C950 Task-2 WGUPS Write-Up

(Task-2: The implementation phase of the WGUPS Routing Program).

(Zip your source code and upload it with this file)

Cole Gibbs

ID 010988814

WGU Email: cgib276@wgu.edu

04/16/2024

C950 Data Structures and Algorithms II

# A. Hash Table

A computer screen shot of a program code

Description automatically generated

# B. Look-Up Functions

A black screen with green text

Description automatically generated

# C. Original Code

A computer screen shot of a program

Description automatically generated

# C1. Identification Information

A black screen with colorful text

Description automatically generated

# C2. Process and Flow Comments

A computer screen shot of a program

Description automatically generated

# D. Interface

A computer screen shot of a program

Description automatically generated

# D1. First Status Check

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A black background with orange and black text

Description automatically generated with medium confidence

# D2. Second Status Check

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A black background with a black square

Description automatically generated with medium confidence

# D3. Third Status Check

A screenshot of a computer

Description automatically generated

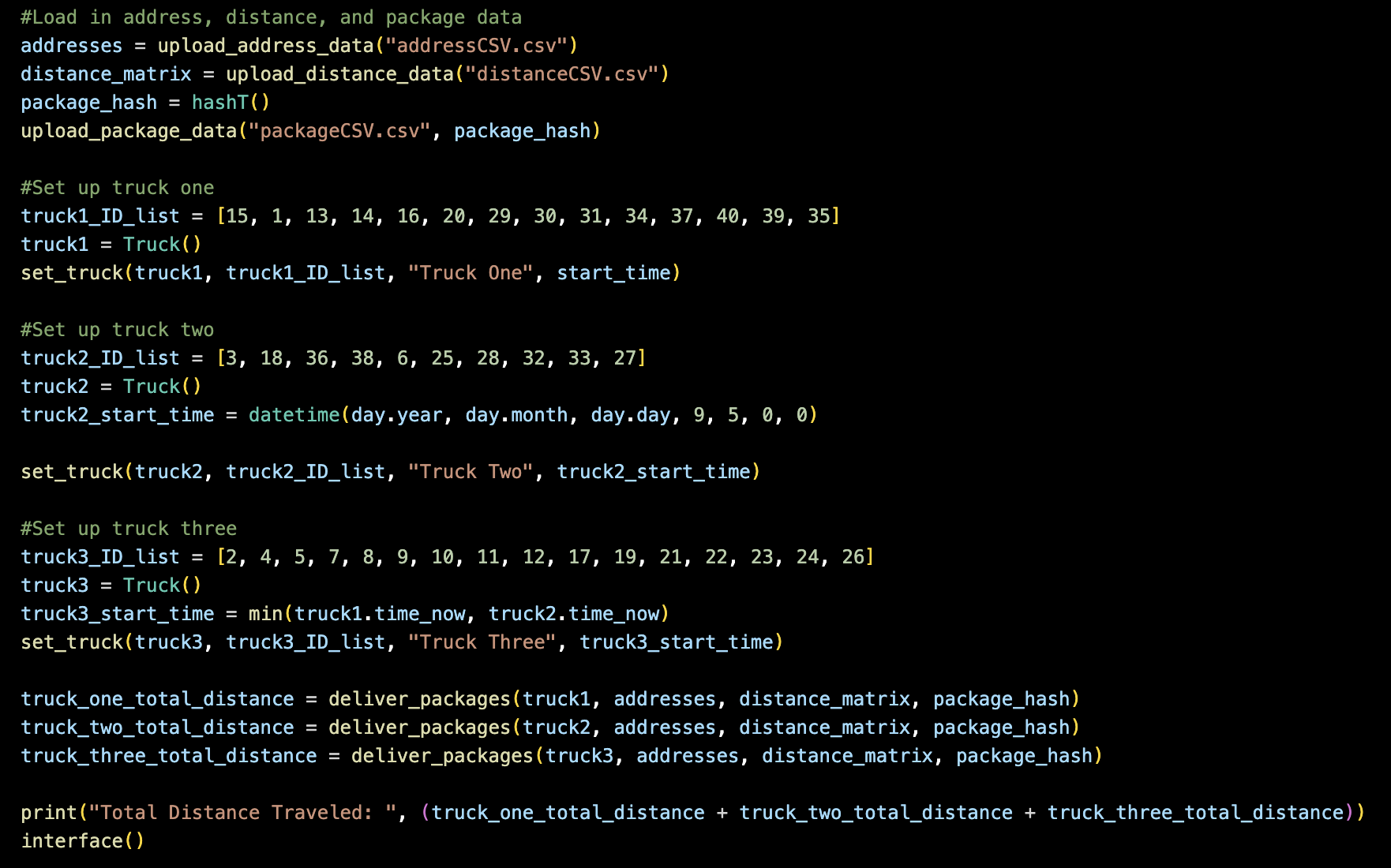
A screenshot of a computer

Description automatically generated

A black background with a black square

Description automatically generated with medium confidence

# E. Screenshot of Code Execution



# F1. Strengths of the Chosen Algorithm

1. The greedy algorithm used is efficient in terms of memory usage. There is no guiding data structure created before the trucks leave the HUB, so all the trucks are finding the next nearest location in real time.
2. The greedy algorithm makes it efficient in terms of travel time by allowing the truck to go to the next closest location. If a minimal spanning tree algorithm was used for example, the path from the HUB to any given location would be most efficient, but if a package has a delivery delay, then the truck could travel more distance than necessary at that time.

# F2. Verification of Algorithm

1. Each truck carries no more than 16 packages.
2. Each truck travels 18 miles per hour.
3. A maximum of two trucks are away from the HUB at any given time (two drivers.
