Students Management System project

Learn In Depth

Eng . kerolous shenoda

Name	Roaa Aiman Fahmy

Table of contents

Problem Statement

A simple software for student information management system which can perform the following operations:

1. Store first name of the student.
2. Store last name of the student.
3. Store unique roll number for every student.
4. Store GPA for every student.
5. Store courses registered by the students

Approach

The idea is to form individual functions for every operation. All the functions are unified to form software.

1. Add student details from the file.
2. Add student details manually.
3. Find the student by the given roll number.
4. Find the student by the given first name.
5. Find the student registered in a course.
6. Count many students.
7. Delete a student by the given roll number.
8. Update a student by the given roll number.
9. Print all student's data.
10. Exit the program and save all data in file to recover back

FIFO.h file

The Student_sys.h consist of:

- define array of struct of 50 elements which is the max number of students.
- Struct student_info_t which contains the details of a student.
- Struct FIFO_info_t which contains pointers to help us in search.
- Struct FIFO_Status_t which contains error types.
- Function prototypes with their usage.

```
2  * FIFO.h
7
8  #ifndef FIFO_H_
9  #define FIFO_H_
10 #include "Platform_Types.h"
11
12 // USER CONFIGRATION
13 #define NAME_LENGTH 50
14 #define COURSES_NUMBER 5
15 #define STUDENTS_NUMBER 50
16 #define DPRINTF(...)        {fflush(stdout);fflush(stdin);\
17                              printf(__VA_ARGS__);\
18                              fflush(stdout);fflush(stdin);}
19
20
21 // Student Structures
22 typedef struct {
23     char first_name[NAME_LENGTH];
24     char last_name[NAME_LENGTH];
25     uint32_t roll_number;
26     float GPA;
27     uint32_t course_id[COURSES_NUMBER];
28 }student_info_t;
29
30 student_info_t buf[STUDENTS_NUMBER];
31
32 // structure  declaration
33 typedef struct{
34     uint32_t count ;
35     uint32_t length ;
36     student_info_t* base ;
37     student_info_t* head ;
38     student_info_t* tail ;
39
40 }FIFO_info_t ;
41
```

```c
// structure  declaration
typedef struct{
    uint32_t count ;
    uint32_t length ;
    student_info_t* base ;
    student_info_t* head ;
    student_info_t* tail ;

}FIFO_info_t ;

// enumeration decleration
typedef enum
{
    FIFO_no_error,
    FIFO_error,
    FIFO_full,
    FIFO_empty,
    FIFO_Null
}FIFO_Status_t;

// function declaration
FIFO_Status_t FIFO_init (FIFO_info_t* Fifo_buf , student_info_t* buf,uint32_t length);
FIFO_Status_t FIFO_Add_Student_From_File (FIFO_info_t* Fifo_buf);
FIFO_Status_t FIFO_Add_Student_manually (FIFO_info_t* Fifo_buf,student_info_t NewStudent );
FIFO_Status_t Search_For_Roll_Number(FIFO_info_t* Fifo_buf , uint32_t roll_number );
void Count_Student(FIFO_info_t* Fifo_buf);
FIFO_Status_t Find_student_by_roll_number (FIFO_info_t* Fifo_buf  );
FIFO_Status_t Find_student_by_First_Number (FIFO_info_t* Fifo_buf);
FIFO_Status_t Find_student_by_course_ID (FIFO_info_t* Fifo_buf);
FIFO_Status_t Delete_Student (FIFO_info_t* Fifo_buf);
FIFO_Status_t Update_student (FIFO_info_t* Fifo_buf);
void Update_courses (student_info_t *Ptemp);
FIFO_Status_t Print_Students (FIFO_info_t* Fifo_buf);
FIFO_Status_t FIFO_enqueue (FIFO_info_t* Fifo_buf , student_info_t NewStudent);
FIFO_Status_t FIFO_IS_FULL (FIFO_info_t* Fifo_buf);
FIFO_Status_t FIFO_is_empty(FIFO_info_t* Fifo_buf);
#endif /* FIFO_H_ */

```

FIFO.c file

The Student_sys.c consist of:

- Functions implementations.

