Mastering Embedded System Online Diploma

First term (Final Project(2))

Project : Students Database Using Queue

Eng. Omar Ahmed

My Profile

…..

Table of Contents

Introduction

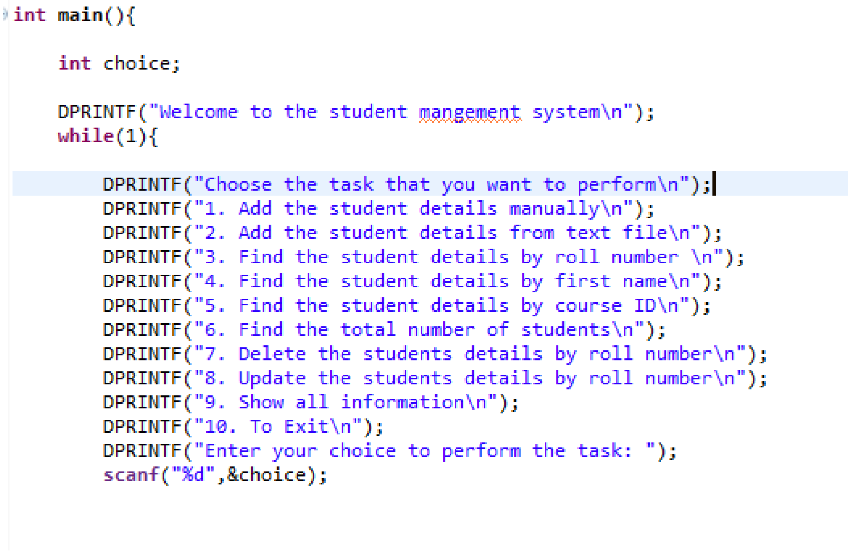
Case Study:
Creating a student's database where you can:

1. Add the student details manually
2. Add the student details from text file
3. Find the student details by roll number
4. Find the student details by first name
5. Find the student details by course ID
6. Find the total number of students
7. Delete the student's details by roll number
8. Update the students' details by roll number
9. Show all information

the data structure used is Queue (FIFO) first in first output
the code is divided into a Queue header file where the main definitions for the
queue implementation is present
The other header file is a student's database header file where all the APIS is
present and it’s included in the main function

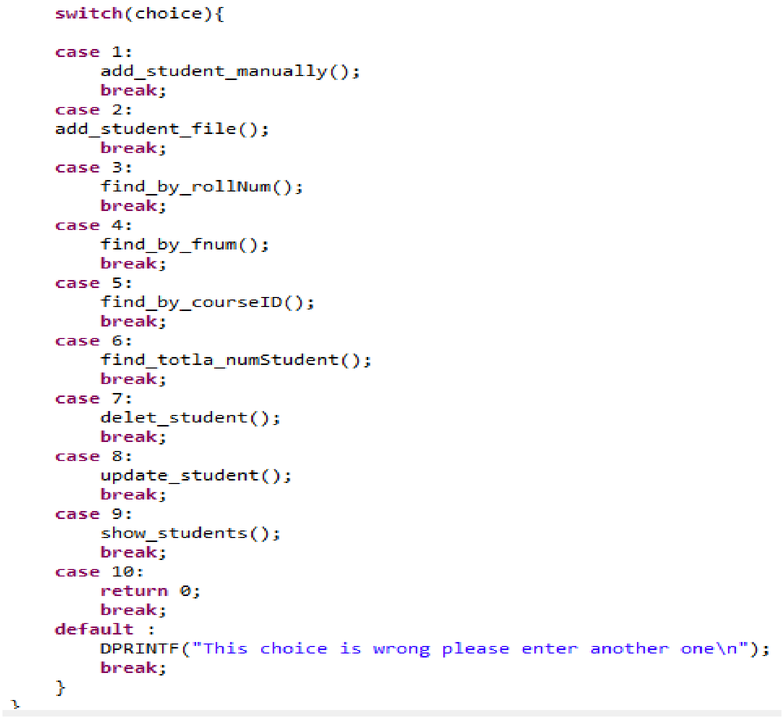
Chapter one: Code Implementation

1. Main:



```
int main(){

    int choice;

    DPRINTF("Welcome to the student mangement system\n");
    while(1){

        DPRINTF("Choose the task that you want to perform\n");
        DPRINTF("1. Add the student details manually\n");
        DPRINTF("2. Add the student details from text file\n");
        DPRINTF("3. Find the student details by roll number \n");
        DPRINTF("4. Find the student details by first name\n");
        DPRINTF("5. Find the student details by course ID\n");
        DPRINTF("6. Find the total number of students\n");
        DPRINTF("7. Delete the students details by roll number\n");
        DPRINTF("8. Update the students details by roll number\n");
        DPRINTF("9. Show all information\n");
        DPRINTF("10. To Exit\n");
        DPRINTF("Enter your choice to perform the task: ");
        scanf("%d",&choice);
```

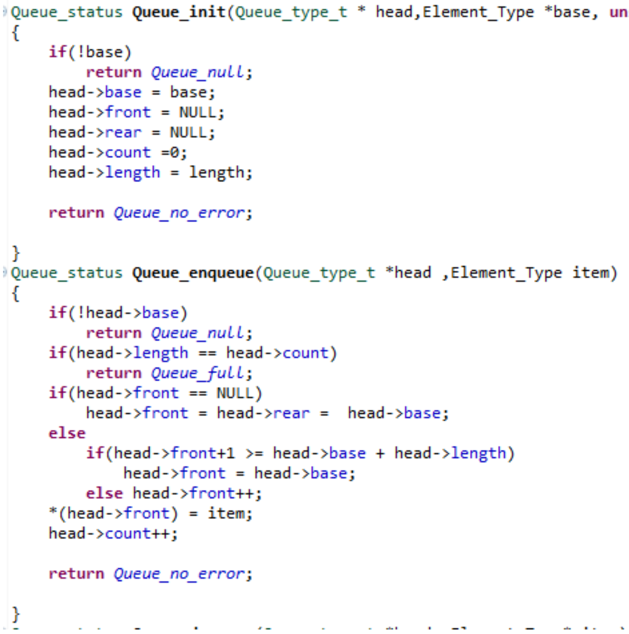
رسم توضيحي 1main1.1



```
    switch(choice){

    case 1:
        add_student_manually();
        break;
    case 2:
    add_student_file();
        break;
    case 3:
        find_by_rollNum();
        break;
    case 4:
        find_by_fnum();
        break;
    case 5:
        find_by_courseID();
        break;
    case 6:
        find_totla_numStudent();
        break;
    case 7:
        delet_student();
        break;
    case 8:
        update_student();
        break;
    case 9:
        show_students();
        break;
    case 10:
        return 0;
        break;
    default :
        DPRINTF("This choice is wrong please enter another one\n");
        break;
    }
}
```

