Test #1

This test consists of 1 problem, with three parts.

Problem 1.

In C++, implement an order book for a single symbol. You are welcome to use any STL or Boost functionality, but please do not use other third-party libraries. You can assume an order is defined as:

```
struct Order
{
  int id;
  char side;    // 'B' for bid, 'O' for offer
  double price;
  int quantity;
};
```

Part A.

Please write a class, `Book`, that efficiently supports the following operations:

1. Given an `Order`, add a new order to the book.
2. Given an `id`, remove the order with that ID from the book.
3. Given an `id` and `quantity` value, modify the order with that ID to have the new quantity.
4. Given a `side` and integer `level`, return the price for that side and level. (For example, with side `B` and level `0`, the method would return the best bid price.)
5. Given a `side` and integer `level`, return the total quantity of all orders for that side and level.

Part B.

For each of the five operations above, please describe the worst-case performance using "Big O" notation. For the purpose of this part you can assume that, before the operation, there are `m` levels each with `n` orders.

Part C.

Please suggest (but do not implement) some additions or changes to the `Order` struct and/or the `Book` API that would make them better suited for use in a real-world, low-latency application.