Assignment – Super Simple Stock Market

|  |  |
| --- | --- |
| Module Name | Super Simple Stock Maeket |
| Company | JP Morgan |
| Author | Mohit Jumrani |
| Version | DRAFT |
| Date | 9th May 2016 |

Table of Contents

[Table of Contents 2](#_Toc450913509)

[1. Requirements 3](#_Toc450913510)

[2. Constraints & Notes 4](#_Toc450913511)

[2.1 Abbreviations 4](#_Toc450913512)

[3. Models and Functionalities 5](#_Toc450913513)

[3.1 Business Objects 5](#_Toc450913514)

[3.1.1 Trade BO 5](#_Toc450913515)

[3.1.2 Stock BO 5](#_Toc450913516)

[3.1.3 Miscellaneous 6](#_Toc450913517)

[3.2 Methods 6](#_Toc450913518)

[3.2.1 dividendYieldOrPERatio 6](#_Toc450913519)

[3.2.2 recordTrade 7](#_Toc450913520)

[3.2.3 volumeWeightedStockPrice 7](#_Toc450913521)

[3.2.4 geometricMean 8](#_Toc450913522)

[3.2.5 root 9](#_Toc450913523)

[4. Production Notes 10](#_Toc450913524)

[4.1 Initial Thoughts 10](#_Toc450913525)

[4.2 Refactoring Step 1 10](#_Toc450913526)

[4.3 Refactoring Step 2 10](#_Toc450913527)

[4.4 Refactoring Step 3 10](#_Toc450913528)

[4.5 Refactoring Step 4 11](#_Toc450913529)

[4.6 Final Step 11](#_Toc450913530)

# 1. Requirements

1. The Global Beverage Corporation Exchange is a new stock market trading in drinks companies.
   1. Your company is building the object-oriented system to run that trading.
   2. You have been assigned to build part of the core object model for a limited phase 1.
2. Provide the source code for an application that will: -
   1. For a given stock,
      1. Given any price as input, calculate the dividend yield
      2. Given any price as input, calculate the P/E Ratio
      3. Record a trade, with timestamp, quantity, buy or sell indicator and price
      4. Calculate Volume Weighted Stock Price based on trades in past 5 minutes
   2. Calculate the GBCE All Share Index using the geometric mean of the Volume Weighted Stock Price for all stocks

# 2. Constraints & Notes

1. Written in one of these languages - Java, C#, C++, Python.
2. The source code should be complete and able to be run and tested.
3. No database, GUI or I/O is required, all data need only be held in memory.
4. No prior knowledge of stock markets or trading is required – all formulas are provided below.
5. The code should provide only the functionality requested, however must be production quality.

## 2.1 Abbreviations

|  |  |
| --- | --- |
| Abbreviation | Full Form |
| TDD | Test-driven Development |
| BO | Business Object |

# 3. Models and Functionalities

As this assignment is to behave as a production quality bolt-on module, I have chosen to follow a top down approach as in a complex system with a common framework, complicated data is passed between utilities or functionalities using BOs.

Following a Test Driven approach requires refactoring of code to be done on the basis of failure of test cases. In this assignment, TDD approach is implemented using JUnit within the module. The production notes with commentary can be found in section 4

The module will contain 1 Utility Class containing all the functionality, and 1 BO each for Stock and Trade.

## 3.1 Business Objects

### 3.1.1 Trade BO

The purpose of this object is to represent the data present in a Trade

|  |  |  |
| --- | --- | --- |
| Property Type | Property Name | Other Details |
| stockSymbol | String | Contains the Stock Symbol |
| quantity | BigDecimal | Contains quantity of stock traded |
| tradeTimeStamp | Timestamp | Contains timestamp value of trade |
| tradeTypeObj | TradeType\* | Contains the type of trade, i.e., Buy or Sell |
| tradePrice | BigDecimal | Contains the stock price at which the trade occured |

