**Connor Timpe**

#000976484

C950 Data Structures and Algorithms II

***Section 1: Programming/Coding***

A.  Identify the algorithm that will be used to create a program to deliver the packages and meets all requirements specified in the scenario.

For this project, I chose to implement a greedy algorithm. In the algorithm, all packages in the truck are searched to find the next nearest package. The closest package is then added to an optimized list and removed from the initial list. After all packages are added to the optimized list, and the original list is empty, the new list is returned and so the trucks path is optimized. After loading an optimizing the packages using this algorithm, all packages are able to be delivered in 99.0 miles, and all time deadlines are met.

B.  Write a core algorithm overview, using the sample given, in which you do the following:

1. Comment using pseudocode to show the logic of the algorithm applied to this software solution.

Pseudocode:

While the truck is not empty:

For each package in truck:

Find distance from current location to package

Add package with smallest distance to shortest\_path list and

Remove package from truck

Return shortest\_path list

1. Apply programming models to the scenario.

The current version of this software is limited to the local machine. The data is all pulled from csv files within the project directory. Therefore, there is no need for communication protocols at this time.

1. Evaluate space-time complexity using Big O notation throughout the coding and for the entire program.

The Big O notation for each function is displayed throughout the program in comments. A majority of the program runs in either O(1) or O(N) time. However, the greedy algorithm to optimize the package delivery order has a worst case of O(N^2). Although the effect of this is limited due to the small truck capacity. Given a large increase in number of trucks and capacity of the trucks, additional time for the program to run or optimization of the algorithm may be needed.

1. Discuss the ability of your solution to adapt to a changing market and to scalability.

Currently, the program has three trucks hard coded into it. As the company grows and gains more trucks and drivers, some modifications will need to be made. These changes should not be difficult, however. Rather than hard coding the trucks, an array of trucks may be created and passed to the path optimizing function. Additionally, as the company grows, some consideration may need to be paid to the loading of packages so that packages bound for different cities do not end up on the same truck. Overall, however, the program will easily be able to grow and scale as the company does and is currently adequately effective in planning deliveries.

1. Discuss the efficiency and maintainability of the software.

Most aspects of the program are separated into distinct modules, so changes that need to be made should have limited effects across the whole program. All data including package info, address info, and distance info currently must be entered in csv format. This should be easy to work with and extend as the company grows. However, some care must be places in creating systems to facilitate gathering data and formatting it appropriately.

1. Discuss the self-adjusting data structures chosen and their strengths and weaknesses based on the scenario.

Package data structure may be easily changed by adding or removing fields. Support may also be easily added for packages with incomplete data. Trucks are currently set up as a simple list which holds packages up to a certain capacity. This capacity may be updated quickly if the capacity of trucks changes in the future. The hashtable holds all package information, stored based on a hash of the package ID. As long as the ID’s are unique, there should not be a problem with this structure. However, if the ID’s are ever repeated, they may be some collisions in the hashtable, which may result in inconsistencies.

D.  Identify a data structure that can be used with your chosen algorithm to store the package data.

1.  Explain how your data structure includes the relationship between the data points you are storing.

To store the package data within the program, I chose to utilize a chaining hash table. This prevents collisions in storing after hashing the package by storing the hashed values into a list. The packages are then loaded into a list which represents a truck by their ID’s. Using the ID’s, the program is able to look up any information needed including addresses and distances values to the next stops. (Zybooks, 7.1)

G.  Provide an interface for the insert and look-up functions to view the status of any package at any time. This function should return all information about each package, including delivery status.

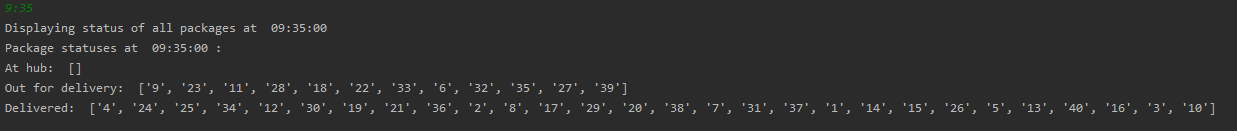
1.  Provide screenshots to show package status of all packages at a time between 8:35 a.m. and 9:25 a.m.

