Mastering Embedded System Online Diploma
https://www.learn-in-depth.com/
https://www.facebook.com/groups/embedded.system.KS/

Student Management System Report

First term (Final Project 2)

Made by: Eng. Farha Emad Mohamed

My Profile:
https://www.learn-in-depth.com/online-diploma/farhaemad59%40gmail.com

Problem Statement:

Write a program to build simple software for a student information management system that can perform the following operations:

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

Approach:

Using a queue to be the database that contains the students. The processes performed on the system are divided into functions that do:

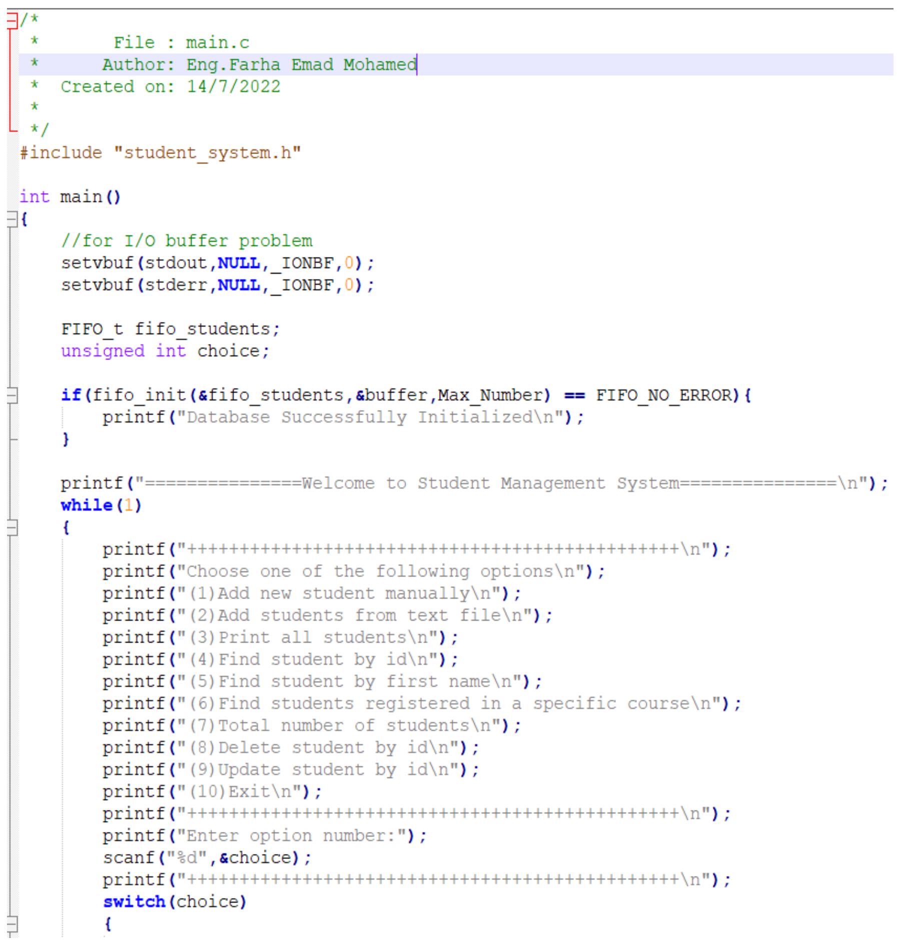
1. Add student details manually.
2. Add student details from a text file.
3. Find students by a given roll number.
4. Find students by a given first name.
5. Find students registered in a specific course.
6. Number of students registered.
7. Delete students by roll number.
8. Update student information.

Implementation:

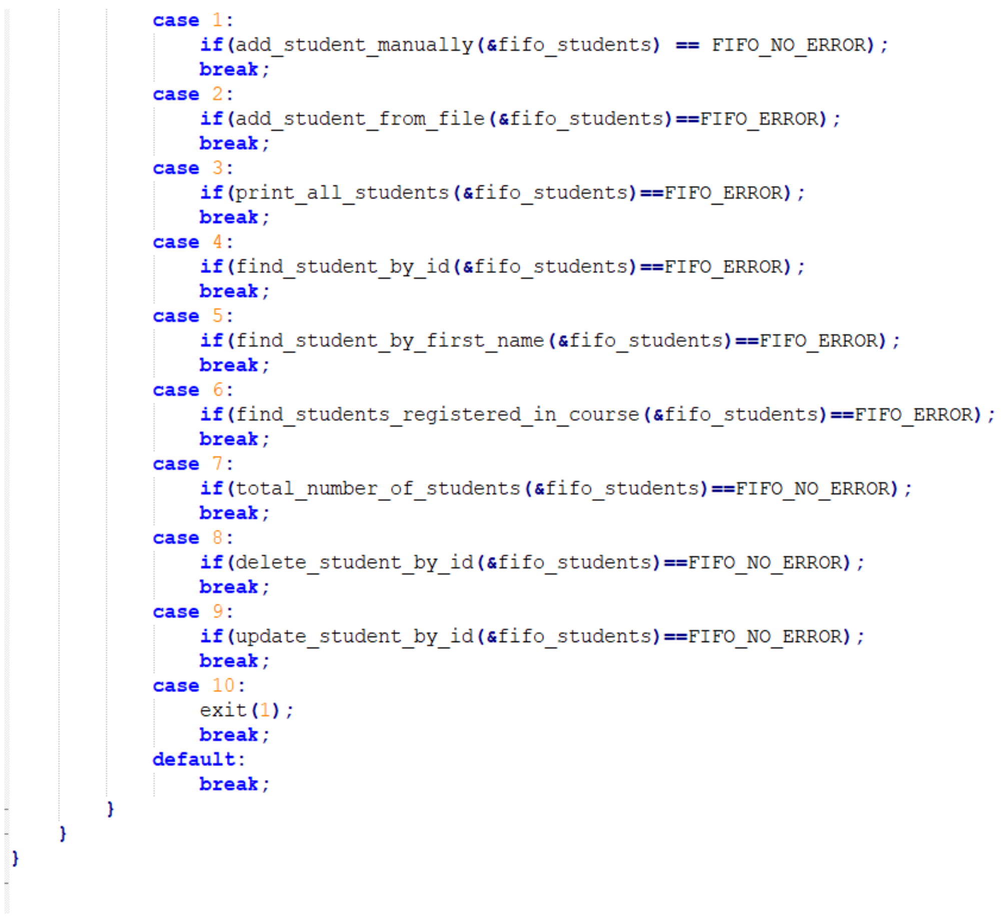
The project has 4 files:

1. Main.c which has the main program interface and calling the functions.
2. Student_system.h which has the functions' prototypes, a global struct for student data, another global struct for th queue and an array of students to be the database.
3. Student_system.c which has the functions' definitions.
4. Students.txt which has a set of some students data.

Main.c



```c
/*
 *       File : main.c
 *      Author: Eng.Farha Emad Mohamed
 *  Created on: 14/7/2022
 *
 */
#include "student_system.h"

int main()
{
    //for I/O buffer problem
    setvbuf(stdout,NULL,_IONBF,0);
    setvbuf(stderr,NULL,_IONBF,0);

    FIFO_t fifo_students;
    unsigned int choice;

    if(fifo_init(&fifo_students,&buffer,Max_Number) == FIFO_NO_ERROR){
        printf("Database Successfully Initialized\n");
    }

    printf("===============Welcome to Student Management System===============\n");
    while(1)
    {
        printf("+++++++++++++++++++++++++++++++++++++++++++++++\n");
        printf("Choose one of the following options\n");
        printf("(1)Add new student manually\n");
        printf("(2)Add students from text file\n");
        printf("(3)Print all students\n");
        printf("(4)Find student by id\n");
        printf("(5)Find student by first name\n");
        printf("(6)Find students registered in a specific course\n");
        printf("(7)Total number of students\n");
        printf("(8)Delete student by id\n");
        printf("(9)Update student by id\n");
        printf("(10)Exit\n");
        printf("+++++++++++++++++++++++++++++++++++++++++++++++\n");
        printf("Enter option number:");
        scanf("%d",&choice);
        printf("+++++++++++++++++++++++++++++++++++++++++++++++\n");
        switch(choice)
        {
```



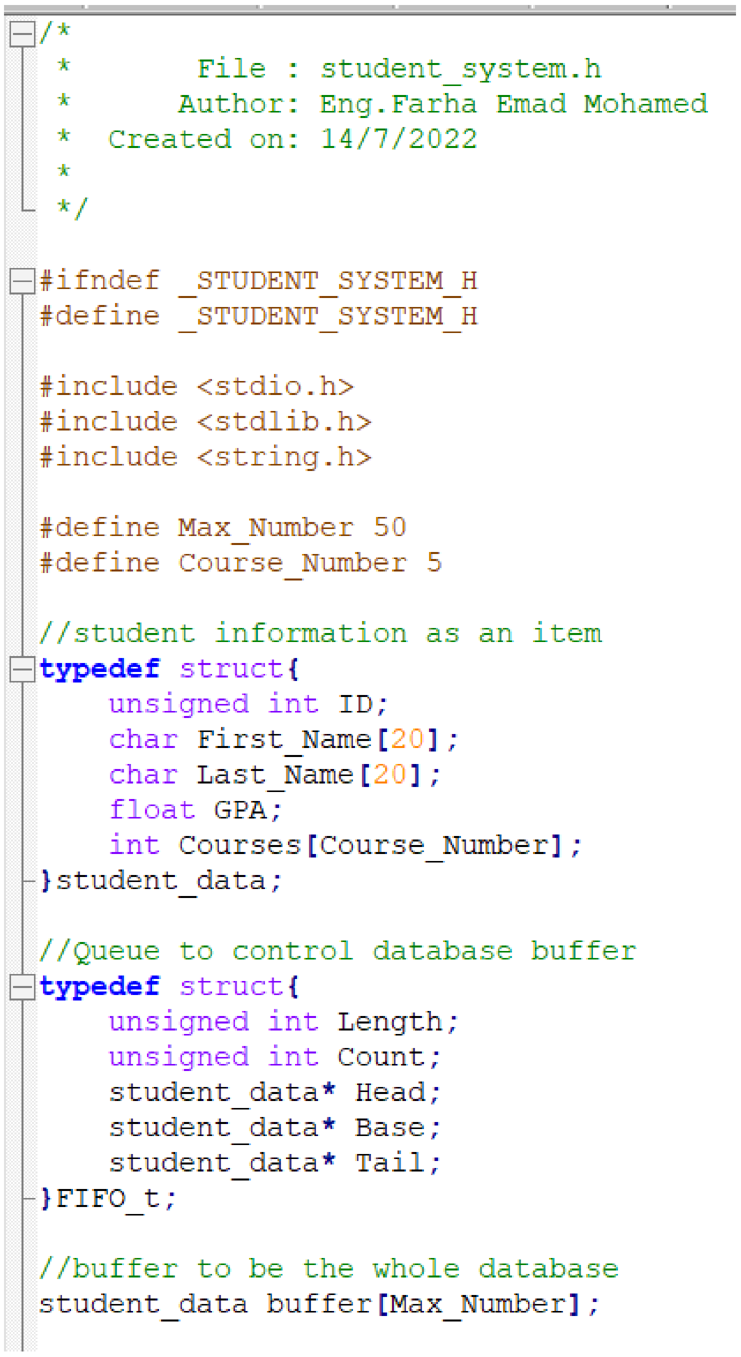
```c
            case 1:
                if(add_student_manually(&fifo_students) == FIFO_NO_ERROR);
                break;
            case 2:
                if(add_student_from_file(&fifo_students)==FIFO_ERROR);
                break;
            case 3:
                if(print_all_students(&fifo_students)==FIFO_ERROR);
                break;
            case 4:
                if(find_student_by_id(&fifo_students)==FIFO_ERROR);
                break;
            case 5:
                if(find_student_by_first_name(&fifo_students)==FIFO_ERROR);
                break;
            case 6:
                if(find_students_registered_in_course(&fifo_students)==FIFO_ERROR);
                break;
            case 7:
                if(total_number_of_students(&fifo_students)==FIFO_NO_ERROR);
                break;
            case 8:
                if(delete_student_by_id(&fifo_students)==FIFO_NO_ERROR);
                break;
            case 9:
                if(update_student_by_id(&fifo_students)==FIFO_NO_ERROR);
                break;
            case 10:
                exit(1);
                break;
            default:
                break;
        }
    }
}
```

Student_system.h

- A structure represents student object.

- A structure of to be FIFO to control buffer.

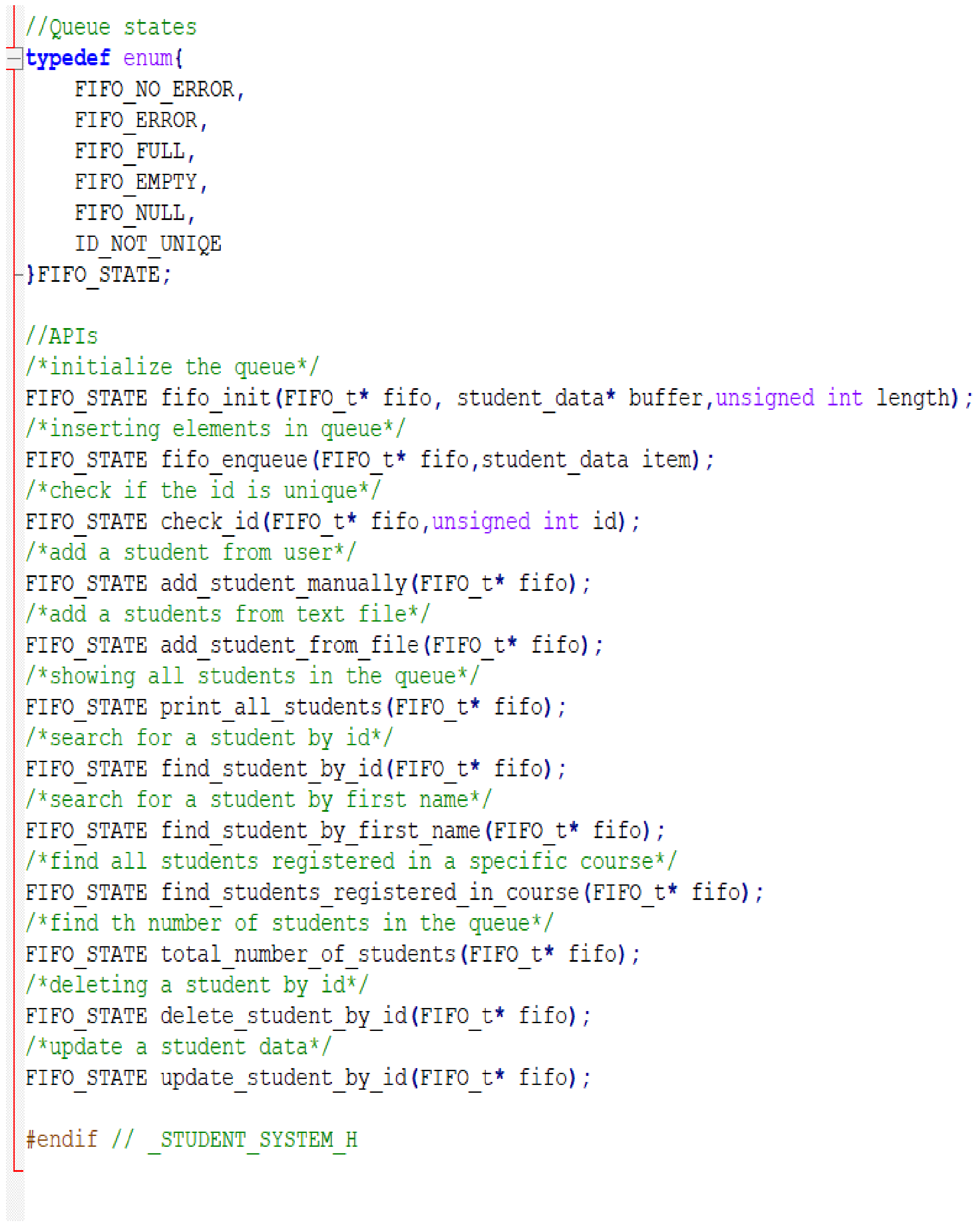
- A buffer array of students contain the whole database.



```c
/*
 *       File : student_system.h
 *      Author: Eng.Farha Emad Mohamed
 *  Created on: 14/7/2022
 *
 */

#ifndef _STUDENT_SYSTEM_H
#define _STUDENT_SYSTEM_H

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

#define Max_Number 50
#define Course_Number 5

//student information as an item
typedef struct{
    unsigned int ID;
    char First_Name[20];
    char Last_Name[20];
    float GPA;
    int Courses[Course_Number];
}student_data;

//Queue to control database buffer
typedef struct{
    unsigned int Length;
    unsigned int Count;
    student_data* Head;
    student_data* Base;
    student_data* Tail;
}FIFO_t;

//buffer to be the whole database
student_data buffer[Max_Number];
```

