

\*\*\*\*\* Algorithm Overview

\*\*\*\*\* Algorithm Overview

Tenny Akihary

ID #\*\*\*\*\*

Email: takihary@hotmail.com

Date: May 11<sup>th</sup>, 2022

\*\*\*\*\* Routing Program

## Introduction

This program focuses on using data structures and a self-adjusting algorithm. In this scenario, \*\*\*\*\* needs help determining an efficient route for delivering packages. \*\*\*\*\* has many assumptions and specific criteria that must be held consistently during daily local deliveries. This program will implement said structures and algorithms to meet all constraints in delivering each package while accruing less than 140 miles total.

---

### A. Algorithm Identification

The algorithm in the program is like the nearest neighbor algorithm. Each truck has its list of packages that need to be delivered. The truck will travel from the hub and visit the package address closest to its current location, completing the delivery. This will be repeated until all distinct addresses have been visited once and all packages on the truck have been delivered. The truck will then return to the starting position to deliver more potential packages.

---

### B1. Logic Comments

```
1   Initialize truck:= [list of package IDs] # Manually loading the truck
2   Time:= user inputs a desired time
3   Truck miles left:= Miles to travel at 18mph from 8am to desired time
4   Truck mileage:= 0
5   Truck time = 8am
6
7   Loop when truck's miles left != 0 and package list length != 0
8       Closest_address:= find_closest_address(truck)
9       Distance:= distance_betweeen(closest_address, trucks location)
10      If distance < trucks miles left
11          Trucks miles left -= distance
```

#### \*\*\*\*\* Algorithm Overview

```
12     Truck location = closest_address
13     Truck mileage += distance
14     Truck time += distance / 18mph
15     Remove package(s) from list, set package(s) status + delivery time
16     Else:
17         Truck mileage += trucks miles left
18         Truck miles left = 0
19         Break out of the loop
20 If package list length == 0
21     Distance = distance_between(trucks location, the hub)
22     If distance < trucks miles left
23         Trucks miles left -= distance
24         Truck location = the hub
25         Truck mileage += distance
26         Truck time += distance / 18mph
27     Else:
28         Truck mileage += trucks miles left
29         Truck miles left = 0
```

The “find\_closest\_address” method compares all package addresses with the truck's current location and returns the address with the smallest distance value. The “distance\_between” method compares two addresses and returns the distance value.

The truck object is initialized and has internal values that hold a list of package IDs, total mileage, miles left, time, and the truck's location (Lines 1-5.)

If the truck has miles left to spend and has packages in the list, the loop will iterate until one of the two is 0 or if the distance is more than the remaining miles (Lines 7-19.)

#### \*\*\*\*\* Algorithm Overview

The closest address is found and saved, then the distance between the truck and that location is saved (Lines 8-9.)

If the distance is less than the miles left, then the delivery will execute. Mileage adds that distance, miles left subtract that distance, and the truck location changes to that closest location. Time is added and the package(s) are removed from the truck list and updated to the hash table with the truck's current time (Lines 10-15.)

If the miles left are less than the distance, then the truck will be stopped partially. The partial miles left will be taken and added to the mileage and the loop breaks (Lines 16-19.)

After the loop, the if statement checks for the completion of delivering all packages and finds the distance back to the hub. If the truck has miles left to cover the distance, the truck will return to the hub. Mileage will add the distance to the hub, miles left will subtract that distance, and the time it will take will be added (Lines 20-26.)

If the truck doesn't have miles left to cover the distance, then the partial miles are added to the mileage, and the miles left are set to 0 (Lines 27-29.)

## B2. Development Environment

The software of the development environment consisted of PyCharm by JetBrains.

