Anthony Coots

ID #010958511

June 16th, 2023.

C950 Data Structures and Algorithms II

Introduction.

Everyday tasks are relying on technology more than ever. Though, this technology was not born with an adaptive brain and is to be constructed programmatically to achieve such everyday tasks. The ‘brain’, rather the functionality, of these tasks comes from the data structures and algorithms in place to attempt these tasks. These tasks are dedicated, meaning the algorithm and data structures are constructed for a particular task only, generally.

An example of an everyday task is mail delivery. Packages are received from multiple locations, some sooner rather than later. Packages are loaded onto a truck for delivery. Then algorithms are in place to find an efficient path to deliver the appropriate number of packages in appropriate time. Algorithms can also be in place to decide which packages go on which trucks, though, due to many constraints such as time (on time, delayed, etc.) or weight or size, an algorithm to consider these is not a simple one at that, which often results in manual truck dictation.

The information following is of project work describing a simple yet efficient method to the data structures and algorithms for a mail delivery task of everyday use. The project is given from WGU, course C950, and grants multiple assumptions before construction such that:

- Each truck can carry a maximum of 16 packages, and the ID number of each package is unique.
- The trucks travel at an average speed of 18 miles per hour and have an infinite amount of gas with no need to stop.
- There are no collisions.

- Three trucks and two drivers are available for deliveries. Each driver stays with the same truck as long as that truck is in service.
- Drivers leave the hub no earlier than 8:00 a.m., with the truck loaded, and can return to the hub for packages if needed.
- The delivery and loading times are instantaneous, i.e., no time passes while at a delivery or when moving packages to a truck at the hub (that time is factored into the calculation of the average speed of the trucks).
- There is up to one special note associated with a package.
- The delivery address for package #9, *Third District Juvenile Court*, is wrong and will be corrected at 10:20 a.m. WGUPS is aware that the address is incorrect and will be updated at 10:20 a.m. However, WGUPS does not know the correct address (410 S State St., Salt Lake City, UT, 84111) until 10:20 a.m.
- The distances provided in the WGUPS Distance Table are equal regardless of the direction traveled.
- The day ends when all 40 packages have been delivered.

The project is of requirements alphabetized A though M.

A. Algorithm Identification.

This section states that a self-adjusting algorithm must be identified that is used to create the program to deliver the packages. The program makes use of an algorithm known as the 'Nearest Neighbor' algorithm.

Requirement B is divided into 6 sections, B1, …, B6:

B1. Logic Comments.

The algorithm's construction itself, at least the concept of it, is rather elementary. The construction is as follows:

Preface, before the algorithm is executing:

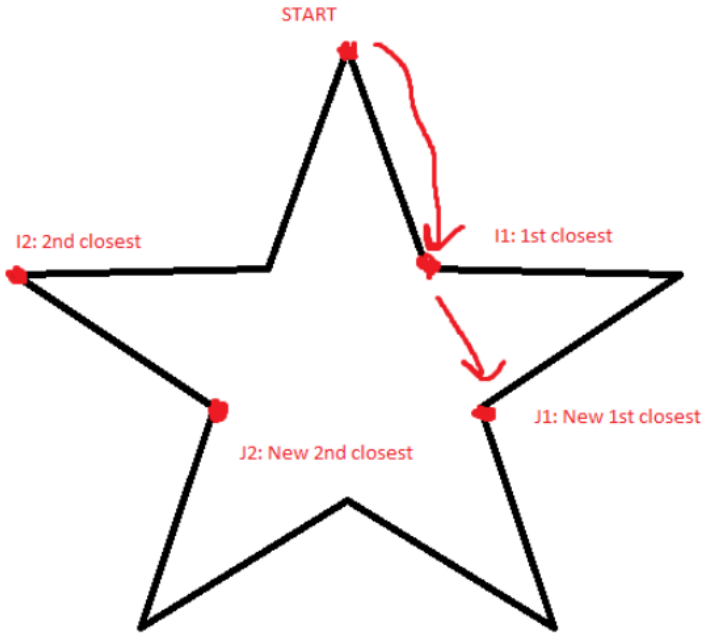
a. An import of packages related data is read through the compiled code from the 'PackageTable.csv' file. Then, an import of addresses and distances is read from the 'DistanceTableFormatted.csv' file. This data can be read in CSV format with the use of programming language 'Python'.
b. The read package information is set into a table, in this particular project a hash table is used and will be further explained later.
c. The package data is sorted properly such that each package is of a respective key-value pair in memory, then each truck object package ID's list (a list each truck has given by its class) are set by manual (human) computation. Also, the trucks initial address is the mail delivery hub.

'Nearest Neighbor' algorithm:

1. For a list of packages, for each package identify the package's string address which is defined in the Packages class.
2. With use of a for loop, for each package in the trucks' list, find a corresponding position of an address ID number, given by the row integer of the array of addresses set by reading the addresses array (see preface 1).

3. For each package address integer returned, a separate calculation must be done on the same table, which gives the distances between two addresses. Knowing the initial address is at the mail delivery hub, for each address integer generated, there is a temporary list made consisting of all package IDs and the distance between the current address and the address of the packages not yet delivered.
4. Then, a package is 'delivered', removed from the truck objects list. This must reconsider the current address has now changed to the address of the package just delivered. With the list of packages to be delivered now removing one package at a time, the list is reconstructed with new comparisons of the now new current address and the distances between the next packages from said current address.
5. One final trip back to the hub is added into the algorithm to consider the current address and the address of the mail delivery hub so that the trucks consider the trip back after the last package is delivered.

I.E., a graphical demonstration mid-route:



A package list will start at ‘START’ (identified in photo) then detect that ‘I1’ is its closest address post temporarily list sort and that ‘I2’ is the second closest address. Once ‘START’ makes the jump to address ‘I1’, ‘START’ must be removed from the list and new closest points must be calculated.

B2. Development Environment.

The hardware and software component used to develop the program is as listed:

- Processor: i7-7700k
- Main Memory: 16 GB
- Operating System: Windows 10 64-bit
- Python v3.10 64-bit. Please note that there is use of the ‘match’ syntax within the code of this development, execution will render error if a version below v3.10 is used.

B3. Space-time complexity.

Starting with the algorithm construction in B1.2, the forementioned for-loop gives a time-complexity of O(N) since the for-loop searches linearly through the address column, a choice is made after searching through each entry within the column for a matching string of a packages address. After doing so, the current address and the next address array integer location points are used for a row and column. The time and space-complexity are identical, being O(1) as the location values are determined and only once cell of the array is assessed. The space-complexity makes use of linear time and space-complexity in B1.3 and B1.4 where the number of packages the program must iterate through increases or decreases, this is considered linear in the sense that with the addition or subtraction of packages on a truck occur, this is not constant as the number of packages a truck may have is variable.

The fundamental concept of the ‘Nearest Neighbor’ algorithm seems to be that the algorithm is linear, O(N) in both respects to time and space. More input, more neighbors, more time and size, if less then less; Should there be more datapoints, n, then n more datapoints are to be considered taking more time and space.

A part of the code that sorts the temporary lists should not go undocumented, that being, the temporary list sort for which packages are closest after getting the package distances with respective package IDs comes from a native Python function, is of time-complexity O(NlogN), which could play a key factor in development for a number of packages higher than 40 (the project amount). The space-complexity is linear, O(N).

B4. Scalability and Adaptability.

