Task 1A

(a) Description
My hash table has a hash table struct that contains a pointer to a pointer of an entry, and the size of the table. The entry double pointer is initialized to an array the size of the hash table with an entry. Each entry has an ip_address_t as the key, a integer value which is the port number, and a pointer to the next element. The next element allows for the creation of a linked list in the case where there is a conflict and multiple entries are stored at the same index.

My hash function is a simple multiplication of the 4 integers that make up the address and taking the modulus of the size of the table.

(b) The average time to perform a lookup.
In my virtual machine it took an average of 0.300 microseconds per lookup.

(c) An estimated average throughput of a switch built using this routing table.
An estimated throughput of a switch built using this routing table is 1/0.3*10^6 = 3,333,333 packets per second given that you have 40,000 entries in your routing table.

(d) Because of my hash function the worst case would be that 4474 addresses map to each index of the array so that if there are 40,000 entries in the table, it would map to 3 indexes in the table with 4474 37 each. The 4474 is derived from 24 permutations of numbers from ip addresses and the modulus operator from the total possible 256^4 entries (256^4/24/40000). II would have to search through up to n = 4475 entries per lookup to find the one I was looking for. One for finding the correct index, and 4474 others to find the correct key in the linked list.

To simulate the worst case, I created a table that has only 8 indexes and modified the randomness of the ip addresses so that it would place them in the table with 4474 entries per index.

Initializing the Routing Table
Adding 40000 entries to the routing table
Performing 50000 lookups
Time difference: 4.072576 seconds
That is 81.451525 microseconds per lookup