IDE: PyCharm 2022.1

Interpreter: Python 3.9

Hardware includes as follows:

Processor – AMD Ryzen 9 3900x 12-Core Processor 3.80 GHz

RAM – 128GB

System type – 64-bit operating system

Operating System – Windows 10

### B3. Space-Time and Big-O

Entire program:                      Time Complexity –  $O(n^2)$       Space Complexity –  $O(n)$

#### Segments

Loading data:                      Time Complexity –  $O(n)$       Space Complexity –  $O(n)$

User Interface/Display data:      Time Complexity –  $O(n^2)$       Space Complexity –  $O(n)$

Delivery algorithm:                Time Complexity –  $O(n^2)$       Space Complexity –  $O(n)$

### B4. Scalability and Adaptability

The project is capable of handling large amounts of packages. The additional packages will need to be manually loaded or the program will need to be updated for a self-loading algorithm.

However, given the algorithm and the data structure the program will be able to handle any number of packages with the time and space complexity scaling above. As long as the packages remain consistent with the format in the CSV files, the program should be able to adapt to any size package or distance list from any location.

### B5. Software Efficiency and Maintainability

The software is organized to recognize the flow of the program. The important objects, such as the truck, hash table, and package, each have their file. For future developers, comments are made throughout the entire program to help explain each class, method, variable, and almost every line of code. This program has been broken down into smaller segments, therefore, it is easy to find and adjust certain processes if needed.

#### \*\*\*\*\* Algorithm Overview

The efficiency of the program is not at its maximum. Although the program fits the constraints given by the requirements, the program does not find the most optimal route. The program, using the nearest neighbor algorithm and the order in which the trucks are manually loaded, finds a sub-optimal route for the trucks to make the deliveries. Implementing a self-loading algorithm that helps fit future constraints would help make this program more efficient. It would also be more efficient to implement a different algorithm or function resulting in the shortest total mileage. Most of these algorithms will sacrifice a worse time and space complexity for finding the shortest total mileage. For this requirement, the total mileage being under a certain value is efficient enough. However, if finding the most efficient route is of more value, the sacrifice may be beneficial. When implementing a new algorithm, it would be best to find a balance between business needs and the time and space complexity.

#### B6. Self-Adjusting Data Structures

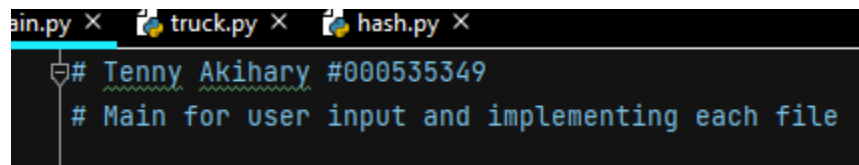
The strengths of the hash table data structure used include the ability to search using a key (in this case the package ID) which is great for database indexing. The weaknesses of a hashing table are the practically unavoidable collisions. As the number of items increases, inevitably more collisions will occur.

---

#### C. Original Code

See code in main.py, package.py, truck.py, and hash.py.

#### C1. Identification Information



```
main.py × truck.py × hash.py ×  
# Tenny Akihary #000535349  
# Main for user input and implementing each file
```

## C2. Process and Flow Comments

See comments in main.py, package.py, truck.py, and hash.py.

---

## D. Data Structure

The hash table goes well with the nearest neighbor algorithm in that data is easily indexed and pulled for use. It holds any number of key-value pairs given the index and item which in this case is a package object.

### D1. Explanation of Data Structure

Data structure has specific cells from cell 0 to cell x that hold the key and an item. The key is hashed to find a certain cell that it and the item will then be stored in (Tepe). When collisions occur, the key-value pair is simply appended to that cell to store a list of pairs, meaning a cell can hold multiple value pairs. This method of collision handling was adopted by James (2016), for its simplicity. This data structure is great for holding packages that have unique IDs and the package object that holds its information. The nearest neighbor algorithm can then pull the object, given the ID, to view/modify data.

