SRM
INSTITUTE OF SCIENCE & TECHNOLOGY
Deemed to be University u/s 3 of UGC Act, 1956

STUDENT RECORD SYSTEM

MADHIRA SURYA SESHA SAI SRIKAR

RA2111002010006

MECHANICAL ENGINEERING

SUBMITTED TO

DR. R. RAJKUMAR

DSBS

SCHOOL OF COMPUTING

SRMIST

JANUARY 2022

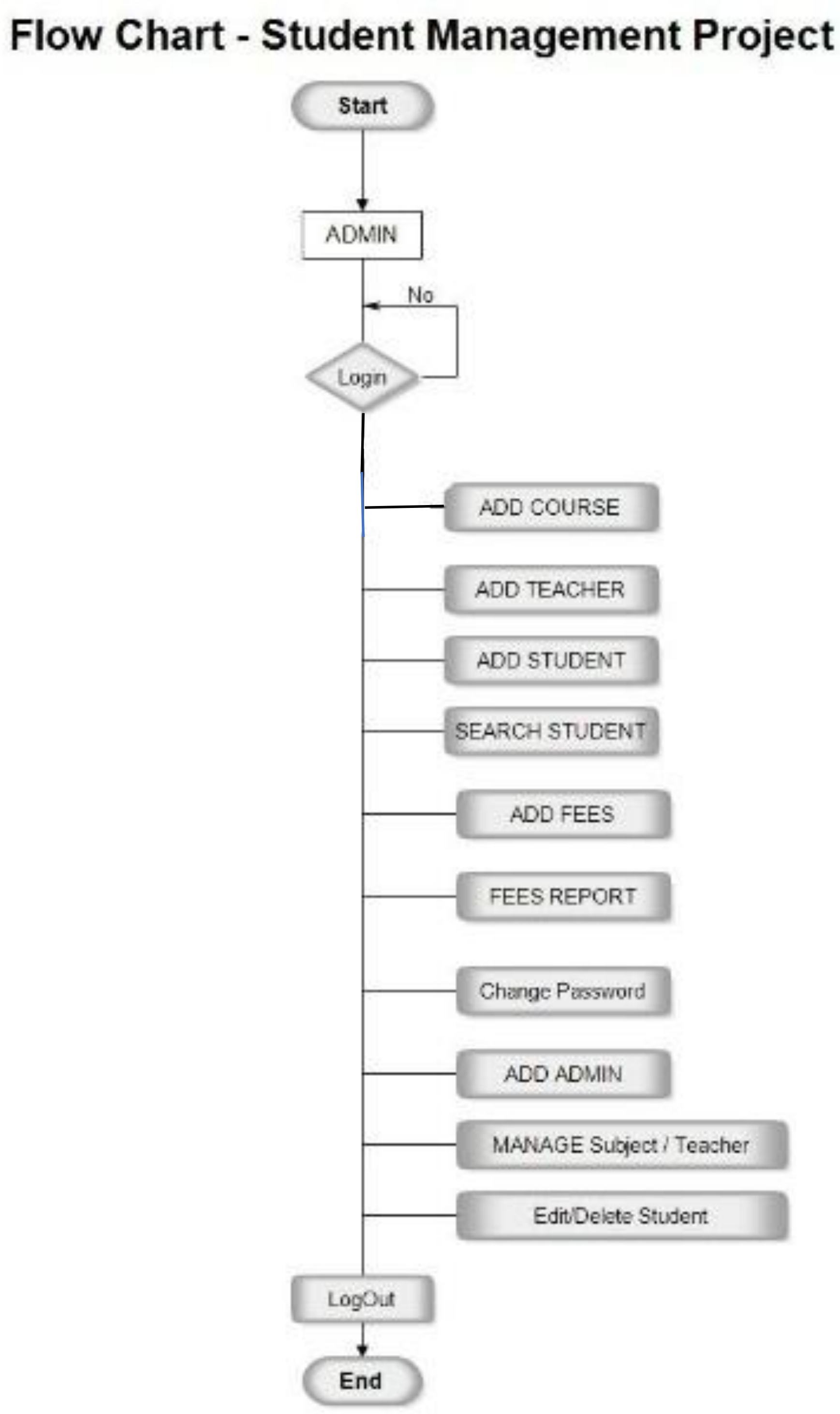
ABSRACT

Talking about the features of this Simple system, the user can perform the CRUD operations to it. Like, add student details by entering his/her name, roll number, age, address, parents name, class, school name, etc. The user can also view all the available student records. Besides, the user can edit information as well as remove a student's whole data. The system creates an external file to store the user's data permanently. This system is developed using C Programming Language and different variables, strings have been used for the development of it.

Simple functions have been to manipulate data structure and file handling, so here I will just list the features of this project.

- Add student record
- Search student record
- Modify student record
- Delete student record
- Display Record
- Top students Record
- Length of the Record

FLOW CHART


Flow Chart - Student Management Project
Start
ADMIN
No
Login
ADD COURSE
ADD TEACHER
ADD STUDENT
SEARCH STUDENT
ADD FEES
FEES REPORT
Change Password
ADD ADMIN
MANAGE Subject / Teacher
Edit/Delete Student
LogOut
End

