STUDENT MANAGEMENT SYSTEM

IC 1302 – Programming I
Level 1 – Semester 1

Jeyakrishnan Jeyapriya

2021/T/01213

Department of Information and Communication Technology
Faculty of Technology
University of Colombo

TABLE OF CONTENTS

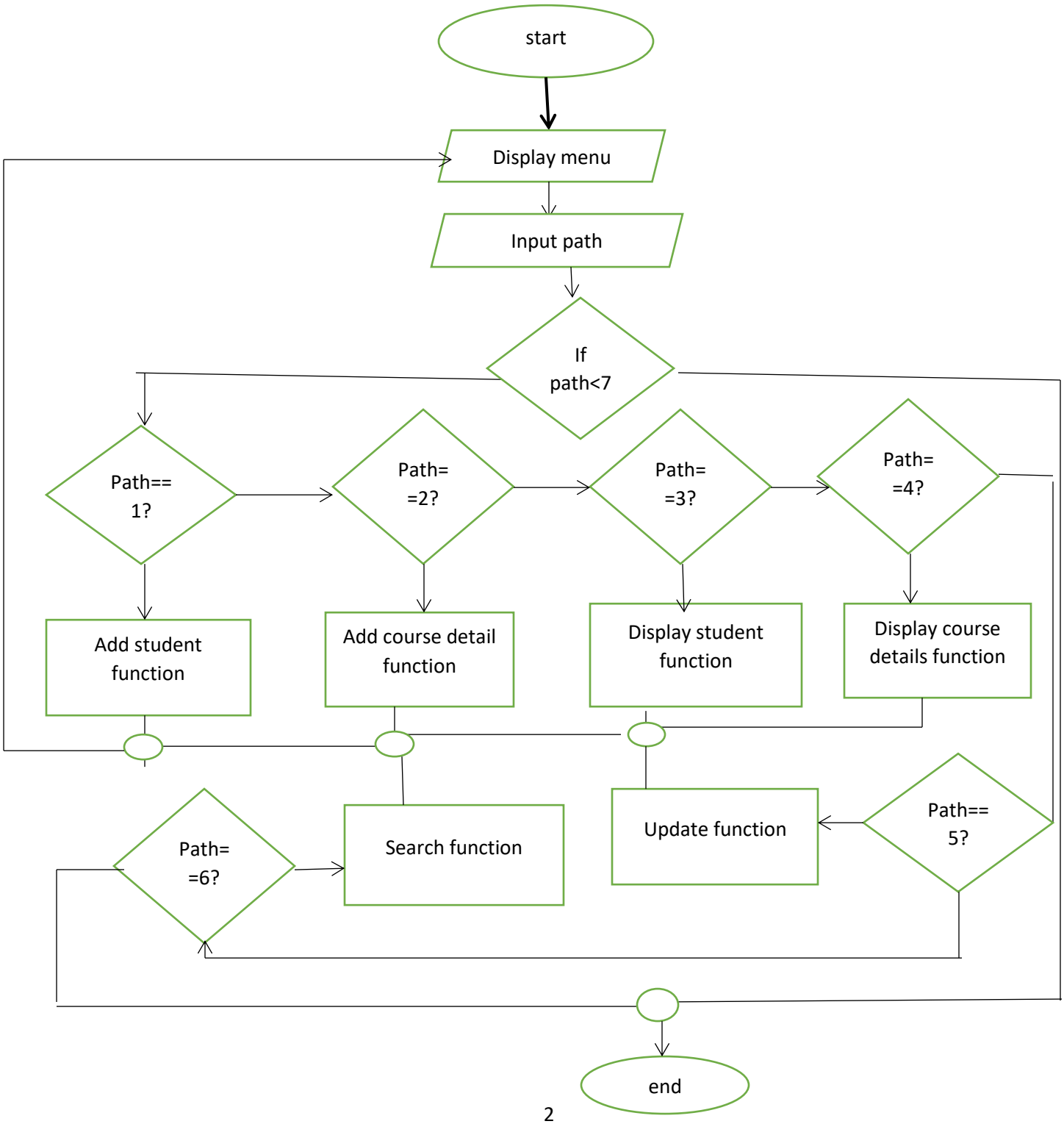
Introduction

This project is creating a Student Management System developed using c language to store information of students. In this console application is a solution tool that is designed to track, maintain and manage all the data generated by a School, University and institutes, including the grades of a student.

