***Institute of Computer And Technology***

***B.Tech – CSE(BDA)***

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

***Sem:- 2***

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

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

***Prac:- 4***

***Date:- 12/2/2024***

**Q.1.**

Q.3. DMA: Definition: Admission Process in College.

At Mumbai, Wellington college is situated at Bandra, where this college is offering so many varieties of courses like diploma, undergraduate, post graduate and technical education courses. At the time of admission college suffers a lot of problems due to manual work. So, for the betterment process of admission, college authority decides to adopt technical help from

software technologies. For that, the college authority wants to make a software for performing all the tasks related to the admission process in a single desk. So that, the College authorities want, parents or students should not suffer for getting any types of information related to college from home before taking admission in different-different stream courses. So, whenever

he/she wants to get information, they can check all information from the site and at the end just for the admission process he/she should come to college. For that, College gave one sample admission form to the developer to perform the specific task, the remaining part will see the latter as per the functionality workout. In the admission form, all the details are given for the admission purpose like college\_ID, College\_Name, course\_stream, year, semester, subject group and their fee structure. So, as per the requirement, make a proper dynamic memory allocation program using “double link list concept”, where you have to perform all the above said requirements. Follow is the following instruction.

1. You have to input a number of college information at runtime.

2. You have to accept college info like college\_ID, College\_Name, course\_stream, year, semester, subject\_group and their fee structure from the user.

3. If a user wants to search college information, he / she can find it by ID or Name.

**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:-**

*/\**

*Name :- Dwij desai*

*En. No: 23162121027*

*2nd sem(BDA)*

*\*/*

*#include* <stdio.h>

*#include* <stdlib.h>

*#include* <string.h>

struct Car

{

    int CID;

    char Cname[30];

    char Cprice[20];

    char Ccolor[20];

    struct Car \*prev;

    struct Car \*next;

};

struct Car \*newnode, \*head = NULL, \*end;

void Last()

{

    newnode = (struct Car \*)malloc(sizeof(struct Car));

    printf("Enter the value as: ID of car, Name of car, Price, color of car:-\n");

    scanf("%d %s %s %s", &newnode->CID, &newnode->Cname, &newnode->Cprice, &newnode->Ccolor);

    printf("\n");

*if* (head == NULL)

    {

        newnode->next = NULL;

        newnode->prev = head;

        head = newnode;

        end = newnode;

    }

*else*

    {

        newnode->prev = end;

        end->next = newnode;

        end = newnode;

        end->next = NULL;

    }

}

void display()

{

    struct Car \*ttemp;

*if* (head == NULL)

    {

        printf("List is empty\n");

    }

*else*

    {

        printf("\nDisplay value:\n");

*for* (ttemp = head; ttemp != NULL; ttemp = ttemp->next)

        {

            printf("%d %s %s %s\n", ttemp->CID, ttemp->Cname, ttemp->Cprice, ttemp->Ccolor);

        }

        printf("\n");

    }

}

void findUnit()

{

    int choice, id;

    char name[30];

    printf("Enter choice (1 for ID, 2 for Name): ");

    scanf("%d", &choice);

*if* (choice == 1)

    {

        printf("Enter Car ID to find: ");

        scanf("%d", &id);

    }

*else* *if* (choice == 2)

    {

        printf("Enter Car Name to find: ");

        scanf("%s", name);

    }

    struct Car \*temp = head;

    int found = 0;

*while* (temp != NULL)

    {

*if* ((choice == 1 && temp->CID == id) || (choice == 2 && strcmp(temp->Cname, name) == 0))

        {

            printf("Car found: %d %s %s %s\n", temp->CID, temp->Cname, temp->Cprice, temp->Ccolor);

            found = 1;

        }

        temp = temp->next;

    }

*if* (!found)

    {

        printf("Car not found.\n");

    }

}

void deleteUnit()

{

    int choice, id;

    char name[30];

    printf("Enter choice (1 for ID, 2 for Name): ");

    scanf("%d", &choice);

*if* (choice == 1)

    {

        printf("Enter Car ID to delete: ");

        scanf("%d", &id);

    }

*else* *if* (choice == 2)

