University of Central Punjab

Faculty of Information Technology

Object Oriented Programming
Fall 2020

Mini Project Two

Vehicle Management System

Instructions for Students:

1. Please create file with appropriate name
2. Late submissions will **NOT** be considered
3. Create as many classes and functions as required. Remember one function for one functionality.
4. Take care, plagiarism will not be tolerated at any case.
5. No .**rar** files are accepted
6. Submit screen shots of your testing.

Problem Statement:

We have a showroom known as BILAL MOTORS PVT LTD. Due to the COVID-19 pandemic, we want to launch an online system "BILAL MOTORS DATABASE" for our showroom, so that information of the different vehicles can be stored and people can check the information and availability of the vehicles sitting in their home by using online system. This will allow reducing the movement of people and minimizing the spread of the disease. The proposed system will offer information and availability of different vehicles like Bike, Car and Truck. Now you have to design a set of classes for storing Vehicle information, along with a class that will manage database of vehicles. Data should also be exported to file for storage of the information of the available vehicles at out showroom.

1. Design a set of classes that can store vehicles information. There should be one base class named as ***Vehicle*** to store common data, and three derived classes that divides the set of vehicle into three categories: **Bike**, **Car**, and **Truck**. All **data** stored in these classes should be private. Any access to class data from outside should be done through public member functions known as setters and getters. The **Vehicle** class should allocate storage for the following data.

   Attributes:

   - companyName **:** (*character pointer*)
   - color **:** (*character pointer*)
   - numberOfWheels **:** (*int*)
   - powerCC **:** (int)
   - typeOfVehicle **:** (*char pointer*) eg: bike, car, truck

   Methods:

   - virtual void checkType()=0;
     This function should be responsible to determine the type of vehicle, but you are not allowed to define it in this Class.
     For eg: if it has two wheels then it is bike.
   - virtual void display()const;
     This function display() will only print the attributes.

Remember:

Your Programme should not allow to make an **object** of Base Class *Vehicle*.

2. Each derived class should have a function **checkType()** that will check the type of vehicle and **set** the Vehicles's **typeOfVehcile** based on the stored information about number of wheels (attribute). All types are based on number of wheels.

Bike:

Attributes:

- height **:** (*double*)
- selfStart **:** (*bool*)
- discBrake **:** (*bool*)

- numberOfBikes : static int //This is to store the number of available bikes

Car:

Attributes:

- noOfDoors **:** (*int*)
- trasmission **:** (*character pointer*) eg: automatic or manual
- noOfSeats **:** (*int*)

- numberOfCars : static int //This is to store the number of available cars

Truck:

Attributes:

- containerSize **:** (*double*)
- category **:** (*character pointer*) eg: double-cabin or single-cabin
- fourWheelDrive : (*bool*)
- numberOfTrucks : static int //This is to store the number of available trucks

What else is required:

- **For each class *above* provide:**
  - Parameterized constructor with default arguments and Copy Constructor(use base initializer list with both constructors)
  - Destructor(with no memory leakage)
  - Getters/Setters for all private attributes(with no memory leakage and no returning of original memory handler)
  - Assignment Operator(with no memory leakage)
  - Don't provide function where you can overload operators, therefore must provide the following operators (only in child classes):
    - cin>> and cout<< operators (Do not use **friend** keyword)
    - fin>> and fout<< operators (Do not use **friend** keyword)

3. Write a class called **BilalMotors**, which will be used to store the list of various vehicles, using a **single array of pointers** of flexible size. **Note:** Each item in the array should be a Vehicle **pointer**, which should point to the appropriate type of Vehicle. Your list should only be big enough to store the vehicles flexibly.

   Note that the **BilalMotors** class is **NOT** part of the inheritance hierarchy described in the first two items above. This class is used to store and manage a LIST of individual vehicles.

   So all these vehicles are held by a list class named as BilalMotors.

BilalMotors:

Attributes:

- Array of pointers to hold all the vehicles
- A counter to maintain number of Vehicles added and indexing of the array

Methods:

Your **BilalMotors** class needs to have these public functions available:

- **BilalMotors( )**

  Default constructor: Sets up an empty list.

- **addVehicle(Vehicle *)**

  It will add a new vehicle into the array.

- **searchVehicle(char*)**

  Function to find all the available vehicles by type e.g. **Bike** will find and display all such vehicles with type **Bike**.

- **~ BilalMotors ( )**

  Destructor: Needs to reset all the array pointers with nullptr.

- **bool saveData(const char* fileName) const**;

  This function should create a **file** and write the number of all the vehicles available and all the data of the vehicles into an output file (filename should be passed by the user in the function). For example if user passes "**Bilal Ishfaq Motors**". Your function should open file **Bilal Ishfaq Motors.txt** and write all data required in it.

File format is described below:

```
Vehicle Information
--------------------
Number of Bikes: 2
Number of Cars:  2
Number of Trucks: 2

Company Name                    Type          Color    Power
-----------------------------------------------------------------------
Suzuki                           Bike         Red      150
Road Prince                      Bike         Black    70
Suzuki                           Car          White    800
Honda                            Car          Black    1349
Toyota                           Truck        Red      2799
Isuzu                            Truck        Black    2499
```

If the output file cannot be opened, return false for failure. Otherwise, after writing into file, return true for success.

Remember:

You are not allowed to return the memory handler (no getter) of the array outside the class.

Hint: Overload **indexing/subscript []** operator (both const and non-const) for it.

Driver Programme/main():

A Non-Member Function:

- Function that display whole information of all the vehicles on the screen.

Hint: Will require indexing/subscript[] operator.

void ShowVehicles(const BilalMotors &) ;

You cannot call any getters in this above function, except getCount().

This function should print the number of total vehicles available to the screen and the current list of vehicles, one vehicle per line. All the information needs to be printed in this printout including inherited information and its own information.

Output format should be:

```
                  Number of Total Vehicles: 2

-----------------------------------------------------------------

Company  Color  Wheels  Power  Type  Doors Transmission  Seats

Suzuki   White    4      800    Car    4      automatic     5

-----------------------------------------------------------------

Company  Color  Wheels  Power  Type   Height    SelfStart  DiscBrake

Honda     Red     2      70     Bike    22.2     No          No

-----------------------------------------------------------------
```

4. Write a main program (in a separate file) that creates a single **BilalMotors** object and then implements a menu interface to allow interaction with the object. Your main program should implement the following menu loop (any single letter options should work):

```
          *** BILAL MOTORS ***

S       Show vehicles list (brief)
E       Create a data file (output file)
A       Add new vehicle
        B for Bike
        C for Car
        T for Truck
F       Find Vehicle by type
Q       Quit Program
```

The **Create a data file** option should call the appropriate class function for printing the available vehicles information to file, respectively.

The **Show vehicles list** option should call the **non-member function** that prints the whole information of all available vehicles to screen.

The **Find Vehicle** option should find all the vehicles of the given type and display its information.

Quit should end the menu program. (Until this option is selected, keep looping back for menu selections).

General Requirements:

- No global variables!
- All member data of your classes must be private
- Use the const qualifier on member functions wherever it is appropriate.
- The code for this program should be portable. Test on compiler on sufficient test cases before submitting

Also submit the UML Diagram of the project. You can draw it on www.draw.io. Submit both XML and PNG file.