COP 3330

Programming Assignment 3: Car Data Processing System

Program Overview

In this assignment, you will work with a Java application that processes information about different types of cars from a data file. The system defines three classes: Car, ElectricCar, and CarDataProcess. You will learn how to use inheritance, polymorphism, and file handling. The assignment will focus on reading data from a text file, processing it, and performing basic operations such as calculating the total price of all cars and displaying their details.

Classes:

1. **Car Class:**

   Represents a general car with attributes such as brand name as String and price as int.

   Constructor:

   - `Car(String brandName, int price):`

     Initializes a car object with the specified brand name and price.

   - `Car(String line):`

     Initializes a car object from a string that represents a line from the data file. The string is tokenized to extract the brand name and price. Example line: `g BMW 25000`

   - `Car():`

     A default constructor that initializes a car with default values ("Invalid" for brand name and -1 for price).

   Methods:

   - `int getPrice():`

     Returns the price of the car.

   - `void setPrice(int price):`

     Sets the price of the car.

   - `String getBrandName():`

     Returns the brand name of the car.

   -`void setBrandName(String brandName):`

     Sets the brand name of the car (void).

   - `String toString():`

     Returns a string representation of the car object in the format "GCar brandname: [brandName] price: [price]"

2. **ElectricCar Class (Subclass of Car):**

   – Represents an electric car, which extends the Car class and adds the attribute *isLithiumIon* to represent the type of battery used in the car. *isLithiumIon* is a Boolean instance variable.

   Constructor:

   `ElectricCar(String brandName, int price):`

   Initializes an electric car object with the specified brand name, price, and a default battery type (*isLithiumIon* set to false).

   `ElectricCar(String line):`

   Initializes an electric car object from a string that represents a line from the data file. The string is tokenized to extract the brand name, price, and battery type (either LithiumIon or NickelMetal). Example line: `e Honda 15000 0`

   Methods:

   - `String batteryType():`

     Returns the type of battery used in the electric car, if isLithiumIon is true then "LithiumIon" else "NickelMetal".

   - `String toString():`

     Returns a string representation of the electric car object in the format

"ECar brandname: [brandName] price: [price] battery: [batteryType]"

- `int getPrice():`

  You will override this method. If the electric car has lithiumIon battery then returns the 1.2 times of the price variable. Otherwise it returns the getPrice() of the Car class (super class's method will be called).

3. **CarDataProcess Class:**

   Contains methods for processing and managing car data from the text file.

   Methods:

   - `int getTotal(ArrayList<Car> arr):`

     Calculates the total price of all cars in the provided list (ArrayList<Car>) and returns the sum (int).

   - `void printCars(ArrayList<Car> arr):`

     Prints the details of all cars in the provided list using toString() method of the objects (ArrayList<Car>).

   - `ArrayList<Car> readIntoArrayList(String fname):`

     Reads car data from the specified file (using the file name provided as input) and returns an ArrayList<Car> containing the created Car objects. Each line in the file is read and a new car is created (either Car or ElectricCar) depending on the type of car (the first character, g for gasoline and e for electric) in the data file.

   `main method:`

     The main method where the program starts. It prompts the user to input the file name, reads the data into an array list, prints the car details, and calculates the total price using the methods you have implemented in this class.

Data File (cars.txt):

The cars.txt file contains data about cars. Each line represents one car and is formatted as follows:
- Electric car: e [brandName] [price] [batteryType]
- Gasoline car: g [brandName] [price]

Example data:

```
e Tesla 20000 1
e Honda 15000 0
g BMW 25000
g Buick 10000
```

Here, e represents an electric car, and g represents a gasoline car. The battery type for electric cars is represented by 1 (LithiumIon) or 0 (NickelMetal).

Example output for the cars.txt file is below (user input is written bold)

Input file name:**cars.txt**
ECar brandname: Tesla price: 24000 battery: LithiumIon
ECar brandname: Honda price: 15000 battery: NickelMetal
GCar brandname: BMW price: 25000
GCar brandname: Buick price: 10000
Total price is 74000

Assumptions

Assume the input file given by user (in this case *cars.txt*) exist and it is syntactically correct. There are no incomplete fields. Numbers will be integers. All the prices are large enough so that 1.2 times of that price will be also an integer.

If you see a space, it is a single space.

You can use stringtokenizer to tokenize the data line, extracting fields.

Submission Guidelines

- Your Java source file must begin with the following package declaration:
  package assignment3;
- Do not use the default package.
- Include a main method in CarDataProcess file only. If another file has also a main method, you will lose 20 pts.
- Submit only **three .java** files — do not submit your entire IDE project/class files. Do **not** put the java files inside a folder and zip that folder. There should be no folder structure in your submission. If you fail to do so, you will lose 20 pts.
- Do not use any third-party libraries.
- Your code must compile without errors. Non-compiling submissions will be penalized.
- Your program's output must match with overall format of the expected output.
- Do not concatenate more than one class in your java file. You will receive 0 if you do so.
- All the instance variables stated in this assignment will be private, whereas all the methods will be public.

Evaluation Criteria

Your submission will be assessed based on the following criteria:

10 pts: Proper Indentation (Inconsistent indentation may cost 10 pts)
10 pts: Header comments,package statement and descriptive comments
  If no package statement or incorrect package statement -10 pts

20 pts: The output matches exactly with the sample one for the sample data
20 pts: The Car class is implemented
  10 pts: all the methods are public, all instance variables are private
  10 pts: Two constructors are implemented as stated in the assignment correctly.
20 pts: ElectricCar class is implemented.
  10 pts : getPrice() method is overriden correctly.
  10 pts: toString() method is overriden correctly.
20 pts: CarDataProcess class is implemented
  5 pts : main method is implemented as stated in the assignment
  15 pts: other three methods are implemented correctly.

If the submission does not get compiled successfully, you will lose 40 pts
If no polymorphism is used in the assignment, you will lose 40 pts. Please implement the stated classes and methods. Just printing the correct output will not be sufficient to get full points from this assignment.