\*See 3.1.3 for further details

### 3.1.2 Stock BO

The purpose of this object is to represent the basic data present for every stock symbol

|  |  |  |
| --- | --- | --- |
| Property Type | Property Name | Other Details |
| stockSymbol | String | Contains the Stock Symbol |
| stockTypeObj | StockType\* | Contains type of stock, i.e., Preferred or Common |
| lastDividend | BigDecimal | Contains value of Last Dividend received for stock |
| fixedDividend | BigDecimal | Contains value of Fixed Dividend received for stock |
| parValue | BigDecimal | Contains Par Value of stock |

\*See 3.1.3 for further details

### 3.1.3 Miscellaneous

**ErrorStrings** – Contains the sting values of errors and exceptions in the program.

**StockFunctionType** – Contains the differentiator for the *dividendYieldOrPERatio* method.

**StockType** – Contains the differentiator for the type of stock, i.e., Preferred or Common.

**TradeType** – Contains the differentiator for the type of trade, i.e., Buy or Sell.

**VolumeWeightType** – Contains the differentiator for the *volumeWeightedStockPrice* method.

## 3.2 Methods

### 3.2.1 dividendYieldOrPERatio

|  |  |  |
| --- | --- | --- |
| Details | Value | Other Details |
| Purpose | Calculate Dividend Yield or P/E Ratio for a given price | |
| Return Type | Object | Varies according to error code or calculated value. Returns BigDecimal upon success |
| Visibility | Public |  |
| Static | Yes |  |
| Exception Handling | Yes |  |

Parameter List:

|  |  |  |
| --- | --- | --- |
| Parameter | Type | Other Details |
| stockPrice | BigDecimal | Received Stock Price for Calculation |
| stockSymbol | String | Received Stock Symbol to get associated values for calculation. |
| stockList | List<Stock> | List of stocks which will be traversed to find the values needed for performing a function |
| stockFunction | StockFunctionType | Calculate Dividend Yield or P/E Ratio |

### 3.2.2 recordTrade

|  |  |  |
| --- | --- | --- |
| Details | Value | Other Details |
| Purpose | Record incoming trade | |
| Return Type | Object | Varies according to error code or completion of task. Returns List<Trade> upon success |
| Visibility | Public |  |
| Static | Yes |  |
| Exception Handling | Yes |  |

Parameter List:

|  |  |  |
| --- | --- | --- |
| Parameter | Type | Other Details |
| stockSymbol | String | Stock Symbol of Trade |
| quantity | BigDecimal | Quantity of stocks traded |
| currentTime | Timestamp | Time of trade |
| typeOfTrade | TradeType\* | Buy or Sell |
| tradePrice | BigDecimal | Price at which the trade occurred |
| tradeList | List<Trade> | List of Trades to which new trade entry will be added |

\* Refer to 3.1.3 for further information.

### 3.2.3 volumeWeightedStockPrice

|  |  |  |
| --- | --- | --- |
| Details | Value | Other Details |
| Purpose | Calculate Volume Weighted Stock Price | |
| Return Type | Object | Varies according to error code or completion of task. Returns BigDecimal upon success |
| Visibility | Public |  |
| Static | Yes |  |
| Exception Handling | Yes |  |

Parameter List:

|  |  |  |
| --- | --- | --- |
| Parameter | Type | Other Details |
| stockSymbol | String | Stock Symbol of Trade for calculation |
| currentTime | Timestamp | Current time |
| tradeList | List<Trade> | List of Trades from which data for calculation will be collected |
| volumeWeightTypeObj | VolumeWeightType\* | Calculation for all trades or only ones which occurred up to 5 minutes prior to current time |

\* Refer to 3.1.3 for further information.

### 3.2.4 geometricMean

|  |  |  |
| --- | --- | --- |
| Details | Value | Other Details |
| Purpose | Calculate GBCE All Share Index using geometric mean of Volume Weighted Stock Price for all trades | |
| Return Type | Object | Varies according to error code or completion of task. Returns BigDecimal upon success |
| Visibility | Public |  |
| Static | Yes |  |
| Exception Handling | Yes |  |

Parameter List:

|  |  |  |
| --- | --- | --- |
| Parameter | Type | Other Details |
| stockList | List<Stocks> | Lists of all listed stocks used to gather stock symbols |
| tradeList | List<Trade> | List of Trades from which data for calculation will be collected |