- An Enum to contain the FIFO states

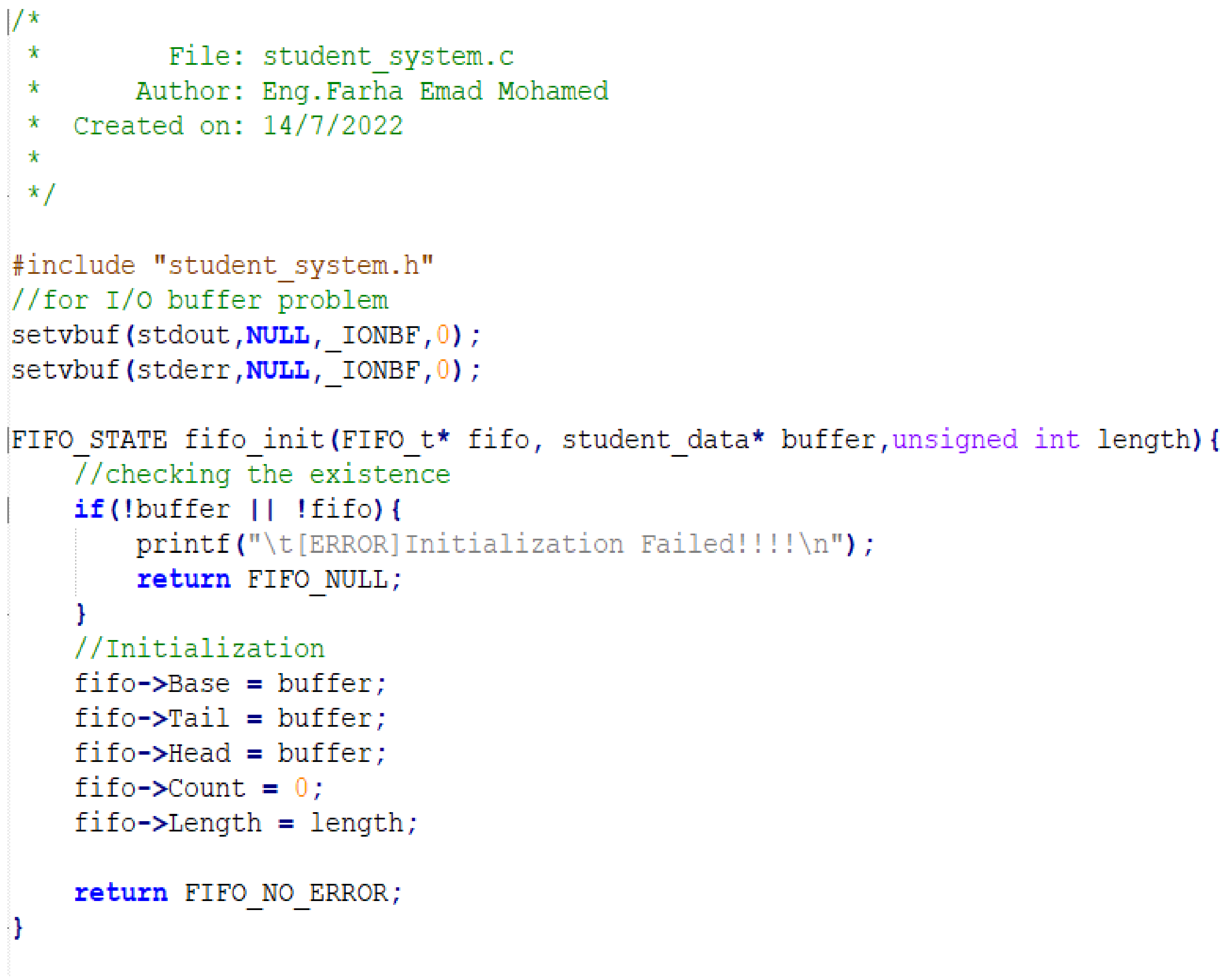
- All Funtions definition



```
//Queue states
typedef enum{
    FIFO_NO_ERROR,
    FIFO_ERROR,
    FIFO_FULL,
    FIFO_EMPTY,
    FIFO_NULL,
    ID_NOT_UNIQE
}FIFO_STATE;

//APIs
/*initialize the queue*/
FIFO_STATE fifo_init(FIFO_t* fifo, student_data* buffer,unsigned int length);
/*inserting elements in queue*/
FIFO_STATE fifo_enqueue(FIFO_t* fifo,student_data item);
/*check if the id is unique*/
FIFO_STATE check_id(FIFO_t* fifo,unsigned int id);
/*add a student from user*/
FIFO_STATE add_student_manually(FIFO_t* fifo);
/*add a students from text file*/
FIFO_STATE add_student_from_file(FIFO_t* fifo);
/*showing all students in the queue*/
FIFO_STATE print_all_students(FIFO_t* fifo);
/*search for a student by id*/
FIFO_STATE find_student_by_id(FIFO_t* fifo);
/*search for a student by first name*/
FIFO_STATE find_student_by_first_name(FIFO_t* fifo);
/*find all students registered in a specific course*/
FIFO_STATE find_students_registered_in_course(FIFO_t* fifo);
/*find th number of students in the queue*/
FIFO_STATE total_number_of_students(FIFO_t* fifo);
/*deleting a student by id*/
FIFO_STATE delete_student_by_id(FIFO_t* fifo);
/*update a student data*/
FIFO_STATE update_student_by_id(FIFO_t* fifo);

#endif // _STUDENT_SYSTEM_H
```

Student_system.c

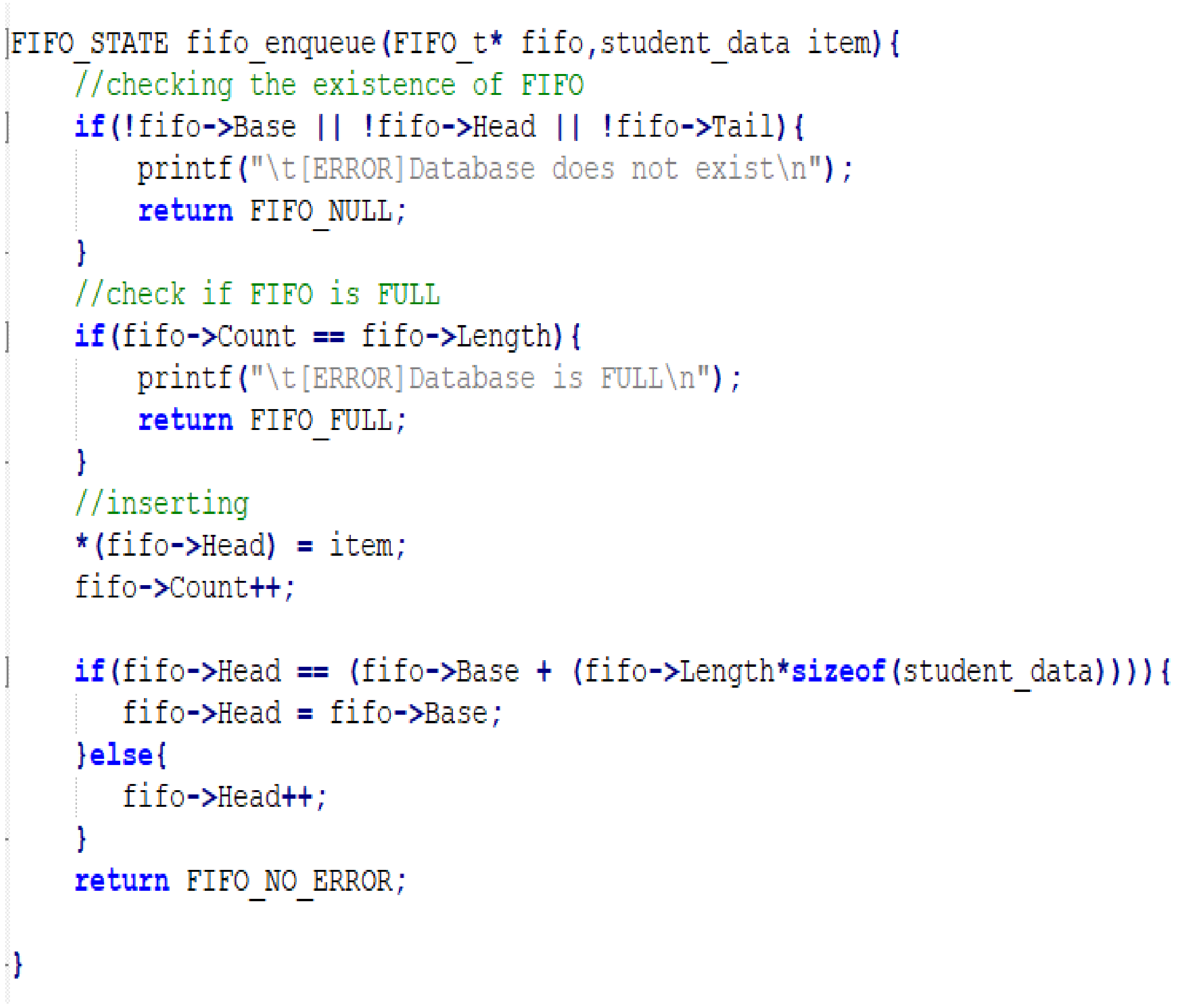
1) Fifo_init:
   Queue intialization



```c
/*
 *         File: student_system.c
 *       Author: Eng.Farha Emad Mohamed
 *   Created on: 14/7/2022
 *
 */

#include "student_system.h"
//for I/O buffer problem
setvbuf(stdout,NULL,_IONBF,0);
setvbuf(stderr,NULL,_IONBF,0);

FIFO_STATE fifo_init(FIFO_t* fifo, student_data* buffer,unsigned int length){
    //checking the existence
    if(!buffer || !fifo){
        printf("\t[ERROR]Initialization Failed!!!!\n");
        return FIFO_NULL;
    }
    //Initialization
    fifo->Base = buffer;
    fifo->Tail = buffer;
    fifo->Head = buffer;
    fifo->Count = 0;
    fifo->Length = length;

    return FIFO_NO_ERROR;
}
```

2) Fifo_enqueue:

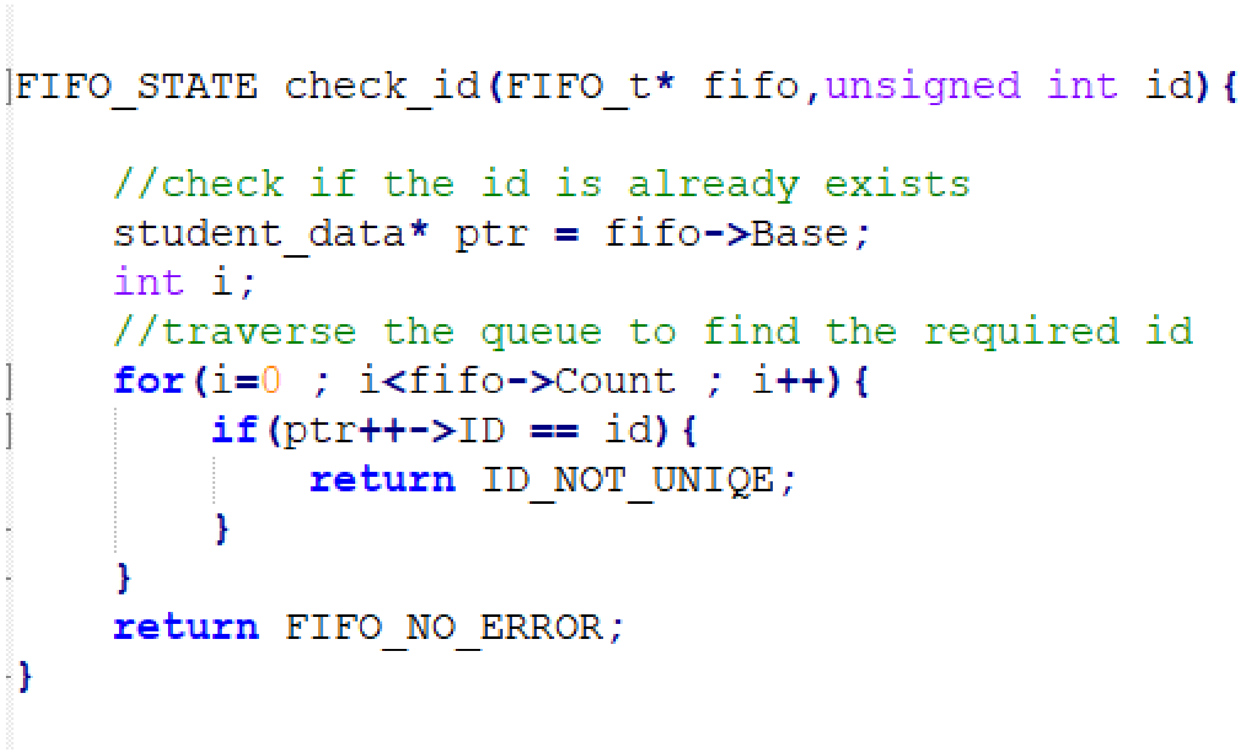
starts with checking if the buffer is existing or not then checks if the buffer is full or not. Then inserting the student item to the buffer under the control of the FIFO.



```c
FIFO_STATE fifo_enqueue(FIFO_t* fifo,student_data item){
    //checking the existence of FIFO
    if(!fifo->Base || !fifo->Head || !fifo->Tail){
        printf("\t[ERROR]Database does not exist\n");
        return FIFO_NULL;
    }
    //check if FIFO is FULL
    if(fifo->Count == fifo->Length){
        printf("\t[ERROR]Database is FULL\n");
        return FIFO_FULL;
    }
    //inserting
    *(fifo->Head) = item;
    fifo->Count++;

    if(fifo->Head == (fifo->Base + (fifo->Length*sizeof(student_data)))){
       fifo->Head = fifo->Base;
    }else{
       fifo->Head++;
    }
    return FIFO_NO_ERROR;

}
```

3) Check_id:

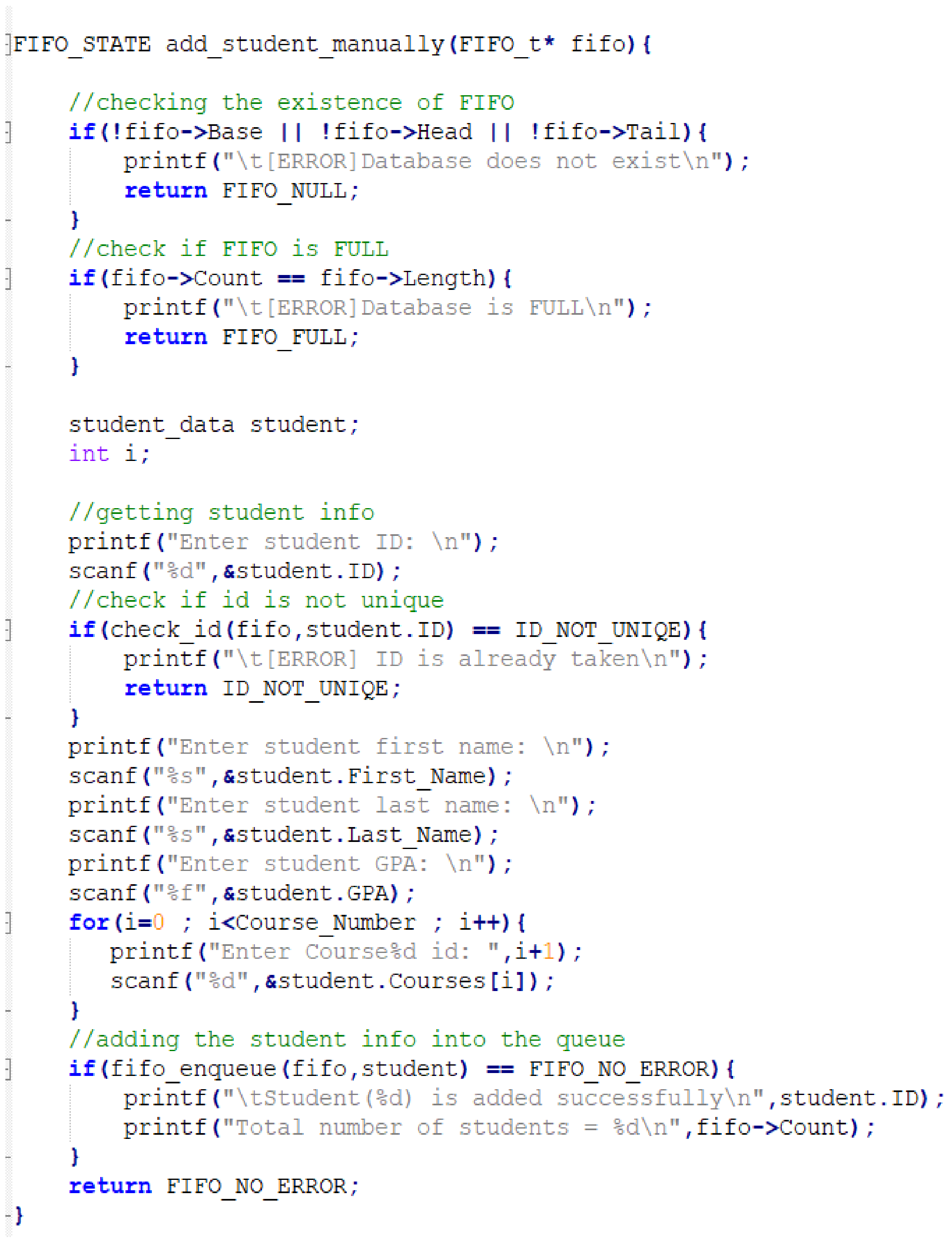
Check the uniqueness of the student roll number (ID) to ensure that no students have the same ID.



```
FIFO_STATE check_id(FIFO_t* fifo,unsigned int id){

    //check if the id is already exists
    student_data* ptr = fifo->Base;
    int i;
    //traverse the queue to find the required id
    for(i=0 ; i<fifo->Count ; i++){
        if(ptr++->ID == id){
            return ID_NOT_UNIQE;
        }
    }
    return FIFO_NO_ERROR;
}
```

