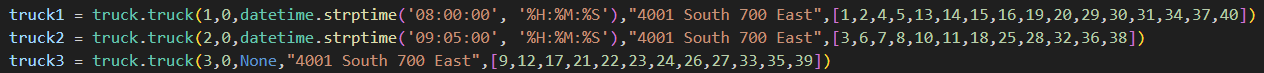
F1. One strength of the nearest neighbor algorithm is the is that it’s easy to implement. Due to the nature of the algorithm, no complex calculations are required since it only considers the current location and looks for the shortest destination after, easily done with simple comparisons operations. Another strength is that it doesn’t require many parameters – just distance. In a similar vein, it keeps the overall implementation of the algorithm simple for future changes and updates.

F2.



* Packages 3,18,36,38 are on truck 2.
* Packages 6, 25, 28, 32 do not leave until 09:05am, accounts for the delay on flight.
* Packages 14,26,20 are delivered with 13,15,19 on truck 1.
* Package 9 address is updated and delivered after 10:20.

A screen shot of a computer

Description automatically generated

* All packages with deadlines are delivered before the deadline (see 3 screenshots in the next page).
* Total truck driving distances are less than 140 miles.

A screenshot of a computer

Description automatically generated

* The function deliver\_trucks() does deliveries for trucks 1 and 2. It then checks which returns first along with what time. It then sets truck 3’s departure time to either 10:20 or the earliest returning truck’s return time whichever is later. It updates package 9’s address data before delivering.

A screen shot of a computer

Description automatically generatedA screen shot of a computer program

Description automatically generatedA screenshot of a computer

Description automatically generated

F3.Brute force algorithm and Dijkstra’s algorithm.

F3a. Brute force algorithm has a much higher time complexity due to it going through all possible paths to find the shortest one. It requires significantly more resources. Dijkstra’s algorithm is more complex than the nearest neighbor algorithm but not as accurate as it looks for the shortest path. It is generally used in navigation systems unlike nearest neighbor which is good for delivery services that need a quick approximate solution.

G. One major difference I would make in a remake of the project would be to load packages into trucks without choosing them manually. This may require some changes to the source of the package notes files or additions to the packages that could capture the requirements of either being delivered with other packages or being on a specific truck. The package would have a field that is limited to certain inputs.

For example, the package could have a variable that denotes the required truck (1 or 2) or be empty to denote that any is fine. Another method would then look through packages with similar requirements and group those together into 1 truck if space permits.

H. The hash table takes the key (ID) input along with the package object created to insert into it.

A screen shot of a computer program

Description automatically generated

A screen shot of a computer program

Description automatically generated

H1. Queues and doubly linked lists are possible alternative data structures.

H1a. Queues are a data structure that follows the first-in-first-out approach. The key difference between queues and hash tables is that the data is organized in a linear manner without using keys, and accessing data requires the searching of data in the order it was inserted. Hash tables have a time complexity of O(1) because you can directly access the item using the matching key, but queues can have a complexity of O(n) due to searching each item in order until the desired is found.

Doubly linked lists are lists that use pointers to point towards the previous and next entries. This allows for traversal in both directions. It has similar drawbacks to queues when compared to hash tables. However, entries can be placed at either the beginning or the end of the structure. Doubly linked lists excel at insertions and deletions at arbitrary locations while hash tables depend on a key to determine where an item ‘s location.