***Institute of Computer And Technology***

***B.Tech – CSE(BDA)***

***Name:- Dwij Vatsal Desai***

***Sem:- 2***

***Sub: - ESFP-II***

***Enrollment No.:- 23162121027***

***Prac:- 6***

***Date:- 27/2/2024***

**Q.1.**

Objective: Make sample project using class, object, cout, cin, endl, getline() function, ignore(), and looping.

Preform the following instruction in sample project.

1. Insert minimum 5 newly available cars information in a showroom.

2. Display all newly cars to customer if selected to display.

3. Find most expensive car from the showroom.

4. Find most cheaper car from showroom.

5. Sort the cars by price in ascending or descending order to display as per the customers choice.

Car input:

Car model, Car Brand, Car Manufacturing year, Car Color, Car Price.

**Algorithm:-**

1. Start

2. Create a structure for entering data about students.

3. Program a code with the use of DMA.

4. Collect the Data from the user.

5. Show the data using printf.

6. Select a person's name.

7. Show the data of the person's name.

8. End

**Code:-**

/\*

Insertion: You'll need a method or function to add new cars to your showroom.

Think about what information you'll need for each car and how you'll store this

information in your program.

Display: Consider how you will show the newly available cars to customers.

Will you print them to the console? Display them in a graphical user interface

(GUI)? Think about how you want to present this information.

Finding: Think about how you will find the most expensive and cheapest cars in

your showroom. You'll need to iterate through your list of cars and compare

their prices to determine which is the most expensive or cheapest.

Sorting: Consider how you will sort the cars by price in either ascending or

descending order. You'll need to decide on a sorting algorithm and implement

it in your program.

\*/

#include <iostream>

#include <limits> // for numeric\_limits

#include <string>

using namespace std;

class Car

{

   public:

   // Car model, Car Brand, Car Manufacturing year, Car Color, Car Price.

    string Car\_model;

    int Car\_year;

    string Car\_Color;

    int Car\_Price;          // considering Price is in $

    char Car\_Brand[20];     // This is for taking getline

    // int found=0;

    Car()

    {}

    void Insertion()

    {

        cout<<endl<<"Enter the ditail of car(Car model, Car Manufacturing year, Car Color, Car Price.)\n";

        cin>>Car\_model>>Car\_year>>Car\_Color>>Car\_Price;

        cin.ignore();

        cout<<"Now Enter car Brand name:\t";

        cin.getline(Car\_Brand,20);

        cout <<"Here is Car Brand:"<<Car\_Brand;

    }

    void Display()

    {

        cout <<"Here is the ditail of car:"<<" "<<Car\_model<<" "<<Car\_year<<" "<<Car\_Color<<" "<<Car\_Price<<" "<<Car\_Brand<<endl<<endl;

    }

    void Finding(string a)

    {

        int i;

        if(a==Car\_model)

        {

            Display();

        }

    }

    static Car\* highest(Car cars[], int size)

    {

        if (size <= 0)

        {

            return nullptr; // Or some other value indicating no car found

        }

        Car\* mostExpensiveCar = &cars[0]; // Assume the first car is the most expensive

        for (int i = 1; i < size; ++i)

        {

            if (cars[i].Car\_Price > mostExpensiveCar->Car\_Price)

            {

                mostExpensiveCar = &cars[i]; // Update the most expensive car

            }

        }

        return mostExpensiveCar;

    }

    static Car\* lowest(Car cars[], int size)

    {

        if (size <= 0)

        {

            return nullptr; // Or some other value indicating no car found

        }

        Car\* mostAffordableCar = &cars[0]; // Assume the first car is the most affordable

        for (int i = 1; i < size; ++i)

        {

            if (cars[i].Car\_Price < mostAffordableCar->Car\_Price)

            {

                mostAffordableCar = &cars[i]; // Update the most affordable car

            }

        }

        return mostAffordableCar;

    }

    static void sortByPriceAscending(Car cars[], int size)

    {

        for (int i = 0; i < size - 1; ++i)

        {

            for (int j = 0; j < size - i - 1; ++j)