4) Add_student_manually:

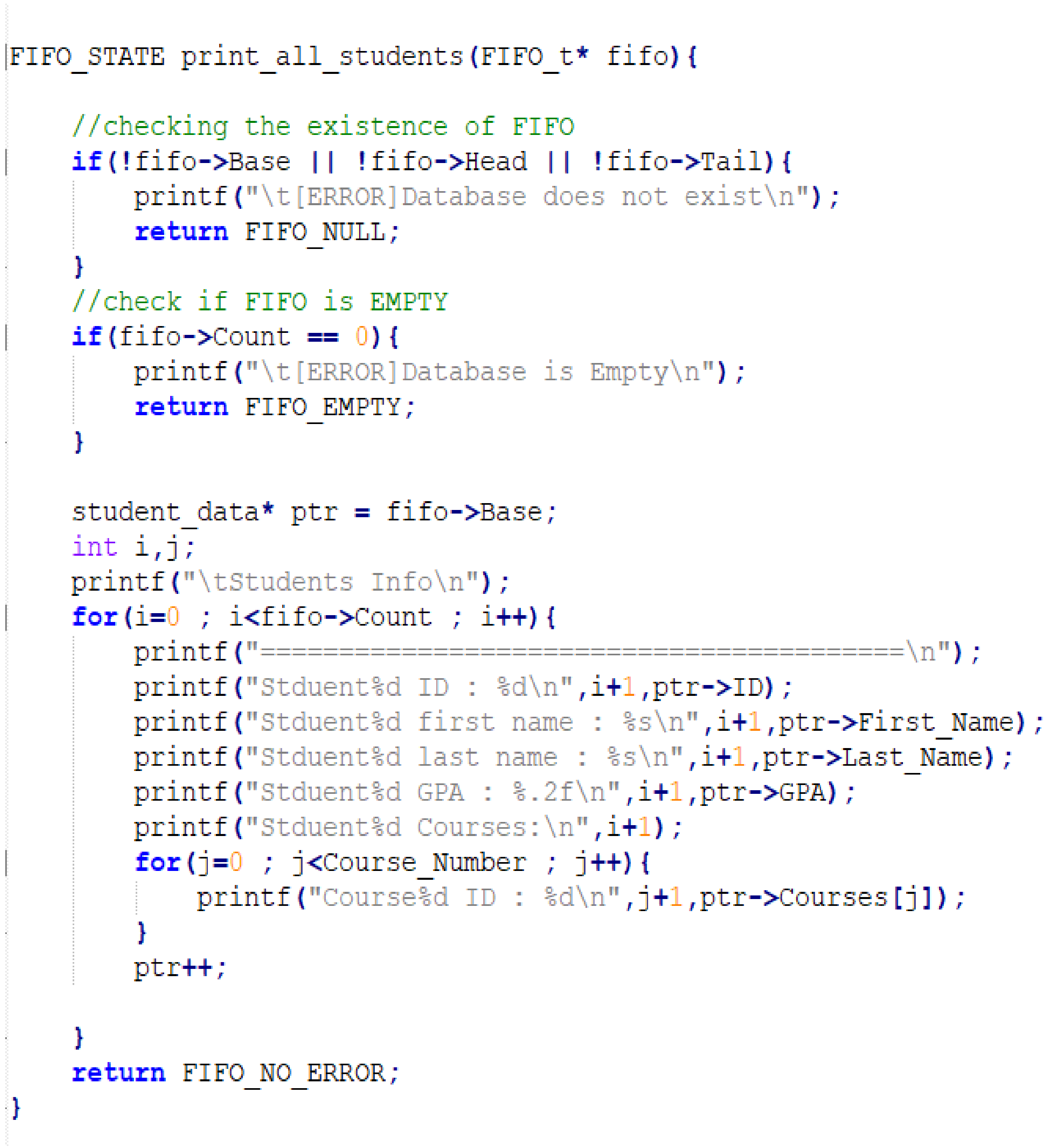
starts with checking if the buffer is existing or not then checks if the buffer is full or not. Then takes student information from the user and check if the ID has been taken before or not.



```
FIFO_STATE add_student_manually(FIFO_t* fifo){

    //checking the existence of FIFO
    if(!fifo->Base || !fifo->Head || !fifo->Tail){
        printf("\t[ERROR]Database does not exist\n");
        return FIFO_NULL;
    }
    //check if FIFO is FULL
    if(fifo->Count == fifo->Length){
        printf("\t[ERROR]Database is FULL\n");
        return FIFO_FULL;
    }

    student_data student;
    int i;

    //getting student info
    printf("Enter student ID: \n");
    scanf("%d",&student.ID);
    //check if id is not unique
    if(check_id(fifo,student.ID) == ID_NOT_UNIQE){
        printf("\t[ERROR] ID is already taken\n");
        return ID_NOT_UNIQE;
    }
    printf("Enter student first name: \n");
    scanf("%s",&student.First_Name);
    printf("Enter student last name: \n");
    scanf("%s",&student.Last_Name);
    printf("Enter student GPA: \n");
    scanf("%f",&student.GPA);
    for(i=0 ; i<Course_Number ; i++){
       printf("Enter Course%d id: ",i+1);
       scanf("%d",&student.Courses[i]);
    }
    //adding the student info into the queue
    if(fifo_enqueue(fifo,student) == FIFO_NO_ERROR){
        printf("\tStudent(%d) is added successfully\n",student.ID);
        printf("Total number of students = %d\n",fifo->Count);
    }
    return FIFO_NO_ERROR;
}
```

5) print_all_students:

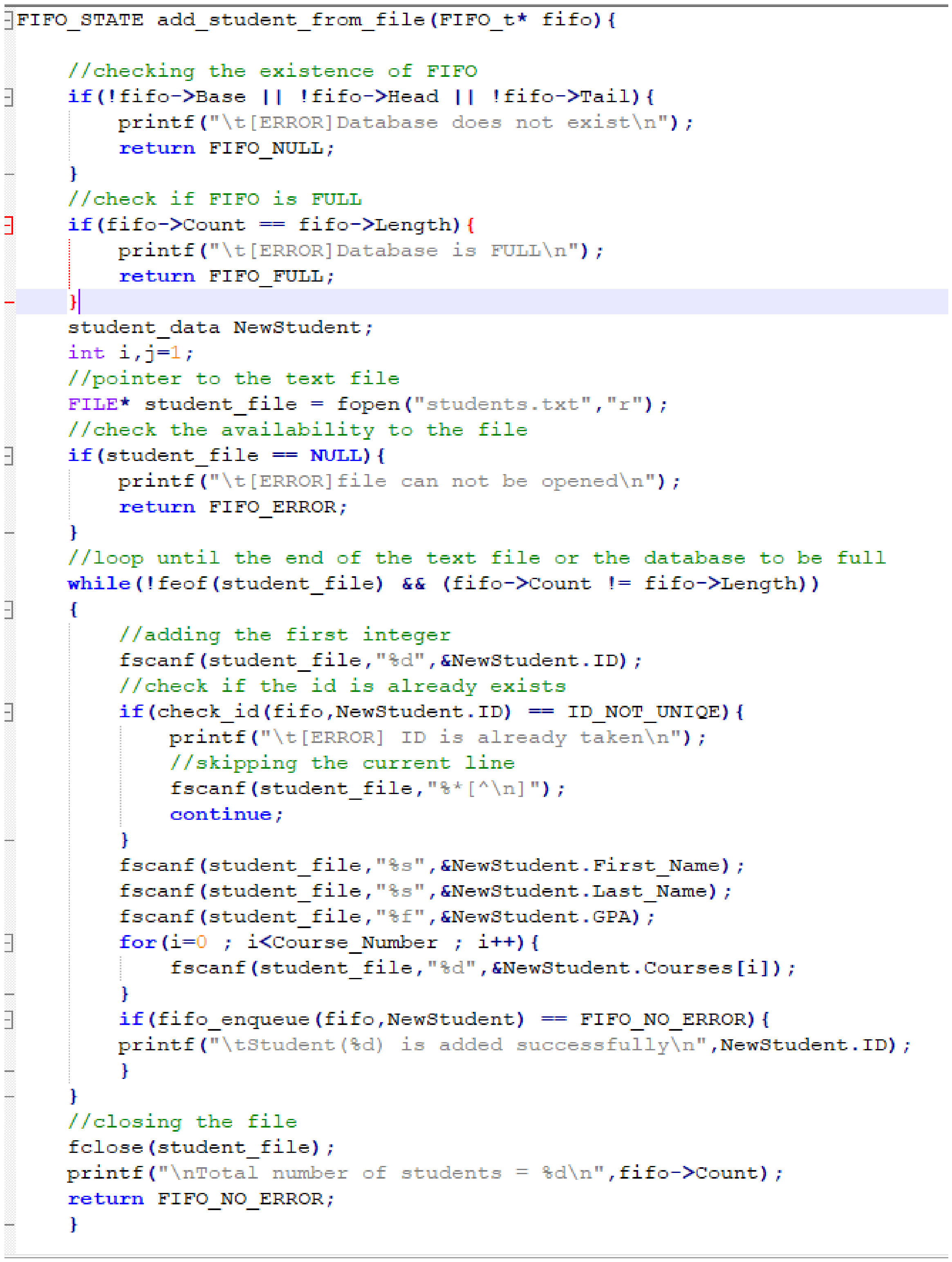
starts with checking if the buffer is existing or not then checks if the buffer is empty or not. Display all data for all students in the database. Also printsthe total number of students.



```c
FIFO_STATE print_all_students(FIFO_t* fifo){

    //checking the existence of FIFO
    if(!fifo->Base || !fifo->Head || !fifo->Tail){
        printf("\t[ERROR]Database does not exist\n");
        return FIFO_NULL;
    }
    //check if FIFO is EMPTY
    if(fifo->Count == 0){
        printf("\t[ERROR]Database is Empty\n");
        return FIFO_EMPTY;
    }

    student_data* ptr = fifo->Base;
    int i,j;
    printf("\tStudents Info\n");
    for(i=0 ; i<fifo->Count ; i++){
        printf("=========================================\n");
        printf("Stduent%d ID : %d\n",i+1,ptr->ID);
        printf("Stduent%d first name : %s\n",i+1,ptr->First_Name);
        printf("Stduent%d last name : %s\n",i+1,ptr->Last_Name);
        printf("Stduent%d GPA : %.2f\n",i+1,ptr->GPA);
        printf("Stduent%d Courses:\n",i+1);
        for(j=0 ; j<Course_Number ; j++){
            printf("Course%d ID : %d\n",j+1,ptr->Courses[j]);
        }
        ptr++;

    }
    return FIFO_NO_ERROR;
}
```

6) add_student_from_file:

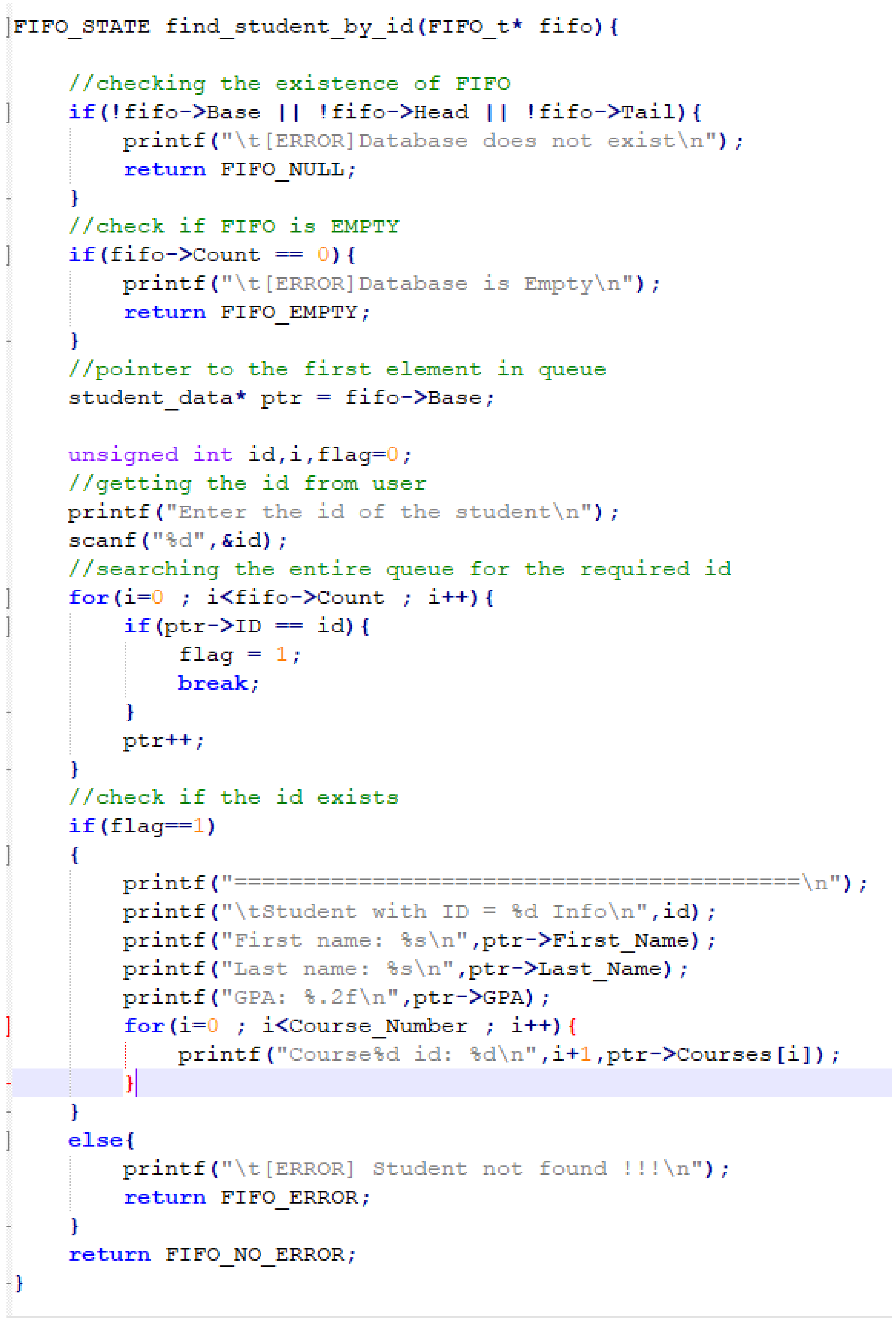
starts with checking if the buffer is existing or not then checks if the buffer is full or not. Then try to access a text file and check if the file exists. Start reading data from the students.txt file until it reaches the end of it or the buffer is full. Data is saved in the database.



```c
FIFO_STATE add_student_from_file(FIFO_t* fifo){

    //checking the existence of FIFO
    if(!fifo->Base || !fifo->Head || !fifo->Tail){
        printf("\t[ERROR]Database does not exist\n");
        return FIFO_NULL;
    }
    //check if FIFO is FULL
    if(fifo->Count == fifo->Length){
        printf("\t[ERROR]Database is FULL\n");
        return FIFO_FULL;
    }
    student_data NewStudent;
    int i,j=1;
    //pointer to the text file
    FILE* student_file = fopen("students.txt","r");
    //check the availability to the file
    if(student_file == NULL){
        printf("\t[ERROR]file can not be opened\n");
        return FIFO_ERROR;
    }
    //loop until the end of the text file or the database to be full
    while(!feof(student_file) && (fifo->Count != fifo->Length))
    {
        //adding the first integer
        fscanf(student_file,"%d",&NewStudent.ID);
        //check if the id is already exists
        if(check_id(fifo,NewStudent.ID) == ID_NOT_UNIQE){
            printf("\t[ERROR] ID is already taken\n");
            //skipping the current line
            fscanf(student_file,"%*[^\n]");
            continue;
        }
        fscanf(student_file,"%s",&NewStudent.First_Name);
        fscanf(student_file,"%s",&NewStudent.Last_Name);
        fscanf(student_file,"%f",&NewStudent.GPA);
        for(i=0 ; i<Course_Number ; i++){
            fscanf(student_file,"%d",&NewStudent.Courses[i]);
        }
        if(fifo_enqueue(fifo,NewStudent) == FIFO_NO_ERROR){
        printf("\tStudent(%d) is added successfully\n",NewStudent.ID);
        }
    }
    //closing the file
    fclose(student_file);
    printf("\nTotal number of students = %d\n",fifo->Count);
    return FIFO_NO_ERROR;
    }
```

