STUDENT MANAGEMENT SYSTEM

JAYESH S CHAUDHARI [RA2011028010094]

Functions used in the program :

- Insert Function to insert data in linked list
- Search Function to search student record
- Delete Function to delete existing student record
- Display Function to display student records
- Main Function where the main code runs and main menu

Approach :

- The program should be menu driven
- Using option task would be performed
- Creating a node student
- Inserting Data (Name,roll number,marks,course) as preferred data type (char,string,integer etc.).
- Creating a linked list.using insert( ) function
- Using roll number we would be using functions like search,display,delete
- Each and every node will be traversed in a linked list if roll number matches the function will be executed or else it will return to the main menu with an error message.

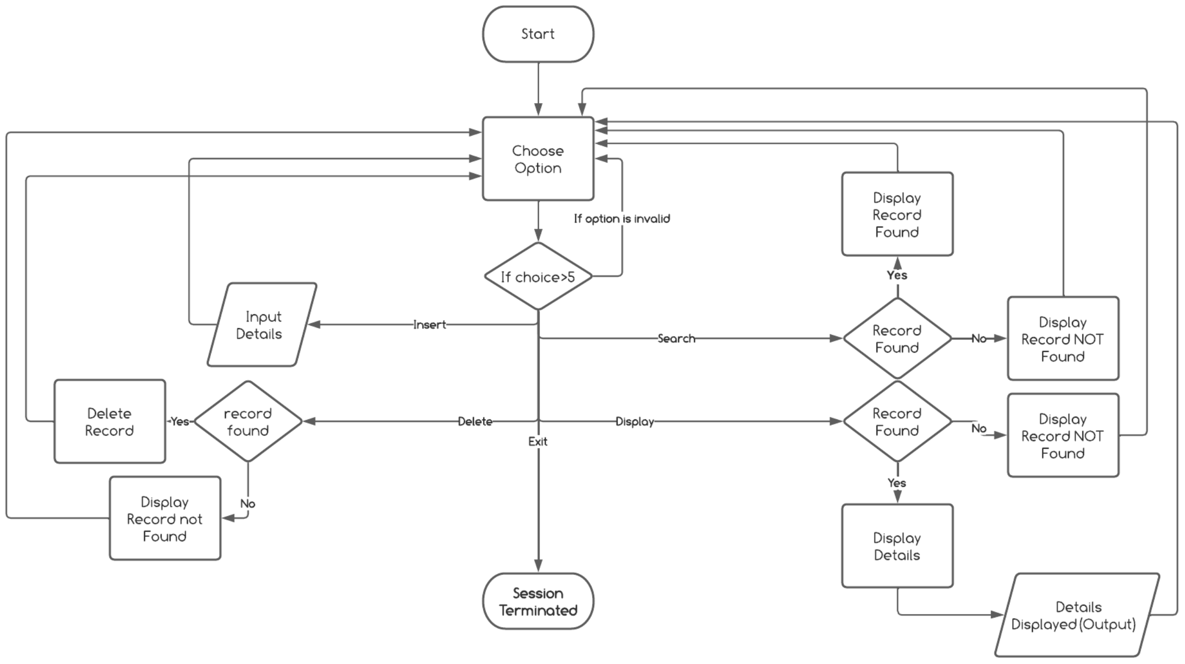
Algorithm

Diagram (Flow Chart)

Student Management System

Algorithm

JAYESH S CHAUDHARI [RA2011028010094]



Algorithm for Main( ) [Main Menu]

Input : Option
Ouput : Function Performed

Step I : Do While loop for Menu

```
    do//do loop for entering choice
    {    ________________________

        } while (choice != 0);
```

Step II : Input Choice

```
  scanf("%d", &choice);
```

Step III : Using Switch( ) for Menu Function

```
switch (choice)//switch for chosing the options
         {
_________________________________
}
```

Step IV : Function would be called according to given menu choice

```
Case 2: //Case two is for search function
                printf("Enter roll number to search: ");
                scanf("%d", &rollnumber);
                search(rollnumber);//Search() function called
                break;
```

Step V : Default if wrong choice

```
default:printf("\nEnter a valid choice.\n");
```

Algorithm for Insert( ) [Creating a linked list]