    {

        printf("Enter Car Name to delete: ");

        scanf("%s", name);

    }

    struct Car \*temp = head;

    struct Car \*prev = NULL;

*while* (temp != NULL)

    {

*if* ((choice == 1 && temp->CID == id) || (choice == 2 && strcmp(temp->Cname, name) == 0))

        {

*if* (prev == NULL)

            {

                head = temp->next;

            }

*else*

            {

                prev->next = temp->next;

            }

            free(temp);

            printf("Car deleted successfully.\n");

*return*;

        }

        prev = temp;

        temp = temp->next;

    }

    printf("Car not found.\n");

}

void add()

{

    int choice;

    printf("Press <1> to add at front\n");

    printf("Press <2> to add at end\n");

    scanf("%d", &choice);

    newnode = (struct Car \*)malloc(sizeof(struct Car));

    printf("Enter the value as: ID of car, Name of car, Price, color of car:-\n");

    scanf("%d %s %s %s", &newnode->CID, &newnode->Cname, &newnode->Cprice, &newnode->Ccolor);

    printf("\n");

*if* (head == NULL)

    {

        newnode->next = NULL;

        newnode->prev = NULL;

        head = newnode;

        end = newnode;

    }

*else*

    {

*if* (choice == 1)

        {

            newnode->prev = NULL;

            newnode->next = head;

            head->prev = newnode;

            head = newnode;

        }

*else* *if* (choice == 2)

        {

            newnode->prev = end;

            newnode->next = NULL;

            end->next = newnode;

            end = newnode;

        }

    }

}

int main()

{

    int a = 0;

*for* (;;)

    {

        printf("Press <1> to add value at end \n");

        printf("Press <2> to display value \n");

        printf("Press <3> to find car \n");

        printf("Press <4> to delete car \n");

        printf("Press <5> to add car \n");

        printf("Press <6> to end code \n");

        printf("\nEnter number for menu:  ");

        scanf("%d", &a);

*switch* (a)

        {

*case* 1:

            int num\_Car;

            printf("\nHow many units do you want: ");

            scanf("%d", &num\_Car);

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

            {

                Last();

            }

*break*;

*case* 2:

            display();

*break*;

*case* 3:

            findUnit();

*break*;

*case* 4:

            deleteUnit();

*break*;

*case* 5:

            add();

*break*;

*case* 6:

*return* 1;

*break*;

*default*:

            printf("Enter right number\n");

*break*;