7) find_student_by_id:

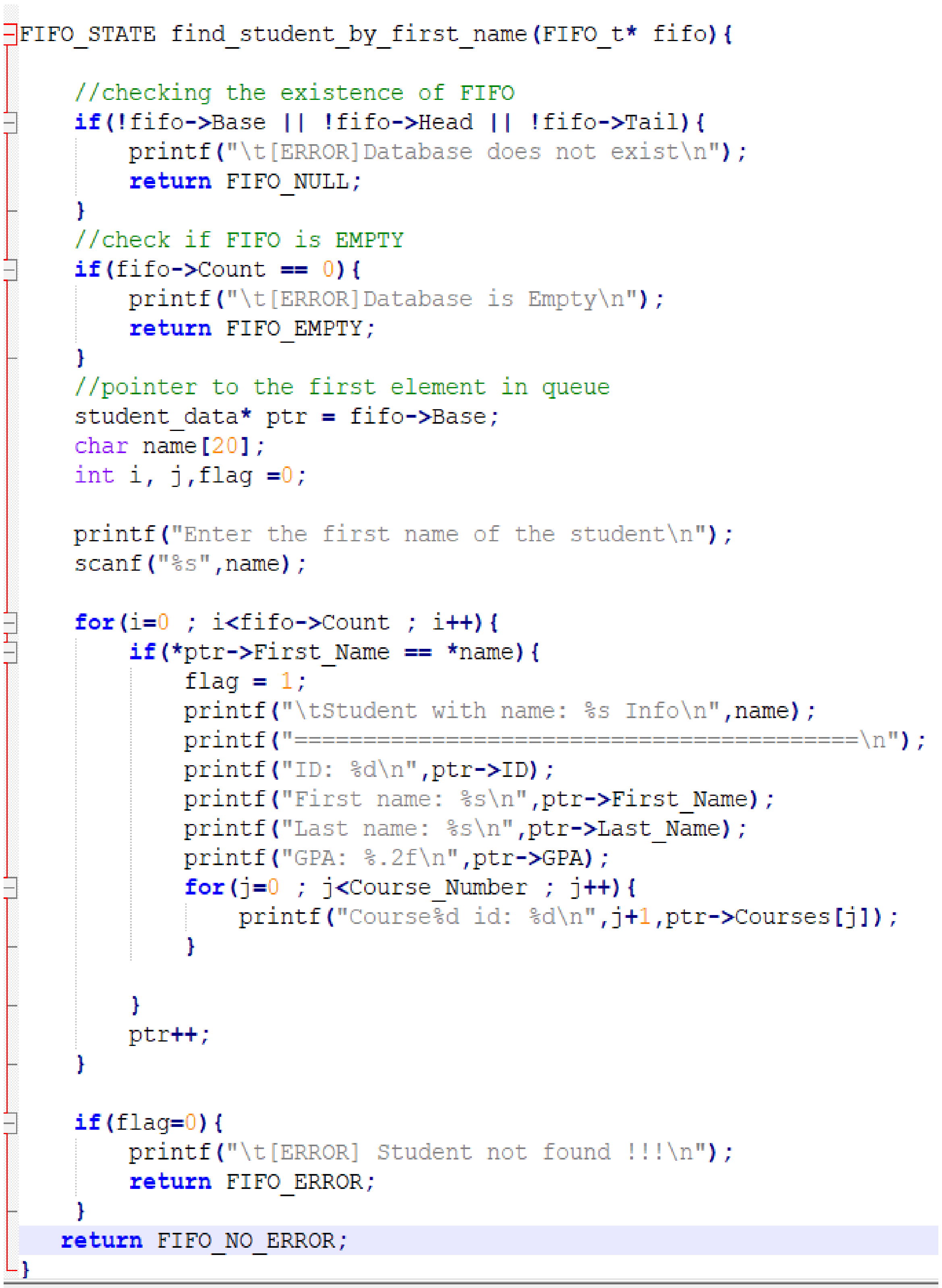
starts with checking if the buffer is existing or not then checks if the buffer is empty or not. Searching for a student by his unique roll number. It will print an error message if roll number is not found .



```c
FIFO_STATE find_student_by_id(FIFO_t* fifo){

    //checking the existence of FIFO
    if(!fifo->Base || !fifo->Head || !fifo->Tail){
        printf("\t[ERROR]Database does not exist\n");
        return FIFO_NULL;
    }
    //check if FIFO is EMPTY
    if(fifo->Count == 0){
        printf("\t[ERROR]Database is Empty\n");
        return FIFO_EMPTY;
    }
    //pointer to the first element in queue
    student_data* ptr = fifo->Base;

    unsigned int id,i,flag=0;
    //getting the id from user
    printf("Enter the id of the student\n");
    scanf("%d",&id);
    //searching the entire queue for the required id
    for(i=0 ; i<fifo->Count ; i++){
        if(ptr->ID == id){
            flag = 1;
            break;
        }
        ptr++;
    }
    //check if the id exists
    if(flag==1)
    {
        printf("==========================================\n");
        printf("\tStudent with ID = %d Info\n",id);
        printf("First name: %s\n",ptr->First_Name);
        printf("Last name: %s\n",ptr->Last_Name);
        printf("GPA: %.2f\n",ptr->GPA);
        for(i=0 ; i<Course_Number ; i++){
            printf("Course%d id: %d\n",i+1,ptr->Courses[i]);
        }
    }
    else{
        printf("\t[ERROR] Student not found !!!\n");
        return FIFO_ERROR;
    }
    return FIFO_NO_ERROR;
}
```

8) find_student_by_first_name:

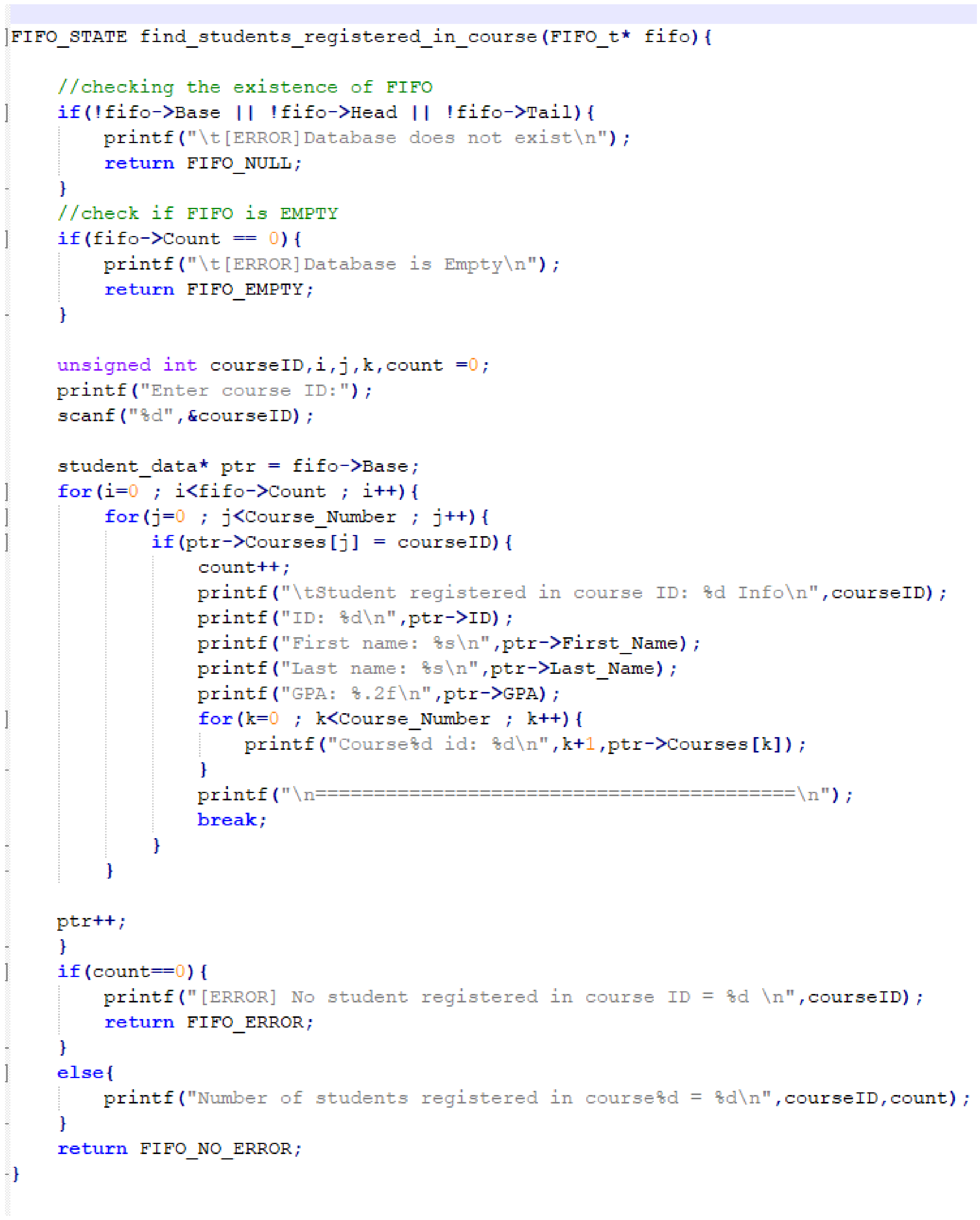
starts with checking if the buffer is existing or not then checks if the buffer is empty or not. Searching for students by their first name (which can be duplicated). If the name is not found it prints an error message.



```c
FIFO_STATE find_student_by_first_name(FIFO_t* fifo){

    //checking the existence of FIFO
    if(!fifo->Base || !fifo->Head || !fifo->Tail){
        printf("\t[ERROR]Database does not exist\n");
        return FIFO_NULL;
    }
    //check if FIFO is EMPTY
    if(fifo->Count == 0){
        printf("\t[ERROR]Database is Empty\n");
        return FIFO_EMPTY;
    }
    //pointer to the first element in queue
    student_data* ptr = fifo->Base;
    char name[20];
    int i, j,flag =0;

    printf("Enter the first name of the student\n");
    scanf("%s",name);

    for(i=0 ; i<fifo->Count ; i++){
        if(*ptr->First_Name == *name){
            flag = 1;
            printf("\tStudent with name: %s Info\n",name);
            printf("=========================================\n");
            printf("ID: %d\n",ptr->ID);
            printf("First name: %s\n",ptr->First_Name);
            printf("Last name: %s\n",ptr->Last_Name);
            printf("GPA: %.2f\n",ptr->GPA);
            for(j=0 ; j<Course_Number ; j++){
                printf("Course%d id: %d\n",j+1,ptr->Courses[j]);
            }

        }
        ptr++;
    }

    if(flag=0){
        printf("\t[ERROR] Student not found !!!\n");
        return FIFO_ERROR;
    }
   return FIFO_NO_ERROR;
}
```

9) find_students_registered_in_course:

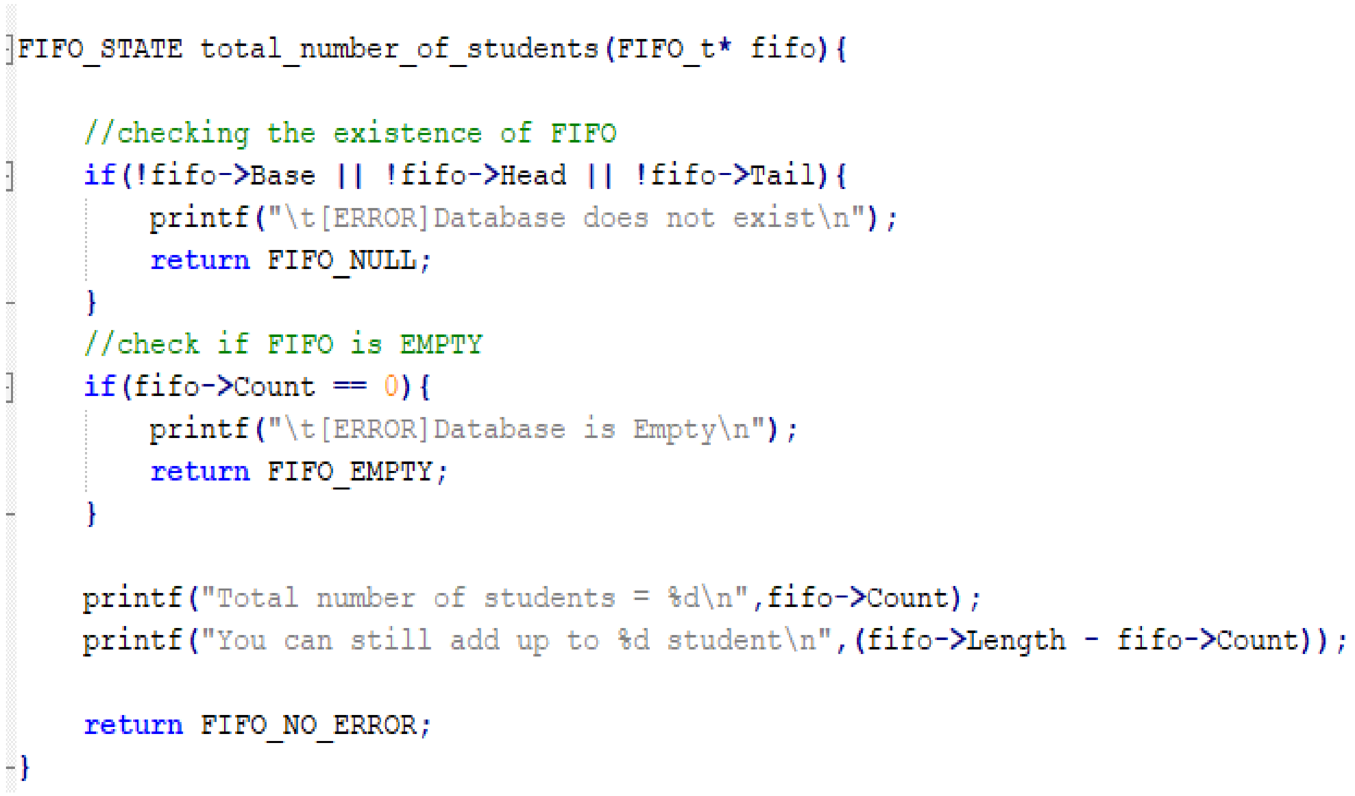
starts with checking if the buffer is existing or not then checks if the buffer is empty or not. Searches for all students that registered the same course using the course id. If there are no students registered this course an error message printed.



```
FIFO_STATE find_students_registered_in_course(FIFO_t* fifo){

    //checking the existence of FIFO
    if(!fifo->Base || !fifo->Head || !fifo->Tail){
        printf("\t[ERROR]Database does not exist\n");
        return FIFO_NULL;
    }
    //check if FIFO is EMPTY
    if(fifo->Count == 0){
        printf("\t[ERROR]Database is Empty\n");
        return FIFO_EMPTY;
    }

    unsigned int courseID,i,j,k,count =0;
    printf("Enter course ID:");
    scanf("%d",&courseID);

    student_data* ptr = fifo->Base;
    for(i=0 ; i<fifo->Count ; i++){
        for(j=0 ; j<Course_Number ; j++){
            if(ptr->Courses[j] = courseID){
                count++;
                printf("\tStudent registered in course ID: %d Info\n",courseID);
                printf("ID: %d\n",ptr->ID);
                printf("First name: %s\n",ptr->First_Name);
                printf("Last name: %s\n",ptr->Last_Name);
                printf("GPA: %.2f\n",ptr->GPA);
                for(k=0 ; k<Course_Number ; k++){
                    printf("Course%d id: %d\n",k+1,ptr->Courses[k]);
                }
                printf("\n=========================================\n");
                break;
            }
        }

    ptr++;
    }
    if(count==0){
        printf("[ERROR] No student registered in course ID = %d \n",courseID);
        return FIFO_ERROR;
    }
    else{
        printf("Number of students registered in course%d = %d\n",courseID,count);
    }
    return FIFO_NO_ERROR;
}
```

