**OrderMatchingEngine**

Test trading matching operations engine. **BUY, SELL, MODIFY, CANCEL** operation implantations. Code developed using C# The first column specifies the operation to be done. Our supported operations are:

1. BUY
2. SELL
3. CANCEL
4. MODIFY
5. PRINT

If the first column is **BUY or SELL**, then this line has five columns in total:

1. The second column is the order type, it's either "IOC" or "GFD".
2. The third column is the price you want to buy or sell, it's an integer.
3. The fourth column shows the quantity of that buy or sell, it's an integer. Both the price and quantity are positive numbers.
4. The fifth column is the order id which is an arbitrary word.

If the first column is **CANCEL**, then this line has two columns in total:

1. The second column is the order id. This order needs to be deleted, it can't be traded anymore. If the provided Order ID doesn't exist then do nothing.

If the first column is **MODIFY**, then this line has five columns in total:

1. The second column is the order id, it means that the order needs to be modified.
2. The third column is either BUY or SELL.
3. The fourth column is the new price of that order.
4. The fifth column is the new quantity of that order.

*If the provided Order ID doesn't exist then do nothing.* *(Note: that we can't modify an IOC order type, as we'll mention later.)*

MODIFY is a powerful operation, e.g. a BUY order can be modified to a SELL order: BUY GFD 1000 10 order1 MODIFY order1 SELL 1000 10

If the first column is PRINT, then there'll be no following columns in this line. You're supposed to output the current price book in the following format:

SELL:

price1 qty1

price2 qty2

BUY:

price1 qty1

price2 qty2

Next, to each price a quantity is printed. This quantity represents the sum of all order quantities at the printed price. The prices for both the SELL and BUY sections must be in decreasing order. For example:

SELL:

1010 10

1000 20

BUY:

999 10

800 20

Now let's talk about the order type: The GFD (Good For Day) order types will stay in the order book until it's all been traded. the IOC (Insert Or Cancel) order type means if the order can't be traded immediately, it will be canceled right away. If it's only partially traded, the non-traded part is canceled.

The rule for matching is simple, if there's a price cross meaning someone is willing to buy at a price higher than or equal with the current selling price, these two orders are traded. The matching engine should also print out information about the trade when it occurs.

**For example:**

BUY GFD 1000 10 order1

SELL GFD 900 10 order2

After the second line has been processed, the two orders will have traded and the corresponding output should be as follows: TRADE order1 1000 10 order2 900 10

(NOTE: The "TRADE order1 price1 qty1 order2 price2 qty2" message should be output whenever there's a trade from the matching engine. For every trade that occurs, a corresponding output must be made automatically - the user should not have to rely on the "PRINT" operation.)

The trade output shows information about the two traded orders: order id followed by the order price and the traded quantity. (Note: a real matching engine would only have one traded price, but to make things simple, we output a price for each traded order.) The sequence for order1 and order2 is decided based on which order arrived first.

**For example:**

SELL GFD 900 10 order2

BUY GFD 1000 10 order1

Then the output is: TRADE order2 900 10 order1 1000 10

An important principle for a matching engine is that it should be FAIR: this means that whoever sends the first order gets their order traded first if it meets the criteria.

**For example:**

BUY GFD 1000 10 order1

BUY GFD 1000 10 order2

SELL GFD 900 20 order3

The output will be:

TRADE order1 1000 10 order3 900 10

TRADE order2 1000 10 order3 900 10

There's another interesting thing to **MODIFY** operation, "MODIFY" will lose its original place. So, in this case:

BUY GFD 1000 10 order1

BUY GFD 1000 10 order2

MODIFY order1 BUY 1000 20

SELL GFD 900 20 order3

Because order1 has been modified in the middle, order2 will be in first place followed by order1, so the output will be: TRADE order2 1000 10 order3 900 10 TRADE order1 1000 10 order3 900 10

Example 1:

BUY GFD 1000 10 order1

PRINT

Output:

SELL:

BUY:

1000 10

Example 2:

BUY GFD 1000 10 order1

BUY GFD 1000 20 order2

PRINT

Output:

SELL:

BUY:

1000 30

Example 3:

BUY GFD 1000 10 order1

BUY GFD 1001 20 order2

PRINT

Output:

SELL:

BUY:

1001 20

1000 10

Example 4:

BUY GFD 1000 10 order1

SELL GFD 900 20 order2

PRINT

Output:

TRADE order1 1000 10 order2 900 10

SELL:

900 10

BUY:

Example 5:

BUY GFD 1000 10 ORDER1

BUY GFD 1010 10 ORDER2

SELL GFD 1000 15 ORDER3

Output:

TRADE ORDER2 1010 10 ORDER3 1000 10

TRADE ORDER1 1000 5 ORDER3 1000 5