            {

                if (cars[j].Car\_Price > cars[j + 1].Car\_Price)

                {

                    // Swap cars[j] and cars[j+1]

                    Car temp = cars[j];

                    cars[j] = cars[j + 1];

                    cars[j + 1] = temp;

                }

            }

        }

    }

};

int main()

{

    Car obj[10];

    int num\_Car = 0;  // Initialize num\_Car to 0

    for(;;)

    {

        int i,Choice;

        string find\_car;

        cout<<"<1> Insert newly available cars information for the showroom."<<endl;

        cout<<"<2> Display all newly cars"<<endl;

        cout<<"<3> Find car by model."<<endl;

        cout<<"<4> Sort the cars by price in ascending order."<<endl;

        cout<<"<5> most expensive car from the showroom."<<endl;

        cout<<"<6> most cheaper car from showroom.."<<endl;

        cout<<"<7> Exit"<<endl<<endl;

        cout<<"Enter the no. here: ";

        cin>>Choice;

        cin.ignore();

            switch (Choice)

            {

                case 1:

                int numNewCars;

                cout<<"\nHow many units do you want: ";

                cin>>numNewCars;

                for (i = num\_Car; i < num\_Car + numNewCars && i < 10; i++)

                {

                    obj[i].Insertion();

                }

                num\_Car = i;

                break;

                case 2:

                for (i = 0; i < num\_Car ; i++)

                {

                    obj[i].Display();

                }

                cout<<endl;

                break;

                case 3:

                cout<<"Enter the Car model you want to find: ";

                cin>>find\_car;

                    for (i=0 ; i < num\_Car ; i++)

                    {

                        obj[i].Finding(find\_car);

                    }

                break;

                case 4:

                    // Sort the cars by price in ascending order

                    Car:: sortByPriceAscending(obj, num\_Car);

                    cout << "Cars sorted by price in ascending order:" << endl;

                    for (i = 0; i < num\_Car; ++i)

                    {

                        obj[i].Display();

                    }

                    break;

                case 5:

                {

                    // Find the most expensive car

                    Car\* mostExpensiveCar = Car::highest(obj, num\_Car);

                    if (mostExpensiveCar != nullptr)

                    {

                        cout << "The most expensive car is: " << mostExpensiveCar->Car\_model << " ($" << mostExpensiveCar->Car\_Price << ")" << endl;

                    }

                    else

                    {

                        cout << "No cars found." << endl;

                    }

                    break;

                }

                case 6:

                {

                    // Find the most affordable car

                    Car\* mostAffordableCar = Car::lowest(obj, num\_Car);

                    if (mostAffordableCar != nullptr)

                    {

                        cout << "The most affordable car is: " << mostAffordableCar->Car\_model << " ($" << mostAffordableCar->Car\_Price << ")" << endl;

                    }

                    else

                    {

                        cout << "No cars found." << endl;

                    }

                    break;

                }

                case 7:

                    return 0;

                break;

                default:

                cout<<endl<<">>>>Enter right number<<<<"<<endl<<endl;

                break;