---

## E. Hash Table

```
def add(self, key, item):
    # Variables to hold which cell and the key item
    cell = hash(key) % self.size
    key_item = [key, item]
    # If the cell is empty then add the key item.
    if self.table[cell] is None:
        self.table[cell] = list([key_item])
        return True
    # If the cell is not empty.
    else:
        # Check each item in the cell for the key. Update item if key is present.
        for item in self.table[cell]:
            if item[0] == key:
                item[1] = item
                return True
        # If no key is found in the cell, append the key item to the cell.
        self.table[cell].append(key_item)
        return True
```

See hash.py lines (22-39)

---

## F. Look-Up Function

```
def get(self, key):
    # Variable to hold which cell.
    cell = hash(key) % self.size
    # If the cell is not empty.
    if self.table[cell] is not None:
        # Check for the key in the cell. Return the item if found.
        for item in self.table[cell]:
            if item[0] == key:
                return item[1]
    # If there is no item given the key return None.
    return None
```

See hash.py lines (44-55)



## \*\*\*\*\* Algorithm Overview

### G. Interface

The interface receives the time input from the user and displays all package data or specific package data at that given time.

```
Enter a four digit time (in military time) throughout this day's delivery process (e.g., HHMM 1300):
Enter time: 800
Menu
1: View All Package Data
2: View Specific Package
Enter selection: 1
ID: 33 Address: 2530 S 500 E City: Salt Lake City State: UT Zip: 84106 Delivery Deadline: EOD
ID: 39 Address: 2010 W 500 S City: Salt Lake City State: UT Zip: 84104 Delivery Deadline: EOD
ID: 15 Address: 4580 S 2300 E City: Holladay State: UT Zip: 84117 Delivery Deadline: 8:00 AM
```

