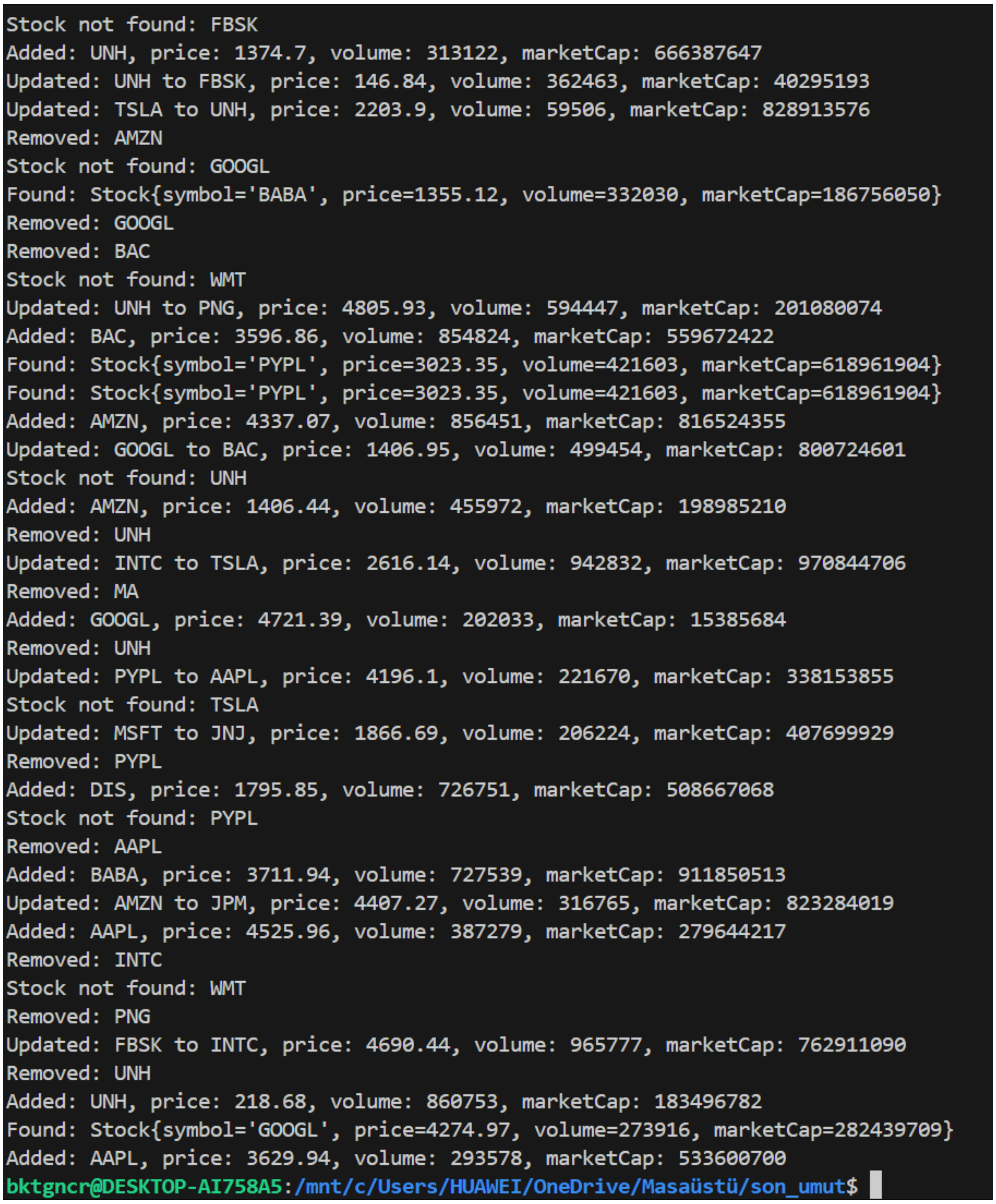
Analysis Results and Visuals

I added diffrent output from different input (commands)file

Some Output of Terminal:

