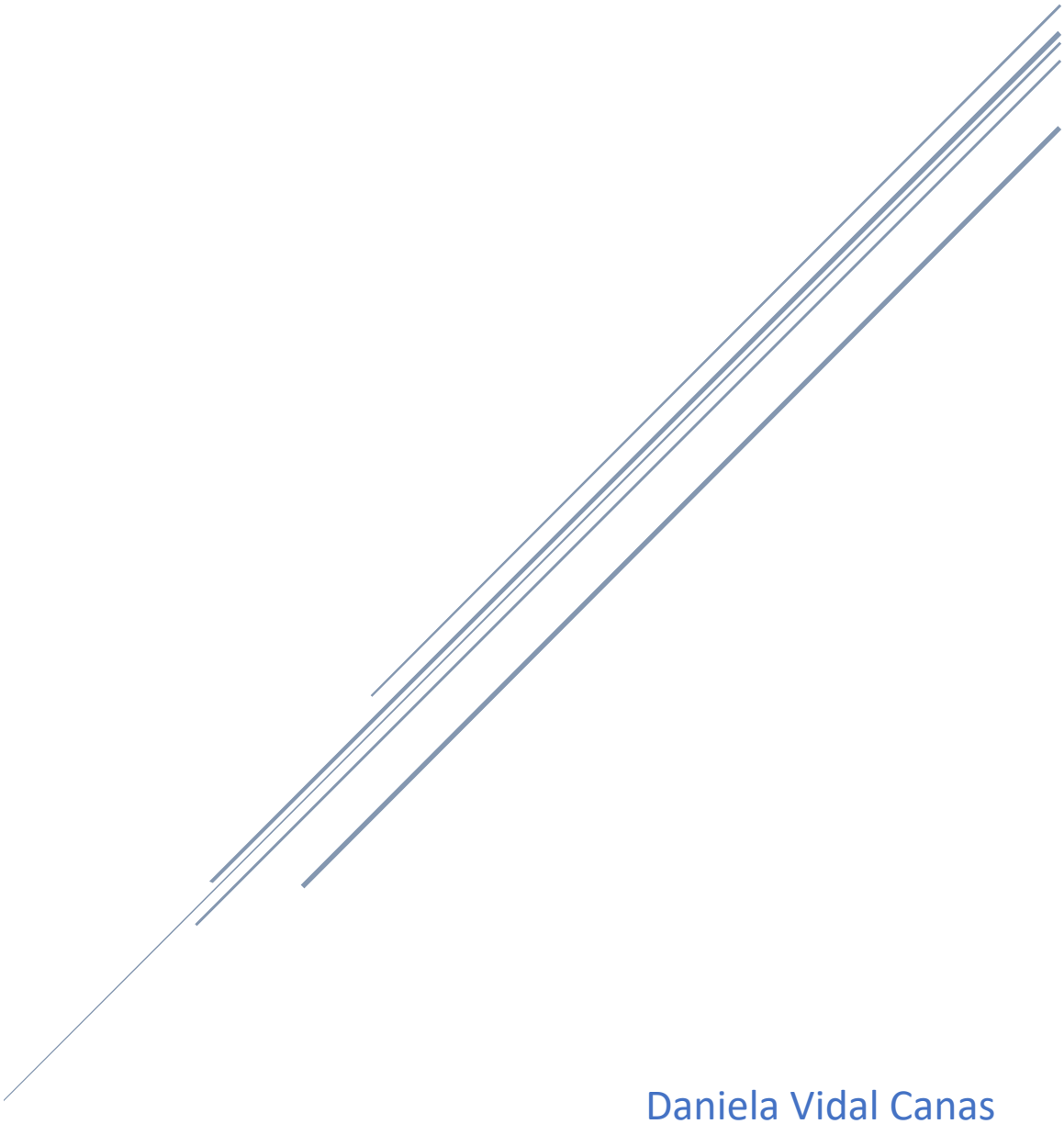
C950 – DATA STRUCTURES AND ALGORITHMS II

Project Documentation



Daniela Vidal Canas
Student ID: 001172091

Core algorithm overview

Stated Problem:

The purpose of this project is to create an algorithm using Python as core language to develop an efficient greedy algorithm to determine the route of a delivery service that meets all packages requirements (as deadlines, delays or change of address) and does it in the shortest miles. In the scenario, there is an average on 40 packages to deliver with 2 available drivers and 3 trucks. Any requirement may be changed at any time and the supervisor must be able to review the status of all packages at any given time. The algorithm needs to be broad enough to be used in different cities.

