**# Bitcoin Order Book**

**## Instructions**

In the initial commit to this project, you have been provided with a basic project setup as well as a set of tests which currently fail. Your task is to create a `reconcileOrder` function which will properly reconcile an incoming cryptocurrency order given the existing book of orders.

Working in a branch called answer, implement your function to get your tests to pass. Once you are done, push your branch up to GitHub, create a pull request, and request review from the instructor.

**## Requirements**

An order book tracks all buy and sell requests from traders for Bitcoin assets at various price levels. When a trader submits a trade it is either fulfilled immediately, if there is a matching order in the book, or it is added to the book to be fulfilled later.

Assume starting with an empty order book, let's walkthrough several scenarios.

**\*\*Buy 15 BTC at 5800:\*\*** The order book is empty so this order cannot be fulfilled immediately. This order is added to the book.

**\*\*Buy 10 BTC at 6000:\*\*** The order book only has buy orders and no sell orders so this order cannot be fulfilled immediately. This order is added to the book.

**\*\*Sell 5 BTC at 6100:\*\*** The order book has two buy orders but neither match this as the prices mismatch. This order is added to the book.

**\*\*Sell 10 BTC at 6000:\*\*** The order book has a buy order for 10 BTC at 6000 meaning this order can be fulfilled immediately. This order is executed and the corresponding buy is removed from the order book.

**\*\*Buy 2 BTC at 6100:\*\*** The order book has a sell order at this price point for more than this order's quantity. This means this order can be fulfilled immediately and the corresponding sell order can be reduced by 2 as it has been partially fulfilled but the remainder stays in the book.

**\*\*Sell 25 BTC at 5800:\*\*** The order book has a buy order at this price point but for a smaller quantity. This order can be partially fulfilled, the corresponding buy order is used up and is removed from the order book and the remainder of this order (Sell 10 BTC at 5800) is added to the book.

**## SLA Note**

In order to fairly distribute order reconciliation, order fulfillment should take from the front of the existing book and partially fulfilled orders should be placed at the end of the existing book. This is true of incoming orders as well as existing orders.

\* Incoming Example: an incoming order with a quantity of 15 is partially fulfilled by an existing order with a quantity of 10. The order with the remaining quantity of 5 should be placed at the end of the array.

\* Existing Example: an incoming order with a quantity of 10 is matched fulfilled by an existing order with a quantity of 15. The order with the remaining quantity of 5 should be placed at the end of the array.

**## Extra Credit**

No one is ever upset about a good deal. We want to adjust the code so that it will fulfill orders that are not exact matches but would result in both parties benefitting.

For example, assume the book contains an order to buy 15 BTC at 6000. When we receive a sell order for 15 BTC at 5900 we should fulfill this order given that the buyer will pay less (5900 vs 6000) and the seller will receive the amount requested (5900).

The inverse however would not be true. Given a book with a buy 15 BTC at 5900, when we receive a sell order for 15 BTC at 6000 we should not fulfill this order since it would result in the seller getting less (5900 vs 6000).

There are two tests associated with this extra created that are currently being skipped. Remove the `.skip` on these tests so they run when beginning this work.