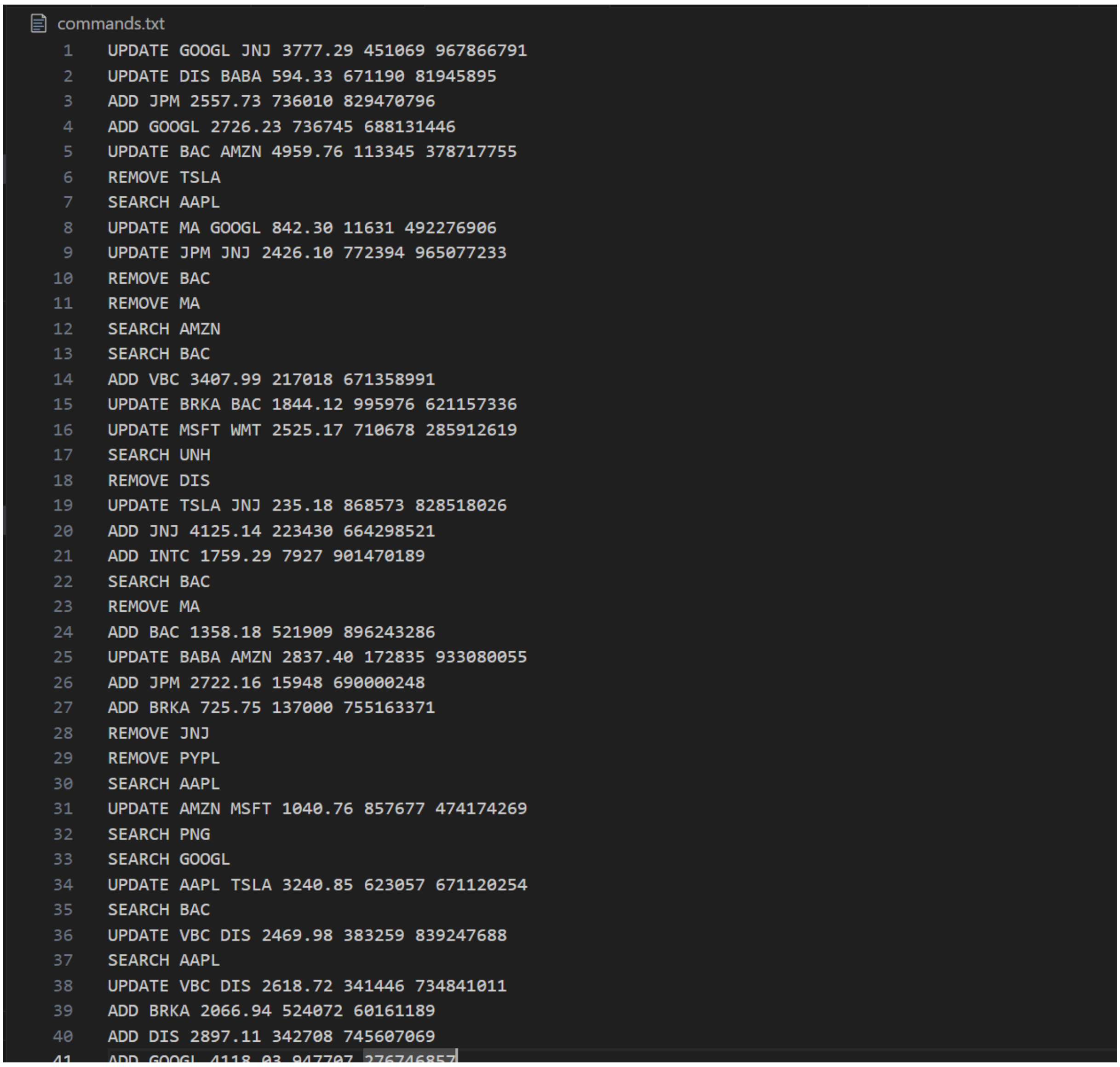


```
Removed: PYPL
Removed: GOOGL
Stock not found: WMT
Added: AAPL, price: 3052.98, volume: 55883, marketCap: 764274197
Updated: DIS to FBSK, price: 1424.7, volume: 546012, marketCap: 942186389
Updated: WMT to UNH, price: 873.98, volume: 399868, marketCap: 395595286
Found: Stock{symbol='BAC', price=2452.39, volume=921246, marketCap=345865886}
Removed: JPM
Added: BAC, price: 442.92, volume: 74182, marketCap: 430467700
Added: BAC, price: 2597.81, volume: 438144, marketCap: 354747079
Found: Stock{symbol='UNH', price=4515.92, volume=280862, marketCap=830295607}
Found: Stock{symbol='UNH', price=4515.92, volume=280862, marketCap=830295607}
Removed: HD
Updated: TSLA to FBSK, price: 2242.69, volume: 555071, marketCap: 754372724
Stock not found: MSFT
Stock not found: MSFT
Updated: FBSK to JPM, price: 2937.06, volume: 840382, marketCap: 234323081
Removed: JNJ
Added: AMZN, price: 3956.75, volume: 343189, marketCap: 481401094
Removed: UNH
Updated: AMZN to DIS, price: 4856.78, volume: 846002, marketCap: 204545909
Added: GOOGL, price: 3889.85, volume: 699840, marketCap: 890728027
Removed: AMZN
Updated: FBSK to AAPL, price: 1817.71, volume: 339380, marketCap: 496404020
Updated: VBC to INTC, price: 2078.59, volume: 759976, marketCap: 47331570
Updated: BABA to VBC, price: 828.29, volume: 206116, marketCap: 168627865
Found: Stock{symbol='GOOGL', price=3889.85, volume=699840, marketCap=890728027}
Removed: BRKA
Updated: BAC to JNJ, price: 2112.01, volume: 916738, marketCap: 421130621
Updated: AMZN to VBC, price: 4437.8, volume: 790426, marketCap: 562275943
Removed: GOOGL
Added: PYPL, price: 3453.97, volume: 249695, marketCap: 943954891
Stock not found: WMT
Added: PYPL, price: 2221.78, volume: 332545, marketCap: 566077482
Added: BRKA, price: 1633.67, volume: 715109, marketCap: 372384273
Updated: BRKA to AMZN, price: 4246.88, volume: 720367, marketCap: 4221092
Removed: UNH
```



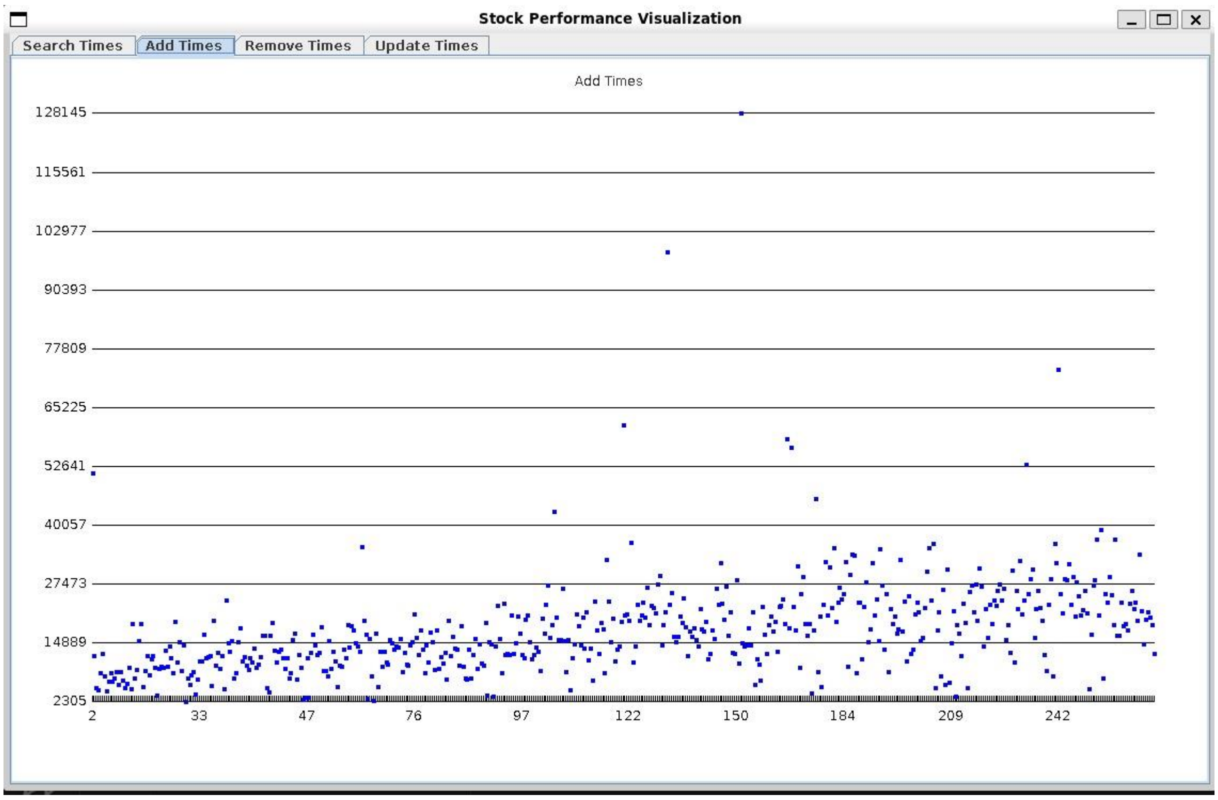
```
Stock not found: FBSK
Added: UNH, price: 1374.7, volume: 313122, marketCap: 666387647
Updated: UNH to FBSK, price: 146.84, volume: 362463, marketCap: 40295193
Updated: TSLA to UNH, price: 2203.9, volume: 59506, marketCap: 828913576
Removed: AMZN
Stock not found: GOOGL
Found: Stock{symbol='BABA', price=1355.12, volume=332030, marketCap=186756050}
Removed: GOOGL
Removed: BAC
Stock not found: WMT
Updated: UNH to PNG, price: 4805.93, volume: 594447, marketCap: 201080074
Added: BAC, price: 3596.86, volume: 854824, marketCap: 559672422
Found: Stock{symbol='PYPL', price=3023.35, volume=421603, marketCap=618961904}
Found: Stock{symbol='PYPL', price=3023.35, volume=421603, marketCap=618961904}
Added: AMZN, price: 4337.07, volume: 856451, marketCap: 816524355
Updated: GOOGL to BAC, price: 1406.95, volume: 499454, marketCap: 800724601
Stock not found: UNH
Added: AMZN, price: 1406.44, volume: 455972, marketCap: 198985210
Removed: UNH
Updated: INTC to TSLA, price: 2616.14, volume: 942832, marketCap: 970844706
Removed: MA
Added: GOOGL, price: 4721.39, volume: 202033, marketCap: 15385684
Removed: UNH
Updated: PYPL to AAPL, price: 4196.1, volume: 221670, marketCap: 338153855
Stock not found: TSLA
Updated: MSFT to JNJ, price: 1866.69, volume: 206224, marketCap: 407699929
Removed: PYPL
Added: DIS, price: 1795.85, volume: 726751, marketCap: 508667068
Stock not found: PYPL
Removed: AAPL
Added: BABA, price: 3711.94, volume: 727539, marketCap: 911850513
Updated: AMZN to JPM, price: 4407.27, volume: 316765, marketCap: 823284019
Added: AAPL, price: 4525.96, volume: 387279, marketCap: 279644217
Removed: INTC
Stock not found: WMT
Removed: PNG
Updated: FBSK to INTC, price: 4690.44, volume: 965777, marketCap: 762911090
Removed: UNH
Added: UNH, price: 218.68, volume: 860753, marketCap: 183496782
Found: Stock{symbol='GOOGL', price=4274.97, volume=273916, marketCap=282439709}
Added: AAPL, price: 3629.94, volume: 293578, marketCap: 533600700
bktgncr@DESKTOP-AI758A5:/mnt/c/Users/HUAWEI/OneDrive/Masaüstü/son_umut$
```

