Sravani Yerramaneni
ID: 001359531

Data Structures & Algorithms 2

B2: APPLICATION OF PROGRAMMING MODELS

Since all the data is stored on the same local machine on which the program is run, there is no communication protocol and there is no target host environment or defined interaction semantics. The program is written in Python 3.8 and is run in the PyCharm IDE. All external data is in the form of .csv files and are located in the project folder itself. They are then processed by the program in execution.

B3: SPACE-TIME AND BIG-O

The space-time complexity for the entire program is $O(N^2)$.

B4: ADAPTABILITY

The chosen algorithm is scalable with an asymptotic space-time complexity of $O(N^2)$ where N is the number of all addresses on a given route.
Even if the program's scale were to increase substantially, the number of addresses on each truck's route will likely not grow much from its current max value of 16 because a truck is limited by the number of packages it can carry. Therefore, the time complexity of the routing algorithm will stay close to $O(16^2)$ no matter how many addresses are stored in the program.

B5: SOFTWARE EFFICIENCY AND MAINTAINABILITY

Currently, the routing algorithm has an asymptotic space-time complexity of $O(N^2)$ where N is the number of all addresses on a given route. Even if the program's scale were to increase substantially, the number of addresses on each truck's route will likely not grow much from its current max value of 16 because a truck is limited by the number of packages it can carry. So then the time complexity of the routing algorithm will stay close to $O(16^2)$ no matter how many addresses are stored in the program.
Another important modification involves the task of choosing which package goes on each truck which was done manually in order to maintain delivery deadlines. If the number of packages to be delivered each day increased substantially, it would not be feasible to manually assign the packages to each truck anymore. This would have to become automated.

Also, the current *address_dict* dictionary only stores the street address of each address. As the program scales to accommodate more addresses, the dictionary should expand to contain the street address and the city to avoid any collisions between two different addresses with the same street address.

The code should be easily maintainable because there are relatively few data structures and functions to keep track of.
There are 4 main data structures used:

- ***Truck*** class instantiated 3 times (truck1, truck2, truck3) and contains the start time and end time for a trip along with a list of packages in the order they will be delivered
- ***package_hash*** which stores all the package information
- ***address_dict*** which maps a string *address* to an integer *address_id*
- ***distances_list*** which stores an adjacency matrix of the distances between all the addresses

There are 5 main functions used:

- ***load_data()*** which loads all the package, address, and distance information from the csv files into the corresponding data structures
- ***pick_truck()*** which assigns each package to a truck (either 1, 2, or 3) based on manually created lists
- ***route()*** which decides the order in which the packages are delivered for a given truck/ load, computes distance traveled, and sets each package's delivery time
    - ***find_next_address()*** which finds the closest address in an *addresses_to_route* list to a given address
- ***check_status()*** which allows the user to interact with the program
    - ***get_valid_time()***
    - ***get_valid_package_no()***
    - ***insert_package()***

I1: STRENGTHS OF THE CHOSEN ALGORITHM

Algorithm Overview:

1. A truck object containing a dictionary mapping addresses to a list of packages is passed into the *route()* function.

```
def route(truck):
    address_package_dict = truck.address_package_dict
```

Sravani Yerramaneni
ID: 001359531

2. The unique addresses of each package are retrieved and stored in a list called *addresses_to_route*. A list *package_route* is initialized. An integer *prev_address* is initialized to 0 and represents the current address's ID.

```
addresses_to_route = list(address_package_dict.keys())
package_route = []
prev_address = 0
```

3. Starting from the hub, the next closest address is computed by calling the *find_next_address()* function and passing it the starting address *prev_address* and the list *addresses_to_route*.

```
find_next_address(prev_address, addresses_to_route)
```

   a. The function *find_next_address()* iterates through each value in the *addresses_to_route* list and finds the address with the minimum distance from the starting address *prev_address*

```
min_distance = -1
next_address = -1
for address in addresses_to_route:
    distance = distance between prev_address and address
    if distance < min_distance or min_distance == -1:
        min_distance = distance
        next_address = address
return [next_address, min_distance]
```