The student management system is used to manage the entire student's data. This project is used for making the process of managing information easier and to also make it accessible. To storing information, use binary file as a database in the project. This information could be the general details like student name, department, index number, and their grades for each course which is following or specific information like collection of data. The Student Management System project cover functionalities of add new student details, update student details, enter new course details, find student details and course details and list students information. Also to increase the readability, I have broken the application in different functions. Each function of the project extensively use in the file handing function, so it is also a great project to understand file handling in C.

Methodology

To develop this project mainly used notepad++ to write code used GCC compiler to compile codes. To find some code parts, error solutions to used internet.



Main menu

This is the first screen of the Student Management System. User can choose the menu item by inputting the menu number. Menu will be displayed again and again when a function is accomplished. There are 7 menu items in the menu. User can select the one item in the main menu. Ask the user to select the option. If the user selects 7, then the application will closed.

1. ADD NEW STUDENT DETAILS - Add to the new student data in the student file
2. ADD NEW COURSE DETAILS - Add to new course details in the course file
3. DISPLAY STUDENT DETAILS – Display to the student details in the terminal as a report
4. DISPLAY COURSE DETAILS - Display the course details in the terminal
5. UPDATE STUDENT DETAILS – Update the student records
6. SEARCH EACH STUDENT DETAILS – Search each student details and display in the terminal
7. EXIT THE ENTIRE PROGRAM – To exit the entire program

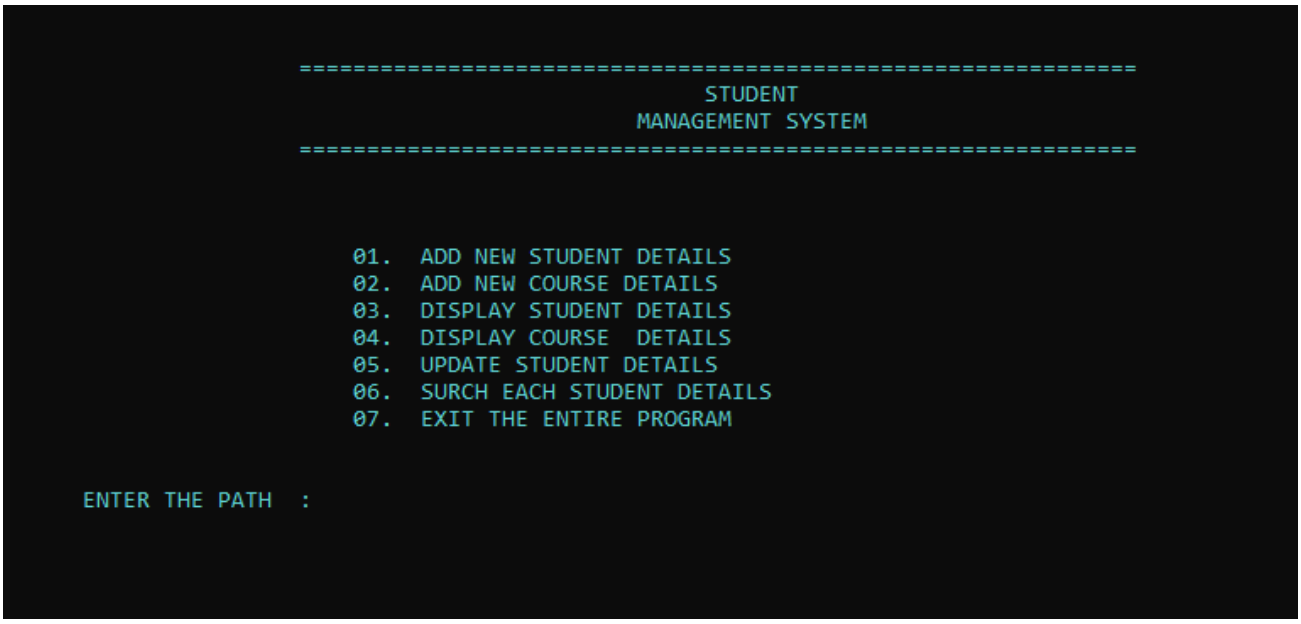


Figure 1: Main Manu

Add Student Details

This function opens the “student.txt” binary file in append mode and writes the student details in the binary file. If you wish to enter another student details input 1 to continue or exit the add student details function press 0.