1. **FIFO_init Function:**

The function will start by checking the input parameters validity and FIFO (Queue) Initialization.

```
9
0 FIFO_Status_t FIFO_init (FIFO_info_t* Fifo_buf , student_info_t* buf,uint32_t length)
1 {
2     if (buf == NULL)
3     {
4         DPRINTF("[ERROR] LIFO initialization failed: NULL pointer \n");
5         return FIFO_Null ;
6     }
7     Fifo_buf->length = length ;
8     Fifo_buf->count = 0 ;
9     Fifo_buf-> base = buf ;
0     Fifo_buf-> head = buf;
1     Fifo_buf-> tail = buf;
2     return FIFO_no_error;
3 }
```

2. **FIFO_Add_Student_From_File Function:**
    - This function starts with opening the student.txt file with the option read-only.
    - Check if the file is exited and if there is data?
    - Put the file into a while loop until reaches the last line in the file.
    - Start reading data from the student_info.txt file:
        1- Roll the number and search if it exists.
        2- First name.
        3- Last name.
        4- GPA Score.
        5- 5 Course IDs sequentially.
    - Printing info if everything is correctly.

```c
FIFO_Status_t FIFO_Add_Student_From_File (FIFO_info_t* Fifo_buf)
{
    student_info_t NewStudent ;
    FILE *file = fopen("Student_info.txt", "r");
    if (file == NULL)
    {
        printf("[ERROR] Error opening the file.\n");
        return FIFO_error;
    }
    while(!feof(file))
    {
        // Reading roll number of the student
        fscanf(file, "%d", &NewStudent.roll_number);

        // Check if the roll number is exists
        if(Search_For_Roll_Number(Fifo_buf, NewStudent.roll_number)==FIFO_error)
        {
            // Printing that we find a number with other student
            printf("\n[ERROR] Roll number %d is already taken\n", NewStudent.roll_number);

            // Ignore the rest of the line
            fscanf(file, "%*[^\n]");

            // Start over form next line in text file
            continue;
        }

        // Reading data first name, last name and GPA sequential
        fscanf(file, "%s", NewStudent.first_name);
        fscanf(file, "%s", NewStudent.last_name);
        fscanf(file, "%f", &NewStudent.GPA);

        // Reading course IDs
        for (int i = 0; i < COURSES_NUMBER; ++i)
        {
            fscanf(file, "%d", &NewStudent.course_id[i]);
        }

        // Enqueue new student
        if((FIFO_enqueue(Fifo_buf, NewStudent))== FIFO_no_error)
        {
            printf("\n[INFO] Roll number %d is saved successfully\n", NewStudent.roll_number);
        }
        else
        {
            printf("\n[ERROR] Adding students by file failed\n");
            return FIFO_error;
        }
    }

    printf("\n[INFO] Students details are saved successfully\n");

    // Closing the file
    fclose(file);

    return FIFO_no_error;
}
```

Student_info.txt

```
1 Macro Magdy 3.5 1 2 3 4 5
1 Pavly Salah 3 80 12 37 29 63
3 Bolis Karam 3.5 45 21 55 18 46
4 Kerolos Gamal 3.5 45 21 55 18 46
```

3-FIFO_Add_Student_manually Function:

- The function will start by asking for the roll number.
- Scan if the roll number exists.
- If not, continue reading data from the user.
- After getting all data add it to the buffer and print information.

```
FIFO_Status_t FIFO_Add_Student_manually (FIFO_info_t* Fifo_buf,student_info_t NewStudent)
{
    char temp_text[NAME_LENGTH];

    if (!Fifo_buf-> base||!Fifo_buf->head ||!Fifo_buf->tail)
    {
        DPRINTF("\n[ERROR] Adding students manually failed\n");
        return FIFO_Null;
    }
    //Check Overflow
    if(FIFO_IS_FULL(Fifo_buf) == FIFO_full)
    {
        DPRINTF("\n[ERROR] Adding students manually failed\nFIFO is full\n");
        return FIFO_full;
    }
    do
    {
        DPRINTF("Enter the Roll number: ");
        NewStudent.roll_number = atoi(temp_text);
    } while (Search_For_Roll_Number(Fifo_buf, NewStudent.roll_number)==FIFO_error);

    DPRINTF("Enter Student first Name: ");
    gets(NewStudent.first_name);

    DPRINTF("Enter Student Second Name: ");
    gets(NewStudent.last_name);

    DPRINTF("Enter the GPA: ");
    NewStudent.GPA = atof(temp_text);