PROGRAM:

```
/**
                        Student's Record
                ---------------------------------
**/



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



struct node
{
    int id;
    char stu_name[50], department[20];
    float university_current_result, university_overall_result;
    long long int stu_p_number, stu_fathers_p_number, stu_mothers_p_number;
    char present_address[200], permanent_address[200];
    char stu_fathers_name[50], stu_mothers_name[50];
    char stu_blood_group[7];
    char university_name[100], section[5], semester[5];
    char college_name[100], hsc_board[20];
    float hsc_gpa, ssc_gpa;
    int hsc_passing_year, ssc_passing_year;
    char school_name[100],ssc_board[20];
    char about_student_details[500];


    struct node *next;
    struct node *previous;

} *start = NULL, *end = NULL;
FILE *file;



/**    Shadhin    **/
void create_students_record()
{
    struct node *new_node, *current;
    int i, number_of_node;
```

```
printf("\n\n\n\n\n\n\t\t\tEnter Number of Student's for Record: ");
scanf("%d", &number_of_node);
cls();

for(i = 1; i <= number_of_node; i++)
{
   new_node = (struct node *)malloc(sizeof(struct node));

   if(new_node == NULL)
   {
      printf("\nMemory Does Not Created.\n");
      exit(0);
   }
   else
   {
      file = fopen("Student's Record.txt","a+");
      if(file == NULL)
      {
         printf("File does not create.\n");
      }
      else
      {
         printf("\n\n\t\t\tStudent's Details\n");
         fprintf(file,"\n\n\t\t\tStudent's Details\n");
         printf("\t\t___________________________________\n");
         fprintf(file,"\t\t___________________________________\n");

         printf("\n\n\tEnter Student Name: ");
         fflush(stdin);
         gets(new_node->stu_name);
         fprintf(file,"\n\tStudent Name: %s", new_node->stu_name);

         printf("\n\tID: ");
         scanf("%d", &new_node->id);
         fprintf(file,"\n\tID: %d",new_node->id);

         printf("\n\tSection: ");
         fflush(stdin);
         gets(new_node->section);
         fprintf(file,"\n\tSection: %s", new_node->section);

         printf("\n\tSemester: ");
         fflush(stdin);
         gets(new_node->semester);
         fprintf(file,"\n\tSemester: %s", new_node->semester);

         printf("\n\tDepartment: ");
         fflush(stdin);
         gets(new_node->department);
         fprintf(file,"\n\tDepartment: %s", new_node->department);

         printf("\n\tUniversity Name: ");
         fflush(stdin);
         gets(new_node->university_name);
```

```
            fprintf(file,"\n\tUniversity Name: %s", new_node->university_name);

            printf("\n\tStudent Blood Group: ");
            fflush(stdin);
            gets(new_node->stu_blood_group);
            fprintf(file,"\n\tStudent Blood Group: %s", new_node->stu_blood_group);

            printf("\n\tUniversity Current Semester Result: ");
            scanf("%f", &new_node->university_current_result);
            fprintf(file,"\n\tUniversity Current Semester Result: %0.2f",new_node-
>university_current_result);

            printf("\n\tUniversity Overall Result: ");
            scanf("%f", &new_node->university_overall_result);
            fprintf(file,"\n\tUniversity Overall Result: %0.2f",new_node->university_overall_result);

            printf("\n\n\n\t\t\tStudent's Background\n");
            fprintf(file,"\n\n\n\t\t\tStudent's Background\n");
            printf("\t\t_____________________________________");
            fprintf(file,"\t\t_____________________________________");

            printf("\n\n\tCollege Name: ");
            fflush(stdin);
            gets(new_node->college_name);
            fprintf(file,"\n\n\tCollege Name: %s", new_node->college_name);

            printf("\n\tH.S.C Passing Year: ");
            scanf("%d", &new_node->hsc_passing_year);
            fprintf(file,"\n\tH.S.C Passing Year: %d",new_node->hsc_passing_year);

            printf("\n\tH.S.C Result in GPA: ");
            scanf("%f", &new_node->hsc_gpa);
            fprintf(file,"\n\tH.S.C Result in GPA: %0.2f",new_node->hsc_gpa);

            printf("\n\n\tH.S.C Board: ");
            fflush(stdin);
            gets(new_node->hsc_board);
            fprintf(file,"\n\tH.S.C Board: %s", new_node->hsc_board);

            printf("\n\tSchool Name: ");
            fflush(stdin);
            gets(new_node->school_name);
            fprintf(file,"\n\tSchool Name: %s", new_node->school_name);

            printf("\n\tS.S.C Passing Year: ");
            scanf("%d", &new_node->ssc_passing_year);
            fprintf(file,"\n\tS.S.C Passing Year: %d",new_node->ssc_passing_year);

            printf("\n\tS.S.C Result in GPA: ");
            scanf("%f", &new_node->ssc_gpa);
            fprintf(file,"\n\tS.S.C Result in GPA: %0.2f",new_node->ssc_gpa);

            printf("\n\n\tS.S.C Board: ");
            fflush(stdin);
            gets(new_node->ssc_board);
```

```
        fprintf(file,"\n\tS.S.C Board: %s", new_node->ssc_board);

        printf("\n\n\n\t\t\tStudent's Contract Information\n");
        fprintf(file,"\n\n\n\t\t\tStudent's Contract Information\n");
        printf("\t\t______________________________________________");
        fprintf(file,"\t\t______________________________________________");

        printf("\n\n\tStudent's Phone Number: ");
        scanf("%lld", &new_node->stu_p_number);
        fprintf(file,"\n\n\tStudent's Phone Number: %lld",new_node->stu_p_number);

        printf("\n\tFather's Name: ");
        fflush(stdin);
        gets(new_node->stu_fathers_name);
        fprintf(file,"\n\tFather's Name: %s",new_node->stu_fathers_name);

        printf("\n\tMother's Name: ");
        fflush(stdin);
        gets(new_node->stu_mothers_name);
        fprintf(file,"\n\tMother's Name: %s",new_node->stu_mothers_name);

        printf("\n\tFather's Phone Number: ");
        scanf("%lld", &new_node->stu_fathers_p_number);