Algorithm Overview:

The Nearest Neighbor algorithm gradually constructs a way by repeatedly selecting the nearest neighbor and adding it to the path (Nilsson) ,

1. Select a point.
2. Find the nearest unvisited point and go there.
3. Are there any unvisited points left? If yes, repeat step 2.
4. 4. Return to the first point

The Nearest Neighbor algorithm will often keep its tours within 25% of the Held-Karp lower bound (McGeoch, 1995)

Algorithm logic and space-time complexities:

```
1.  def shortest_path_for_truck(truck_position, truck_packages):
2.      for package in truck_packages:
3.          if package.address == truck_position:
4.              update_package_status("Delivered”)
5.              truck_packages -> remove (package)
6.      if truck_packages > 0:
7.          min_miles_to_travel, closer_address =
            calculate_closer_adjacent(truck_position,truck_packages)
8.      truck_miles_traveled += min_miles
9.      truck_position = closer_address
10.     return shortest_path_for_truck(truck_position, truck_packages)
11. def calculate_closer_adjacent(truck_position, truck_packages):
12.     min_miles_to_travel = math.inf
13.     for each package in truck_packages:
14.         distance_to_adjacent = city_map -> distances -> get(truck_position, package_address)
15.         if distance_to_adjacent < min_miles_to_travle:
16.             min_miles_to_tavel = distance_to_adjacent
17.             closer_address = package_address
18.     return min_miles_to_travel, cloaser_address
```

On this pseudocode, from the truck_position (on the first iteration it would be the starting Hub), we calculate the distance to all the packages left in the truck and then define the package to which traveling is shortest. Then, we update the miles traveled by the truck, the package status and set the truck_position to the closer_address location, for a new iteration

Being P the number of packages to deliver, the complexity is:

Line	Time Complexity	Space Complexity
2 - 5	**O(P)**	**O(1)**
3	O(1)	O(1)
4	O(1)	O(1)
5	O(1)	O(1)
6	O(1)	O(1)
7	According to lines 11 to 18	According to lines 11 to 18
8	O(1)	O(1)
9	O(1)	O(1)

10	**O(P)**	**O(1)**
12	O(1)	O(1)
13 - 17	**$O(P^2)$**	**O(1)**
14	O(1)	O(1)
15	O(1)	O(1)
16	O(1)	O(1)
17	O(1)	O(1)
18	O(1)	O(1)
Total	$P^2 + P + 1 + P + 1 = P^2$	1

It is important also to mention that to load the trucks with the indicated restrictions and assumptions of the problem, the time complexity varies according to the number of trucks available and the number of packages to be delivered. Being P the number of packages and T the number of trucks available, the time complexity would be **O(P*T)**

Memory allocation, on the other side, sees variation only at the beginning of the program

- When the grid of the city is created: being D the number of addresses in the city, the space complexity would be $O(D^2)$
- When the hash table of packages is created: being P the number of packages, the space complexity would be O(P)
- When loading the trucks with packages: being P the number of packages, the space complexity would be O(P)

Programming model:

A programming model refers to the style of programming where execution is invoked. In this case, to improve maintainability, Object-oriented programing is in place to manage distinct areas of the problem

- Main: To control program flow and function calls
- Restrictions: Functions associated with the loading of the truck and the priorities in order
- HashTable: Class to create a table to keep and retrieve all the packages information
- Truck: Class that keep track of truck-associated information such as velocity and packages
- Driver: Class that keep track of driver-associated information such as miles traveled and time on work
- Package: Class that keeps track of package-associated information, such as ID and address
- Graph: Class supporting CityMap
- CityMap: Based on the distances file, this class creates a grid with weighted edges of all the city points

This application is hosted on a local machine and based on 2 CSV files on the project folder (city distances and packages to deliver)

Adaptation and scalability

This program was designed to avoid any type of hard-coded information. There were some default values given based on the scenario assumptions, but they can be changed at any point (for example, we can add more trucks into the problem).

