**Java Programming Test**

Please produce your solution in the form of a set of classes by completing the attached skeleton maven project. You can use the standard APIs provided in the Java library and optionally google guava (which is already included as a maven dependency). You will require a local Java and Maven installation or an IDE that supports maven projects (like IntelliJ or Eclipse) in order to build the application locally.

The grading of the test will be done not just on the basis of a correct solution, but also on the quality of the code in terms of efficiency, maintainability, testability and test coverage. Any tests should be written in Junit 4.

The submitted solution will also be used to drive the discussions in any future rounds of interviews.

**Calculation of profit and loss**

Please implement the following calculation of profit and loss (PNL) for the set of trades provided in the text file (trades.dat) attached with the skeleton project.

The file contains an individual stocks that have been traded by the specified trader one per line. The file is a comma delimited file and the specification of a single row is as follows,

<trader\_id>,<stock\_symbol>,<quantity>,<buy\_sell>,<price>

**trader\_id** – a string that represents a single trader. The file contains the trades of many traders

**stock\_symbol** – A string that represents a single stock. For example, “IBM”, “MSFT” etc

**quantity** – A decimal value that represents the quantity that was bought or sold. A positive number represents a stock that is bought and a negative number represent the quantity of stocks sold. It is possible to sell a stock that a trader does not own (i.e. short sell)

**buy\_sell** – A “B” for a buy trade and a “S” for a sell trade

**price** – A positive decimal value that represents the price at which the stock was bought or sold in a given currency.

All trades are of the same currency.

Matching of buys and sells can only occur within a single trader and stock. The PNL for a pair of trades is calculated as follows,

PNL = (sell\_trade.quantity x sell\_trade.price) - (buy\_trade.quantity x buy\_trade.price)

A sell trade will be matched with a buy trade on a first-in-first-out basis. For example in the following trades,

T1,IBM,200.0,B,4.50

T1,IBM.200.0,B,4.60

T1,IBM,200,S,4.75

The **3rd** trade will match the **1st** trade. The result of the calculation for the above is as below,

(200.0 x 4.75) – (200.0 x 4.50) = 50 with residual quantity 200

If a trade cannot be fully matched, trades can be split in order to match trades together. See the example below,

T1,IBM,200.0.B, 4.50

T1,IBM,300.0,B,,4.60

T1,IBM,100,S,4.70

T1,IBM,300,S,4.85

(100 x 4.70 – 100 x 4.50) + (100 x 4.85 – 100 x 4.50) + (200 x 4.85 – 200.0 x 4.60) = 105 with residual quantity 100

The 1st trade is split into two with a quantity of 100 each in order to match the 3rd trade. The unmatched buy trade of 100 is then matched with the 4th trade which is split to two with a quantity 100 and 200. Then the 2nd trade is split into two , 200 and 100. After the split trades with quantity of 200 are matched , a quantity of 100 is left unmatched.

The output of the program should be a single row per trade / stock combination with the PNL and residual quantity

For example,

T1,IBM,150.75, 200.0

T1,MSFT,200.0,230.50

T2.IBM,-300.00, -500.0