4. No truck leaves the HUB earlier than 8:00am.
5. Package #9 is delivered correctly.
6. All 40 packages are delivered by EOD.
7. The total distance traveled by all trucks is less than 140 miles.
8. Trucks are loaded at the HUB.
9. Packages #13, #14, #15. #16, #19, and #20 by the same truck.
10. Packages #3, #18, #36, and #38 are delivered by truck 2.
11. Packages #1, 6, 13, 14, 16, 20, 25, 29, 30, 31, 34, 37 and 40 are delivered by 10:30am.
12. Package 15 is delivered by 9:00am.

# F3. Other Possible Algorithms

1. Minimum Spanning Tree
2. Dykstra’s algorithm

# F3a. Algorithm Differences

1. The minimum spanning tree algorithm would give each truck a preset order of delivery addresses instead of the real time closest address behavior of the implemented greedy algorithm.
2. Dykstra’s Algorithm would allow for the most efficient delivery of packages based on the last package delivered and the packages that would be delivered along the way. This would be the most difficult to implement out of the three algorithms presented because the delaying of packages and the requirement for groups of packages to be delivered on the same truck would create complicated delivery routes.

# G. Different Approach

If I were to do this project again, I would make the code adaptable to any use case. In this implementation I though only of the requirements of presented to me in this project and built it with them in mind. Should those requirements change, my code would not perform well. Much of the project is “hard coded” when it could be “dynamic”.

# H. Verification of Data Structure

The Hash Table data structure used meets all package organization, data, and delivery requirements.

# H1. Other Data Structures

1. Array
2. Dictionary

# H1a. Data Structure Differences

1. Instead of indexing the packages based on the hash function, the package ID itself could have been used. In the case a package with ID of 6 would be in index location 6 in the array.
2. A dictionary could have been used that would have the key: value pair of ID: package.

# I. Sources

Lysecky, R., & Vahid, F. (2022, August). *C950: Data Structures and Algorithms II*. zyBooks. Retrieved April, 2024, from <https://learn.zybooks.com/zybook/WGUC950AY20182019/>

Video Tutorials and instructions from C950 instructor, Robert Ferdinand.