        fprintf(file,"\n\tFather's Phone Number: %lld",new_node->stu_fathers_p_number);

        printf("\n\tMother's Phone Number: ");
        scanf("%lld", &new_node->stu_mothers_p_number);
        fprintf(file,"\n\tMother's Phone Number: %lld",new_node->stu_mothers_p_number);


        printf("\n\n\tPresent Address: ");
        fflush(stdin);
        gets(new_node->present_address);
        fprintf(file,"\n\tPresent Address: %s",new_node->present_address);

        printf("\n\tPermanent Address: ");
        fflush(stdin);
        gets(new_node->permanent_address);
        fprintf(file,"\n\tPermanent Address: %s",new_node->permanent_address);

        printf("\n\tOther's Details and Comment: ");
        fflush(stdin);
        gets(new_node->about_student_details);
        fprintf(file,"\n\tOther's Details and Comment: %s",new_node->about_student_details);

        fclose(file);
        fopen("Student's Record.txt","a+");
    }

    new_node->next = NULL;
    new_node->previous = NULL;

    if(start == NULL && end == NULL)
    {
        start = new_node;
```

```
            end = new_node;
            current = new_node;
        }
        else
        {
            current->next = new_node;
            new_node->previous = current;
            current = new_node;
            end = new_node;
        }
    }
  }
}



/**    Shadhin    **/
void add_student_record_at_first()
{
    struct node *new_node, *current;
    new_node = (struct node *)malloc(sizeof(struct node));

    if(new_node == NULL)
    {
        printf("\nMemory Does Not Created.\n");
        exit(0);
    }
    else
    {
        file = fopen("Student's Record.txt","a+");
        if(file == NULL)
        {
            printf("File does not create.\n");
        }
        else
        {
            printf("\n\n\t\t\tStudent's Details\n");
            fprintf(file,"\n\n\t\t\tStudent's Details\n");
            printf("\t\t____________________________________\n");
            fprintf(file,"\t\t____________________________________\n");

            printf("\n\n\tEnter Student Name: ");
            fflush(stdin);
            gets(new_node->stu_name);
            fprintf(file,"\n\tStudent Name: %s", new_node->stu_name);

            printf("\n\tID: ");
            scanf("%d", &new_node->id);
            fprintf(file,"\n\tID: %d",new_node->id);

            printf("\n\tSection: ");
            fflush(stdin);
            gets(new_node->section);
            fprintf(file,"\n\tSection: %s", new_node->section);
```

```
printf("\n\tSemester: ");
fflush(stdin);
gets(new_node->semester);
fprintf(file,"\n\tSemester: %s", new_node->semester);

printf("\n\tDepartment: ");
fflush(stdin);
gets(new_node->department);
fprintf(file,"\n\tDepartment: %s", new_node->department);

printf("\n\tUniversity Name: ");
fflush(stdin);
gets(new_node->university_name);
fprintf(file,"\n\tUniversity Name: %s", new_node->university_name);

printf("\n\tStudent Blood Group: ");
fflush(stdin);
gets(new_node->stu_blood_group);
fprintf(file,"\n\tStudent Blood Group: %s", new_node->stu_blood_group);

printf("\n\tUniversity Current Semester Result: ");
scanf("%f", &new_node->university_current_result);
fprintf(file,"\n\tUniversity Current Semester Result: %0.2f",new_node->university_current_result);

printf("\n\tUniversity Overall Result: ");
scanf("%f", &new_node->university_overall_result);
fprintf(file,"\n\tUniversity Overall Result: %0.2f",new_node->university_overall_result);

printf("\n\n\n\t\t\tStudent's Background\n");
fprintf(file,"\n\n\n\t\t\tStudent's Background\n");
printf("\t\t______________________________________");
fprintf(file,"\t\t______________________________________");

printf("\n\n\tCollege Name: ");
fflush(stdin);
gets(new_node->college_name);
fprintf(file,"\n\n\tCollege Name: %s", new_node->college_name);

printf("\n\tH.S.C Passing Year: ");
scanf("%d", &new_node->hsc_passing_year);
fprintf(file,"\n\tH.S.C Passing Year: %d",new_node->hsc_passing_year);

printf("\n\tH.S.C Result in GPA: ");
scanf("%f", &new_node->hsc_gpa);
fprintf(file,"\n\tH.S.C Result in GPA: %0.2f",new_node->hsc_gpa);

printf("\n\tH.S.C Board: ");
fflush(stdin);
gets(new_node->hsc_board);
fprintf(file,"\n\tH.S.C Board: %s", new_node->hsc_board);

printf("\n\tSchool Name: ");
fflush(stdin);
gets(new_node->school_name);
fprintf(file,"\n\tSchool Name: %s", new_node->school_name);
```

```
printf("\n\tS.S.C Passing Year: ");
scanf("%d", &new_node->ssc_passing_year);
fprintf(file,"\n\tS.S.C Passing Year: %d",new_node->ssc_passing_year);

printf("\n\tS.S.C Result in GPA: ");
scanf("%f", &new_node->ssc_gpa);
fprintf(file,"\n\tS.S.C Result in GPA: %0.2f",new_node->ssc_gpa);

printf("\n\tS.S.C Board: ");
fflush(stdin);
gets(new_node->ssc_board);
fprintf(file,"\n\tS.S.C Board: %s", new_node->ssc_board);

printf("\n\n\n\t\t\tStudent's Contract Information\n");
fprintf(file,"\n\n\n\t\t\tStudent's Contract Information\n");
printf("\t\t_____________________________________________");
fprintf(file,"\t\t_____________________________________________");

printf("\n\n\tStudent's Phone Number: ");
scanf("%lld", &new_node->stu_p_number);
fprintf(file,"\n\n\tStudent's Phone Number: %lld",new_node->stu_p_number);

printf("\n\tFather's Name: ");
fflush(stdin);
gets(new_node->stu_fathers_name);
fprintf(file,"\n\tFather's Name: %s",new_node->stu_fathers_name);

printf("\n\tMother's Name: ");
fflush(stdin);
gets(new_node->stu_mothers_name);
fprintf(file,"\n\tMother's Name: %s",new_node->stu_mothers_name);

printf("\n\tFather's Phone Number: ");
scanf("%lld", &new_node->stu_fathers_p_number);
fprintf(file,"\n\tFather's Phone Number: %lld",new_node->stu_fathers_p_number);

printf("\n\tMother's Phone Number: ");
scanf("%lld", &new_node->stu_mothers_p_number);
fprintf(file,"\n\tMother's Phone Number: %lld",new_node->stu_mothers_p_number);


printf("\n\tPresent Address: ");
fflush(stdin);
gets(new_node->present_address);
fprintf(file,"\n\tPresent Address: %s",new_node->present_address);