### 3.2.5 root

|  |  |  |
| --- | --- | --- |
| Details | Value | Other Details |
| Purpose | Calculate nth root of a BigDecimal | |
| Return Type | Object | Varies according to error code or completion of task. Returns BigDecimal upon success |
| Visibility | private |  |
| Static | Yes |  |
| Exception Handling | Yes |  |

Parameter List:

|  |  |  |
| --- | --- | --- |
| Parameter | Type | Other Details |
| nthRoot | List<Stocks> | Nth root value |
| targetValue | BigDecimal | Value for which nth root is to be calculated |

# 4. Production Notes

## 4.1 Initial Thoughts

In line with TDD, create Junit Test cases for the 5 basic functionalities and run the test. The tests will initially fail for the values entered as the functionality does not exist yet.

Proceed in similar style with incremental changes in unit tests till base functionality tests are successful, as a consequence the BOs will also be created.

## 4.2 Refactoring Step 1

As the more complex testing functions would deal with lists of BOs, a @beforeClass function is created for JUnit to populate the required test data (randomly generated) into such lists.

Assumption: The utility functions will receive BOs as parameters.

These tests would be used to refine the utility functions further.

Note 1: Certain changes had to be made on asserted values due to the different rounding off method used by Excel and BigDecimal. From a production point of view, this should not cause any issues as BigDecimal is compatible with standard databases and the true value remains unaffected.

Note 2: Rounding scale for BigDecimal is set to 20 with HALF\_UP rounding.

## 4.3 Refactoring Step 2

The functionalities for the calculation of the dividend yield and the PE ratio can be combined to reduce the lines of code. Thus, I am combining the code by adding an additional identifier/parameter for the functionality.

Certain tests will become redundant and will be removed.

Assumption 1: The formula given for P/E Ratio is Price/Dividend, I am assuming it refers to the table column *Last Dividend*.

## 4.4 Refactoring Step 3

For the functionality to add a record,

Assumption: Any delay between reading the trade and adding it with the time stamp to the records is negligible at a business level.

Assumption 2: There is no check for the Trade record list being empty because it can be the first trade of the day which is unclear in the problem statement.

Note 1: For the purposes of testing, records will be added with timestamp as per test data; Following which it will be checked if it was added successfully and it may be removed to maintain integrity of test data for the remaining functionalities.

Note 2: Any failure scenarios in Geometric Mean functionality are already covered by the Volume Weighted Stock Price functionality

## 4.5 Refactoring Step 4

For the calculation of Volume Weighted Stock Price Average functionality,

Adding a flag to calculate the same for past 5 minutes or total time.

## 4.6 Final Step

Coding complete. Bringing code up to production level.

