**COMPUTER SCIENCE DEPARTMENT**

**OOP Lab**

**Final Project Code**

# Submitted To: Ma’am Samra Mehboob

**Student Name: Syed Shehzad Ahmed**

# Reg. Number: 2312422

#include <iostream>

#include <string>

#include <vector>

using namespace std;

// Forward declaration of Showroom class

class Showroom;

// Base class Car

class Car {

private:

string model;

float price;

int year;

public:

Car() : model(""), price(0), year(0) {}

Car(string m, float p, int y) : model(m), price(p), year(y) {}

void displayCar() {

cout << "Model: " << model << ", Price: $" << price << ", Year: " << year << endl;

}

float getPrice() { return price; }

string getModel() { return model; }

friend void showroomInfo(Showroom& showroom); // Friend function declaration

};

// Derived class LuxuryCar

class LuxuryCar : public Car {

private:

string luxuryFeature;

public:

LuxuryCar(string m, float p, int y, string feature)

: Car(m, p, y), luxuryFeature(feature) {}

void displayLuxuryCar() {

displayCar();

cout << "Luxury Feature: " << luxuryFeature << endl;

}

};

// Manager class (Composition)

class Manager {

private:

string name;

public:

Manager(string n) : name(n) {}

void displayManager() {

cout << "Manager: " << name << endl;

}

};

// Customer class

class Customer {

private:

string name;

string purchasedModel;

public:

Customer() : name(""), purchasedModel("") {}

Customer(string n, string model) : name(n), purchasedModel(model) {}

void displayCustomer() {

cout << "Customer Name: " << name << ", Purchased Model: " << purchasedModel << endl;

}

};

// Showroom class

class Showroom {

private:

vector<Car> cars; // Vector to manage cars

vector<Customer> customers; // Vector to manage customers

float totalSales;

Manager manager;

public:

Showroom(string managerName) : totalSales(0), manager(managerName) {}

void addCar(Car car) {

cars.push\_back(car);

cout << "Car added: " << car.getModel() << endl;

}

void viewCars() {

if (cars.empty()) {

cout << "No cars available!" << endl;

return;

}

cout << "Available Cars: " << endl;

for (size\_t i = 0; i < cars.size(); i++) {

cars[i].displayCar();

}

}

void sellCar(string model, string customerName) {

for (size\_t i = 0; i < cars.size(); i++) {

if (cars[i].getModel() == model) {

totalSales += cars[i].getPrice();

customers.push\_back(Customer(customerName, model));

cout << "Car sold: " << model << " for $" << cars[i].getPrice() << endl;

// Remove the sold car

cars.erase(cars.begin() + i);

return;

}

}

cout << "Car not found!" << endl;

}

void viewTotalSales() {

cout << "Total Sales: $" << totalSales << endl;

}

void viewCustomers() {

if (customers.empty()) {

cout << "No customers found!" << endl;

return;

}

cout << "Customers: " << endl;

for (size\_t i = 0; i < customers.size(); i++) {

customers[i].displayCustomer();

}

}

friend void showroomInfo(Showroom& showroom);

};

// Friend function

void showroomInfo(Showroom& showroom) {

cout << "Showroom Manager: ";

showroom.manager.displayManager();

cout << "Total Sales: $" << showroom.totalSales << endl;

}

// Main function

int main() {

Showroom showroom("ssj");

int choice;

do {

cout << "\nCar Showroom Management System" << endl;

cout << "1. Add Car" << endl;

cout << "2. View Cars" << endl;

cout << "3. Sell Car" << endl;

cout << "4. View Total Sales" << endl;

cout << "5. View Customers" << endl;

cout << "6. Exit" << endl;

cout << "Enter your choice: ";

cin >> choice;

switch (choice) {

case 1: {

string model;

float price;

int year;

cout << "Enter car model: ";

cin >> model;

cout << "Enter car price: ";

cin >> price;

cout << "Enter car year: ";

cin >> year;

Car newCar(model, price, year);

showroom.addCar(newCar);

break;

}

case 2:

showroom.viewCars();

break;

case 3: {

string model, customerName;

cout << "Enter car model to sell: ";

cin >> model;

cout << "Enter customer name: ";

cin >> customerName;

showroom.sellCar(model, customerName);

break;

}

case 4:

showroom.viewTotalSales();

break;

case 5:

showroom.viewCustomers();

break;

case 6:

cout << "Exiting the system." << endl;

break;

default:

cout << "Invalid choice. Please try again." << endl;

}

} while (choice != 6);

return 0;

}