The ‘Nearest Neighbor’ algorithm is sufficient still when of a scope much bigger than 40 deliveries. O(N), linear time / space-complexity is linear. Suppose that, Salt Lake City were to implement such an approach for routes. The problem then becomes human, such as, does one hire more trucks, more people, or etcetera to deliver more packages understanding that trucks can only hold so many packages and the employed have limits not to be exceeded (such as average speed). Another human sided restriction is time handling, there are only 24 hours in a day, and packages may only be delivered between business hours ranging anywhere from 6 a.m. – 9 p.m. (at least, that is appropriate business rule, these hours fluctuate). There is only so much that can be done humanly, so a mail delivery system making use of ‘Nearest Neighbor’ algorithm is plenty effective.

The data implementation draws little concern as the size of storage becomes more and more available, data is only needed from two data files, as listed in B1.a, considering the truck loading is still of manual contribution. Though an identifiable computational issue stems away from the algorithm itself. That being the use of the hash table. As a database administrator, an issue of database degrade occurs when hash functions may not move properly within referenced lists (data sets). Collisions are when more than one key value pair is stored within a ‘bucket’ of an array post modulo hash function. However, per the project assumptions, there are no collisions (see Introduction).

Moreover, presenting the argument that scalability and adaptability has more human considerations before a computer problem, considering efficacy.

B5. Software Efficiency and Maintainability.

The software itself is not very complex. Only two classes are needed, packages and trucks, and the needed package information is stored in a hash table with the assumption that there are no collisions. As function / definitions are made within the Python script ‘Main.py’ how the data is moving throughout the script is seen with relative naming conventions as the variables of each object packages and trucks. Good readability and comments on code should be of top priority upon all program development especially when in the situation of project submission. Please see the comments within the code to ensure further explanation, software efficiency and maintainability.

B6. Hash Table Strengths and Weaknesses.

Hash tables are effective data structures when making use of keys and values associated to that key. A hash table, is an array making use of cells as “buckets”. The buckets store values usually received after a modulo function is performed upon the key of a key-value pair. With storing such comes a fast way of accessing a key-value pair with the assist of a hash search function (in ‘HashMap.py’, this is the ‘hashFind’ function / definition). Should there have been collisions, rather than linearly searching 40 cells in a normal array, using a hash search function presents a situation that, given in key to be searched, will return a modulo operation value to what bucket a key-value pair resides. I mention once more that collisions are a hash tables biggest weakness, that it may cause database degrade along with poor memory performance when a bucket ‘fills up’.

C. Original Code.

Please run the Python code in version 3.10 (64-bit).

C1. Identification Information.

```
#+-------------------------------------------------------------------------------+#
# Title:        C950 WGUPS
# Script:       main.py
# Name:         Anthony Coots
# Student ID:   010958511
#
# MODIFICATION HISTORY:                                         DATE:
# ---------------------------------------------                 ----------------#
# v1.0 - Creation of Main.py using source below,                06/06/2023
#        able to import data via CSV files, started
#        ordering trucks packages lists for which
#        truck is responsible for which package.
#
# v1.1 - Code to implement data from csv into                   06/07/2023
#        package objects. Successfully implemented
#        and read a packages hash map. Confirming
#        useful structure of HashMap() in HashMap.py
#
#        Added a distance function that reads the
#        distance table excel turned csv and will
#        accurately give the float distance between
#        two points.
#
#        Started development of nearest neighbor
#        algorithm.
#
# v1.2 - Started development of truckRoute function            06/08/2023
#        that loops through and updates the trucks
#        trip (odometer, time, last address, etc)
#
# v1.3 - Final adjustments and comments made.                   06/09/2023
#
# v1.4 - Removed gui. Results are now displayed in
#        the command line interface.
#+-------------------------------------------------------------------------------+#
```

C2. Process and Flow Comments.

Please see ‘Main.py’, ‘Packages.py’ and ‘HashMap.py’ Python v3.10 scripts.

D. Data Structure.

Please see ‘HashMap.py’ Python v3.10 script.

A hash table is defined within the zip-package under the aforementioned script.

D1. Explanation of Data Structure.

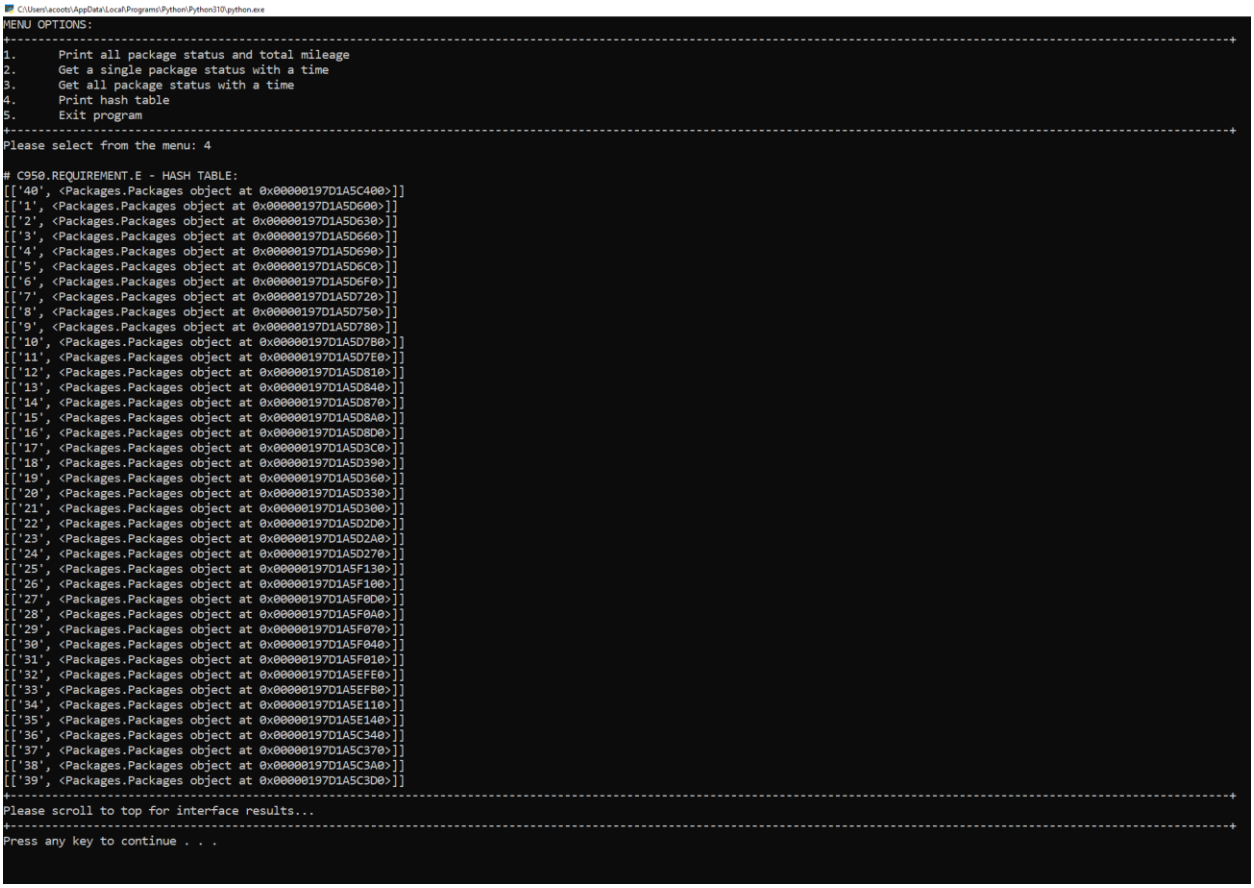
Hash Tables are arrays used to hold key-value pairs and efficiently sort those with the use of ‘buckets’ (cells of the array) that hold one or more key-value pairs upon assignment. Hash tables are sorted using a hash function ‘def hashFunction’ (‘HashMap.py’ Python script) that will take the key and typically perform a modulo operation to assign which cell / bucket the key-value pair will reside. Fortunately, the assumption of the project states that there are no collisions, meaning there is no more than one key-value pair in a cell / bucket, therefore, as declared in the code, the size of the hash table is 40 due to the 40 packages listed in the given excel file. Upon reading the packages data with the python csv module, the packages are assigned to the hash table, making use of the packages ID as the key. The value of a key-value pair is set by the data read for each package of corresponding package IDs.