Input : Student Details
Ouput : Details Saved in a Linked list

Step I : Create a node by allocating memory

```
struct Student * student = (struct Student *) malloc(sizeof(struct Student));
//memory allocation
```

Step II : Assign data

```
    student->rollnumber = rollnumber;
    strcpy(student->name, name);
    strcpy(student->course, course);
    student->marks = marks;
```

Step III : Point Next pointer towards NULL

```
student->next = NULL;
```

Step IV : Check whether header is empty or not

```
if(head==NULL){
        // if head is NULL
        // set student as the new head
        head = student;
    }
```

Step V : If head is not empty create a link list

```
 else{
        // if list is not NULL
        // insert student in beginning of head
        student->next = head;
        head = student;
    }
```

Algorithm for Search( ) [Searching in linked list]

Input : Roll Number
Ouput : Student Details

Step I : Input the key to be searched

```
scanf("%d", &rollnumber);
//This value is taken in main() but passed in search using Search(rollnumber);
```

Step II : create a temporary head for traversal

```
struct Student * temp = head;
```

Step III : Using While loop traverse till last node

```
while(temp!=NULL){________
___________
}
```

Step IV : Using If condition match the searched key and details are displayed

```
if(temp->rollnumber==rollnumber){
      printf("----------------------------------------------------------------------);
           printf("\nRoll Number: %d\n", temp->rollnumber);
           printf("Name: %s\n", temp->name);
           printf("Course: %s\n", temp->course);
           printf("Total Marks: %0.2f\n", temp->marks);

printf("-----------------------------------------------------------------------");
           return;
        }
        temp = temp->next;
    }
```

Step V:If key not found print details not found

```
printf("Student with roll number %d is NOT found !!!\n", rollnumber);
```

Algorithm for Delete( ) [Deleting a node in a linked list]

Input : Roll Number
Output : Student Detail Deleted

Step I : Input the key to be deleted

```
scanf("%d", &rollnumber);
//This value is taken in main() but passed in delete using deleterollnumber);
```

Step II : create two temporary head for traversal(one for current position other one for previous)

```
  struct Student * temp1 = head;
  struct Student * temp2 = head;
```

Step III : Using While loop traverse till last node

```
while(temp!=NULL){_______
___________
}
```

Step IV : Using If condition find the key

```
if(temp->rollnumber==rollnumber){___________________
_________________
}

else printf("Student with roll number %d is NOT found !!!\n", rollnumber);
//if record not found
```

Step V: If node to be deleted is a head node or not

```
if(temp1==temp2){
              ______________________________
            }
            else{
            _________________________________________
            }
```

Step VI : Link nodes accordingly and free the memory

```
if(temp1==temp2){
                // this condition will run if
                // the record that we need to delete is the first node
                // of the linked list
                head = head->next;
                free(temp1);
            }
            else{
                // temp1 is the node we need to delete
                // temp2 is the node previous to temp1
                temp2->next = temp1->next;
                free(temp1);
            }
```

Algorithm for Display( ) [Displaying a linked list]

Input : Option Number
Ouput : All Student Details

Step I : create a temporary head for traversal

```
struct Student * temp = head;
```

Step II : Using While loop traverse till last node

```
while(temp!=NULL){_______
___________
}
```

Step III : Print All Student detail details

```
printf("\n--------------------------------------------------------------------");
        printf("\nRoll Number: %d\n", temp->rollnumber);
        printf("Name: %s\n", temp->name);
        printf("Course: %s\n", temp->course);
        printf("Total Marks: %0.2f\n", temp->marks);

printf("---------------------------------------------------------------------");
```