printf("\n\tPermanent Address: ");
fflush(stdin);
gets(new_node->permanent_address);
fprintf(file,"\n\tPermanent Address: %s",new_node->permanent_address);

printf("\n\tOther's Details and Comment: ");
fflush(stdin);
gets(new_node->about_student_details);
```

```
            fprintf(file,"\n\tOther's Details and Comment: %s",new_node->about_student_details);

            fclose(file);
            fopen("Student's Record.txt","a+");
        }
    }

    new_node->next = NULL;
    new_node->previous = NULL;

    current = start;
    new_node->next = current;
    current->previous = new_node;
    start = new_node;
}



/**    Shadhin    **/
void add_student_record_at_last()
{
    struct node *new_node, *current;
    new_node = (struct node *)malloc(sizeof(struct node));

    if(new_node == NULL)
    {
        printf("\nMemory Does Not Created.\n");
        exit(0);
    }
    else
    {
        file = fopen("Student's Record.txt","a+");
        if(file == NULL)
        {
            printf("File does not create.\n");
        }
        else
        {
            printf("\n\n\t\t\tStudent's Details\n");
            fprintf(file,"\n\n\t\t\tStudent's Details\n");
            printf("\t\t___________________________________\n");
            fprintf(file,"\t\t___________________________________\n");

            printf("\n\n\tEnter Student Name: ");
            fflush(stdin);
            gets(new_node->stu_name);
            fprintf(file,"\n\tStudent Name: %s", new_node->stu_name);

            printf("\n\tID: ");
            scanf("%d", &new_node->id);
            fprintf(file,"\n\tID: %d",new_node->id);

            printf("\n\tSection: ");
            fflush(stdin);
            gets(new_node->section);
```

```
fprintf(file,"\n\tSection: %s", new_node->section);

printf("\n\tSemester: ");
fflush(stdin);
gets(new_node->semester);
fprintf(file,"\n\tSemester: %s", new_node->semester);

printf("\n\tDepartment: ");
fflush(stdin);
gets(new_node->department);
fprintf(file,"\n\tDepartment: %s", new_node->department);

printf("\n\tUniversity Name: ");
fflush(stdin);
gets(new_node->university_name);
fprintf(file,"\n\tUniversity Name: %s", new_node->university_name);

printf("\n\tStudent Blood Group: ");
fflush(stdin);
gets(new_node->stu_blood_group);
fprintf(file,"\n\tStudent Blood Group: %s", new_node->stu_blood_group);

printf("\n\tUniversity Current Semester Result: ");
scanf("%f", &new_node->university_current_result);
fprintf(file,"\n\tUniversity Current Semester Result: %0.2f",new_node->university_current_result);

printf("\n\tUniversity Overall Result: ");
scanf("%f", &new_node->university_overall_result);
fprintf(file,"\n\tUniversity Overall Result: %0.2f",new_node->university_overall_result);

printf("\n\n\n\t\t\tStudent's Background\n");
fprintf(file,"\n\n\n\t\t\tStudent's Background\n");
printf("\t\t______________________________________");
fprintf(file,"\t\t______________________________________");

printf("\n\n\tCollege Name: ");
fflush(stdin);
gets(new_node->college_name);
fprintf(file,"\n\n\tCollege Name: %s", new_node->college_name);

printf("\n\tH.S.C Passing Year: ");
scanf("%d", &new_node->hsc_passing_year);
fprintf(file,"\n\tH.S.C Passing Year: %d",new_node->hsc_passing_year);

printf("\n\tH.S.C Result in GPA: ");
scanf("%f", &new_node->hsc_gpa);
fprintf(file,"\n\tH.S.C Result in GPA: %0.2f",new_node->hsc_gpa);

printf("\n\tH.S.C Board: ");
fflush(stdin);
gets(new_node->hsc_board);
fprintf(file,"\n\tH.S.C Board: %s", new_node->hsc_board);

printf("\n\tSchool Name: ");
fflush(stdin);
```

```
gets(new_node->school_name);
fprintf(file,"\n\tSchool Name: %s", new_node->school_name);

printf("\n\tS.S.C Passing Year: ");
scanf("%d", &new_node->ssc_passing_year);
fprintf(file,"\n\tS.S.C Passing Year: %d",new_node->ssc_passing_year);

printf("\n\tS.S.C Result in GPA: ");
scanf("%f", &new_node->ssc_gpa);
fprintf(file,"\n\tS.S.C Result in GPA: %0.2f",new_node->ssc_gpa);

printf("\n\tS.S.C Board: ");
fflush(stdin);
gets(new_node->ssc_board);
fprintf(file,"\n\tS.S.C Board: %s", new_node->ssc_board);

printf("\n\n\n\t\t\tStudent's Contract Information\n");
fprintf(file,"\n\n\n\t\t\tStudent's Contract Information\n");
printf("\t\t_________________________________________________");
fprintf(file,"\t\t_________________________________________________");

printf("\n\n\tStudent's Phone Number: ");
scanf("%lld", &new_node->stu_p_number);
fprintf(file,"\n\n\tStudent's Phone Number: %lld",new_node->stu_p_number);

printf("\n\tFather's Name: ");
fflush(stdin);
gets(new_node->stu_fathers_name);
fprintf(file,"\n\tFather's Name: %s",new_node->stu_fathers_name);

printf("\n\tMother's Name: ");
fflush(stdin);
gets(new_node->stu_mothers_name);
fprintf(file,"\n\tMother's Name: %s",new_node->stu_mothers_name);

printf("\n\tFather's Phone Number: ");
scanf("%lld", &new_node->stu_fathers_p_number);
fprintf(file,"\n\tFather's Phone Number: %lld",new_node->stu_fathers_p_number);

printf("\n\tMother's Phone Number: ");
scanf("%lld", &new_node->stu_mothers_p_number);
fprintf(file,"\n\tMother's Phone Number: %lld",new_node->stu_mothers_p_number);


printf("\n\tPresent Address: ");
fflush(stdin);
gets(new_node->present_address);
fprintf(file,"\n\tPresent Address: %s",new_node->present_address);

printf("\n\tPermanent Address: ");
fflush(stdin);
gets(new_node->permanent_address);
fprintf(file,"\n\tPermanent Address: %s",new_node->permanent_address);

printf("\n\tOther's Details and Comment: ");
```

```
            fflush(stdin);
            gets(new_node->about_student_details);
            fprintf(file,"\n\tOther's Details and Comment: %s",new_node->about_student_details);

            fclose(file);
            fopen("Student's Record.txt","a+");
        }
    }

    new_node->next = NULL;
    new_node->previous = NULL;


    current = end;
    current->next = new_node;
    new_node->previous = current;
    end = new_node;
}



/**    Shadhin    **/
void add_student_record_at_middle()
{
