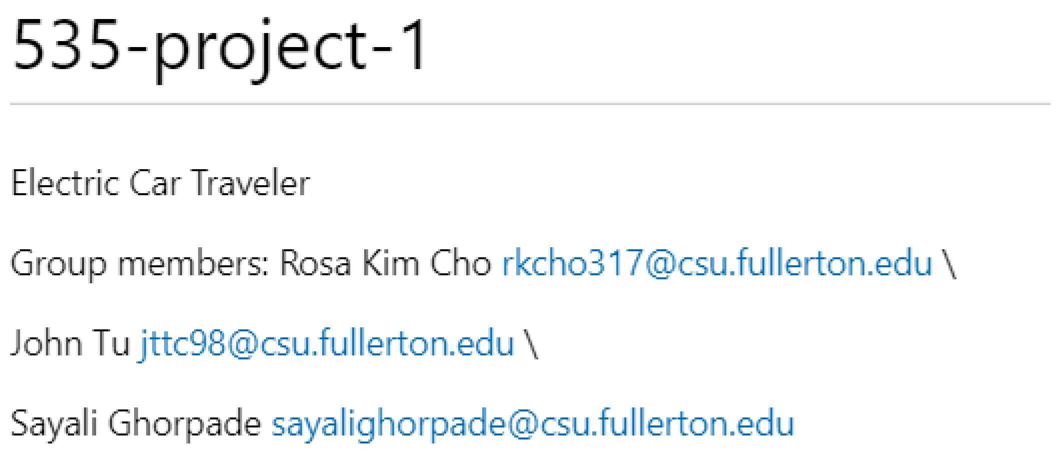
CPSC 535 Project 1 Report

<u>Group Members</u>

John Tu (jttc98@csu.fullerton.edu)

Rosa Cho (rkcho317@csu.fullerton.edu)

Sayali Ghorpade (sayalighorpade@csu.fullerton.edu)



535-project-1

Electric Car Traveler

Group members: Rosa Kim Cho rkcho317@csu.fullerton.edu \

John Tu jttc98@csu.fullerton.edu \

Sayali Ghorpade sayalighorpade@csu.fullerton.edu

Pseudocode

```
ElectricCarTraveler(C, n):
    If C < 250 OR C > 350:
        Print "Invalid capacity."
        Exit
    If n <= 3 OR n >= 20:
        Print "Invalid number of cities."
        Exit
    Let city_list, dist_list, output_list = [], [], []
    For x=0 to n do
        Let city_name be string variable
        Get user input for city_name
        Append city_name to city_list
    For y=0 to (n-1) do
        Let city_dist be int variable
        Get user input for city_dist
        Append city_dist to dist_list
    For z=0 to (n-1) do
        If dist_list[z] <= 10 OR dist_list[z] >= (C/2):
            Print “Invalid distance between two cities.”
            Exit
    For w=0 to (n-1) do
        Print “Distance between City {} and City {}: {}”
            .format(city_list[w], city_list[w+1], dist_list[w])
    Let final_destination = false
    Let refuel = C
    Let current_location_number = 0
    output_list.append(city_list[0])
    While (final_destination != true)
        For v=current_location_number to (n-1) do
            C = C - dist_list[v]
            If C < dist_list[v]:
                Print “Stopping to refuel.”
                output_list.append(city_list[v])
                Let C = refuel
                Let current_location_number = v
                Break
            If city_list[v+1] == city_list[len(city_list)-1]:
                output_list.append(city_list[v+1])
                Let final_destination = true
    Print “Destination arrived.”
    Return output_list

Let mileage and num_cities be integer variables
Get user input for mileage and num_cities
Call ElectricCarTraveler(mileage, num_cities)
```

How to run the code

(Command Prompt):