10) total_number_of_students:

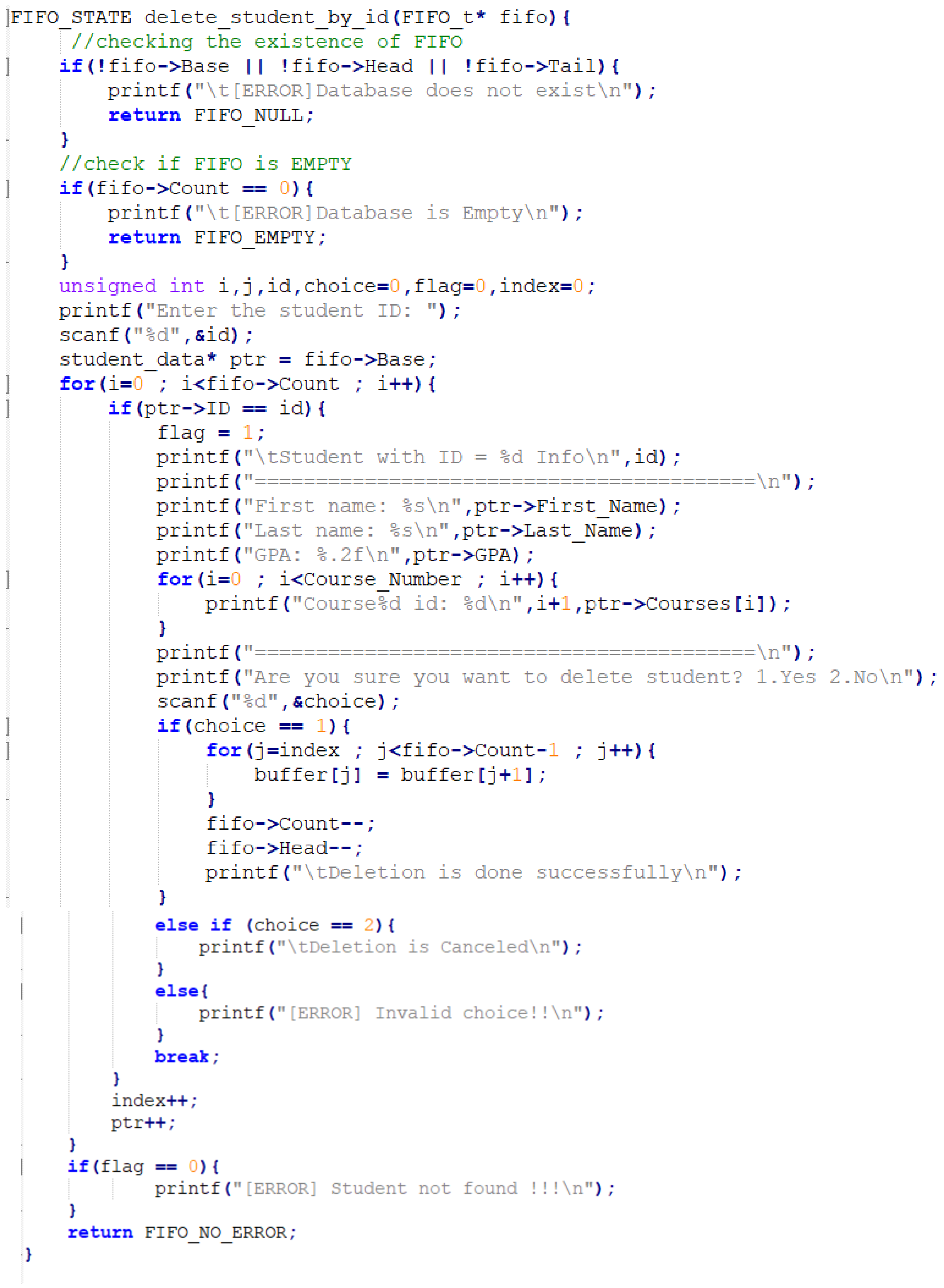
starts with checking if the buffer is existing or not then checks if the buffer is empty or not. Gets the total number of students in the database.



```c
FIFO_STATE total_number_of_students(FIFO_t* fifo){

    //checking the existence of FIFO
    if(!fifo->Base || !fifo->Head || !fifo->Tail){
        printf("\t[ERROR]Database does not exist\n");
        return FIFO_NULL;
    }
    //check if FIFO is EMPTY
    if(fifo->Count == 0){
        printf("\t[ERROR]Database is Empty\n");
        return FIFO_EMPTY;
    }

    printf("Total number of students = %d\n",fifo->Count);
    printf("You can still add up to %d student\n",(fifo->Length - fifo->Count));

    return FIFO_NO_ERROR;
}
```

11) delete_student_by_id:

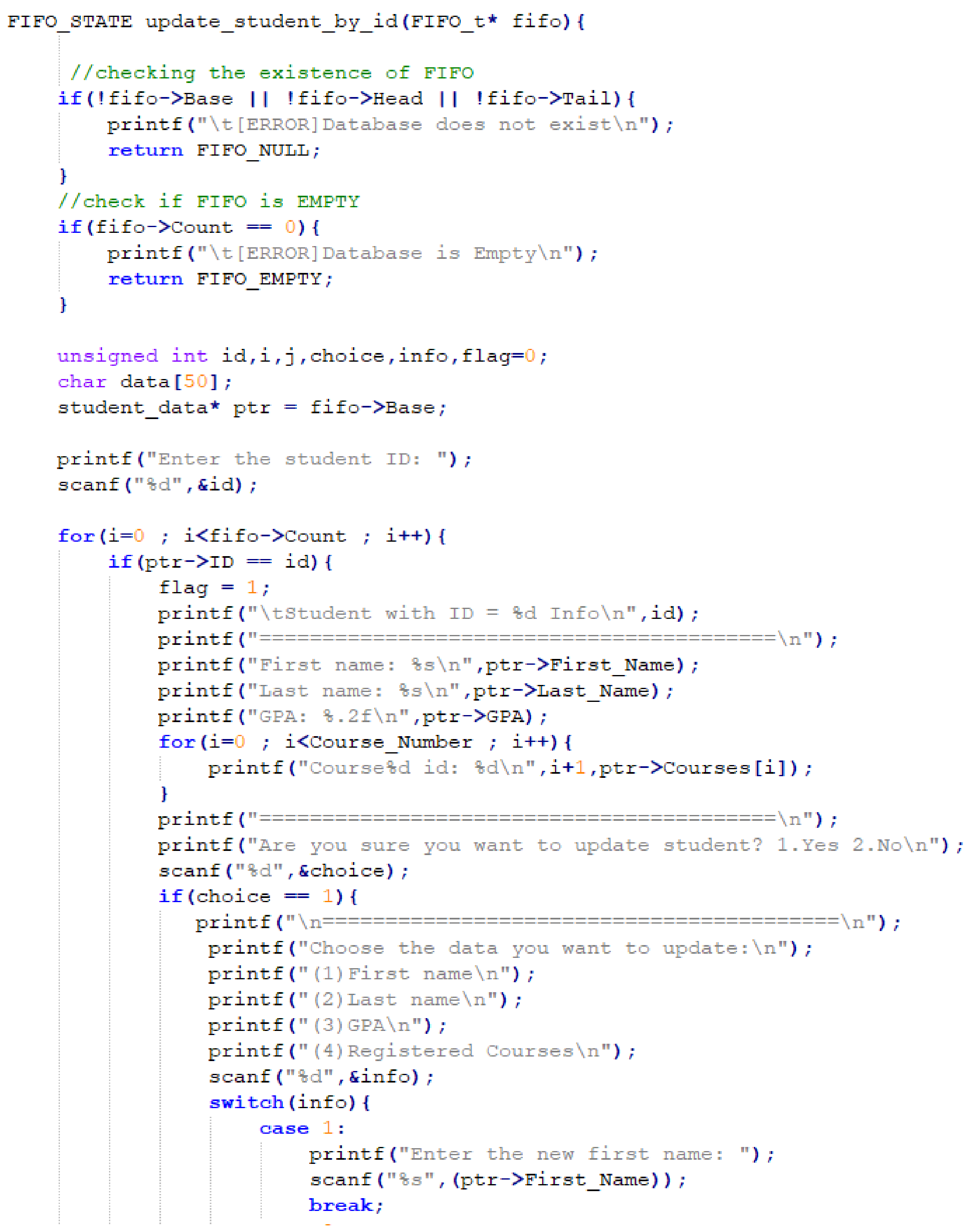
starts with checking if the buffer is existing or not then checks if the buffer is empty or not. Finds the student by id to delete him from database. Makes th user confirms the deletion first. Also shifts the buffer to occupie the empty space left after deletion. It will print an error message if roll number is not found.



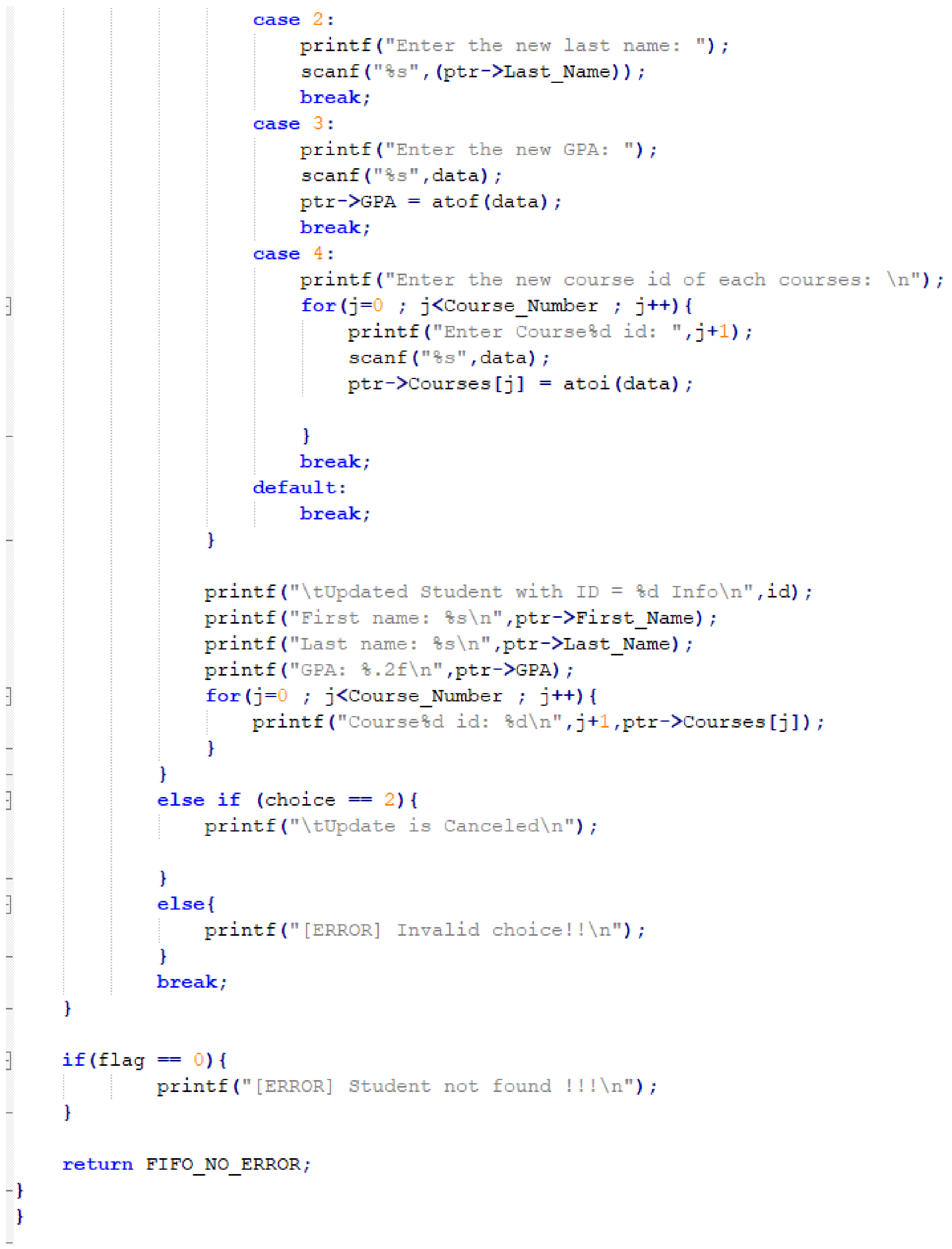
```c
FIFO_STATE delete_student_by_id(FIFO_t* fifo){
    //checking the existence of FIFO
    if(!fifo->Base || !fifo->Head || !fifo->Tail){
        printf("\t[ERROR]Database does not exist\n");
        return FIFO_NULL;
    }
    //check if FIFO is EMPTY
    if(fifo->Count == 0){
        printf("\t[ERROR]Database is Empty\n");
        return FIFO_EMPTY;
    }
    unsigned int i,j,id,choice=0,flag=0,index=0;
    printf("Enter the student ID: ");
    scanf("%d",&id);
    student_data* ptr = fifo->Base;
    for(i=0 ; i<fifo->Count ; i++){
        if(ptr->ID == id){
            flag = 1;
            printf("\tStudent with ID = %d Info\n",id);
            printf("=========================================\n");
            printf("First name: %s\n",ptr->First_Name);
            printf("Last name: %s\n",ptr->Last_Name);
            printf("GPA: %.2f\n",ptr->GPA);
            for(i=0 ; i<Course_Number ; i++){
                printf("Course%d id: %d\n",i+1,ptr->Courses[i]);
            }
            printf("=========================================\n");
            printf("Are you sure you want to delete student? 1.Yes 2.No\n");
            scanf("%d",&choice);
            if(choice == 1){
                for(j=index ; j<fifo->Count-1 ; j++){
                    buffer[j] = buffer[j+1];
                }
                fifo->Count--;
                fifo->Head--;
                printf("\tDeletion is done successfully\n");
            }
            else if (choice == 2){
                printf("\tDeletion is Canceled\n");
            }
            else{
                printf("[ERROR] Invalid choice!!\n");
            }
            break;
        }
        index++;
        ptr++;
    }
    if(flag == 0){
            printf("[ERROR] Student not found !!!\n");
    }
    return FIFO_NO_ERROR;
}
```

12) update_student_by_id:

starts with checking if the buffer is existing or not then checks if the buffer is empty or not. Updates a specific data for a specific student. It searches for the student by his ID and asks to choose the type to data he wants to update. Also makes th user confirms the update. It will print an error message if roll number is not found.



