Benjamin Prendergast
ID: 00[redacted]
2023/11/28

C950 Performance Assessment

Task 2: WGUPS Routing Program Implementation

A. Develop a hash table, without using any additional libraries or classes, that has an insertion function that takes the package ID as input and inserts each of the following data components into the hash table:

- delivery address
- delivery deadline
- delivery city
- delivery zip code
- package weight
- delivery status (i.e., at the hub, en route, or delivered), including the delivery time

hashtable.py

```
2 usages
class HashTable:
    # Constructor, defaults to 10 buckets
    # Assigns each bucket with an empty list
    def __init__(self, initial_capacity=10):
        self.table = []
        for i in range(initial_capacity):
            self.table.append([])

    # O(N) worst case if all keys hash to same bucket
    # Accepts the package ID as input and inserts the package object, which contains the data components
    1 usage
    def insert(self, key, item):
        # Determine target bucket
        bucket = int(key) % len(self.table)
        bucket_list = self.table[bucket]

        # Update item if already exists in 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

    # O(N) worst case all keys hashed to same bucket
    6 usages (6 dynamic)
    def search(self, key):
        # Determine target bucket
        bucket = int(key) % len(self.table)
        bucket_list = self.table[bucket]

        for kv in bucket_list:
            if kv[0] == key:
                return kv[1]
        return None

```

B. Develop a look-up function that takes the package ID as input and returns each of the following corresponding data components:

- delivery address
- delivery deadline
- delivery city
- delivery zip code
- package weight
- delivery status (i.e., at the hub, en route, or delivered), including the delivery time

main.py

```
31  def lookup(package_id, package_hash):
32      p = package_hash.search(package_id)
33
34      if p:
35          package_address = p.address()
36          package_deadline = p.deadline()
37          package_city = p.city()
38          package_zip = p.zipcode()
39          package_weight = p.weight()
40          package_status = p.status()
41          package_delivery_truck = p.delivery_truck()
42          package_delivery_time = p.delivery_time()
43
44          return (f'ID: {package_id}, Address: {package_address}, City: {package_city}, Zip: {package_zip}, '
45                  f'Weight: {package_weight}, Deadline: {package_deadline}, Delivery Status: {package_status} '
46                  f'by {package_delivery_truck} at {package_delivery_time}')
47      else:
48          return None
```

C. Write an original program that will deliver all packages and meet all requirements using the attached supporting documents "Salt Lake City Downtown Map," "WGUPS Distance Table," and "WGUPS Package File."

Program written in PyCharm and submitted with assignment.

C1. Create an identifying comment within the first line of a file named "main.py" that includes your student ID.

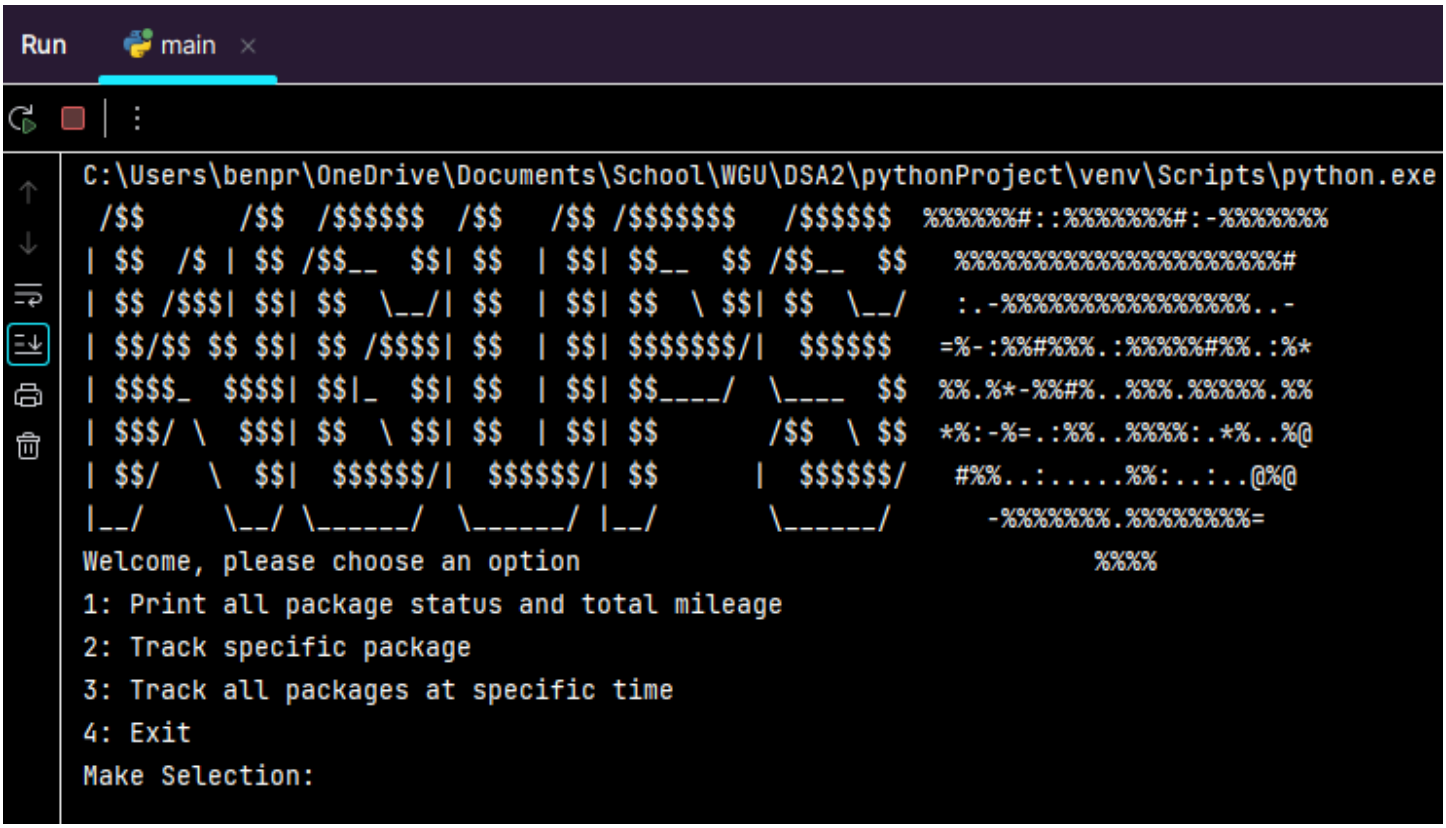
main.py

```
1  """
2  Benjamin Prendergast, Student ID: 00[redacted]
3  C950 Performance Assessment
4  Task 2: WGUPS Routing Program Implementation
5  """
```