### G1. First Status Check

```
Enter a four digit time (in military time) throughout this day's delivery process (e.g., HHMM 1300):
Enter time: 900
Menu
1: View All Package Data
2: View Specific Package
Enter selection: 1
ID: 38 Address: 410 S State St City: Salt Lake City State: UT Zip: 84111 Delivery Deadline: EOD Mass KILO: 9 Special Notes: Can only be on truck 2 Delivery Status: En Route
ID: 19 Address: 177 W Price Ave City: Salt Lake City State: UT Zip: 84115 Delivery Deadline: EOD Mass KILO: 37 Special Notes: Delivery Status: Delivered at 8:29
ID: 22 Address: 4354 South 900 East City: Murray State: UT Zip: 84121 Delivery Deadline: EOD Mass KILO: 2 Special Notes: Delivery Status: AT HUB
ID: 33 Address: 2530 S 500 E City: Salt Lake City State: UT Zip: 84106 Delivery Deadline: EOD Mass KILO: 1 Special Notes: Delivery Status: AT HUB
ID: 2 Address: 2530 S 500 E City: Salt Lake City State: UT Zip: 84106 Delivery Deadline: EOD Mass KILO: 44 Special Notes: Delivery Status: AT HUB
ID: 13 Address: 2010 W 500 S City: Salt Lake City State: UT Zip: 84104 Delivery Deadline: 10:30 AM Mass KILO: 2 Special Notes: Delivery Status: En Route
ID: 3 Address: 233 Canyon Rd City: Salt Lake City State: UT Zip: 84103 Delivery Deadline: EOD Mass KILO: 2 Special Notes: Can only be on truck 2 Delivery Status: En Route
ID: 20 Address: 3595 Main St City: Salt Lake City State: UT Zip: 84115 Delivery Deadline: 10:30 AM Mass KILO: 37 Special Notes: Must be delivered with 13, 15 Delivery Status: Delivered at 8:28
ID: 24 Address: 5025 State St City: Murray State: UT Zip: 84107 Delivery Deadline: EOD Mass KILO: 7 Special Notes: Delivery Status: AT HUB
ID: 40 Address: 380 W 2880 S City: Salt Lake City State: UT Zip: 84115 Delivery Deadline: 10:30 AM Mass KILO: 45 Special Notes: Delivery Status: Delivered at 8:34
ID: 39 Address: 2010 W 500 S City: Salt Lake City State: UT Zip: 84104 Delivery Deadline: EOD Mass KILO: 9 Special Notes: Delivery Status: AT HUB
ID: 35 Address: 1060 Dalton Ave S City: Salt Lake City State: UT Zip: 84104 Delivery Deadline: EOD Mass KILO: 88 Special Notes: Delivery Status: AT HUB
ID: 37 Address: 410 S State St City: Salt Lake City State: UT Zip: 84111 Delivery Deadline: 10:30 AM Mass KILO: 2 Special Notes: Delivery Status: En Route
ID: 1 Address: 195 W Oakland Ave City: Salt Lake City State: UT Zip: 84115 Delivery Deadline: 10:30 AM Mass KILO: 21 Special Notes: Delivery Status: Delivered at 8:37
ID: 10 Address: 400 E 900 South City: Salt Lake City State: UT Zip: 84105 Delivery Deadline: EOD Mass KILO: 1 Special Notes: Delivery Status: AT HUB
ID: 14 Address: 4300 S 1300 E City: Millicreek State: UT Zip: 84117 Delivery Deadline: 10:30 AM Mass KILO: 88 Special Notes: Must be delivered with 15, 19 Delivery Status: Delivered at 8:00
ID: 23 Address: 5100 South 2700 West City: Salt Lake City State: UT Zip: 84118 Delivery Deadline: EOD Mass KILO: 5 Special Notes: Delivery Status: AT HUB
ID: 31 Address: 3365 S 900 W City: Salt Lake City State: UT Zip: 84119 Delivery Deadline: 10:30 AM Mass KILO: 1 Special Notes: Delivery Status: AT HUB
ID: 5 Address: 410 S State St City: Salt Lake City State: UT Zip: 84111 Delivery Deadline: EOD Mass KILO: 5 Special Notes: Delivery Status: AT HUB
ID: 6 Address: 3000 Linder St City: West Valley City State: UT Zip: 84119 Delivery Deadline: 10:30 AM Mass KILO: 88 Special Notes: Delayed on flight--will not arrive to depot until 9:05 am Delivery Status: AT HUB
