C950 WGUPS Algorithm Overview

Zachary Zamiska

ID # 003194341

WGU Email: zzamisk@wgu.edu

Tuesday, October 4, 2022

C950 Data Structures and Algorithms II

Introduction

The task was to make a delivery program that satisfies the following requirements:

- Can store the package information in a hash table.

- Uses a self-adjusting heuristic algorithm to find a solution that delivers all packages using under 140 miles and according to the provided requirements.

- Allows the "user" to check the status of any package at any given time.

A. Algorithm Identification

The decision to use the Nearest Neighbor Algorithm to simplify the delivery at the expense of efficiency. It was able to, however, still meet the expectations of the requirements. The majority of the algorithm was done in the truckDeliverPackages() function in conjunction with the minDistanceFrom() function.

B1. Logic Comments

The algorithm uses two functions explained below:

minDistanceFrom(from_address, truck):

1) Establish a baseline minimum variable.
2) Loop through the list of packages in the given truck.
    i. Assign a variable that is the current package iterations address.
    ii. Obtain the distance between the given address (from_address) and the iteration of the current package on the given truck list. (Using the distanceBetween() function)
    iii. Check if the distance between the two packages is smaller than the minimum value.
        1. If the value (distance) is less than the current minimum value:
            a. Assign the minimum value to the distance.
3) After the iterations are complete, return the package that is closest to the address given (from _address)

truckDeliverPackages(truck):

1) Set and assign the starting location (In this case the HUB)
2) While the number of packages currently in the truck is not 0.
    i. For all the packages in the assigned truck (truck).
        1. Change the status of the package to "IN TRANSIT" (It is on the truck to its' destination.)

2. Assign a variable to the package that is the minimum distance from the current package iteration.
3. Remove (deliver) the package and re-iterate for the next package in the truck.

B2. Development Environment

Programming Language: Python 3.10.7

IDE: PyCharm 2022.2.2 (Community Edition)

Build: #PC-222.4167.33, built on September 15, 2022

Runtime version: 17.0.4+7-b469.53 amd64

VM: OpenJDK 64-Bit Server VM by JetBrains s.r.o.

GC: G1 Young Generation, G1 Old Generation

Memory: 2040M

Cores: 8

Non-Bundled Plugins:

com.andrey4623.rainbowcsv (2.0.2)

Operating System: Windows 10 10.0

B3. Space-Time and Big-O

The separate functions list the space-time complexity in the descriptions.

The program's overall space-time complexity is:

$O(n^2)$

B4. Scalability and Adaptability

The self-adjusting data structure that was used in this project was the standard hash table. Due to the constraints of the project the hash table used must not have been the built-in hash function in Python. The hash table does have the ability to be as large or small when required. It could be an empty list if the need should arise. The structure of the hash table, by nature, functions more like a parent than a child in how often things are pulled rather than input.

For the algorithm used to deliver the packages (NNA) the ability to scale it simply depends on the number of packages that the trucks can carry. However, the total amount of packages in the hash table does not have any bearing on the scalability of the program. This means if the total amount of packages is increased the algorithm will not scale to adjust.

B5. Software Efficiency and Maintainability

The program's maintainability is broken into several different modules, compartmentalized, to help fix and adjust calculations and data, as well as write new methods that pertain to each of the module's functions. Essentially, the algorithm is mostly what occurs in the main.py file besides some distance-to-time conversions.

When a change occurs in the programming, if the main function of the program is not altered there should be no errors. There can be "upgrades" or changes to the auxiliary functions without affecting the other actions as well.

B6. Self-Adjusting Data Structures

Self-adjusting data structures have a decided advantage when it comes to space complexity. It is often time fixed as O(1) because the size of the hash table is fixed (when using a chaining hash table).

Another advantage is that not all indexes have to have a unique value. If an array (list) has a size of 256 there are 256 unique indexes. This is not the case with the hash table. You can have multiple items fall under one index.

One disadvantage of a hash table is the lack of direct addressing. In other words, one would need to program a way to "run" the key back through that hashing function. If this type of access does not exist, it would be impossible to access the items in the has table.

Another possible disadvantage is when a situation arises when the hash table provides a poor hash function. This could cause the amount of "buckets" when the items are placed to be quite small causing a large number of items in a few slots.