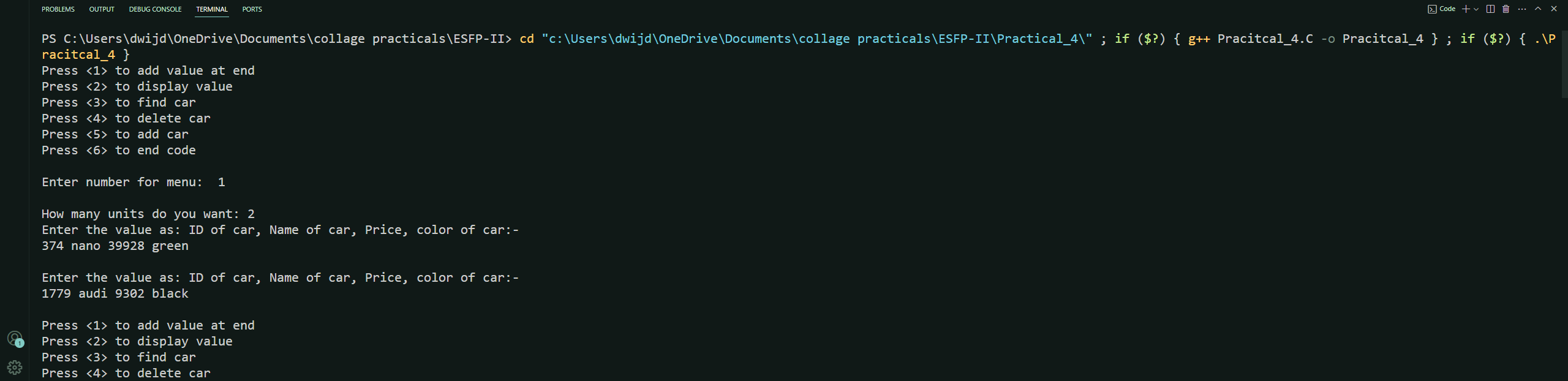
        }

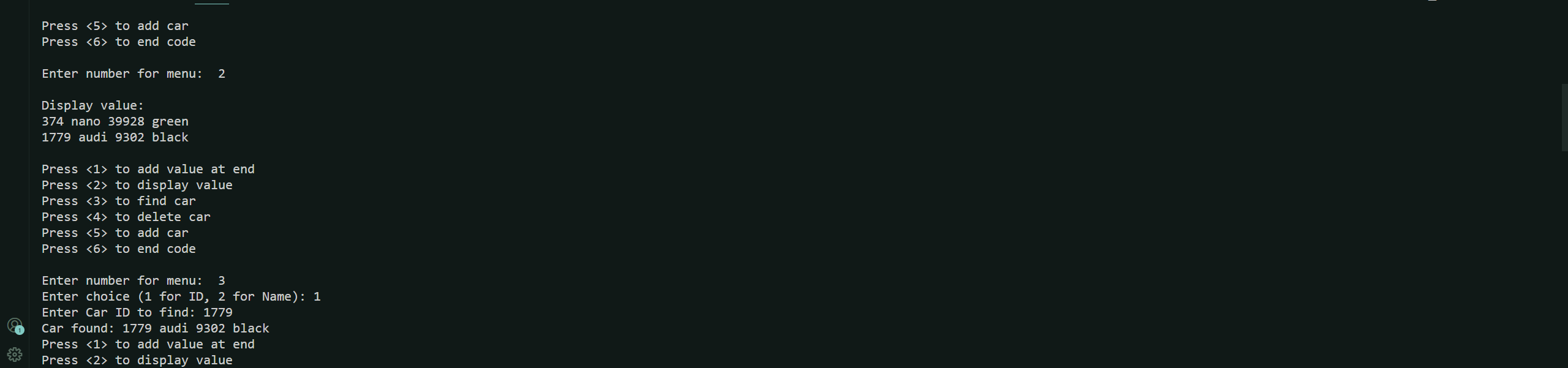
    }

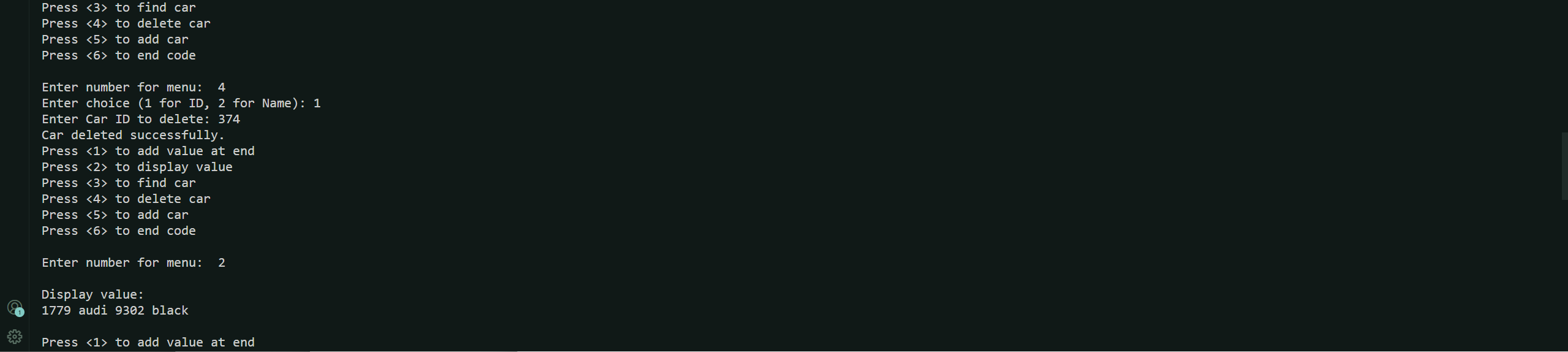
*return* 0;