Step IV: Traverse further till end by pointing towards next node

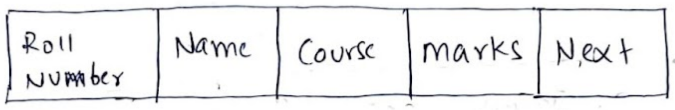
```
   temp = temp->next;
```

Source Code :

```
#include<stdlib.h>
#include<string.h>
#include<stdio.h>


struct Student//stucture of node
{
    int rollnumber;
    char name[100];
    char course[100];
    float marks;
    struct Student *next;

}* head;
void insert(int rollnumber, char* name, char* course, float marks)
//insert function for inserting data
{

    struct Student * student = (struct Student *) malloc(sizeof(struct Student));
//memory allocation
    student->rollnumber = rollnumber;
    strcpy(student->name, name);
    strcpy(student->course, course);
    student->marks = marks;
    student->next = NULL;

    if(head==NULL){
        // if head is NULL
        // set student as the new head
        head = student;
    }
    else{
        // if list is not NULL
        // insert student in beginning of head
        student->next = head;
        head = student;
    }

}
void search(int rollnumber)//function to search detail using rollnumber
{
    struct Student * temp = head;
    while(temp!=NULL){
        if(temp->rollnumber==rollnumber){

printf("------------------------------------------------------------------------------------------------");
            printf("\nRoll Number: %d\n", temp->rollnumber);
            printf("Name: %s\n", temp->name);
            printf("Course: %s\n", temp->course);
            printf("Total Marks: %0.2f\n", temp->marks);

printf("------------------------------------------------------------------------------------------------");
            return;

        }
        temp = temp->next;
    }
    printf("Student with roll number %d is NOT found !!!\n", rollnumber);
}

void Delete(int rollnumber)//Deleting student detail using rollnumber
{
    struct Student * temp1 = head;
    struct Student * temp2 = head;
    while(temp1!=NULL){

        if(temp1->rollnumber==rollnumber){

printf("\n---------------------------------------------------------------------------------
-----");
            printf("\nRecord with roll number %d Found !!!\n", rollnumber);

            if(temp1==temp2){
                // this condition will run if
                // the record that we need to delete is the first node
                // of the linked list
                head = head->next;
                free(temp1);
            }
            else{
                // temp1 is the node we need to delete
                // temp2 is the node previous to temp1
                temp2->next = temp1->next;
                free(temp1);
            }

            printf("Record Successfully Deleted !!!\n");

printf("-----------------------------------------------------------------------------------
---");
            return;

        }
        temp2 = temp1;
        temp1 = temp1->next;

    }
    printf("Student with roll number %d is NOT found !!!\n", rollnumber);//if record not
found

}
void display()
{printf("\nAll Student's Details :\n");
    struct Student * temp = head;
    while(temp!=NULL){

printf("\n---------------------------------------------------------------------------------
---");
        printf("\nRoll Number: %d\n", temp->rollnumber);
        printf("Name: %s\n", temp->name);
        printf("Course: %s\n", temp->course);
        printf("Total Marks: %0.2f\n", temp->marks);

printf("-----------------------------------------------------------------------------------
---");
        temp = temp->next;
    }}
int main()//main function
{   head = NULL;
    int choice;
    char name[100];
    char course[100];
    int rollnumber;
    float marks;
  printf("                                                        STUDENT MANAGEMENT SYSTEM");
    do//do loop for entering choice
    {   printf("\n        MENU\n");
        printf("1 Insert student details\n2 Search for student details\n3 Delete student
details\n4 Display all student details\n5 Exit\n");
        printf("\nEnter Choice: ");
        scanf("%d", &choice);
        switch (choice)//switch for chosing the options
        {
            case 1:
                printf("Enter roll number: ");
                scanf("%d", &rollnumber);
                printf("Enter name: ");
                scanf("%s", name);
                printf("Enter course : ");
                scanf("%s", course);
                printf("Enter total marks: ");
                scanf("%f", &marks);
                insert(rollnumber, name, course, marks);
                break;
            case 2:
                printf("Enter roll number to search: ");
                scanf("%d", &rollnumber);
                search(rollnumber);
                break;
            case 3:
                printf("Enter roll number to delete: ");
                scanf("%d", &rollnumber);
                Delete(rollnumber);
                break;
            case 4:
                display();
                break;
            case 5:  system("clear");

printf("-----------------------------------------------------------------------------------
---");
            printf("\nProgram Session Terminated Successfully\n");
            printf("\nProgram made by : \n>> Jayesh(094)\n>> Mukund(086)\n>>
Pranav(079)\n");

printf("-----------------------------------------------------------------------------------
---");
            exit(0);

            break;
            default:printf("\nEnter a valid choice.\n");
            break;
        }

    } while (choice != 0);}
```