E. Hash Table.

Printed are the results of the hash table, followed by a screen shot of the hash insertion function.

Please see ‘HashMap.py’ Python v3.10 script.

Results:



Insertion Function:

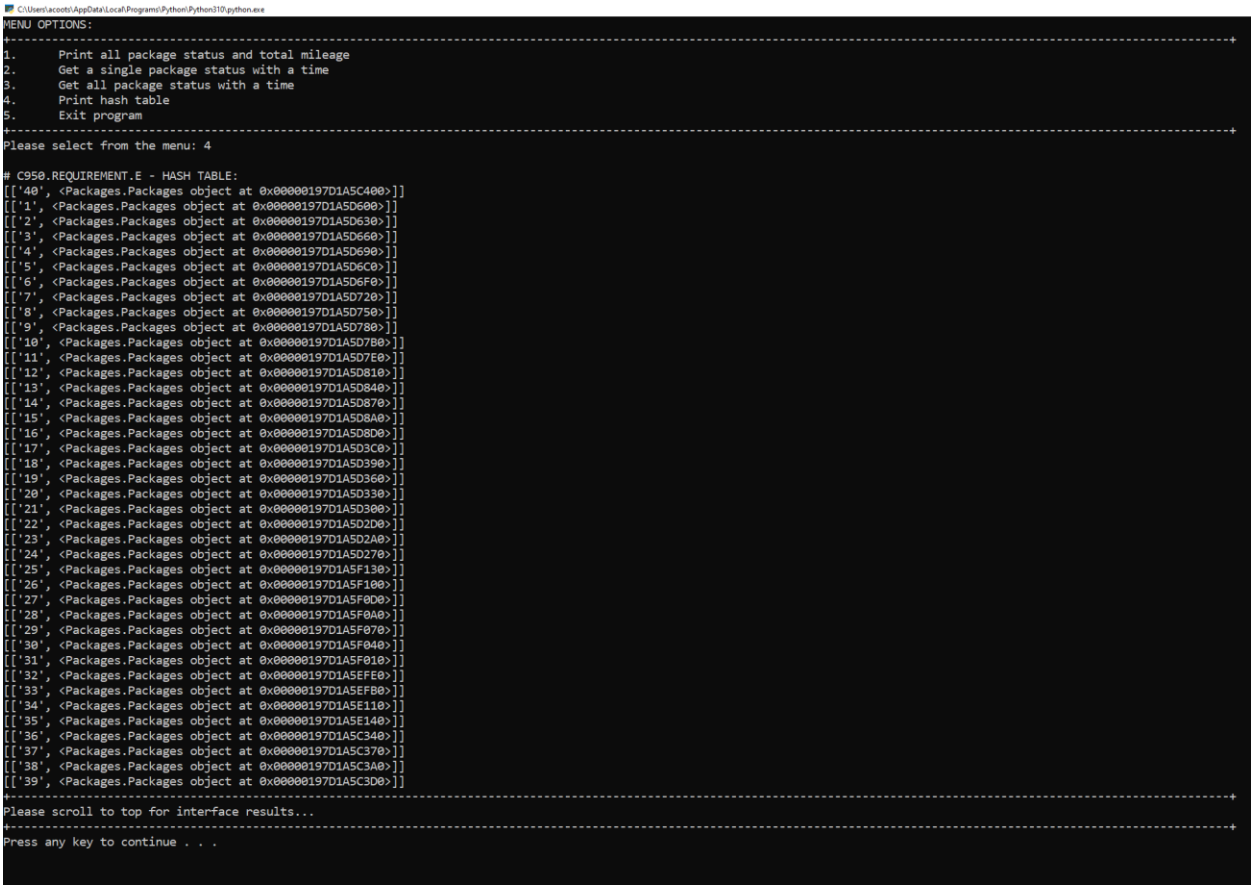
```python
# C950.REQUIREMENT.E: DEVELOP A HASH TABLE, WITHOUT USING ANY ADDITIONAL LIBRARIES OR CLASSES, THAT HAS AN INSERTION FUNCTION... #
    # Add function to take key and value, maps to appropriate cell of hash table.
    def hashAdd(self, key, value):
        # key_index represents the index value of hash table for respective key. Uses hashFunction() above.
        key_index = self.hashFunction(key)
        # key_value is what is to be inserted into the respective cell of the index found with key_index.
        key_value = [key, value]

        # Check to see if the cell is empty.
        if self.map[key_index] is None:
            # If value of hash map at respective index is of empty then add a new list containing the key value pair.
            self.map[key_index] = list([key_value])
            # Return Validation.
            return True

        # Should the cell NOT be empty.
        else:
            # Check to see if key already exists within the array.
            for pair in self.map[key_index]:
                # If the 1st value matches the key.
                if pair[0] == key:
                    # Then update the value of the key-value pair.
                    pair[1] = value
                    # Return Validation.
                    return True
                # If the key is not being used / found then append the list at that index.
                self.map[key_index].append(key_value)
                # Return Validation
                return True
```

F. Look-up Function.

Printed are the results of the hash table, followed by a screen shot of the hash find function.

Please see ‘HashMap.py’ Python v3.10 script.

Results:



Find Function:

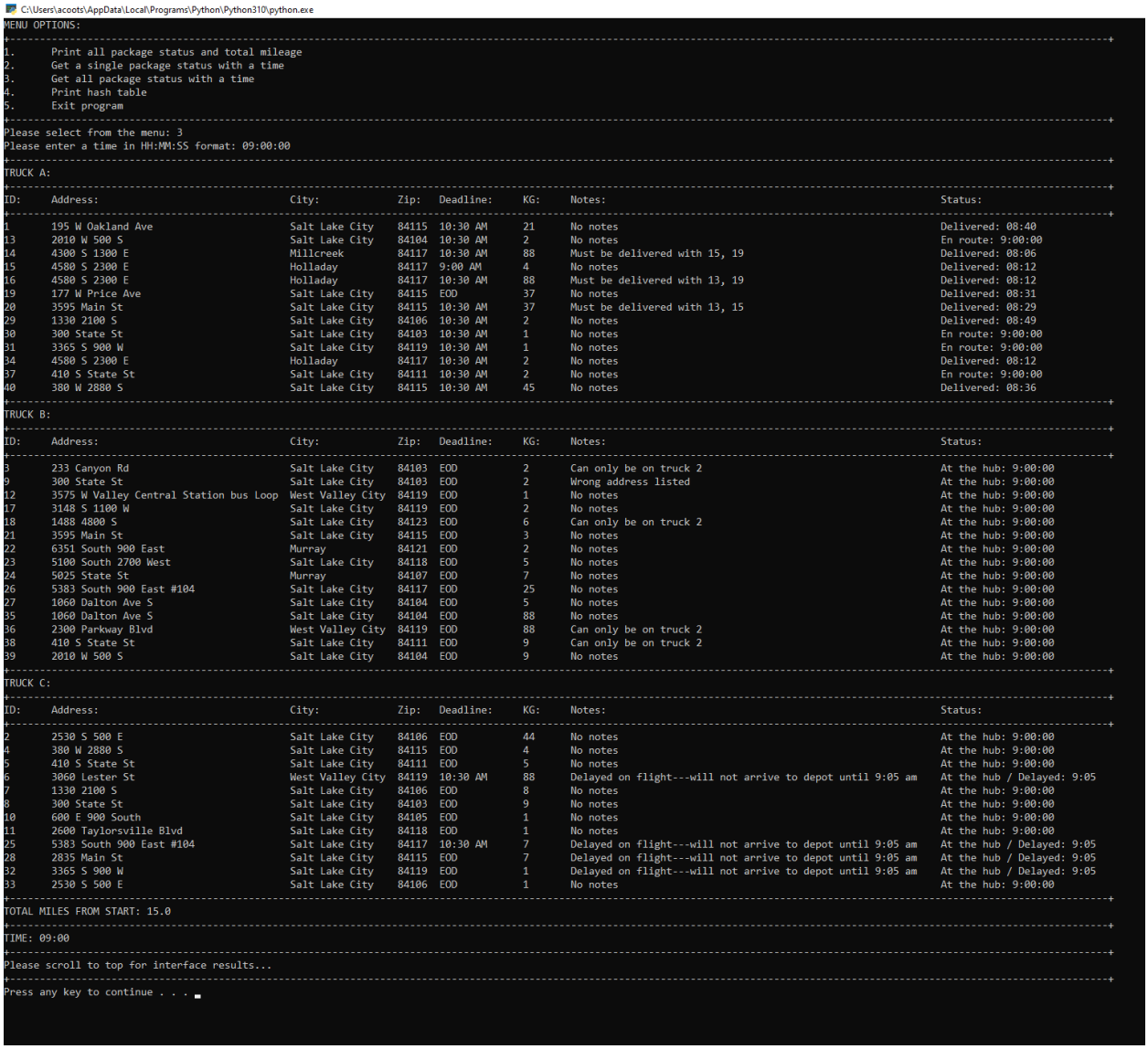
```
# C950.REQUIREMENT.F: DEVELOP A LOOKUP FUNCTION... #
    # Find function returns the key-value pair given the key.
    def hashFind(self, key):
        # key_index represents the index value of hash table for respective key. The same as hashAdd key_index.
        key_index = self.hashFunction(key)
        # If the cell of map at location key_index is not empty (of datatype None).
        if self.map[key_index] is not None:
            # For new array variable 'pair' for comparison of list values in object hash map at position key_index.
            for pair in self.map[key_index]:
                # If the value of array pair[0] equates to input key for position.
                if pair[0] == key:
                    # Return the associated value for key-value pair.
                    return pair[1]
        return None
```

G. Interface.

The following is separated into 3 parts, G1, G2 and G3.

- G1 represents the status of all packages at 09:00:00 a.m.
- G2 represents the status of all packages at 10:00:00 a.m.
- G3 represents the status of all packages at 13:00:00 p.m.

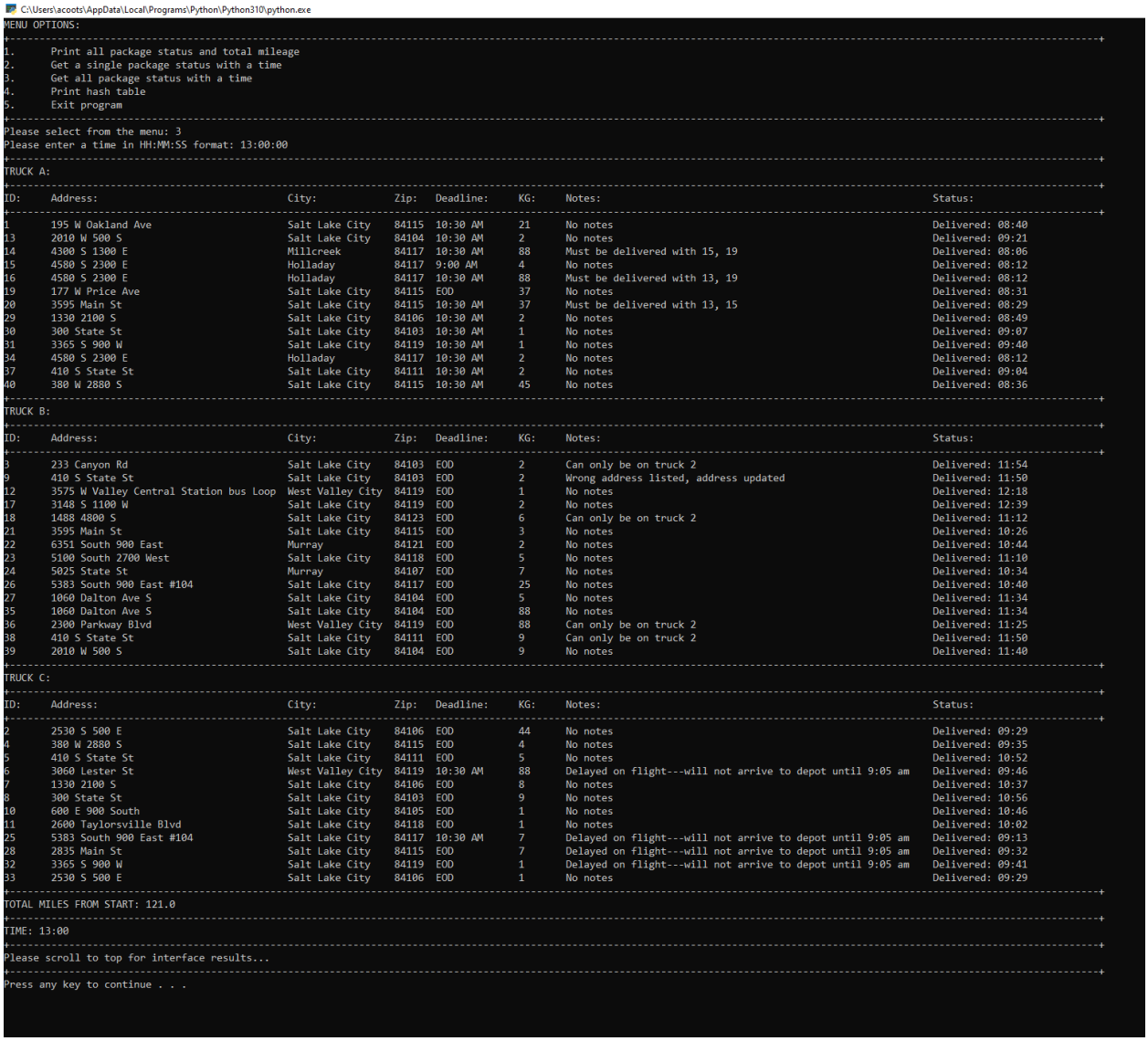
G1. First Status Check.