```

```
63
64      DPRINTF("Enter the ID of each course : ");
65      for (int i= 0 ;i<COURSES_NUMBER;i++)
66      {
67          DPRINTF("course %d ID:",i+1);
68          NewStudent.course_id[i]=atoi(temp_text);
69      }
70
71      // Enqueue new item
72      *(Fifo_buf->head) = NewStudent;
73      if (Fifo_buf->head == (Fifo_buf->base +( Fifo_buf->length*sizeof(student_info_t))))// Check if the hea
74      {
75          Fifo_buf->head = Fifo_buf->base;
76      }
77      else
78      {
79          Fifo_buf->head++;
80      }
81      Fifo_buf->count ++  ;
82      DPRINTF("\n[INFO] Student details added successfully\n");
83      Count_Student(&Fifo_buf);
84      return FIFO_no_error;
85  }
86
```

4-Count_Student Function:

- Printing the Total number of students.
- Printing the remaining students who can enter.

```
87  void Count_Student(FIFO_info_t* Fifo_buf)
88  {
89
90      DPRINTF("\n[INFO] The total number of student is %d\n ",Fifo_buf->count);
91      DPRINTF("\n[INFO] you can add up to %d students\n",STUDENTS_NUMBER);
92      DPRINTF("\n[INFO] you can add  %d more students\n",(STUDENTS_NUMBER - Fifo_buf->count));
93  }
```

5-Search_For_Roll_Number Function:

- check if the roll_number is added before or not.

```
}
FIFO_Status_t Search_For_Roll_Number(FIFO_info_t* Fifo_buf , uint32_t roll_number )
{
    if (!Fifo_buf-> base||!Fifo_buf-> head)
    {
        DPRINTF("\n[ERROR]FIFO Dequeue failed: NULL is passed\n");
        return FIFO_Null;
    }

    if(FIFO_is_empty(Fifo_buf) == FIFO_empty)
    {
        DPRINTF("\n[ERROR] FIFO Dequeue failed: FIFO is empty\n");
        return FIFO_empty;
    }

    student_info_t *Ptemp = Fifo_buf->tail;
    for (int i = 0; i < Fifo_buf->count; i++)
    {
        if (Ptemp->roll_number == roll_number)
        {
            DPRINTF("[ERROR] roll number is unique for each student, please enter another one\n");
            return FIFO_error;
        }
        Ptemp++;
    }
    return FIFO_no_error;
}
```

6-Find_student_by_roll_number Function:

- searching for the student in the queue with the roll number.
- Starting with checking if there are students in the queue.
- I asked him for the roll number and showed all the details.

```
FIFO_Status_t Find_student_by_roll_number (FIFO_info_t* Fifo_buf ,uint32_t roll_number )
{
    if (!Fifo_buf-> base||!Fifo_buf-> head)
    {
        DPRINTF("\n[ERROR]FIFO Dequeue failed: NULL is passed\n");
        return FIFO_Null;
    }

    if(FIFO_is_empty(Fifo_buf) == FIFO_empty)
    {
        DPRINTF("\n[ERROR] FIFO Dequeue failed: FIFO is empty\n");
        return FIFO_empty;
    }
    student_info_t *Ptemp = Fifo_buf->tail;
    for (int i = 0; i < Fifo_buf->count; i++)
    {
        if (Ptemp->roll_number == roll_number)
        {
            DPRINTF("[ERROR] roll number is unique for each student, please enter another one\n");
            return FIFO_error;
        }
        Ptemp++;
    }
    return FIFO_no_error;

}
```