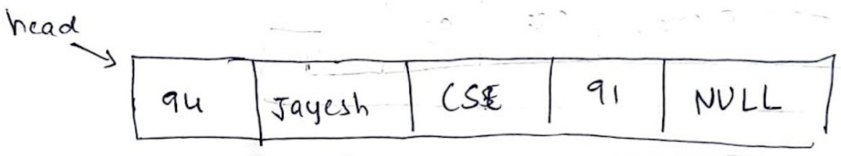
Dry Run :

Student Node structure

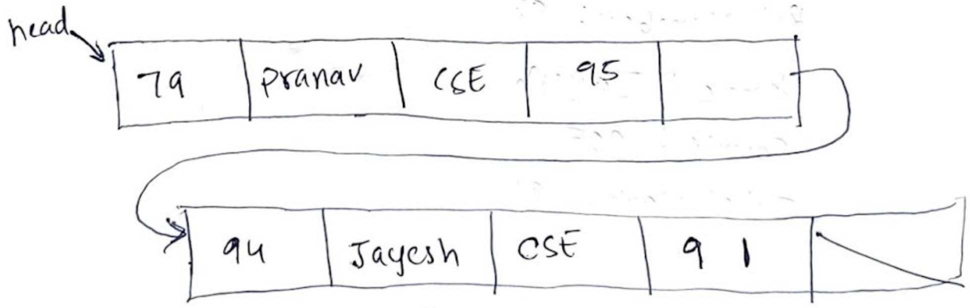


Enter choice : 1
Enter Roll number : 94
Enter Name : Jayesh
Enter Course : CSE
Enter Total Marks : 91

insert (rollnumber, name, course, marks);



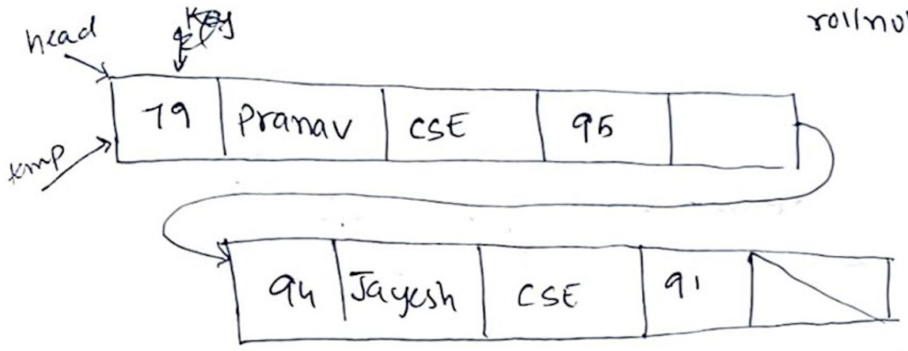
Enter Choice : 1
Enter Roll number : 79
Enter Name : Pranav
Enter Course : CSE
Enter Total Marks : 95



Enter choice: 2

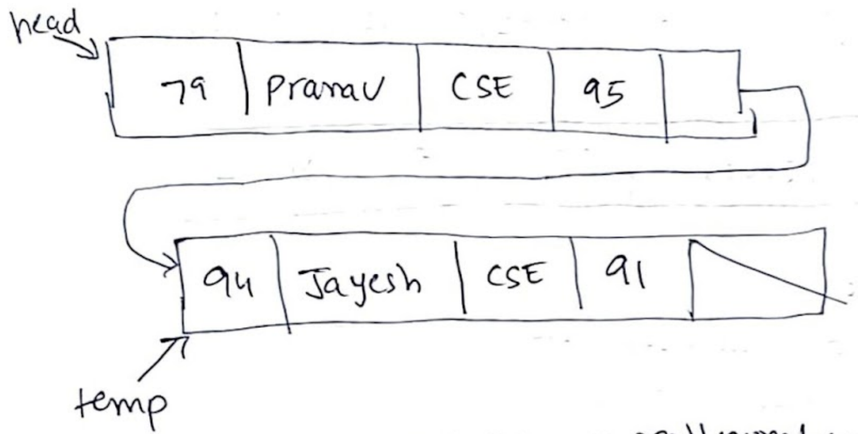
Enter roll number to be searched : 94

rollnumber = 94



temp→rollnumber != rollnumber

temp = temp→next;