    struct node *new_node, *current, *temp1, *temp2;
    int i,student_id,position;

    new_node = (struct node *)malloc(sizeof(struct node));

    if(new_node == NULL)
    {
        printf("\nMemory Does Not Created.\n");
        exit(0);
    }
    else
    {
        file = fopen("Student's Record.txt","a+");
        if(file == NULL)
        {
            printf("File does not create.\n");
        }
        else
        {
            printf("\n\n\t\t\tStudent's Details\n");
            fprintf(file,"\n\n\t\t\tStudent's Details\n");
            printf("\t\t____________________________________\n");
            fprintf(file,"\t\t____________________________________\n");

            printf("\n\n\tEnter Student Name: ");
            fflush(stdin);
            gets(new_node->stu_name);
            fprintf(file,"\n\tStudent Name: %s", new_node->stu_name);

            printf("\n\tID: ");
            scanf("%d", &new_node->id);
```

```
fprintf(file,"\n\tID: %d",new_node->id);

printf("\n\tSection: ");
fflush(stdin);
gets(new_node->section);
fprintf(file,"\n\tSection: %s", new_node->section);

printf("\n\tSemester: ");
fflush(stdin);
gets(new_node->semester);
fprintf(file,"\n\tSemester: %s", new_node->semester);

printf("\n\tDepartment: ");
fflush(stdin);
gets(new_node->department);
fprintf(file,"\n\tDepartment: %s", new_node->department);

printf("\n\tUniversity Name: ");
fflush(stdin);
gets(new_node->university_name);
fprintf(file,"\n\tUniversity Name: %s", new_node->university_name);

printf("\n\tStudent Blood Group: ");
fflush(stdin);
gets(new_node->stu_blood_group);
fprintf(file,"\n\tStudent Blood Group: %s", new_node->stu_blood_group);

printf("\n\tUniversity Current Semester Result: ");
scanf("%f", &new_node->university_current_result);
fprintf(file,"\n\tUniversity Current Semester Result: %0.2f",new_node->university_current_result);

printf("\n\tUniversity Overall Result: ");
scanf("%f", &new_node->university_overall_result);
fprintf(file,"\n\tUniversity Overall Result: %0.2f",new_node->university_overall_result);

printf("\n\n\n\t\t\tStudent's Background\n");
fprintf(file,"\n\n\n\t\t\tStudent's Background\n");
printf("\t\t________________________________________");
fprintf(file,"\t\t________________________________________");

printf("\n\n\tCollege Name: ");
fflush(stdin);
gets(new_node->college_name);
fprintf(file,"\n\n\tCollege Name: %s", new_node->college_name);

printf("\n\tH.S.C Passing Year: ");
scanf("%d", &new_node->hsc_passing_year);
fprintf(file,"\n\tH.S.C Passing Year: %d",new_node->hsc_passing_year);

printf("\n\tH.S.C Result in GPA: ");
scanf("%f", &new_node->hsc_gpa);
fprintf(file,"\n\tH.S.C Result in GPA: %0.2f",new_node->hsc_gpa);

printf("\n\tH.S.C Board: ");
fflush(stdin);
```

```
gets(new_node->hsc_board);
fprintf(file,"\n\tH.S.C Board: %s", new_node->hsc_board);

printf("\n\tSchool Name: ");
fflush(stdin);
gets(new_node->school_name);
fprintf(file,"\n\tSchool Name: %s", new_node->school_name);

printf("\n\tS.S.C Passing Year: ");
scanf("%d", &new_node->ssc_passing_year);
fprintf(file,"\n\tS.S.C Passing Year: %d",new_node->ssc_passing_year);

printf("\n\tS.S.C Result in GPA: ");
scanf("%f", &new_node->ssc_gpa);
fprintf(file,"\n\tS.S.C Result in GPA: %0.2f",new_node->ssc_gpa);

printf("\n\tS.S.C Board: ");
fflush(stdin);
gets(new_node->ssc_board);
fprintf(file,"\n\tS.S.C Board: %s", new_node->ssc_board);

printf("\n\n\n\t\t\tStudent's Contract Information\n");
fprintf(file,"\n\n\n\t\t\tStudent's Contract Information\n");
printf("\t\t______________________________________________");
fprintf(file,"\t\t______________________________________________");

printf("\n\n\tStudent's Phone Number: ");
scanf("%lld", &new_node->stu_p_number);
fprintf(file,"\n\n\tStudent's Phone Number: %lld",new_node->stu_p_number);

printf("\n\tFather's Name: ");
fflush(stdin);
gets(new_node->stu_fathers_name);
fprintf(file,"\n\tFather's Name: %s",new_node->stu_fathers_name);

printf("\n\tMother's Name: ");
fflush(stdin);
gets(new_node->stu_mothers_name);
fprintf(file,"\n\tMother's Name: %s",new_node->stu_mothers_name);

printf("\n\tFather's Phone Number: ");
scanf("%lld", &new_node->stu_fathers_p_number);
fprintf(file,"\n\tFather's Phone Number: %lld",new_node->stu_fathers_p_number);

printf("\n\tMother's Phone Number: ");
scanf("%lld", &new_node->stu_mothers_p_number);
fprintf(file,"\n\tMother's Phone Number: %lld",new_node->stu_mothers_p_number);


printf("\n\tPresent Address: ");
fflush(stdin);
gets(new_node->present_address);
fprintf(file,"\n\tPresent Address: %s",new_node->present_address);

printf("\n\tPermanent Address: ");
```

```
            fflush(stdin);
            gets(new_node->permanent_address);
            fprintf(file,"\n\tPermanent Address: %s",new_node->permanent_address);

            printf("\n\tOther's Details and Comment: ");
            fflush(stdin);
            gets(new_node->about_student_details);
            fprintf(file,"\n\tOther's Details and Comment: %s",new_node->about_student_details);

            fclose(file);
            fopen("Student's Record.txt","a+");
        }
    }

    new_node->next = NULL;
    new_node->previous = NULL;

    cls();

    printf("Enter ID before Insert Record: ");
    scanf("%d", &student_id);

    position = pos(student_id);

    current = start;
    for(i = 1; i <= (position - 1); i++)
    {
        current = current->next;
    }

    temp2 = current;
    temp1 = current->previous;
    temp1->next = new_node;
    new_node->previous = temp1;
    new_node->next = temp2;
    temp2->previous = new_node;
}



/**    Mehedi    **/
void display_students_record_from_forward()
{
    struct node *current;

    current = start;

    if(current == NULL)
    {
        printf("\n\n\n\n\n\n\n\t\t\tThere Are No Record In The List.\n");
    }
    else
    {
        while(current != NULL)
        {
```

```
printf("\n\t\t\tStudent's Details\n");
printf("\t\t____________________________________");