6-Find_student_by_roll_number Function:

- searching for the student in the queue with the roll number.
- Starting with checking if there are students in the queue.
- I asked him for the roll number and showed all the details.

```
FIFO_Status_t Find_student_by_roll_number (FIFO_info_t* Fifo_buf)
{
    char temp_text[NAME_LENGTH];
    int roll_number ;
    student_info_t *Ptemp = Fifo_buf->tail;

    if (!Fifo_buf-> base||!Fifo_buf-> head)
    {
        DPRINTF("\n[ERROR]FIFO Dequeue failed: NULL is passed\n");
        return FIFO_Null;
    }

    if(FIFO_is_empty(Fifo_buf) == FIFO_empty)
    {
        DPRINTF("\n[ERROR] FIFO Dequeue failed: FIFO is empty\n");
        return FIFO_empty;
    }

    DPRINTF("Enter the Roll number of the student: ");
    gets(temp_text);
    roll_number = atoi(temp_text);
    DPRINTF("The Student Details Are:\n ");
    for (int i = 0; i < Fifo_buf->count; i++)
    {
        if (Ptemp->roll_number == roll_number)
        {
            DPRINTF("Student First Name : %s\n",Ptemp->first_name);
            DPRINTF("Student Last Name : %s\n",Ptemp->last_name);
            DPRINTF("Student GPA number: %f\n",Ptemp->GPA);
```

```
    roll_number = atoi(temp_text);
    DPRINTF("The Student Details Are:\n ");
    for (int i = 0; i < Fifo_buf->count; i++)
    {
        if (Ptemp->roll_number == roll_number)
        {
            DPRINTF("Student First Name : %s\n",Ptemp->first_name);
            DPRINTF("Student Last Name : %s\n",Ptemp->last_name);
            DPRINTF("Student GPA number: %f\n",Ptemp->GPA);
            for (int i= 0 ;i<COURSES_NUMBER;i++)
            {
                DPRINTF("The Course ID are :%d \n",Ptemp->course_id[i]);

            }

            DPRINTF("-----------------------------------------------");
            return FIFO_no_error;
        }
        Ptemp++;
    }
    DPRINTF("\n[ERROR] Roll Number %d not founded\n",roll_number);
    return FIFO_error;

}
```

7-Find_student_by_First_Number Function:

- searching for the student in the queue with the first name.
- Starting with checking if there are students in the queue.
- Ask him for the first number and show all details.

```
FIFO_Status_t Find_student_by_First_Number (FIFO_info_t* Fifo_buf)
{
    char First_Name[NAME_LENGTH] ;
    student_info_t *Ptemp = Fifo_buf->tail;

    if (!Fifo_buf-> base||!Fifo_buf-> head)
    {
        DPRINTF("\n[ERROR]FIFO Dequeue failed: NULL is passed\n");
        return FIFO_Null;
    }

    if(FIFO_is_empty(Fifo_buf) == FIFO_empty)
    {
        DPRINTF("\n[ERROR] FIFO Dequeue failed: FIFO is empty\n");
        return FIFO_empty;
    }

    DPRINTF("Enter the First Name of the student: ");
    gets(First_Name);



    for (int i = 0; i < Fifo_buf->count; i++)
    {
        if (strcmp(Ptemp->first_name, First_Name) == 0)
        {
            DPRINTF("The Student Details Are:\n ");
            DPRINTF("Student Last Name : %s\n",Ptemp->last_name);
            DPRINTF("Student roll number : %d\n",Ptemp->roll_number);
            DPRINTF("Student GPA number: %f\n",Ptemp->GPA);
            for (int i= 0 ;i<COURSES_NUMBER;i++)
            {
                DPRINTF("The Course ID are :%d \n",Ptemp->course_id[i]);

            }
            DPRINTF("-----------------------------------------------");
            return FIFO_no_error;
        }
        Ptemp++;
    }
    DPRINTF("\n[ERROR] The First Name of the student %s not founded\n",First_Name);
    return FIFO_error;
}
```