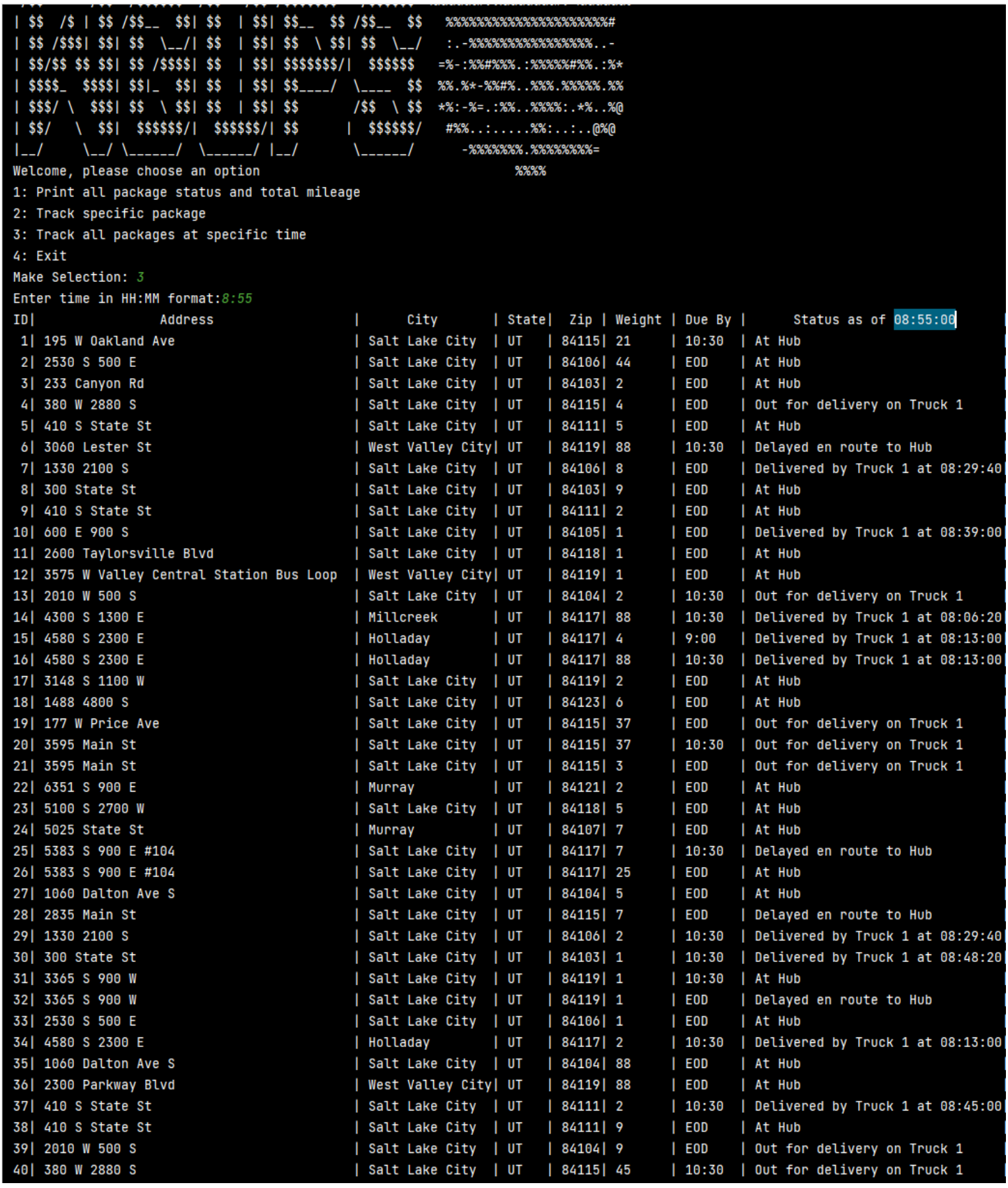
C2. Include comments in your code to explain both the process and the flow of the program.