4. When the next address is retrieved from the *find_next_address()* function, it is set as the new value of *prev_address.* that address is removed from the *addresses_to_route* list.

```
prev_address = find_next_address(prev_address, addresses_to_route)
addresses_to_route.remove(prev_address)
```

5. Then the list of packages at that address is retrieved using the *address_package_dict* dictionary and appended to the *package_route* list.

```
package_list = address_package_dict.get(prev_address)
for p in package_list:
    package_route.append(p)
```

6. Steps 3 through 5 are repeated *x* times where *x* is the number of addresses in the given truck's route.

```
for i in range(length of addresses_to_route):
    perform steps 3 through 5
```

Currently, the routing algorithm has an asymptotic space-time complexity of $O(N^2)$ where N is the number of all addresses on a given route. Even if the program's scale were to increase substantially, the number of addresses on each truck's route will likely not grow much from its current max value of 16 because a truck is limited by the number of packages it can carry. So then the time complexity of the routing algorithm will stay close to $O(16^2)$ no matter how many addresses are stored in the program. This makes the routing algorithm very scalable.

Another benefit of the chosen routing algorithm is that in addition to having a relatively small space-time complexity, the algorithm significantly decreases the distance traveled by each truck. After assigning packages to each truck but before implementing the routing algorithm, the total mileage was 148 miles. After implementation, the algorithm yielded a total mileage of 90 miles.

I3: OTHER POSSIBLE ALGORITHMS

I3A: ALGORITHM DIFFERENCES

My algorithm found in the *find_next_address()* function is a greedy algorithm. It checks each address for the next closest address in the truck's route and assigns that closest address as the next address delivered to. It has a time complexity of $O(N^2)$. Two other approaches to the algorithm include a recursive solution and a dynamic programming solution.

A recursive solution would take in as parameters a starting point address and an addresses_to_route list and recursively break up the addresses_to_route list into smaller elements until there are only two or less addresses in the list. If there are two addresses left, it will then return the address that is closest to the starting point. If there is one address left in the list, it will return that address. The recursive function itself has a time complexity that is slightly larger than O(N) where N is the length of the addresses_to_route list. So, with a list of 8 elements, the function will run 10 times. With a list of 11 elements, the function will run 19 times. When creating the ordered route, the time complexity becomes N^2 because the next address is found for each address in the route. So, the algorithm is called N times making the time complexity N^2.

A dynamic programming solution would take in the list of addresses on a given truck's route and then iteratively compute all possible paths, adding a new address to each path in each iteration. This would yield the optimal solution, but is highly unscalable with a time complexity of $O(N^2*2^N)$. To keep time complexity in polynomial time however, the

maximum path length can be set to some integer. Then, each time the function is called, instead of returning the entire path, the function can return the next *x* addresses to append to the ordered route, where *x* is the maximum path length chosen.

J: DIFFERENT APPROACH

If I were to change one thing about the project, it would be the current way of storing each package's information as a list. This is an unnecessarily unintuitive decision and will likely not scale well if packages were to store more data and needed to have their own methods.

To remedy this, I would create a Package class to store all the fields concerning each package and methods that apply to a package. So the HashTable would go from being a list of lists to being a list of package objects.

K1B: OVERHEAD

Memory and bandwidth aspects aren't very relevant to this program as everything is run on the same local machine. However, in terms of the computational time when handling data, loading the data into the appropriate list or dictionary from the .csv files has a time complexity of $O(N^2)$. Accessing information from the created lists or dictionaries has a space-time complexity of O(1) because all information is accessed through either its index or key.

K1C: IMPLICATIONS

When the number of packages increases, creating a Package class would be useful to help distinguish fields and methods for package objects. Currently all methods and information related to a package is stored in the HashTable class, but adding and modifying functionality would be easier and more intuitive if there was a separate package class. Also, the function that assigns each package to a truck is currently done manually to adhere to delivery deadlines. If the number of packages were to increase, it would be more practical for this function to be completely automated.