G2. Second Status Check.

```
C:\Users\acoots\AppData\Local\Programs\Python\Python310\python.exe
MENU OPTIONS:
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
1.      Print all package status and total mileage
2.      Get a single package status with a time
3.      Get all package status with a time
4.      Print hash table
5.      Exit program
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
Please select from the menu: 3
Please enter a time in HH:MM:SS format: 10:00:00
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
TRUCK A:
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
ID:     Address:                                City:            Zip:   Deadline:     KG:     Notes:                                                          Status:
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
1       195 W Oakland Ave                       Salt Lake City   84115  10:30 AM      21      No notes                                                        Delivered: 08:40
13      2010 W 500 S                            Salt Lake City   84104  10:30 AM      2       No notes                                                        Delivered: 09:21
14      4300 S 1300 E                           Millcreek        84117  10:30 AM      88      Must be delivered with 15, 19                                   Delivered: 08:06
15      4580 S 2300 E                           Holladay         84117  9:00 AM       4       No notes                                                        Delivered: 08:12
16      4580 S 2300 E                           Holladay         84117  10:30 AM      88      Must be delivered with 13, 19                                   Delivered: 08:12
19      177 W Price Ave                         Salt Lake City   84115  EOD           37      No notes                                                        Delivered: 08:31
20      3595 Main St                            Salt Lake City   84115  10:30 AM      37      Must be delivered with 13, 15                                   Delivered: 08:29
29      1330 2100 S                             Salt Lake City   84106  10:30 AM      2       No notes                                                        Delivered: 08:49
30      300 State St                            Salt Lake City   84103  10:30 AM      1       No notes                                                        Delivered: 09:07
31      3365 S 900 W                            Salt Lake City   84119  10:30 AM      1       No notes                                                        Delivered: 09:40
34      4580 S 2300 E                           Holladay         84117  10:30 AM      2       No notes                                                        Delivered: 08:12
37      410 S State St                          Salt Lake City   84111  10:30 AM      2       No notes                                                        Delivered: 09:04
40      380 W 2880 S                            Salt Lake City   84115  10:30 AM      45      No notes                                                        Delivered: 08:36
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
TRUCK B:
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
ID:     Address:                                City:            Zip:   Deadline:     KG:     Notes:                                                          Status:
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
3       233 Canyon Rd                           Salt Lake City   84103  EOD           2       Can only be on truck 2                                          At the hub: 10:00:00
9       300 State St                            Salt Lake City   84103  EOD           2       Wrong address listed                                            At the hub: 10:00:00
12      3575 W Valley Central Station bus Loop  West Valley City  84119  EOD           1       No notes                                                        At the hub: 10:00:00
17      3148 S 1100 W                           Salt Lake City   84119  EOD           2       No notes                                                        At the hub: 10:00:00
18      1488 4800 S                             Salt Lake City   84123  EOD           6       Can only be on truck 2                                          At the hub: 10:00:00
21      3595 Main St                            Salt Lake City   84115  EOD           3       No notes                                                        At the hub: 10:00:00
22      6351 South 900 East                     Murray           84121  EOD           2       No notes                                                        At the hub: 10:00:00
23      5100 South 2700 West                    Salt Lake City   84118  EOD           5       No notes                                                        At the hub: 10:00:00
24      5025 State St                           Murray           84107  EOD           7       No notes                                                        At the hub: 10:00:00
26      5383 South 900 East #104                Salt Lake City   84117  EOD           25      No notes                                                        At the hub: 10:00:00
27      1060 Dalton Ave S                       Salt Lake City   84104  EOD           5       No notes                                                        At the hub: 10:00:00
35      1060 Dalton Ave S                       Salt Lake City   84104  EOD           88      No notes                                                        At the hub: 10:00:00
36      2300 Parkway Blvd                       West Valley City 84119  EOD           88      Can only be on truck 2                                          At the hub: 10:00:00
38      410 S State St                          Salt Lake City   84111  EOD           9       Can only be on truck 2                                          At the hub: 10:00:00
39      2010 W 500 S                            Salt Lake City   84104  EOD           9       No notes                                                        At the hub: 10:00:00
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
TRUCK C:
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
ID:     Address:                                City:            Zip:   Deadline:     KG:     Notes:                                                          Status:
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
2       2530 S 500 E                            Salt Lake City   84106  EOD           44      No notes                                                        Delivered: 09:29
4       380 W 2880 S                            Salt Lake City   84115  EOD           4       No notes                                                        Delivered: 09:35
5       410 S State St                          Salt Lake City   84111  EOD           5       No notes                                                        En route: 10:00:00
6       3060 Lester St                          West Valley City 84119  10:30 AM      88      Delayed on flight---will not arrive to depot until 9:05 am      Delivered: 09:46
7       1330 2100 S                             Salt Lake City   84106  EOD           8       No notes                                                        En route: 10:00:00
8       300 State St                            Salt Lake City   84103  EOD           9       No notes                                                        En route: 10:00:00
10      600 E 900 South                         Salt Lake City   84105  EOD           1       No notes                                                        En route: 10:00:00
11      2600 Taylorsville Blvd                  Salt Lake City   84118  EOD           1       No notes                                                        En route: 10:00:00
25      5383 South 900 East #104                Salt Lake City   84117  10:30 AM      7       Delayed on flight---will not arrive to depot until 9:05 am      Delivered: 09:13
28      2835 Main St                            Salt Lake City   84115  EOD           7       Delayed on flight---will not arrive to depot until 9:05 am      Delivered: 09:32
32      3365 S 900 W                            Salt Lake City   84119  EOD           1       Delayed on flight---will not arrive to depot until 9:05 am      Delivered: 09:41
33      2530 S 500 E                            Salt Lake City   84106  EOD           1       No notes                                                        Delivered: 09:29
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
TOTAL MILES FROM START: 46.0
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
TIME: 10:00
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
Please scroll to top for interface results...
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
Press any key to continue . . .
```