# Appendix

## Appendix A: Basic Test Information

### Stock BO Test Data



### Trade BO Test Data



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Stock Symbol** | **Quantity** | **Time Stamp** | **Trade Type** | **Price** |
| ALE | 7981112 | 2016-05-12 10:18:23.575 | SELL | 94 |
| ALE | 5903563 | 2016-05-12 10:18:59.348 | BUY | 108 |
| ALE | 4341988 | 2016-05-12 10:16:58.320 | SELL | 322 |
| ALE | 8523778 | 2016-05-12 10:12:49.372 | BUY | 370 |
| ALE | 4086365 | 2016-05-12 10:13:9.232 | SELL | 448 |
| ALE | 2035940 | 2016-05-12 10:20:0.939 | BUY | 608 |
| ALE | 1727767 | 2016-05-12 10:17:21.595 | SELL | 627 |
| ALE | 2074092 | 2016-05-12 10:20:45.558 | BUY | 951 |
| ALE | 3809724 | 2016-05-12 10:14:15.211 | SELL | 1297 |
| ALE | 7656676 | 2016-05-12 10:14:15.33 | BUY | 1301 |
| ALE | 6677379 | 2016-05-12 10:18:40.990 | SELL | 1444 |
| ALE | 3292478 | 2016-05-12 10:10:2.886 | BUY | 1647 |
| ALE | 2269539 | 2016-05-12 10:20:25.788 | BUY | 1740 |
| ALE | 5905717 | 2016-05-12 10:14:36.577 | SELL | 1827 |
| ALE | 2839557 | 2016-05-12 10:20:26.884 | BUY | 1842 |
| COFF | 4786313 | 2016-05-12 10:12:54.401 | SELL | 47 |
| COFF | 2883751 | 2016-05-12 10:19:15.627 | SELL | 136 |
| COFF | 1009356 | 2016-05-12 10:19:55.551 | BUY | 227 |
| COFF | 4097126 | 2016-05-12 10:20:59.510 | SELL | 291 |
| COFF | 8867293 | 2016-05-12 10:16:52.595 | BUY | 529 |
| COFF | 8597323 | 2016-05-12 10:20:19.834 | SELL | 550 |
| COFF | 6115446 | 2016-05-12 10:20:15.597 | BUY | 699 |
| COFF | 6648011 | 2016-05-12 10:16:56.276 | SELL | 747 |
| COFF | 7874250 | 2016-05-12 10:16:18.246 | BUY | 856 |
| COFF | 8430353 | 2016-05-12 10:15:29.142 | BUY | 961 |
| COFF | 6181950 | 2016-05-12 10:18:6.180 | BUY | 1038 |
| COFF | 3600458 | 2016-05-12 10:10:39.798 | BUY | 1185 |
| COFF | 744152 | 2016-05-12 10:17:32.28 | BUY | 1280 |
| COFF | 8921900 | 2016-05-12 10:11:15.384 | SELL | 1345 |
| COFF | 8593994 | 2016-05-12 10:18:18.69 | SELL | 1391 |
| COFF | 1249112 | 2016-05-12 10:18:34.765 | BUY | 1590 |
| COFF | 711399 | 2016-05-12 10:18:13.560 | SELL | 1593 |
| COFF | 1356447 | 2016-05-12 10:17:46.307 | BUY | 1656 |
| COFF | 6545200 | 2016-05-12 10:17:36.813 | SELL | 1683 |
| COFF | 5618826 | 2016-05-12 10:11:1.48 | BUY | 1699 |
| COFF | 7534787 | 2016-05-12 10:18:35.603 | BUY | 1809 |
| COK | 7020694 | 2016-05-12 10:10:40.985 | SELL | 72 |
| COK | 3267038 | 2016-05-12 10:10:35.448 | SELL | 149 |
| COK | 3928957 | 2016-05-12 10:10:20.357 | SELL | 248 |
| COK | 1284783 | 2016-05-12 10:15:15.920 | BUY | 340 |
| COK | 6267557 | 2016-05-12 10:12:39.178 | SELL | 473 |
| COK | 5060683 | 2016-05-12 10:10:39.518 | SELL | 513 |
| COK | 7412279 | 2016-05-12 10:20:6.687 | BUY | 693 |
| COK | 4922249 | 2016-05-12 10:16:0.607 | BUY | 754 |
| COK | 9629185 | 2016-05-12 10:20:30.80 | SELL | 822 |