printf("\n\tStudent Name: %s", current->stu_name);

printf("\n\tID: %d", current->id);

printf("\n\tSection: %s", current->section);

printf("\n\tSemester: %s", current->semester);

printf("\n\tDepartment: %s", current->department);

printf("\n\tUniversity Name: %s", current->university_name);

printf("\n\tStudent Blood Group: %s", current->stu_blood_group);

printf("\n\tUniversity Current Semester Result: %0.2f", current->university_current_result);

printf("\n\tUniversity Overall Result: %0.2f", current->university_overall_result);


printf("\n\n\n\t\t\tStudent's Background\n");
printf("\t\t________________________________________");


printf("\n\tCollege Name: %s", current->college_name);

printf("\n\tH.S.C Passing Year: %d", current->hsc_passing_year);

printf("\n\tH.S.C Result in GPA: %0.2f", current->hsc_gpa);

printf("\n\tH.S.C Board: %s", current->hsc_board);

printf("\n\tSchool Name: %s", current->school_name);

printf("\n\tS.S.C Passing Year: %d", current->ssc_passing_year);

printf("\n\tS.S.C Result in GPA: %0.2f", current->ssc_gpa);

printf("\n\tS.S.C Board: %s", current->ssc_board);


printf("\n\n\n\t\t\tStudent's Contract Information\n");
printf("\t\t__________________________________________________");


printf("\n\tStudent's Phone Number: %lld", current->stu_p_number);

printf("\n\tFather's Name: %s", current->stu_fathers_name);

printf("\n\tMothers Name: %s", current->stu_mothers_name);
```

```
            printf("\n\tFather's Phone Number: %lld", current->stu_fathers_p_number);

            printf("\n\tMother's Phone Number: %lld", current->stu_mothers_p_number);

            printf("\n\tPresent Address: %s", current->present_address);

            printf("\n\tPermanent Address: %s", current->permanent_address);

            printf("\n\tOther's Details and Comment: %s", current->about_student_details);



            current = current->next;
            printf("\n");
        }
    }
}



/**    Mehedi    **/
void display_students_record_from_backward()
{
    struct node *current;

    current = end;

    if(current == NULL)
    {
        printf("\n\n\n\n\n\n\n\n\t\t\tThere Are No Record In The List.\n");
    }
    else
    {
        while(current != NULL)
        {
            printf("\n\t\t\tStudent's Details\n");
            printf("\t\t___________________________________");

            printf("\n\tStudent Name: %s", current->stu_name);

            printf("\n\tID: %d", current->id);

            printf("\n\tSection: %s", current->section);

            printf("\n\tSemester: %s", current->semester);

            printf("\n\tDepartment: %s", current->department);

            printf("\n\tUniversity Name: %s", current->university_name);

            printf("\n\tStudent Blood Group: %s", current->stu_blood_group);

            printf("\n\tUniversity Current Semester Result: %0.2f", current->university_current_result);
            printf("\n\tUniversity Overall Result: %0.2f", current->university_overall_result);



            printf("\n\n\n\t\t\tStudent's Background\n");
            printf("\t\t_______________________________________");


            printf("\n\tCollege Name: %s", current->college_name);

            printf("\n\tH.S.C Passing Year: %d", current->hsc_passing_year);

            printf("\n\tH.S.C Result in GPA: %0.2f", current->hsc_gpa);

            printf("\n\tH.S.C Board: %s", current->hsc_board);

            printf("\n\tSchool Name: %s", current->school_name);

            printf("\n\tS.S.C Passing Year: %d", current->ssc_passing_year);

            printf("\n\tS.S.C Result in GPA: %0.2f", current->ssc_gpa);

            printf("\n\tS.S.C Board: %s", current->ssc_board);



            printf("\n\n\n\t\t\tStudent's Contract Information\n");
            printf("\t\t_________________________________________________");



            printf("\n\tStudent's Phone Number: %lld", current->stu_p_number);

            printf("\n\tFather's Name: %s", current->stu_fathers_name);

            printf("\n\tMothers Name: %s", current->stu_mothers_name);

            printf("\n\tFather's Phone Number: %lld", current->stu_fathers_p_number);

            printf("\n\tMother's Phone Number: %lld", current->stu_mothers_p_number);

            printf("\n\tPresent Address: %s", current->present_address);

            printf("\n\tPermanent Address: %s", current->permanent_address);

            printf("\n\tOther's Details and Comment: %s", current->about_student_details);



            current = current->previous;
            printf("\n");
        }
    }
}
```

```
/**    Shadhin    **/
int pos(student_id)
{
    int position = 0;
    struct node *current;

    current = start;

    while(current != NULL)
    {
        position++;
        if(student_id == current->id)
        {
            return position;
        }
        current = current->next;
    }
    return -1;
}



/**    Antor    **/
int search_students_record(student_id)
{
    int position = 0;
    struct node *current;

    current = start;

    while(current != NULL)
    {
        position++;
        if(student_id == current->id)
        {
            printf("\n\t\t\tStudent's Details\n");
            printf("\t\t___________________________________");

            printf("\n\tStudent Name: %s", current->stu_name);

            printf("\n\tID: %d", current->id);

            printf("\n\tSection: %s", current->section);

            printf("\n\tSemester: %s", current->semester);

            printf("\n\tDepartment: %s", current->department);

            printf("\n\tUniversity Name: %s", current->university_name);

            printf("\n\tStudent Blood Group: %s", current->stu_blood_group);

            printf("\n\tUniversity Current Semester Result: %0.2f", current->university_current_result);
    printf("\n\tUniversity Overall Result: %0.2f", current->university_overall_result);



    printf("\n\n\n\t\t\tStudent's Background\n");
    printf("\t\t_____________________________________");