temp→rollnumber == rollnumber

Roll Number: 94

Name: Jayesh

Course: CSE

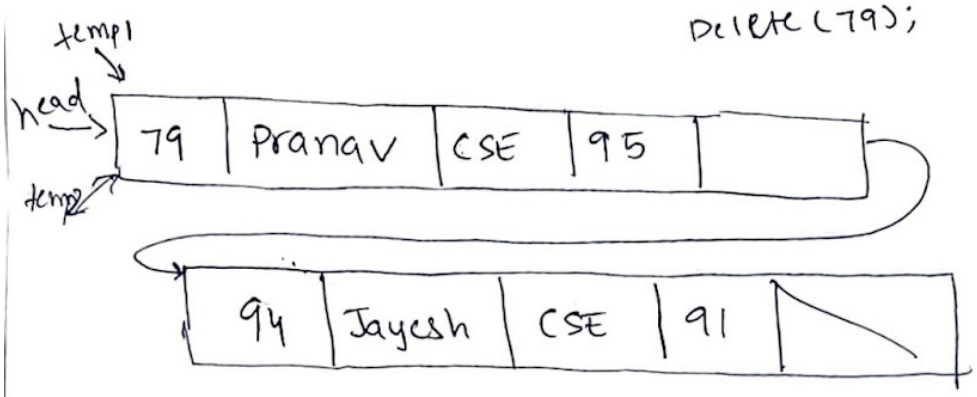
Total Marks: 91

Enter choice : 3

Enter roll number to delete : 79

roll number = 79

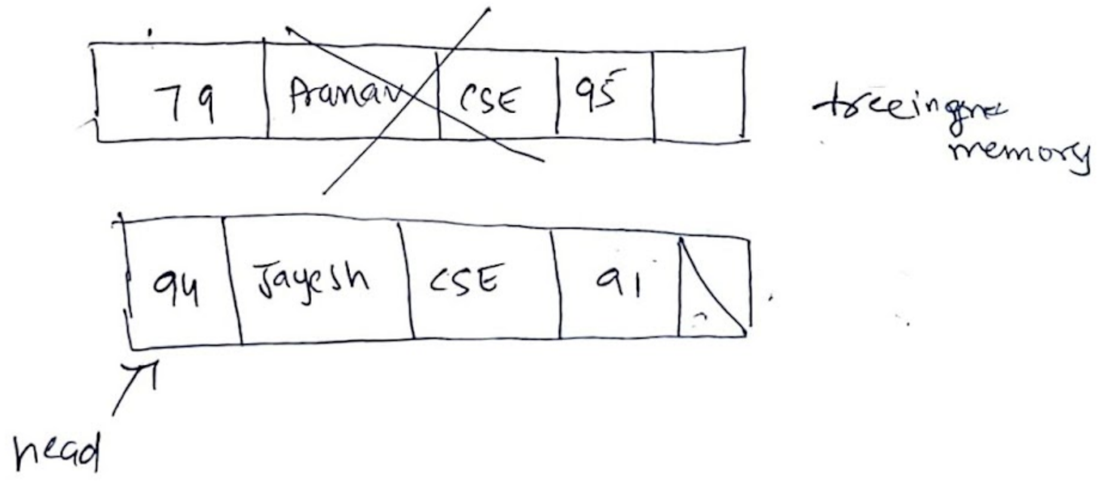
Delete (79);



temp1 == temp2

head = head → next

free (temp1);



Time Complexity :

Insert ( ) - O (1)

Search ( ) - O (n)

Delete ( ) - O (n)

Display ( ) - O (n)