1. Enter the name – input the name of the student
2. Enter the index number – input the student index number in to the system
3. Enter the department – Enter the department to each student study
4. Enter the course count –Input the course count to the system each student following or followed
5. Enter the course name- Enter the course name
6. Enter the grade- Input the each course grade in the system.

```
        ------------ADD STUDENT DETAILS----------


         ENTER THE NAME           :  Jeyapriya_Jeyakrishnan
         ENTER THE INDEX NUMBER   :  2021/T/0123
         ENTER THE DEPARTMENT     :  Information_&_Communication_Technology
         ENTER THE COURSE COUNT   :  3
                         ENTER COURSE NAME  :  Programming
                         ENTER COURSE GRADE  :  A
                         ENTER COURSE NAME  :  Basic_Mathamatics
                         ENTER COURSE GRADE  :  B
                         ENTER COURSE NAME  :  EICT
                         ENTER COURSE GRADE  :  A

(press 1 to add another student details
 or press 0   : 0



ENTER THE PATH  :
```

Figure 2: Add Student Details

Add to Course details

This function opens the “course.txt” binary file in append mode and writes the course details in the binary file. If you wish to enter another course details input 1 to continue or exit the add course details function press 0.

1. Enter the Course name- input the course name
2. Enter the Course ID - input course id

```
             -------------ADD COURSE DETAILS------------

              ENTER THE COURSE NAME   :  Programming
              ENTER THE COURSE ID     :  IC_1302


press 1 to add another student
press 0 to exit  :  1



             -------------ADD COURSE DETAILS------------

              ENTER THE COURSE NAME   :  Application_Lab
              ENTER THE COURSE ID     :  IC_1301


press 1 to add another student
press 0 to exit  :
```

Figure 3: Add Course Details

Display Student details

* Display in the terminal all student records as a report from the “student.txt” binary file.

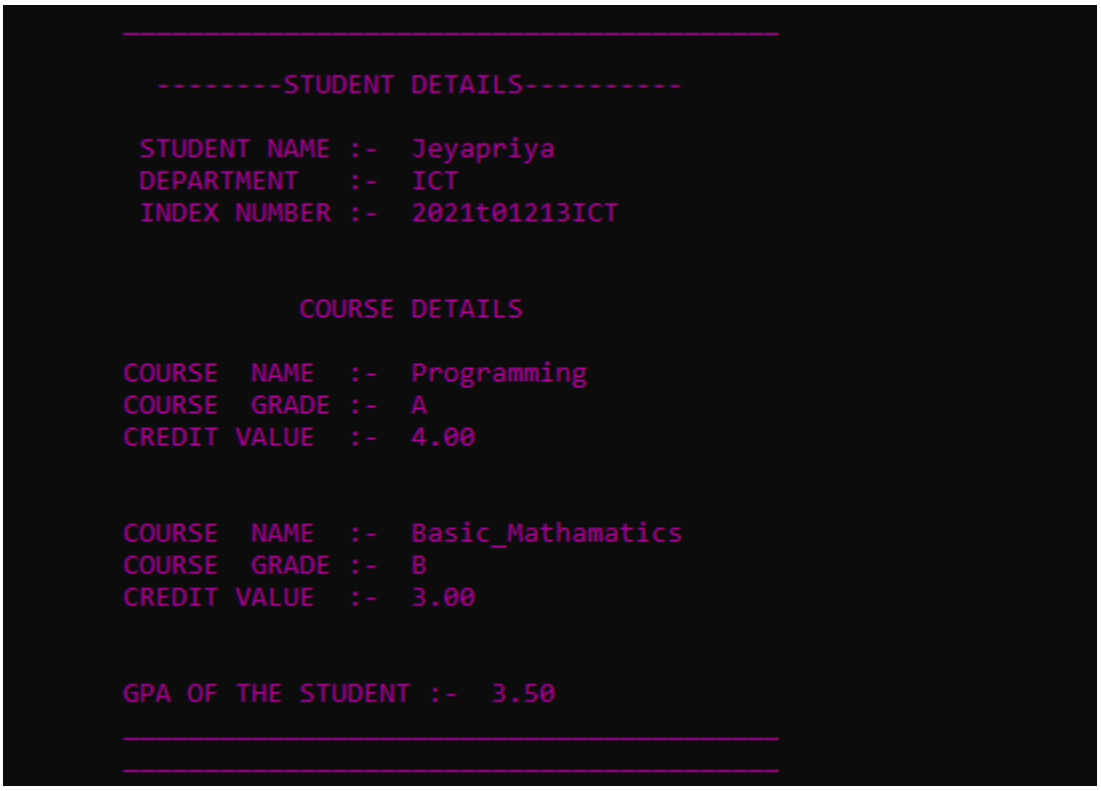


Figure 4: Display Student Details

Display course details

* Display all course details in the terminal from the course.txt” binary text file.