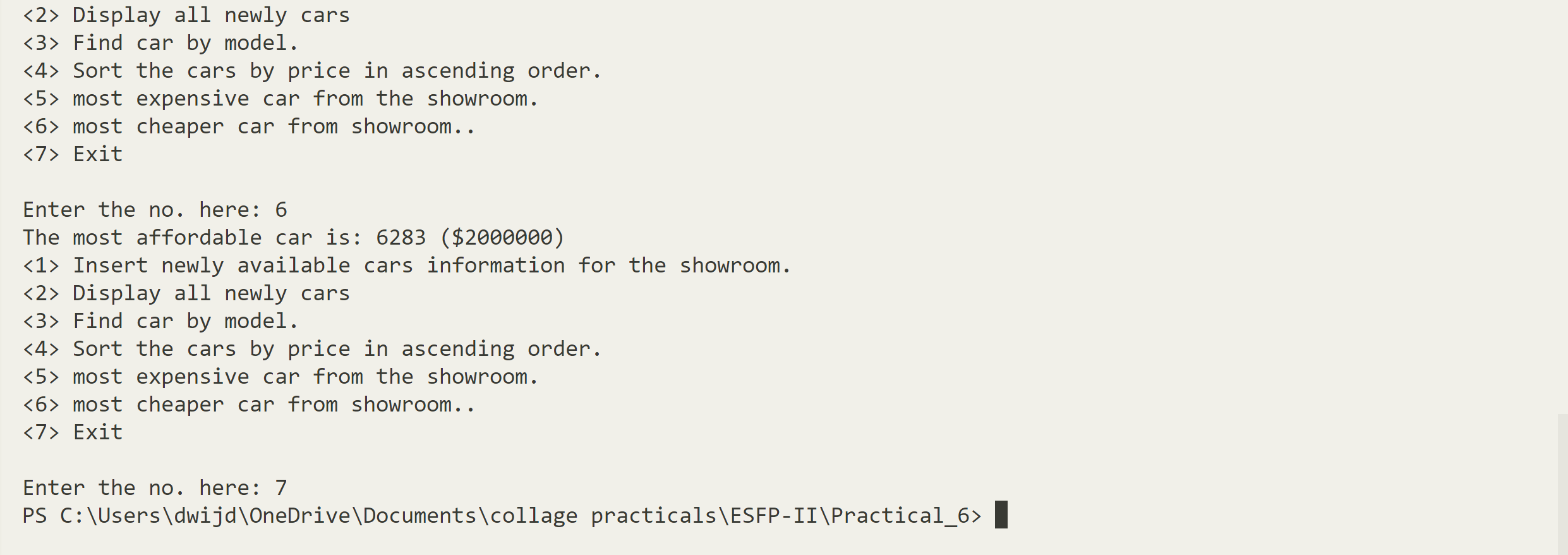
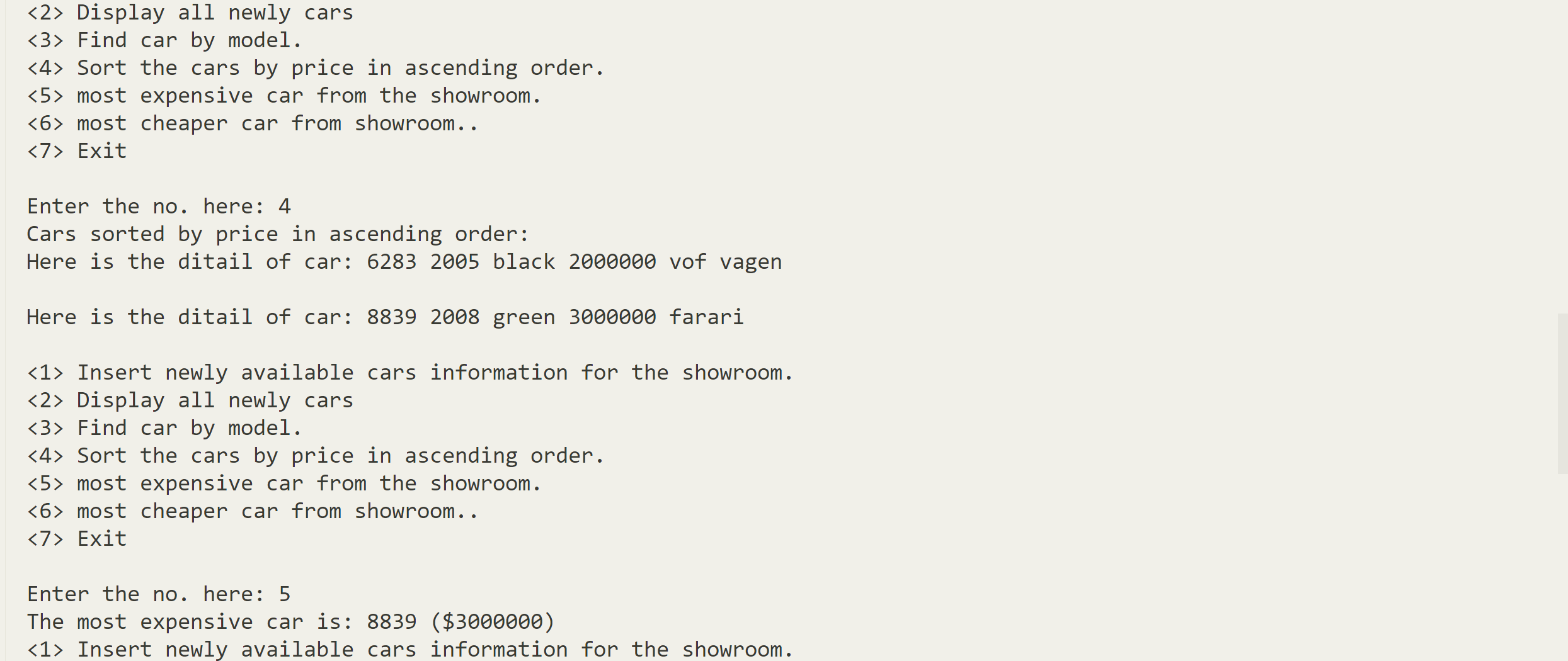
