Stock Exchange Simulator

The exchange is a place where transactions (Trade) are made between buyers and sellers. Buyers want to Buy something, and sellers want to sell something. A purchase/sale transaction is made if the seller and buyer are satisfied with the terms of the transaction - for example, the price and quantity are satisfied with both parties.

Sellers try to sell their goods at the highest possible price, and buyers try to buy at the lowest possible price. On the exchange, the desire to buy or sell is expressed by submitting electronic orders(orders), each of which indicates:

a) a unique order number (ID)
b) buy or sell (side) - instruction to buy or sell ((B) uy or (S) ell)
c) name of security (Instrument)
d) quantity (Qty) - how much to buy / sell
e) buy/sell price (Price) - desire to buy / sell at the price no worse than the one indicated in the application. For buy orders, the price must be greater than or equal to the specified price, for sell orders, the price must be equal to or less than the specified price.

For example:

O, 4, B, Apples, 10, 256.12 -> application ((O)rder) with the number 4 in which someone wants to buy ((B)uy) Apples in the amount of 10 kg at the price of 256.12 rub / kg

or

O, 50, S, Apples, 4, 255.5 -> application ((O)rder) with the number 50 in which someone wants to sell((S) ell) Apples in the amount of 4 kg at the price of 255.5 rub / kg

If the purchase/sale condition satisfies both parties, then a Trade is made. In this example, market conditions satisfy the requirements for making a trade:

a) there is a sufficient amount in this case min(Buy Qty, Sell Qty) = min(10, 4) = 4
b) the buyer's price allows the seller to fulfill the sell condition no worse than his own the sale price (Sell Price >= Buy Price), and the buyer has the opportunity to buy at the price that he stated, which also satisfies the purchase condition-buy is no worse

So the transaction price will be equal to 256.12 and the exchange will generate the final transaction:

T, 1, B,4,50,4,256.12 -> a trade ((T)rade) was made for the purchase with the number 1 for the purchase between orders 4 and 50 in the amount of 4 kg at the price of 256.12

It should be noted that the buyer's price is chosen as the transaction price, since the buyer submitted a wilt earlier than the seller, which follows from the fact that the buyer's request number is less than the seller's wilt number

After the transaction is completed, the seller's order is removed from the exchange, because it was executed in full, and the buyer's order remains on the exchange, but with a balance of 10 - 4 = 6. If there are no conditions for making transactions, then orders remain on the exchange until they are withdrawn. In order to withdraw an order, you must send an order to cancel the order to the exchange:

C, 4 -> cancel ((C)ancel) application number 4

In real-world conditions, there are many sellers and buyers on the exchange, so the order execution algorithm will be as follows:

For a new purchase request:
1) All sales orders are sorted by price in the ascending direction and transactions are made primarily with the lowest price orders in order to meet the best buy rule (at the lowest price). If there are several orders for sale at the same price, then first of all orders with the lowest order number(ID) are executed - FIFO rule (First In First Out)
2) If there are no orders for making transactions, then the purchase order remains on the exchange

For a new sales order:
1) All purchase orders are sorted by price in descending order and transactions are made primarily with orders to buy at the highest price, in order to meet the rule of best sale (at the highest price). If there are several purchase orders at the same price, then first of all orders with the lowest order number(ID) are executed - FIFO rule (First In First Out)
2) If there are no delays for making transactions, then the order for sale remains on the exchange

Example:
1. The following purchase orders are active on the exchange:
O, 1, B, Apples, 10, 250.12
O, 2, B, Apples, 8, 249.10
O, 3, B, Apples, 5, 250.12
O, 4, B, Apples, 15, 250.6
2. A new application is submitted to the exchange for the sale of 31 kg of apples:
O, 5, S, Apples, 31, 248.5
3. Sort purchase orders in descending order by price + order number:
O, 4, B, Apples, 15, 250.6
O, 1, B, Apples, 10, 250.12
O, 3, B, Apples, 5, 250.12
O, 2,B, Apples, 8, 249.10
4. we see that a number of orders meet the conditions for making transactions, so

the exchange algorithm generates transactions:
T,1, B, 4, 5, 15, 250.6
T, 2, B, 1, 5, 10, 250.12
T, 3, B,3, 5, 5, 250.12
T, 4,B, 2, 5, 1, 249.10
5. the purchase order remains on the exchange but with a balance of 7:
O, 2, B, Apples, 7, 249.10
6. all other errands are deleted, because they were fully executed
Task
======
Develop an algorithm for an exchange simulator:
a) the exchange trades only apples
b) the simulator accepts input signals and executes them:
1) O,<OID>,<Side>,<Qty>,<Price> – bid to buy((B) uy) or sell
((S) ell):
•

<OID> - unique bid identifier (number). In fact, the OID
is generated by the exchange itself, so in our case it is always
an increasing number
•
< Side> - buy(B)/sale(S)
•
<Qty> - buy/sell quantity. Integer
•
<Price> - purchase/sale price. Floating-point number - maximum of 2
decimal
places 2) C,<OID> - cancels the previously set limit by <OID>number. If
there is an attempt to cancel a request that has already been canceled or
executed,
the cancel operation is ignored
c) the simulator should generate trades at the output and print:
1) T,<ID>,<Side>,<OID1>,<OID2>,<Trade Qty>,<Trade Price> - trade:
•
<ID> - unique identifier of the trade (a monotonically
increasing number)
•
> - side of the transaction. Buy or Sell
•
<OID1> - ID of the order with which the transaction was made (issued
earlier)
•
<OID2> - ID of the order that initiated the transaction. <OID1>
is always less than <OID2>
•
<Trade Qty> - quantity of purchased / sold goods
•
<Trade Price> - the transaction price
2) X,<OID> - information that the request was canceled
d) the algorithm should be a console application that accepts a
stream (file or stdin) of requests(O) and a cancellation operation(C) and prints
(stdout
or file) transactions(T) and cancellation confirmations(X)
e) for testing, it is suggested to use the file input.txt with requests and
cancellation operations, and a file output.txt with transactions and confirmations
of order cancellations
f) it should be taken into account that the algorithm should work with the
highest
possible performance.
g) something else