Marvelsoft Limited
88 Connaught Road, Central, Hong Kong
www.marvelsoft.net

Development task

Introduction

Exchanges (or trading venues) are entities that ensure fair and orderly trading of securities, commodities, derivatives and other financial instruments. When referring to an exchange in this document, we're not including the crypto exchanges. Normal exchanges are quite strict with regards to the trading rules.

Financial instruments are assets that can be traded. We can, for example, trade AAPL instrument on the NASDAQ stock exchange. This means that we can trade the stocks of Apple Inc. on a given exchange. Stocks represent one of asset classes for financial instruments.

The process of trading on a financial instrument is driven by supply and demand. This supply and demand is represented by the concept called "Order Book". Order Book can be represented using a table that's split in two main parts. The left side of the table is used for demand (it will be further addressed as **bid** side) and right side of the table is used for supply (it will be further addressed as **ask** side). Every row of bid/ask side is consisted of price limits or so called levels. Levels are a cumulative representation of all orders for that price limit. They are consisted of number of orders, cumulative quantity of all those orders and limit price of that level. Format of a level is usually written as: `(5) 15000@45.5` which can be read as: there are `5` orders that accumulate to `15000` quantity at a price of `45.5`. The total amount that this level is worth is `15000x45.5=682500` of whatever currency this instrument is trading in. Levels are sorted in descending order for bid side and in ascending order for ask side. This is because the more money you're willing to pay for a unit of something your bid is considered "better" and you have bigger priority over the lower bid. If two bids have the same price tag, the one that got there first has bigger priority. This is so called price-time priority. For the ask side the logic is inverted for obvious reasons. The example of the order book can be found below:

Bid	Ask
`(6) 60'000@50.00`	`(4) 30'000@50.15`
`(3) 30'000@49.95`	`(3) 50'000@50.20`

Order Book Workflow

Let's say now that we want to send an order to buy `5000` stocks at a price point of `50.05`. This will change the order book and it will look like this:

Bid	Ask
(1) 5'000@50.05	(4) 30'000@50.15
(6) 60'000@50.00	(3) 50'000@50.20
(3) 30'000@49.95	

Our order had no offer to match with so it ended up on the order book. Since it has the best price of all orders on bid side it is put on the top of the order book. This order made no trade so the action is considered **passive**.

Let's consider another example where we want to buy `9000` stocks at a price point of `50.20`. If we take a look into the order book we see that there is no room to place intention on the bid side since the best offer (sell side) is `50.15`. This will result in exchange of shares. Next question is at what price? If prices do not match (as in this case `BUY@50.20` and `SELL@50.15`) price is taken from the order that arrived earlier. So in our particular case it will be `50.15`. After this order, first there will be Trade generated:

Trade: `9000@50.15`

The new order book will be generated with reduced quantity at sell side:

Bid	Ask
(1) 5'000@50.05	(3) 21'000@50.15
(6) 60'000@50.00	(3) 50'000@50.20
(3) 30'000@49.95	

Since the last order we sent created a trade the action is considered **aggressive**.

Another thing that can happen on the order book is that quantity can be reduced without a trade being generated. This happens when someone cancels part of the quantity. This simply refreshes the order book and reduces the quantity of one of the levels without generating a trade.

The Problem

Your task is to regenerate the intentions together with their classification if they were passive, aggressive or a cancel/reduce request. The input to your application will be a raw feed generated by our internal market simulator. This input is json formatted and consists of two messages: book and trade message. When you recognize a passive, aggressive or a reduction of order your application needs to print that in a file in following format:

```
Intention Side Quantity @ Price
```

Intention can be: AGGRESSIVE, PASSIVE or CANCEL. Side can be Buy or Sell.