Currently, 3 truck objects are instantiated within the main program and are then passed into the *pick_truck()* and *route()* functions where a truck is assigned it's packages and where a truck's list of packages are ordered in the order they will be delivered. When the number of trucks increases, it may be more practical to have truck objects created

automatically within the *pick_truck()* function. A global list could be created to keep track of all the trucks that are created within that function.

The current *address_dict* dictionary only stores the street address of each address. As the program scales to accommodate more addresses and therefore cities, the dictionary should expand to contain the street address and the city to avoid any collisions between two different addresses with the same street address.

B6: SELF-ADJUSTING DATA STRUCTURES

When an instance of the HashTable class is created, that HashTable's list is initialized by default to hold 'None' inside its first *x* number of indexes where *x* is taken in as an optional parameter and is set to 41 by default. Since each package number is unique, when a package is inserted into the list, its package number is set as its index in the list. If a user wants to enter a package with a package id greater than the number of elements in the HashTable's list, the HashTable class is able to increase the size of its list by extending the list by *y* number of indexes containing 'None' where *y* is the difference between the package number being inserted and the current size of the list. Extending the list is done using the built-in '.extend()' method for lists. It has a space-time complexity of O(k) where k is the number of elements the current list is being extended by. Having a linear-time complexity makes this very scalable and does not substantially affect running time.

D: DATA STRUCTURE

D1: EXPLANATION OF DATA STRUCTURE

K1A: EFFICIENCY

Each package's information is stored as a list. The HashTable storing all the packages is implemented as a list of lists. The *insert()* function in the HashTable class takes a package's information (id, address, deadline… etc.) as parameters and then creates a list containing those parameters. That list is then inserted into the HashTable's list with the index being the package's id. This has a space-time complexity of O(k) k is the amount by which the list has to grow. If the package id inserted is within the size of the list, the list does not grow and the insert function becomes a constant-time operation.

To modify a package's information, there are a series of *'setter'* functions (*set_address(), set_zip_code()*... etc.) that take in the package id and the value to be updated to. They then access the relevant indices of the HashTable's list of lists and set

the relevant package information to the given value. The '*setter'* functions all have a space-time complexity of O(1).

To retrieve a list containing a package's information, there is a *get()* function which takes in a package ID as a parameter. This has a space-time complexity of O(1).
To retrieve specific information about a package, there is a *search()* function which takes in as parameters: the package id and a string. The string specifies which package information the user wants to receive and can be '*address', 'deadline', 'city',* etc. This has a space-time complexity of O(1).

The HashTable class stores a list of indexes that have been populated in the hash table called *indexes*. The indexes are the same as the package ids so to iterate over all the packages, you can iterate over all the package IDs in the *indexes* list. This ensures that only the populated indices of the HashTable's list are accessed.

I2: VERIFICATION OF ALGORITHM
K1: VERIFICATION OF DATA STRUCTURE
Package IDs range from 1 to 40 and address IDs range from 0 to 26.

TRUCK 1:
Start Time: 8:00 AM
End Time: 9:55 AM
Distance: 34.6 miles
Package Route: 14, 15, 16, 34, 7, 29, 20, 21, 5, 37, 38, 3, 8, 30, 13, 39
Address Route: 20, 21, 2, 17, 19, 8, 12, 6

TRUCK 2:
Start Time: 9:05 AM
End Time: 10:19 AM
Distance: 22.4 miles
Package Route: 19, 12, 36, 6, 17, 31, 32, 4, 40, 28, 1, 2, 33, 25, 26
Address Route: 4, 16, 7, 13, 14, 15, 18, 11, 5, 9, 24

TRUCK 3:
Start Time: 10:20 AM
End Time: 12:08 AM
Distance: 32.4 miles

Sravani Yerramaneni
ID: 001359531

Package Route: 24, 22, 11, 23, 18, 27, 35, 10, 9
Address Route: 22, 26, 10, 23, 3, 1, 25, 19