Program code includes comments describing both process and flow.

D. Provide an intuitive interface for the user to view the delivery status (including the delivery time) of any package at any time and the total mileage traveled by all trucks.



D1. Provide screenshots to show the status of all packages loaded onto each truck at a time between 8:35 a.m. and 9:25 a.m.



```
Welcome, please choose an option
1: Print all package status and total mileage
2: Track specific package
3: Track all packages at specific time
4: Exit
Make Selection: 3
Enter time in HH:MM format:8:55
```

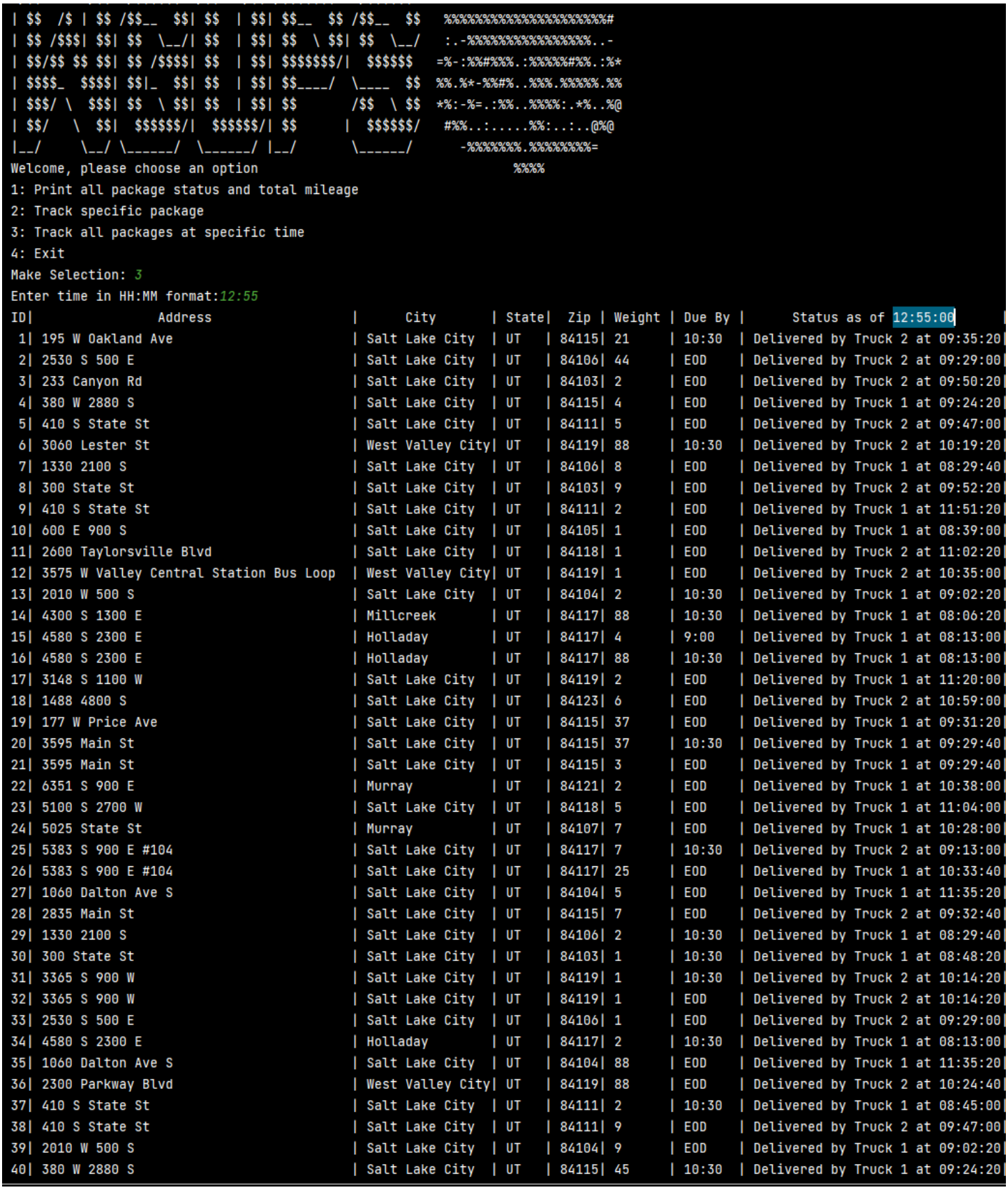
ID	Address	City	State	Zip	Weight	Due By	Status as of 08:55:00
1	195 W Oakland Ave	Salt Lake City	UT	84115	21	10:30	At Hub
2	2530 S 500 E	Salt Lake City	UT	84106	44	EOD	At Hub
3	233 Canyon Rd	Salt Lake City	UT	84103	2	EOD	At Hub
4	380 W 2880 S	Salt Lake City	UT	84115	4	EOD	Out for delivery on Truck 1
5	410 S State St	Salt Lake City	UT	84111	5	EOD	At Hub
6	3060 Lester St	West Valley City	UT	84119	88	10:30	Delayed en route to Hub
7	1330 2100 S	Salt Lake City	UT	84106	8	EOD	Delivered by Truck 1 at 08:29:40
8	300 State St	Salt Lake City	UT	84103	9	EOD	At Hub
9	410 S State St	Salt Lake City	UT	84111	2	EOD	At Hub
10	600 E 900 S	Salt Lake City	UT	84105	1	EOD	Delivered by Truck 1 at 08:39:00
11	2600 Taylorsville Blvd	Salt Lake City	UT	84118	1	EOD	At Hub
12	3575 W Valley Central Station Bus Loop	West Valley City	UT	84119	1	EOD	At Hub
13	2010 W 500 S	Salt Lake City	UT	84104	2	10:30	Out for delivery on Truck 1
14	4300 S 1300 E	Millcreek	UT	84117	88	10:30	Delivered by Truck 1 at 08:06:20
15	4580 S 2300 E	Holladay	UT	84117	4	9:00	Delivered by Truck 1 at 08:13:00
16	4580 S 2300 E	Holladay	UT	84117	88	10:30	Delivered by Truck 1 at 08:13:00
17	3148 S 1100 W	Salt Lake City	UT	84119	2	EOD	At Hub