    printf("\n\tCollege Name: %s", current->college_name);

    printf("\n\tH.S.C Passing Year: %d", current->hsc_passing_year);

    printf("\n\tH.S.C Result in GPA: %0.2f", current->hsc_gpa);

    printf("\n\tH.S.C Board: %s", current->hsc_board);

    printf("\n\tSchool Name: %s", current->school_name);

    printf("\n\tS.S.C Passing Year: %d", current->ssc_passing_year);

    printf("\n\tS.S.C Result in GPA: %0.2f", current->ssc_gpa);

    printf("\n\tS.S.C Board: %s", current->ssc_board);



    printf("\n\n\n\t\t\tStudent's Contract Information\n");
    printf("\t\t______________________________________________");



    printf("\n\tStudent's Phone Number: %lld", current->stu_p_number);

    printf("\n\tFather's Name: %s", current->stu_fathers_name);

    printf("\n\tMothers Name: %s", current->stu_mothers_name);

    printf("\n\tFather's Phone Number: %lld", current->stu_fathers_p_number);

    printf("\n\tMother's Phone Number: %lld", current->stu_mothers_p_number);

    printf("\n\tPresent Address: %s", current->present_address);

    printf("\n\tPermanent Address: %s", current->permanent_address);

    printf("\n\tOther's Details and Comment: %s", current->about_student_details);


    printf("\n\n");
    getch();
    cls();

    return position;
}
```

```
        current = current->next;
    }
    return -1;
}



/**    Shukdeb    **/
void delete_students_record()
{
    struct node *current, *temp1, *temp2;
    int i, delet_id, position;

    printf("Enter ID for delete: ");
    scanf("%d", &delet_id);
    position = pos(delet_id);

    current = start;
    for(i = 1; i <= (position - 1); i++)
    {
        current = current->next;
    }
    if(current == start && current->previous == NULL)
    {
        current = current->next;
        start = current;
        current->previous = NULL;
        printf("\nFirst ID Delete Successfully.\n");
    }
    else if(current->next == NULL && current == end)
    {
        current = current->previous;
        end = current;
        current->next = NULL;
        printf("\nLast ID Delete Successfully.\n");
    }
    else
    {
        temp2 = current->next;
        temp1 = current->previous;
        temp1->next = temp2;
        temp2->previous = temp1;
        printf("\nDelete Successfully.\n");
    }
}



/**    Shadhin    **/
void length_of_the_students_record()
{
    struct node *current;
    int count = 0;
    current = start;

    while(current != NULL)
    {
        count++;
        current = current->next;
    }

    printf("\n\n\n\n\n\n\n\n\t\t\tThere are %d number of Student's in the Record.", count);
    printf("\n                                        And the length is %d.\n",count);
}



void cls()
{
    system("cls");
}



int main()
{
    system("color A");

    printf("\n\n\n\n\n\n\n\n\n\t\t\t\t\t|_____________________|\n\t\t\t\t\t|                     |\n\t\t\t\t\t|\tWelcome
|\n\t\t\t\t\t|_____________________|\n\t\t\t\t\t|\t\t     |\n\n\n");
    getch();
    cls();

    int choice, position, student_id, insert_choice, display_choice;

    while(1)
    {
        printf("\n\t\t\t\t\tStudent's Record\n");
        printf("\t\t\t__________________________________________________");
        printf("\n\n");

        printf("--------------------------------------------------------------------------------------------");
        printf("\n\n");
        printf("\t\t\t\t1. Create Record.\n");
        printf("\t\t\t\t2. Add Record.\n");
        printf("\t\t\t\t3. Display Record.\n");
        printf("\t\t\t\t4. Search Record.\n");
        printf("\t\t\t\t5. Delete Record.\n");
        printf("\t\t\t\t6. Modify Record.\n");
        printf("\t\t\t\t7. Length of the Record.\n");
        printf("\t\t\t\t8. Top Student's Record.\n");
        printf("\t\t\t\t9. Exit.\n\n");
        printf("--------------------------------------------------------------------------------------------");

        printf("\n\t\t\tPlease Enter your choice: ");
        scanf("%d", &choice);
        printf("\n");
```

```
cls();

switch(choice)
{
    case 1: create_students_record();
            cls();
            printf("\n\n\n\n\n\n\n\t\t\tRecord Created Successfully.");
            getch();
            cls();
            break;


    case 2: while(1)
            {
                printf("\n\n\n\n\n");
                printf("----------------------------------------------------------------------------------------");
                printf("\n\n");
                printf("\t\t\t\t1. Add Record At First.\n");
                printf("\t\t\t\t2. Add Record At Last.\n");
                printf("\t\t\t\t3. Add Record At Middle.\n");
                printf("\t\t\t\t4. Back To Main Program.\n\n");
                printf("----------------------------------------------------------------------------------------");

                printf("\n\n\n\t\t\tPlease Enter your choice: ");
                scanf("%d", &insert_choice);
                printf("\n\n");

                cls();

                switch(insert_choice)
                {
                    case 1: add_student_record_at_first();
                            cls();
                            printf("\n\n\n\n\n\n\n\t\t\tRecord Added successfully.");
                            getch();
                            cls();
                            break;

                    case 2: add_student_record_at_last();
                            cls();
                            printf("\n\n\n\n\n\n\n\t\t\tRecord Added successfully.");
                            getch();
                            cls();
                            break;

                    case 3: add_student_record_at_middle();
                            cls();
                            printf("\n\n\n\n\n\n\n\t\t\tRecord Added successfully.");
                            getch();
                            cls();
                            break;

                    case 4: main();
                            break;
```

