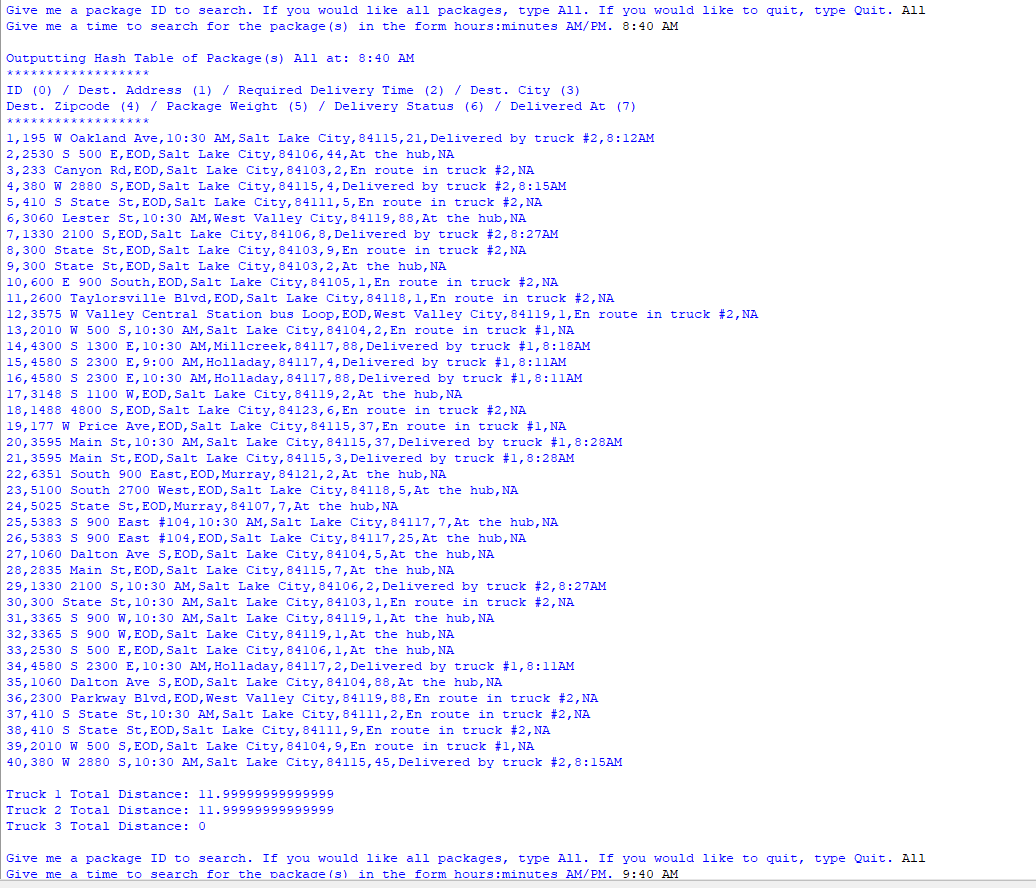
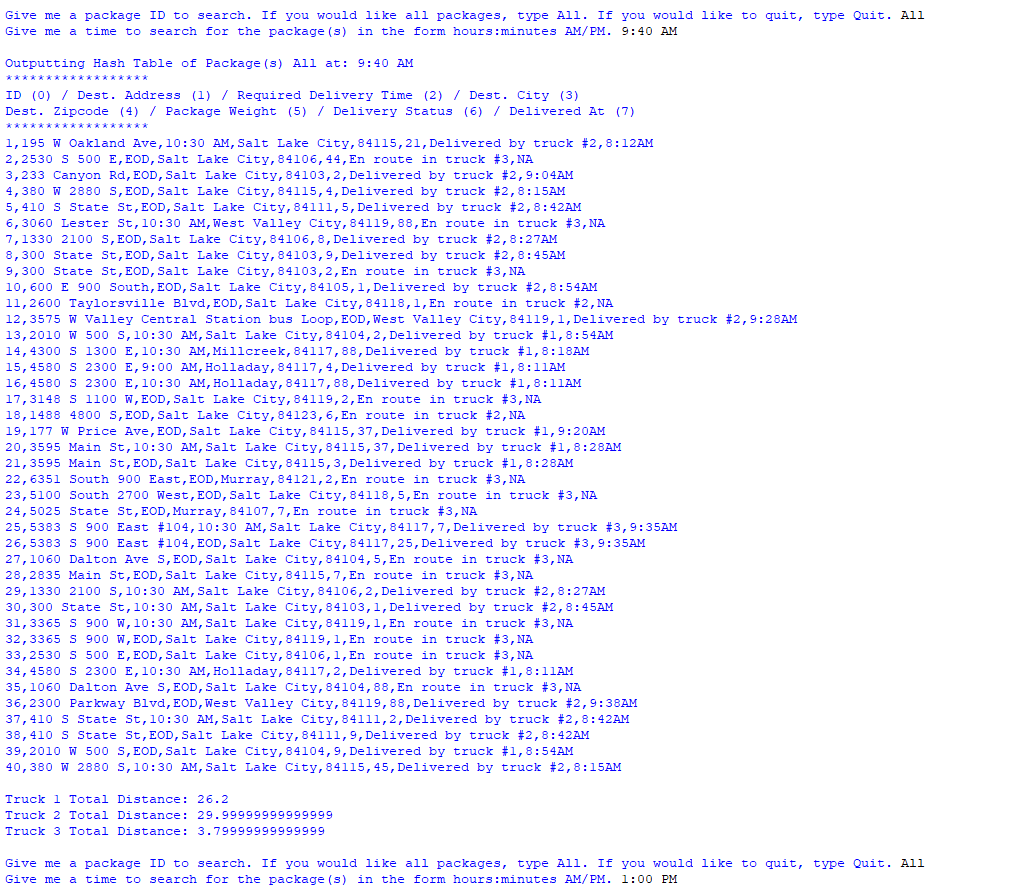