G3. Third Status Check.



```
C:\Users\acoots\AppData\Local\Programs\Python\Python310\python.exe
MENU OPTIONS:
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
1.      Print all package status and total mileage
2.      Get a single package status with a time
3.      Get all package status with a time
4.      Print hash table
5.      Exit program
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
Please select from the menu: 3
Please enter a time in HH:MM:SS format: 13:00:00
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
TRUCK A:
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
ID:     Address:                                City:            Zip:   Deadline:     KG:     Notes:                                                          Status:
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
1       195 W Oakland Ave                       Salt Lake City   84115  10:30 AM      21      No notes                                                        Delivered: 08:40
13      2010 W 500 S                            Salt Lake City   84104  10:30 AM      2       No notes                                                        Delivered: 09:21
14      4300 S 1300 E                           Millcreek        84117  10:30 AM      88      Must be delivered with 15, 19                                   Delivered: 08:06
15      4580 S 2300 E                           Holladay         84117  9:00 AM       4       No notes                                                        Delivered: 08:12
16      4580 S 2300 E                           Holladay         84117  10:30 AM      88      Must be delivered with 13, 19                                   Delivered: 08:12
19      177 W Price Ave                         Salt Lake City   84115  EOD           37      No notes                                                        Delivered: 08:31
20      3595 Main St                            Salt Lake City   84115  10:30 AM      37      Must be delivered with 13, 15                                   Delivered: 08:29
29      1330 2100 S                             Salt Lake City   84106  10:30 AM      2       No notes                                                        Delivered: 08:49
30      300 State St                            Salt Lake City   84103  10:30 AM      1       No notes                                                        Delivered: 09:07
31      3365 S 900 W                            Salt Lake City   84119  10:30 AM      1       No notes                                                        Delivered: 09:40
34      4580 S 2300 E                           Holladay         84117  10:30 AM      2       No notes                                                        Delivered: 08:12
37      410 S State St                          Salt Lake City   84111  10:30 AM      2       No notes                                                        Delivered: 09:04
40      380 W 2880 S                            Salt Lake City   84115  10:30 AM      45      No notes                                                        Delivered: 08:36
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
TRUCK B:
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
ID:     Address:                                City:            Zip:   Deadline:     KG:     Notes:                                                          Status:
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
3       233 Canyon Rd                           Salt Lake City   84103  EOD           2       Can only be on truck 2                                          Delivered: 11:54
9       410 S State St                          Salt Lake City   84103  EOD           2       Wrong address listed, address updated                           Delivered: 11:50
12      3575 W Valley Central Station bus Loop  West Valley City  84119  EOD           1       No notes                                                        Delivered: 12:18
17      3148 S 1100 W                           Salt Lake City   84119  EOD           2       No notes                                                        Delivered: 12:39
18      1488 4800 S                             Salt Lake City   84123  EOD           6       Can only be on truck 2                                          Delivered: 11:12
21      3595 Main St                            Salt Lake City   84115  EOD           3       No notes                                                        Delivered: 10:26
22      6351 South 900 East                     Murray           84121  EOD           2       No notes                                                        Delivered: 10:44
23      5100 South 2700 West                    Salt Lake City   84118  EOD           5       No notes                                                        Delivered: 11:10
24      5025 State St                           Murray           84107  EOD           7       No notes                                                        Delivered: 10:34
26      5383 South 900 East #104                Salt Lake City   84117  EOD           25      No notes                                                        Delivered: 10:40
27      1060 Dalton Ave S                       Salt Lake City   84104  EOD           5       No notes                                                        Delivered: 11:34
35      1060 Dalton Ave S                       Salt Lake City   84104  EOD           88      No notes                                                        Delivered: 11:34
36      2300 Parkway Blvd                       West Valley City  84119  EOD           88      Can only be on truck 2                                          Delivered: 11:25
38      410 S State St                          Salt Lake City   84111  EOD           9       Can only be on truck 2                                          Delivered: 11:50
39      2010 W 500 S                            Salt Lake City   84104  EOD           9       No notes                                                        Delivered: 11:40
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
TRUCK C:
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
ID:     Address:                                City:            Zip:   Deadline:     KG:     Notes:                                                          Status:
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
2       2530 S 500 E                            Salt Lake City   84106  EOD           44      No notes                                                        Delivered: 09:29
4       380 W 2880 S                            Salt Lake City   84115  EOD           4       No notes                                                        Delivered: 09:35
5       410 S State St                          Salt Lake City   84111  EOD           5       No notes                                                        Delivered: 10:52
6       3060 Lester St                          West Valley City  84119  10:30 AM      88      Delayed on flight---will not arrive to depot until 9:05 am      Delivered: 09:46
7       1330 2100 S                             Salt Lake City   84106  EOD           8       No notes                                                        Delivered: 10:37
8       300 State St                            Salt Lake City   84103  EOD           9       No notes                                                        Delivered: 10:56
10      600 E 900 South                         Salt Lake City   84105  EOD           1       No notes                                                        Delivered: 10:46
11      2600 Taylorsville Blvd                  Salt Lake City   84118  EOD           1       No notes                                                        Delivered: 10:02
25      5383 South 900 East #104                Salt Lake City   84117  10:30 AM      7       Delayed on flight---will not arrive to depot until 9:05 am      Delivered: 09:13
28      2835 Main St                            Salt Lake City   84115  EOD           7       Delayed on flight---will not arrive to depot until 9:05 am      Delivered: 09:32
32      3365 S 900 W                            Salt Lake City   84119  EOD           1       Delayed on flight---will not arrive to depot until 9:05 am      Delivered: 09:41
33      2530 S 500 E                            Salt Lake City   84106  EOD           1       No notes                                                        Delivered: 09:29
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
TOTAL MILES FROM START: 121.0
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
TIME: 13:00
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
Please scroll to top for interface results...
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
Press any key to continue . . .
```

H. Screenshots of Code Execution.

```
C:\Users\acoots\AppData\Local\Programs\Python\Python310\python.exe
MENU OPTIONS:
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
1.      Print all package status and total mileage
2.      Get a single package status with a time
3.      Get all package status with a time
4.      Print hash table
5.      Exit program
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
Please select from the menu: 1
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
TRUCK A:
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
ID:     Address:                                City:            Zip:   Deadline:     KG:     Notes:                                                          Status:
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
1       195 W Oakland Ave                       Salt Lake City   84115  10:30 AM      21      No notes                                                        Delivered: 08:40
13      2010 W 500 S                            Salt Lake City   84104  10:30 AM      2       No notes                                                        Delivered: 09:21
14      4300 S 1300 E                           Millcreek        84117  10:30 AM      88      Must be delivered with 15, 19                                   Delivered: 08:06
15      4580 S 2300 E                           Holladay         84117  9:00 AM       4       No notes                                                        Delivered: 08:12
16      4580 S 2300 E                           Holladay         84117  10:30 AM      88      Must be delivered with 13, 19                                   Delivered: 08:12
19      177 W Price Ave                         Salt Lake City   84115  EOD           37      No notes                                                        Delivered: 08:31
20      3595 Main St                            Salt Lake City   84115  10:30 AM      37      Must be delivered with 13, 15                                   Delivered: 08:29
29      1330 2100 S                             Salt Lake City   84106  10:30 AM      2       No notes                                                        Delivered: 08:49
30      300 State St                            Salt Lake City   84103  10:30 AM      1       No notes                                                        Delivered: 09:07
31      3365 S 900 W                            Salt Lake City   84119  10:30 AM      1       No notes                                                        Delivered: 09:40
34      4580 S 2300 E                           Holladay         84117  10:30 AM      2       No notes                                                        Delivered: 08:12
37      410 S State St                          Salt Lake City   84111  10:30 AM      2       No notes                                                        Delivered: 09:04
40      380 W 2880 S                            Salt Lake City   84115  10:30 AM      45      No notes                                                        Delivered: 08:36
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
TRUCK B:
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
ID:     Address:                                City:            Zip:   Deadline:     KG:     Notes:                                                          Status:
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
3       233 Canyon Rd                           Salt Lake City   84103  EOD           2       Can only be on truck 2                                          Delivered: 11:54
9       410 S State St                          Salt Lake City   84103  EOD           2       Wrong address listed, address updated                           Delivered: 11:50
12      3575 W Valley Central Station bus Loop  West Valley City  84119  EOD           1       No notes                                                        Delivered: 12:18
17      3148 S 1100 W                           Salt Lake City   84119  EOD           2       No notes                                                        Delivered: 12:39
18      1488 4800 S                             Salt Lake City   84123  EOD           6       Can only be on truck 2                                          Delivered: 11:12
21      3595 Main St                            Salt Lake City   84115  EOD           3       No notes                                                        Delivered: 10:26
22      6351 South 900 East                     Murray           84121  EOD           2       No notes                                                        Delivered: 10:44
23      5100 South 2700 West                    Salt Lake City   84118  EOD           5       No notes                                                        Delivered: 11:10
24      5025 State St                           Murray           84107  EOD           7       No notes                                                        Delivered: 10:34
26      5383 South 900 East #104                Salt Lake City   84117  EOD           25      No notes                                                        Delivered: 10:40
27      1060 Dalton Ave S                       Salt Lake City   84104  EOD           5       No notes                                                        Delivered: 11:34
35      1060 Dalton Ave S                       Salt Lake City   84104  EOD           88      No notes                                                        Delivered: 11:34
36      2300 Parkway Blvd                       West Valley City 84119  EOD           88      Can only be on truck 2                                          Delivered: 11:25
38      410 S State St                          Salt Lake City   84111  EOD           9       Can only be on truck 2                                          Delivered: 11:50
39      2010 W 500 S                            Salt Lake City   84104  EOD           9       No notes                                                        Delivered: 11:40
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
TRUCK C:
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
ID:     Address:                                City:            Zip:   Deadline:     KG:     Notes:                                                          Status:
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
2       2530 S 500 E                            Salt Lake City   84106  EOD           44      No notes                                                        Delivered: 09:29
4       380 W 2880 S                            Salt Lake City   84115  EOD           4       No notes                                                        Delivered: 09:35
5       410 S State St                          Salt Lake City   84111  EOD           5       No notes                                                        Delivered: 10:52
6       3060 Lester St                          West Valley City 84119  10:30 AM      88      Delayed on flight---will not arrive to depot until 9:05 am      Delivered: 09:46
7       1330 2100 S                             Salt Lake City   84106  EOD           8       No notes                                                        Delivered: 10:37
8       300 State St                            Salt Lake City   84103  EOD           9       No notes                                                        Delivered: 10:56
10      600 E 900 South                         Salt Lake City   84105  EOD           1       No notes                                                        Delivered: 10:46
11      2600 Taylorsville Blvd                  Salt Lake City   84118  EOD           1       No notes                                                        Delivered: 10:02
25      5383 South 900 East #104                Salt Lake City   84117  10:30 AM      7       Delayed on flight---will not arrive to depot until 9:05 am      Delivered: 09:13
28      2835 Main St                            Salt Lake City   84115  EOD           7       Delayed on flight---will not arrive to depot until 9:05 am      Delivered: 09:32
32      3365 S 900 W                            Salt Lake City   84119  EOD           1       Delayed on flight---will not arrive to depot until 9:05 am      Delivered: 09:41
33      2530 S 500 E                            Salt Lake City   84106  EOD           1       No notes                                                        Delivered: 09:29
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
TOTAL MILES: 121.0
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
Please scroll to top for interface results...
+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
Press any key to continue . . .
```