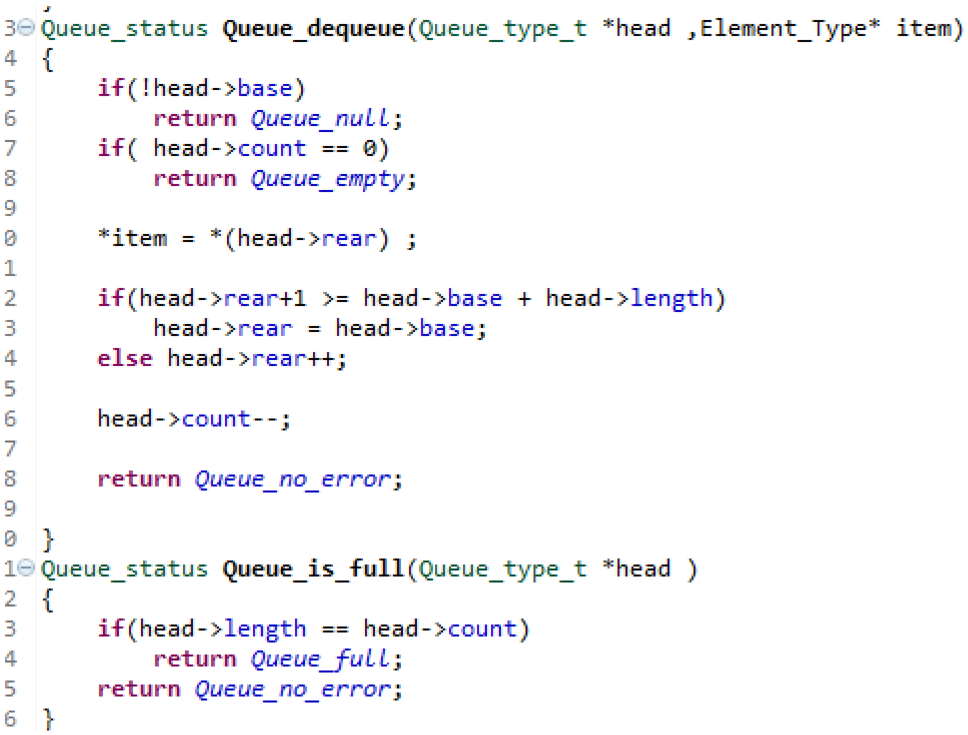
رسم توضيحي 2main1.2

2.Queue:



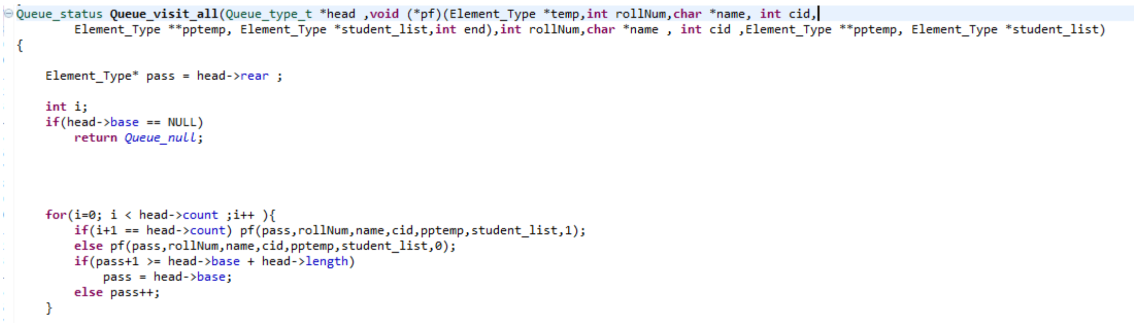
```
Queue_status Queue_init(Queue_type_t * head,Element_Type *base, unsigned int length)
{
    if(!base)
        return Queue_null;
    head->base = base;
    head->front = NULL;
    head->rear = NULL;
    head->count =0;
    head->length = length;

    return Queue_no_error;

}
Queue_status Queue_enqueue(Queue_type_t *head ,Element_Type item)
{
    if(!head->base)
        return Queue_null;
    if(head->length == head->count)
        return Queue_full;
    if(head->front == NULL)
        head->front = head->rear =  head->base;
    else
        if(head->front+1 >= head->base + head->length)
            head->front = head->base;
        else head->front++;
    *(head->front) = item;
    head->count++;

    return Queue_no_error;

}
```

3Queue1.1 رسم توضيحي



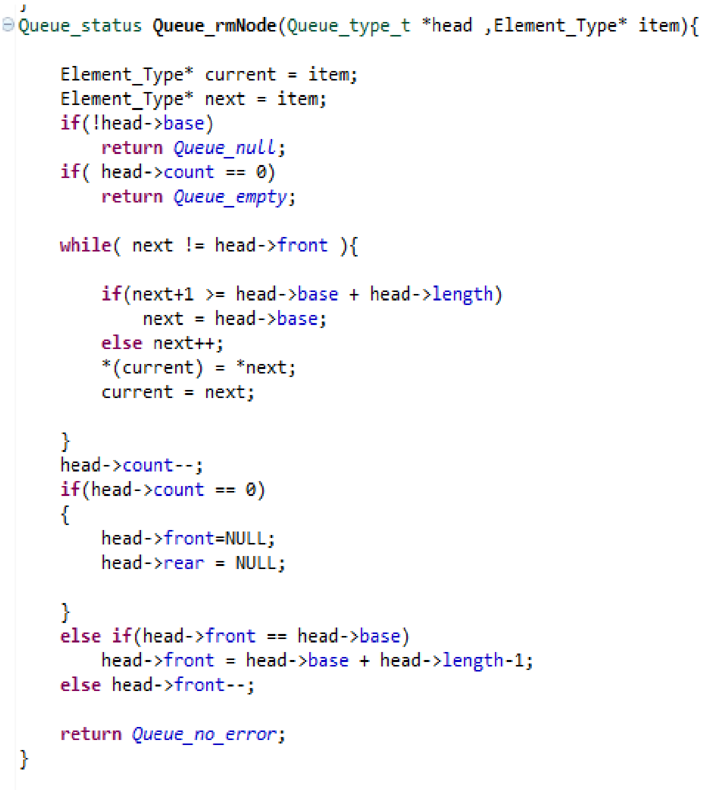
```
Queue_status Queue_dequeue(Queue_type_t *head ,Element_Type* item)
{
    if(!head->base)
        return Queue_null;
    if( head->count == 0)
        return Queue_empty;

    *item = *(head->rear) ;

    if(head->rear+1 >= head->base + head->length)
        head->rear = head->base;
    else head->rear++;

    head->count--;

    return Queue_no_error;

}
Queue_status Queue_is_full(Queue_type_t *head )
{
    if(head->length == head->count)
        return Queue_full;
    return Queue_no_error;
}
```