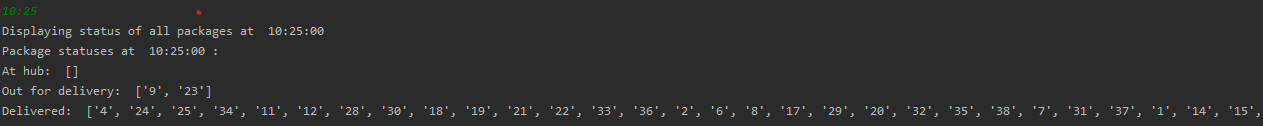
A screen shot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

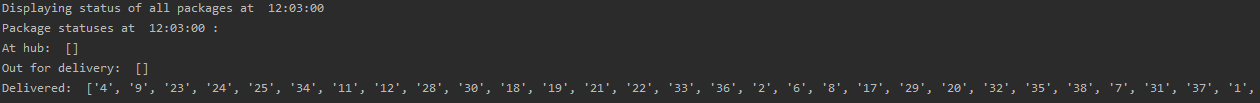
2.  Provide screenshots to show package status of all packages at a time between 9:35 a.m. and 10:25 a.m.







3.  Provide screenshots to show package status of all packages at a time between 12:03 p.m. and 1:12 p.m.





A close up of a computer

Description automatically generated



H.  Run your code and provide screenshots to capture the complete execution of your code.

A screenshot of a cell phone

Description automatically generated

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screen shot of a computer

Description automatically generatedA close up of a keyboard

Description automatically generated

***Section 2: Annotations***

I.  Justify your choice of algorithm by doing the following:

1.  Describe at least two strengths of the algorithm you chose.

One key advantage of the greedy algorithm is that it is simple to understand and therefore easy to update as needed in the future. Another strength of the algorithm is that it is effective in finding an efficient path with the small number of packages allowed per truck. The program is currently configured to first load the trucks, and then run the greedy algorithm on the trucks. This allows for specific shipping requirements to be considered separately from the sorting algorithm, creating more flexibility in the program.

2.  Verify that the algorithm you chose meets all the criteria and requirements given in the scenario.

As shown in the provided screenshots, all criteria and requirements are met, including delivery deadlines and mileage limitations.

3.  Identify **two** other algorithms that could be used and would have met the criteria and requirements given in the scenario.

Two potential other algorithms that could have been used are Dijkstra’s algorithm and the A\* search algorithm.

1. Describe how each algorithm identified in part I3 is different from the algorithm you chose to use in the solution.

While all three algorithms are effective in finding an optimal shipping route, the methods they use are slightly different. Dijkstra’s algorithm is a type of greedy algorithm, but rather than checking the distance to the next node once, it continually updates the potential path distance. A\* is also similar to the preview two search algorithms in that it searches through the connecting nodes to find the optimal path. However, A\* uses different heuristics to make its choices. Along with considering the distance to the next node, A\* also makes an estimate on the remaining distance to the end node.

J.  Describe what you would do differently if you did this project again.

If I were to do this project again, I would focus more heavily on the future growth potential of the company while creating the program. The program I created is effective currently, but if the company grows significantly, some issues may arise, particularly in the ease of use of inputting data. It would also be interesting to implement some networking systems in order to allow the program to be run and accessed from multiple machines. This would also allow for better optimization by utilizing various APIs such as Google Maps.

K.  Justify your choice of data structure by doing the following:

1.  Verify that the data structure you chose meets all the criteria and requirements given in the scenario.

a.  Describe the efficiency of the data structure chosen.

The program utilizes a chaining hash table to store data and simple lists as representations of the delivery trucks. These data structures are very flexible and easy to work with and the efficiency of looking up data from the hash table is O(1).

b.  Explain the expected overhead when linking to the next data item.

Due to the data being stored in a hash table, there is very little overhead when adding a new package into the truck list.

c.  Describe the implications of when more package data is added to the system or other changes in scale occur.

In some hash table variations, adding more data may eventually result in collisions and therefore errors. However, by utilizing a chaining hash table, any collisions that occur after hashing the package data is added to a bucket list. This allows for a large amount of data to be stored and easily accessed.

2.  Identify **two** other data structures that can meet the same criteria and requirements given in the scenario.

Two other potential data structures that could have been used are a binary search tree or a graph.

a.  Describe how each data structure identified in part K2 is different from the data structure you chose to use in the solution.

A binary search tree sorts data into a tree format as soon as data is added to it. This would allow for easy optimization using a search algorithm, as the data could be pre-sorted by location. A graph could also be effectively utilized in this program. Like binary search trees, graphs can be sorted by related distances to allow for easy optimization later in the program.

L.   Acknowledge sources, using in-text citations and references, for content that is quoted, paraphrased, or summarized.

Zybooks, *C950: Data Structures and Algorithms II.* https://learn.zybooks.com/zybook/WGUC950AY20182019/chapter/7/section/1