| COK | 7371953 | 2016-05-12 10:19:8.986 | SELL | 890 |
| COK | 3603076 | 2016-05-12 10:14:35.916 | BUY | 1113 |
| COK | 7039350 | 2016-05-12 10:11:34.910 | BUY | 1275 |
| COK | 4621879 | 2016-05-12 10:12:55.842 | SELL | 1288 |
| COK | 2866043 | 2016-05-12 10:17:11.215 | BUY | 1414 |
| COK | 711291 | 2016-05-12 10:13:36.988 | SELL | 1429 |
| COK | 6895608 | 2016-05-12 10:10:40.830 | BUY | 1468 |
| COK | 711401 | 2016-05-12 10:20:15.562 | SELL | 1856 |
| COK | 6038258 | 2016-05-12 10:13:39.378 | SELL | 1891 |
| GIN | 4662350 | 2016-05-12 10:17:17.783 | BUY | 5 |
| GIN | 2601751 | 2016-05-12 10:13:35.209 | SELL | 207 |
| GIN | 6164652 | 2016-05-12 10:20:20.551 | SELL | 336 |
| GIN | 4271911 | 2016-05-12 10:14:27.77 | BUY | 361 |
| GIN | 9282799 | 2016-05-12 10:10:1.276 | SELL | 365 |
| GIN | 6457256 | 2016-05-12 10:15:37.90 | BUY | 530 |
| GIN | 8381522 | 2016-05-12 10:17:10.654 | BUY | 781 |
| GIN | 3230468 | 2016-05-12 10:16:53.987 | SELL | 800 |
| GIN | 4682118 | 2016-05-12 10:13:2.947 | SELL | 1002 |
| GIN | 1324715 | 2016-05-12 10:15:55.728 | SELL | 1036 |
| GIN | 2903430 | 2016-05-12 10:17:23.175 | SELL | 1039 |
| GIN | 8226066 | 2016-05-12 10:17:23.449 | BUY | 1116 |
| GIN | 6026822 | 2016-05-12 10:18:32.506 | SELL | 1141 |
| GIN | 3207832 | 2016-05-12 10:17:30.478 | BUY | 1311 |
| GIN | 2900029 | 2016-05-12 10:13:12.386 | BUY | 1386 |
| GIN | 6545340 | 2016-05-12 10:20:27.161 | BUY | 1556 |
| GIN | 8341867 | 2016-05-12 10:14:30.353 | SELL | 1557 |
| GIN | 5590394 | 2016-05-12 10:17:14.856 | SELL | 1823 |
| GIN | 6379867 | 2016-05-12 10:19:22.193 | BUY | 1938 |
| JAE | 3509099 | 2016-05-12 10:14:4.983 | SELL | 158 |
| JAE | 2333425 | 2016-05-12 10:16:0.197 | BUY | 216 |
| JAE | 6490833 | 2016-05-12 10:17:49.525 | SELL | 249 |
| JAE | 6900281 | 2016-05-12 10:10:29.130 | BUY | 374 |
| JAE | 1429581 | 2016-05-12 10:20:51.879 | SELL | 466 |
| JAE | 3857546 | 2016-05-12 10:16:58.56 | BUY | 471 |
| JAE | 2587049 | 2016-05-12 10:18:33.675 | SELL | 917 |
| JAE | 3675453 | 2016-05-12 10:14:34.861 | SELL | 1008 |
| JAE | 1993534 | 2016-05-12 10:11:6.284 | BUY | 1010 |
| JAE | 3779535 | 2016-05-12 10:10:27.596 | SELL | 1036 |
| JAE | 2771876 | 2016-05-12 10:15:57.532 | SELL | 1047 |
| JAE | 8906929 | 2016-05-12 10:15:43.200 | SELL | 1166 |
| JAE | 2193555 | 2016-05-12 10:20:51.972 | BUY | 1323 |
| JAE | 8438009 | 2016-05-12 10:17:29.493 | SELL | 1475 |
| JAE | 6933692 | 2016-05-12 10:12:55.678 | BUY | 1594 |
| JAE | 9110798 | 2016-05-12 10:15:11.635 | SELL | 1742 |
| JAE | 5208174 | 2016-05-12 10:18:54.116 | SELL | 1880 |
| JAE | 7466457 | 2016-05-12 10:12:20.870 | BUY | 1918 |
| JOE | 9503683 | 2016-05-12 10:11:38.892 | BUY | 174 |
| JOE | 8925985 | 2016-05-12 10:20:49.134 | SELL | 325 |
| JOE | 9799037 | 2016-05-12 10:12:1.710 | BUY | 428 |