ID: 16 Address: 4580 S 2300 E City: Holladay State: UT Zip: 84117 Delivery Deadline: 9:00 AM Mass KILO: 4 Special Notes: Delivery Status: Delivered at 8:12
ID: 12 Address: 3575 W Valley Central Station bus Loop City: West Valley City State: UT Zip: 84119 Delivery Deadline: EOD Mass KILO: 1 Special Notes: Delivery Status: AT HUB
ID: 30 Address: 300 State St City: Salt Lake City State: UT Zip: 84103 Delivery Deadline: 10:30 AM Mass KILO: 1 Special Notes: Delivery Status: En Route
ID: 36 Address: 2300 Parkway Blvd City: West Valley City State: UT Zip: 84119 Delivery Deadline: EOD Mass KILO: 88 Special Notes: Can only be on truck 2 Delivery Status: En Route
ID: 25 Address: 5383 South 900 East #104 City: Salt Lake City State: UT Zip: 84117 Delivery Deadline: 10:30 AM Mass KILO: 7 Special Notes: Delayed on flight--will not arrive to depot until 9:05 am Delivery Status: AT HUB
ID: 27 Address: 1060 Dalton Ave S City: Salt Lake City State: UT Zip: 84104 Delivery Deadline: EOD Mass KILO: 5 Special Notes: Delivery Status: AT HUB
ID: 29 Address: 1330 2100 S City: Salt Lake City State: UT Zip: 84106 Delivery Deadline: 10:30 AM Mass KILO: 2 Special Notes: Delivery Status: Delivered at 8:40
ID: 26 Address: 5383 South 900 East #104 City: Salt Lake City State: UT Zip: 84117 Delivery Deadline: EOD Mass KILO: 26 Special Notes: Delivery Status: AT HUB
ID: 32 Address: 3365 S 900 W City: Salt Lake City State: UT Zip: 84119 Delivery Deadline: EOD Mass KILO: 1 Special Notes: Delayed on flight--will not arrive to depot until 9:05 am Delivery Status: AT HUB
ID: 8 Address: 300 State St City: Salt Lake City State: UT Zip: 84103 Delivery Deadline: EOD Mass KILO: 9 Special Notes: Delivery Status: AT HUB
ID: 28 Address: 2635 Main St City: Salt Lake City State: UT Zip: 84115 Delivery Deadline: EOD Mass KILO: 7 Special Notes: Delayed on flight--will not arrive to depot until 9:05 am Delivery Status: AT HUB
ID: 7 Address: 1330 2100 S City: Salt Lake City State: UT Zip: 84106 Delivery Deadline: EOD Mass KILO: 8 Special Notes: Delivery Status: AT HUB
ID: 11 Address: 2600 Taylorsville Blvd City: Salt Lake City State: UT Zip: 84118 Delivery Deadline: EOD Mass KILO: 1 Special Notes: Delivery Status: AT HUB
ID: 18 Address: 1480 4800 S City: Salt Lake City State: UT Zip: 84123 Delivery Deadline: EOD Mass KILO: 6 Special Notes: Can only be on truck 2 Delivery Status: En Route
ID: 14 Address: 4580 S 2300 E City: Holladay State: UT Zip: 84117 Delivery Deadline: 10:30 AM Mass KILO: 88 Special Notes: Must be delivered with 13, 19 Delivery Status: Delivered at 8:12
ID: 34 Address: 4580 S 2300 E City: Holladay State: UT Zip: 84117 Delivery Deadline: 10:30 AM Mass KILO: 2 Special Notes: Delivery Status: Delivered at 8:12
ID: 9 Address: 300 State St City: Salt Lake City State: UT Zip: 84103 Delivery Deadline: EOD Mass KILO: 2 Special Notes: At 10:20 address updated. Delivery Status: AT HUB
ID: 21 Address: 3595 Main St City: Salt Lake City State: UT Zip: 84115 Delivery Deadline: EOD Mass KILO: 3 Special Notes: Delivery Status: AT HUB
ID: 4 Address: 380 W 2880 S City: Salt Lake City State: UT Zip: 84115 Delivery Deadline: EOD Mass KILO: 4 Special Notes: Delivery Status: AT HUB
ID: 17 Address: 5148 S 1100 W City: Salt Lake City State: UT Zip: 84119 Delivery Deadline: EOD Mass KILO: 2 Special Notes: Delivery Status: AT HUB
Total mileage: 18.0
```

