**Task 2 Written Submission 1**

**By: Vinh Dinh**

**Date: 11/30/2023**

**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. (The delivery status should report the package as at the hub, en route, or delivered. Delivery status must include the time.)**

**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 black screen with white dots

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.**

A black screen with white text

Description automatically generated

**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 black screen with white text

Description automatically generated

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

A black rectangular object with white text

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.**

**Low computational overhead:** The dataset for this project is very small. In turn, the time complexity is relatively low. The computational resources required for this project is minimal making it very efficient and quick.

**Adaptability:** This algorithm has a lot of adaptability and flexibility in its simplicity. It can be adapted to used different distance metrics. If the dataset becomes dynamic and grows over time, it can be incorporated without changing much of the code.

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

All packages were delivered and kept under the 140 miles requirement (103.4 miles). Special instruction packages were accommodated as well as the delivery time of said packages.

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

Kruskal’s and Dijkstra’s algorithm would work in this scenario.

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

Kruskal's algorithm is mainly employed to discover the minimum spanning tree within a weighted connected graph. The approach involves gradual construction, selecting the smallest available edge at each step while ensuring it does not create a cycle in the existing solution. Additionally, Kruskal's algorithm has a marginally greater computational overhead compared to the Nearest Neighbor algorithm. While both are greedy algorithms, they are optimized for different problems. Nearest Neighbor is used in path optimization with Kruskal’s algorithm being used to minimize the cost of connections in a graph ("Kruskal’s Minimum Spanning Tree (MST) Algorithm," GeeksforGeeks, 2012).

Dijkstra's algorithm is employed to determine the most concise route from a source node to all other nodes within a graph with weights. It keeps track of a set of vertices whose shortest distance from the source is identified and adjusts these distances while navigating through the graph. To efficiently choose the next vertex for exploration, it necessitates the use of a priority queue or an equivalent data structure. Generally, its time complexity is higher when contrasted with the Nearest Neighbor algorithm ("Dijkstra’s Shortest Path Algorithm - A Detailed and Visual Introduction," freeCodeCamp.Org, 2020).

**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 had to do this project differently again, I would like to implement a GUI. In Visual Studio Code, I would utilize a GUI library such as Tkinter. I would write a script and import the necessary Tkinter modules. Design the GUI with components like buttons, canvas for drawing, and entry fields for user input.

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

The hash table indeed meets all requirements. It houses the data-key value pairs in buckets for all packaging information. The package ID is the unique key identifier that allows the program to retrieve information from each package. The hash table serves as the structure for the Truck and Package class to coexist and work with one another. Functions that are able to call on the hash table are able to do so.

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

A priority queue and adjacency matrix or list would work

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

**Priority Queue**: A priority queue is essentially what Dijkstra’s algorithm is. It is designed to store elements with associated priorities and allow retrieval of said element based on the highest or lowest priority. It would allow to explore paths and prioritize the best one based on the cost (in our case distance from package to package) ("What Is Priority Queue | Introduction to Priority Queue," GeeksforGeeks, 2014).

**Adjacency Matrix/List:** Adjacency matrix/list would be used to represent the connections between cities in a graph. They would store the information and allow efficient access to the weighted distances between cities. In a matrix it would be in a 2D array representing connections and their weights, while in a list it would be a collection of lists of arrays specifying neighboring cities ("Comparison between Adjacency List and Adjacency Matrix Representation of Graph," GeeksforGeeks, 2020).

**Sources:**

“Kruskal’s Minimum Spanning Tree (MST) Algorithm.” GeeksforGeeks, 30 Oct. 2012, <https://www.geeksforgeeks.org/kruskals-minimum-spanning-tree-algorithm-greedy-algo-2/>.

“Dijkstra’s Shortest Path Algorithm - A Detailed and Visual Introduction.” freeCodeCamp.Org, 28 Sept. 2020, <https://www.freecodecamp.org/news/dijkstras-shortest-path-algorithm-visual-introduction/>.

“What Is Priority Queue | Introduction to Priority Queue.” GeeksforGeeks, 2 July 2014, <https://www.geeksforgeeks.org/priority-queue-set-1-introduction/>.

“Comparison between Adjacency List and Adjacency Matrix Representation of Graph.” GeeksforGeeks, 20 June 2020, <https://www.geeksforgeeks.org/comparison-between-adjacency-list-and-adjacency-matrix-representation-of-graph/>.