C. Original Code

The code uses some structures that are well known, but the original user or owner of the material is cited in (Section L).

C1. Identification Information

Found in main.py:

```
"""
First Name: Zachary
Last Name: Zamiska
Student ID: 003194341

NOTE: To correctly run the program please start with MainMenu.py
"""
```

C2. Process and Flow Comments

Almost all the major functions are described.

D. Data Structure

The self-adjusting data structure used in this submission is a hash table. No matter how many packages need to be amended onto the table the function itself does not require a change. This data structure included the proper actions required for the project (an insertion function as well as a look-up function).

D1. Explanation of Data Structure

The hash table stores package information by assigning each package and all its information to a single ID. Once the ID is assigned it is run through the hash function (hash(key) % len(self.table)). All you need to pull out the correct item is the id. This is a chaining hash table so the actual size of the list is a set amount of items established by the hash function given.

The hash table is a data structure that does not require iterating through the whole list to search for the correct item/object. This is just one of the advantages. Another is the ability to store objects in a manner where only one number is needed to access the whole object.

E. Hash Table

Found in Hash.py:

```
def insert(self, key, item):
    bucket = hash(key) % len(self.table)
    bucket_list = self.table[bucket]

    for kv in bucket_list:
        if kv[0] == key:
            kv[1] = item
            return True

    key_value = [key, item]
    bucket_list.append(key_value)
    return True
```

F. Look-Up Function

Found in Hash.py:

```
def search(self, key):

    bucket = hash(key) % len(self.table)
    bucket_list = self.table[bucket]

    for kv in bucket_list:

        if kv[0] == key:
            return kv[1]
    return None
```

G. Interface

Main Menu:

****

1. Print All Package Status and Total Mileage

2. Get a Single Package Status with an ID

3. Get Package Status with a Time

4. Get a Single Package Status with a Time

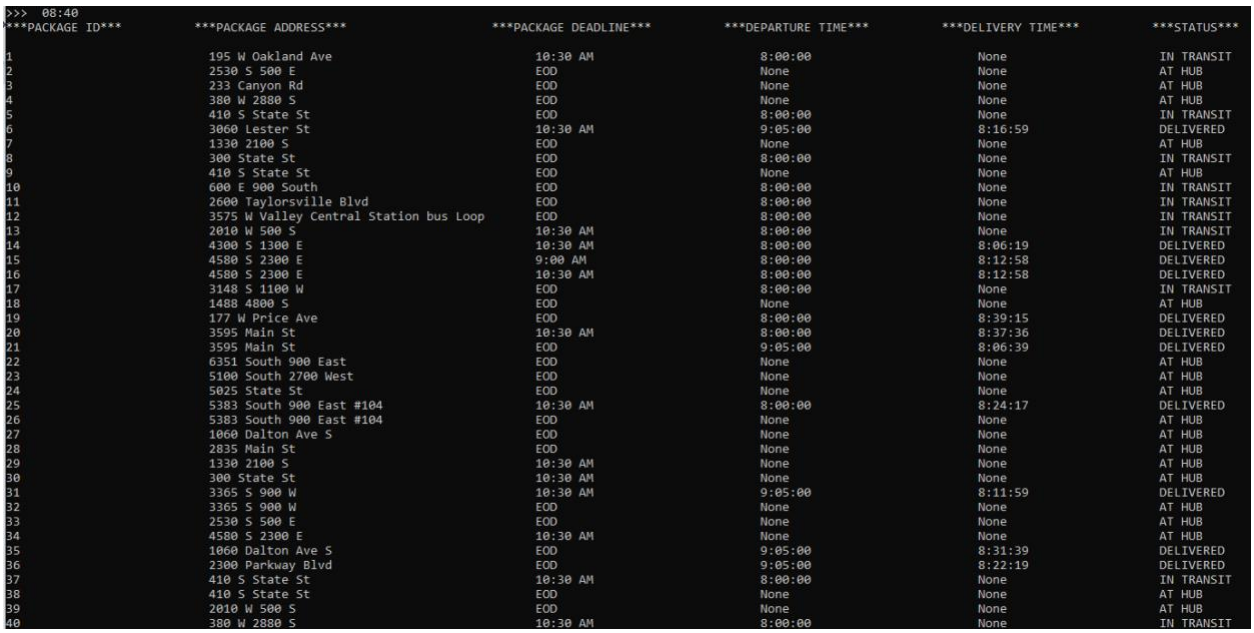
5. Exit the Program

****

>>>

G1. First Status Check

08:40



>>> 08:40

PACKAGE ID	***PACKAGE ADDRESS***	***PACKAGE DEADLINE***	***DEPARTURE TIME***	***DELIVERY TIME***	***STATUS***
1	195 W Oakland Ave	10:30 AM	8:00:00	None	IN TRANSIT
2	2530 S 500 E	EOD	None	None	AT HUB
3	233 Canyon Rd	EOD	None	None	AT HUB
4	380 W 2880 S	EOD	None	None	AT HUB
5	410 S State St	EOD	8:00:00	None	IN TRANSIT
6	3060 Lester St	10:30 AM	9:05:00	8:16:59	DELIVERED
7	1330 2100 S	EOD	None	None	AT HUB
8	300 State St	EOD	8:00:00	None	IN TRANSIT
9	410 S State St	EOD	None	None	AT HUB
10	600 E 900 South	EOD	8:00:00	None	IN TRANSIT
11	2600 Taylorsville Blvd	EOD	8:00:00	None	IN TRANSIT
12	3575 W Valley Central Station bus Loop	EOD	8:00:00	None	IN TRANSIT
13	2010 W 500 S	10:30 AM	8:00:00	None	IN TRANSIT
14	4300 S 1300 E	10:30 AM	8:00:00	8:06:19	DELIVERED
15	4580 S 2300 E	9:00 AM	8:00:00	8:12:58	DELIVERED
16	4580 S 2300 E	10:30 AM	8:00:00	8:12:58	DELIVERED
17	3148 S 1100 W	EOD	8:00:00	None	IN TRANSIT
18	1488 4800 S	EOD	None	None	AT HUB
19	177 W Price Ave	EOD	8:00:00	8:39:15	DELIVERED
20	3595 Main St	10:30 AM	8:00:00	8:37:36	DELIVERED
21	3595 Main St	EOD	9:05:00	8:06:39	DELIVERED
22	6351 South 900 East	EOD	None	None	AT HUB
23	5100 South 2700 West	EOD	None	None	AT HUB
24	5025 State St	EOD	None	None	AT HUB
25	5383 South 900 East #104	10:30 AM	8:00:00	8:24:17	DELIVERED
26	5383 South 900 East #104	EOD	None	None	AT HUB
27	1060 Dalton Ave S	EOD	None	None	AT HUB
28	2835 Main St	EOD	None	None	AT HUB
29	1330 2100 S	10:30 AM	None	None	AT HUB
30	300 State St	10:30 AM	None	None	AT HUB
31	3365 S 900 W	10:30 AM	9:05:00	8:11:59	DELIVERED
32	3365 S 900 W	EOD	None	None	AT HUB
33	2530 S 500 E	EOD	None	None	AT HUB
34	4580 S 2300 E	10:30 AM	None	None	AT HUB
35	1060 Dalton Ave S	EOD	9:05:00	8:31:39	DELIVERED
36	2300 Parkway Blvd	EOD	9:05:00	8:22:19	DELIVERED
37	410 S State St	10:30 AM	8:00:00	None	IN TRANSIT
38	410 S State St	EOD	None	None	AT HUB
39	2010 W 500 S	EOD	None	None	AT HUB
40	380 W 2880 S	10:30 AM	8:00:00	None	IN TRANSIT