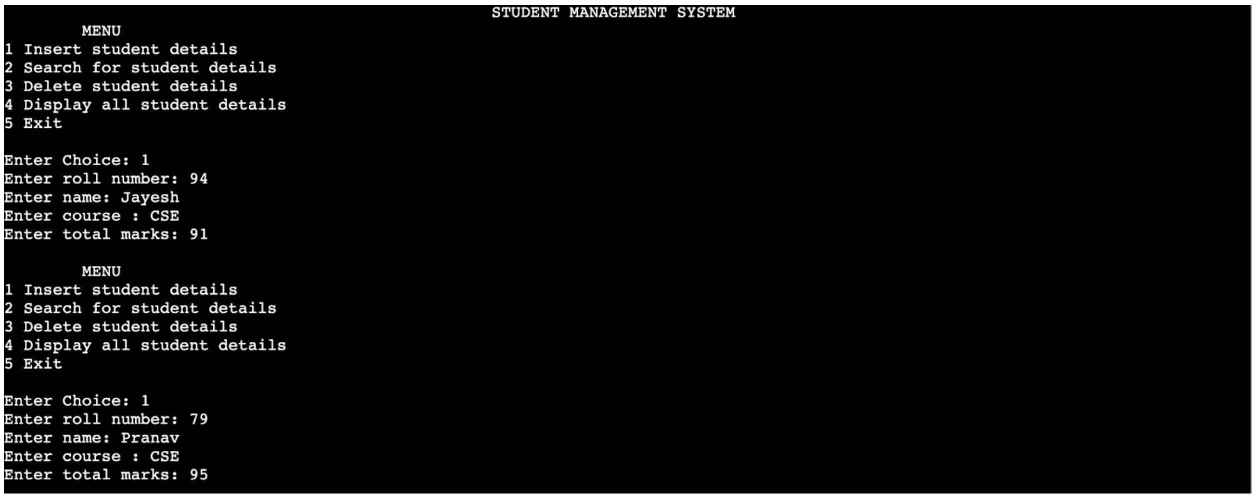
Main ( ) - O (n)

Overall Time Complexity : O (n)

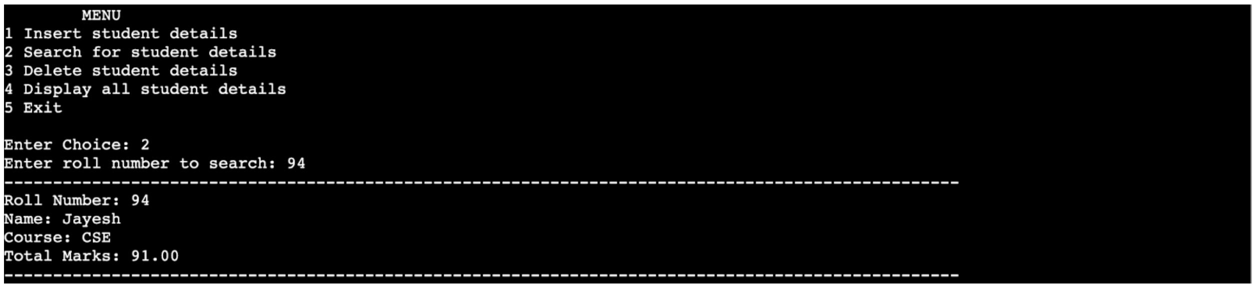
Space Complexity :

Space complexity of the program is O (n) because memory allocation is done dynamically.

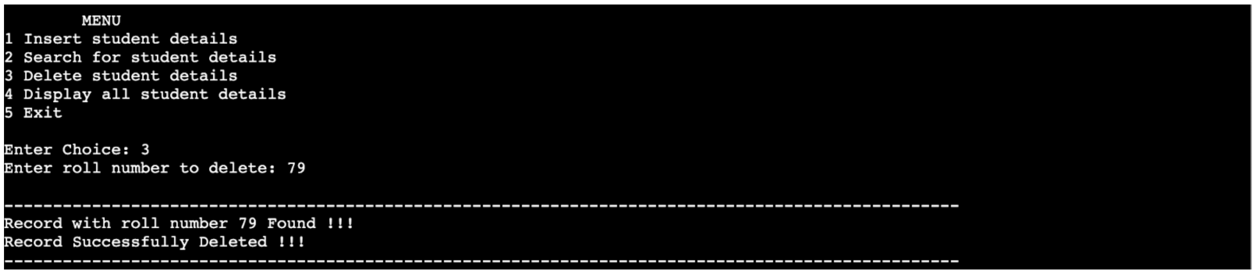
Result :



```
                                                   STUDENT MANAGEMENT SYSTEM
        MENU
1 Insert student details
2 Search for student details
3 Delete student details
4 Display all student details
5 Exit

Enter Choice: 1
Enter roll number: 94
Enter name: Jayesh
Enter course : CSE
Enter total marks: 91

        MENU
1 Insert student details
2 Search for student details
3 Delete student details
4 Display all student details
5 Exit

Enter Choice: 1
Enter roll number: 79
Enter name: Pranav
Enter course : CSE
Enter total marks: 95
```

