CS4354 – Fall 2014 – Assignment 2

Due date: Wednesday, Sept. 24, 2014 at 12:00 noon.

Project Title: Car dealership (phase 2)

The car dealership of the first assignment, has now decided to expand its business and sell trucks and motorcycles as well, besides cars. It has also decided to not just keep an inventory of the vehicles on sale, but to also record user information and keep a record of the sales transactions completed.
To keep up with the new demands, the dealership needs to expand its management software to support the new information. For every vehicle, the software will now maintain the following information:

- VIN: String (instead of the license plate)
- Make: String
- Model: String
- Year: int
- Mileage: int
- Price: float

In addition, some additional information needs to be recorded for the different types vehicles. That information is listed bellow.

For passenger cars:
- Body style: String (Sedan, Coupe, Mini-Van, SUV, etc.)

For trucks:
- Maximum Load Weight (lb): float
- Length (ft): float

For motorcycles:
- Type: String (Scooter, Touring, Street Bike, etc.)
- Engine displacement (CC): int

The dealership now also keeps records of the users that the dealership has to deal with. The users are of two types, employees and customers.

For all user types, the following information is kept in the system:
- ID (unique number generated for each user): int
- First Name: String
- Last Name: String

In addition, the following information is kept for each user type.

Employee
- Monthly salary: float
- Direct deposit bank account number: int

Customer
- Phone number: String
- Driver License number: int

Finally, the dealership records information about completed transactions (sales). A sales transaction record consists of the following:
- Customer id: int
- Vehicle VIN: String
- Sale date: Date
- Final sale price: float
- Employee id (salesman): int

Note that the customer id, the employee id and the VIN, should match existing entries of customers, employees and vehicles already existing in the repository.

The new functionality of the system now looks as follows:

```
1. Add a new vehicle to the database.
2. Delete a vehicle from a database (given its VIN).
3. Show all existing vehicles in the database.
4. Show a list of vehicles within a given price range.
5. Add a new user to the database.
6. Update user info (given their id).
7. Show list of users.
8. Sell a vehicle.
9. Exit program.
```

For #1, when the user selects to add a new vehicle to the database, they should be asked to specify the type of the vehicle (car, truck, motorcycle) before they enter the vehicle information. The same applies for #5 and #6 regarding the user types.

For #4, the user has to also specify the type of the vehicle that they are looking for (car, truck, motorcycle).

For #8, when a vehicle is sold, a new sales record is created and the vehicle is automatically deleted from the database. The program should not allow a sale to be completed if any of the vehicle VIN or user ids do not exist in the database.

Tasks:

1. Create and object oriented java program that implements the dealership management software described above. Make use of the inheritance and polymorphism concepts that we saw in class. For example a `print()` method used to print information about a vehicle or a user, may me defined multiple times in the different classes and subclasses used in the program.

2. Try to make your program as robust as possible, by using Exception handling to deal with possible problems that may occur during the program execution.

3. This time do not save your data as text files, but as serializable objects. You may use more than one file to store your data, if it is more convenient to you.

4. Use a standard Java coding style to improve your program's visual appearance and make it more readable. I suggest the Google Java coding style: https://google-styleguide.googlecode.com/svn/trunk/javaguide.html

5. Use Javadoc is for every public class, and every public or protected member of such a class.
   Other classes and members still have Javadoc as needed. Whenever an implementation comment would be used to define the overall purpose or behavior of a class, method or field, that comment is written as Javadoc instead. (It's more uniform, and more tool-friendly.)

Notes:

This assignment is to be done in **groups of 2** (same groups as in assignment 1). If problems occur, please contact the instructor.

Logistics:

Please submit your files in a single zip file (assign1_xxxxxx_yyyyyy.zip). The xxxxxx and yyyyyy are your TX State NetIDs (mine is v_m137, you have two, one for each partner).
Submit an electronic copy only, using the Assignments tool on the TRACS website for this class.
Submit using the TRACS account of just ONE member of your partnership. Do NOT include executables or .class files in your submission.