Also, it is important to mention that de CSV files attached to the project folder may be changed and the program will still work (just with the precaution of keeping the format of the current file). This is possible due to the creation of a grid for the addresses of the city and a hash table for the packages at the time that the project is run.

Nevertheless, considering that the time complexity is O(P2) and the space complexity is (D2), if the number of packages or addresses are doubled or tripled, this will necessarily imply a major time of execution and memory allocation. For example, if they are doubled, both complexities would increase 4 times. This gives a restriction on scalability and adaptation, but that shouldn't impact the company in the foreseeable future.

Efficiency and maintainability

The efficiency of the algorithm, as stated before, may not be the best due to that if you increase the number of packages by N, the time complexity will increase by 2N, this would make it very difficult for a normal computer to run if the number of packages in increased by hundreds (for example), nevertheless, the traveling salesman problem, is a known NP-hard problem, and for that, managing a large amount of packages would be very time consuming on any case. In this scenario, a big-O of O(P2) makes it efficient enough to manage the company necessities, and the scalability would be enough for the foreseeable future.

Maintainability, on the other side, is very high. The program uses different classes and well self-descriptive functions. The functions do only one thing at a time, and therefore, changing one variable or method will be reflected in the whole program, without needing any more editing. There is very little code duplication.

Data structures chosen (Justification, strengths and weaknesses)

The primary data structures implemented through the project are hash tables and classes A hash table is the main structure to hold the packages information with package ID as key for bucketing. And classes

were implemented to hold the properties of entities as truck, driver, package and graph (for the city map)

Hash tables implements an associative array abstract data type, with a hash function that determines the position (bucket) of the information to be stored. This characteristic makes hash tables on average more efficient than any other table lookup structure. This data structure was useful to hold distances or package information since it holds associative relationships, with an easy way of retrieval (in this case, address and all the distances to other addresses from there). One disadvantage thought of this data structure is the need of selection of a proper key for each value. If for any reason, this value is unknown and we still want to retrieve information with a non-key value as reference, the searching of the data can be really time and space consuming. Nevertheless, this case was not encountered through the project. Dictionary uses associative array data structure that has a complexity of O(N) on average.

Classes, on the other side, are a user-defined type of object containing groups of related variables and functions with variables closely related that are treated as one variable with multiple parts. This was extremely convenient through the project since it's a real-life situation and there were many variables that, thanks to the creation of multiples classes were easy to manage. But it is also important to mention, that, even though the implementation of classes is a widely accepted way of making code more maintainable and manageable, it may make the project look as too verbose, and creates the need to check multiple different pages to understand the logic behind the main functions, which could be intimidating for the early programmer.

Other possible Data Structures:

Other possible data structures that could have been used in this scenario would have been:

1. **Stack:** Stacks are considered as a linear data structure, where the push and pop operations occur only at one end of the structure (top of the stack). This data structure could have been used to sort the order of delivery of the packages on each truck once (by minimum distance value) and then, always work with the top of the stack (and pop the item once delivered).

   What was used in that situation was a simple loop to iterate over all the remaining packages to determine which one is closer to the current position of the truck. Which was more time consuming but with a smaller space complexity and with this the truck was able to handle a change of address at any point

2. **Binary search tree (BST):** BST is a node-based binary tree data structure with specific ordering properties that facilitates order statistics or finding closest lower and greater elements. With such characteristics, this data structure would have been the structure to choose if the amounts of data to manage on the scenario were larger.

   What was used was a hash table, this type of data structure has operations (search, deletion and insert) with a complexity of O(1) and considering the scenario presented, it was the most straightforward choice

Other possible algorithms

Other possible algorithms could have been used for the resolution of this problem:

- **Greedy Algorithm:** The Greedy heuristic gradually constructs a tour by repeatedly selecting the shortest edge and adding it to the tour as long as it doesn't create a cycle with less than N edges or increases the degree of any node to more than 2. We must not add the same edge twice of course (Nilsson)
    - Complexity: Greedy, O(n2log2(n))
    - Steps:
        1. Sort all edges.
        2. Select the shortest edge and add it to our tour if it doesn't violate any of the above constraints.
        3. Do we have N edges in our tour? If no, repeat step 2.

