Bozhidar Mindov

COS221a

Vladimir Georgiev

Course Project User Guide

**Car Dealership Project**

**User Guide**

**Overview:**

The idea of the program is to create different vehicle objects using the implemented classes. Those objects are then placed into a collection, and the user will be able to sort the collection in different ways. The user will also be able to look for items in that collection.

The vehicle objects are read from a file called *“vehicles.txt”*, which is located in the project’s folder. If you want to add a vehicle to the file, you should do it there.

The vehicle types are:

**Car, SUV, Van, Truck, PickupTruck, Motorcycle and ElectricScooter.**

Example vehicles specified in the file:

*Car,Civic,Honda,Red,2022,110,500.6,10.5,1.5,petrol,true*

*SUV,Kodiaq,Skoda,Brown,2020,150,50000,100,2,diesel,false,7*

**Remark**: There should be no spaces between the commas and the different strings in the file. Also, the first string in each row specifies the type of vehicle you want to create.

**When the program is run:**

1. You will be presented with the following screen. You can choose how to sort the collection of vehicles: by total price, warranty or insurance:

Text

Description automatically generated

Depending on what you choose, the collection will be sorted and displayed in that way.

Example: (when you choose 1, the collection is sorted by *total price*):

Text

Description automatically generated

1. You will then be asked if you want to look for a vehicle in the database:

If you answer with **“no”**, the program will **exit**.

If you answer with **“yes”**, you will be asked to enter the model or the brand of the car you want to look for:

Graphical user interface, text

Description automatically generated

If there is a vehicle with that brand or model, its information will be displayed on screen.

Example output when we want to look for a vehicle of the “Skoda” brand in our database:

Text

Description automatically generated

If there were no matches found, you will see the following output:

Text

Description automatically generated

1. After that is done, the program will exit.

**The Technical Side of the Program:**

**Hierarchy of Classes:**

Diagram

Description automatically generated

**Remark:** You cannot create objects of the Vehicle class, since it is an abstract class, which contains just polymorphic methods that the classes inheriting it must implement. You can create objects of every other class.

**Polymorphic methods:**

**virtual int CalculateWarranty()** = 0; It returns the years of warranty that a certain vehicle has, usually based on its fuel type and the type of vehicle.

**virtual double CalculateTotalPrice(**) = 0; It calculates and returns the total price of a vehicle. It is based on the type of the vehicle, its power, size and other properties, such as fuel type.

**virtual double CalculateInsurance()** = 0; It calculates and returns the price of the monthly insurance that will have to be paid for the vehicle. It is again based on the type of the vehicle, its power, size, and other properties, such as fuel type.

**virtual void DisplayInfo() = 0**; - displays information about each vehicle. For example, it will display its brand, model, total price, insurance per month, warranty, etc.

**virtual string getBrand() = 0;** - returns the brand of the vehicle. It is required and used primarily in the pattern matching method of the main function.

**virtual string getModel() = 0;** - returns the model of the vehicle. It is required and used primarily in the pattern matching method of the main function.

**Algorithms used in the program:**

The program uses **Merge Sort** to sort the collection of vehicle objects.

The program also uses the **Boyer-Moore** pattern matching algorithm for the part where the user is allowed to search for a vehicle by its model or brand.

**Classes:**

**Vehicle** (base class)

Text

Description automatically generated

**Car** (inherits Vehicle)

Text

Description automatically generated

**SUV** (inherits Car)

Text

Description automatically generated

**Van** (inherits Car)

Text

Description automatically generated

**Truck** (inherits Vehicle)

Text

Description automatically generated

**PickupTruck** (inherits Truck)

Text

Description automatically generated

**Motorcycle** (inherits Vehicle)

Text

Description automatically generated

**ElectricScooter** (inherits Motorcycle)

Text

Description automatically generated