8-Find_student_by_course_ID Function:

- This function it's about searching for the student in the queue with course ID.
- Starting with checking if there are students in the queue.
- I asked him for the ID and showed him all the details.
- Telling the number of students is enrolled.

```
284 FIFO_Status_t Find_student_by_course_ID (FIFO_info_t* Fifo_buf)
285 {
286     char temp_text[NAME_LENGTH];
287     int Course_ID ;
288     student_info_t *Ptemp = Fifo_buf->tail;
289 
290     if (!Fifo_buf-> base||!Fifo_buf-> head)
291     {
292         DPRINTF("\n[ERROR]FIFO Dequeue failed: NULL is passed\n");
293         return FIFO_Null;
294     }
295 
296     if(FIFO_is_empty(Fifo_buf) == FIFO_empty)
297     {
298         DPRINTF("\n[ERROR] FIFO Dequeue failed: FIFO is empty\n");
299         return FIFO_empty;
300     }
301 
302     DPRINTF("Enter the Course ID of the student: ");
303     gets(temp_text);
```

```
302     DPRINTF("Enter the Course ID of the student: ");
303     gets(temp_text);
304     Course_ID = atoi(temp_text);
305     for (int i = 0; i < Fifo_buf->count; i++)
306     {
307         for (int i= 0 ;i<COURSES_NUMBER;i++)
308         {
309             if (Ptemp->course_id[i] == Course_ID)
310             {
311                 DPRINTF("The Student Details Are:\n ");
312                 DPRINTF("Student First Name : %s\n",Ptemp->first_name);
313                 DPRINTF("Student Last Name : %s\n",Ptemp->last_name);
314                 DPRINTF("Student GPA number: %f\n",Ptemp->GPA);
315 
316                 DPRINTF("-----------------------------------------------");
317                 return FIFO_no_error;
318             }
319 
320         }
321 
322         Ptemp++;
323     }
324     DPRINTF("\n[ERROR] Course ID %d not founded\n",Course_ID);
325     return FIFO_error;
326 
327 }
```

9-Print_Students Function:

- searching for the student in the queue with course ID.
- Starting with checking if there are students in the queue
- Asking him for the ID and show all the details.

```
252 FIFO_Status_t Print_Students (FIFO_info_t* Fifo_buf)
253 {
254     if (!Fifo_buf-> base||!Fifo_buf-> head)
255     {
256         DPRINTF("\n[ERROR]FIFO Dequeue failed: NULL is passed\n");
257         return FIFO_Null;
258     }
259
260     if(FIFO_is_empty(Fifo_buf) == FIFO_empty)
261     {
262         DPRINTF("\n[ERROR] FIFO Dequeue failed: FIFO is empty\n");
263         return FIFO_empty;
264     }
265     student_info_t *Ptemp = Fifo_buf->tail;
266     for (int i = 0; i < Fifo_buf->count; i++)
267     {
268
269         DPRINTF("Student First Name : %s\n",Ptemp->first_name);
270         DPRINTF("Student Last Name : %s\n",Ptemp->last_name);
271         DPRINTF("Student roll number : %d\n",Ptemp->roll_number);
272         DPRINTF("Student GPA number: %f\n",Ptemp->GPA);
273         for (int i= 0 ;i<COURSES_NUMBER;i++)
274         {
275             DPRINTF("The Course ID are :%d \n",Ptemp->course_id[i]);
276
277         }
278         DPRINTF("-------------------------------------------------\n");
279         Ptemp++;
280     }
281     return FIFO_no_error;
282 }
```

10-Delete_Students Function:

- Stary with checking if there are students in the queue.
- Asking to enter the roll number to remove.
- Start removing progress and show information.

```c
FIFO_Status_t Delete_Student (FIFO_info_t* Fifo_buf)
{
    char temp_text[NAME_LENGTH];
    int roll_number;
    if (!Fifo_buf-> base||!Fifo_buf-> head)
    {
        printf("FIFO Dequeue failed: NULL is passed\n");
        return FIFO_Null;
    }

    if(FIFO_is_empty(Fifo_buf) == FIFO_empty)
    {
        printf("FIFO Dequeue failed: FIFO is empty\n");
        return FIFO_empty;
    }
    DPRINTF("Enter the Roll number of the student: ");
    gets(temp_text);
    roll_number = atoi(temp_text);
    if (Fifo_buf->tail->roll_number == roll_number)
    {
        //student_info_t *Ptemp = Fifo_buf->tail;
        if(Fifo_buf->tail == (Fifo_buf->base + ( Fifo_buf->length*sizeof(student_info_t))))
        {
            Fifo_buf->tail = Fifo_buf->base;
        }
        else
        {
            Fifo_buf->tail++;
        }

    }
    Fifo_buf->count--;
    return FIFO_no_error;
}
```

11-Update_student Function:

- Stary with checking there is students in the queue.
- Asking to enter the roll number to update.
- Start updating progress:

    1- Roll number and search if it exists. O
    2- First name.
    3- Last name.
    4- GPA Score.
    5- Course IDs and ask which one using switch case.

```c
FIFO_Status_t Update_student (FIFO_info_t* Fifo_buf)
{
    char temp_text[NAME_LENGTH];
    int roll_number ;
    student_info_t *Ptemp = Fifo_buf->tail;

    if (!Fifo_buf-> base||!Fifo_buf-> head)
    {
        DPRINTF("\n[ERROR]FIFO Dequeue failed: NULL is passed\n");
        return FIFO_Null;
    }

    if(FIFO_is_empty(Fifo_buf) == FIFO_empty)
    {
        DPRINTF("\n[ERROR] FIFO Dequeue failed: FIFO is empty\n");
        return FIFO_empty;
    }


     DPRINTF("Enter the Roll number to update the entry: ");
     gets(temp_text);
     roll_number = atoi(temp_text);
     for (int i = 0; i < Fifo_buf->count; i++)
     {
         if (Ptemp->roll_number == roll_number)
         {
             break;
         }
         Ptemp++;
     }
     DPRINTF("Choose what need to update:\n ");
     DPRINTF(" ============================ ");
     DPRINTF("\n\t 1: First Name");
     DPRINTF("\n\t 2: Last Name");
     DPRINTF("\n\t 3: roll no");
     DPRINTF("\n\t 4: GPA");
     DPRINTF("\n\t 5: Courses");
     DPRINTF("\n\n Enter option number: ");
     gets(temp_text);
     DPRINTF("\n ============================ \n");
     switch(atoi(temp_text))
     {
     case 1:
         DPRINTF("\n Enter The New First Name: ");
         gets(temp_text);
         strcpy(Ptemp->first_name,temp_text);
         break;
     case 2:
         DPRINTF("\n Enter The New Last Name: ");
         gets(temp_text);
         strcpy(Ptemp->last_name,temp_text);
         break;
```

```
    case 3:
        DPRINTF("\n Enter The New roll no: ");
        gets(temp_text);
        Ptemp->roll_number=atoi(temp_text);
        break;
    case 4:
        DPRINTF("\n Enter The New GPA: ");
        gets(temp_text);
        Ptemp->GPA=atof(temp_text);
        break;
    case 5:
        Update_courses(Ptemp);
        break;
    default:
        DPRINTF("\n Wrong Option: Try Again \n\n");
        break;
    }
    for (int i = 0; i < Fifo_buf->count; i++)
    {
        if (Ptemp->roll_number == roll_number)
        {
            DPRINTF("Student First Name : %s\n",Ptemp->first_name);
            DPRINTF("Student Last Name : %s\n",Ptemp->last_name);
            DPRINTF("Student GPA number: %f\n",Ptemp->GPA);
            for (int i= 0 ;i<COURSES_NUMBER;i++)
            {
                DPRINTF("The Course ID are :%d \n",Ptemp->course_id[i]);

            }
            DPRINTF("-------------------------------------------------");
            return FIFO_no_error;
        }
        Ptemp++;
    }
    }
    DPRINTF("\n[ERROR] Roll Number %d not founded\n",roll_number);
    return FIFO_error;

}

void Update_courses (student_info_t *Ptemp)
{
    char temp_text[NAME_LENGTH];
    DPRINTF("\n Enter The New Course_ID: ");
    for (int i= 0 ;i<COURSES_NUMBER;i++)
    {
        DPRINTF("course %d ID:",i+1);
        gets(temp_text);
        Ptemp->course_id[i]=atoi(temp_text);
    }

}
```