18	1488 4800 S	Salt Lake City	UT	84123	6	EOD	At Hub
19	177 W Price Ave	Salt Lake City	UT	84115	37	EOD	Out for delivery on Truck 1
20	3595 Main St	Salt Lake City	UT	84115	37	10:30	Out for delivery on Truck 1
21	3595 Main St	Salt Lake City	UT	84115	3	EOD	Out for delivery on Truck 1
22	6351 S 900 E	Murray	UT	84121	2	EOD	At Hub
23	5100 S 2700 W	Salt Lake City	UT	84118	5	EOD	At Hub
24	5025 State St	Murray	UT	84107	7	EOD	At Hub
25	5383 S 900 E #104	Salt Lake City	UT	84117	7	10:30	Delayed en route to Hub
26	5383 S 900 E #104	Salt Lake City	UT	84117	25	EOD	At Hub
27	1060 Dalton Ave S	Salt Lake City	UT	84104	5	EOD	At Hub
28	2835 Main St	Salt Lake City	UT	84115	7	EOD	Delayed en route to Hub
29	1330 2100 S	Salt Lake City	UT	84106	2	10:30	Delivered by Truck 1 at 08:29:40
30	300 State St	Salt Lake City	UT	84103	1	10:30	Delivered by Truck 1 at 08:48:20
31	3365 S 900 W	Salt Lake City	UT	84119	1	10:30	At Hub
32	3365 S 900 W	Salt Lake City	UT	84119	1	EOD	Delayed en route to Hub
33	2530 S 500 E	Salt Lake City	UT	84106	1	EOD	At Hub
34	4580 S 2300 E	Holladay	UT	84117	2	10:30	Delivered by Truck 1 at 08:13:00
35	1060 Dalton Ave S	Salt Lake City	UT	84104	88	EOD	At Hub
36	2300 Parkway Blvd	West Valley City	UT	84119	88	EOD	At Hub
37	410 S State St	Salt Lake City	UT	84111	2	10:30	Delivered by Truck 1 at 08:45:00
38	410 S State St	Salt Lake City	UT	84111	9	EOD	At Hub
39	2010 W 500 S	Salt Lake City	UT	84104	9	EOD	Out for delivery on Truck 1
40	380 W 2880 S	Salt Lake City	UT	84115	45	10:30	Out for delivery on Truck 1

D2. Provide screenshots to show the status of all packages loaded onto each truck at a time between 9:35 a.m. and 10:25 a.m.

```
Welcome, please choose an option
1: Print all package status and total mileage
2: Track specific package
3: Track all packages at specific time
4: Exit
Make Selection: 3
Enter time in HH:MM format:9:55
```