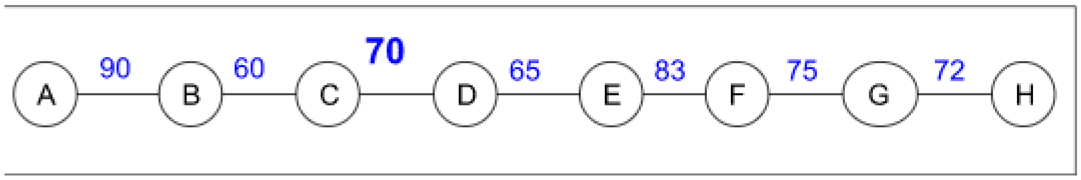
1. Download the java file from the GitHub repository.
2. Open command prompt and use cd to switch to the directory where the java file is saved.
3. Type javac followed by the filename of the Java file to compile the code.
4. Type java followed by the filename of the Java file to execute the code.

(Visual Studio Code):

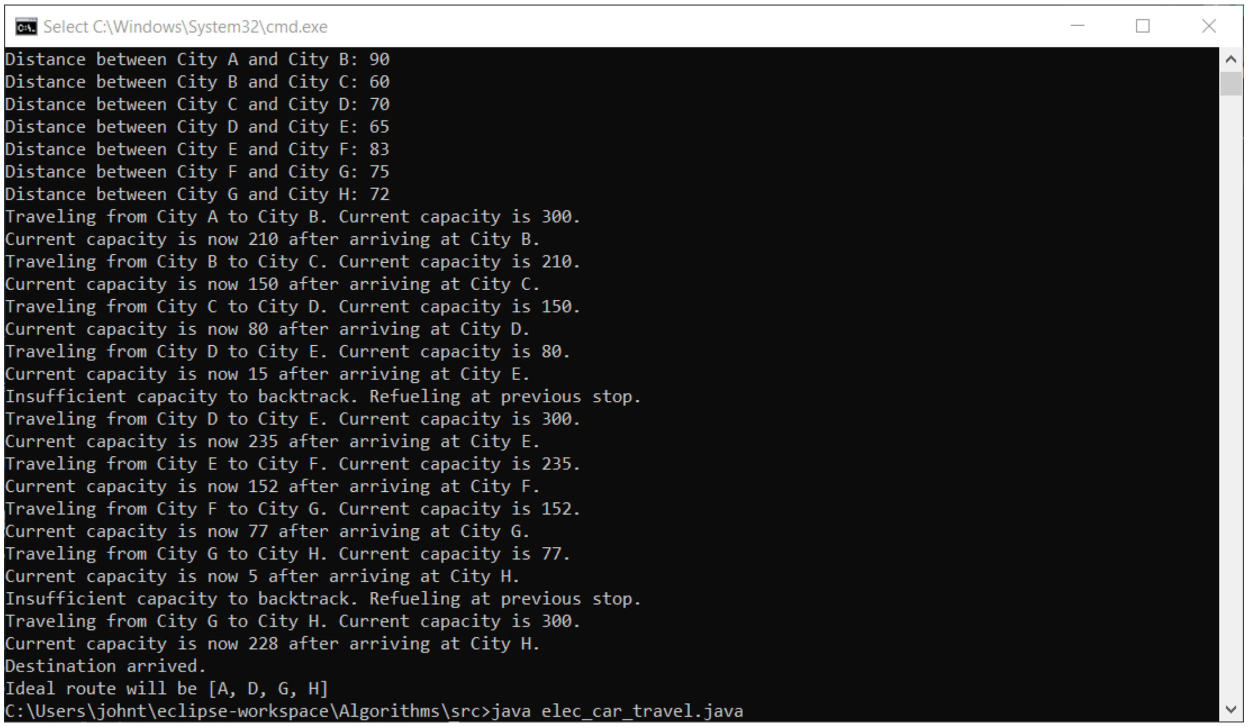
1. Download the java file from the GitHub repository and place it in the directory with the Visual Studio Code.
2. Run the code.