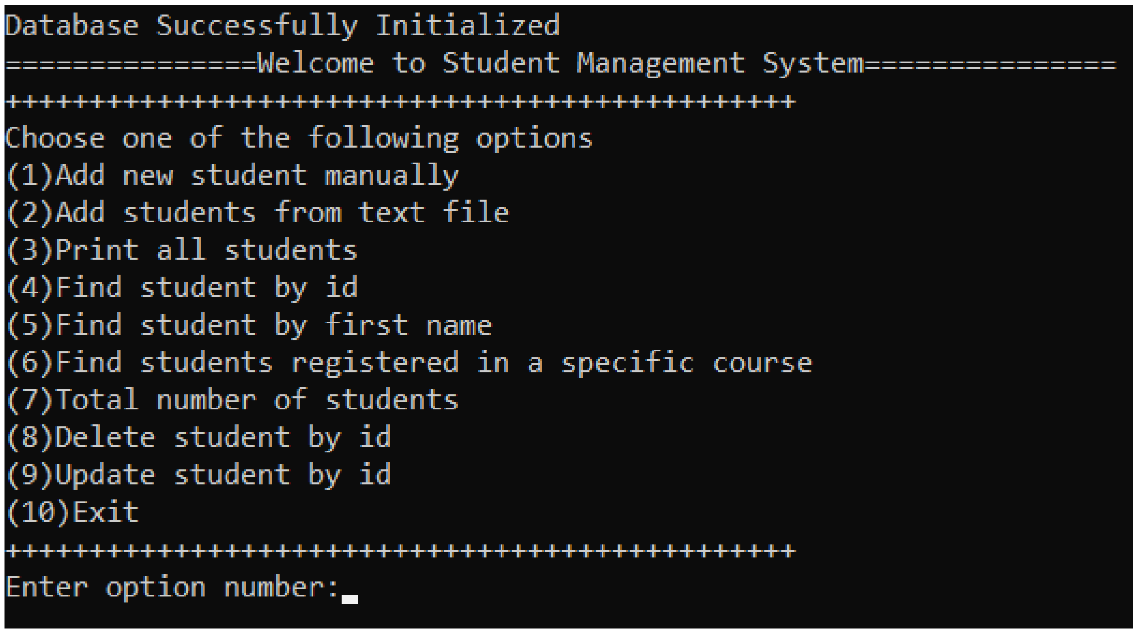
```c
FIFO_STATE update_student_by_id(FIFO_t* fifo){

    //checking the existence of FIFO
    if(!fifo->Base || !fifo->Head || !fifo->Tail){
        printf("\t[ERROR]Database does not exist\n");
        return FIFO_NULL;
    }
    //check if FIFO is EMPTY
    if(fifo->Count == 0){
        printf("\t[ERROR]Database is Empty\n");
        return FIFO_EMPTY;
    }

    unsigned int id,i,j,choice,info,flag=0;
    char data[50];
    student_data* ptr = fifo->Base;

    printf("Enter the student ID: ");
    scanf("%d",&id);

    for(i=0 ; i<fifo->Count ; i++){
        if(ptr->ID == id){
            flag = 1;
            printf("\tStudent with ID = %d Info\n",id);
            printf("=========================================\n");
            printf("First name: %s\n",ptr->First_Name);
            printf("Last name: %s\n",ptr->Last_Name);
            printf("GPA: %.2f\n",ptr->GPA);
            for(i=0 ; i<Course_Number ; i++){
                printf("Course%d id: %d\n",i+1,ptr->Courses[i]);
            }
            printf("=========================================\n");
            printf("Are you sure you want to update student? 1.Yes 2.No\n");
            scanf("%d",&choice);
            if(choice == 1){
               printf("\n=========================================\n");
                printf("Choose the data you want to update:\n");
                printf("(1)First name\n");
                printf("(2)Last name\n");
                printf("(3)GPA\n");
                printf("(4)Registered Courses\n");
                scanf("%d",&info);
                switch(info){
                    case 1:
                        printf("Enter the new first name: ");
                        scanf("%s",(ptr->First_Name));
                        break;
```



```
                    case 2:
                        printf("Enter the new last name: ");
                        scanf("%s",(ptr->Last_Name));
                        break;
                    case 3:
                        printf("Enter the new GPA: ");
                        scanf("%s",data);
                        ptr->GPA = atof(data);
                        break;
                    case 4:
                        printf("Enter the new course id of each courses: \n");
                        for(j=0 ; j<Course_Number ; j++){
                            printf("Enter Course%d id: ",j+1);
                            scanf("%s",data);
                            ptr->Courses[j] = atoi(data);

                        }
                        break;
                    default:
                        break;
                }

                printf("\tUpdated Student with ID = %d Info\n",id);
                printf("First name: %s\n",ptr->First_Name);
                printf("Last name: %s\n",ptr->Last_Name);
                printf("GPA: %.2f\n",ptr->GPA);
                for(j=0 ; j<Course_Number ; j++){
                    printf("Course%d id: %d\n",j+1,ptr->Courses[j]);
                }
            }
            else if (choice == 2){
                printf("\tUpdate is Canceled\n");

            }
            else{
                printf("[ERROR] Invalid choice!!\n");
            }
            break;
    }

    if(flag == 0){
            printf("[ERROR] Student not found !!!\n");
    }

    return FIFO_NO_ERROR;
}
}
```

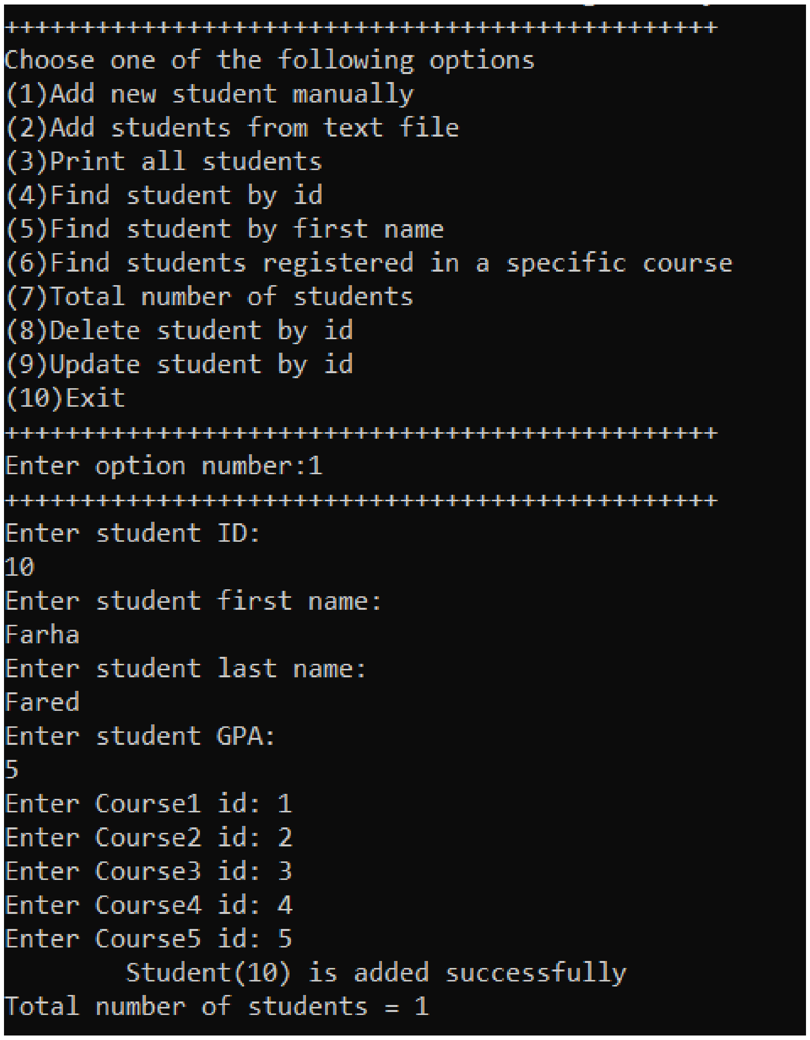
Testing the code output

1) Main page



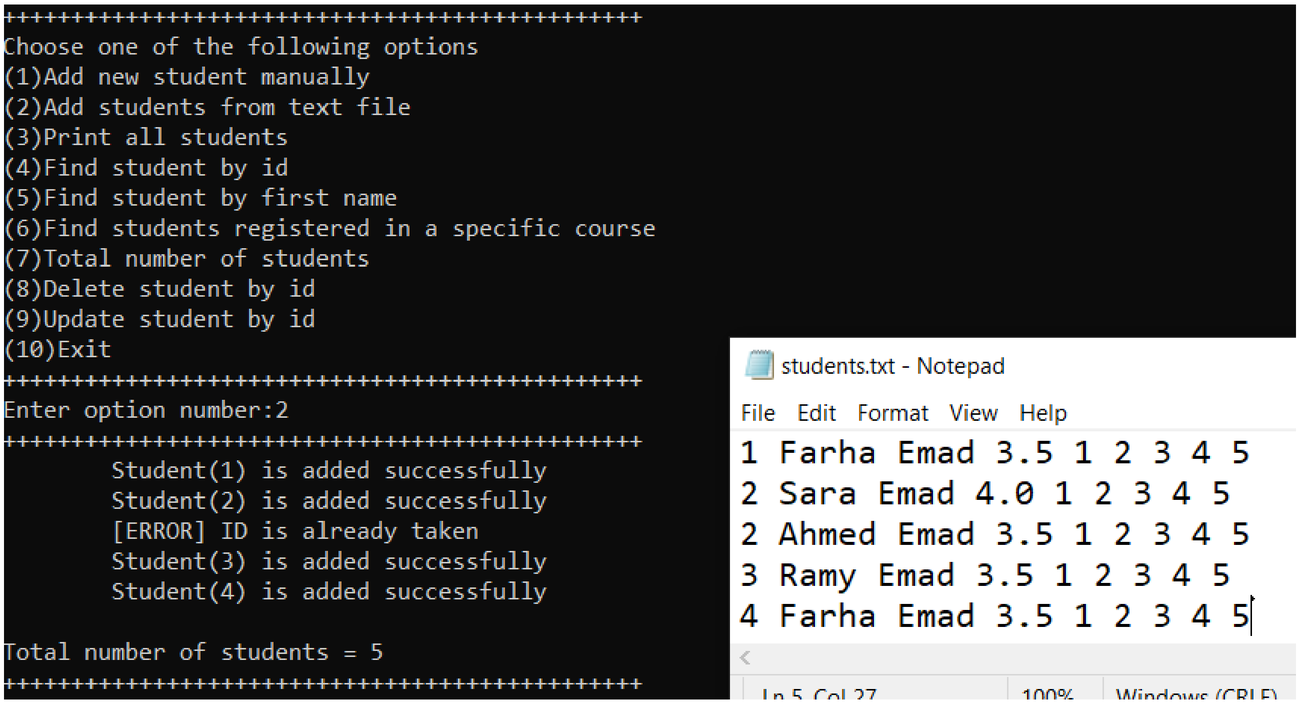
```
Database Successfully Initialized
===============Welcome to Student Management System===============
+++++++++++++++++++++++++++++++++++++++++++++++
Choose one of the following options
(1)Add new student manually
(2)Add students from text file
(3)Print all students
(4)Find student by id
(5)Find student by first name
(6)Find students registered in a specific course
(7)Total number of students
(8)Delete student by id
(9)Update student by id
(10)Exit
+++++++++++++++++++++++++++++++++++++++++++++++
Enter option number:
```

2) Adding a student manually



```
+++++++++++++++++++++++++++++++++++++++++++++++
Choose one of the following options
(1)Add new student manually
(2)Add students from text file
(3)Print all students
(4)Find student by id
(5)Find student by first name
(6)Find students registered in a specific course
(7)Total number of students
(8)Delete student by id
(9)Update student by id
(10)Exit
+++++++++++++++++++++++++++++++++++++++++++++++
Enter option number:1
+++++++++++++++++++++++++++++++++++++++++++++++
Enter student ID:
10
Enter student first name:
Farha
Enter student last name:
Fared
Enter student GPA:
5
Enter Course1 id: 1
Enter Course2 id: 2
Enter Course3 id: 3
Enter Course4 id: 4
Enter Course5 id: 5
        Student(10) is added successfully
Total number of students = 1
```

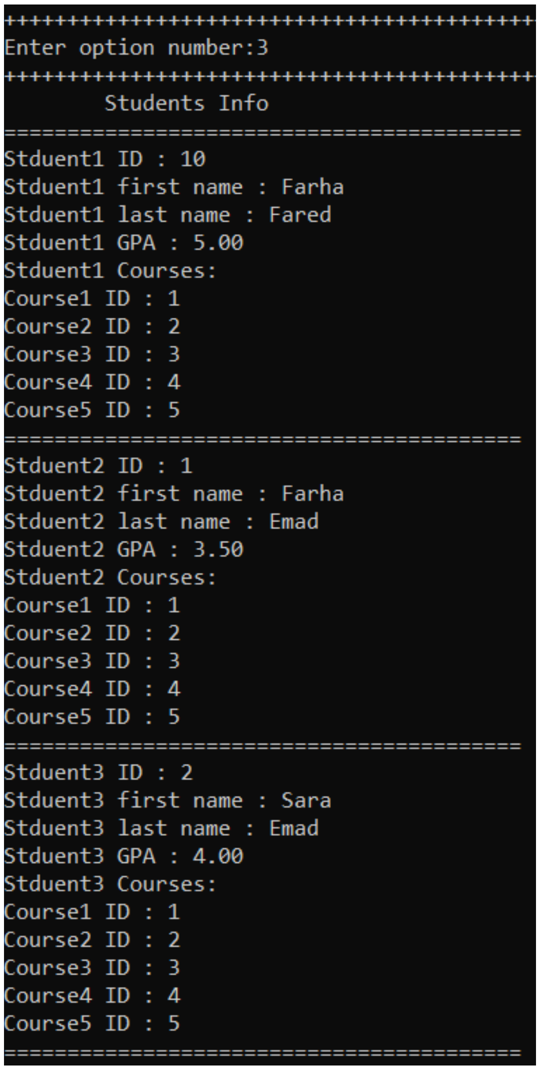
3) Adding students from a text file



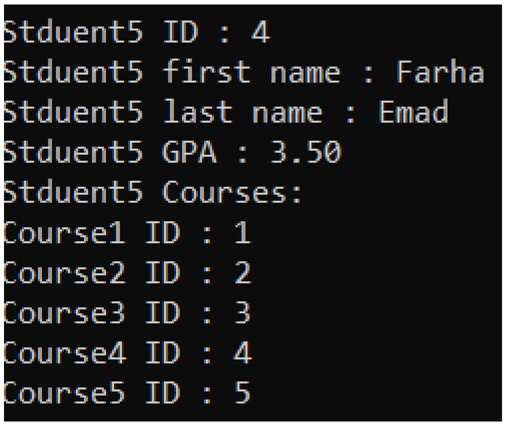
```
+++++++++++++++++++++++++++++++++++++++++++++++
Choose one of the following options
(1)Add new student manually
(2)Add students from text file
(3)Print all students
(4)Find student by id
(5)Find student by first name
(6)Find students registered in a specific course
(7)Total number of students
(8)Delete student by id
(9)Update student by id
(10)Exit
+++++++++++++++++++++++++++++++++++++++++++++++
Enter option number:2
+++++++++++++++++++++++++++++++++++++++++++++++
        Student(1) is added successfully
        Student(2) is added successfully
        [ERROR] ID is already taken
        Student(3) is added successfully
        Student(4) is added successfully

Total number of students = 5
+++++++++++++++++++++++++++++++++++++++++++++++
```

```
1 Farha Emad 3.5 1 2 3 4 5
2 Sara Emad 4.0 1 2 3 4 5
2 Ahmed Emad 3.5 1 2 3 4 5
3 Ramy Emad 3.5 1 2 3 4 5
4 Farha Emad 3.5 1 2 3 4 5
```

4) Printing all students in the database

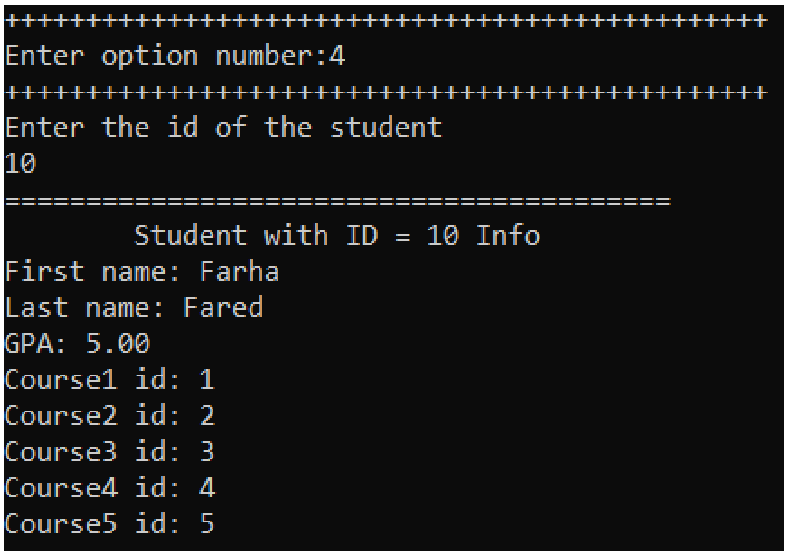


```
+++++++++++++++++++++++++++++++++++++++
Enter option number:3
+++++++++++++++++++++++++++++++++++++++
       Students Info
=======================================
Stduent1 ID : 10
Stduent1 first name : Farha
Stduent1 last name : Fared
Stduent1 GPA : 5.00
Stduent1 Courses:
Course1 ID : 1
Course2 ID : 2
Course3 ID : 3
Course4 ID : 4
Course5 ID : 5
=======================================
Stduent2 ID : 1
Stduent2 first name : Farha
Stduent2 last name : Emad
Stduent2 GPA : 3.50
Stduent2 Courses:
Course1 ID : 1
Course2 ID : 2
Course3 ID : 3
Course4 ID : 4
Course5 ID : 5
=======================================
Stduent3 ID : 2
Stduent3 first name : Sara
Stduent3 last name : Emad
Stduent3 GPA : 4.00
Stduent3 Courses:
Course1 ID : 1
Course2 ID : 2
Course3 ID : 3
Course4 ID : 4
Course5 ID : 5
=======================================
```