K2: OTHER DATA STRUCTURES

K2A: DATA STRUCTURE DIFFERENCES

One data structure that could be used is a graph. Since the use of dictionaries is prohibited, the graph could be implemented as two separate lists, one containing lists of package information and the other list containing adjacent vertices. The 'adjacent vertices' could include packages that have to be delivered on the same truck and/or packages that have to be delivered to the same address. This could come in handy when assigning packages to each truck. Through a breadth-first search, when one package is assigned to a truck, all of its adjacent vertices are assigned to that truck as well. This has a space-time complexity of O(V+E) where V is the number of packages and E is the number of connections between packages. Currently, the data structure in use in the program stores no relationships between the packages. The packages are not sorted or grouped in any way. While this may be slightly more efficient now with most operations being of constant-time, in the future when the assigning of the packages to each truck has to become automated, the graph structure could provide an efficient way to load each truck.

Another data structure that could have been implemented is a binary search tree where each node of the tree stores package information and the packages could have been sorted by their delivery deadline. This could prove to be useful if the *pick_truck()* function were to become automated so packages near the top of the tree (with earlier delivery deadlines) are routed first in order to comply with their delivery deadlines. The binary search tree has a space-time complexity of O(N) for inserting, retrieving, and modifying nodes.
Currently, though, the *pick_truck()* function is not automated and so there is no need to sort the packages based on any attribute. So the data structure in use in the program now would be more efficient with a space-time complexity of O(1) for retrieving and modifying package information and a space-time complexity of O(k) for inserting information where k is the number of indices the list needs to extend by.