G2. Second Status Check

09:40

```
>>> 9:40
***PACKAGE ID***        ***PACKAGE ADDRESS***                  ***PACKAGE DEADLINE***        ***DEPARTURE TIME***        ***DELIVERY TIME***        ***STATUS***

1                       195 W Oakland Ave                      10:30 AM                      8:00:00                     9:02:55                    DELIVERED
2                       2530 S 500 E                           EOD                           None                        None                       AT HUB
3                       233 Canyon Rd                          EOD                           9:05:00                     8:47:39                    DELIVERED
4                       380 W 2880 S                           EOD                           None                        None                       AT HUB
5                       410 S State St                         EOD                           8:00:00                     None                       IN TRANSIT
6                       3060 Lester St                         10:30 AM                      9:05:00                     8:16:59                    DELIVERED
7                       1330 2100 S                            EOD                           9:05:00                     9:07:17                    DELIVERED
8                       300 State St                           EOD                           8:00:00                     None                       IN TRANSIT
9                       410 S State St                         EOD                           None                        None                       AT HUB
10                      600 E 900 South                        EOD                           8:00:00                     None                       IN TRANSIT
11                      2600 Taylorsville Blvd                 EOD                           8:00:00                     9:31:15                    DELIVERED
12                      3575 W Valley Central Station bus Loop EOD                           8:00:00                     8:43:55                    DELIVERED
13                      2010 W 500 S                           10:30 AM                      8:00:00                     None                       IN TRANSIT
14                      4300 S 1300 E                          10:30 AM                      8:00:00                     8:06:19                    DELIVERED
15                      4580 S 2300 E                          9:00 AM                       8:00:00                     8:12:58                    DELIVERED
16                      4580 S 2300 E                          10:30 AM                      8:00:00                     8:12:58                    DELIVERED
17                      3148 S 1100 W                          EOD                           8:00:00                     9:13:55                    DELIVERED
18                      1488 4800 S                            EOD                           9:05:00                     None                       IN TRANSIT
19                      177 W Price Ave                        EOD                           8:00:00                     8:39:15                    DELIVERED
20                      3595 Main St                           10:30 AM                      8:00:00                     8:37:36                    DELIVERED
21                      3595 Main St                           EOD                           9:05:00                     8:06:39                    DELIVERED
22                      6351 South 900 East                    EOD                           9:05:00                     9:39:36                    DELIVERED
23                      5100 South 2700 West                   EOD                           9:05:00                     None                       IN TRANSIT
24                      5025 State St                          EOD                           9:05:00                     None                       IN TRANSIT
25                      5383 South 900 East #104               10:30 AM                      8:00:00                     8:24:17                    DELIVERED
26                      5383 South 900 East #104               EOD                           9:05:00                     9:35:16                    DELIVERED
27                      1060 Dalton Ave S                      EOD                           None                        None                       AT HUB
28                      2835 Main St                           EOD                           None                        None                       AT HUB
29                      1330 2100 S                            10:30 AM                      9:05:00                     9:07:17                    DELIVERED
30                      300 State St                           10:30 AM                      9:05:00                     8:49:39                    DELIVERED
31                      3365 S 900 W                           10:30 AM                      9:05:00                     8:11:59                    DELIVERED
32                      3365 S 900 W                           EOD                           None                        None                       AT HUB
33                      2530 S 500 E                           EOD                           None                        None                       AT HUB
34                      4580 S 2300 E                          10:30 AM                      9:05:00                     9:23:57                    DELIVERED
35                      1060 Dalton Ave S                      EOD                           9:05:00                     8:31:39                    DELIVERED
36                      2300 Parkway Blvd                      EOD                           9:05:00                     8:22:19                    DELIVERED
37                      410 S State St                         10:30 AM                      8:00:00                     None                       IN TRANSIT
38                      410 S State St                         EOD                           9:05:00                     8:52:58                    DELIVERED
39                      2010 W 500 S                           EOD                           None                        None                       AT HUB
40                      380 W 2880 S                           10:30 AM                      8:00:00                     9:06:35                    DELIVERED
```

