Sridhar Karumuri

1. This application program enables communication between a client and a server to make modifications to the inventory file maintained in the server. In the program, we devised 3 functions that take requests from the client to carry out their functions: a sorting function, a delete function, and an update function. The sorting function returns a sorted version of the inventory file, the update function updates the inventory file on the server, and the delete function deletes an item from the inventory.
2. As I had very little experience with this area, it was tough to learn from scratch as I had only grazed python in the past. By spending time during Thanksgiving break I was able to learn these skills and could easily apply it in the project. Other than this there were only minor normal programming errors.
3. This project taught me many things, like working in a team, and how a real production environment might function. I learned some very useful networking skills and tools like mininet. Working on this project also showed me that a hard task can be easy while working on small simpler parts.
4. We used mininet to create a network setup that had two hosts connected through a switch. For returning sorted versions of the list, we used the built in sort method Python has. It uses the timsort, which takes ideas from both the insertion sort and merge sort. The idea is that it uses subarrays that are already sorted to sort the whole array, making it extremely efficient. There is no guarantee on if inventory.txt is sorted, so we used a linear search when looking for items to update and delete.
5. Right now, the application and protocol can only handle one server and one client at a time and, the receive code in the server is a blocking call, meaning that once a client connects, the server just waits until the client sends it a request. To fix this, instead of waiting on the receive call in the server code, we could create a new thread that handles the communication with that specific client. This would free up the server to listen to new clients and handle multiple requests at the same time.