CS 342: Software Design
Spring 2015

Second Java Project

Due time: **9:00 pm on Monday 2/16/2015**

You are required to code a *Vehicle Rental Manager (VRM)* in Java. The VRM helps people working at a local car rental shop keep track of their vehicles. You must use inheritance to model different kinds of vehicles and to support polymorphism. There are three kinds of vehicles:

1. *Car*.
2. *Truck*.
3. *Minivan*.

All vehicles will have the following items of information, regardless of their kind

1. *Make*—The maker of the vehicle (e.g., Ford, Honda, etc.).
2. *Model*—The model of the vehicle (e.g., Taurus, Accord, etc).
3. *Year*—The model year of the vehicle
4. *Status*—The status of a vehicle is a string indicating whether the vehicle is available for rental, out rented, or down for repairs.
5. *Id*—This is an integer uniquely identifying each vehicle in the shop.
6. *Daily cost*—This is the price that the rental shop charges for renting a vehicle for a day.

In addition, the VRM keeps additional information for each kind of vehicle. In particular, trucks will have a field indicating the maximum cargo weight that the truck can carry (a floating point number). Cars will have a string indicating the body type (e.g., sedan, hatchback, station wagon, convertible, coupe). Minivans will have two fields indicating the overall length and width of the minivan.

All vehicles managed by the VRM must be kept in a single data structure, a linked list similar to the linked list that we saw in class. Make sure to use identifier polymorphism appropriately in order to hold different kinds of vehicles in different list nodes (i.e., there should be only one *Node* class). Also, your VRM must contain at least one abstract superclass.

Each class will be appropriately equipped with one or more constructors including at least a programmer-defined no-arg constructor. *You must code these classes from scratch. You are not allowed to use predefined Java libraries, except for the Scanner class for standard input.* Write a client class that implements a command line interface allowing a user to enter the commands below. Your command line interface will prompt the user for a command, and then execute the command. Here is a list of commands.

1. `v`—Add a new vehicle to the system. The user is prompted for the items of information pertaining to the vehicle (kind, make, model, etc.). Next, the user is prompted for information pertaining to the specific kind of vehicle. Finally, a new instance of an appropriate vehicle class is created with a new id and added to the list of vehicles.
2. `c`—Change vehicle status. The user is prompted for a new status for a given vehicle id.
3. `d`—Resets the linked list of vehicles to be an empty list.
4. `l`—Lists all the vehicles. All vehicles in the VRM are printed on the console display (including all fields of each vehicle). The list does not need to be sorted by vehicles unless command `s` was entered before `l`.
5. `a`—Lists all the vehicles whose status is available for rental. All such vehicles in the VRM are printed. The list does not need to be sorted by vehicles unless command `s` was entered before `l`.

6. s—The vehicles in the original linked list are sorted alphabetically by the make and model of the vehicle. You are free to reuse the current list (e.g., by rearranging its nodes), or to create a new list.

7. q—Quits the VRM.

You must work alone on this project. Use Eclipse to code this project. Create a new project and a package called edu.uic.cs342.NetID, where NetID is your UIC NetID. Collect all your source code files (.java files) in a zip archive and submit the archive by clicking on the link provided with this assignment. Your code should compile under the javac compiler. You grade will be based on the following criteria: (1) Full compliance with the specifications above; (2) the presence of helpful source code comments; (3) appropriate naming, use and access level of all identifiers (classes, methods and fields). No late submissions will be accepted.