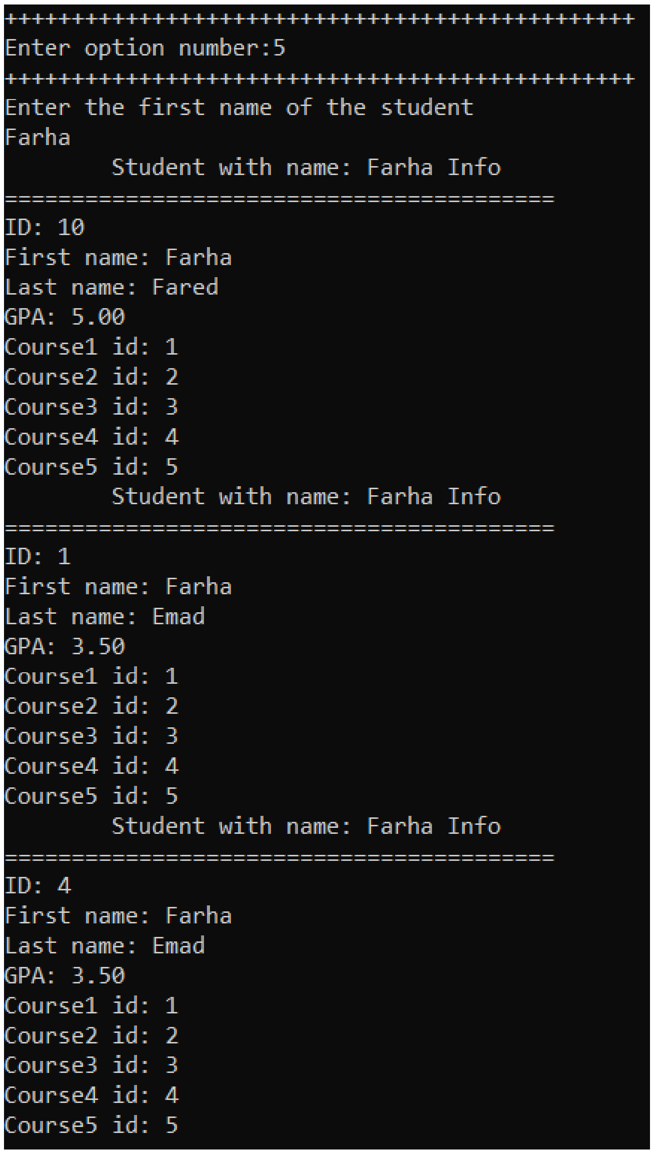
```
Stduent5 ID : 4
Stduent5 first name : Farha
Stduent5 last name : Emad
Stduent5 GPA : 3.50
Stduent5 Courses:
Course1 ID : 1
Course2 ID : 2
Course3 ID : 3
Course4 ID : 4
Course5 ID : 5
```

5) Find students by ID



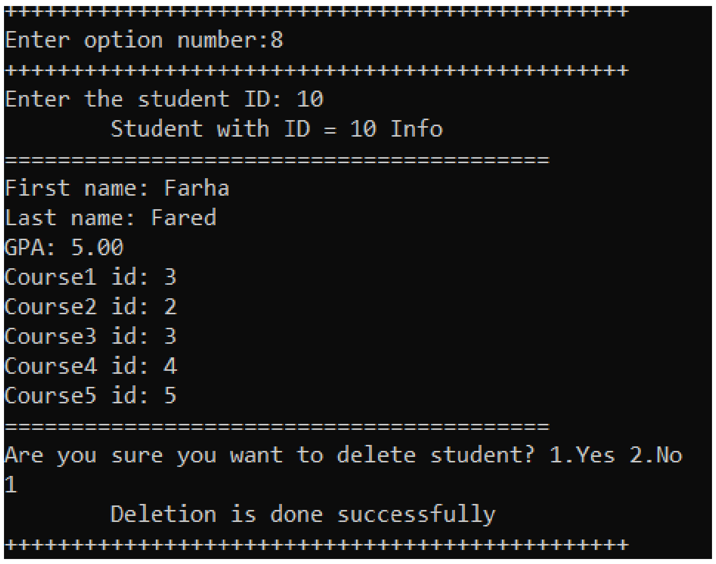
```
++++++++++++++++++++++++++++++++++++++++++++++
Enter option number:4
++++++++++++++++++++++++++++++++++++++++++++++
Enter the id of the student
10
========================================
        Student with ID = 10 Info
First name: Farha
Last name: Fared
GPA: 5.00
Course1 id: 1
Course2 id: 2
Course3 id: 3
Course4 id: 4
Course5 id: 5
```

6) Find students with their first name

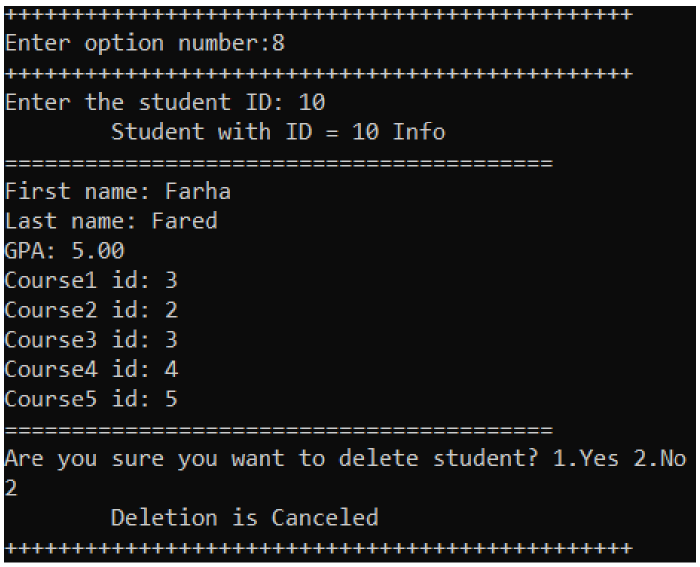


```
++++++++++++++++++++++++++++++++++++++++++++++
Enter option number:5
++++++++++++++++++++++++++++++++++++++++++++++
Enter the first name of the student
Farha
        Student with name: Farha Info
=========================================
ID: 10
First name: Farha
Last name: Fared
GPA: 5.00
Course1 id: 1
Course2 id: 2
Course3 id: 3
Course4 id: 4
Course5 id: 5
        Student with name: Farha Info
=========================================
ID: 1
First name: Farha
Last name: Emad
GPA: 3.50
Course1 id: 1
Course2 id: 2
Course3 id: 3
Course4 id: 4
Course5 id: 5
        Student with name: Farha Info
=========================================
ID: 4
First name: Farha
Last name: Emad
GPA: 3.50
Course1 id: 1
Course2 id: 2
Course3 id: 3
Course4 id: 4
Course5 id: 5
```

7) Deletion and canceling deletion

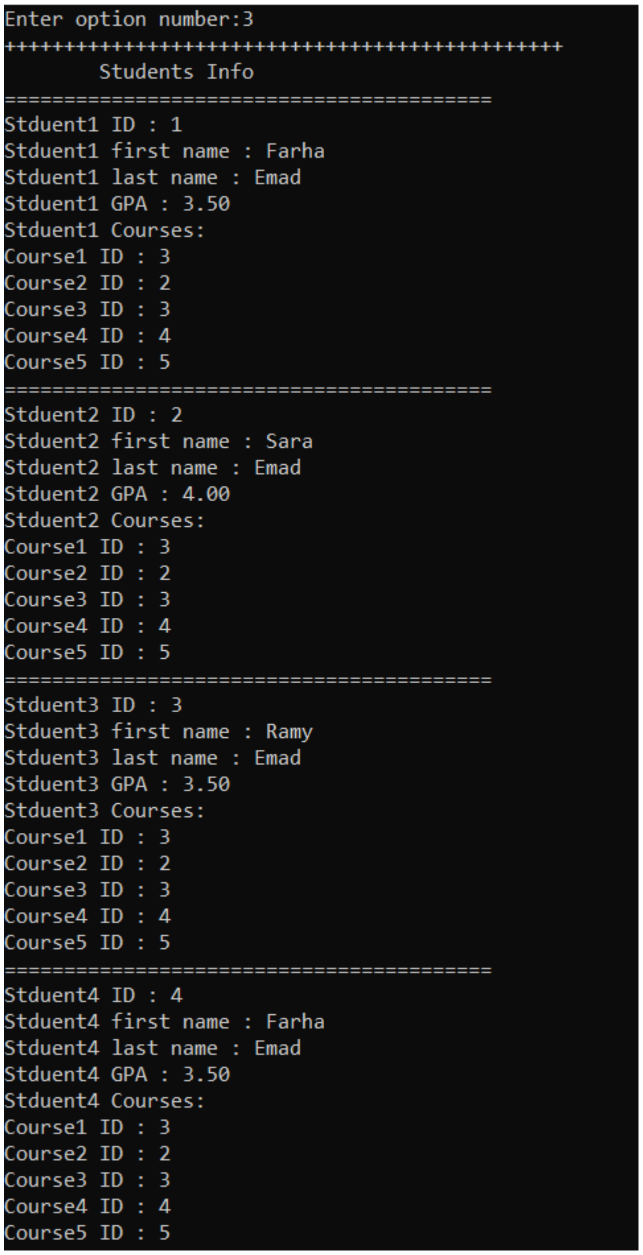


```
+++++++++++++++++++++++++++++++++++++++++++++++
Enter option number:8
+++++++++++++++++++++++++++++++++++++++++++++++
Enter the student ID: 10
        Student with ID = 10 Info
=========================================
First name: Farha
Last name: Fared
GPA: 5.00
Course1 id: 3
Course2 id: 2
Course3 id: 3
Course4 id: 4
Course5 id: 5
=========================================
Are you sure you want to delete student? 1.Yes 2.No
1
        Deletion is done successfully
+++++++++++++++++++++++++++++++++++++++++++++++
```



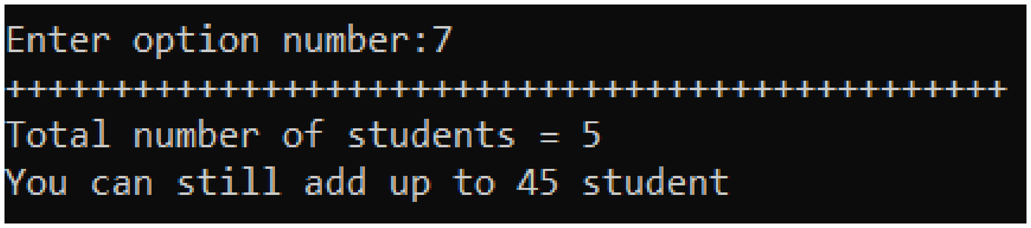
```
+++++++++++++++++++++++++++++++++++++++++++++++
Enter option number:8
+++++++++++++++++++++++++++++++++++++++++++++++
Enter the student ID: 10
        Student with ID = 10 Info
=========================================
First name: Farha
Last name: Fared
GPA: 5.00
Course1 id: 3
Course2 id: 2
Course3 id: 3
Course4 id: 4
Course5 id: 5
=========================================
Are you sure you want to delete student? 1.Yes 2.No
2
        Deletion is Canceled
+++++++++++++++++++++++++++++++++++++++++++++++
```

8) After Deletion



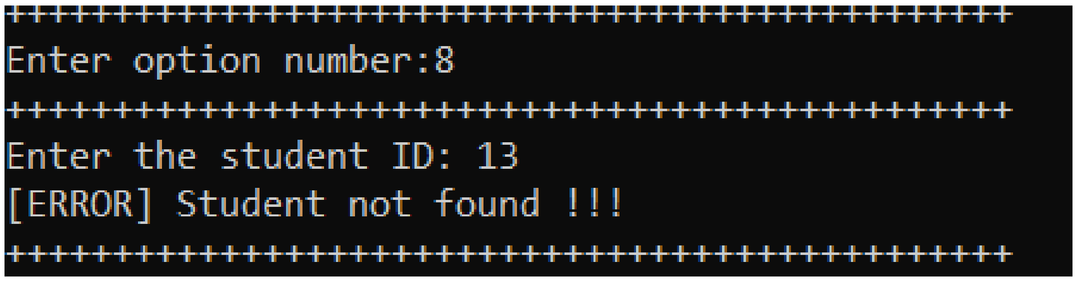
```
Enter option number:3
+++++++++++++++++++++++++++++++++++++++++++++++
        Students Info
=========================================
Stduent1 ID : 1
Stduent1 first name : Farha
Stduent1 last name : Emad
Stduent1 GPA : 3.50
Stduent1 Courses:
Course1 ID : 3
Course2 ID : 2
Course3 ID : 3
Course4 ID : 4
Course5 ID : 5
=========================================
Stduent2 ID : 2
Stduent2 first name : Sara
Stduent2 last name : Emad
Stduent2 GPA : 4.00
Stduent2 Courses:
Course1 ID : 3
Course2 ID : 2
Course3 ID : 3
Course4 ID : 4
Course5 ID : 5
=========================================
Stduent3 ID : 3
Stduent3 first name : Ramy
Stduent3 last name : Emad
Stduent3 GPA : 3.50
Stduent3 Courses:
Course1 ID : 3
Course2 ID : 2
Course3 ID : 3
Course4 ID : 4
Course5 ID : 5
=========================================
Stduent4 ID : 4
Stduent4 first name : Farha
Stduent4 last name : Emad
Stduent4 GPA : 3.50
Stduent4 Courses:
Course1 ID : 3
Course2 ID : 2
Course3 ID : 3
Course4 ID : 4
Course5 ID : 5
```

9) Count the total number of registered students



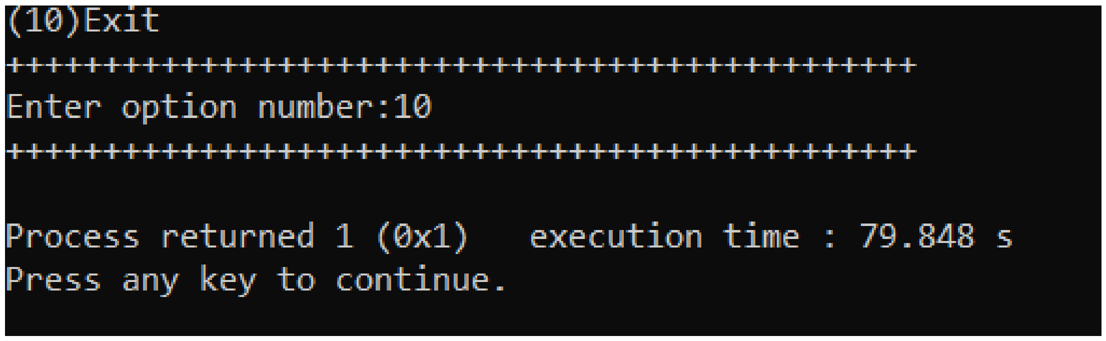
```
Enter option number:7
+++++++++++++++++++++++++++++++++++++++++++++++
Total number of students = 5
You can still add up to 45 student
```