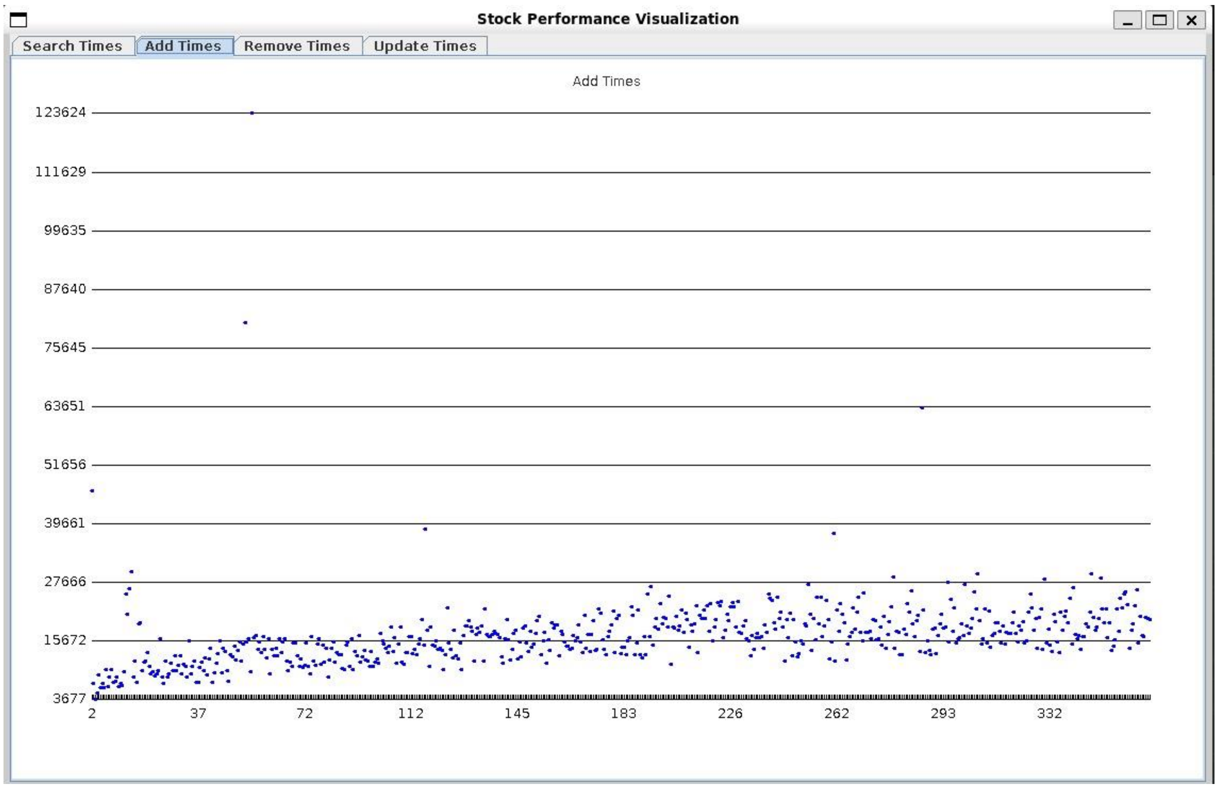
Examples of input file:

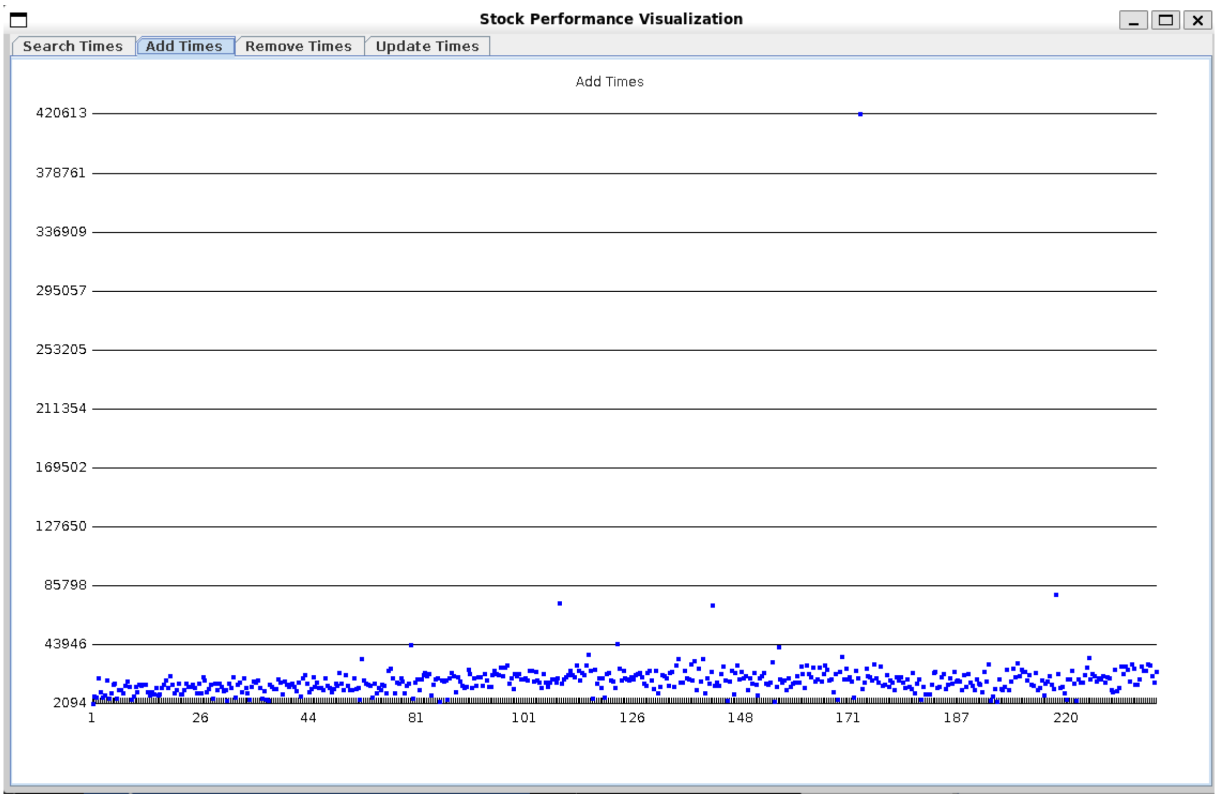

commands.txt

```
UPDATE GOOGL JNJ 3777.29 451069 967866791
UPDATE DIS BABA 594.33 671190 81945895
ADD JPM 2557.73 736010 829470796
ADD GOOGL 2726.23 736745 688131446
UPDATE BAC AMZN 4959.76 113345 378717755
REMOVE TSLA
SEARCH AAPL
UPDATE MA GOOGL 842.30 11631 492276906
UPDATE JPM JNJ 2426.10 772394 965077233
REMOVE BAC
REMOVE MA
SEARCH AMZN
SEARCH BAC
ADD VBC 3407.99 217018 671358991
UPDATE BRKA BAC 1844.12 995976 621157336
UPDATE MSFT WMT 2525.17 710678 285912619
SEARCH UNH
REMOVE DIS
UPDATE TSLA JNJ 235.18 868573 828518026
ADD JNJ 4125.14 223430 664298521
ADD INTC 1759.29 7927 901470189
SEARCH BAC
REMOVE MA
ADD BAC 1358.18 521909 896243286
UPDATE BABA AMZN 2837.40 172835 933080055
ADD JPM 2722.16 15948 690000248
ADD BRKA 725.75 137000 755163371
REMOVE JNJ
REMOVE PYPL
SEARCH AAPL
UPDATE AMZN MSFT 1040.76 857677 474174269
SEARCH PNG
SEARCH GOOGL
UPDATE AAPL TSLA 3240.85 623057 671120254
SEARCH BAC
UPDATE VBC DIS 2469.98 383259 839247688
SEARCH AAPL
UPDATE VBC DIS 2618.72 341446 734841011
ADD BRKA 2066.94 524072 60161189
ADD DIS 2897.11 342708 745607069
ADD GOOGL 4118.03 947707 276746857
```