ID	Address	City	State	Zip	Weight	Due By	Status as of 09:55:00
1	195 W Oakland Ave	Salt Lake City	UT	84115	21	10:30	Delivered by Truck 2 at 09:35:20
2	2530 S 500 E	Salt Lake City	UT	84106	44	EOD	Delivered by Truck 2 at 09:29:00
3	233 Canyon Rd	Salt Lake City	UT	84103	2	EOD	Delivered by Truck 2 at 09:50:20
4	380 W 2880 S	Salt Lake City	UT	84115	4	EOD	Delivered by Truck 1 at 09:24:20
5	410 S State St	Salt Lake City	UT	84111	5	EOD	Delivered by Truck 2 at 09:47:00
6	3060 Lester St	West Valley City	UT	84119	88	10:30	Out for delivery on Truck 2
7	1330 2100 S	Salt Lake City	UT	84106	8	EOD	Delivered by Truck 1 at 08:29:40
8	300 State St	Salt Lake City	UT	84103	9	EOD	Delivered by Truck 2 at 09:52:20
9	410 S State St	Salt Lake City	UT	84111	2	EOD	At Hub
10	600 E 900 S	Salt Lake City	UT	84105	1	EOD	Delivered by Truck 1 at 08:39:00
11	2600 Taylorsville Blvd	Salt Lake City	UT	84118	1	EOD	Out for delivery on Truck 2
12	3575 W Valley Central Station Bus Loop	West Valley City	UT	84119	1	EOD	Out for delivery on Truck 2
13	2010 W 500 S	Salt Lake City	UT	84104	2	10:30	Delivered by Truck 1 at 09:02:20
14	4300 S 1300 E	Millcreek	UT	84117	88	10:30	Delivered by Truck 1 at 08:06:20
15	4580 S 2300 E	Holladay	UT	84117	4	9:00	Delivered by Truck 1 at 08:13:00
16	4580 S 2300 E	Holladay	UT	84117	88	10:30	Delivered by Truck 1 at 08:13:00
17	3148 S 1100 W	Salt Lake City	UT	84119	2	EOD	At Hub
18	1488 4800 S	Salt Lake City	UT	84123	6	EOD	Out for delivery on Truck 2
19	177 W Price Ave	Salt Lake City	UT	84115	37	EOD	Delivered by Truck 1 at 09:31:20
20	3595 Main St	Salt Lake City	UT	84115	37	10:30	Delivered by Truck 1 at 09:29:40
21	3595 Main St	Salt Lake City	UT	84115	3	EOD	Delivered by Truck 1 at 09:29:40
22	6351 S 900 E	Murray	UT	84121	2	EOD	At Hub
23	5100 S 2700 W	Salt Lake City	UT	84118	5	EOD	At Hub
24	5025 State St	Murray	UT	84107	7	EOD	At Hub
25	5383 S 900 E #104	Salt Lake City	UT	84117	7	10:30	Delivered by Truck 2 at 09:13:00
26	5383 S 900 E #104	Salt Lake City	UT	84117	25	EOD	At Hub
27	1060 Dalton Ave S	Salt Lake City	UT	84104	5	EOD	At Hub
28	2835 Main St	Salt Lake City	UT	84115	7	EOD	Delivered by Truck 2 at 09:32:40
29	1330 2100 S	Salt Lake City	UT	84106	2	10:30	Delivered by Truck 1 at 08:29:40
30	300 State St	Salt Lake City	UT	84103	1	10:30	Delivered by Truck 1 at 08:48:20
31	3365 S 900 W	Salt Lake City	UT	84119	1	10:30	Out for delivery on Truck 2
32	3365 S 900 W	Salt Lake City	UT	84119	1	EOD	Out for delivery on Truck 2
33	2530 S 500 E	Salt Lake City	UT	84106	1	EOD	Delivered by Truck 2 at 09:29:00
34	4580 S 2300 E	Holladay	UT	84117	2	10:30	Delivered by Truck 1 at 08:13:00
35	1060 Dalton Ave S	Salt Lake City	UT	84104	88	EOD	At Hub
36	2300 Parkway Blvd	West Valley City	UT	84119	88	EOD	Out for delivery on Truck 2
37	410 S State St	Salt Lake City	UT	84111	2	10:30	Delivered by Truck 1 at 08:45:00
38	410 S State St	Salt Lake City	UT	84111	9	EOD	Delivered by Truck 2 at 09:47:00
39	2010 W 500 S	Salt Lake City	UT	84104	9	EOD	Delivered by Truck 1 at 09:02:20
40	380 W 2880 S	Salt Lake City	UT	84115	45	10:30	Delivered by Truck 1 at 09:24:20