10) Searching for a student with an invalid id



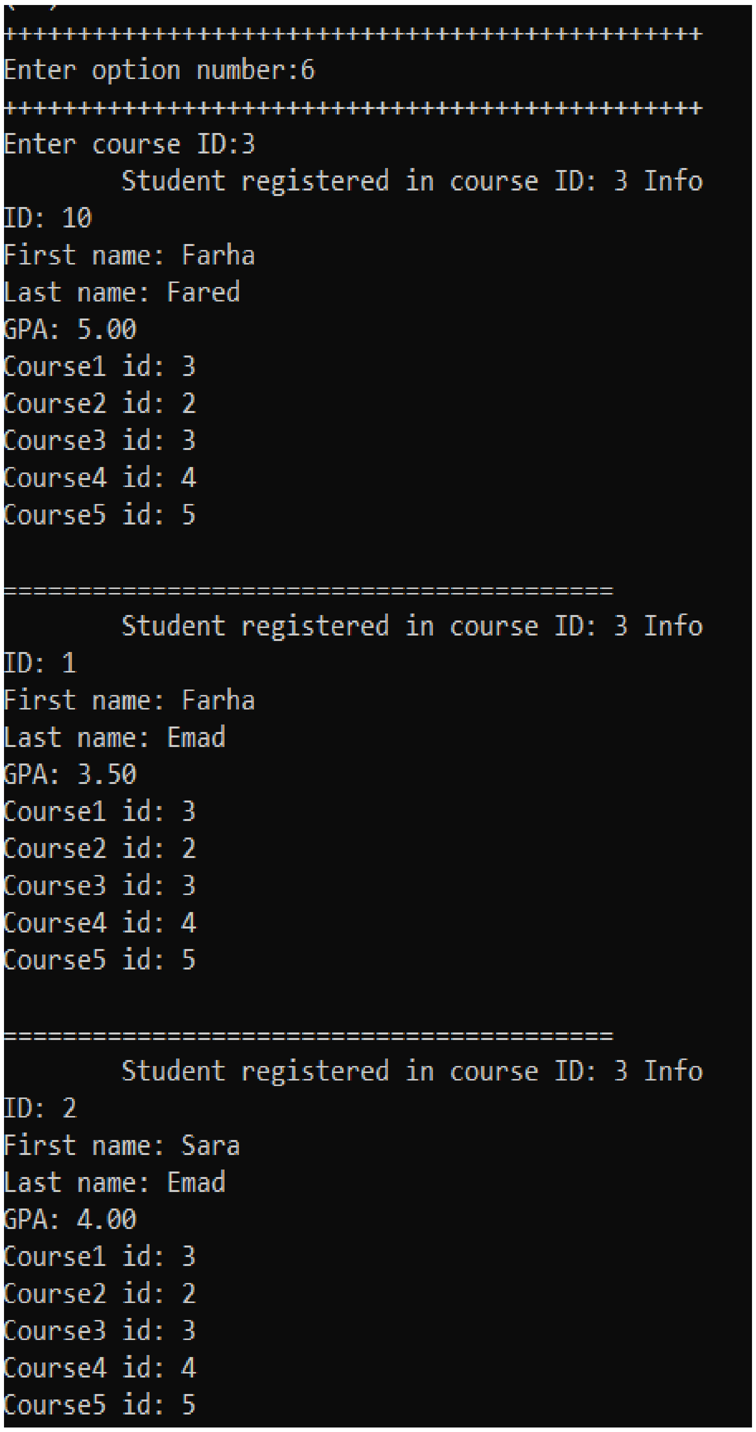
```
+++++++++++++++++++++++++++++++++++++++++++++++
Enter option number:8
+++++++++++++++++++++++++++++++++++++++++++++++
Enter the student ID: 13
[ERROR] Student not found !!!
+++++++++++++++++++++++++++++++++++++++++++++++
```

11) Exiting the program

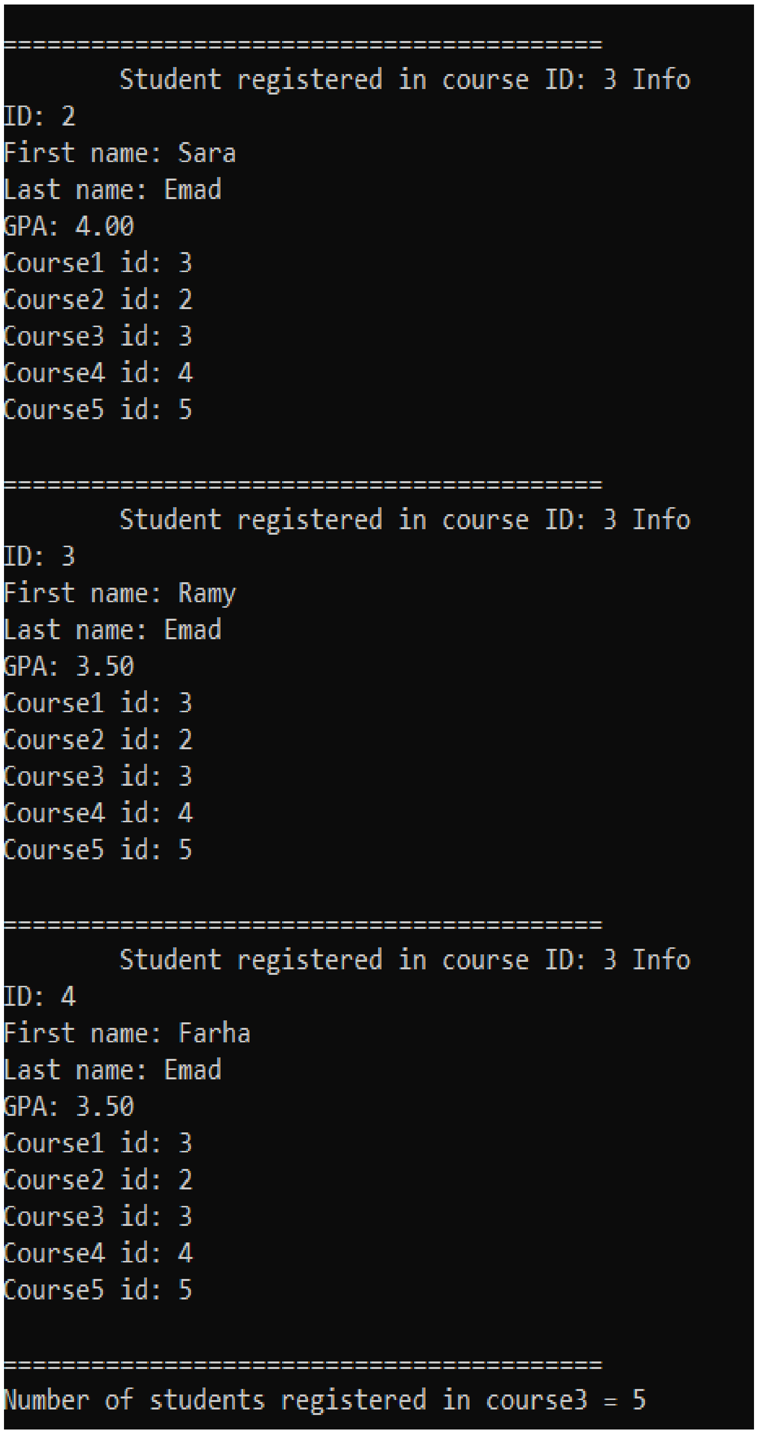


```
(10)Exit
+++++++++++++++++++++++++++++++++++++++++++++++
Enter option number:10
+++++++++++++++++++++++++++++++++++++++++++++++

Process returned 1 (0x1)   execution time : 79.848 s
Press any key to continue.
```

12) Updating specific data for specific student

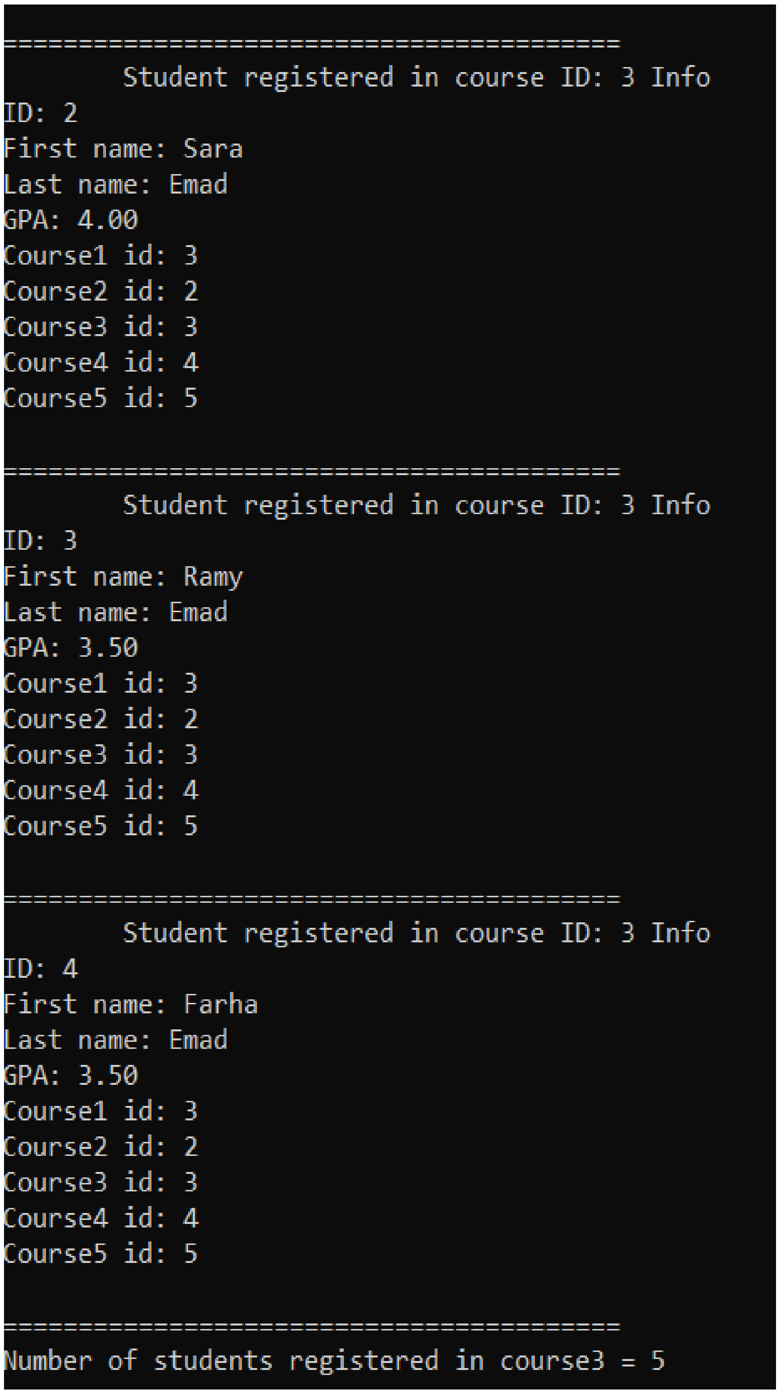


```
+++++++++++++++++++++++++++++++++++++++++++++++
Enter option number:6
+++++++++++++++++++++++++++++++++++++++++++++++
Enter course ID:3
        Student registered in course ID: 3 Info
ID: 10
First name: Farha
Last name: Fared
GPA: 5.00
Course1 id: 3
Course2 id: 2
Course3 id: 3
Course4 id: 4
Course5 id: 5

=========================================
        Student registered in course ID: 3 Info
ID: 1
First name: Farha
Last name: Emad
GPA: 3.50
Course1 id: 3
Course2 id: 2
Course3 id: 3
Course4 id: 4
Course5 id: 5

=========================================
        Student registered in course ID: 3 Info
ID: 2
First name: Sara
Last name: Emad
GPA: 4.00
Course1 id: 3
Course2 id: 2
Course3 id: 3
Course4 id: 4
Course5 id: 5
```

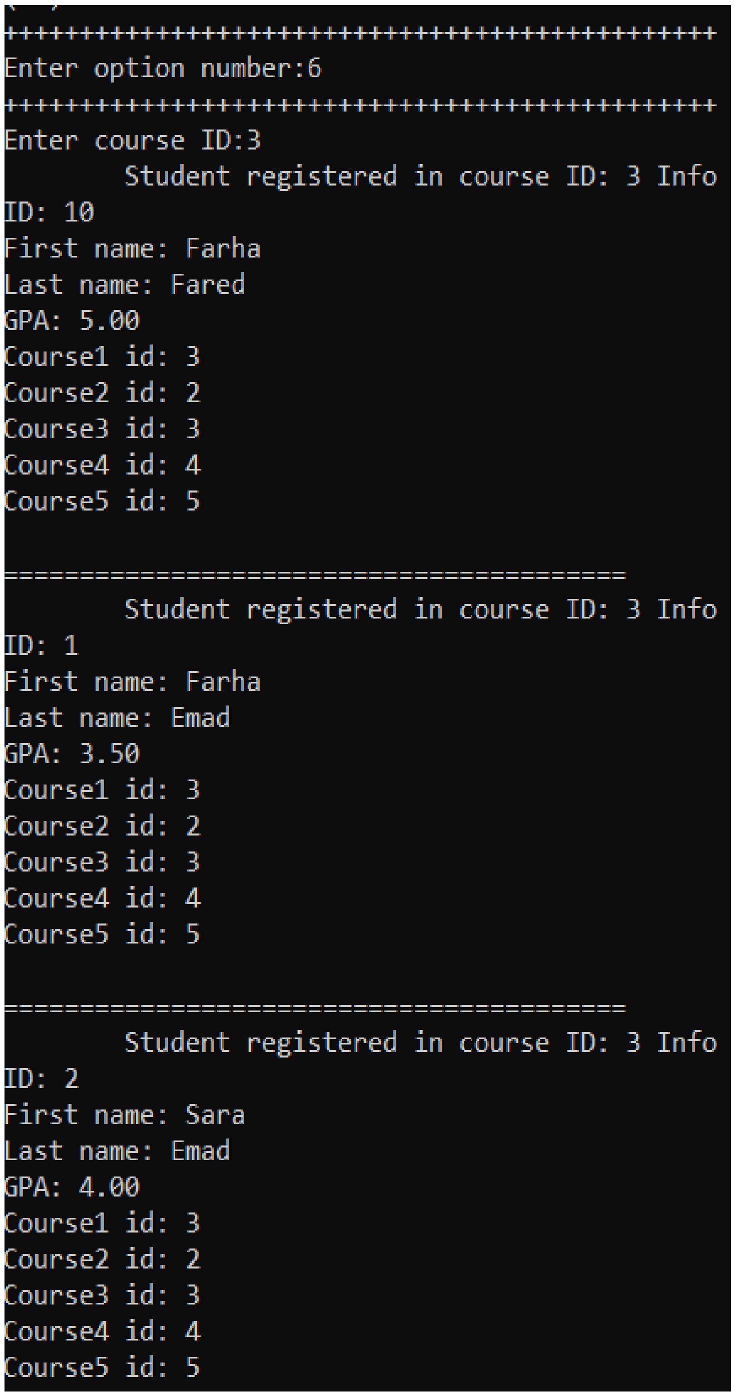


```
=========================================
        Student registered in course ID: 3 Info
ID: 2
First name: Sara
Last name: Emad
GPA: 4.00
Course1 id: 3
Course2 id: 2
Course3 id: 3
Course4 id: 4
Course5 id: 5

=========================================
        Student registered in course ID: 3 Info
ID: 3
First name: Ramy
Last name: Emad
GPA: 3.50
Course1 id: 3
Course2 id: 2
Course3 id: 3
Course4 id: 4
Course5 id: 5

=========================================
        Student registered in course ID: 3 Info
ID: 4
First name: Farha
Last name: Emad
GPA: 3.50
Course1 id: 3
Course2 id: 2
Course3 id: 3
Course4 id: 4
Course5 id: 5

=========================================
Number of students registered in course3 = 5
```

13) Searches for all students that registered for the same course using the course id.



```
=========================================
        Student registered in course ID: 3 Info
ID: 2
First name: Sara
Last name: Emad
GPA: 4.00
Course1 id: 3
Course2 id: 2
Course3 id: 3
Course4 id: 4
Course5 id: 5

=========================================
        Student registered in course ID: 3 Info
ID: 3
First name: Ramy
Last name: Emad
GPA: 3.50
Course1 id: 3
Course2 id: 2
Course3 id: 3
Course4 id: 4
Course5 id: 5

=========================================
        Student registered in course ID: 3 Info
ID: 4
First name: Farha
Last name: Emad
GPA: 3.50
Course1 id: 3
Course2 id: 2
Course3 id: 3
Course4 id: 4
Course5 id: 5

=========================================
Number of students registered in course3 = 5
```



```
+++++++++++++++++++++++++++++++++++++++++++++++
Enter option number:6
+++++++++++++++++++++++++++++++++++++++++++++++
Enter course ID:3
        Student registered in course ID: 3 Info
ID: 10
First name: Farha
Last name: Fared
GPA: 5.00
Course1 id: 3
Course2 id: 2
Course3 id: 3
Course4 id: 4
Course5 id: 5

=========================================
        Student registered in course ID: 3 Info
ID: 1
First name: Farha
Last name: Emad
GPA: 3.50
Course1 id: 3
Course2 id: 2
Course3 id: 3
Course4 id: 4
Course5 id: 5

=========================================
        Student registered in course ID: 3 Info
ID: 2
First name: Sara
Last name: Emad
GPA: 4.00
Course1 id: 3
Course2 id: 2
Course3 id: 3
Course4 id: 4
Course5 id: 5
```