Screenshots

Sample Input 1: C = 300, 8 cities, Starting City = A, Destination City = H

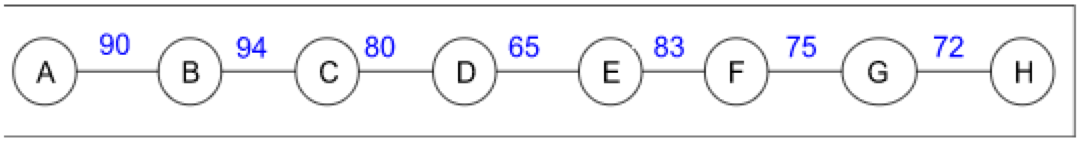


Output for Sample 1 (expected output is {A, D, G, H}):

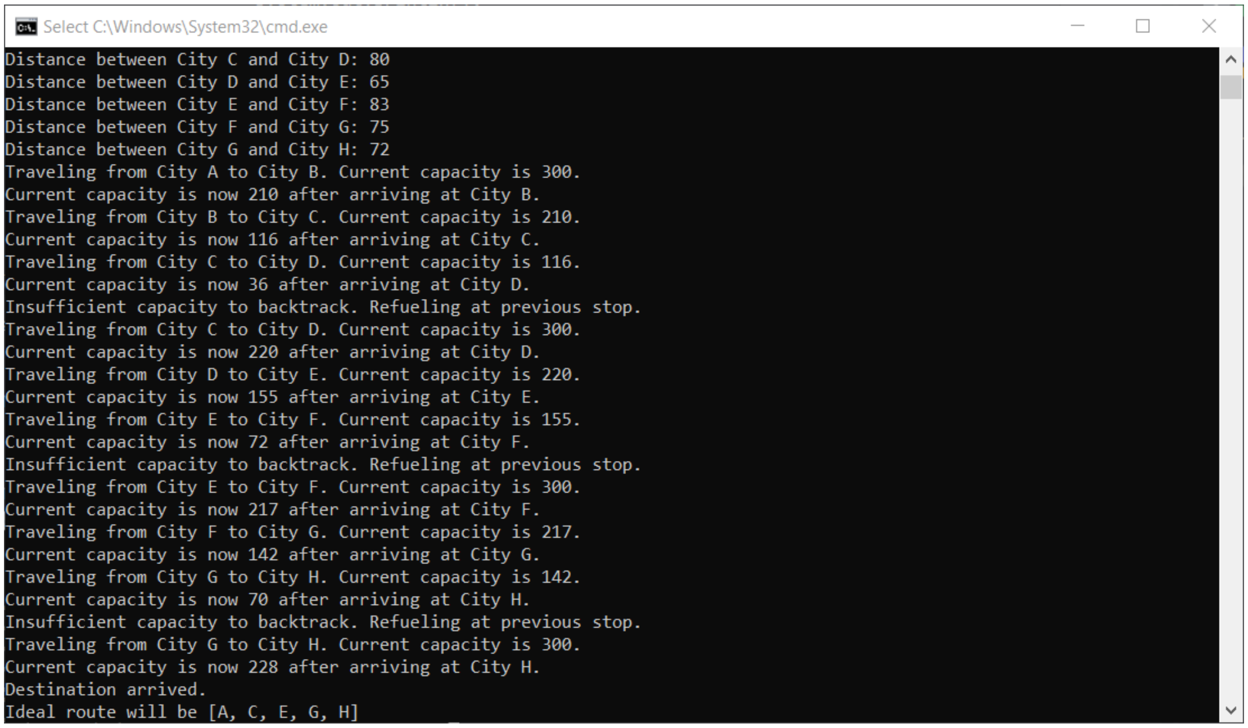


```
Distance between City A and City B: 90
Distance between City B and City C: 60
Distance between City C and City D: 70
Distance between City D and City E: 65
Distance between City E and City F: 83
Distance between City F and City G: 75
Distance between City G and City H: 72
Traveling from City A to City B. Current capacity is 300.
Current capacity is now 210 after arriving at City B.
Traveling from City B to City C. Current capacity is 210.
Current capacity is now 150 after arriving at City C.
Traveling from City C to City D. Current capacity is 150.
Current capacity is now 80 after arriving at City D.
Traveling from City D to City E. Current capacity is 80.
Current capacity is now 15 after arriving at City E.
Insufficient capacity to backtrack. Refueling at previous stop.
Traveling from City D to City E. Current capacity is 300.
Current capacity is now 235 after arriving at City E.
Traveling from City E to City F. Current capacity is 235.
Current capacity is now 152 after arriving at City F.
Traveling from City F to City G. Current capacity is 152.
Current capacity is now 77 after arriving at City G.
Traveling from City G to City H. Current capacity is 77.
Current capacity is now 5 after arriving at City H.
Insufficient capacity to backtrack. Refueling at previous stop.
Traveling from City G to City H. Current capacity is 300.
Current capacity is now 228 after arriving at City H.
Destination arrived.
Ideal route will be [A, D, G, H]
C:\Users\johnt\eclipse-workspace\Algorithms\src>java elec_car_travel.java
```