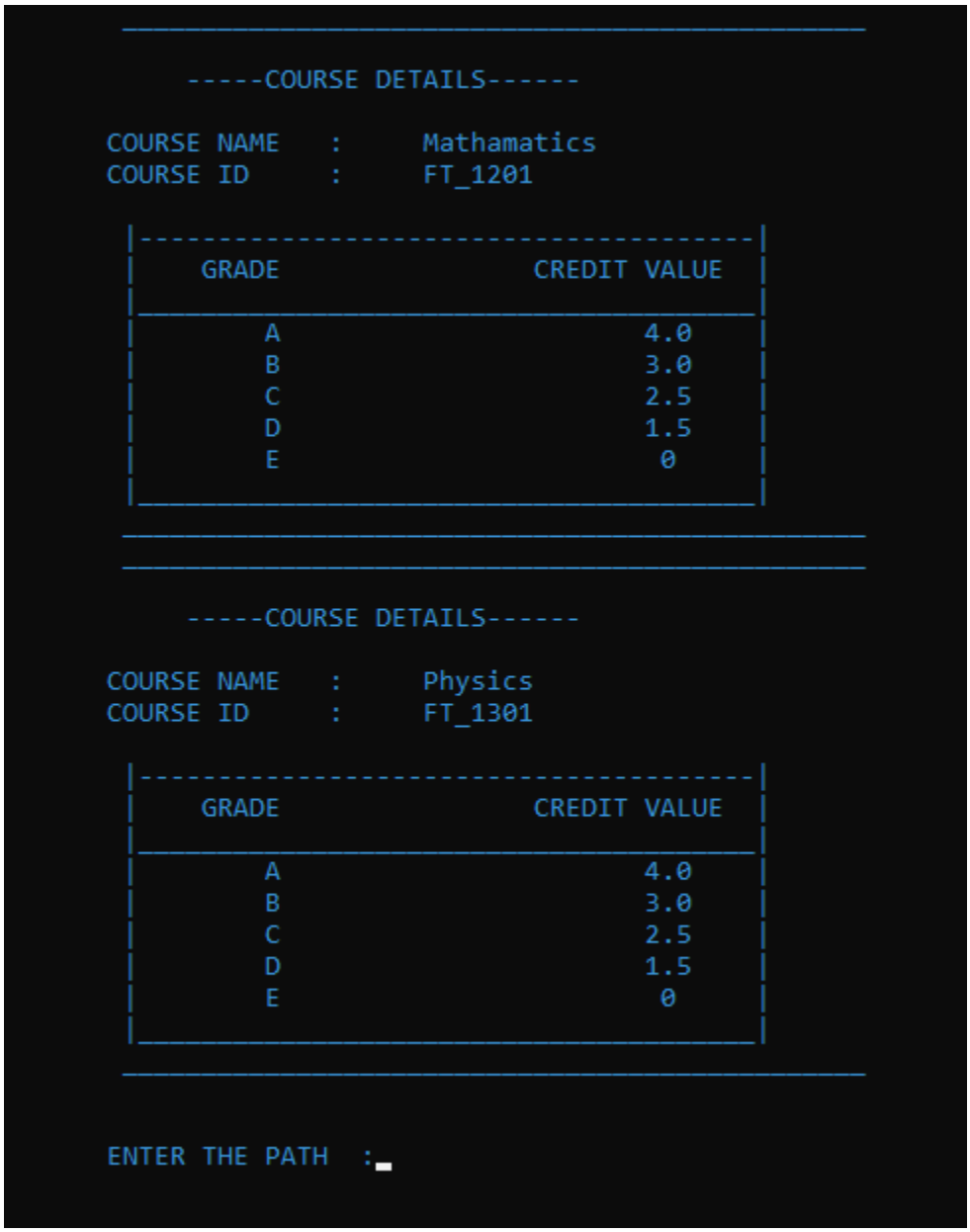


Figure 5: Display Student Details

Update Student Details

If any of the Student details have been mistyped or wrong, the user should be allowed to update the information in the text file using the terminal. The user first enters the index number of the student concurrently the system can search when the index number matches, and after the user can update the student's details.

* Users can enter the index number in the terminal

```
ENTER STUDENT INDEX NUMBER  :123
```

Figure 6: enter the index number of the student

* Users can change the name, department, index number and course details of each student.

```
ENTER STUDENT INDEX NUMBER  :123


        ------------UPDATE STUDENT DETAILS----------


         ENTER THE NAME          :
```

Figure 7: Get new student details

* The user enters the new student details then the system gives a message to user “record updated successfully.”

```
ENTER STUDENT INDEX NUMBER  :123


        ------------UPDATE STUDENT DETAILS----------


         ENTER THE NAME          : Midhun
         ENTER THE INDEX NUMBER  : 1234
         ENTER THE DEPARTMENT    : ICT
        ENTER THE COURSE COUNT   : 2
                        ENTER COURSE NAME  :  EICT
                        ENTER COURSE GRADE  :  A
                        ENTER COURSE NAME  :  ICS
                        ENTER COURSE GRADE  :  B


        Record updated succsesfully...........
```

Figure 8: Get new student details

Search each student details

This function opens the binary file in reading mode and asks the user to enter the student index number he wants to search for. If the student information is unavailable on the list, it shows the message “record not found”.