I1. Strengths of Chosen Algorithm.

There are strong arguments for use of a ‘Nearest Neighbor’ algorithm. For example, one, time and two, space. As mentioned prior, the general draws for concern really start at the human level far before the computer level (see B4). This is an efficient algorithm such that it is of linear time-complexity with the number of packages to be delivered, considering the model is to calculate the shortest distance to travel after an iteration to reduce distance traveled upon arrival at any given destination. It is efficient similarly in terms of space, considering a multitude of packages may exist at one time along with address locations, every address in the world is stored via GPS systems, so in the use of one state, Utah, the argument of massive space consumption loses its validity. Also considering that CSV files would need a generous amount of data to be considered ‘too much’, which then refers back to the human side of our mail delivery problem and that so many packages can be delivered in a limited time frame. Space is also parallel in big O notation (see B3), it solely relies on the datapoints it is given. Therefore, one, time, and two, space are two considerable strengths of the ‘Nearest Neighbor’ algorithm.

I2. Verification of Algorithm.

The scenario requirements of the solution are that:

- There must be an algorithm in place to deliver 40 packages.
- The packages must be delivered on time.
- The total distance combined must be below 140 miles total.

As seen within program results of all trucks by EOD (End of Day), the trucks combine to a total mileage of 121.0 miles even (see H). For each individual package that has a deadline, should the time be given in selection 3 of the CLI interface, it is ensured that all packages meet their delivery deadline requirements, also that, all 40 packages are delivered. Along with this, the algorithm meets the special requirements that package #9, at time 10:19:59 has this initially incorrect address, and at 10:20:00, the address is updated.

10:19:59 -

TRUCK B:

ID:	Address:	City:	Zip:	Deadline:	KG:	Notes:	Status:
3	233 Canyon Rd	Salt Lake City	84103	EOD	2	Can only be on truck 2	At the hub: 10:19:59
9	300 State St	Salt Lake City	84103	EOD	2	Wrong address listed	At the hub: 10:19:59
12	3575 W Valley Central Station bus Loop	West Valley City	84119	EOD	1	No notes	At the hub: 10:19:59
17	3148 S 1100 W	Salt Lake City	84119	EOD	2	No notes	At the hub: 10:19:59
18	1488 4800 S	Salt Lake City	84123	EOD	6	Can only be on truck 2	At the hub: 10:19:59
21	3595 Main St	Salt Lake City	84115	EOD	3	No notes	At the hub: 10:19:59
22	6351 South 900 East	Murray	84121	EOD	2	No notes	At the hub: 10:19:59
23	5100 South 2700 West	Salt Lake City	84118	EOD	5	No notes	At the hub: 10:19:59
24	5025 State St	Murray	84107	EOD	7	No notes	At the hub: 10:19:59
26	5383 South 900 East #104	Salt Lake City	84117	EOD	25	No notes	At the hub: 10:19:59
27	1060 Dalton Ave S	Salt Lake City	84104	EOD	5	No notes	At the hub: 10:19:59
35	1060 Dalton Ave S	Salt Lake City	84104	EOD	88	No notes	At the hub: 10:19:59
36	2300 Parkway Blvd	West Valley City	84119	EOD	88	Can only be on truck 2	At the hub: 10:19:59
38	410 S State St	Salt Lake City	84111	EOD	9	Can only be on truck 2	At the hub: 10:19:59
39	2010 W 500 S	Salt Lake City	84104	EOD	9	No notes	At the hub: 10:19:59

10:20:00 –

TRUCK B:

ID:	Address:	City:	Zip:	Deadline:	KG:	Notes:	Status:
3	233 Canyon Rd	Salt Lake City	84103	EOD	2	Can only be on truck 2	At the hub: 10:20:00
9	410 S State St	Salt Lake City	84103	EOD	2	Wrong address listed, address updated	At the hub: 10:20:00
12	3575 W Valley Central Station bus Loop	West Valley City	84119	EOD	1	No notes	At the hub: 10:20:00
17	3148 S 1100 W	Salt Lake City	84119	EOD	2	No notes	At the hub: 10:20:00
18	1488 4800 S	Salt Lake City	84123	EOD	6	Can only be on truck 2	At the hub: 10:20:00
21	3595 Main St	Salt Lake City	84115	EOD	3	No notes	At the hub: 10:20:00
22	6351 South 900 East	Murray	84121	EOD	2	No notes	At the hub: 10:20:00
23	5100 South 2700 West	Salt Lake City	84118	EOD	5	No notes	At the hub: 10:20:00
24	5025 State St	Murray	84107	EOD	7	No notes	At the hub: 10:20:00
26	5383 South 900 East #104	Salt Lake City	84117	EOD	25	No notes	At the hub: 10:20:00
27	1060 Dalton Ave S	Salt Lake City	84104	EOD	5	No notes	At the hub: 10:20:00
35	1060 Dalton Ave S	Salt Lake City	84104	EOD	88	No notes	At the hub: 10:20:00
36	2300 Parkway Blvd	West Valley City	84119	EOD	88	Can only be on truck 2	At the hub: 10:20:00
38	410 S State St	Salt Lake City	84111	EOD	9	Can only be on truck 2	At the hub: 10:20:00
39	2010 W 500 S	Salt Lake City	84104	EOD	9	No notes	At the hub: 10:20:00

I3. Other Possible Algorithms.