Sravani Yerramaneni
ID: 001359531

G1: FIRST STATUS CHECK

```
TOTAL MILEAGE: 89.4
ALL PACKAGES DELIVERED ON TIME.

Enter 'insert' to insert a package.
To look-up package information, enter 'lookup': lookup

[EXAMPLE: 13:30] Enter a time in 24-hour format: 9:00
[Enter 'all' for all packages.] Enter a package number: all

1             195 W Oakland Ave, Salt Lake City 84115              Deadline: 10:30:00       Weight: 21            Status: AT HUB
2                2530 S 500 E, Salt Lake City 84106                Deadline: 23:59:59       Weight: 44            Status: AT HUB
3              233 Canyon Rd, Salt Lake City 84103                 Deadline: 23:59:59       Weight: 2             Status: EN ROUTE
4               380 W 2880 S, Salt Lake City 84115                 Deadline: 23:59:59       Weight: 4             Status: AT HUB
5              410 S State St, Salt Lake City 84111                Deadline: 23:59:59       Weight: 5             Status: DELIVERED
6             3060 Lester St, West Valley City 84119               Deadline: 10:30:00       Weight: 88            Status: AT HUB
7               1330 2100 S, Salt Lake City 84106                  Deadline: 23:59:59       Weight: 8             Status: DELIVERED
8               300 State St, Salt Lake City 84103                 Deadline: 23:59:59       Weight: 9             Status: EN ROUTE
9               300 State St, Salt Lake City 84103                 Deadline: 23:59:59       Weight: 2             Status: AT HUB
10             600 E 900 South, Salt Lake City 84105               Deadline: 23:59:59       Weight: 1             Status: AT HUB
11          2600 Taylorsville Blvd, Salt Lake City 84118           Deadline: 23:59:59       Weight: 1             Status: AT HUB
12  3575 W Valley Central Station bus Loop, West Valley City 84119 Deadline: 23:59:59       Weight: 1             Status: AT HUB
13              2010 W 500 S, Salt Lake City 84104                 Deadline: 10:30:00       Weight: 2             Status: EN ROUTE
14                4300 S 1300 E, Millcreek 84117                   Deadline: 10:30:00       Weight: 88            Status: DELIVERED
15                4580 S 2300 E, Holladay 84117                    Deadline: 09:00:00       Weight: 4             Status: DELIVERED
16                4580 S 2300 E, Holladay 84117                    Deadline: 10:30:00       Weight: 88            Status: DELIVERED
17             3148 S 1100 W, Salt Lake City 84119                 Deadline: 23:59:59       Weight: 2             Status: AT HUB
18              1488 4800 S, Salt Lake City 84123                  Deadline: 23:59:59       Weight: 6             Status: AT HUB
19             177 W Price Ave, Salt Lake City 84115               Deadline: 23:59:59       Weight: 37            Status: AT HUB
20               3595 Main St, Salt Lake City 84115                Deadline: 10:30:00       Weight: 37            Status: DELIVERED
21               3595 Main St, Salt Lake City 84115                Deadline: 23:59:59       Weight: 3             Status: DELIVERED
22              6351 South 900 East, Murray 84121                  Deadline: 23:59:59       Weight: 2             Status: AT HUB
23          5100 South 2700 West, Salt Lake City 84118             Deadline: 23:59:59       Weight: 5             Status: AT HUB
24                 5025 State St, Murray 84107                     Deadline: 23:59:59       Weight: 7             Status: AT HUB
25         5383 South 900 East #104, Salt Lake City 84117          Deadline: 10:30:00       Weight: 7             Status: AT HUB
26         5383 South 900 East #104, Salt Lake City 84117          Deadline: 23:59:59       Weight: 25            Status: AT HUB
27            1060 Dalton Ave S, Salt Lake City 84104              Deadline: 23:59:59       Weight: 5             Status: AT HUB
28               2835 Main St, Salt Lake City 84115                Deadline: 23:59:59       Weight: 7             Status: AT HUB
29               1330 2100 S, Salt Lake City 84106                 Deadline: 10:30:00       Weight: 2             Status: DELIVERED
30               300 State St, Salt Lake City 84103                Deadline: 10:30:00       Weight: 1             Status: EN ROUTE
31               3365 S 900 W, Salt Lake City 84119                Deadline: 10:30:00       Weight: 1             Status: AT HUB
32               3365 S 900 W, Salt Lake City 84119                Deadline: 23:59:59       Weight: 1             Status: AT HUB
33               2530 S 500 E, Salt Lake City 84106                Deadline: 23:59:59       Weight: 1             Status: AT HUB
34                 4580 S 2300 E, Holladay 84117                   Deadline: 10:30:00       Weight: 2             Status: DELIVERED
35            1060 Dalton Ave S, Salt Lake City 84104              Deadline: 23:59:59       Weight: 88            Status: AT HUB
36           2300 Parkway Blvd, West Valley City 84119             Deadline: 23:59:59       Weight: 88            Status: AT HUB
37              410 S State St, Salt Lake City 84111               Deadline: 10:30:00       Weight: 2             Status: DELIVERED
38              410 S State St, Salt Lake City 84111               Deadline: 23:59:59       Weight: 9             Status: DELIVERED
39               2010 W 500 S, Salt Lake City 84104                Deadline: 23:59:59       Weight: 9             Status: EN ROUTE
40               380 W 2880 S, Salt Lake City 84115                Deadline: 10:30:00       Weight: 45            Status: AT HUB

Enter 'exit' to exit application.
To continue, enter 'c': exit


Process finished with exit code 0
```