      (Nilsson)
    - Difference with nearest neighbor algorithm: The greedy algorithm makes a sort of all the edges to be handled prior starting to go over them.
- **Insertion Heuristics (Convex Hull):** start with a tour of a subset of all the points in the map, and then inserting the rest by some heuristic. (Nilsson)
    - Complexity: $O(n^2 \log_2(n))$
        1. Find the convex hull of our set of cities and make it our initial subtour.
        2. For each city not in the subtour, find its cheapest insertion (as in step 3 of Nearest Insertion). Then chose the city with the least cost/increase ratio and insert it.
        3. Repeat step 2 until no more cities remain

      (Nilsson)
    - Difference with convex hull algorithm: The convex hull is a type of heuristics where different paths are "tried" on a heuristic way until the shortest path is found. This differs from the nearest neighbor algorithm, where no prior selection is made

What I would do different

As long as this project was build, I did a lot of research to find the most convenient data structures to make a code that was both maintainable and scalable, but I must admit that for the sake of time I stuck with the ones that I understood the most. If I had to do this project again, I'd give some more time to this step and maybe get rid of some of the loops of the program.

Also, it is important to mention that once I got the idea of the project I dived in directly into coding and actually miss some important parts of the instructions (including the packages table) and therefore, I had a couple of days just fixing what I did that didn't align with the requirements. That said, if I had to do this project again, I'll make sure that I read and understand all the instructions before dive into the project.

Screenshots

Screenshots of Code Execution

```
"C:\Users\danie\Anaconda3\envs\WGUPS ROUTING PROGRAM\python.exe" "C:/Users/danie/OneDrive - Western Governors University/C950 Data
**********************************************************************
Hi! I can give you the packages status, you can finish by entering "Q"
Please select an option:
1 - Status of all packages
2 - Status of specific packages
Enter your selection: 1
At what time? Enter your selection in 24h format (Ex. 13:00) or EOD: eod
----
Pckg 40 (380 W 2880 S), Last status update: Delivered at 8:35:00 (Truck: 2)
Pckg 1 (195 W Oakland Ave), Last status update: Delivered at 8:38:40 (Truck: 2)
Pckg 2 (2530 S 500 E), Last status update: Delivered at 11:23:40 (Truck: 3)
Pckg 3 (233 Canyon Rd), Last status update: Delivered at 9:05:40 (Truck: 2)
Pckg 4 (380 W 2880 S), Last status update: Delivered at 11:30:40 (Truck: 3)
Pckg 5 (410 S State St), Last status update: Delivered at 11:51:00 (Truck: 3)
Pckg 6 (3060 Lester St), Last status update: Delivered at 10:25:00 (Truck: 3)