G3. Third Status Check

12:30

```
>>> 12:30
***PACKAGE ID***        ***PACKAGE ADDRESS***                  ***PACKAGE DEADLINE***        ***DEPARTURE TIME***        ***DELIVERY TIME***        ***STATUS***

1                       195 W Oakland Ave                      10:30 AM                      8:00:00                     9:02:55                    DELIVERED
2                       2530 S 500 E                           EOD                           10:20:00                    10:29:20                   DELIVERED
3                       233 Canyon Rd                          EOD                           9:05:00                     8:47:39                    DELIVERED
4                       380 W 2880 S                           EOD                           10:20:00                    10:36:19                   DELIVERED
5                       410 S State St                         EOD                           8:00:00                     10:10:34                   DELIVERED
6                       3060 Lester St                         10:30 AM                      9:05:00                     8:16:59                    DELIVERED
7                       1330 2100 S                            EOD                           9:05:00                     9:07:17                    DELIVERED
8                       300 State St                           EOD                           8:00:00                     10:13:53                   DELIVERED
9                       410 S State St                         EOD                           10:20:00                    11:12:58                   DELIVERED
10                      600 E 900 South                        EOD                           8:00:00                     10:23:13                   DELIVERED
11                      2600 Taylorsville Blvd                 EOD                           8:00:00                     9:31:15                    DELIVERED
12                      3575 W Valley Central Station bus Loop EOD                           8:00:00                     8:43:55                    DELIVERED
13                      2010 W 500 S                           10:30 AM                      8:00:00                     9:59:54                    DELIVERED
14                      4300 S 1300 E                          10:30 AM                      8:00:00                     8:06:19                    DELIVERED
15                      4580 S 2300 E                          9:00 AM                       8:00:00                     8:12:58                    DELIVERED
16                      4580 S 2300 E                          10:30 AM                      8:00:00                     8:12:58                    DELIVERED
17                      3148 S 1100 W                          EOD                           8:00:00                     9:13:55                    DELIVERED
18                      1488 4800 S                            EOD                           9:05:00                     10:06:56                   DELIVERED
19                      177 W Price Ave                        EOD                           8:00:00                     8:39:15                    DELIVERED
20                      3595 Main St                           10:30 AM                      8:00:00                     8:37:36                    DELIVERED
21                      3595 Main St                           EOD                           9:05:00                     8:06:39                    DELIVERED
22                      6351 South 900 East                    EOD                           9:05:00                     9:39:36                    DELIVERED
23                      5100 South 2700 West                   EOD                           9:05:00                     10:04:56                   DELIVERED
24                      5025 State St                          EOD                           9:05:00                     9:49:56                    DELIVERED
25                      5383 South 900 East #104               10:30 AM                      8:00:00                     8:24:17                    DELIVERED
26                      5383 South 900 East #104               EOD                           9:05:00                     9:35:16                    DELIVERED
27                      1060 Dalton Ave S                      EOD                           10:20:00                    10:56:58                   DELIVERED
28                      2835 Main St                           EOD                           10:20:00                    10:33:00                   DELIVERED
29                      1330 2100 S                            10:30 AM                      9:05:00                     9:07:17                    DELIVERED
30                      300 State St                           10:30 AM                      9:05:00                     8:49:39                    DELIVERED
31                      3365 S 900 W                           10:30 AM                      9:05:00                     8:11:59                    DELIVERED
32                      3365 S 900 W                           EOD                           10:20:00                    10:41:58                   DELIVERED
33                      2530 S 500 E                           EOD                           10:20:00                    10:29:20                   DELIVERED
34                      4580 S 2300 E                          10:30 AM                      9:05:00                     9:23:57                    DELIVERED
35                      1060 Dalton Ave S                      EOD                           9:05:00                     8:31:39                    DELIVERED
36                      2300 Parkway Blvd                      EOD                           9:05:00                     8:22:19                    DELIVERED
37                      410 S State St                         10:30 AM                      8:00:00                     10:10:34                   DELIVERED
38                      410 S State St                         EOD                           9:05:00                     8:52:58                    DELIVERED
39                      2010 W 500 S                           EOD                           10:20:00                    11:02:18                   DELIVERED
40                      380 W 2880 S                           10:30 AM                      8:00:00                     9:06:35                    DELIVERED
```

H. Screenshots of Code Execution

Program (main menu option 1) ran to completion:

PACKAGE ID	***PACKAGE ADDRESS***	***PACKAGE DEADLINE***	***DEPARTURE TIME***	***DELIVERY TIME***	***STATUS***
1	195 W Oakland Ave	10:30 AM	8:00:00	9:02:55	DELIVERED
2	2530 S 500 E	EOD	10:20:00	10:29:20	DELIVERED
3	233 Canyon Rd	EOD	9:05:00	8:47:39	DELIVERED
4	380 W 2880 S	EOD	10:20:00	10:36:19	DELIVERED
5	410 S State St	EOD	8:00:00	10:10:34	DELIVERED
6	3060 Lester St	10:30 AM	9:05:00	8:16:59	DELIVERED
7	1330 2100 S	EOD	9:05:00	9:07:17	DELIVERED
8	300 State St	EOD	8:00:00	10:13:53	DELIVERED
9	410 S State St	EOD	10:20:00	11:12:58	DELIVERED
10	600 E 900 South	EOD	8:00:00	10:23:13	DELIVERED
11	2600 Taylorsville Blvd	EOD	8:00:00	9:31:15	DELIVERED
12	3575 W Valley Central Station bus Loop	EOD	8:00:00	8:43:55	DELIVERED
13	2010 W 500 S	10:30 AM	8:00:00	9:59:54	DELIVERED
14	4300 S 1300 E	10:30 AM	8:00:00	8:06:19	DELIVERED
15	4580 S 2300 E	9:00 AM	8:00:00	8:12:58	DELIVERED
16	4580 S 2300 E	10:30 AM	8:00:00	8:12:58	DELIVERED
17	3148 S 1100 W	EOD	8:00:00	9:13:55	DELIVERED
18	1488 4800 S	EOD	9:05:00	10:06:56	DELIVERED
19	177 W Price Ave	EOD	8:00:00	8:39:15	DELIVERED
20	3595 Main St	10:30 AM	8:00:00	8:37:36	DELIVERED
21	3595 Main St	EOD	9:05:00	8:06:39	DELIVERED
22	6351 South 900 East	EOD	9:05:00	9:39:36	DELIVERED
23	5100 South 2700 West	EOD	9:05:00	10:04:56	DELIVERED
24	5025 State St	EOD	9:05:00	9:49:56	DELIVERED
25	5383 South 900 East #104	10:30 AM	8:00:00	8:24:17	DELIVERED
26	5383 South 900 East #104	EOD	9:05:00	9:35:16	DELIVERED
27	1060 Dalton Ave S	EOD	10:20:00	10:56:58	DELIVERED
28	2835 Main St	EOD	10:20:00	10:33:00	DELIVERED
29	1330 2100 S	10:30 AM	9:05:00	9:07:17	DELIVERED
30	300 State St	10:30 AM	9:05:00	8:49:39	DELIVERED
31	3365 S 900 W	10:30 AM	9:05:00	8:11:59	DELIVERED
32	3365 S 900 W	EOD	10:20:00	10:41:58	DELIVERED
33	2530 S 500 E	EOD	10:20:00	10:29:20	DELIVERED
34	4580 S 2300 E	10:30 AM	9:05:00	9:23:57	DELIVERED
35	1060 Dalton Ave S	EOD	9:05:00	8:31:39	DELIVERED
36	2300 Parkway Blvd	EOD	9:05:00	8:22:19	DELIVERED
37	410 S State St	10:30 AM	8:00:00	10:10:34	DELIVERED
38	410 S State St	EOD	9:05:00	8:52:58	DELIVERED
39	2010 W 500 S	EOD	10:20:00	11:02:18	DELIVERED
40	380 W 2880 S	10:30 AM	8:00:00	9:06:35	DELIVERED

```
___TOTAL DISTANCE___
       129 MILES
```

I1. Strengths of Chosen Algorithm

One of the largest strengths of the Nearest Neighbor algorithm is why it is one of the staples of Computer Science (and often the first algorithm taught). For one, it is very simple to implement into a problem, which I will state later can also cause efficiency to suffer. Another strength of this algorithm is the low number of variables required to execute it. Again, if the number of inputs increases, this can also qualify as a weakness. It is what they call a “lazy algorithm” and NN is not quite scalable in the long term.

I2. Verification of Algorithm

Requirements:

(NOTE: All the requirement info can be found using selection 1 in the main menu)

Total Milage is Less Than 140 miles:

- The following code handles the arithmetic to find the total mileage:

```
print("\t%i MILES" % int(
    main.truck_1.getDistance() + main.truck_1.getDistance() +
main.truck_1.getDistance()))
```

- This can be found in the MainMenu.py file. It adds all the total distance from the program (main.py).

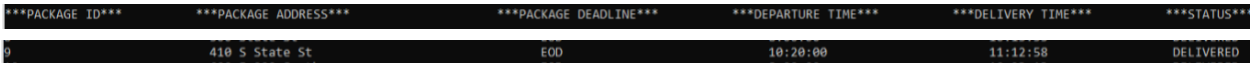
```
___TOTAL DISTANCE___
       129 MILES
```

All The Packages Were Delivered on Time:

- Since the trucks, in the case of my submission, are loaded heuristically the requirements are simply handled by what truck they are loaded on.
- I was able to simply adjust what package was loaded on what truck to make sure the requirement was satisfied. The packages were all delivered by the end of the day at least and all the earlier requirements were satisfied. (See section G3)

All Packages Were Delivered According to Specifications:

- Just like the previous requirement the trucks, in the case of my submission, are loaded heuristically the requirements are simply handled by what truck they are loaded on.
- An example of this can be demonstrated by truck 3.
    - This truck has the package that is "leaving" at the latest. That of which is package #9. Since every other package has a deadline of EOD, the truck can wait to leave till 10:20 AM.