Sravani Yerramaneni
ID: 001359531

G2: SECOND STATUS CHECK

```
TOTAL MILEAGE: 89.4
ALL PACKAGES DELIVERED ON TIME.

Enter 'insert' to insert a package.
To look-up package information, enter 'lookup': lookup

[EXAMPLE: 13:30] Enter a time in 24-hour format: 10:00
[Enter 'all' for all packages.] Enter a package number: all

1            195 W Oakland Ave, Salt Lake City 84115              Deadline: 10:30:00       Weight: 21            Status: DELIVERED
2               2530 S 500 E, Salt Lake City 84106                Deadline: 23:59:59       Weight: 44            Status: DELIVERED
3              233 Canyon Rd, Salt Lake City 84103                Deadline: 23:59:59       Weight: 2             Status: DELIVERED
4               380 W 2880 S, Salt Lake City 84115                Deadline: 23:59:59       Weight: 4             Status: DELIVERED
5              410 S State St, Salt Lake City 84111               Deadline: 23:59:59       Weight: 5             Status: DELIVERED
6             3060 Lester St, West Valley City 84119              Deadline: 10:30:00       Weight: 88            Status: DELIVERED
7               1330 2100 S, Salt Lake City 84106                 Deadline: 23:59:59       Weight: 8             Status: DELIVERED
8               300 State St, Salt Lake City 84103                Deadline: 23:59:59       Weight: 9             Status: DELIVERED
9               300 State St, Salt Lake City 84103                Deadline: 23:59:59       Weight: 2             Status: AT HUB
10            600 E 900 South, Salt Lake City 84105               Deadline: 23:59:59       Weight: 1             Status: AT HUB
11         2600 Taylorsville Blvd, Salt Lake City 84118           Deadline: 23:59:59       Weight: 1             Status: AT HUB
12  3575 W Valley Central Station bus Loop, West Valley City 84119 Deadline: 23:59:59       Weight: 1             Status: DELIVERED
13              2010 W 500 S, Salt Lake City 84104                Deadline: 10:30:00       Weight: 2             Status: DELIVERED
14                4300 S 1300 E, Millcreek 84117                  Deadline: 10:30:00       Weight: 88            Status: DELIVERED
15                4580 S 2300 E, Holladay 84117                   Deadline: 09:00:00       Weight: 4             Status: DELIVERED
16                4580 S 2300 E, Holladay 84117                   Deadline: 10:30:00       Weight: 88            Status: DELIVERED
17             3148 S 1100 W, Salt Lake City 84119                Deadline: 23:59:59       Weight: 2             Status: DELIVERED
18              1488 4800 S, Salt Lake City 84123                 Deadline: 23:59:59       Weight: 6             Status: AT HUB
19            177 W Price Ave, Salt Lake City 84115               Deadline: 23:59:59       Weight: 37            Status: DELIVERED
20              3595 Main St, Salt Lake City 84115                Deadline: 10:30:00       Weight: 37            Status: DELIVERED
21              3595 Main St, Salt Lake City 84115                Deadline: 23:59:59       Weight: 3             Status: DELIVERED
22             6351 South 900 East, Murray 84121                  Deadline: 23:59:59       Weight: 2             Status: AT HUB
23          5100 South 2700 West, Salt Lake City 84118            Deadline: 23:59:59       Weight: 5             Status: AT HUB
24                5025 State St, Murray 84107                     Deadline: 23:59:59       Weight: 7             Status: AT HUB
25         5383 South 900 East #104, Salt Lake City 84117         Deadline: 10:30:00       Weight: 7             Status: EN ROUTE
26         5383 South 900 East #104, Salt Lake City 84117         Deadline: 23:59:59       Weight: 25            Status: EN ROUTE
27            1060 Dalton Ave S, Salt Lake City 84104             Deadline: 23:59:59       Weight: 5             Status: AT HUB
28               2835 Main St, Salt Lake City 84115               Deadline: 23:59:59       Weight: 7             Status: DELIVERED
29              1330 2100 S, Salt Lake City 84106                 Deadline: 10:30:00       Weight: 2             Status: DELIVERED
30              300 State St, Salt Lake City 84103                Deadline: 10:30:00       Weight: 1             Status: DELIVERED
31              3365 S 900 W, Salt Lake City 84119                Deadline: 10:30:00       Weight: 1             Status: DELIVERED
32              3365 S 900 W, Salt Lake City 84119                Deadline: 23:59:59       Weight: 1             Status: DELIVERED
33              2530 S 500 E, Salt Lake City 84106                Deadline: 23:59:59       Weight: 1             Status: DELIVERED
34                4580 S 2300 E, Holladay 84117                   Deadline: 10:30:00       Weight: 2             Status: DELIVERED
35           1060 Dalton Ave S, Salt Lake City 84104              Deadline: 23:59:59       Weight: 88            Status: AT HUB
36           2300 Parkway Blvd, West Valley City 84119            Deadline: 23:59:59       Weight: 88            Status: DELIVERED
37              410 S State St, Salt Lake City 84111              Deadline: 10:30:00       Weight: 2             Status: DELIVERED
38              410 S State St, Salt Lake City 84111              Deadline: 23:59:59       Weight: 9             Status: DELIVERED
39              2010 W 500 S, Salt Lake City 84104                Deadline: 23:59:59       Weight: 9             Status: DELIVERED
40              380 W 2880 S, Salt Lake City 84115                Deadline: 10:30:00       Weight: 45            Status: DELIVERED

Enter 'exit' to exit application.
To continue, enter 'c': exit


Process finished with exit code 0
```