Pckg 7 (1330 2100 S), Last status update: Delivered at 11:18:20 (Truck: 3)
Pckg 8 (300 State St), Last status update: Delivered at 11:54:20 (Truck: 3)
Pckg 9 (410 S State St), Last status update: Delivered at 11:51:00 (Truck: 3)
Pckg 10 (600 E 900 South), Last status update: Delivered at 11:45:00 (Truck: 3)
Pckg 11 (2600 Taylorsville Blvd), Last status update: Delivered at 10:28:20 (Truck: 3)
Pckg 12 (3575 W Valley Central Station bus Loop), Last status update: Delivered at 10:59:20 (Truck: 3)
Pckg 13 (2010 W 500 S), Last status update: Delivered at 9:21:40 (Truck: 2)
Pckg 14 (4300 S 1300 E), Last status update: Delivered at 8:06:20 (Truck: 2)
Pckg 15 (4580 S 2300 E), Last status update: Delivered at 8:13:00 (Truck: 2)
Pckg 16 (4580 S 2300 E), Last status update: Delivered at 8:13:00 (Truck: 2)
Pckg 17 (3148 S 1100 W), Last status update: Delivered at 10:45:40 (Truck: 3)
Pckg 18 (1488 4800 S), Last status update: Delivered at 10:01:00 (Truck: 2)
Pckg 19 (177 W Price Ave), Last status update: Delivered at 10:54:40 (Truck: 3)
Pckg 20 (3595 Main St), Last status update: Delivered at 8:29:40 (Truck: 2)
Pckg 21 (3595 Main St), Last status update: Delivered at 10:53:00 (Truck: 3)
Pckg 22 (6351 South 900 East), Last status update: Delivered at 8:18:00 (Truck: 1)
Pckg 23 (5100 South 2700 West), Last status update: Delivered at 9:31:40 (Truck: 1)
Pckg 24 (5025 State St), Last status update: Delivered at 8:08:00 (Truck: 1)
Pckg 25 (5383 South 900 East #104), Last status update: Delivered at 10:01:00 (Truck: 3)
Pckg 26 (5383 South 900 East #104), Last status update: Delivered at 8:13:40 (Truck: 1)
Pckg 27 (1060 Dalton Ave S), Last status update: Delivered at 8:59:00 (Truck: 1)
Pckg 28 (2835 Main St), Last status update: Delivered at 11:27:20 (Truck: 3)
Pckg 29 (1330 2100 S), Last status update: Delivered at 8:48:00 (Truck: 2)
Pckg 30 (300 State St), Last status update: Delivered at 9:07:40 (Truck: 2)
Pckg 31 (3365 S 900 W), Last status update: Delivered at 9:46:20 (Truck: 2)
Pckg 32 (3365 S 900 W), Last status update: Delivered at 10:47:40 (Truck: 3)
Pckg 33 (2530 S 500 E), Last status update: Delivered at 8:38:00 (Truck: 1)
Pckg 34 (4580 S 2300 E), Last status update: Delivered at 8:13:00 (Truck: 2)
Pckg 35 (1060 Dalton Ave S), Last status update: Delivered at 8:59:00 (Truck: 1)
Pckg 36 (2300 Parkway Blvd), Last status update: Delivered at 9:35:00 (Truck: 2)
Pckg 37 (410 S State St), Last status update: Delivered at 9:02:20 (Truck: 2)
Pckg 38 (410 S State St), Last status update: Delivered at 9:02:20 (Truck: 2)
Pckg 39 (2010 W 500 S), Last status update: Delivered at 9:04:20 (Truck: 1)
* Driver 1 at 11:54:20, in 300 State St. Miles traveled = 70.3
* Driver 2 at 10:01:00, in 1488 4800 S. Miles traveled = 36.3

Total miles: 106.6
```