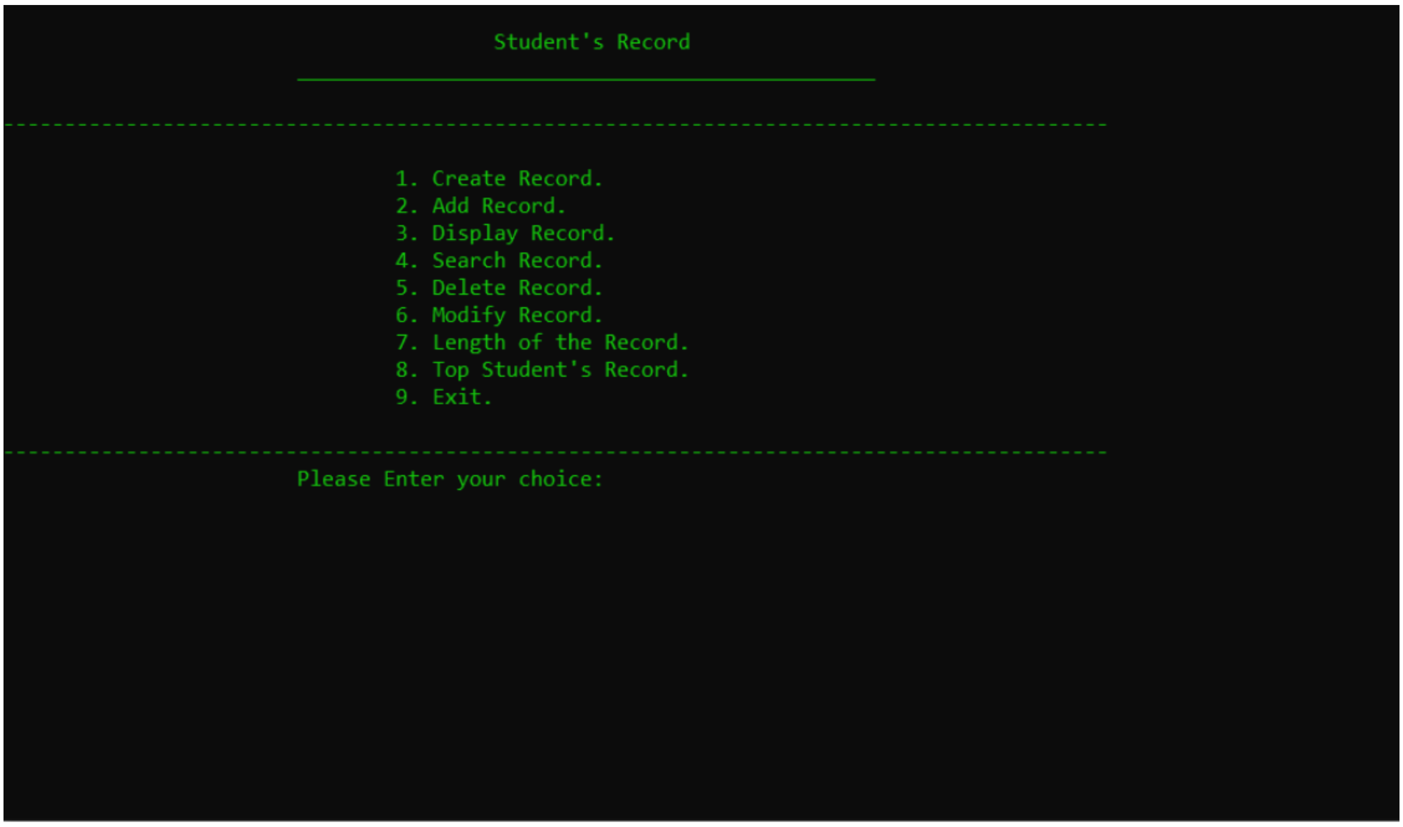
```
        default:printf("\n\n\n\n\n\n\n\t\t\t\tInvalid input!!!");
            printf("\n\t\t\tPlease Enter Correct Key to Access.");
            printf("\n\t\t\t\tOr Enter 4 to Main menu.\n");
            getch();
            cls();
      }
    }
    getch();
    cls();
    break;


case 3: while(1)
    {
      printf("\n\n\n\n\n");
      printf("------------------------------------------------------------------------------------------");
      printf("\n\n");
      printf("\t\t\t\t1. Display From Forward.\n");
      printf("\t\t\t\t2. Display From Backward.\n");
      printf("\t\t\t\t3. Back To Main Program.\n\n");
      printf("------------------------------------------------------------------------------------------");

      printf("\n\n\n\t\t\tPlease Enter your choice: ");
      scanf("%d", &display_choice);
      printf("\n\n");

      cls();

      switch(display_choice)
      {
        case 1: display_students_record_from_forward();
            getch();
            cls();
            break;

        case 2: display_students_record_from_backward();
            getch();
            cls();
            break;

        case 3: main();
            break;

        default:printf("\n\n\n\n\n\n\n\t\t\t\tInvalid input!!!");
            printf("\n\t\t\tPlease Enter Correct Key to Access.");
            printf("\n\t\t\t\tOr Enter 3 to Main menu.\n");
            getch();
            cls();
      }
    }
    getch();
    cls();
    break;
```

```
            case 4: printf("\n\n\n\n\n\n\t\t\tEnter ID: ");
                    scanf("%d",&student_id);
                    cls();

                    position = search_students_record(student_id);
                    if(position == -1)
                    {
                       printf("\n\n\n\n\n\n\t\t\tThis ID is not in the Record.\n");
                       getch();
                       cls();
                    }
                    else
                    {
                       printf("\n\n\n\n\n\n\t\t\tThe Position of this Record is at Number %d.\n", position);
                       getch();
                       cls();
                    }
                    break;


            case 5: delete_students_record();
                    getch();
                    cls();
                    break;


            case 6: break;


            case 7: length_of_the_students_record();
                    getch();
                    cls();
                    break;


            case 8: break;


            case 9: exit(1);
                    break;


            default:printf("\n\n\n\n\n\n\n\t\t\t\tInvalid input!!!");
                    printf("\n\t\t\tPlease Enter Correct Key to Access.");
                    printf("\n\t\t\t\tOr Enter 8 to Exit.\n");
                    getch();
                    cls();
         }
      }
   }
```

RESULTS

The code has been tested successfully.

SCREENSHOTS



```
                              Student's Record
                        _______________________________________

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

                                1. Create Record.
                                2. Add Record.
                                3. Display Record.
                                4. Search Record.
                                5. Delete Record.
                                6. Modify Record.
                                7. Length of the Record.
                                8. Top Student's Record.
                                9. Exit.

------------------------------------------------------------------------------------
                        Please Enter your choice:
```

DECLARATION

I, Madhira Surya Sesha Sai srikar, solemnly declare that mini-project STUDENT RECORD SYSTEM is based on the knowledge gathered during the course of our study under the guidance of DR. R. RAJKUMAR SIR, DSBS, SCHOOL OF COMPUTING, SRM INSTITUTE OF SCIENCE AND TECHNOLOGY, and this project work is submitted in the partial fulfilment of the requirements of for the award of the degree of Bachelor of Technology in Mechanical Engineering.

RERERENCES

The following reference has been used as per the guideline given:-
https://www.codewithc.com