4Queue1.2 رسم توضيحي



```c
Queue_status Queue_visit_all(Queue_type_t *head ,void (*pf)(Element_Type *temp,int rollNum,char *name, int cid,
        Element_Type **pptemp, Element_Type *student_list,int end),int rollNum,char *name , int cid ,Element_Type **pptemp, Element_Type *student_list)
{

    Element_Type* pass = head->rear ;

    int i;
    if(head->base == NULL)
        return Queue_null;



    for(i=0; i < head->count ;i++ ){
        if(i+1 == head->count) pf(pass,rollNum,name,cid,pptemp,student_list,1);
        else pf(pass,rollNum,name,cid,pptemp,student_list,0);
        if(pass+1 >= head->base + head->length)
            pass = head->base;
        else pass++;
    }
```

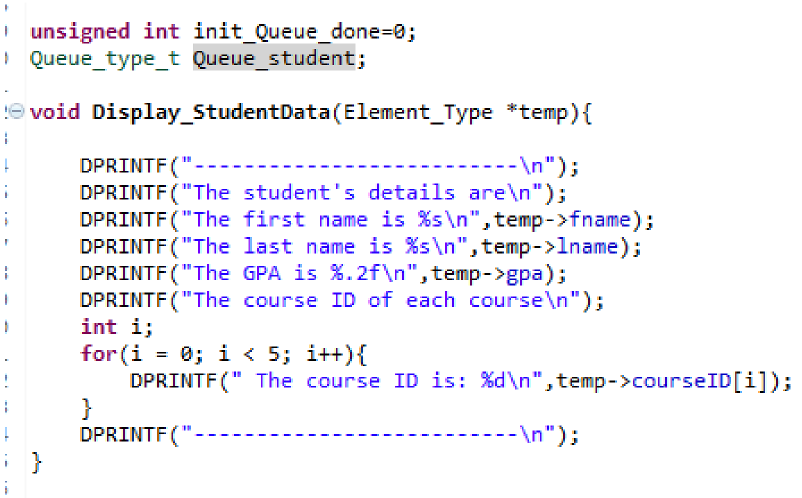
رسم توضيحي 5Queue1.3



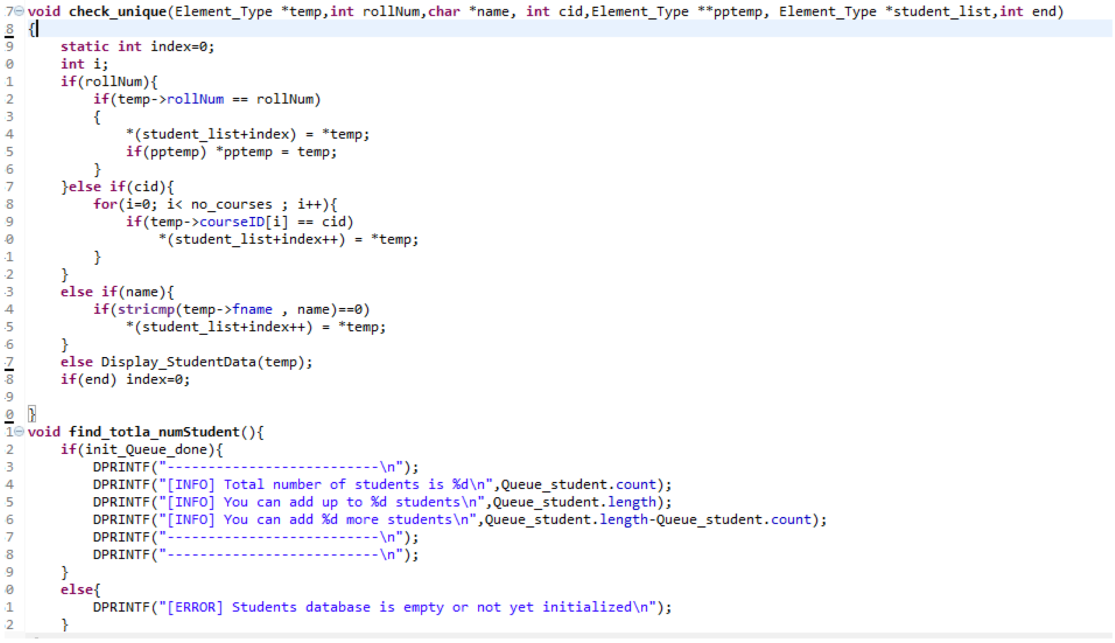
```c
Queue_status Queue_rmNode(Queue_type_t *head ,Element_Type* item){

    Element_Type* current = item;
    Element_Type* next = item;
    if(!head->base)
        return Queue_null;
    if( head->count == 0)
        return Queue_empty;

    while( next != head->front ){

        if(next+1 >= head->base + head->length)
            next = head->base;
        else next++;
        *(current) = *next;
        current = next;

    }
    head->count--;
    if(head->count == 0)
    {
        head->front=NULL;
        head->rear = NULL;

    }
    else if(head->front == head->base)
        head->front = head->base + head->length-1;
    else head->front--;

    return Queue_no_error;
}
```

رسم توضيحي 6Queue1.4

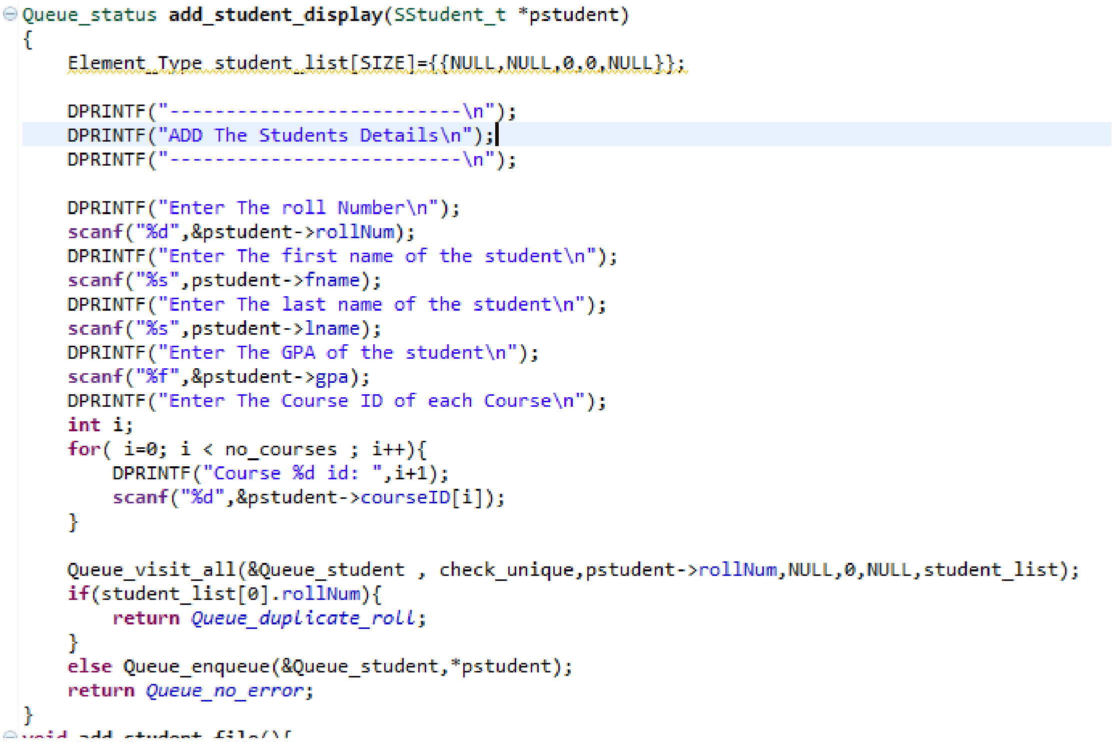
3. Students Database:



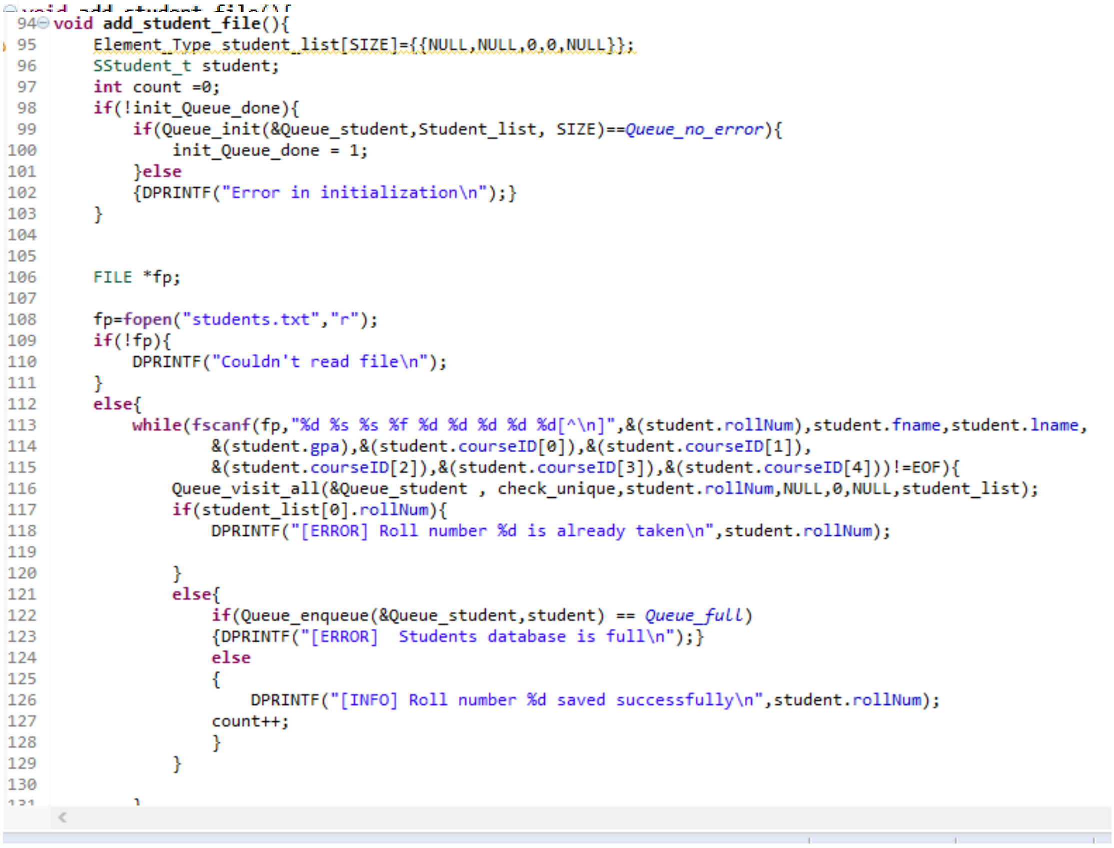
```c
unsigned int init_Queue_done=0;
Queue_type_t Queue_student;

void Display_StudentData(Element_Type *temp){

    DPRINTF("--------------------------\n");
    DPRINTF("The student's details are\n");
    DPRINTF("The first name is %s\n",temp->fname);
    DPRINTF("The last name is %s\n",temp->lname);
    DPRINTF("The GPA is %.2f\n",temp->gpa);
    DPRINTF("The course ID of each course\n");
    int i;
    for(i = 0; i < 5; i++){
        DPRINTF(" The course ID is: %d\n",temp->courseID[i]);
    }
    DPRINTF("--------------------------\n");
}
```



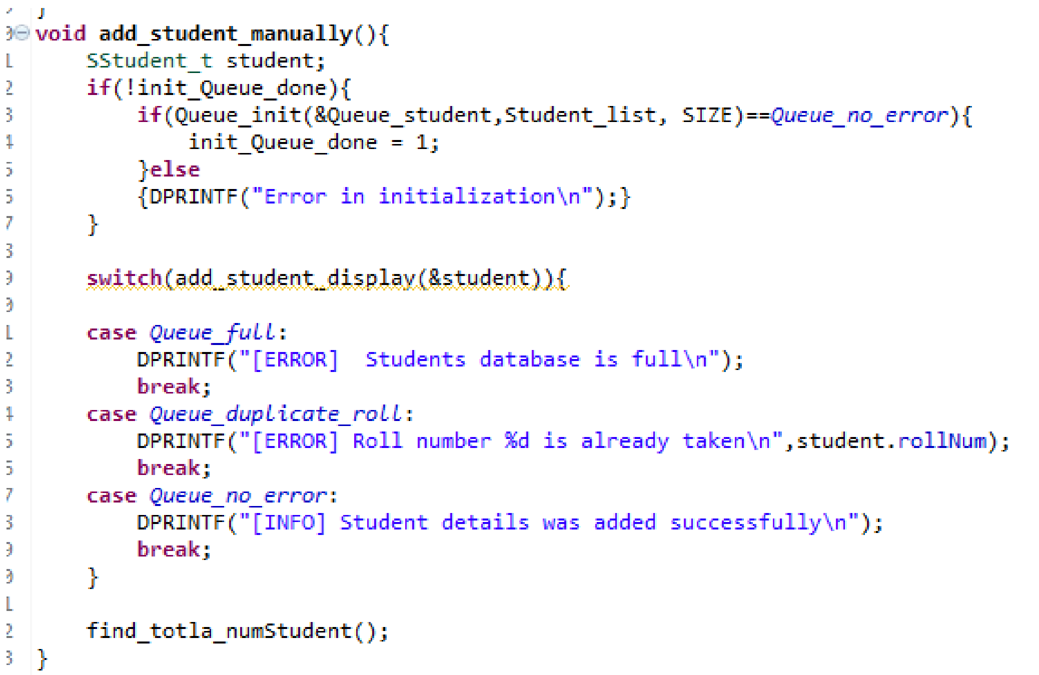
```c
void check_unique(Element_Type *temp,int rollNum,char *name, int cid,Element_Type **pptemp, Element_Type *student_list,int end)
{
    static int index=0;
    int i;
    if(rollNum){
        if(temp->rollNum == rollNum)
        {
            *(student_list+index) = *temp;
            if(pptemp) *pptemp = temp;
        }
    }else if(cid){
        for(i=0; i< no_courses ; i++){
            if(temp->courseID[i] == cid)
                *(student_list+index++) = *temp;
        }
    }
    else if(name){
        if(stricmp(temp->fname , name)==0)
            *(student_list+index++) = *temp;
    }
    else Display_StudentData(temp);
    if(end) index=0;

}
void find_totla_numStudent(){
    if(init_Queue_done){
        DPRINTF("-------------------------\n");
        DPRINTF("[INFO] Total number of students is %d\n",Queue_student.count);
        DPRINTF("[INFO] You can add up to %d students\n",Queue_student.length);
        DPRINTF("[INFO] You can add %d more students\n",Queue_student.length-Queue_student.count);
        DPRINTF("-------------------------\n");
        DPRINTF("-------------------------\n");
    }
    else{
        DPRINTF("[ERROR] Students database is empty or not yet initialized\n");
    }
```