* User can enter the index number in the terminal he wants to search for.

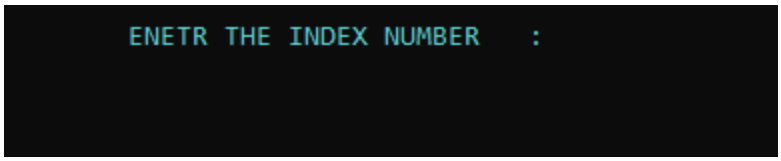


Figure 9: input index number

* If each record is not found display a message to the user “record not found.”

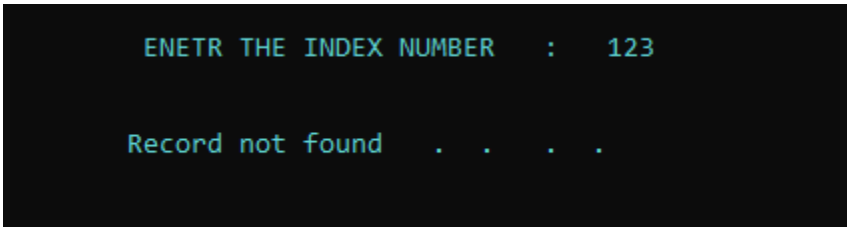


Figure 10: display record note found

* If the specific record is found then display the student details in the terminal.

```
ENETR THE INDEX NUMBER   :   147

 --------STUDENT DETAILS----------

STUDENT NAME :-  Kavin
DEPARTMENT   :-  ET
INDEX NUMBER :-  147

 ----------COURSE DETAILS-----------
COURSE  NAME  :-  ET
COURSE  GRADE :-  B
CREDIT VALUE  :-  3.00

GPA OF THE STUDENT :-  3.00
_____________________________________________


ENTER THE PATH  :
```

Figure 11: display specific student details

Exit the entire program

When the user wants to exit the entire program input the number “7”. Then user can exit the entire program.

```
                    ==============================================================
                                                STUDENT
                                            MANAGEMENT SYSTEM
                    ==============================================================


                        01.  ADD NEW STUDENT DETAILS
                        02.  ADD NEW COURSE DETAILS
                        03.  DISPLAY STUDENT DETAILS
                        04.  DISPLAY COURSE  DETAILS
                        05.  UPDATE STUDENT DETAILS
                        06.  SURCH EACH STUDENT DETAILS
                        07.  EXIT THE ENTIRE PROGRAM


        ENTER THE PATH  :7

C:\Users\DELL\Desktop>
```

Figure 12: Exit the program

Appendix A: The C program

The complete C Program with functions folded. (450-lines)

```
1    //---Header files----//
2    #include<stdio.h>
3    #include<string.h>
4    #include<conio.h>
5    #include<stdlib.h>
6    //---structures---//
7    struct course
8    {
13   struct student
14   {
23   struct c{
28
29   void add();
30   void course_details();
31   void display();
32   void display_c_details();
33   void grade();
34   void update();
35   void search();
36   int main()
37   {
93   void add()
94   {
198  void course_details()
199  {
230  void display()
231  {
269  void display_c_details()
270  {
303  void update()
304  {
420  void search()
421  {
465
```

Figure 13: entire program

Including necessary header files

1. stdio.h
2. string.h
3. conio.h
4. stdlib.h

```
//---Header files----//
#include<stdio.h>
#include<string.h>
#include<conio.h>
#include<stdlib.h>
```

Figure 14: header file

Definitions of database structure

* Generally, the database in C is handled by File. Here we declare the Structure which would store to a binary or regular file. Including structure can store course details and student details in binary files.

```
//---structures---//
struct course
{
    char c_name[20];
    char grade;
    float c_value;
};
struct student
{
    char name[20];
    char index[20];
    char dep[20];
    struct course c[10];
    int c_count;
    float total;
    float gpa;
};
struct c{
    char name[20];
    char id[10];

};
```

Figure 15: structures

Including functions

1. Void add - finding student details from the user and store them in the binary file.
2. Void course_details –finding course details from the user and store them in the binary file.
3. Void display – display all student details in the terminal as a report.
4. Void display_c_details –display all course details in the structured way in the terminal.
5. Void update – Update student details when the details wrong or mistyped.
6. Void search- display each student detail in the terminal.

```
//including functions
void add();
void course_details();
void display();
void display_c_details();
void update();
void search();
int main()
```