First Status Check

```
********************************************************************************
Hi! I can give you the packages status, you can finish by entering "Q"
Please select an option:
1 - Status of all packages
2 - Status of specific packages
Enter your selection: 1
At what time? Enter your selection in 24h format (Ex. 13:00) or EOD: 9:00
----
Pckg 40 (380 W 2880 S), Last status update: Delivered at 8:35:00 (Truck: 2)
Pckg 1 (195 W Oakland Ave), Last status update: Delivered at 8:38:40 (Truck: 2)
Pckg 2 (2530 S 500 E), Last status update: In Hub at 8:00:00 (Truck: 3)
Pckg 3 (233 Canyon Rd), Last status update: On Transit at 8:48:00 (Truck: 2)
Pckg 4 (380 W 2880 S), Last status update: In Hub at 8:00:00 (Truck: 3)
Pckg 5 (410 S State St), Last status update: In Hub at 8:00:00 (Truck: 3)
Pckg 6 (3060 Lester St), Last status update: Delayed in flight at 8:00:00 (Truck: 3)
Pckg 7 (1330 2100 S), Last status update: In Hub at 8:00:00 (Truck: 3)
Pckg 8 (300 State St), Last status update: In Hub at 8:00:00 (Truck: 3)
Pckg 9 (300 State St), Last status update: In Hub at 8:00:00 (Truck: 3)
Pckg 10 (600 E 900 South), Last status update: In Hub at 8:00:00 (Truck: 3)
Pckg 11 (2600 Taylorsville Blvd), Last status update: In Hub at 8:00:00 (Truck: 3)
Pckg 12 (3575 W Valley Central Station bus Loop), Last status update: In Hub at 8:00:00 (Truck: 3)
Pckg 13 (2010 W 500 S), Last status update: On Transit at 8:48:00 (Truck: 2)
Pckg 14 (4300 S 1300 E), Last status update: Delivered at 8:06:20 (Truck: 2)
Pckg 15 (4580 S 2300 E), Last status update: Delivered at 8:13:00 (Truck: 2)
Pckg 16 (4580 S 2300 E), Last status update: Delivered at 8:13:00 (Truck: 2)
Pckg 17 (3148 S 1100 W), Last status update: In Hub at 8:00:00 (Truck: 3)
Pckg 18 (1488 4800 S), Last status update: On Transit at 8:48:00 (Truck: 2)
Pckg 19 (177 W Price Ave), Last status update: In Hub at 8:00:00 (Truck: 3)
Pckg 20 (3595 Main St), Last status update: Delivered at 8:29:40 (Truck: 2)
Pckg 21 (3595 Main St), Last status update: In Hub at 8:00:00 (Truck: 3)
Pckg 22 (6351 South 900 East), Last status update: Delivered at 8:18:00 (Truck: 1)
Pckg 23 (5100 South 2700 West), Last status update: On Transit at 8:59:00 (Truck: 1)
Pckg 24 (5025 State St), Last status update: Delivered at 8:08:00 (Truck: 1)
Pckg 25 (5383 South 900 East #104), Last status update: Delayed in flight at 8:00:00 (Truck: 3)
Pckg 26 (5383 South 900 East #104), Last status update: Delivered at 8:13:40 (Truck: 1)
Pckg 27 (1060 Dalton Ave S), Last status update: Delivered at 8:59:00 (Truck: 1)
Pckg 28 (2835 Main St), Last status update: Delayed in flight at 8:00:00 (Truck: 3)
Pckg 29 (1330 2100 S), Last status update: Delivered at 8:48:00 (Truck: 2)
Pckg 30 (300 State St), Last status update: On Transit at 8:48:00 (Truck: 2)
Pckg 31 (3365 S 900 W), Last status update: On Transit at 8:48:00 (Truck: 2)
Pckg 32 (3365 S 900 W), Last status update: Delayed in flight at 8:00:00 (Truck: 3)
Pckg 33 (2530 S 500 E), Last status update: Delivered at 8:38:00 (Truck: 1)
Pckg 34 (4580 S 2300 E), Last status update: Delivered at 8:13:00 (Truck: 2)
Pckg 35 (1060 Dalton Ave S), Last status update: Delivered at 8:59:00 (Truck: 1)
Pckg 36 (2300 Parkway Blvd), Last status update: On Transit at 8:48:00 (Truck: 2)
Pckg 37 (410 S State St), Last status update: On Transit at 8:48:00 (Truck: 2)
Pckg 38 (410 S State St), Last status update: On Transit at 8:48:00 (Truck: 2)
Pckg 39 (2010 W 500 S), Last status update: On Transit at 8:59:00 (Truck: 1)
********************************************************************************
```

