Issue Date: Monday – March 27th, 2023
Submission Deadline: Friday – March 31st, 2023 (Till 11:59 PM)

Instructions!

- You must complete all tasks individually. Absolutely NO collaboration is allowed. Any case of plagiarism/cheating would result in 0 marks in sessional activity.
- Indent your code properly.
- Use meaningful variable and function names. Use the camelCase notation, do proper commenting (follow all coding conventions).
- Zip your homework, file name must be in proper format 'yourROLL#_H01'
- Submit your homework at following addresses, with subject line must be 'yourROLL#_Homework_01':

  For morning students: webcsf20m@gmail.com
- Homeworks not following the above instructions or received after deadline will not be acceptable at all.

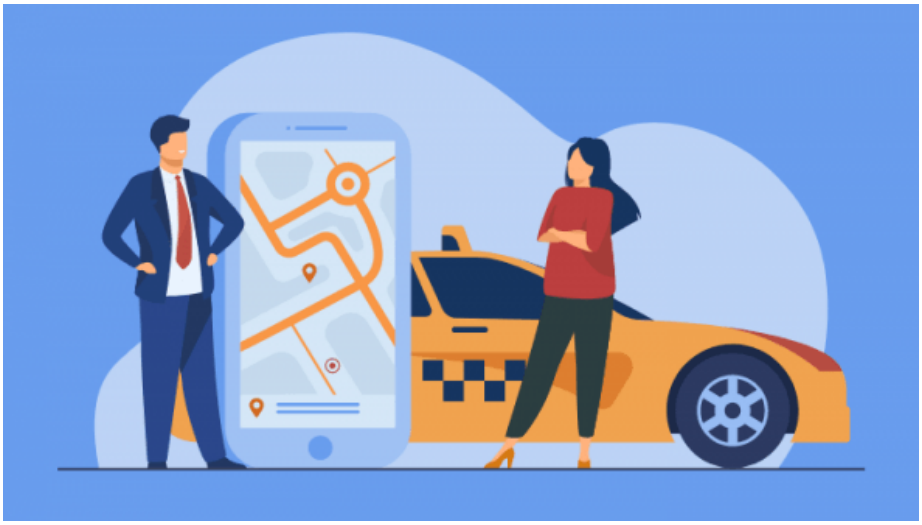
Note: Make sure to attempt this Homework properly as it will be a part of upcoming homeworks.

MYRIDE

(A Ride Hailing Application)

Inspiration:

All of you must be familiar with the famous app Uber. From its inception in 2008, humanity has become dependent on this ride hailing app. But imagine, if one day, Uber stops working and we are left deprived of its valuable services, what would we do? Well, obviously we will start using Careem or some other ride application. But that still doesn't stop me from giving you this Homework :) So, we are going to develop a ride hailing application named as 'MYRIDE'. Hopefully, it will be much more interesting for you.



Problem Statement:

Problem statement is quite simple, you have to implement a ride hailing application just like Uber, Careem, in-drive, bykea etc. Users of this application are passengers who want to book a ride, drivers whose vehicles are registered in this application and an admin who registers the drivers on application. You'll come to know about the details later in this handout.

Classes

Following is the set of classes to be made in order to develop this application. You need to make a separate class library for each class and demonstrate the working of all projects together.

Note: Make properties for all attributes of each class.

Class: Driver

- **Attributes:**
  - **name (str):** Name of driver
  - **age (int):** Age of driver
  - **gender (str):** Gender of driver
  - **address (str):** Address of driver
  - **phone no (str):** Phone no of driver (Make sure all are digits, no slash '-' or space ' ' etc. are allowed)
  - **curr_location (location):** Current location of the driver
  - **vehicle (Vehicle):** The vehicle used by the driver
  - **rating (list):** The list of rating to driver, based on passengers' feedback
  - **availability (bool):** whether the driver is currently available to take a ride

- **Functions:**
  - **updateAvailability():** In this function driver will be asked to change his availability status to true or false.
  - **getRating():** Returns the average rating of the driver
  - **updateLocation():** In this function driver will be asked to change his current location.

Class: Passenger

- **Attributes:**
  - **name (str):** The name of the passenger
  - **phone no (str):** Phone number of the passenger
- **Functions:**
  - **bookRide():** This function will book a new ride for the passenger. It will ask for ride details from passenger i.e, starting location, ending location etc.
  - **giveRating():** Passenger will give rating from 1 to 5 for current ride. No other digit acceptable.