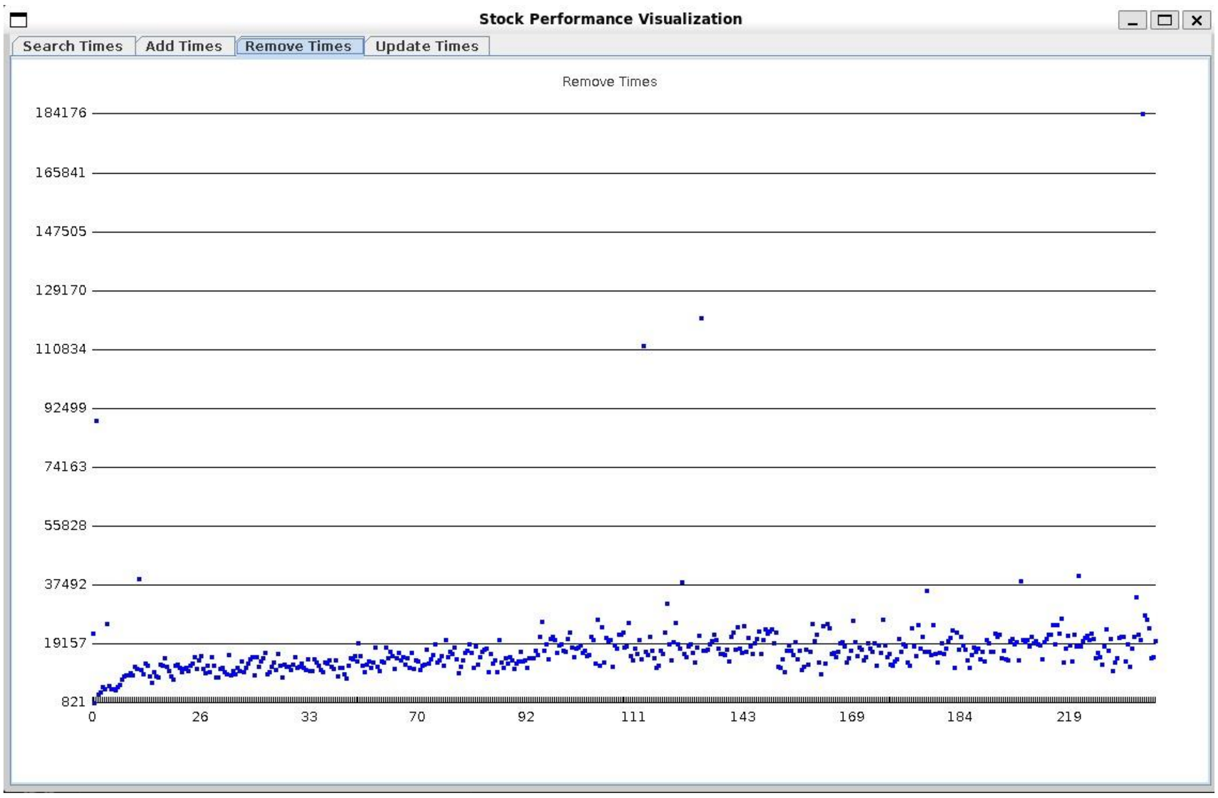
Add Operation

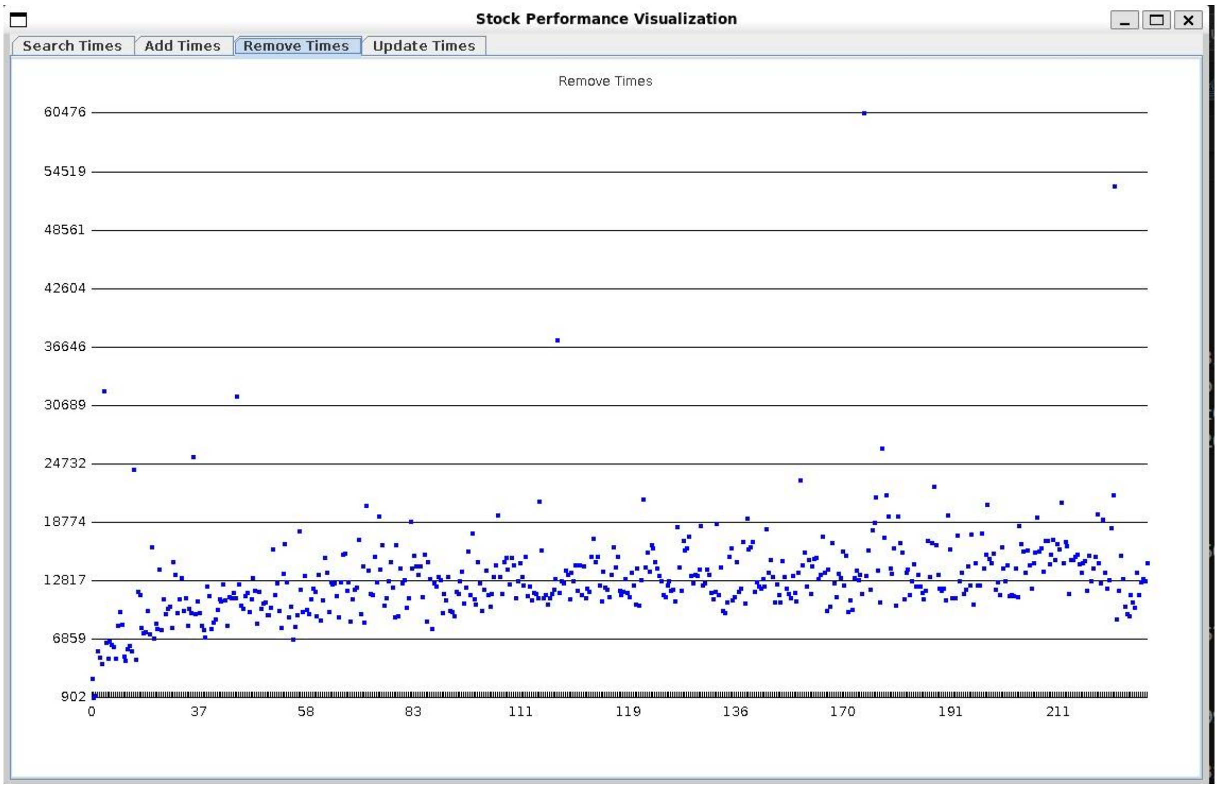


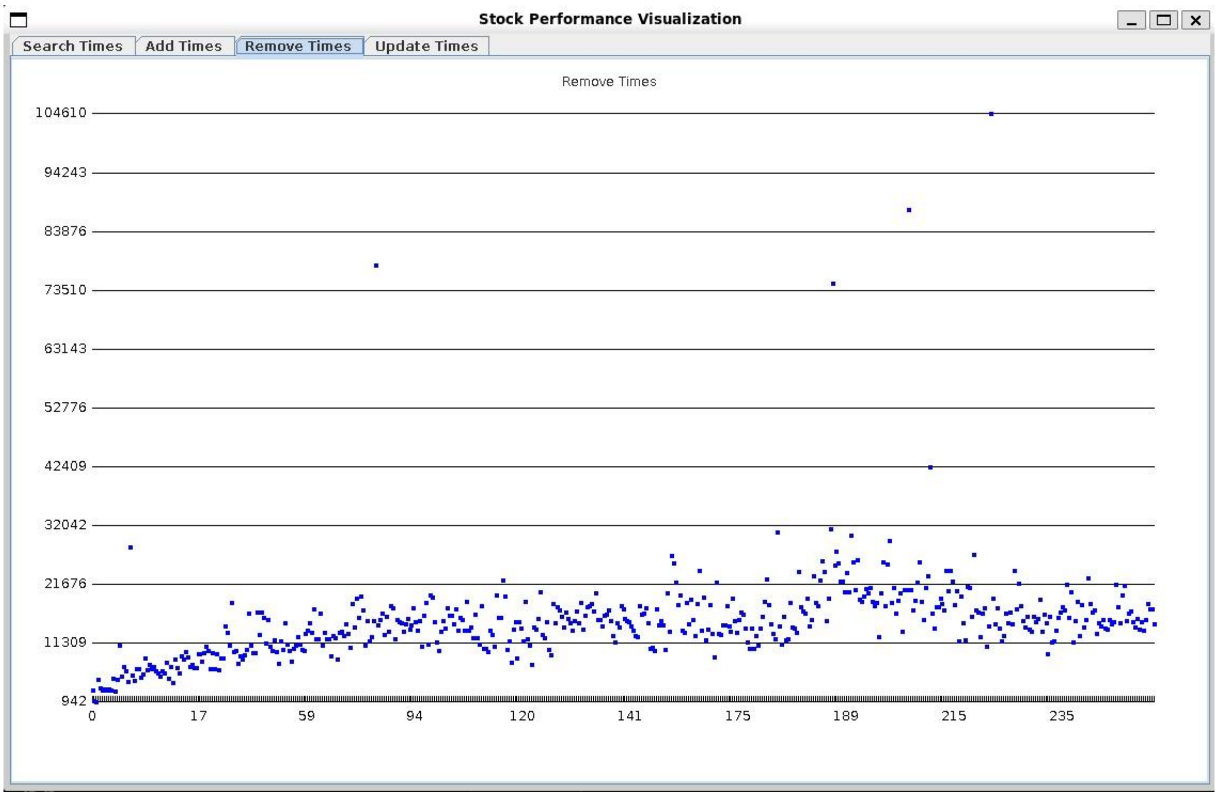




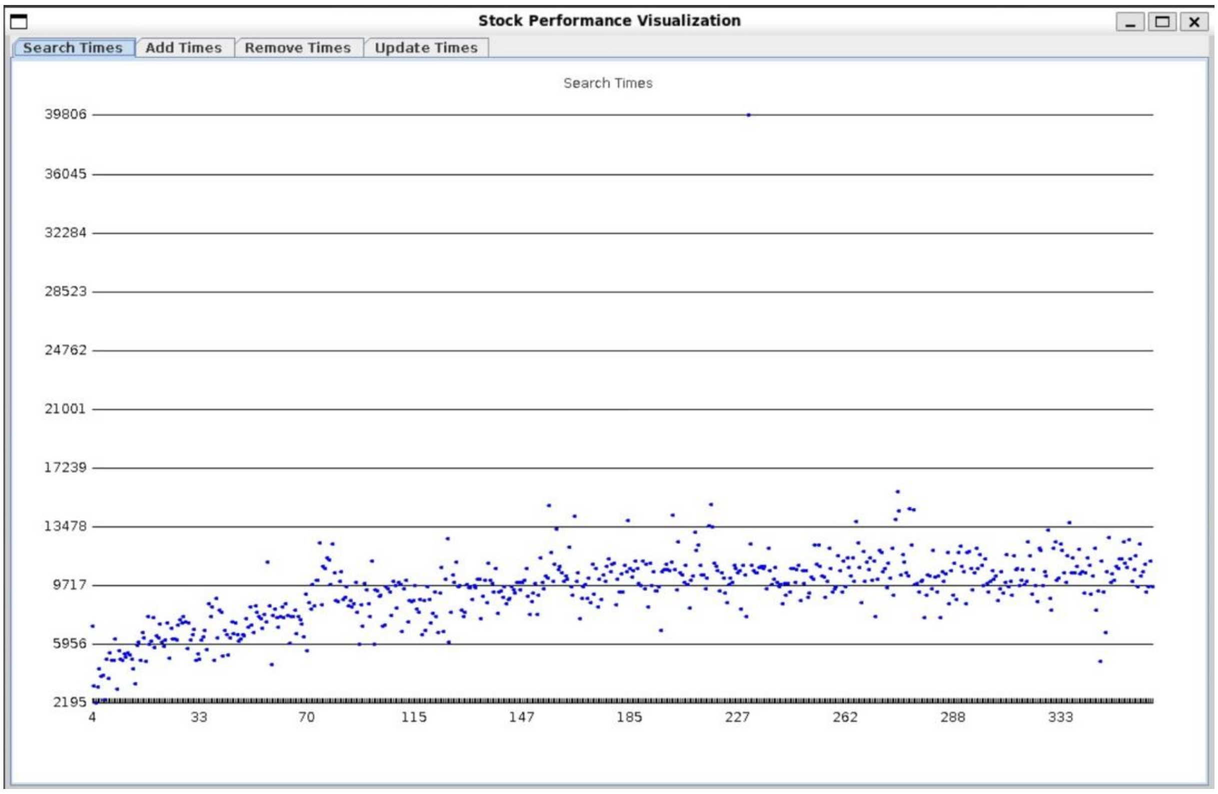
Remove Operation

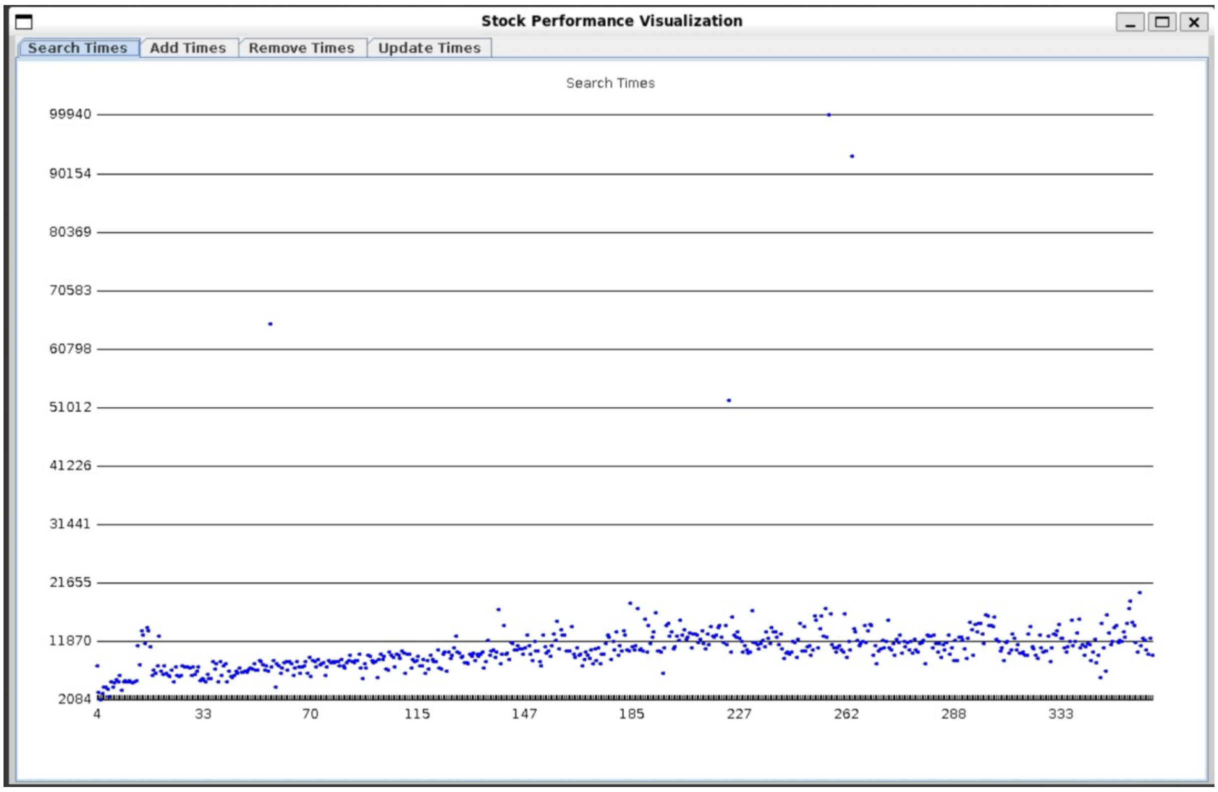


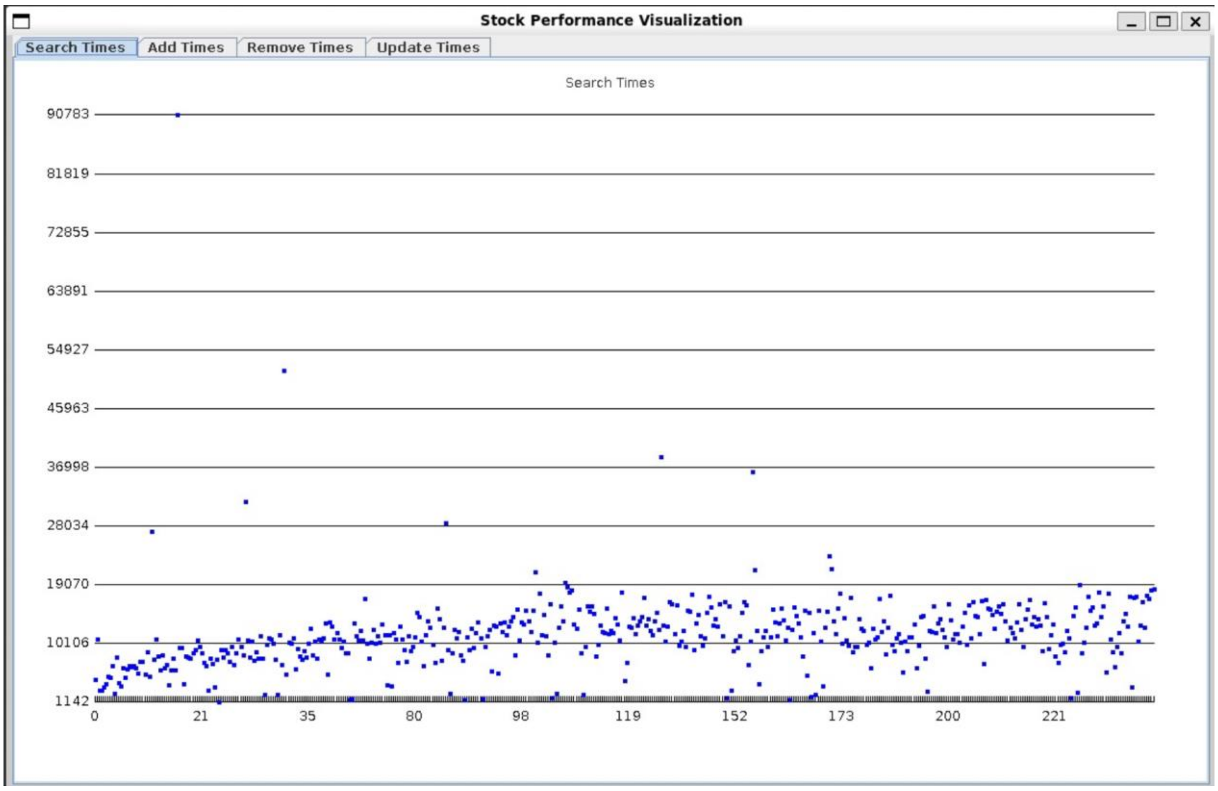