Figure 16: functions

Main function

```
int main()
{
    int n;
    system("cls");

    system("COLOR B");
    printf("\n\n\t\t\t================================"
                        "================================\n");
    printf(  "\t\t\t                                STUDENT\n");
    printf(  "\t\t\t                            MANAGEMENT SYSTEM\n");

    printf(    "\t\t\t================================"
                        "================================\n\n\n");

    printf("\n\t\t\t    01.  ADD NEW STUDENT DETAILS ");
    printf("\n\t\t\t    02.  ADD NEW COURSE DETAILS ");
    printf("\n\t\t\t    03.  DISPLAY STUDENT DETAILS ");
    printf("\n\t\t\t    04.  DISPLAY COURSE  DETAILS ");
    printf("\n\t\t\t    05.  UPDATE STUDENT DETAILS ");
    printf("\n\t\t\t    06.  SURCH EACH STUDENT DETAILS ");
    printf("\n\t\t\t    07.  EXIT THE ENTIRE PROGRAM ");

```

Figure 17: main function

```c
    printf("\n\t\t\t    06.  SURCH EACH STUDENT DETAILS ");
    printf("\n\t\t\t    07.  EXIT THE ENTIRE PROGRAM ");

    do{

        printf("\n\n\n\tENTER THE PATH  :");
        scanf("%d",&n);
        switch(n){
            case 1:
                add();
                goto choice;
                break;

            case 2:
                course_details();
                goto choice;
                break;
            case 3:
                display();
                goto choice;
                break;
            case 4:
                display_c_details();
                goto choice;
                break;
            case 5:
                update();
                goto choice;
                break;
            case 6:
                search();
                goto choice;
                break;
            }
        }while(n>=1&&n<7);
}
```

Figure 18: main function

Add student detail function and calculate the GPA of the student

```
void add()
{
    system("cls");
    int a;
    FILE *fp;

    fp=fopen("student.txt","a");
    if(fp == NULL)
    {
      printf("Error!");
      exit(1);
    }

    struct student s;


    do {
        s.gpa=0;
        s.total=0;
        float k[5]={4.0,3.0,2.5,1.5,0};
        system("COLOR E");
        printf("\n\n\n\t------------ADD STUDENT DETAILS----------\n\n");
        printf("\n\t ENTER THE NAME           :  ");
        scanf("%s",s.name);
        printf("\t ENTER THE INDEX NUMBER   :  ");
        scanf("%s",s.index);
        printf("\t ENTER THE DEPARTMENT     :  ");
        scanf("%s",s.dep);
        fflush(stdin);
        printf("\t ENTER THE COURSE COUNT   :  ");
        scanf("%d",&s.c_count);
        for(int  i=0;i<s.c_count;i++)
        {
```

Figure 19: add function

```
        printf("\t ENTER THE COURSE COUNT   :  ");
        scanf("%d",&s.c_count);
        for(int  i=0;i<s.c_count;i++)
        {
            printf("\t\t\t ENTER COURSE NAME  :  ");
            scanf("%s",s.c[i].c_name);
            fflush(stdin);
            printf("\t\t\t ENTER COURSE GRADE  :  ");
            scanf("%c",&s.c[i].grade);
            if(s.c[i].grade=='a')
            {
                s.c[i].c_value=k[0];
                s.total+=s.c[i].c_value;
            }

            else if(s.c[i].grade=='A')
            {
                s.c[i].c_value=k[0];
                s.total+=s.c[i].c_value;

            }
            else if(s.c[i].grade=='B')
            {
                s.c[i].c_value=k[1];
                s.total+=s.c[i].c_value;

            }
            else if(s.c[i].grade=='b')
            {
                s.c[i].c_value=k[1];
                s.total+=s.c[i].c_value;
            }
            else if(s.c[i].grade=='C')
            {
                s.c[i].c_value=k[2];
                s.total+=s.c[i].c_value;
            }
```

Figure 20: add function

```
            }
            else if(s.c[i].grade=='D')
            {
                s.c[i].c_value=k[3];
                s.total+=s.c[i].c_value;
            }
            else if(s.c[i].grade=='d')
            {
                s.c[i].c_value=k[3];
                s.total+=s.c[i].c_value;
            }
            else if(s.c[i].grade=='E')
            {
                s.c[i].c_value=k[4];
                s.total+=s.c[i].c_value;
            }
            else if(s.c[i].grade=='e')
            {
                s.c[i].c_value=k[4];
                s.total+=s.c[i].c_value;
            }

        }

        s.gpa=s.total/(float)s.c_count;
        fwrite(&s,sizeof(struct student),1,fp);
        printf("\n(press 1 to add another student details"
                 "\n or press 0   : ");
        scanf("%d",&a);


    }while(a!=0);
    fclose(fp);

}
```