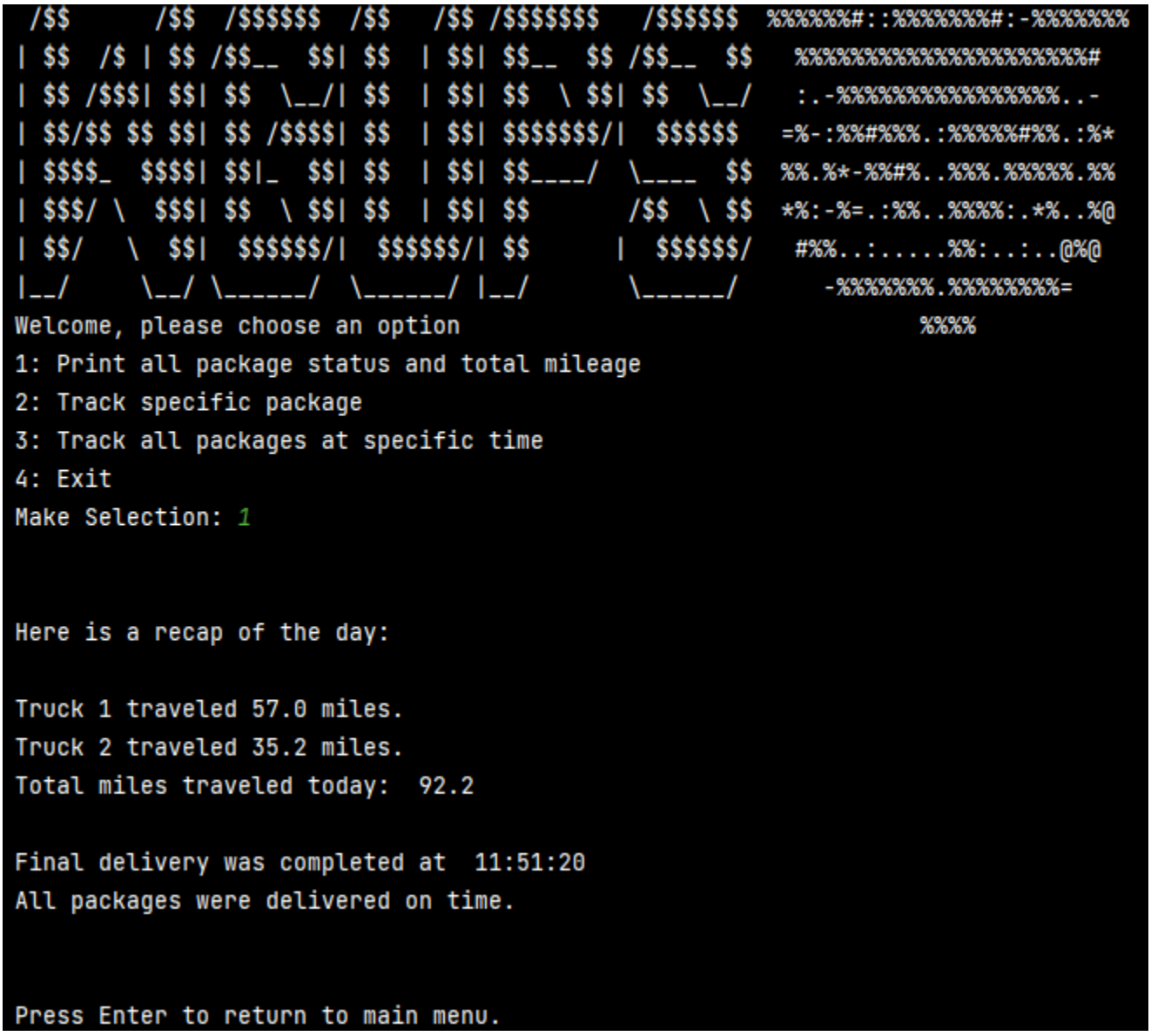
D3. Provide screenshots to show the status of all packages loaded onto each truck at a time between 12:03 p.m. and 1:12 p.m.



```
| $$  /$ | $$ /$$__  $$| $$  | $$| $$__  $$ /$$__  $$
| $$ /$$$| $$| $$  \__/| $$  | $$| $$  \ $$| $$  \__/
| $$/$$ $$ $$| $$ /$$$$| $$  | $$| $$$$$$$/|  $$$$$$
| $$$$_  $$$$| $$|_  $$| $$  | $$| $$____/  \____  $$
| $$$/ \  $$$| $$  \ $$| $$  | $$| $$       /$$  \ $$
| $$/   \  $$|  $$$$$$/|  $$$$$$/| $$      |  $$$$$$/
|__/     \__/ \______/  \______/ |__/       \______/
Welcome, please choose an option
1: Print all package status and total mileage
2: Track specific package
3: Track all packages at specific time
4: Exit
Make Selection: 3
Enter time in HH:MM format:12:55
```