## G2. Second Status Check

### G3. Third Status Check

## H. Screenshots of Code Execution

Page 10 of 15

## \*\*\*\*\* Algorithm Overview

```
Enter a four digit time (in military time) throughout this day's delivery process (e.g., HMM 1300):
Enter time: 2359
Menu
1: View All Package Data
2: View Specific Package
Enter selection: 1
ID: 16 Address: 4580 S 2300 E City: Holladay State: UT Zip: 84117 Delivery Deadline: 10:30 AM Mass KILO: 88 Special Notes: Must be delivered with 13, 19 Delivery Status: Delivered at 8:12
ID: 7 Address: 1330 2100 S City: Salt Lake City State: UT Zip: 84106 Delivery Deadline: EOD Mass KILO: 8 Special Notes: Delivery Status: Delivered at 10:06
ID: 28 Address: 2835 Main St City: Salt Lake City State: UT Zip: 84115 Delivery Deadline: EOD Mass KILO: 7 Special Notes: Delayed on flight---will not arrive to depot until 9:05 am Delivery Status: Delivered at 9:30
ID: 33 Address: 2530 S 900 E City: Salt Lake City State: UT Zip: 84106 Delivery Deadline: EOD Mass KILO: 1 Special Notes: Delivery Status: Delivered at 10:01
ID: 27 Address: 1600 Dalton Ave S City: Salt Lake City State: UT Zip: 84104 Delivery Deadline: EOD Mass KILO: 5 Special Notes: Delivery Status: Delivered at 10:40
ID: 12 Address: 3570 W Valley Central Station Bus Loop City: West Valley City State: UT Zip: 84119 Delivery Deadline: EOD Mass KILO: 1 Special Notes: Delivery Status: Delivered at 12:08
ID: 26 Address: 5353 South 900 East #104 City: Salt Lake City State: UT Zip: 84117 Delivery Deadline: EOD Mass KILO: 25 Special Notes: Delivery Status: Delivered at 9:13
ID: 30 Address: 300 State St City: Salt Lake City State: UT Zip: 84103 Delivery Deadline: 10:30 AM Mass KILO: 1 Special Notes: Delivery Status: Delivered at 9:05
ID: 39 Address: 2010 W 500 S City: Salt Lake City State: UT Zip: 84104 Delivery Deadline: EOD Mass KILO: 9 Special Notes: Delivery Status: Delivered at 10:51
ID: 35 Address: 1060 Dalton Ave S City: Salt Lake City State: UT Zip: 84104 Delivery Deadline: EOD Mass KILO: 88 Special Notes: Delivery Status: Delivered at 10:40
ID: 13 Address: 2010 W 500 S City: Salt Lake City State: UT Zip: 84104 Delivery Deadline: 10:30 AM Mass KILO: 2 Special Notes: Delivery Status: Delivered at 9:19
ID: 18 Address: 1488 4800 S City: Salt Lake City State: UT Zip: 84123 Delivery Deadline: EOD Mass KILO: 4 Special Notes: Can only be on truck 2 Delivery Status: Delivered at 9:46
ID: 32 Address: 3340 S 900 W City: Salt Lake City State: UT Zip: 84119 Delivery Deadline: EOD Mass KILO: 1 Special Notes: Delayed on flight---will not arrive to depot until 9:05 am Delivery Status: Delivered at 9:39
ID: 22 Address: 6351 South 900 East City: Murray State: UT Zip: 84121 Delivery Deadline: EOD Mass KILO: 2 Special Notes: Delivery Status: Delivered at 11:20
ID: 15 Address: 4580 S 2300 E City: Holladay State: UT Zip: 84117 Delivery Deadline: 9:00 AM Mass KILO: 4 Special Notes: Delivery Status: Delivered at 8:12
ID: 21 Address: 3595 Main St City: Salt Lake City State: UT Zip: 84115 Delivery Deadline: EOD Mass KILO: 3 Special Notes: Delivery Status: Delivered at 9:50
ID: 17 Address: 3148 S 1100 W City: Salt Lake City State: UT Zip: 84119 Delivery Deadline: EOD Mass KILO: 2 Special Notes: Delivery Status: Delivered at 10:55
ID: 31 Address: 3100 S 900 W City: Salt Lake City State: UT Zip: 84119 Delivery Deadline: 10:30 AM Mass KILO: 1 Special Notes: Delivery Status: Delivered at 9:39
ID: 20 Address: 3090 Main St City: Salt Lake City State: UT Zip: 84115 Delivery Deadline: 10:30 AM Mass KILO: 37 Special Notes: Must be delivered with 13, 15 Delivery Status: Delivered at 8:28