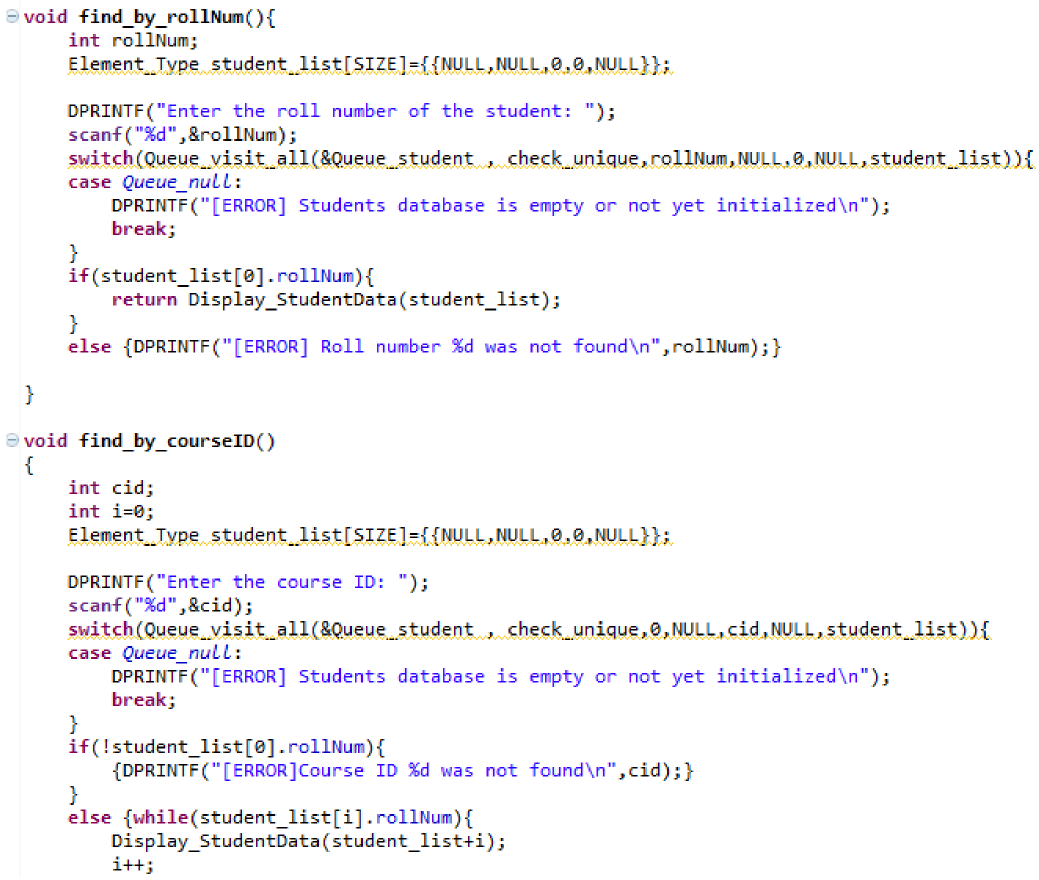
```c
Queue_status add_student_display(SStudent_t *pstudent)
{
    Element_Type student_list[SIZE]={{NULL,NULL,0,0,NULL}};

    DPRINTF("-------------------------\n");
    DPRINTF("ADD The Students Details\n");
    DPRINTF("-------------------------\n");

    DPRINTF("Enter The roll Number\n");
    scanf("%d",&pstudent->rollNum);
    DPRINTF("Enter The first name of the student\n");
    scanf("%s",pstudent->fname);
    DPRINTF("Enter The last name of the student\n");
    scanf("%s",pstudent->lname);
    DPRINTF("Enter The GPA of the student\n");
    scanf("%f",&pstudent->gpa);
    DPRINTF("Enter The Course ID of each Course\n");
    int i;
    for( i=0; i < no_courses ; i++){
        DPRINTF("Course %d id: ",i+1);
        scanf("%d",&pstudent->courseID[i]);
    }

    Queue_visit_all(&Queue_student , check_unique,pstudent->rollNum,NULL,0,NULL,student_list);
    if(student_list[0].rollNum){
        return Queue_duplicate_roll;
    }
    else Queue_enqueue(&Queue_student,*pstudent);
    return Queue_no_error;
}
```



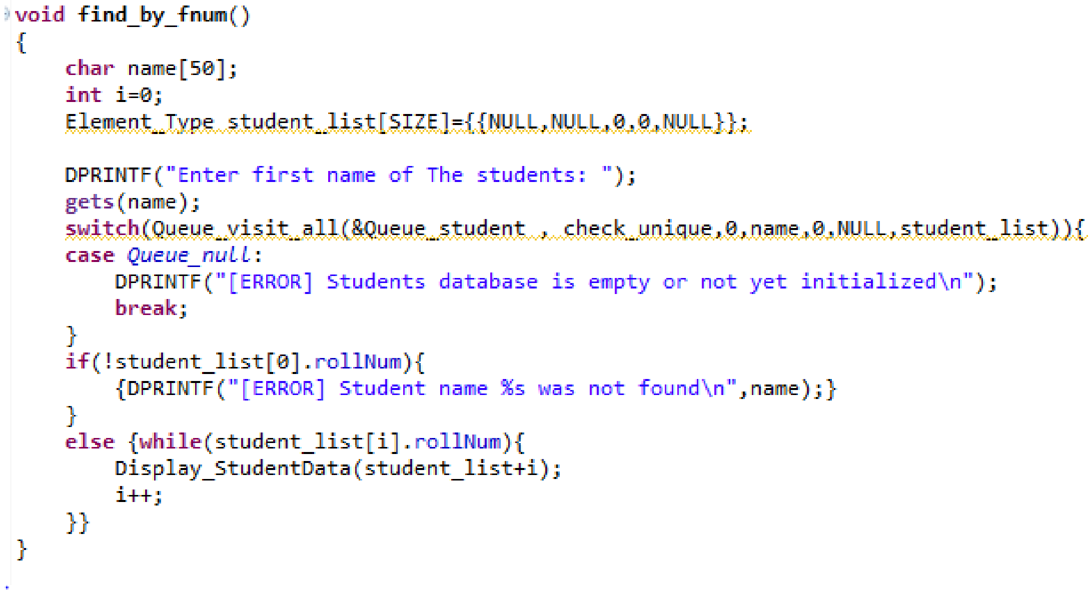
```c
void add_student_file(){
    Element_Type student_list[SIZE]={{NULL,NULL,0,0,NULL}};
    SStudent_t student;
    int count =0;
    if(!init_Queue_done){
        if(Queue_init(&Queue_student,Student_list, SIZE)==Queue_no_error){
            init_Queue_done = 1;
        }else
        {DPRINTF("Error in initialization\n");}
    }


    FILE *fp;

    fp=fopen("students.txt","r");
    if(!fp){
        DPRINTF("Couldn't read file\n");
    }
    else{
        while(fscanf(fp,"%d %s %s %f %d %d %d %d %d[^\n]",&(student.rollNum),student.fname,student.lname,
                &(student.gpa),&(student.courseID[0]),&(student.courseID[1]),
                &(student.courseID[2]),&(student.courseID[3]),&(student.courseID[4]))!=EOF){
            Queue_visit_all(&Queue_student , check_unique,student.rollNum,NULL,0,NULL,student_list);
            if(student_list[0].rollNum){
                DPRINTF("[ERROR] Roll number %d is already taken\n",student.rollNum);

            }
            else{
                if(Queue_enqueue(&Queue_student,student) == Queue_full)
                {DPRINTF("[ERROR]  Students database is full\n");}
                else
                {
                    DPRINTF("[INFO] Roll number %d saved successfully\n",student.rollNum);
                count++;
                }
            }

        }
```



