Matching engine

The goal of this short coding exercise is to build a simple engine that matches buy and sell orders of the same product. An individual order will have the following properties:

- **`orderId: long`** - unique id for this order. Guaranteed by source system to be unique
- **`product: String`** - key for the financial product the customer wishes to trade. Two orders are for the same product if `order1.product.equals(order2.product)`
- **`customerSide: Side`** - BUY if the customer wishes to buy the product, SELL if they wish to sell
- **`size: long`** - The amount of the order in AUD cents. For new orders, it will always be a non-zero positive value.

Your matching engine will be delivered a series of orders. For each order you will update the internal state of unmatched orders, and output any orders that match. Two orders match if they are for the same product and the customerSide value differs (so one is a buy, the other a sell). Partial matches are allowed, so a BUY order of size 5000 will match a SELL order of size 2000. You will be left with a BUY order of size 3000 after the match which could then be matched against other SELL orders.

The order matching rules are:

1. Match the oldest order of the same product, where customerSide differs.
2. Adjust size of both matched orders accordingly removing any order where size is 0 from your state.
3. An order should never have negative size.
4. Continue attempting to match against the rest of orders if new order size > 0
5. If the new order still has a non-zero size after matching is complete, add it to your internal state
6. Output array of ids for orders that were matched, ordered oldest to newest. If no orders matched, return an empty array.

All new orders will be delivered sequentially from the same thread. Your responses are returned synchronously.

YOUR ANSWER

We recommend you take a quick tour of our editor before you proceed. The timer will pause up to 90 seconds for the tour. Start tour

Draft saved 09:07 am | View Code Diff | Java 8

```java
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {
    public static void main(String args[] ) throws Exception {
        /* Enter your code here. Read input from STDIN. Print output to STDOUT */
    }
}
```

Line: 11 Col: 2

Test against custom input

Run Code | Submit code & Continue

(You can submit any number of times)

Download sample test cases *The input/output files have Unix line endings. Do not use Notepad to edit them on windows.*