Task 2 Written Submission 1

By: Nyla Gaston

Date: 8/24/2023

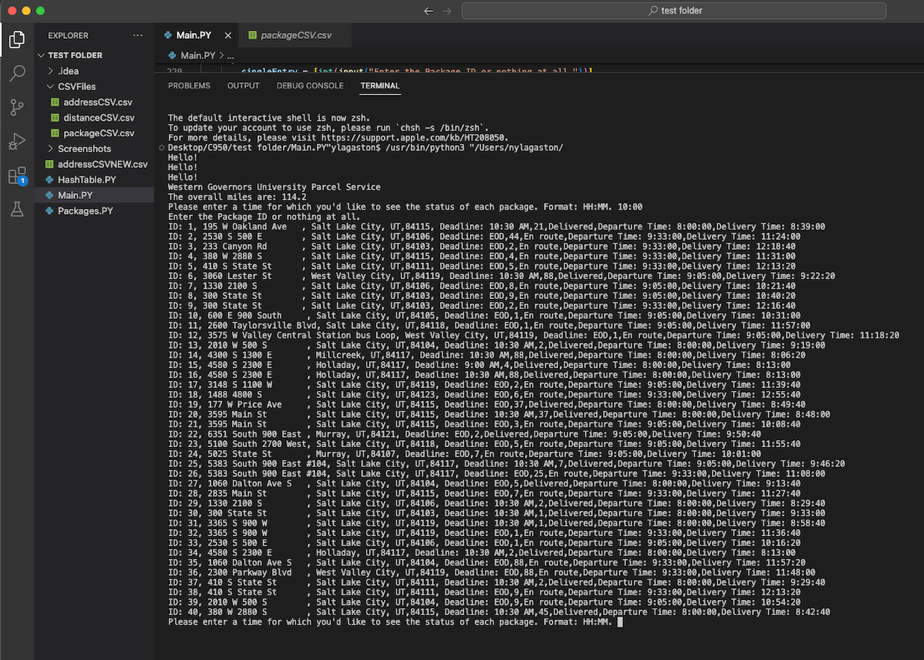
D. Provide an intuitive interface for the user to view the delivery status (including the delivery time) of any package at any time and the total mileage traveled by all trucks.

1. Provide screenshots to show the status of all packages loaded onto each truck at a time between 8:35 a.m. and 9:25 a.m.

A screenshot of a computer program

Description automatically generated

2. Provide screenshots to show the status of all packages loaded onto each truck at a time between 9:35 a.m. and 10:25 a.m.



3. Provide screenshots to show the status of all packages loaded onto each truck at a time between 12:03 p.m. and 1:12 p.m.

A screenshot of a computer program

Description automatically generated

E. Provide screenshots showing successful completion of the code that includes the total mileage traveled by all trucks.

A screenshot of a computer program

Description automatically generated

**F. Justify the package delivery algorithm used in the solution as written in the original program by doing the following:**

**1. Describe two or more strengths of the algorithm used in the solution.**

There are many strengths to the Nearest Neighbor Algorithm. First, is that it is able to handle situations that use multiple classes. For this project, I have at least two classes and that could cause bugs for other algorithms, but the nearest neighbor algorithm can handle this naturally. Secondly, this algorithm is great because it can handle data that isn’t always linear. When there is a question of where to go next and that isn’t clearly laid out this algorithm will not break the code because of non-linear requirements.

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

The nearest neighbor algorithm meets all of the requirements for this scenario because it gets all packages to their destinations on time, keeps the total mileage under 140 miles, and accommodates the packages that need to be delivered with special instructions. For the packages that wouldn’t get to the hub until 9:05 those didn’t leave until 9:05, for those that had to be delivered with other packages, were delivered with other packages, and it doesn’t end until all the packages were delivered.

**3. Identify two other named algorithms that are different from the algorithm implemented in the solution and would meet all requirements in the scenario.**

Two other named algorithms that are different from the implemented algorithm but would meet all requirements for this scenario are Kruskal’s algorithm and Dijkstra’s algorithm.

**a. Describe how both algorithms identified in part F3 are different from the algorithm used in the solution.**

Dijkstra’s algorithm is used for a graphing data structure and the nearest neighbor algorithm is used for classes. Dijsktra’s takes a weighted graph and finds the shortest path from one node to the other nodes on the graph or in this situation finds the shortest distance to the next address. This differs from the nearest neighbor because the nearest neighbor is best for classes and not graphs. While the nearest neighbor algorithm could be used for graphing as well, it worked best with the hash table data structure.

Kruskal’s algorithm is another possible solution that could have worked for delivering the packages. This algorithm would’ve satisfied the requirements, but it was not the most efficient way to solve the problem and fulfill all of the requirements. This algorithm also uses graphing and constantly picks the smallest edge on a weighted graph, but they don’t have to be connected like in Dijkstra’s algorithm. This could cause the truck to go back to the same area a few different times instead of one connected route. This is different from the nearest neighbor algorithm because again the nearest neighbor is a class-based algorithm and also must stay connected on a path where the Kruskal algorithm doesn’t.

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

If I were to do this project over again, I wouldn’t do it in one giant page of code. In order to make it more readable and definitely easier to customize I would put the classes into different files and import them into the Main.PY file. This way, for example, the package class could be modified without affecting the project in its entirety. This would make the coding and debugging process have a kind of sandbox effect where the classes are isolated and independent of each other. I would have a separate file for the hash table, the package class, and the truck class and then put the functions and the algorithm in the Main.PY file.

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

The hash table was the data structure used and it meets all the requirements for the scenario. To start it accurately stores all of the package information. The unique package ID acts as the key for the hash table, and the address, city, state, zip, weight, status, notes, and deadline are stored as values for the matching key. The hash table allows the package information to be malleable for whatever the program needs such as searching for package components, inserting or deleting packages if necessary, and loading the trucks with all the packages according to their requirements.

**1. Identify two other data structures that could meet the same requirements in the scenario.**

The doubly linked list and the queue data structures could have also met the requirements for the scenario.

**a. Describe how each data structure identified in H1 is different from the data structure used in the solution.**

A doubly linked list “is a special type of linked list in which each node contains a pointer to the previous node as well as the next node of the linked list.” (GeeksforGeeks 2023). This data structure is a list of values that uses pointers to go to the next and previous items in the list. This is different from a hash table because a hash table isn’t a list that uses pointers it consists of boxes that hold keys. For a hash table to work there must be key-value pairs, keys are stored in the hash and when called using a search the value for the corresponding key is demonstrated.

A queue is a data structure where data is listed and the first item in the queue is the first to go. Items are inserted at the end of the queue and are removed from the front of the queue. The search for a queue is also less efficient as it must go through all prior entries in the queue until it reaches what it is looking for. This is different from a hash table because items can enter and exit the hash table from any position. Also, since hash tables require key-value pairs, the search will go directly to that key and not take as much time.

Sources:

GeeksforGeeks. (2023, April 19). *Introduction to doubly linked list – data structure and algorithm tutorials*. GeeksforGeeks. https://www.geeksforgeeks.org/data-structures/linked-list/doubly-linked-list/