Second Status Check

```
**************************************************************************
Hi! I can give you the packages status, you can finish by entering "Q"
Please select an option:
1 - Status of all packages
2 - Status of specific packages
Enter your selection: 1
At what time? Enter your selection in 24h format (Ex. 13:00) or EOD: 10:00
----
Pckg 40 (380 W 2880 S), Last status update: Delivered at 8:35:00 (Truck: 2)
Pckg 1 (195 W Oakland Ave), Last status update: Delivered at 8:38:40 (Truck: 2)
Pckg 2 (2530 S 500 E), Last status update: In Hub at 8:00:00 (Truck: 3)
Pckg 3 (233 Canyon Rd), Last status update: Delivered at 9:05:40 (Truck: 2)
Pckg 4 (380 W 2880 S), Last status update: In Hub at 8:00:00 (Truck: 3)
Pckg 5 (410 S State St), Last status update: In Hub at 8:00:00 (Truck: 3)
Pckg 6 (3060 Lester St), Last status update: On Transit at 9:53:00 (Truck: 3)
Pckg 7 (1330 2100 S), Last status update: In Hub at 8:00:00 (Truck: 3)
Pckg 8 (300 State St), Last status update: In Hub at 8:00:00 (Truck: 3)
Pckg 9 (300 State St), Last status update: In Hub at 8:00:00 (Truck: 3)
Pckg 10 (600 E 900 South), Last status update: In Hub at 8:00:00 (Truck: 3)
Pckg 11 (2600 Taylorsville Blvd), Last status update: In Hub at 8:00:00 (Truck: 3)
Pckg 12 (3575 W Valley Central Station bus Loop), Last status update: In Hub at 8:00:00 (Truck: 3)
Pckg 13 (2010 W 500 S), Last status update: Delivered at 9:21:40 (Truck: 2)
Pckg 14 (4300 S 1300 E), Last status update: Delivered at 8:06:20 (Truck: 2)
Pckg 15 (4580 S 2300 E), Last status update: Delivered at 8:13:00 (Truck: 2)
Pckg 16 (4580 S 2300 E), Last status update: Delivered at 8:13:00 (Truck: 2)
Pckg 17 (3148 S 1100 W), Last status update: In Hub at 8:00:00 (Truck: 3)
Pckg 18 (1488 4800 S), Last status update: On Transit at 9:46:20 (Truck: 2)
Pckg 19 (177 W Price Ave), Last status update: In Hub at 8:00:00 (Truck: 3)
Pckg 20 (3595 Main St), Last status update: Delivered at 8:29:40 (Truck: 2)
Pckg 21 (3595 Main St), Last status update: In Hub at 8:00:00 (Truck: 3)
Pckg 22 (6351 South 900 East), Last status update: Delivered at 8:18:00 (Truck: 1)
Pckg 23 (5100 South 2700 West), Last status update: Delivered at 9:31:40 (Truck: 1)
Pckg 24 (5025 State St), Last status update: Delivered at 8:08:00 (Truck: 1)
Pckg 25 (5383 South 900 East #104), Last status update: On Transit at 9:53:00 (Truck: 3)
Pckg 26 (5383 South 900 East #104), Last status update: Delivered at 8:13:40 (Truck: 1)
Pckg 27 (1060 Dalton Ave S), Last status update: Delivered at 8:59:00 (Truck: 1)
Pckg 28 (2835 Main St), Last status update: Delayed in flight at 8:00:00 (Truck: 3)
Pckg 29 (1330 2100 S), Last status update: Delivered at 8:48:00 (Truck: 2)
Pckg 30 (300 State St), Last status update: Delivered at 9:07:40 (Truck: 2)
Pckg 31 (3365 S 900 W), Last status update: Delivered at 9:46:20 (Truck: 2)
Pckg 32 (3365 S 900 W), Last status update: Delayed in flight at 8:00:00 (Truck: 3)
Pckg 33 (2530 S 500 E), Last status update: Delivered at 8:38:00 (Truck: 1)
Pckg 34 (4580 S 2300 E), Last status update: Delivered at 8:13:00 (Truck: 2)
Pckg 35 (1060 Dalton Ave S), Last status update: Delivered at 8:59:00 (Truck: 1)
Pckg 36 (2300 Parkway Blvd), Last status update: Delivered at 9:35:00 (Truck: 2)
Pckg 37 (410 S State St), Last status update: Delivered at 9:02:20 (Truck: 2)
Pckg 38 (410 S State St), Last status update: Delivered at 9:02:20 (Truck: 2)
Pckg 39 (2010 W 500 S), Last status update: Delivered at 9:04:20 (Truck: 1)
**************************************************************************
```