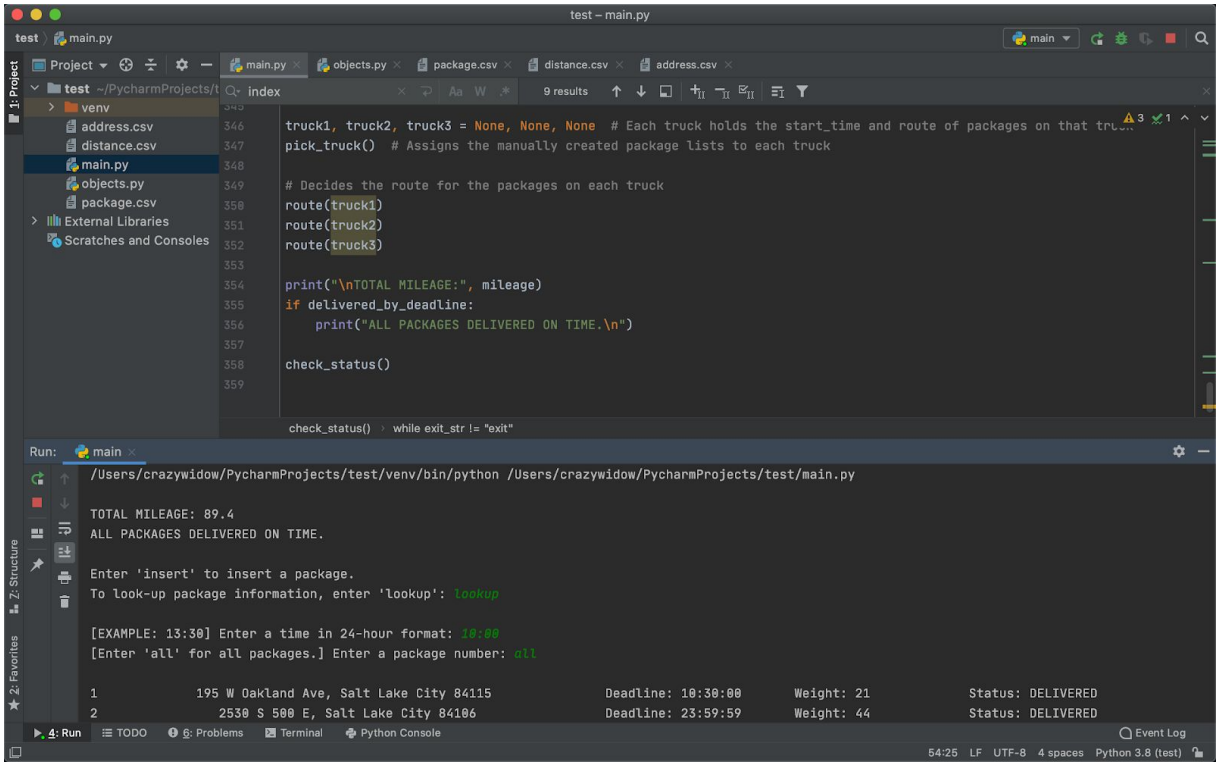
Sravani Yerramaneni
ID: 001359531

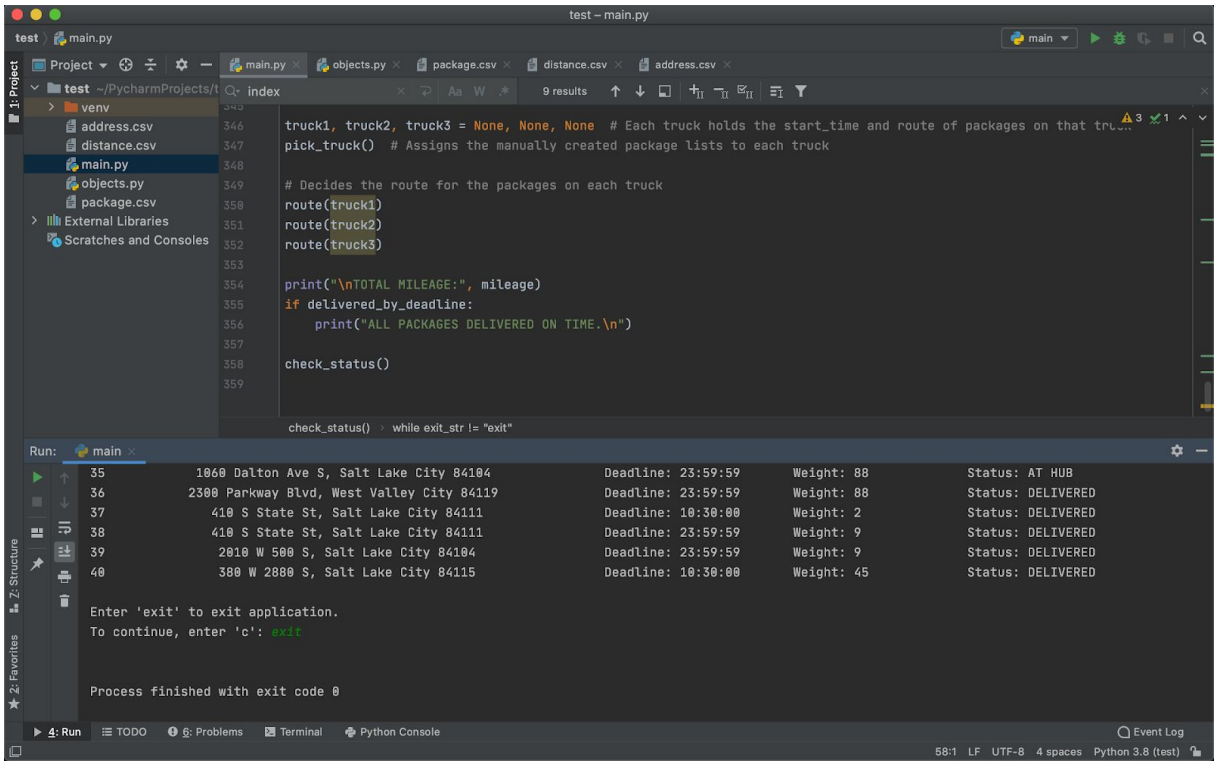
G3: THIRD STATUS CHECK

```
TOTAL MILEAGE: 89.4
ALL PACKAGES DELIVERED ON TIME.

Enter 'insert' to insert a package.
To look-up package information, enter 'lookup': lookup

[EXAMPLE: 13:30] Enter a time in 24-hour format: 13:00
[Enter 'all' for all packages.] Enter a package number: all

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

Enter 'exit' to exit application.
To continue, enter 'c': exit


Process finished with exit code 0
```

Sravani Yerramaneni
ID: 001359531

H: SCREENSHOTS OF CODE EXECUTION





Sravani Yerramaneni
ID: 001359531

References

Lysecky, R. (2018). *Data Structures and Algorithms (WGU 949/950)*. Retrieved from https://learn.zybooks.com/zybook/WGUC950AY20182019