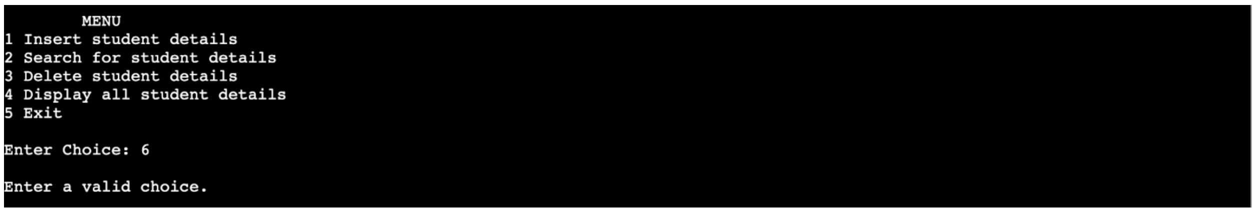


```
        MENU
1 Insert student details
2 Search for student details
3 Delete student details
4 Display all student details
5 Exit

Enter Choice: 2
Enter roll number to search: 94
---------------------------------------------------------------------------------------------------
Roll Number: 94
Name: Jayesh
Course: CSE
Total Marks: 91.00
---------------------------------------------------------------------------------------------------
```

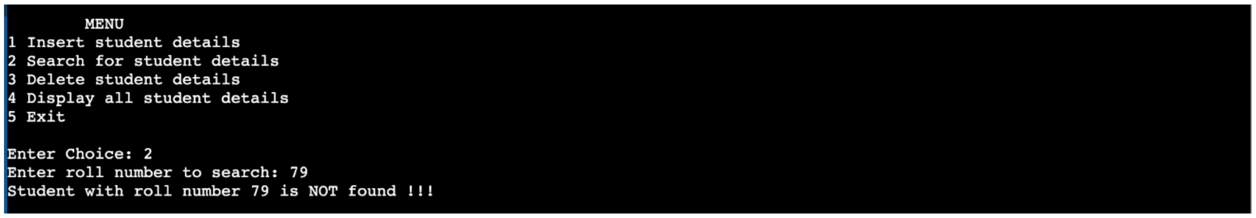


```
        MENU
1 Insert student details
2 Search for student details
3 Delete student details
4 Display all student details
5 Exit

Enter Choice: 3
Enter roll number to delete: 79

---------------------------------------------------------------------------------------------------
Record with roll number 79 Found !!!
Record Successfully Deleted !!!
---------------------------------------------------------------------------------------------------
```

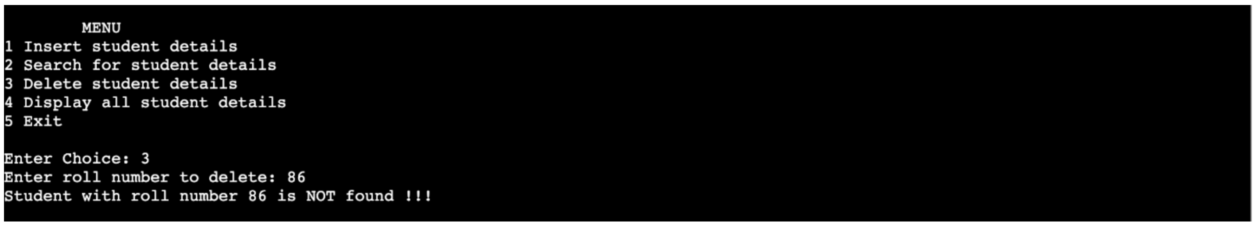
Validation :



```
        MENU
1 Insert student details
2 Search for student details
3 Delete student details
4 Display all student details
5 Exit

Enter Choice: 6

Enter a valid choice.
```



```
        MENU
1 Insert student details
2 Search for student details
3 Delete student details
4 Display all student details
5 Exit

Enter Choice: 2
Enter roll number to search: 79
Student with roll number 79 is NOT found !!!
```



```
        MENU
1 Insert student details
2 Search for student details
3 Delete student details
4 Display all student details
5 Exit

Enter Choice: 3
Enter roll number to delete: 86
Student with roll number 86 is NOT found !!!
```