APURBA POKHAREL
11627243

The github link to my project repo is here.
(https://github.com/apurbapokharel/CSCE-5210/tree/main/Assignment%201)
The google collab link is here.
(https://colab.research.google.com/drive/1Z8H0YZZjBF0hvLRhoNZPdqNOuyBjQBsn#scrollTo=ER_nHRYNEd6d)

My approach and Additional Assumption:

OOP is something that I have a nice understanding of. So, my code is based on the ideas of classes and objects. I find it easy to work with classes and also, having a different class for each of the things like graph, customer, car, agent made writing the code easier and gave a structure to my code.
The classes that I have used and the main functionality of them are defined below:

1. Graph

- Generates the graph
- Computes the Astar path length as well as determines the shortest Astar path
- Can plot the graph if visualization is needed.
- The graph code is referred from Dr. Russel's tutorial 1 with slight changes as necessary.

2. Car

- Has methods to handle customer pickup requests, pickup customer, drop off customer.
- Stores the distance and trip for each car object.
- The information about capacity is stored here in the class.

3. Customer

- Has just the pickup and drop off node info (a randomly generated info) stored here.

4. Agent

- The brains of the entire operation.
- Has an array that stores all the cars and customer objects.
- An index (the actual array index) is used to refer to these cars and customers in the codebase.
  For Example: a car at index 0 in the car_array will be referred to by index 0 all over the code.
  Similarly, customer has the same rule.
- The Car object creation and customer object creation are done by the agent as specified by us in the main function.

- The request for picking up new customers, and selecting a car based on the shortest distance as well as the current capacity is handled here by the agent.
- The process of updating the wait queue based on the distance to the nearest customer is done here as well.
- Picking up and dropping off the customer is done by the agent.

The scheduling algorithm is the core of the code. Once that is figured out the rest of the code worked out on it's own.
Since, the scheduling algorithm works in a queue based manner. The data type that I used an array, had to have a queue based implementation for adding new customers and serving customers in the 0^{th} index of the array. So, this queue based approach can be seen in my code.

Some important consideration for my code:

1. As per Dr. Russel's tutorial 2 (clock tick 3), if two or more customers share the same pickup points then they need to be picked up together as long as the space is available.

2. The same goes for dropoff, if the currently being served customer shares dropoff with two or more customers (that are already picked up) then they need to be dropped off together.

Additional assumptions:

1. My program does not show the currently serving customer in the wait queue, I use something called a current_serving_customer to keep track of which customer needs to be picked up and dropped off.

2. Instead of having s1={(id1,p,8),(id1,d,9)} as service queue for tick 1, this program uses index of customer(starting from 0) like [0] for service queue in tick1.

   Additionaly for clock tick 3 instead of using notation like {(id1,d,9), (id3,p,4),(id3,d,7),(id5,p,1),(id6,p,1),(id5,d,7),(id6,d,9),(id4,p,2),(id4,d,4)}
   I use this [1,3,5,6,4]

   The index of customer can be used to get their pickup and drop off nodes. So, I don't store them.

3. The service queue is only updated and printed if there is request for customer.

4. Program counts customer and cars from 0 not 1.

5. Clock tick starts in 0 not 1.

The answers for each of the requirements are:

1. For R3

The number of nodes = 100
The number of cars = 30
The connectivity = 3
The reservation per hour = 600

a. the average distance traveled (over the fleet) per day when run on a road network of 100 nodes and average connectivity of 3?

   29.07

b. the average number of trips (over the fleet) per day when run on a road network of 100 nodes and average connectivity of 3?

   19.63

2. For R4

   The number of nodes = 100
   The number of cars = 60
   The connectivity = 3
   The reservation per hour = 600

   a. the average distance traveled (over the fleet) per day when run on a road network of 100 nodes and average connectivity of 3?

      16.29

   b. the average number of trips (over the fleet) per day when run on a road network of 100 nodes and average connectivity of 3?

      9.83

3. For R5

   The number of nodes = 100
   The number of cars = 60
   The connectivity = 4
   The reservation per hour = 600

   a. the average distance traveled (over the fleet) per day when run on a road network of 100 nodes and average connectivity of 3?

      14.346666666666671

   Explanation: The distance travelled decreases because there are more paths for the cars to take from one node to the other. Granted the decrease is not always the case since the values

of the edges are generated in random. But for a graph with 100 nodes increasing connectivity from 3 to 4 decreases the distance travelled in most cases. Though in some cases the distance travelled was almost equal or even more than that of R4 but that is solely due to the randomness that is involved in assigning edges to the nodes.