Hao Chen
C950

PART F.

1. A strength of the Nearest Neighbor algorithm is that there are no assumptions about the data distribution which means that it works on all sorts of data that does not require the data to first be filtered and sorted or follow a pattern. Another strength is that it is very simplicit in both practice and in understanding, which makes it easy to apply to almost any situation.
2. The algorithm has met all requirements.
3. Branch and Bound is another algorithm that can be used. Unlike Nearest Neighbor, Branch and Bound explores every single possible route combination and eliminates sub-optimal routes early. It is only really effective for small sample sizes. Another algorithm is Greedy Algorithm which selects the best path at each step of the route. Unlike the nearest neighbor, Greedy will take in other factors such as cost or distance when deciding the best path unlike Nearest Neighbor which only chooses base on the raw distance each point is from each other.

PART G

I would definitely optimize the routes better by partitioning and dividing the addresses by zip code before loading them onto specific trucks. This way all trucks are guaranteed to deliver in the same zip codes where all the locations are generally closer together to each other. Another thing I would do is create a function to automatically load each truck, since as the packages get more and more, it would be unviable to manually divide and load each truck myself.

PART H

The data structure used meets all requirements

1) Linked Lists
    a. Linked lists differ from hash table in that each item in a linked list is connected to another item which makes it easy for quick insertions and deletions, but since it would require traversing the entire list to find specific elements, the time complexity for accessing an element in a linked list is O(n) compared to Hash table which is O(1)

2) Priority Queue
    a. Priority queues can also work and it differs from a Hash table in that priority queues are stacked based on a defined priority element in this case could be a deadline or weight. Priority queues are much slower than hash tables for insertions/deletions taking O(logn) because it requires moving the whole queue in order to delete or insert something into the middle of the queue.