1). How did you choose your implementation language?

I chose python for two reasons:

* It’s the language that I’m most comfortable with;
* Because it can be used for both scripting and object-oriented programming.

2). What is the time complexity for processing an Add Order message?

The time complexity of processing an Add Order message is O(N log N).

Book().process\_all() takes O(1) time

Both Book().process\_bid() and Book.process\_ask() take O(N log N) time.

Book().build\_order() takes O(1) time.

Book().reduce\_orders() takes O(N log N) time.

Sorting a list of N orders takes O(N log N) time in the worst case.

-For loop will take O(N) time.

- Upper bound of method is O(N log N).

- Comparison conditional takes O(1) time.

- Conditional logging takes O(1) time.

- In the worst case, if all the entries are bids or asks, that means the next entry will be the same; therefore, Book().process\_stock\_option() will process present entries.

3). What is the time complexity for processing a Reduce Order message?

The time complexity of processing a Reduce Order message is O(N log N

Book().process\_all() takes O(1) time.

Book().record\_reduce\_order() takes O(N log N) time.

- handle\_reduce() takes O(1) time

- Both \_calculate\_bid() and \_calculate\_asks() take O(1) time

- Both \_spent*\_*on\_share() and \_share\_size\_gain take O(N log N) time, as they depend on Book().process\_stock\_option().

4). If your implementation were put into production and found to be too slow, what ideas would you try out to improve its performance? (Other than reimplementing it in a different language such as C or C++.)

The idea of distributed systems can be employed. All the processing for bids can be in a different computing environment, while all the processing for asks can run in another environment. Both environments can run parallel to each other. However, there might be some difficulty when it comes to syncing the output from these different routes to compute the final output.