Input for one instrument that looks like this:


```
{"book":{"symbol":"ABBN", "bid": [], "ask": []}}
{"book":{"symbol":"ABBN", "bid": [{"count":1, "quantity":1300,
"price":50.100000}], "ask": []}}
{"book":{"symbol":"ABBN", "bid": [{"count":1, "quantity":900,
"price":50.120000}, {"count":1, "quantity":1300, "price":50.100000}], "ask":
[]}}
{"book":{"symbol":"ABBN", "bid": [{"count":1, "quantity":900,
"price":50.120000}, {"count":1, "quantity":1300, "price":50.100000}], "ask":
[{"count":1, "quantity":1900, "price":50.140000}]}}
{"book":{"symbol":"ABBN", "bid": [{"count":2, "quantity":1300,
"price":50.120000}, {"count":1, "quantity":1300, "price":50.100000}], "ask":
[{"count":1, "quantity":1900, "price":50.140000}]}}
{"book":{"symbol":"ABBN", "bid": [{"count":3, "quantity":1530,
"price":50.120000}, {"count":1, "quantity":1300, "price":50.100000}], "ask":
[{"count":1, "quantity":1900, "price":50.140000}]}}
{"book":{"symbol":"ABBN", "bid": [{"count":1, "quantity":200,
"price":50.130000}, {"count":3, "quantity":1530, "price":50.120000},
{"count":1, "quantity":1300, "price":50.100000}], "ask": [{"count":1,
"quantity":1900, "price":50.140000}]}}
{"trade":{"symbol":"ABBN", "quantity":200, "price":50.130000}}
{"book":{"symbol":"ABBN", "bid": [{"count":3, "quantity":1530,
"price":50.120000}, {"count":1, "quantity":1300, "price":50.100000}], "ask":
[{"count":1, "quantity":220, "price":50.130000}, {"count":1,
```

```
"quantity":1900, "price":50.140000}]}}
{"book":{"symbol":"ABBN", "bid": [{"count":3, "quantity":1530,
"price":50.120000}, {"count":1, "quantity":1300, "price":50.100000}], "ask":
[{"count":2, "quantity":550, "price":50.130000}, {"count":1,
"quantity":1900, "price":50.140000}]}}
{"book":{"symbol":"ABBN", "bid": [{"count":3, "quantity":1530,
"price":50.120000}, {"count":1, "quantity":1300, "price":50.100000}], "ask":
[{"count":3, "quantity":655, "price":50.130000}, {"count":1,
"quantity":1900, "price":50.140000}]}}
{"book":{"symbol":"ABBN", "bid": [{"count":3, "quantity":1530,
"price":50.120000}, {"count":1, "quantity":1300, "price":50.100000}], "ask":
[{"count":4, "quantity":1245, "price":50.130000}, {"count":1,
"quantity":1900, "price":50.140000}]}}
{"trade":{"symbol":"ABBN", "quantity":220, "price":50.130000}}
{"trade":{"symbol":"ABBN", "quantity":330, "price":50.130000}}
{"trade":{"symbol":"ABBN", "quantity":105, "price":50.130000}}
{"trade":{"symbol":"ABBN", "quantity":345, "price":50.130000}}
{"book":{"symbol":"ABBN", "bid": [{"count":3, "quantity":1530,
"price":50.120000}, {"count":1, "quantity":1300, "price":50.100000}], "ask":
[{"count":1, "quantity":245, "price":50.130000}, {"count":1,
"quantity":1900, "price":50.140000}]}}
```


Output should look like this:

```
PASSIVE BUY 1300 @ 50.10
PASSIVE BUY 900 @ 50.12
PASSIVE SELL 1900 @ 50.14
PASSIVE BUY 400 @ 50.12
PASSIVE BUY 230 @ 50.12
PASSIVE BUY 200 @ 50.13
AGGRESSIVE SELL 420 @ 50.13
PASSIVE SELL 330 @ 50.13
PASSIVE SELL 105 @ 50.13
PASSIVE SELL 590 @ 50.13
AGGRESSIVE BUY 1000 @ 50.13
```

Unfortunately, in the example above there is no cancel intention. Please notice, that every message (both trade and book) carries a symbol property. This tells you the financial instrument for which the event is intended.

Goal & Deliverables

Your job is to parse out the input.json file attached alongside this document. This input contains events for multiple instruments. One file should be generated for every instrument. The content of every file should be an output that is explained in the previous section.

The application must be multi-threaded and input file must be parsed and normalized before writing to the output files. The time it took your application to recreate these events, measured in milliseconds, must be written to the log file or console output.

Task deliverables:

- Basic development and runtime environment specifications (compiler version, IDE used, target OS)
- Class and sequence diagrams (UML)
- The source code
- The unit tests code
- The build system files
- Any additional documentation or explanation you find necessary