Sample Input 2: C = 300, 8 cities, Starting City = A, Destination City = H

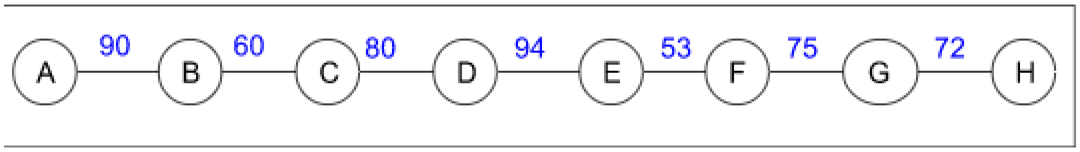


Output for Sample 2 (expected output is {A, C, E, G, H}):

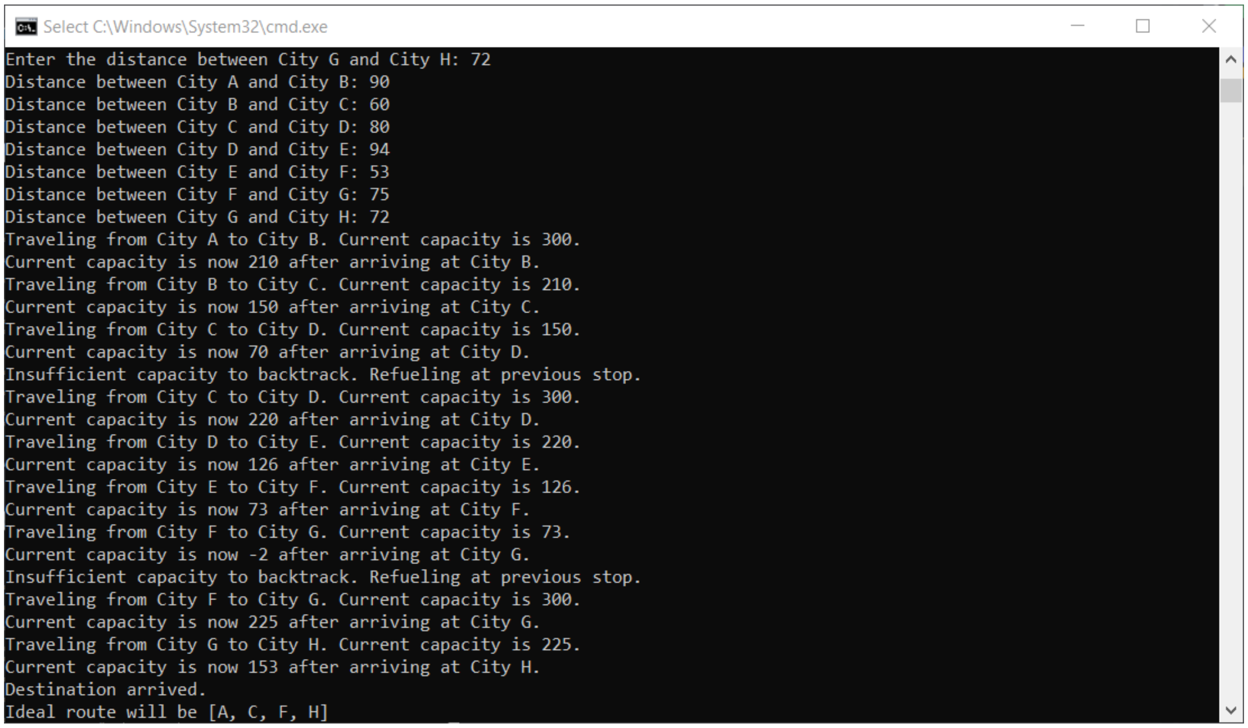

Select C:\Windows\System32\cmd.exe

```
Distance between City C and City D: 80
Distance between City D and City E: 65
Distance between City E and City F: 83
Distance between City F and City G: 75
Distance between City G and City H: 72
Traveling from City A to City B. Current capacity is 300.
Current capacity is now 210 after arriving at City B.
Traveling from City B to City C. Current capacity is 210.
Current capacity is now 116 after arriving at City C.
Traveling from City C to City D. Current capacity is 116.
Current capacity is now 36 after arriving at City D.
Insufficient capacity to backtrack. Refueling at previous stop.
Traveling from City C to City D. Current capacity is 300.
Current capacity is now 220 after arriving at City D.
Traveling from City D to City E. Current capacity is 220.
Current capacity is now 155 after arriving at City E.
Traveling from City E to City F. Current capacity is 155.
Current capacity is now 72 after arriving at City F.
Insufficient capacity to backtrack. Refueling at previous stop.
Traveling from City E to City F. Current capacity is 300.
Current capacity is now 217 after arriving at City F.
Traveling from City F to City G. Current capacity is 217.
Current capacity is now 142 after arriving at City G.
Traveling from City G to City H. Current capacity is 142.
Current capacity is now 70 after arriving at City H.
Insufficient capacity to backtrack. Refueling at previous stop.
Traveling from City G to City H. Current capacity is 300.
Current capacity is now 228 after arriving at City H.