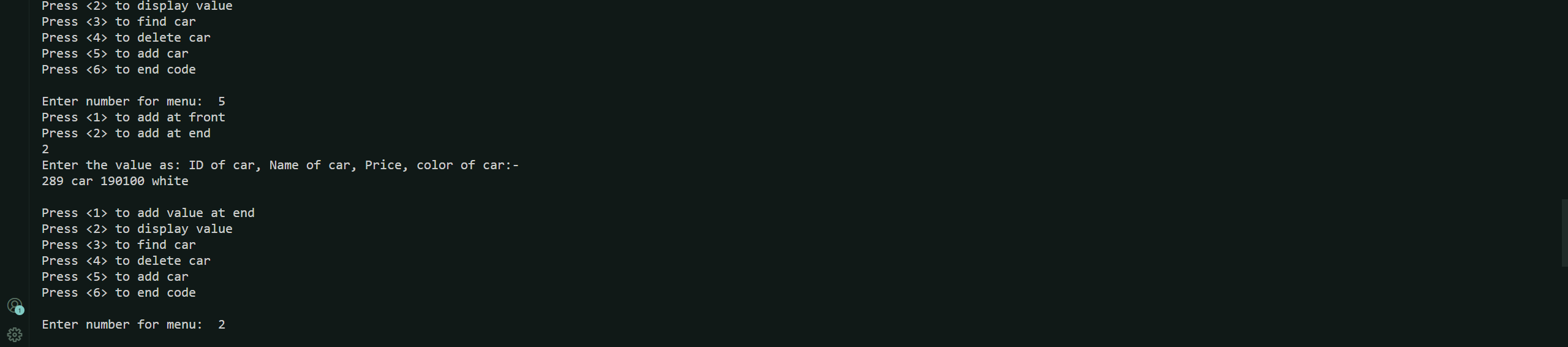
}

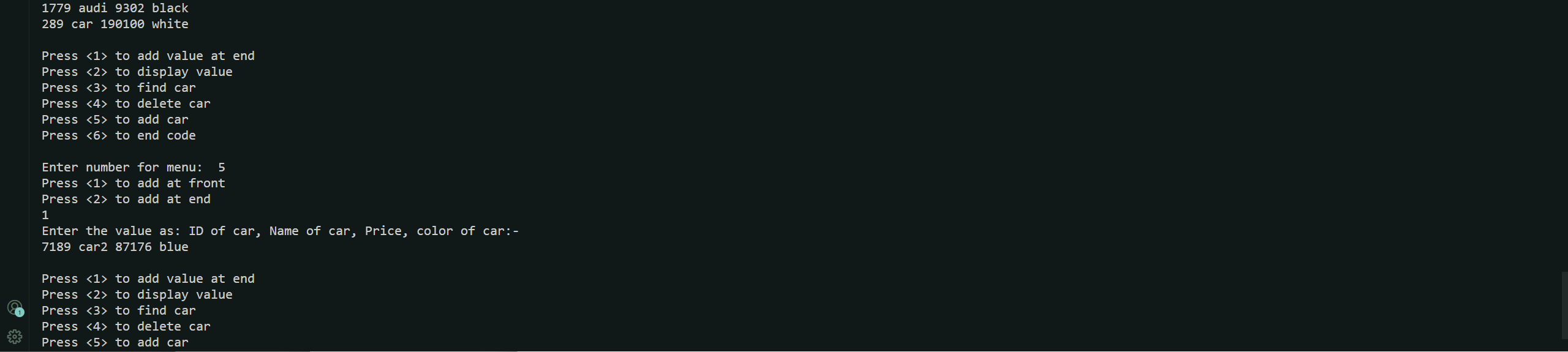
***Output-***

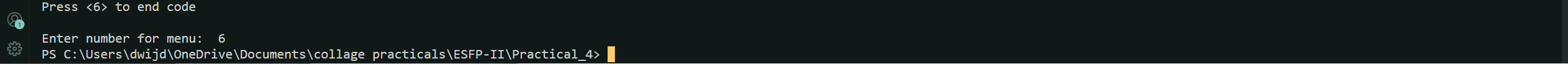
**

**









***Photo of code:-***