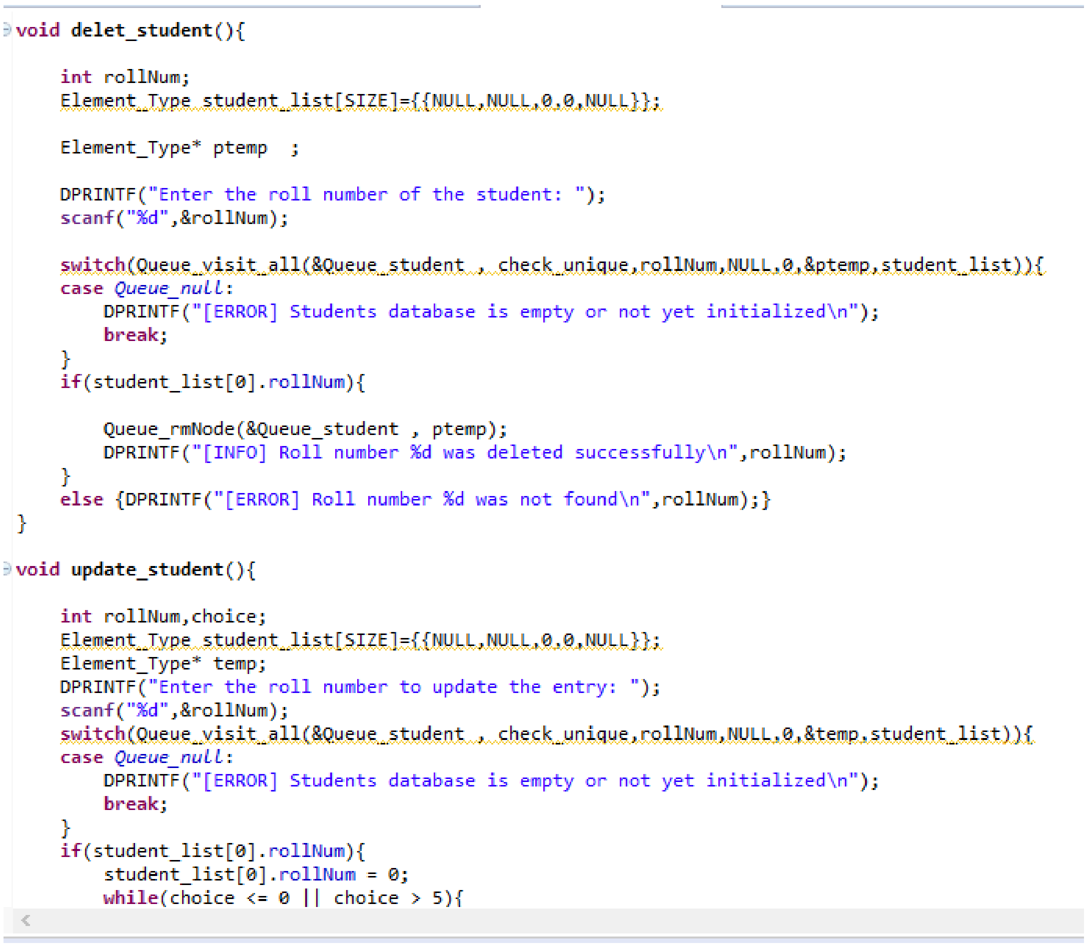
```
void add_student_manually(){
    SStudent_t student;
    if(!init_Queue_done){
        if(Queue_init(&Queue_student,Student_list, SIZE)==Queue_no_error){
            init_Queue_done = 1;
        }else
        {DPRINTF("Error in initialization\n");}
    }

    switch(add_student_display(&student)){

    case Queue_full:
        DPRINTF("[ERROR]  Students database is full\n");
        break;
    case Queue_duplicate_roll:
        DPRINTF("[ERROR] Roll number %d is already taken\n",student.rollNum);
        break;
    case Queue_no_error:
        DPRINTF("[INFO] Student details was added successfully\n");
        break;
    }

    find_totla_numStudent();
}
```