Third Status Check

```
**************************************************************************
Hi! I can give you the packages status, you can finish by entering "Q"
Please select an option:
1 - Status of all packages
2 - Status of specific packages
Enter your selection: 1
At what time? Enter your selection in 24h format (Ex. 13:00) or EOD: 12:30
----
Pckg 40 (380 W 2880 S), Last status update: Delivered at 8:35:00 (Truck: 2)
Pckg 1 (195 W Oakland Ave), Last status update: Delivered at 8:38:40 (Truck: 2)
Pckg 2 (2530 S 500 E), Last status update: Delivered at 11:23:40 (Truck: 3)
Pckg 3 (233 Canyon Rd), Last status update: Delivered at 9:05:40 (Truck: 2)
Pckg 4 (380 W 2880 S), Last status update: Delivered at 11:30:40 (Truck: 3)
Pckg 5 (410 S State St), Last status update: Delivered at 11:51:00 (Truck: 3)
Pckg 6 (3060 Lester St), Last status update: Delivered at 10:25:00 (Truck: 3)
Pckg 7 (1330 2100 S), Last status update: Delivered at 11:18:20 (Truck: 3)
Pckg 8 (300 State St), Last status update: Delivered at 11:54:20 (Truck: 3)
Pckg 9 (410 S State St), Last status update: Delivered at 11:51:00 (Truck: 3)
Pckg 10 (600 E 900 South), Last status update: Delivered at 11:45:00 (Truck: 3)
Pckg 11 (2600 Taylorsville Blvd), Last status update: Delivered at 10:28:20 (Truck: 3)
Pckg 12 (3575 W Valley Central Station bus Loop), Last status update: Delivered at 10:59:20 (Truck: 3)
Pckg 13 (2010 W 500 S), Last status update: Delivered at 9:21:40 (Truck: 2)
Pckg 14 (4300 S 1300 E), Last status update: Delivered at 8:06:20 (Truck: 2)
Pckg 15 (4580 S 2300 E), Last status update: Delivered at 8:13:00 (Truck: 2)
Pckg 16 (4580 S 2300 E), Last status update: Delivered at 8:13:00 (Truck: 2)
Pckg 17 (3148 S 1100 W), Last status update: Delivered at 10:45:40 (Truck: 3)
Pckg 18 (1488 4800 S), Last status update: Delivered at 10:01:00 (Truck: 2)
Pckg 19 (177 W Price Ave), Last status update: Delivered at 10:54:40 (Truck: 3)
Pckg 20 (3595 Main St), Last status update: Delivered at 8:29:40 (Truck: 2)
Pckg 21 (3595 Main St), Last status update: Delivered at 10:53:00 (Truck: 3)
Pckg 22 (6351 South 900 East), Last status update: Delivered at 8:18:00 (Truck: 1)
Pckg 23 (5100 South 2700 West), Last status update: Delivered at 9:31:40 (Truck: 1)
Pckg 24 (5025 State St), Last status update: Delivered at 8:08:00 (Truck: 1)
Pckg 25 (5383 South 900 East #104), Last status update: Delivered at 10:01:00 (Truck: 3)
Pckg 26 (5383 South 900 East #104), Last status update: Delivered at 8:13:40 (Truck: 1)
Pckg 27 (1060 Dalton Ave S), Last status update: Delivered at 8:59:00 (Truck: 1)
Pckg 28 (2835 Main St), Last status update: Delivered at 11:27:20 (Truck: 3)
Pckg 29 (1330 2100 S), Last status update: Delivered at 8:48:00 (Truck: 2)
Pckg 30 (300 State St), Last status update: Delivered at 9:07:40 (Truck: 2)
Pckg 31 (3365 S 900 W), Last status update: Delivered at 9:46:20 (Truck: 2)
Pckg 32 (3365 S 900 W), Last status update: Delivered at 10:47:40 (Truck: 3)
Pckg 33 (2530 S 500 E), Last status update: Delivered at 8:38:00 (Truck: 1)
Pckg 34 (4580 S 2300 E), Last status update: Delivered at 8:13:00 (Truck: 2)
Pckg 35 (1060 Dalton Ave S), Last status update: Delivered at 8:59:00 (Truck: 1)
Pckg 36 (2300 Parkway Blvd), Last status update: Delivered at 9:35:00 (Truck: 2)
Pckg 37 (410 S State St), Last status update: Delivered at 9:02:20 (Truck: 2)
Pckg 38 (410 S State St), Last status update: Delivered at 9:02:20 (Truck: 2)
Pckg 39 (2010 W 500 S), Last status update: Delivered at 9:04:20 (Truck: 1)
**************************************************************************
```

References

McGeoch, D. J. (1995). *The Traveling Salesman Problem: A Case Study in Local Optimization.*

Nilsson, C. (n.d.). *Heuristics for the Traveling Salesman Problem.* Retrieved from http://160592857366.free.fr/joe/ebooks/ShareData/Heuristics%20for%20the%20Traveling%20Salesman%20Problem%20By%20Christian%20Nillson.pdf