PACKAGE ID	***PACKAGE ADDRESS***	***PACKAGE DEADLINE***	***DEPARTURE TIME***	***DELIVERY TIME***	***STATUS***
9	410 S State St	EOD	10:20:00	11:12:58	DELIVERED

I3. Other Possible Algorithm

Nearest Insertion Heuristic:

- Although the NI algorithm has the same Big-O analysis as the Nearest Neighbor ($O(n^2)$) it is slightly more efficient in finding a path. The TSP is an NP-hard problem there are only algorithms that can find the best solution more often than others. To utilize the Nearest Insertion one would need to start with two points. In the case of the problem of WGUPS, it would be two addresses. Next, you would find the closest address to either of the addresses already on the tour. Doing this until all of the addresses have been visited will net you a complete tour.

2-OPT Heuristic :

- Using the concept of 2-Changes a complete tour removes two vertices and replaces them with another two. This comes with some qualifiers, however. The first qualifier is that it must be a complete cycle (no disjointed) and another is it shouldn't be the original cycle.
- The 2-Opt algorithm takes this concept of 2-Change and repeats it as long as there is a more efficient ("shorter") route. When you get to the point where the route is equal to or greater than the algorithm stops and you have the given tour.

I3A. Algorithm Differences

Nearest Insertion Heuristic:

- One of the largest differences between the Nearest Insertion algorithm and the Closet Neighbor algorithm is the final vertices. For the CN these vertices are often very inefficient (high in magnitude). The NI algorithm helps alleviate this by "stretching" (inserting stops into) the route out with the current route rather than creating the route as a line and connecting the last stop with the first post hoc.
- Another difference is that once more locations are added to the pool it increases the amount of possible next locations. This is different from the going single file.

2-OPT Heuristic:

- In this algorithm in very rare cases the number of iterations that are used could be exponential ($O(2^n)$). This, however, can be very rare.
- An advantage of this algorithm is the fact that it is an "anytime algorithm". This means that it could be stopped after a certain or given amount of time. This can help greatly with the time requirements.

J. Different Approach

One approach that could be beneficial is to make the program more scalable. What is meant by that is the input of logic to help or completely load the truck through programming (relative to the requirements in the notes).

Another perceived issue is to use of a more efficient algorithm to deliver the package. The NN algorithm is (I hate to say it) the way that tends to be the most inefficient when it comes to solving the TSP.

K1. Verification of Data Structure

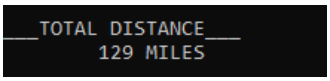
Requirements:

Total Mileage is Less Than 140:

- The following code handles the arithmetic to find the total mileage:

```
print("\t%i MILES" % int(
    main.truck_1.getDistance() + main.truck_1.getDistance() +
main.truck_1.getDistance()))
```

- This can be found in the MainMenu.py file. It adds all the total distance from the program (main.py).



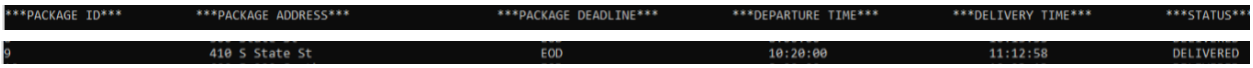
All Packages Delivered on Time:

- Since the trucks, in the case of my submission, are loaded heuristically the requirements are simply handled by what truck they are loaded on.

- By simply adjusting what package was loaded on what truck to make sure the requirement was satisfied. The packages were all delivered by the end of the day at least and all the earlier requirements were satisfied. (See section G3)

All Packages are Delivered According to Their Delivery Specifications:

- Just like the previous requirement the trucks, in the case of my submission, are loaded heuristically the requirements are simply handled by what truck they are loaded on.
- An example of this can be demonstrated by truck 3.
    - This truck has the package that is "leaving" at the latest. That of which is package #9. Since every other package has a deadline of EOD, the truck can wait to leave till 10:20 AM.