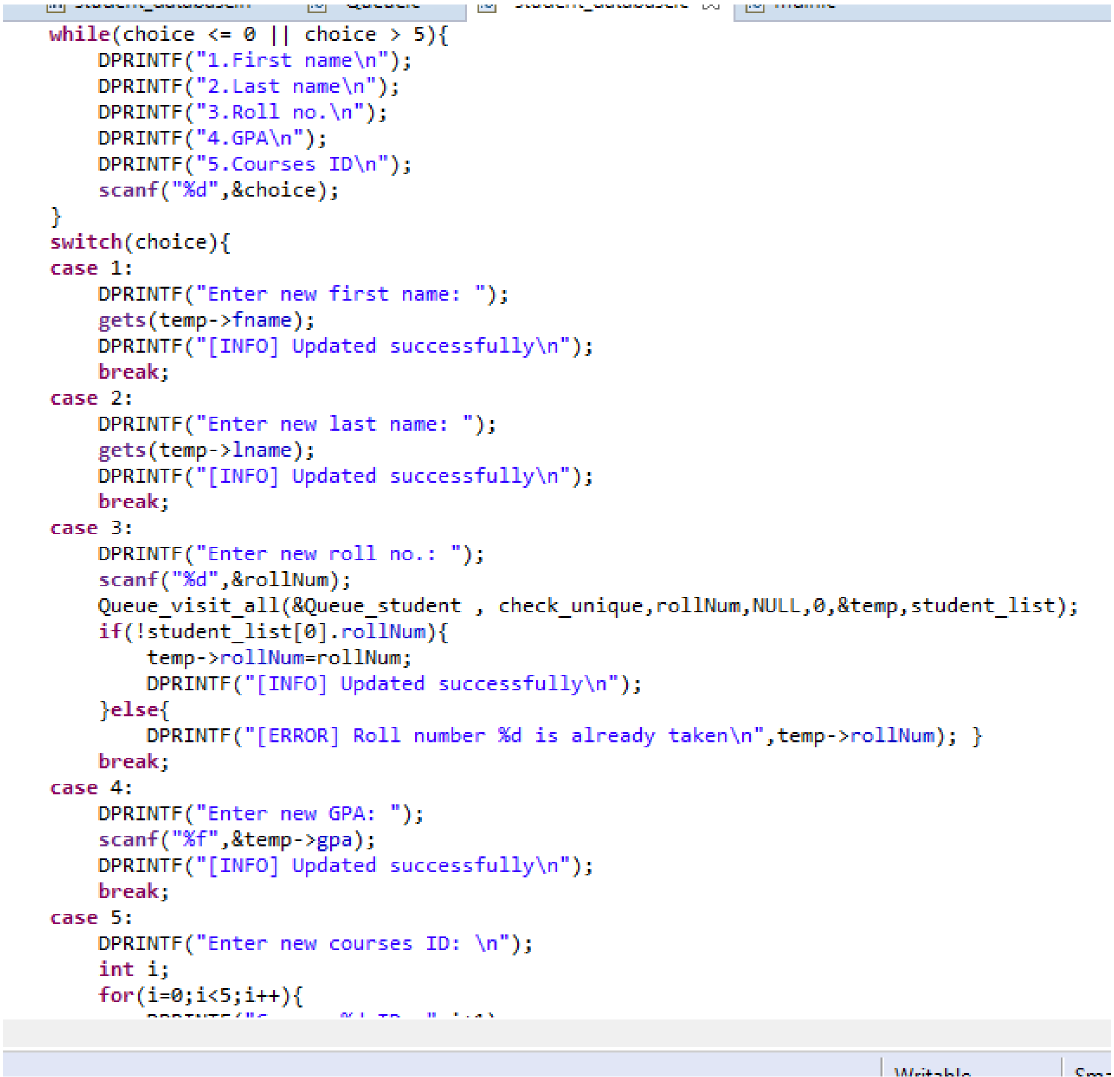
```
void find_by_rollNum(){
    int rollNum;
    Element_Type student_list[SIZE]={{NULL,NULL,0,0,NULL}};

    DPRINTF("Enter the roll number of the student: ");
    scanf("%d",&rollNum);
    switch(Queue_visit_all(&Queue_student , check_unique,rollNum,NULL,0,NULL,student_list)){
    case Queue_null:
        DPRINTF("[ERROR] Students database is empty or not yet initialized\n");
        break;
    }
    if(student_list[0].rollNum){
        return Display_StudentData(student_list);
    }
    else {DPRINTF("[ERROR] Roll number %d was not found\n",rollNum);}

}

void find_by_courseID()
{
    int cid;
    int i=0;
    Element_Type student_list[SIZE]={{NULL,NULL,0,0,NULL}};

    DPRINTF("Enter the course ID: ");
    scanf("%d",&cid);
    switch(Queue_visit_all(&Queue_student , check_unique,0,NULL,cid,NULL,student_list)){
    case Queue_null:
        DPRINTF("[ERROR] Students database is empty or not yet initialized\n");
        break;
    }
    if(!student_list[0].rollNum){
        {DPRINTF("[ERROR]Course ID %d was not found\n",cid);}
    }
    else {while(student_list[i].rollNum){
        Display_StudentData(student_list+i);
        i++;
```



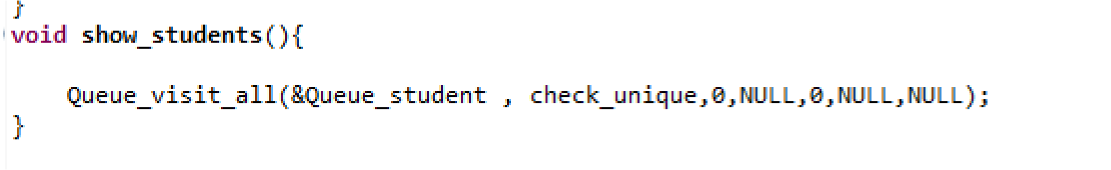
```
void find_by_fnum()
{
    char name[50];
    int i=0;
    Element_Type student_list[SIZE]={{NULL,NULL,0,0,NULL}};

    DPRINTF("Enter first name of The students: ");
    gets(name);
    switch(Queue_visit_all(&Queue_student , check_unique,0,name,0,NULL,student_list)){
    case Queue_null:
        DPRINTF("[ERROR] Students database is empty or not yet initialized\n");
        break;
    }
    if(!student_list[0].rollNum){
        {DPRINTF("[ERROR] Student name %s was not found\n",name);}
    }
    else {while(student_list[i].rollNum){
        Display_StudentData(student_list+i);
        i++;
    }}
}
```



```
void delet_student(){

    int rollNum;
    Element_Type student_list[SIZE]={{NULL,NULL,0,0,NULL}};

    Element_Type* ptemp   ;

    DPRINTF("Enter the roll number of the student: ");
    scanf("%d",&rollNum);

    switch(Queue_visit_all(&Queue_student , check_unique,rollNum,NULL,0,&ptemp,student_list)){
    case Queue_null:
        DPRINTF("[ERROR] Students database is empty or not yet initialized\n");
        break;
    }
    if(student_list[0].rollNum){

        Queue_rmNode(&Queue_student , ptemp);
        DPRINTF("[INFO] Roll number %d was deleted successfully\n",rollNum);
    }
    else {DPRINTF("[ERROR] Roll number %d was not found\n",rollNum);}
}

void update_student(){

    int rollNum,choice;
    Element_Type student_list[SIZE]={{NULL,NULL,0,0,NULL}};
    Element_Type* temp;
    DPRINTF("Enter the roll number to update the entry: ");
    scanf("%d",&rollNum);
    switch(Queue_visit_all(&Queue_student , check_unique,rollNum,NULL,0,&temp,student_list)){
    case Queue_null:
        DPRINTF("[ERROR] Students database is empty or not yet initialized\n");
        break;
    }
    if(student_list[0].rollNum){
        student_list[0].rollNum = 0;
        while(choice <= 0 || choice > 5){
```