ID	Address	City	State	Zip	Weight	Due By	Status as of 12:55:00
1	195 W Oakland Ave	Salt Lake City	UT	84115	21	10:30	Delivered by Truck 2 at 09:35:20
2	2530 S 500 E	Salt Lake City	UT	84106	44	EOD	Delivered by Truck 2 at 09:29:00
3	233 Canyon Rd	Salt Lake City	UT	84103	2	EOD	Delivered by Truck 2 at 09:50:20
4	380 W 2880 S	Salt Lake City	UT	84115	4	EOD	Delivered by Truck 1 at 09:24:20
5	410 S State St	Salt Lake City	UT	84111	5	EOD	Delivered by Truck 2 at 09:47:00
6	3060 Lester St	West Valley City	UT	84119	88	10:30	Delivered by Truck 2 at 10:19:20
7	1330 2100 S	Salt Lake City	UT	84106	8	EOD	Delivered by Truck 1 at 08:29:40
8	300 State St	Salt Lake City	UT	84103	9	EOD	Delivered by Truck 2 at 09:52:20
9	410 S State St	Salt Lake City	UT	84111	2	EOD	Delivered by Truck 1 at 11:51:20
10	600 E 900 S	Salt Lake City	UT	84105	1	EOD	Delivered by Truck 1 at 08:39:00
11	2600 Taylorsville Blvd	Salt Lake City	UT	84118	1	EOD	Delivered by Truck 2 at 11:02:20
12	3575 W Valley Central Station Bus Loop	West Valley City	UT	84119	1	EOD	Delivered by Truck 2 at 10:35:00
13	2010 W 500 S	Salt Lake City	UT	84104	2	10:30	Delivered by Truck 1 at 09:02:20
14	4300 S 1300 E	Millcreek	UT	84117	88	10:30	Delivered by Truck 1 at 08:06:20
15	4580 S 2300 E	Holladay	UT	84117	4	9:00	Delivered by Truck 1 at 08:13:00
16	4580 S 2300 E	Holladay	UT	84117	88	10:30	Delivered by Truck 1 at 08:13:00
17	3148 S 1100 W	Salt Lake City	UT	84119	2	EOD	Delivered by Truck 1 at 11:20:00
18	1488 4800 S	Salt Lake City	UT	84123	6	EOD	Delivered by Truck 2 at 10:59:00
19	177 W Price Ave	Salt Lake City	UT	84115	37	EOD	Delivered by Truck 1 at 09:31:20
20	3595 Main St	Salt Lake City	UT	84115	37	10:30	Delivered by Truck 1 at 09:29:40
21	3595 Main St	Salt Lake City	UT	84115	3	EOD	Delivered by Truck 1 at 09:29:40
22	6351 S 900 E	Murray	UT	84121	2	EOD	Delivered by Truck 1 at 10:38:00
23	5100 S 2700 W	Salt Lake City	UT	84118	5	EOD	Delivered by Truck 1 at 11:04:00
24	5025 State St	Murray	UT	84107	7	EOD	Delivered by Truck 1 at 10:28:00
25	5383 S 900 E #104	Salt Lake City	UT	84117	7	10:30	Delivered by Truck 2 at 09:13:00
26	5383 S 900 E #104	Salt Lake City	UT	84117	25	EOD	Delivered by Truck 1 at 10:33:40
27	1060 Dalton Ave S	Salt Lake City	UT	84104	5	EOD	Delivered by Truck 1 at 11:35:20
28	2835 Main St	Salt Lake City	UT	84115	7	EOD	Delivered by Truck 2 at 09:32:40
29	1330 2100 S	Salt Lake City	UT	84106	2	10:30	Delivered by Truck 1 at 08:29:40
30	300 State St	Salt Lake City	UT	84103	1	10:30	Delivered by Truck 1 at 08:48:20
31	3365 S 900 W	Salt Lake City	UT	84119	1	10:30	Delivered by Truck 2 at 10:14:20
32	3365 S 900 W	Salt Lake City	UT	84119	1	EOD	Delivered by Truck 2 at 10:14:20
33	2530 S 500 E	Salt Lake City	UT	84106	1	EOD	Delivered by Truck 2 at 09:29:00
34	4580 S 2300 E	Holladay	UT	84117	2	10:30	Delivered by Truck 1 at 08:13:00
35	1060 Dalton Ave S	Salt Lake City	UT	84104	88	EOD	Delivered by Truck 1 at 11:35:20
36	2300 Parkway Blvd	West Valley City	UT	84119	88	EOD	Delivered by Truck 2 at 10:24:40
37	410 S State St	Salt Lake City	UT	84111	2	10:30	Delivered by Truck 1 at 08:45:00
38	410 S State St	Salt Lake City	UT	84111	9	EOD	Delivered by Truck 2 at 09:47:00
39	2010 W 500 S	Salt Lake City	UT	84104	9	EOD	Delivered by Truck 1 at 09:02:20
40	380 W 2880 S	Salt Lake City	UT	84115	45	10:30	Delivered by Truck 1 at 09:24:20

E. Provide screenshots showing successful completion of the code that includes the total mileage traveled by all trucks.



F. Justify the package delivery algorithm used in the solution as written in the original program by doing the following:
F1. Describe two or more strengths of the algorithm used in the solution.

The package delivery algorithm chosen for this project was the nearest neighbor algorithm. One strength of this algorithm is that it is capable of finding a fairly optimized solution relatively quickly. While it may not be the best possible solution, the nearest neighbor algorithm provides a solution that is reasonably good and meets the requirements of the assignment. The algorithm is also simple and easy to implement. A simple algorithm is ideal because it allows the programmer to focus on other areas of coding, such as creating efficient data structures. Finally, the nearest neighbor algorithm is relatively space efficient, requiring fairly little memory to accomplish its goal.