Figure 21: add function

Finding course details in the binary file

```c
void course_details()
{
    system("cls");
    int a;
    FILE *fpl;
    fpl=fopen("course.txt","a");
    if(fpl == NULL)
    {
      printf("Error!");
      exit(1);
    }
    struct c cl;


    do {
        system("cls");
        system("COLOR C");
        printf("\n\n\n\t\t-------------ADD COURSE DETAILS------------\n\n");

        printf("\t\t ENTER THE COURSE NAME   :  ");
        scanf("%s",cl.name);
        printf("\t\t ENTER THE COURSE ID     :  ");
        scanf("%s",cl.id);
        fwrite(&cl,sizeof(struct c),1,fpl);
        printf("\n\n press 1 to add another student\n"
                " press 0 to exit  :  ");
        scanf("%d",&a);

    }while(a!=0);
    fclose(fpl);

}
```

Figure 22: add course details function

Display student details in the terminal

```c
void display()
{
    system("cls");
    FILE *fp;
    struct student s;
    fp=fopen("student.txt","r");
    if(fp == NULL)
    {
      printf("Error!");
      exit(1);
    }
    printf("\n\t\t\t£££££££££££££££££££££££££££££££££££££££££££££££");
    printf("\n\t\t\t\tStudent Report");
    printf("\n\t\t\t£££££££££££££££££££££££££££££££££££££££££££££££");
    while(fread(&s,sizeof(struct student),1,fp))
    {
        system("COLOR D");
        printf("\n\t ____________________________________________");
        printf("\n\n\t   --------STUDENT DETAILS----------  ");
        printf("\n\n\t  STUDENT NAME :-  %s",s.name);
        printf("\n\t  DEPARTMENT   :-  %s",s.dep);
        printf("\n\t  INDEX NUMBER :-  %s",s.index);
        printf("\n");
        printf("\n\n\t            COURSE DETAILS      ");
        for(int i=0;i< s.c_count;i++)
        {
            printf("\n\n\t COURSE  NAME  :-  %s",s.c[i].c_name);
            printf("\n\t COURSE  GRADE :-  %c",s.c[i].grade);
            printf("\n\t CREDIT VALUE  :-  %.2f\n",s.c[i].c_value);
        }
        printf("\n\n\t GPA OF THE STUDENT :-  %.2f",s.gpa);
        printf("\n\t ____________________________________________");
    }
    fclose(fp);
}
```

Figure 23: Display Student Details

Display course details in the terminal

```
void display_c_details()
{
    system("cls");
    int a;
    FILE *fpl;
    fpl=fopen("course.txt","r");
    if(fpl == NULL)
    {
      printf("Error!");
      exit(1);
    }
    struct c cl;
    while(fread(&cl,sizeof(struct c),1,fpl))
    {
        system("COLOR 3");
        printf("\n\t ____________________________________________");
        printf("\n\n\t     -----COURSE DETAILS------\n");
        printf("\n\tCOURSE NAME   :     %s",cl.name);
        printf("\n\tCOURSE ID     :     %s\n",cl.id);
        printf("\n\t |---------------------------------------|");
        printf("\n\t |    GRADE                CREDIT VALUE  |");
        printf("\n\t |_______________________________________|");
        printf("\n\t |        A                      4.0     |");
        printf("\n\t |        B                      3.0     |");
        printf("\n\t |        C                      2.5     |");
        printf("\n\t |        D                      1.5     |");
        printf("\n\t |        E                       0      |");
        printf("\n\t |_______________________________________|");
        printf("\n\t ____________________________________________");
    }
    fclose(fpl);
}
```

Figure 24: Display Course Details

Update student details

```
void update()
{
    system("cls");
    system("COLOR B");
    FILE *fp,*temp;
    char temp_idex[10];
    struct student s;
    int found=0;
    fp=fopen("student.txt","r+");
    temp=fopen("temp.txt","a+");
    if(fp == NULL)
    {
      printf("Error!");
      exit(1);
    }
    printf("\nENTER STUDENT INDEX NUMBER  :");
    scanf("%s",temp_idex);

    while(fread(&s,sizeof(struct student),1,fp))
    {
        if(strcmp(temp_idex,s.index)!=0)
        {
            fwrite(&s,sizeof(struct student),1,temp);
        }
        else if(strcmp(temp_idex,s.index)==0)
        {

            found=1;
            s.gpa=0;
            s.total=0;
            float k[5]={4.0,3.0,2.5,1.5,0};
            printf("\n\n\n\t------------UPDATE STUDENT DETAILS----------\n\n");
            printf("\n\n\t ENTER THE NAME          : ");
            scanf("%s",s.name);
            printf("\t ENTER THE INDEX NUMBER  : ");
            scanf("%s",s.index);
```