ID: 4 Address: 380 W 2880 S City: Salt Lake City State: UT Zip: 84115 Delivery Deadline: EOD Mass KILO: 4 Special Notes: Delivery Status: Delivered at 9:55
ID: 36 Address: 2380 Parkway Blvd City: West Valley City State: UT Zip: 84119 Delivery Deadline: EOD Mass KILO: 88 Special Notes: Can only be on truck 2 Delivery Status: Delivered at 9:32
ID: 34 Address: 4580 S 2300 E City: Holladay State: UT Zip: 84117 Delivery Deadline: 10:30 AM Mass KILO: 2 Special Notes: Delivery Status: Delivered at 8:12
ID: 3 Address: 233 Canyon Rd City: Salt Lake City State: UT Zip: 84105 Delivery Deadline: EOD Mass KILO: 2 Special Notes: Can only be on truck 2 Delivery Status: Delivered at 9:03
ID: 9 Address: 410 S State St City: Salt Lake City State: UT Zip: 84111 Delivery Deadline: EOD Mass KILO: 2 Special Notes: At 10:20 address updates. Delivery Status: Delivered at 10:41
ID: 40 Address: 310 W 2880 S City: Salt Lake City State: UT Zip: 84115 Delivery Deadline: 10:30 AM Mass KILO: 45 Special Notes: Delivery Status: Delivered at 8:34
ID: 6 Address: 3060 Lester St City: West Valley City State: UT Zip: 84119 Delivery Deadline: 10:30 AM Mass KILO: 88 Special Notes: Delayed on flight---will not arrive to depot until 9:05 am Delivery Status: Delivered at 9:44
ID: 8 Address: 300 State St City: Salt Lake City State: UT Zip: 84103 Delivery Deadline: EOD Mass KILO: 9 Special Notes: Delivery Status: Delivered at 10:24
ID: 5 Address: 410 S State St City: Salt Lake City State: UT Zip: 84111 Delivery Deadline: EOD Mass KILO: 5 Special Notes: Delivery Status: Delivered at 10:21
ID: 11 Address: 4600 Taylorsville Blvd City: Salt Lake City State: UT Zip: 84118 Delivery Deadline: EOD Mass KILO: 1 Special Notes: Delivery Status: Delivered at 11:42
ID: 25 Address: 4583 South 900 East #104 City: Salt Lake City State: UT Zip: 84117 Delivery Deadline: 10:30 AM Mass KILO: 7 Special Notes: Delayed on flight---will not arrive to depot until 9:05 am Delivery Status: Delivered at 9:13
ID: 29 Address: 1330 2100 S City: Salt Lake City State: UT Zip: 84106 Delivery Deadline: 10:30 AM Mass KILO: 2 Special Notes: Delivery Status: Delivered at 8:40
ID: 38 Address: 410 S State St City: Salt Lake City State: UT Zip: 84111 Delivery Deadline: EOD Mass KILO: 9 Special Notes: Can only be on truck 2 Delivery Status: Delivered at 9:00
ID: 2 Address: 2530 S 900 E City: Salt Lake City State: UT Zip: 84106 Delivery Deadline: EOD Mass KILO: 44 Special Notes: Delivery Status: Delivered at 10:01
ID: 23 Address: 5100 South 2700 West City: Salt Lake City State: UT Zip: 84118 Delivery Deadline: EOD Mass KILO: 5 Special Notes: Delivery Status: Delivered at 11:43
ID: 14 Address: 4300 S 1300 E City: Millcreek State: UT Zip: 84117 Delivery Deadline: 10:30 AM Mass KILO: 88 Special Notes: Must be delivered with 15, 19 Delivery Status: Delivered at 8:06
ID: 37 Address: 410 S State St City: Salt Lake City State: UT Zip: 84111 Delivery Deadline: 10:30 AM Mass KILO: 2 Special Notes: Delivery Status: Delivered at 9:00
ID: 24 Address: 5025 State St City: Murray State: UT Zip: 84107 Delivery Deadline: EOD Mass KILO: 7 Special Notes: Delivery Status: Delivered at 11:10
ID: 19 Address: 177 W Price Ave City: Salt Lake City State: UT Zip: 84115 Delivery Deadline: EOD Mass KILO: 37 Special Notes: Delivery Status: Delivered at 8:29
ID: 1 Address: 105 W Deland Ave City: Salt Lake City State: UT Zip: 84115 Delivery Deadline: 10:30 AM Mass KILO: 21 Special Notes: Delivery Status: Delivered at 8:37
ID: 10 Address: 600 E 900 South City: Salt Lake City State: UT Zip: 84105 Delivery Deadline: EOD Mass KILO: 1 Special Notes: Delivery Status: Delivered at 10:15
Total mileage: 133.9
Process finished with exit code 0
```

