EC2492 : Object Oriented Programming : Final Project / Assignment:
2nd Year semester1 2021

Reason for this Assignment:

The aim of this assignment is to assess your current object-oriented programming skills. You will be asked to implement a program in which objects interact in order to comply with a set of functional requirements.
In this assignment the first task you should perform is a careful analysis of the problem statement in order to make sure you have all the information to construct a solution. Do not make assumptions about what is needed! If you are not sure about the information provided, ask questions. Work on this task by yourself, but you are encouraged to speak to others. You are not allowed to borrow their code. You may ask questions via the WhatsApp group created for OOP or via e-mail.

- Task 1: **Design a solution and create a simple design document: (15 marks)**
  The design of your system/solution should be consistent with Object Oriented principles, and should be easy to understand (by another programmer). You are required to document your program design using UML diagrams. In particular you have to draw a class diagram of your design.
  Use a Class Diagram drawing tool (use https://www.draw.io if you wish)

- Task 2**: Code the design. (50 marks)**
  Use a Java IDE such as eclipse to code your solution. The code you write needs to be added to the above design document, and also submitted via courseweb.
  Deliverable 1: Submit a Document via Courseweb that includes the work you did for Task1 and Task2 above. Part of your grade will for this document.

- Task 3: **Submit a 3 minute video demo of your code.**
  For this task you need to run through some reasonable test cases and show that your code works. (Task 1 + Task 2 + Task3 = 70 marks) The document that you create for Task1 and Task2 needs to include your design document (Task- 1) and code listings (Task-2). This document needs to be submitted in .pdf format via courseweb. (due dates will be notified via courseweb)

- Task 4**. Participate in a one-on-one online mini viva of about 3-5**
  You maybe asked to demo your code again during the mini-viva/Demo. You will also be asked questions from the code you wrote.

  Please Note: if we think your knowledge of the subject matter based on your submissions are not reflected in the demo/mini-viva, we will modify the grades for Task1, Task2)

Task3 and Task4 will have a combined 35 Marks. The Video and the Viva will have at least 10 marks each. If the Video is clear the viva will be short. Try not to exceed 4 minutes for the video. A 3 minute video is requested.

Problem description and requirements statement:

You are required to complete a program that implements a basic Car Park System. The car park has a maximum of 20 parking slots and only Cars, Vans and Three Wheeler's can park in the parking slots.

In this assignment, you will be required to implement the following functionality:

1) Design and implement a class Vehicle (abstract class or Interface) and the sub classes Car, Van, ThreeWheeler. The classes should include appropriate methods and hold information about the ID plate of the vehicle, the brand of the vehicle and the entry time/date into the parking location.

In particular:

- The Car class should also include appropriate methods and hold information about the number of the doors of the car and the color.
- The Van class should also include methods and information about the number of seats in the van if it's a passenger van, and Goods capacity in Kilo Grams, if it's a goods van.
- The class ThreeWheeler should also have methods and information if the Three Wheeler is a Taxi or NotForHire.

You should implement a class DateTime to represent the time/date of the entrance of the vehicle into the parking location.

2) Design and implement a class called SLIITCarParkManager to mange the available parking slots, this class has to implement the interface CarParkManager. SLIITCarParkManager maintains the list of the vehicles currently in the parking location.

```
interface CarParkManager{
    final int totalVehicleSlots = 20;
    boolean vehicleEntered();
    int vehicleLeft();
    int getNumEmptySlots();
    int getNumOccupiedSlots();
}
```

You do not need to assign a specific vehicle to a specific slot for this assignment. Simply keep track of vehicles going in and out of the parking location.

boolean vehicleEntered()

Should add a new vehicle into the parking location only if there are free slots available for parking (considering that the max number of slots cannot be changed, make sure there are available parking slots before allowing a vehicle in). If you allow a vehicle into the parking location , then also add a Entry to the Vehicle class (via the Car, Van or ThreeWheeler sub classes as appropriate), Display a message with the number of free slots remaining or informing that there are no further parking slots available. The method should return back ‘true’ or ‘false’ depending on if the current processing completed successfully or not.

int vehicleLeft()

should delete a vehicle from the vehicle Class (by selecting the ID plate), from the list when the vehicle leaves the car park, and should reduce the number of Occupied slots in the car park by one, and return the number of free slots that will be available after vehicle leaves. Make sure to print a message displaying the type of the vehicle leaving the parking (if it is a Car, a Van or a ThreeWheeler) and its ID.

provide a method to Print the list of the vehicles currently parked. For each vehicle print the ID plate, the entry time and the type of vehicle (if is a Car, a Van or a ThreeWheeler).

Demonstration

You will be asked to demonstrate the system. The following will be evaluated.

- understanding of the code
- quality of the code (e.g. good use of comments, naming of variables, structure of the code)

Some things to consider:

- We have not worked on collections (Lists, ArrayLists .etc.), so you may use a simple array to represent the parking slots in this location, or any other mechanism you wish including Lists and ArrayLists. Make sure you can explain the choices you make at the demo/mini viva.
- You do not need to provide any graphical representation.
- Please ask for help (from instructors, fellow students, and lecturer) if you have missed classes due to prevailing situation, and need help. DO NOT COPY. But DO get help, if you need it.