Class: Ride

- **Attributes:**
  - **start_location (location):** The starting location of the ride
  - **end_location (location):** The destination of the ride
  - **price (int):** The price of the ride
  - **driver (Driver):** The driver assigned to the ride
  - **passenger (Passenger):** The passenger who booked the ride
- **Functions:**
  - **assignPassenger():** This function will assign passenger for this ride. You will ask for the passenger object and set it to the passenger object
  - **assignDriver():** This function will assign a driver to the ride.
    - You need to consider two factors in assigning the driver:
      1. Driver must be available
      2. You have to choose the driver that is closest to the starting location of passenger, you can do it by checking for smallest distance (euclidean distance between starting location and driver's location)
  - **getLocations():** In this function you will set start and end locations of the ride
  - **calculatePrice():** In this function you will calculate the price of the ride

- To calculate the price use the following formula and Table:

  Price = ((Distance*fuel_price)/fuel_average) + Commission

- For now, you can hardcode fuel_price (petrol price per liter)
- Distance is the Euclidean distance between start and end location

Vehicle Type	Fuel Average	Company Commission
Bike	50 km / liter	5 %
Rickshaw	35 km / liter	10 %
Car	15 km / liter	20 %

Class: Vehicle

- **Attributes:**
  - **Type (str):** Vehicle type (Car, Bike, Rickshaw)
  - **model (str):** the model of the vehicle
  - **license_plate (str):** the license plate of the vehicle

Class: Location

- **Attributes:**
  - **latitude (float):** latitude of the location
  - **longitude (float):** longitude of the location
- **Functions:**
  - **setLocation(longitude, latitude):** This function will set longitude and latitude

Class: Admin

- **Attributes:**
  - **List of Drivers (List or Arraylist):** This list will contain all the registered drivers
- **Functions:**
  - **addDriver():** This function will add a new driver to the driver's list of the company. You will take input all the details of driver and his/her vehicle and add the driver object to the list.
  - **updateDriver():** This function will update an existing driver in the driver's list, based on the fields that admin wants to update.

- **Remove Driver():** This function will ask for the driver ID to be removed and will remove the driver from list if available.
- **Search Driver():** This function will search a particular driver or drivers based on the search parameters given by admin and will display in the form of a table.

User Interfaces

You are not put in the trouble of creating some fancy GUIs at this moment. What you are supposed to do is to develop a simple console-based system. When the system starts the user is presented with the main menu as follows:

Main Menu:

WELCOME TO MYRIDE

---

1. Book a Ride
2. Enter as Driver
3. Enter as Admin

Press 1 to 3 to select an option:

Upon selecting 1 'Book a Ride' menu will be opened:

Book a Ride

Enter Name: Ahmad

Enter Phone no: 03123456789

Enter Start Location: 1,2

Enter End Location: 5,6

Enter Ride Type: Car

---------- THANK YOU ----------

Price for this ride is: 200

Enter 'Y' if you want to Book the ride, enter 'N' if you want to cancel operation: Y

Happy Travel…!

Give rating out of 5: 4

Upon selecting 2 'Enter as Driver' menu will be opened:

Enter as Driver

(In the beginning, the driver will be asked for his ID and Name to check if the driver is registered or not by the admin).

Enter ID: 121

Enter Name: Ali

If the driver is registered, the following message will be displayed and asked to enter the current location:

Hello Ali!

Enter your current Location: 1,2

If the driver is not registered, show the error message and go back to the main menu.

If driver is found successfully and current location is collected following options will be displayed to the driver:

1. Change availability (upon selecting 1, diver will be asked to update his availability to Available/Unavailable)
2. Change Location (upon selecting 2, driver will be asked to update his Location)
3. Exit as Driver (upon selecting 3, driver will be redirected to Main Menu)

Upon selecting 3 'Enter as Admin' menu will be opened:

Enter as Admin:

Upon entering as admin following menu will be displayed:

1. Add Driver
2. Remove Driver
3. Update Driver
4. Search Driver
5. Exit as Admin

Upon selecting 1 Admin will be asked to Add Driver:

Enter Name: Mahmood

Enter Age: 31

Enter Gender: Male

Enter Address: Edward Road, Lahore

Enter Vehicle Type: Bike

Enter Vehicle Model: 2010

Enter Vehicle License Plate: LCO 1288

After getting all these details, the Driver object will be created and stored in the list.

Upon selecting 2 Admin will be asked to remove Driver:

Admin will be asked for Driver ID if driver with such ID exists remove it from the list else display the error message.

Upon selecting 3 Admin will be asked to Update the Driver:

Admin will be asked for Driver ID. If driver with ID exists, display a message and enter the data in fields that you want to update and leave other fields empty (your program should not crash upon leaving fields empty).

Enter Driver ID: 121

----------Driver with ID 121 exists----------

Enter Name:

Enter Age: 25

Enter Gender:

Enter Address:

Enter Vehicle Type: Bike

Enter Vehicle Model: 2013

Enter Vehicle License Plate: LPE 1989

(Note above that we didn't update Name, Gender and Address of driver)

After Collecting the information, update the driver. You should update only those attributes of driver that we wanted to update (i.e, that were not empty).

Upon selecting 4 Admin will be asked to Search the Driver:

Admin will enter the data in fields according he want to search and left other fields empty.

Enter Driver ID: 121

Enter Name:

Enter Age: 25

Enter Gender:

Enter Address:

Enter Vehicle Type: Bike

Enter Vehicle Model: 2013

Enter Vehicle License Plate: LPE 1989

After Collecting the information show the search results as following:

Name	Age	Gender	V.Type	V.Model	V.License

Ali	25	Male	Bike	2013	LPE 1989

Upon selecting 5 Admin will exit to the main menu.

Bonus Task:

Make the color of console inputs ‘Green’ as shown above. You may Google to check for it.

GOOD LUCK!