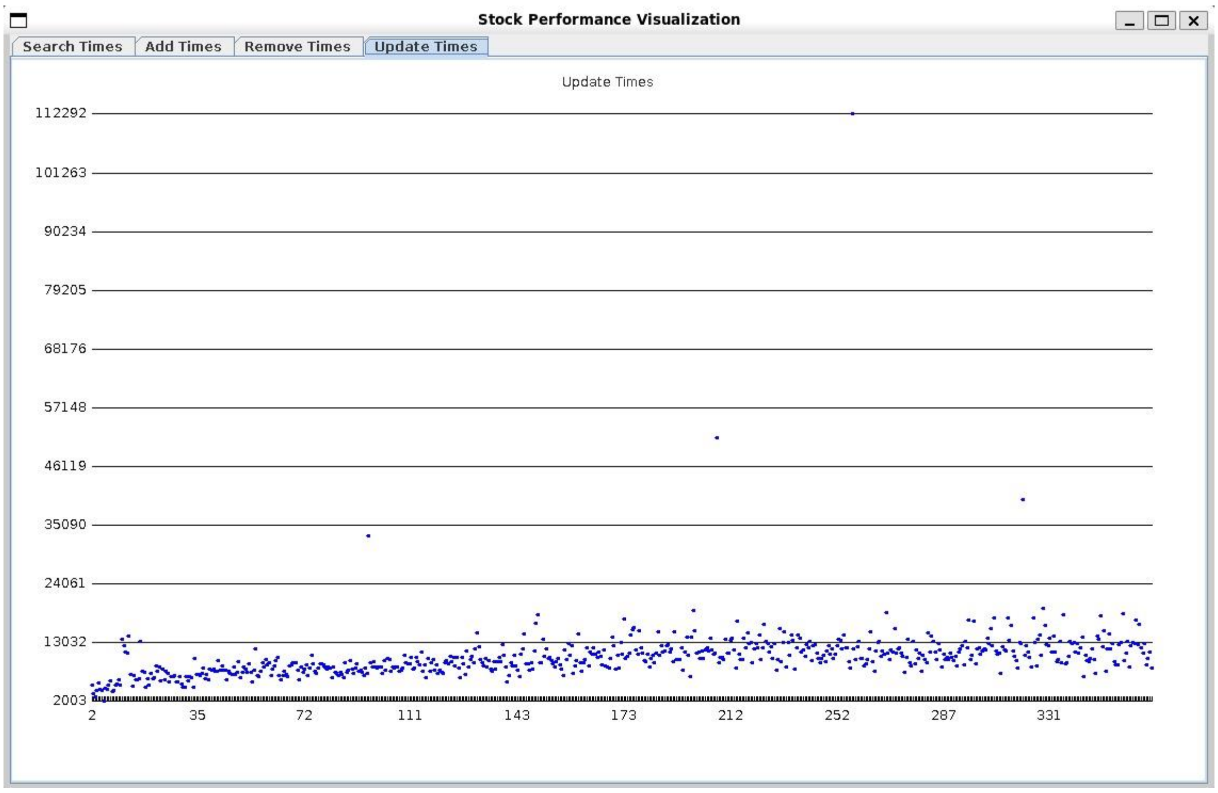
Search Operation

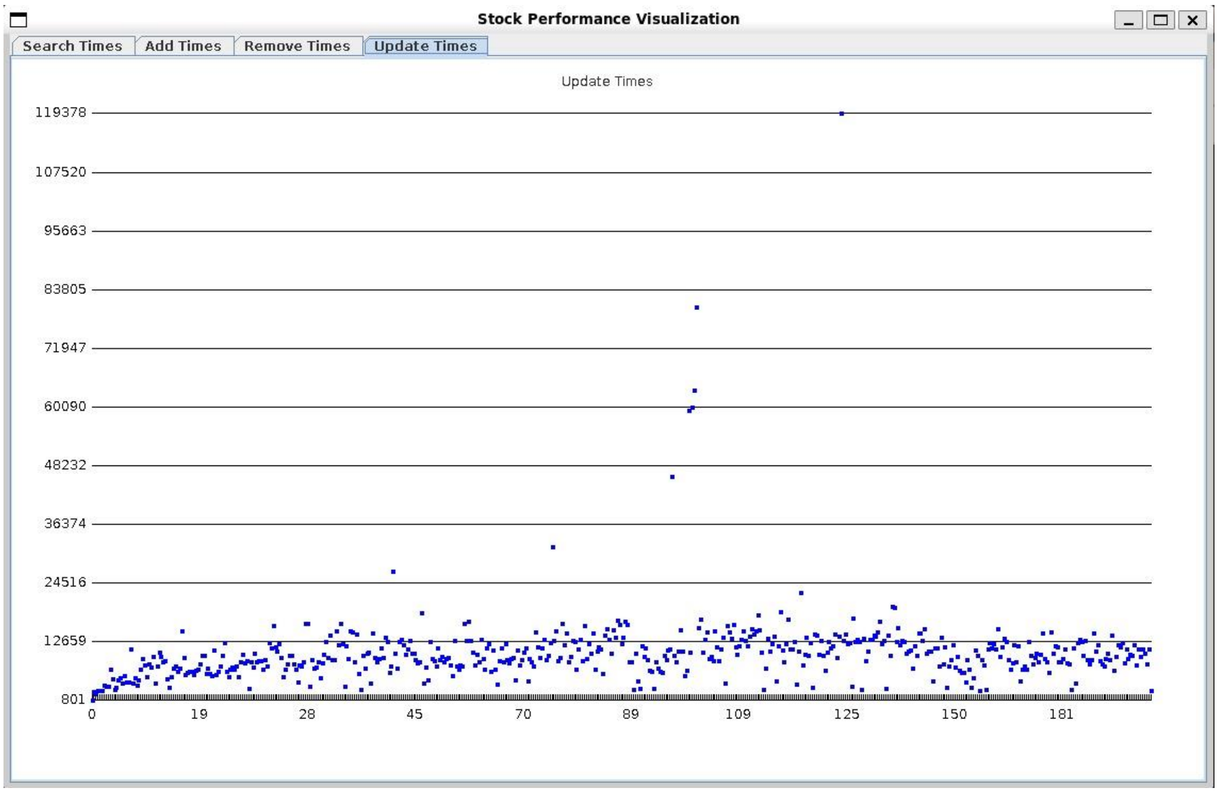

Stock Performance Visualization
Search Times
Add Times
Remove Times
Update Times
Search Times
39806
36045
32284
28523
24762
21001
17239
13478
9717
5956
2195
4
33
70
115
147
185
227
262
288
333

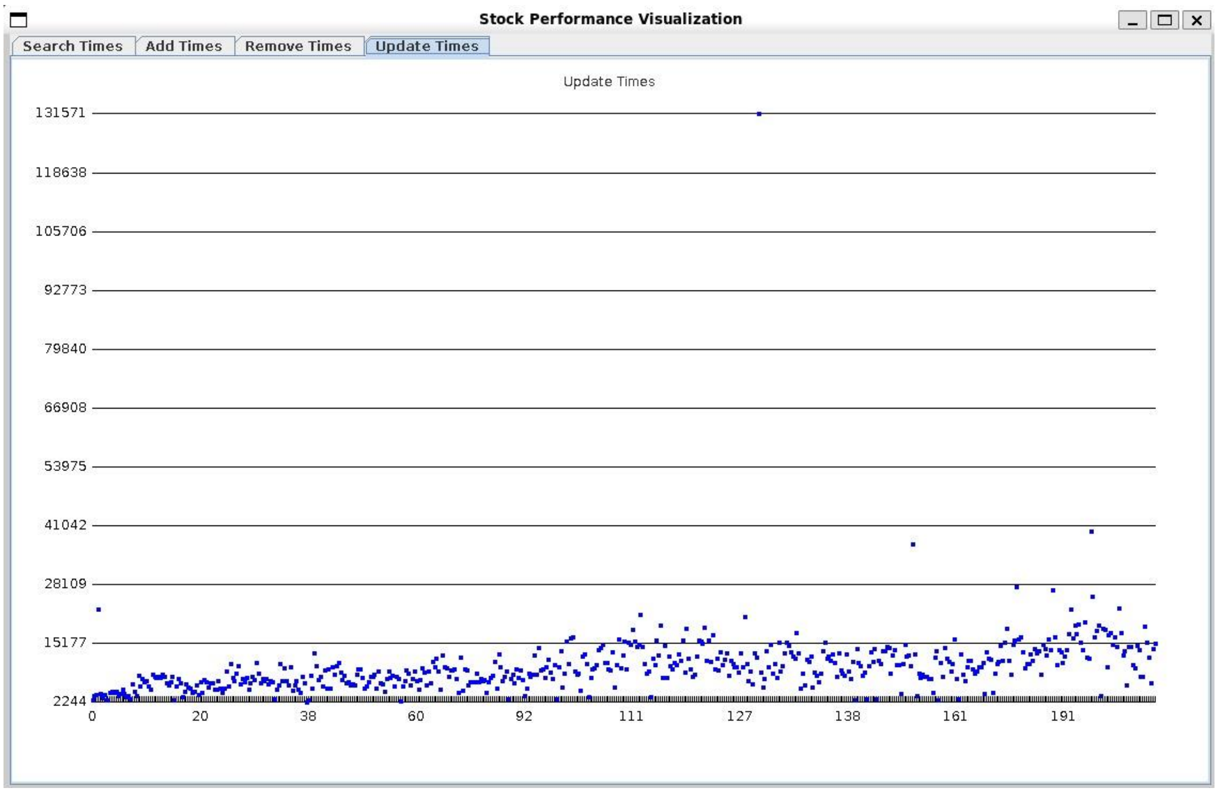

Stock Performance Visualization
Search Times
Add Times
Remove Times
Update Times
Search Times
99940
90154
80369
70583
60798
51012
41226
31441
21655
11870
2084
4
33
70
115
147
185
227
262
288
333



Update Operation




Stock Performance Visualization
Search Times
Add Times
Remove Times
Update Times
Update Times
119378
107520
95663
83805
71947
60090
48232
36374
24516
12659
801
0
19
28
45
70
89
109
125
150
181


Stock Performance Visualization
Search Times
Add Times
Remove Times
Update Times
Update Times
131571
118638
105706
92773
79840
66908
53975
41042
28109
15177
2244
0
20
38
60
92
111
127
138
161
191