Main.c file

The main. c is just a simple file that consists of:

- *Infinite loop until the 10 choices.*
- *Calling the program functions through switch cases.*

```
#define DPRINTF(...)        {fflush(stdout);fflush(stdin);\
        printf(__VA_ARGS__);\
        fflush(stdout);fflush(stdin);}

int main(void)
{
    char temp_text[NAME_LENGTH];
    FIFO_info_t Fifo_Student;
    student_info_t NewStudent;
    FIFO_init (&Fifo_Student ,buf,STUDENTS_NUMBER);
    DPRINTF(" ============================== ");
    DPRINTF("\n Welcome to the student management \n");
    while(1)
    {
        DPRINTF(" ============================== ");
        DPRINTF("\n Choose the task that you want to perform \n");
        DPRINTF("\n\t 1: Add the student Details Manually");
        DPRINTF("\n\t 2: Add the Student Detail From Text");
        DPRINTF("\n\t 3: Find Student by Roll Number");
        DPRINTF("\n\t 4: Find Student by First Name");
        DPRINTF("\n\t 5: Find Student by Course ID");
        DPRINTF("\n\t 6: Print Students Number");
        DPRINTF("\n\t 7: Delete Student by Roll Number");
        DPRINTF("\n\t 8: Update Student by Roll Number");
        DPRINTF("\n\t 9: View Students");
        DPRINTF("\n\t 10: Exit");
        DPRINTF("\n\n Enter option number: ");
```

```
42          gets(temp_text);
43          DPRINTF("\n ============================ \n");
44          switch(atoi(temp_text))
45          {
46          case 1:
47              FIFO_Add_Student_From_File (&Fifo_Student);
48              break;
49          case 2:
50              FIFO_Add_Student_manually (&Fifo_Student,NewStudent);
51              break;
52          case 3:
53              Find_student_by_roll_number (&Fifo_Student);
54              break;
55          case 4:
56              Find_student_by_First_Number (&Fifo_Student);
57              break;
58          case 5:
59              Find_student_by_course_ID (&Fifo_Student);
60              break;
61          case 6:
62              Count_Student (&Fifo_Student);
63              break;
64          case 7:
65               Delete_Student(&Fifo_Student);
66              break;
67          case 8:
68              Update_student (&Fifo_Student);
69              break;
70          case 9:
71              Print_Students (&Fifo_Student);
72              break;
73          case 10:
74              return 0;
75
```