| JOE | 9537097 | 2016-05-12 10:11:30.215 | BUY | 474 |
| JOE | 9164239 | 2016-05-12 10:18:21.848 | SELL | 667 |
| JOE | 4904716 | 2016-05-12 10:16:13.229 | SELL | 745 |
| JOE | 3581922 | 2016-05-12 10:16:4.883 | SELL | 789 |
| JOE | 7117122 | 2016-05-12 10:14:35.410 | SELL | 807 |
| JOE | 7889259 | 2016-05-12 10:15:51.803 | SELL | 1044 |
| JOE | 1887795 | 2016-05-12 10:13:10.373 | SELL | 1215 |
| JOE | 4665913 | 2016-05-12 10:17:25.898 | SELL | 1225 |
| JOE | 3264200 | 2016-05-12 10:12:48.765 | SELL | 1245 |
| JOE | 6411300 | 2016-05-12 10:17:54.609 | BUY | 1355 |
| JOE | 4619390 | 2016-05-12 10:16:26.95 | SELL | 1418 |
| JOE | 6565392 | 2016-05-12 10:18:42.317 | SELL | 1560 |
| JOE | 8991230 | 2016-05-12 10:20:44.377 | BUY | 1574 |
| JOE | 3181305 | 2016-05-12 10:20:18.707 | BUY | 1585 |
| JOE | 1084410 | 2016-05-12 10:13:53.862 | SELL | 1626 |
| JOE | 7351319 | 2016-05-12 10:11:56.560 | BUY | 1691 |
| JOE | 2507192 | 2016-05-12 10:12:55.499 | SELL | 1951 |
| PEP | 4348363 | 2016-05-12 10:15:50.589 | BUY | 127 |
| PEP | 5959893 | 2016-05-12 10:10:44.479 | SELL | 143 |
| PEP | 9168551 | 2016-05-12 10:12:54.579 | BUY | 278 |
| PEP | 225290 | 2016-05-12 10:12:45.959 | BUY | 297 |
| PEP | 9567109 | 2016-05-12 10:10:15.355 | BUY | 457 |
| PEP | 5419655 | 2016-05-12 10:20:59.736 | SELL | 844 |
| PEP | 6729518 | 2016-05-12 10:17:55.152 | BUY | 913 |
| PEP | 6117391 | 2016-05-12 10:19:28.218 | SELL | 920 |
| PEP | 9588444 | 2016-05-12 10:14:30.966 | BUY | 977 |
| PEP | 5486770 | 2016-05-12 10:16:25.412 | BUY | 1065 |
| PEP | 5973612 | 2016-05-12 10:15:40.406 | SELL | 1163 |
| PEP | 1278778 | 2016-05-12 10:13:13.615 | BUY | 1288 |
| PEP | 461727 | 2016-05-12 10:16:48.829 | BUY | 1296 |
| PEP | 428833 | 2016-05-12 10:19:55.376 | BUY | 1449 |
| PEP | 3358662 | 2016-05-12 10:15:26.480 | SELL | 1605 |
| PEP | 3407938 | 2016-05-12 10:16:39.384 | BUY | 1641 |
| PEP | 92571 | 2016-05-12 10:11:51.498 | BUY | 1650 |
| PEP | 3111524 | 2016-05-12 10:13:3.386 | BUY | 1871 |
| PEP | 9271804 | 2016-05-12 10:18:23.458 | SELL | 1985 |
| POP | 282943 | 2016-05-12 10:14:24.887 | SELL | 76 |
| POP | 4598776 | 2016-05-12 10:20:11.501 | SELL | 76 |
| POP | 9163592 | 2016-05-12 10:18:42.65 | SELL | 140 |
| POP | 9468940 | 2016-05-12 10:16:8.808 | SELL | 253 |
| POP | 5865105 | 2016-05-12 10:13:4.29 | SELL | 404 |
| POP | 2811584 | 2016-05-12 10:10:45.842 | SELL | 554 |
| POP | 6740101 | 2016-05-12 10:20:40.69 | SELL | 592 |
| POP | 5033601 | 2016-05-12 10:16:49.443 | SELL | 696 |
| POP | 8324967 | 2016-05-12 10:12:23.343 | SELL | 738 |
| POP | 5229303 | 2016-05-12 10:16:56.237 | SELL | 755 |
| POP | 3794936 | 2016-05-12 10:16:51.757 | SELL | 765 |
| POP | 1304876 | 2016-05-12 10:16:4.689 | BUY | 881 |
| POP | 5837386 | 2016-05-12 10:13:8.773 | SELL | 1016 |