F2. Verify that the algorithm used in the solution meets all requirements in the scenario.

The requirements for this scenario state that all package delivery constraints, including delivery deadlines, must be met, while traveling under 140 miles. During the course of program execution, the algorithm heuristically sorts the packages into an optimized order, and then all deliveries are completed. After the delivery operations are complete, reviewing package delivery times through the track package functions within the user interface shows that all packages were delivered on time and according to their delivery specifications. The total mileage was 92.2 miles.

F3. Identify two other named algorithms that are different from the algorithm implemented in the solution and would meet all requirements in the scenario.

Two other named algorithms that could be implemented for this scenario are the 2-Opt Algorithm and the Genetic Algorithm. These algorithms are known to find solutions that are at least as efficient as nearest neighbor, meaning they would be capable of delivering all packages on time and according to their delivery specifications while not exceeding 140 miles traveled.

F3a. Describe how both algorithms identified in part F3 are different from the algorithm used in the solution.

The 2-Opt and Genetic algorithms are different in their approach to path finding than the nearest neighbor algorithm. One difference is that while the nearest neighbor algorithm iterates through nodes to establish a single path, both the 2-Opt and Genetic algorithms begin with an initial path, and then employ additional methods to further revise the initial solution to find a more optimal path. Another difference is that 2-Opt and the Genetic algorithms have more steps in their operation. This means that the algorithms require more complex coding implementations than nearest neighbor. Also, due to the extra steps, the time complexity of the 2-Opt and Genetic algorithms tend to be higher than the Nearest Neighbor algorithm. However, the added coding complexity and computational resource requirements of implementing the 2-Opt or Genetic algorithm pays off in the form of a more optimized solution relative to the nearest neighbor approach (Johnson, McGeoch).

G. Describe what you would do differently, other than the two algorithms identified in part F3, if you did this project again, including details of the modifications that would be made.

If I were to do this assignment over again, I would attempt to eliminate manually created package groups and instead find a way to group packages automatically. This would enable the program to better adapt to package delivery requirements and would also enable the program to handle entirely different package sets. The approach I would take would be to replace the manual package group assignments with an algorithm that iterates through the packages and groups them based on their delivery requirements. Then, package addresses could be examined to identify packages destined for the same locations so they could be grouped together. Taking these steps would eliminate the need to manual truck loading, and if done well, would enable the path finding algorithm to achieve better results.

H. Verify that the data structure used in the solution meets all requirements in the scenario.

The scenario requirements state that a hash table must be developed without using any additional libraries or classes, that has an insertion function that takes the package ID as input and inserts the package and its data components into the hash table. Below is a screenshot of the hash table implementation which clearly shows that no additional libraries or classes have been used and shows the presence of an insert function that takes the package ID as input.

hashtable.py

```
2 usages
class HashTable:
    # Constructor, defaults to 10 buckets
    # Assigns each bucket with an empty list
    def __init__(self, initial_capacity=10):
        self.table = []
        for i in range(initial_capacity):
            self.table.append([])

    # O(N) worst case if all keys hash to same bucket
    # Accepts the package ID as input and inserts the package object, which contains the data components
    1 usage
    def insert(self, key, item):
        # Determine target bucket
        bucket = int(key) % len(self.table)
        bucket_list = self.table[bucket]

        # Update item if already exists in 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

    # O(N) worst case all keys hashed to same bucket
    6 usages (6 dynamic)
    def search(self, key):
        # Determine target bucket
        bucket = int(key) % len(self.table)
        bucket_list = self.table[bucket]

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

H1. Identify two other data structures that could meet the same requirements in the scenario.

Two alternative data structures that could have been utilized in this scenario are a Linked List and a Binary Search Tree. Each of these could successfully store package data and enable searching, insertion, and deletion.

H1a. Describe how each data structure identified in H1 is different from the data structure used in the solution.

Linked lists and binary search trees have characteristics that set them apart from hash tables in terms of storage and search efficiency. A linked list is a linear structure in which each element points to the next element. Unlike hash tables, linked lists are ordered, which is advantageous for traversal. However, searching linked lists requires iterating through all elements until the desired element is found, which is less efficient than hash tables. Binary Search Trees organize data in a tree structure in which elements which are less than the previous element (such as package IDs) go to the left sub tree, while greater elements go to the right. Searching a binary search tree, while quicker than searching a linked list, is still not as efficient as searching a hash table. The primary distinction between binary search trees, linked lists, and hash tables lies in the trade-offs between ordering and search efficiency (Zybooks 3.3).

References

R. Lysecky, F. Vahid, and E. Olds. *C 950: Data Structures and Algorithms II*. zyBook, zyBooks Inc., 2022.

D. Johnson, L. McGeoch. The Traveling Salesman Problem: A Case Study in Local Optimization. 1995, November 20. https://www.cs.ubc.ca/~hutter/previous-earg/EmpAlgReadingGroup/TSP-JohMcg97.pdf