Figure 25: Update student Details

```
            printf("\t ENTER THE DEPARTMENT    : ");
            scanf("%s",s.dep);
            fflush(stdin);
            printf("\tENTER THE COURSE COUNT   : ");
            scanf("%d",&s.c_count);
            for(int i=0;i<s.c_count;i++)
            {
                printf("\t\t\t ENTER COURSE NAME  :  ");
                scanf("%s",s.c[i].c_name);
                fflush(stdin);
                printf("\t\t\t ENTER COURSE GRADE  :  ");
                scanf("%c",&s.c[i].grade);
                if(s.c[i].grade=='a')
                {
                    s.c[i].c_value=k[0];
                    s.total+=s.c[i].c_value;
                }

                else if(s.c[i].grade=='A')
                {
                    s.c[i].c_value=k[0];
                    s.total+=s.c[i].c_value;

                }
                else if(s.c[i].grade=='B')
                {
                    s.c[i].c_value=k[1];
                    s.total+=s.c[i].c_value;
                }
                else if(s.c[i].grade=='b')
                {
                    s.c[i].c_value=k[1];
                    s.total+=s.c[i].c_value;
                }
                else if(s.c[i].grade=='C')
                {
```

Figure 26: Update student Details

```
                else if(s.c[i].grade=='D')
                {
                    s.c[i].c_value=k[3];
                    s.total+=s.c[i].c_value;
                }
                else if(s.c[i].grade=='d')
                {
                    s.c[i].c_value=k[3];
                    s.total+=s.c[i].c_value;
                }
                else if(s.c[i].grade=='E')
                {
                    s.c[i].c_value=k[4];
                    s.total+=s.c[i].c_value;
                }
                else if(s.c[i].grade=='e')
                {
                    s.c[i].c_value=k[4];
                    s.total+=s.c[i].c_value;
                }
                else{
                    s.c[i].c_value=k[4];
                    s.total+=s.c[i].c_value;
                }

            }
        s.gpa=s.total/(float)s.c_count;
        fwrite(&s,sizeof(struct student),1,temp);

        }
    }
    fclose(fp);
    fclose(temp);
    remove("student.txt");
    rename("temp.txt","student.txt");
    printf("\n\n\tRecord updated succsesfully...........");
}
```

Figure 27: Update student Details

Search student details

```c
void search()
{
    system("cls");
    FILE *fp;
    struct student s;
    char temp[20];
    int templ=0;
    fp=fopen("student.txt","r");
    if(fp == NULL)
    {
      printf("Error!");
      exit(1);
    }
    printf("\n\t ENETR THE INDEX NUMBER   :   ");
    scanf("%s",temp);
    while(fread(&s,sizeof(struct student),1,fp))
    {
        if(strcmp(temp,s.index)==0)
        {
            templ=1;
            system("COLOR A");
            printf("\n\t  --------STUDENT DETAILS----------");
            printf("\n\n\t STUDENT NAME :-  %s",s.name);
            printf("\n\t DEPARTMENT   :-  %s",s.dep);
            printf("\n\t INDEX NUMBER :-  %s",s.index);
            printf("\n\n\t  ----------COURSE DETAILS-----------");
            for(int i=0;i<s.c_count;i++)
            {
                printf("\n\t COURSE  NAME  :-  %s",s.c[i].c_name);
                printf("\n\t COURSE  GRADE :-  %c",s.c[i].grade);
                printf("\n\t CREDIT VALUE  :-  %.2f\n",s.c[i].c_value);
            }
            printf("\n\t GPA OF THE STUDENT :-  %.2f",s.gpa);
            printf("\n\t_____________________________________________");
        }
    }
```

Figure 28: Search student Detail

```c
        }
    }
    if(templ==0)
        printf("\n\n\tRecord not found   .  .   .  .");
    fclose(fp);
}
```

Figure 29: Search student Details

Reference

1. w3school –

https://www.w3schools.com/c/

2. YouTube videos-

https://www.youtube.com/results?search_query=how+to+use+colourse+in+c+programming

https://www.youtube.com/watch?v=8nIilb2kiSU