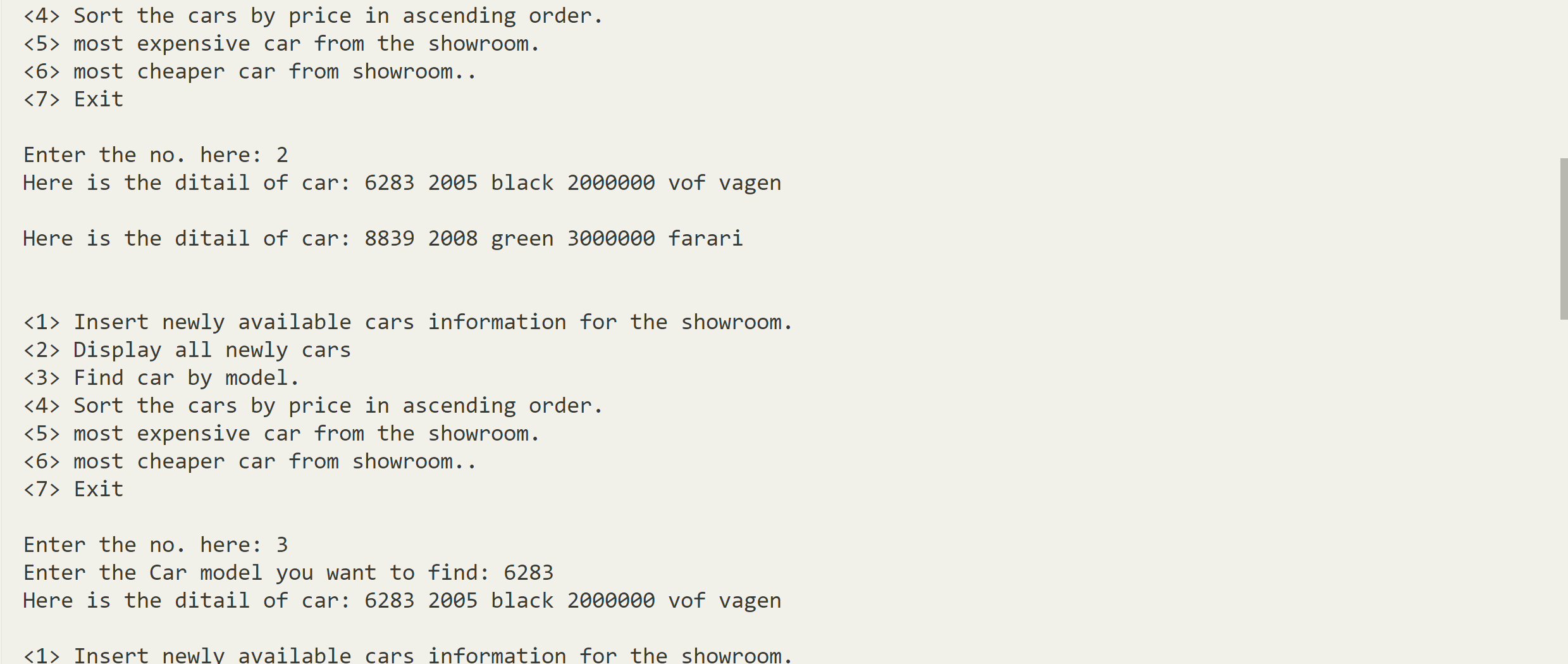
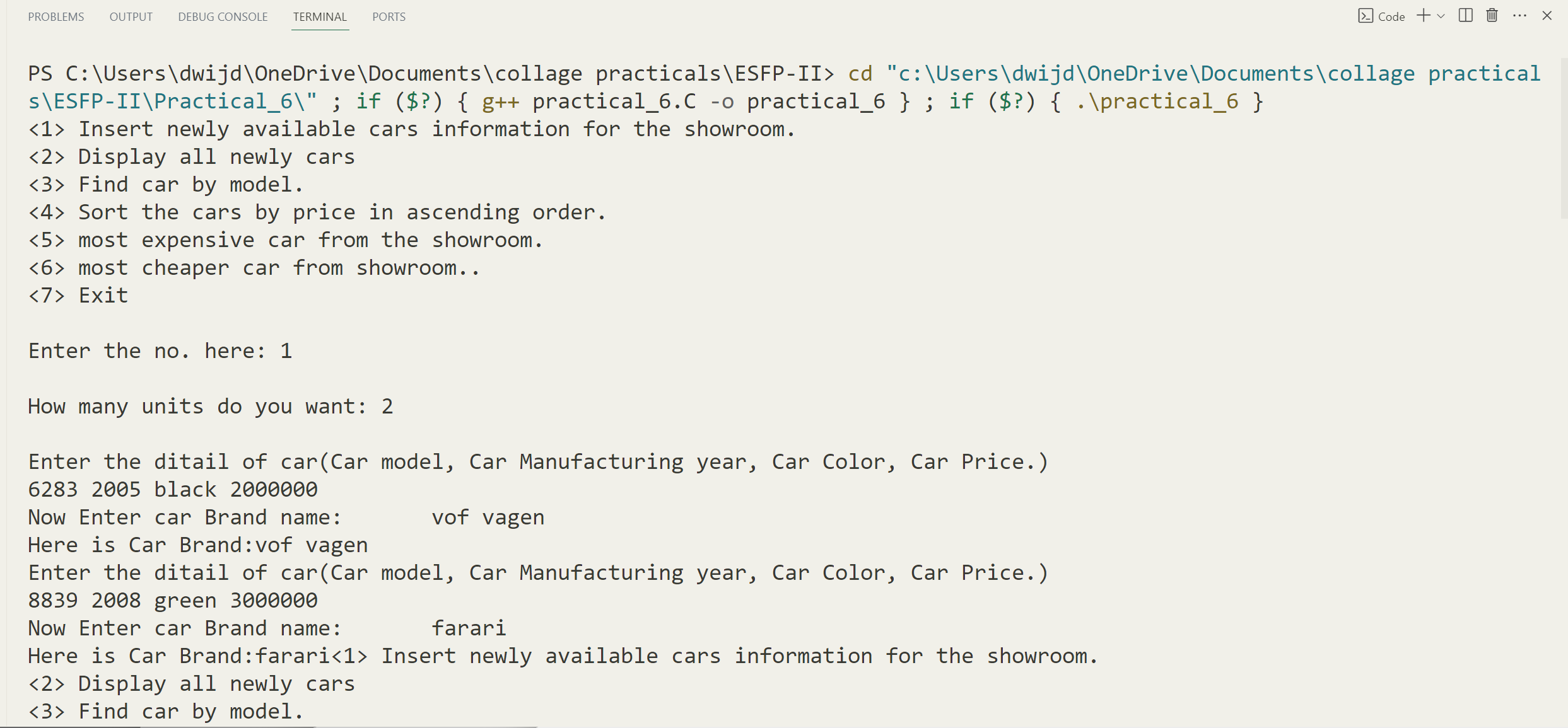
            }

    }

    return 0;

}

***Output-***

***Photo of code:-***