Destination arrived.
Ideal route will be [A, C, E, G, H]
```

Sample Input 3: C = 300, 8 cities, Starting City = A, Destination City = H



Output for Sample 3 (expected output is {A, C, F, H}):


Select C:\Windows\System32\cmd.exe

```
Enter the distance between City G and City H: 72
Distance between City A and City B: 90
Distance between City B and City C: 60
Distance between City C and City D: 80
Distance between City D and City E: 94
Distance between City E and City F: 53
Distance between City F and City G: 75
Distance between City G and City H: 72
Traveling from City A to City B. Current capacity is 300.
Current capacity is now 210 after arriving at City B.
Traveling from City B to City C. Current capacity is 210.
Current capacity is now 150 after arriving at City C.
Traveling from City C to City D. Current capacity is 150.
Current capacity is now 70 after arriving at City D.
Insufficient capacity to backtrack. Refueling at previous stop.
Traveling from City C to City D. Current capacity is 300.
Current capacity is now 220 after arriving at City D.
Traveling from City D to City E. Current capacity is 220.
Current capacity is now 126 after arriving at City E.
Traveling from City E to City F. Current capacity is 126.
Current capacity is now 73 after arriving at City F.
Traveling from City F to City G. Current capacity is 73.
Current capacity is now -2 after arriving at City G.
Insufficient capacity to backtrack. Refueling at previous stop.
Traveling from City F to City G. Current capacity is 300.
Current capacity is now 225 after arriving at City G.
Traveling from City G to City H. Current capacity is 225.
Current capacity is now 153 after arriving at City H.
Destination arrived.
Ideal route will be [A, C, F, H]
```