| POP | 6808492 | 2016-05-12 10:18:50.410 | SELL | 1062 |
| POP | 1528404 | 2016-05-12 10:11:32.856 | SELL | 1142 |
| POP | 6235217 | 2016-05-12 10:10:49.406 | SELL | 1228 |
| POP | 487197 | 2016-05-12 10:16:57.245 | BUY | 1428 |
| POP | 714563 | 2016-05-12 10:19:0.897 | SELL | 1531 |
| POP | 1347633 | 2016-05-12 10:10:38.954 | BUY | 1636 |
| POP | 4002849 | 2016-05-12 10:15:8.283 | SELL | 1805 |
| POP | 895148 | 2016-05-12 10:12:41.79 | SELL | 1885 |
| POP | 1963444 | 2016-05-12 10:16:11.471 | SELL | 1996 |
| TEA | 2261593 | 2016-05-12 10:12:28.944 | BUY | 14 |
| TEA | 3970511 | 2016-05-12 10:11:57.978 | BUY | 60 |
| TEA | 5117422 | 2016-05-12 10:16:45.145 | BUY | 177 |
| TEA | 43259 | 2016-05-12 10:14:13.551 | SELL | 237 |
| TEA | 7219828 | 2016-05-12 10:13:51.656 | BUY | 383 |
| TEA | 4214571 | 2016-05-12 10:18:4.143 | SELL | 411 |
| TEA | 441269 | 2016-05-12 10:19:27.428 | SELL | 552 |
| TEA | 3170153 | 2016-05-12 10:13:28.528 | SELL | 686 |
| TEA | 9509647 | 2016-05-12 10:19:57.628 | SELL | 1076 |
| TEA | 1257550 | 2016-05-12 10:16:32.465 | SELL | 1220 |
| TEA | 1767170 | 2016-05-12 10:15:18.809 | SELL | 1253 |
| TEA | 2406943 | 2016-05-12 10:10:2.507 | SELL | 1294 |
| TEA | 2375580 | 2016-05-12 10:17:2.157 | BUY | 1305 |
| TEA | 7143371 | 2016-05-12 10:11:23.250 | SELL | 1306 |
| TEA | 4769687 | 2016-05-12 10:19:2.908 | SELL | 1329 |
| TEA | 2206513 | 2016-05-12 10:14:18.748 | SELL | 1430 |
| TEA | 3245213 | 2016-05-12 10:14:14.480 | BUY | 1606 |
| TEA | 327535 | 2016-05-12 10:11:30.65 | SELL | 1767 |
| TEA | 5094014 | 2016-05-12 10:18:39.108 | SELL | 1802 |
| TEA | 6432520 | 2016-05-12 10:13:43.930 | BUY | 1825 |
| TEA | 3761788 | 2016-05-12 10:19:19.125 | BUY | 1948 |
| WAT | 6848807 | 2016-05-12 10:17:50.164 | BUY | 150 |
| WAT | 8590479 | 2016-05-12 10:12:13.550 | BUY | 268 |
| WAT | 4633062 | 2016-05-12 10:20:52.867 | SELL | 331 |
| WAT | 3973989 | 2016-05-12 10:10:11.656 | BUY | 483 |
| WAT | 9646356 | 2016-05-12 10:14:29.21 | SELL | 514 |
| WAT | 4892535 | 2016-05-12 10:19:18.274 | SELL | 600 |
| WAT | 1405440 | 2016-05-12 10:14:47.241 | SELL | 644 |
| WAT | 6230845 | 2016-05-12 10:11:15.492 | SELL | 743 |
| WAT | 2020532 | 2016-05-12 10:19:24.230 | BUY | 0 |
| WAT | 1097718 | 2016-05-12 10:15:45.278 | SELL | 798 |
| WAT | 6667852 | 2016-05-12 10:20:2.812 | SELL | 810 |
| WAT | 4909009 | 2016-05-12 10:14:45.587 | BUY | 1019 |
| WAT | 3890288 | 2016-05-12 10:15:49.311 | BUY | 1271 |
| WAT | 9331741 | 2016-05-12 10:14:0.395 | SELL | 1289 |
| WAT | 4385288 | 2016-05-12 10:17:26.558 | BUY | 1502 |
| WAT | 7492842 | 2016-05-12 10:15:30.126 | SELL | 1621 |
| WAT | 8619595 | 2016-05-12 10:20:41.233 | BUY | 1683 |
| WAT | 563298 | 2016-05-12 10:11:38.918 | SELL | 1859 |
| WAT | 148297 | 2016-05-12 10:10:58.431 | SELL | 1930 |