The complications / constraints of mail delivery may limit what algorithms may be used when designing a program. Two other algorithms that would meet requirements would be the ‘Greedy’ and ‘Held-Karp’ algorithms, the ‘Brute Force’ algorithm.

The ‘Greedy’ algorithm, is an algorithm that could be described as making use of the best choice currently available, meaning, the algorithm is constructed to follow the ‘greedy’ path given the rules of what a ‘greedy’ choice is. This algorithm at the time-complexity level is similar as it is of O(N) or occasionally O(NlogN) notation, and O(1) space-complexity. The knapsack problem is a great interpretation of how the algorithm comes to be. The largest item is taken that will meet constraints then the rest is filled similarly. Execution of this algorithm would work in a way such that:

1. The truck is also manually loaded.
2. The largest distance away from the hub is taken first and the truck works its way back to the hub, taking the biggest distances out of the equation leaving only smaller distances to travel as the execution continues.

Like time constraints in the ‘Nearest Neighbor’ algorithm, a separate idea to consider when loading the trucks in the ‘Greedy’ algorithm is meeting a limited distance so that, not too many times is a large path traveled, summing up to over 140 miles. This type of idea can be seen in the traditional Knapsack problem where the knapsack may only carry a certain weight for instance.

The ‘Brute Force’ algorithm is an algorithm that could be described by finding all possible solutions and selecting the minimum of those solutions. This algorithm is of similar time and space-complexity for this problem, O(N). Execution of this algorithm would work in a way such that:

1. The truck, similarly, to ‘Nearest Neighbor’ and ‘Greedy’ algorithms are manually loaded.
2. Mathematical analysis is done on each possible path for the packages given and their respective distances from each other to find which order should the packages be delivered. Say there are three locations 1, 2 and 3, that 1 is the start and end location. The possibilities are 1-2-3-1 and 1-3-2-1, whichever of those two paths meet the requirements, miles under 140 and is the shorter path, will be chosen.

I3A. Algorithm Differences.

These algorithms are similar in terms of time and space-complexity. However, the differences stem from the structural representation of which package is to be delivered next. The ‘Nearest Neighbor’ algorithm, as it delivers packages reiterates the list to consider that the next best delivery option must be decided upon package delivery. The ‘Greedy’ algorithm is different in the sense that due to its nature it limits itself by taking the ‘greediest’ option first, not the nearest, and then basing the rest of its path from what resource (time, miles) limits it while still being as greedy as it can to effectively take out the taller distances for itself and future trucks whereas the ‘Nearest Neighbor’ algorithm finds the closest only and then goes back to the mail delivery hub. The ‘Brute Force’ algorithm finds the best path based on criteria (minimum time or miles) after calculating all paths, opposite of nearest or greediest, and does no further calculation on the path, which is opposite of the ‘Nearest Neighbor’ algorithm.

J. Different Approach.

To do this project once over again I would start with my main function first. In a sense I ‘reverse engineered’ this project, starting with the hash table, then the packages and truck classes, and then the logic to present the code via CLI. The project does put a good amount of emphasis on knowing your data structure, at least the hash table, which ultimately put my focus there instead of developing the program itself which I believe would’ve helped me make a more effective and practical program starting from the interface down to the classes.

K1. Verification of Data Structure.

The use of the hash table gives creation of the algorithm, the ‘Nearest Neighbor’ algorithm needs two things to create its own foundation, those being a key, representing a value. The algorithm makes use of computing the distance between two packages, but only knows respective information on the packages are as it references the hash table with the key needed to use the right data as it is attempting to perform computations.

K1A. Efficiency.

The time needed using a hash table is constant, meaning of big O notation, O(1), considering that all that is needed to find the data of a key-value pair, is the key itself, returning the key-value for that pair. This does only the search of the cell the respective key is in post modulo operation.

K1B. Overhead.

The space needed to manage this program increases and decreases linearly with input. If the assumption that there were no collisions, the hash table would need to increase in size considering what a collision is at the surface level. It would increase the size of the table 1:1 with the amount of data points, this is the case regardless of the presence of collisions though collisions must be properly handled though ideas the likes of database degrade become inevitable with larger data sets.

K1C. Implications.

The number of trucks or the number of cities does not affect the look up time, the truck is external data of the hash table, it is not part of the time or space makeup of the table. Alongside the city of a package, the city does not represent either the key or the value, specifically the whole package itself represents the value of a key-value pair. The concept of more cities only adds more packages if the cities are near and relative to the location of the program. Should more cities, within relative area, be added the biggest concern is the addition of more packages which is handled accordingly with size of the hash table.

K2. Other Data Structures.

Two of many data structures of relevant comparison could be a ‘Binary Search’ tree and a ‘Dictionary’. For comparison purposes each are their own bullet points:

- ‘Binary Search’ trees are parent-child trees where the left subtree contains values less than the nodes key and the right subtree contains values higher than the nodes key. Example being having ‘5’ be your parent-node key then, for example, ‘2’ would be a key for the child-node on the left and the key of ‘7’ would be on the right. ‘Binary Search’ trees would initially find themselves with a middle value, perhaps key of 20, and would contain keys from 1 to 19 on the left and 21 through 39 to the right. The structural differences between these and hash tables are apparent though. This approach would be a bit more complex in terms of time, searching left and right downwards of the root until finding the key to return the key-value pair, though considering the common pattern of the tree, the time-complexity remains linear along with space.

- ‘Dictionaries’ are like hash tables though more limiting than hash tables. Hash tables are known to be non-generic data structures while dictionaries are more generic. Fundamentally built using key-value storage along with buckets, dictionaries, store specific key-value pairs of specific data types. The approach for implementing dictionaries for a project of this scope would either involve multiple dictionaries involving different data types per package data field on matching keys or making the key-value pair such that the key be the ID and the value be one string, later divided up for algorithmic use by a complex substring loop that would also do variable conversions.

L. Sources – Works Cited.

Oggi AI - Artificial Intelligence Today. (2016). *Python: Creating a HASHMAP using Lists*. *YouTube*. YouTube. Retrieved June 4, 2023, from https://www.youtube.com/watch?v=9HFbhPscPU0&t=452s&ab_channel=OggiAI-ArtificialIntelligenceToday.

Western Governors University. (n.d.-a). *C950 WGUPS Project Implementation Steps - Example*. C950 WGUPS Project Implementation Steps. https://srm--c.vf.force.com/apex/coursearticle?Id=kA03x000001DbBGCA0

Western Governors University. (n.d.-b). C950_WGUPS_Algorithm_Overview_Template. Millcreek; WGU.

NOTE: Reference may exist in Python scripts.

M. Professional Communication.

Name: Coots, Anthony

Student ID: 010958511

Email: acoots5@wgu.edu