The output of the program

```
 ============================
 Welcome to the student management
 ============================
 Choose the task that you want to perform

         1: Add the student Details from Text file
         2: Add the student Details Manually
         3: Find Student by Roll Number
         4: Find Student by First Name
         5: Find Student by Course ID
         6: Print Students Number
         7: Delete Student by Roll Number
         8: Update Student by Roll Number
         9: View Students
         10: Exit

 Enter option number: 1

 ============================

[INFO] Roll number 1 is saved successfully
[ERROR] roll number is unique for each student, please enter another one

[ERROR] Roll number 1 is already taken

[INFO] Roll number 3 is saved successfully

[INFO] Roll number 4 is saved successfully

[INFO] The total number of student is 3

[INFO] you can add up to 50 students

[INFO] you can add  47 more students
 ============================
```

```
 Enter option number: 2

 ============================
Enter the Roll number: 5
Enter Student first Name: roaa
Enter Student Second Name: aiman
Enter the GPA: 345
Enter the ID of each course :
 course 1 ID:1
course 2 ID:2
course 3 ID:3
course 4 ID:4
course 5 ID:5

[INFO] Student details added successfully

[INFO] The total number of student is 4

[INFO] you can add up to 50 students

[INFO] you can add  46 more students
 ============================
```

```
 Enter option number: 3

 ==============================
Enter the Roll number of the student: 3
The Student Details Are:
 Student First Name : Bolis
Student Last Name : Karam
Student GPA number: 3.500000
The Course ID are :45
The Course ID are :21
The Course ID are :55
The Course ID are :18
The Course ID are :46
------------------------------------------------ ------------
```

```
 Enter option number: 4

 ==============================
Enter the First Name of the student: roaa
The Student Details Are:
 Student Last Name : aiman
Student roll number : 5
Student GPA number: 345.000000
The Course ID are :1
The Course ID are :2
The Course ID are :3
The Course ID are :4
The Course ID are :5
------------------------------------------------ ----------------------------
```

```
 Enter option number: 5

 ==============================
Enter the Course ID of the student: 21
The Student Details Are:
 Student First Name : Bolis
Student Last Name : Karam
Student GPA number: 3.500000
------------------------------------------------The Student Details Are:
 Student First Name : Kerolos
Student Last Name : Gamal
Student GPA number: 3.500000
------------------------------------------------ ------------------------
```

```
 Enter option number: 6

 ==============================

[INFO] The total number of student is 4

[INFO] you can add up to 50 students

[INFO] you can add  46 more students
 ==============================
```

```
 Enter option number: 8

 ============================
Enter the Roll number to update the entry: 1
Choose what need to update:
  ============================
        1: First Name
        2: Last Name
        3: roll no
        4: GPA
        5: Courses

 Enter option number: 1

 ============================

 Enter The New First Name: mark
Student First Name : mark
Student Last Name : Magdy
Student GPA number: 3.500000
The Course ID are :1
The Course ID are :2
The Course ID are :3
The Course ID are :4
The Course ID are :5
```

```
 Enter option number: 9

 ============================
Student First Name : mark
Student Last Name : Magdy
Student roll number : 1
Student GPA number: 3.500000
The Course ID are :1
The Course ID are :2
The Course ID are :3
The Course ID are :4
The Course ID are :5
-----------------------------------------------
Student First Name : Bolis
Student Last Name : Karam
Student roll number : 3
Student GPA number: 3.500000
The Course ID are :45
The Course ID are :21
The Course ID are :55
The Course ID are :18
The Course ID are :46
-----------------------------------------------
Student First Name : Kerolos
Student Last Name : Gamal
Student roll number : 4
Student GPA number: 3.500000
The Course ID are :45
The Course ID are :21
The Course ID are :55
The Course ID are :18
The Course ID are :46
-----------------------------------------------
Student First Name : roaa
Student Last Name : aiman
Student roll number : 5
Student GPA number: 345.000000
```

```
 Enter option number: 7

 ============================
Enter the Roll number of the student: 1

[INFO] The student is removed successfully
```

```
 ============================
Enter the Roll number of the student: 10

[ERROR] the roll number 10 is not founded
 ============================
```

```
         10. Exit

 Enter option number: 9

 ============================
Student First Name : Bolis
Student Last Name : Karam
Student roll number : 3
Student GPA number: 3.500000
The Course ID are :45
The Course ID are :21
The Course ID are :55
The Course ID are :18
The Course ID are :46
-----------------------------------------------
Student First Name : Kerolos
Student Last Name : Gamal
Student roll number : 4
Student GPA number: 3.500000
The Course ID are :45
The Course ID are :21
The Course ID are :55
The Course ID are :18
The Course ID are :46
-----------------------------------------------
Student First Name : roaa
Student Last Name : aiman
Student roll number : 5
Student GPA number: 345.000000
The Course ID are :1
The Course ID are :2
The Course ID are :3
The Course ID are :4
The Course ID are :5
-----------------------------------------------
 ============================
```