PACKAGE ID	***PACKAGE ADDRESS***	***PACKAGE DEADLINE***	***DEPARTURE TIME***	***DELIVERY TIME***	***STATUS***
9	410 S State St	EOD	10:20:00	11:12:58	DELIVERED

Hash Table:

Efficient Hash Table:

- The hash table is present, and it has a time complexity of O(n), this can be seen as relatively efficient.

Look-Up Function:

- The lookup function is present and has a time complexity of O(n) as well.

Reporting Through UI:

- Using the MainMenu.py, the user can obtain all the required information.

Correct Information:

- All the information shown from the main menu is proper and correct.

K1A. Efficiency

The use of a hash table allows the program to have a consistent ability to access items in the list by using the hash function and the ID of the object. The fact that accessing a package does not require an iteration of a loop the speed of the function is O(1) no matter how many items are added to the list. However, there could be a situation (using a loop to add items) where the time complexity can be O(n). This is, nevertheless, the worst-case scenario.

K1B. Overhead

The way that packages are loaded into the program is by the hash table function in the file Hash.py named insert(). The space complexity of a closed addressing hash table depends on the number of elements it currently stores. This means that the space complexity of O(n). The overhead of the hash table used is dependent on the number of packages that the user would use.

K1C. Implications

Unlike with the packages above, there is a module within the program that creates a new truck. To instantiate a new truck has a time complexity of O(1), which means that there are no additional time requirements to add another truck. As far as space complexity is concerned when you add a truck to the program it takes up a consistent (or constant) amount of space. Therefore, the space complexity is O(1) as well.

K2. Other Data Structures

Linked Lists:

- A linked list is a list (array) that is stored using pointers instead of being in a sequence one after the other (contiguous).

Graph:

- A graph data structure would use nodes and vertices to "map" out different nodes in the graph.

K2a. Data Structure Differences

Linked Lists:

- A linked list would differ from the use of a hash table by causing it to be much easier to add and remove elements, but much more difficult, if not impossible, to access elements that are out of order. In my opinion, this would be a terrible alternative to the hash table.

Graph:

- The biggest benefit that would be instigated using a graph as a data structure would be the fact that it is a great "real-world" representation of a map. It would ease the ability to run any TSP algorithm as it helps map out the structure.

M. Professional Communication

All text was checked for spelling and grammar errors.

L. Sources - Works Cited

Lysecky, R., & Vahid, F. (2018, June). *C950: Data Structures and Algorithms II*. zyBooks. Retrieved March 22, 2021, from https://learn.zybooks.com/zybook/WGUC950AY20182019/

Andrews, D. (2019, September 4). *Big O: How to calculate time and Space Complexity*. In Out Code. Retrieved October 10, 2022, from https://www.inoutcode.com/concepts/big-o/

Linked List Data Structure. GeeksforGeeks. (n.d.). Retrieved October 10, 2022, from https://www.geeksforgeeks.org/data-structures/linked-list/#singlyLinkedList

Rosenkrantz, Stearns, & Lewis. (1974). *TRAVELING SALESMAN PROBLEM: Insertion Algorithms*. Traveling Salesman Problem. Retrieved October 10, 2022, from https://www2.isye.gatech.edu/~mgoetsch/cali/VEHICLE/TSP/TSP009__.HTM

Traveling salesman problem. (n.d.). Retrieved October 10, 2022, from https://www2.isye.gatech.edu/~mgoetsch/cali/VEHICLE/TSP/TSP009__.HTM

Weru, L. (2021, August 24). *11 animated algorithms for the traveling salesman problem*. STEM Lounge. Retrieved October 10, 2022, from https://stemlounge.com/animated-algorithms-for-the-traveling-salesman-problem/#:~:text=3%3A%20Nearest%20Insertion&text=One%20implementation%20of%20Nearest%20Insertion,to%20be%20the%20shortest%20possible.

Western Governors University. (n.d.). *C950 WGUPS Project Implementation Steps*. C950 WGUPS Project Implementation Steps - Supplemental Resources. Retrieved October 10, 2022, from https://srm--c.na127.visual.force.com/apex/coursearticle?Id=kA03x000001DbBGCA0