Zoomed in on total mileage:

```
ID: 10 Address: 600 E 900 South City: Salt Lake City State: UT Zip: 84105 Delivery Deadline: EOD Mass KILO: 1 Special Notes: Delivery Status: Delivered at 10:15
Total mileage: 133.9
Process finished with exit code 0
```

## 11. Strengths of Chosen Algorithm

Many algorithms seek to find the most optimal path given the distances. Many of those algorithms have extreme run times to find the most optimal path. The algorithm used in the program has a few strengths: finding the path is simple (finding the closest address), has a lower run-time, and still manages to keep the total mileage under 140.

## 12. Verification of Algorithm

The submission verifies all the required elements were met. Each package with a delivery deadline had its time constraint met. Truck 3 did not leave the hub until the packages from the



delayed flight arrived. Packages that were required to be delivered together had their respective truck. The packages that were required to be on truck 2 were loaded as such. When selecting “View All Package Data” from the user interface, the total mileage from all trucks combined is displayed at the bottom. In this case, the total mileage to deliver all packages totaled 133.9 miles. Which was under the maximum mileage allowed of 140 miles.

### I3. Other possible Algorithms

The greedy algorithm and the nearest insertion algorithm would have been possible for this problem. Neither will most likely give the most optimal route for the deliveries. However, each algorithm will fit the scenario and total less than 140 miles.

#### I3A. Algorithm Differences

In the greedy algorithm, the distance from each address is sorted from shortest to longest. Afterward, it searches for the shortest distances that will not cause the deliveries to miss an address or visit an address more than once.

Unlike the greedy algorithm, the nearest neighbor algorithm did not need to sort the distances. Both algorithms searched for the shortest edge or distance between two addresses. However, the greedy algorithm builds the route in different segments, eventually connecting to make a single route. When in the nearest neighbor algorithm, the edges of the route are built one after the other.

The nearest insertion algorithm is another potential algorithm for this problem. The algorithm can begin with two addresses in a route. The algorithm then finds the next address with the shortest distance from any points within the current route. Compares inserting the new address in the route between the closest address and the next closest addresses in which the route will give the shortest new route.

Just like all the other mentioned algorithms, the nearest insertion algorithm also uses a function to find the shortest distance between two addresses. However, the nearest insertion algorithm starts with an initial route and constantly changes the route, inserting addresses accordingly and changing the previous, incomplete, route until all addresses are visited.

---

## J. Different Approach

If this project were rewritten or attempted again, there would be a time object implemented to improve the way time is maintained. Regarding time within the project, many lines repeat the same calculations. In which all such calculations would be held within the new time object. The time object would have decimal, hour, minute, and second attributes. Functions that will automatically return a displayable time in the correct format (i.e., 10:00 am) and add time to itself given a specific distance and speed.

---

## K1. Verification of Data Structure

The hash table data structure used in the solution does not use any additional libraries or classes. It has an insertion function that takes the package ID number as the key and stores the key and package object. The structure has a lookup function that takes the key, which would be the package ID number, and returns the package object.

The package object holds the following data:

- Package ID number
- Delivery address
- Delivery deadline
- Delivery city
- Delivery zip code

- Package weight
- Delivery status that may include the delivery time

### K1A. Efficiency

The number of packages and the time required for the look-up function in the worst-case would follow a linear relationship, in which the time complexity would be  $O(n)$ . This means the amount of time required is proportionate to the number of packages.

### K1B. Overhead

The number of packages and the space required for the data structure in the worst-case would follow a linear relationship, in which the space complexity would be  $O(n)$ . This means that the amount of space required is proportionate to the number of packages.

### K1C. Implications

The number of trucks and the number of cities would not affect the look-up time and space usage. As the number of trucks and cities nears infinity the time and space complexity of the look-up function will remain  $O(n)$ .

## K2. Other Data Structures

Although they may not be as efficient for this scenario, stack as well as queue data structures could meet the requirements. The lookup function would need to be adjusted to follow the iteration and comparison through the items within a different structure.

### K2a. Data Structure Differences

The hash table data structure follows a key-value pair that can be indexed. Stack data structures follow a last in first out list where the first item added is the last out. Queue data structures follow a first in first out list where the last item added is the last out.

---

## M. Professional Communication

See files within TennyAkiharyPA.01.zip,

---

## L. Sources - Works Cited

James J. *Python: Creating A HASHMAP Using Lists*. YouTube.

<https://www.youtube.com/watch?v=9HFbhPscPU0>. Published January 22, 2016.

Accessed May 8, 2022.

Tepe C. *Webinar-1 - Let's Go Hashing*. Panopto.

<https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=f08d7871-d57a-496e-a6a1-ac7601308c71>. Accessed May 8, 2022.