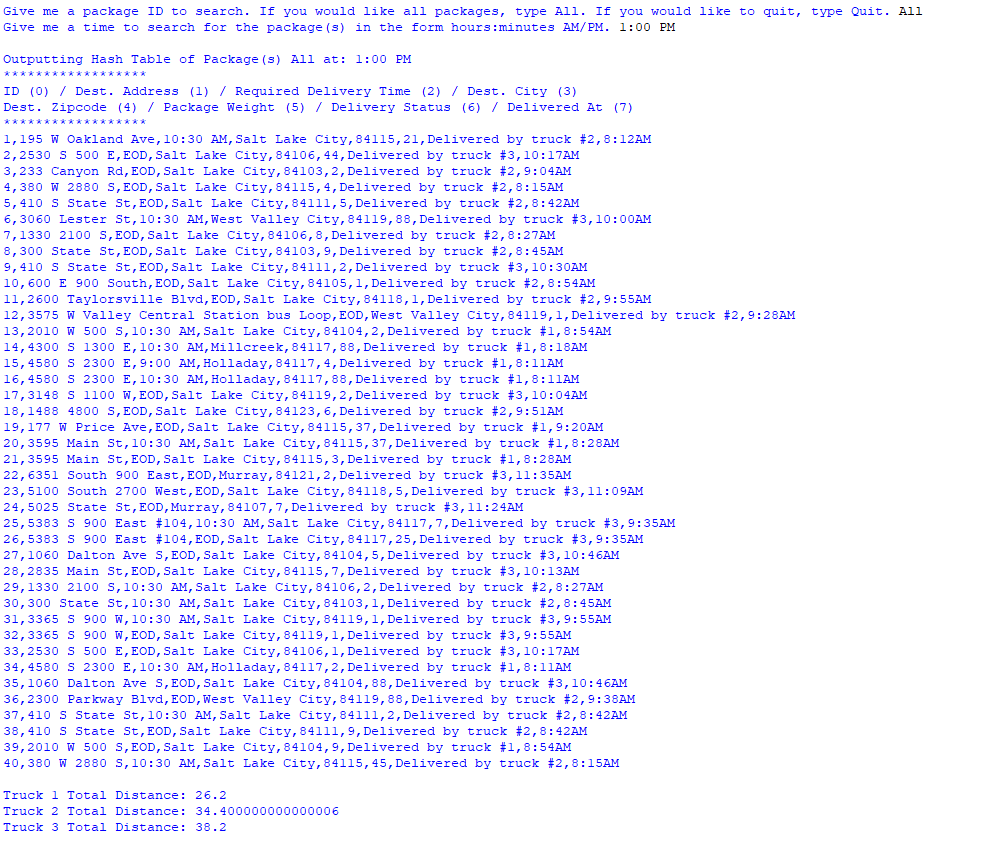
A-C. Included in code zip file.

D.

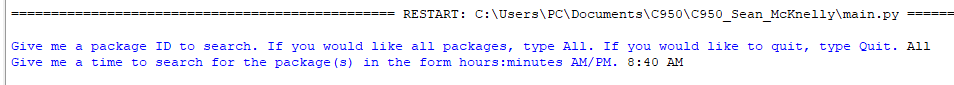




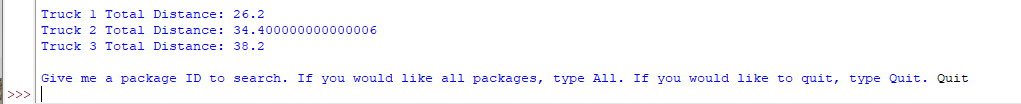


E. Part D (MIDDLE) plus the following screenshots:

BEGINNING:



END:



**F.**

**1. Two or more strengths of the algorithms:**

1. **The greedy algorithm is easy to implement.**
2. **The greedy algorithm is scalable with minor alteration of code.**
3. **The greedy algorithm is easy for another coder to understand with basic DS&A knowledge.**

2. Verify that the algorithm used in the solution meets all requirements in the scenario:

Most scenario requirements demonstrated in part D.

For the assumptions:







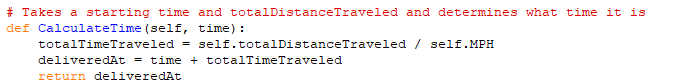




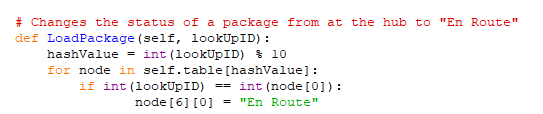
ID uniqueness shown by output in pt. D













Truck 1 returns when it is empty to WGU then the driver begins route on truck 3. There are 3 trucks.





The 8 represents 8 AM

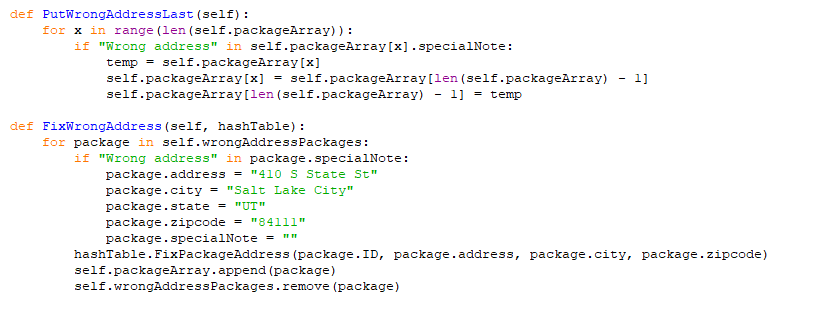


Calculated Accordingly













Code does not account for differences in direction traveled.



Shown in output pt. D

3. Identify two other named algorithms different from the hash table implemented that would fulfill the same requirements.

Nearest Neighbor algorithm

Genetic Algorithm

1. Describe how both algorithms identified in part F3 are different from the algorithm implemented.

Nearest Neighbor – Drawing from GeeksForGeeks (2024), while there are aspects that are similar in my code, the sorting by zipcode in my code does it sequentially, when it could have located the nearest zipcode by average distance within that zip code.

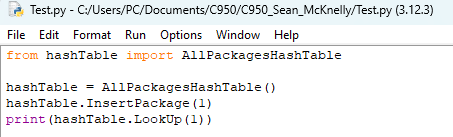
Genetic algorithm – Drawing from Ramez Shendy (2023), my code does not check different potential routes and select the most optimal. Genetic does.

**G. Identify what you would do differently, other than the two algorithms identified in part F3, if you did this project again, including the details of the modifications that would be made.**

**I would use the package class within the hash table. I was under the impression I could not use a package class within the hash table, but after my meeting with Professor Denchy, I now know I can. This would have a better overall design than using an array. The package class would be essentially the same as package.py, but would add fields such as deliveredBy and status (en route, etc.).**

I would also try to make the time complexity more efficient. For instance, there are probably more efficient ways to sort through the matrix of distances.

H. Verify that the data structure used in the solution meets all requirements in the scenario:





1. Identify two other data structures that could have been used.

* Dictionary
* Array

1. Dictionary would have been different because it would have built in the ID with the associated fields.
2. Array would have been different because it would be less efficient to sort through to obtain the proper node.

I. Works Cited:

GeeksForGeeks. (2024, Jan. 25) *K-Nearest Neighbor (KNN) Algorithm*. GeeksForGeeks.org. https://www.geeksforgeeks.org/k-nearest-neighbours/

Shendy, Ramez. (2023, Aug. 5) *Traveling Salesman Problem (TSP) using Genetic Algorithm (Python).* Medium.com. https://medium.com/aimonks/traveling-salesman-problem-tsp-using-genetic-algorithm-fea640713758