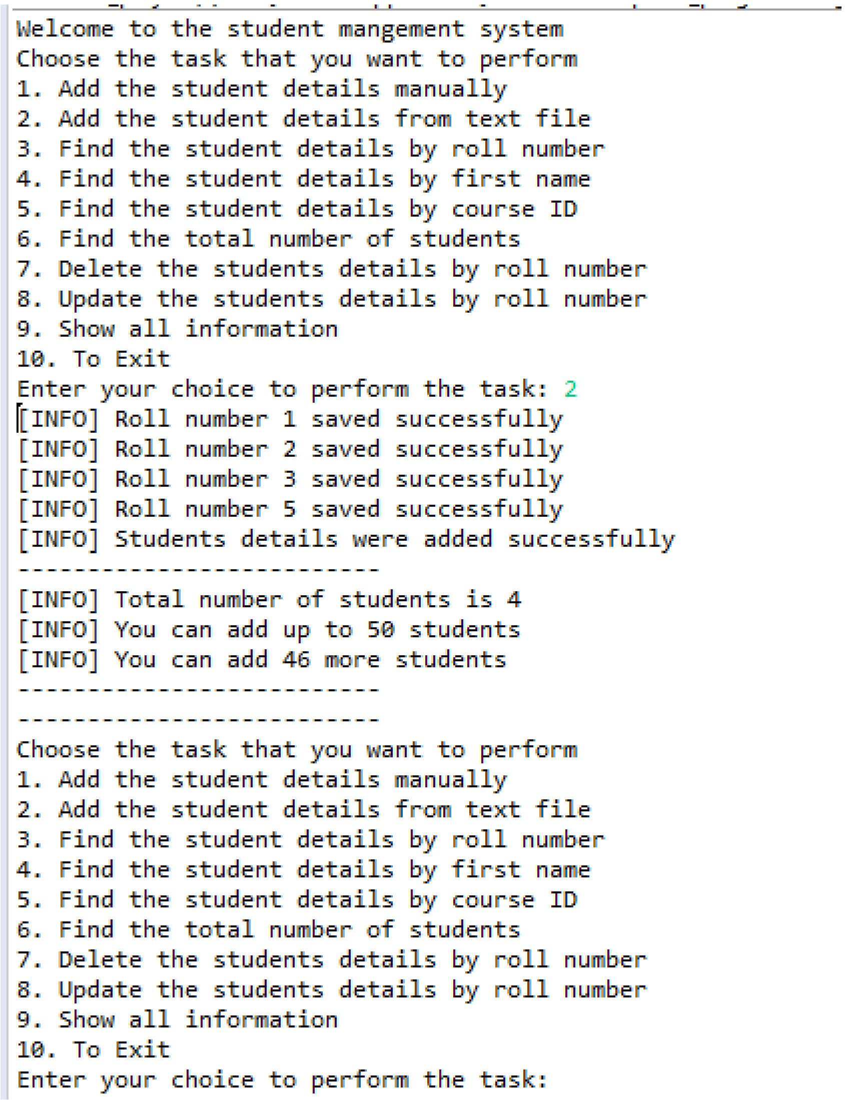
```
    while(choice <= 0 || choice > 5){
        DPRINTF("1.First name\n");
        DPRINTF("2.Last name\n");
        DPRINTF("3.Roll no.\n");
        DPRINTF("4.GPA\n");
        DPRINTF("5.Courses ID\n");
        scanf("%d",&choice);
    }
    switch(choice){
    case 1:
        DPRINTF("Enter new first name: ");
        gets(temp->fname);
        DPRINTF("[INFO] Updated successfully\n");
        break;
    case 2:
        DPRINTF("Enter new last name: ");
        gets(temp->lname);
        DPRINTF("[INFO] Updated successfully\n");
        break;
    case 3:
        DPRINTF("Enter new roll no.: ");
        scanf("%d",&rollNum);
        Queue_visit_all(&Queue_student , check_unique,rollNum,NULL,0,&temp,student_list);
        if(!student_list[0].rollNum){
            temp->rollNum=rollNum;
            DPRINTF("[INFO] Updated successfully\n");
        }else{
            DPRINTF("[ERROR] Roll number %d is already taken\n",temp->rollNum); }
        break;
    case 4:
        DPRINTF("Enter new GPA: ");
        scanf("%f",&temp->gpa);
        DPRINTF("[INFO] Updated successfully\n");
        break;
    case 5:
        DPRINTF("Enter new courses ID: \n");
        int i;
        for(i=0;i<5;i++){
```



```
}
void show_students(){

    Queue_visit_all(&Queue_student , check_unique,0,NULL,0,NULL,NULL);
}
```

2. Chapter two: Testing Program

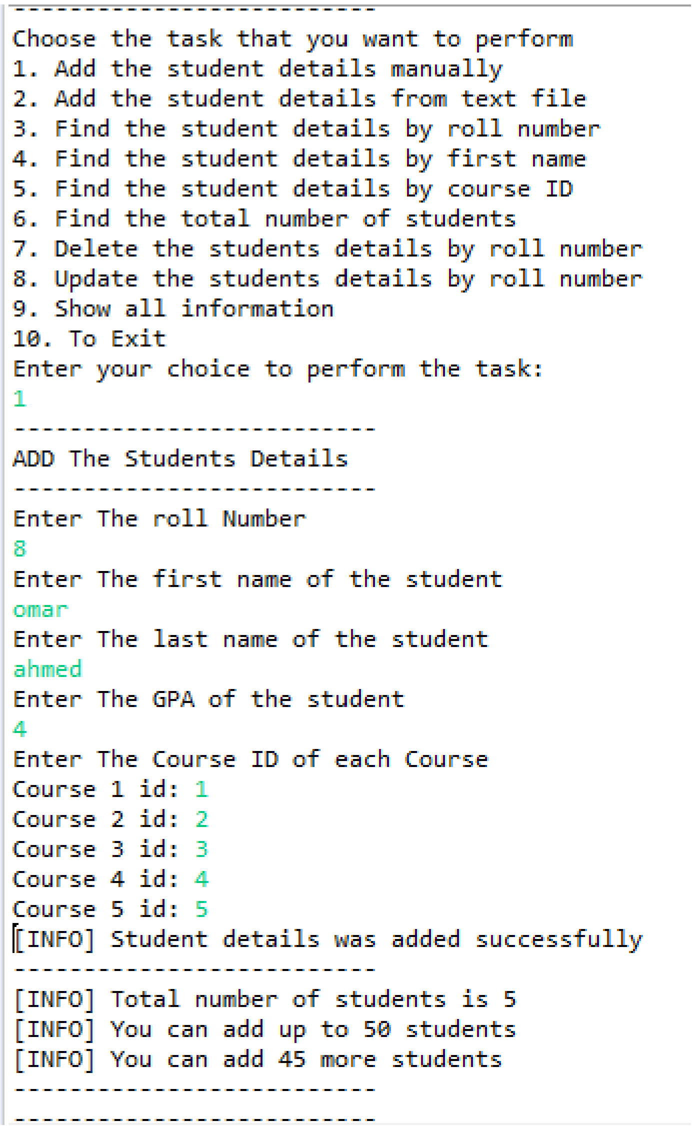
2.1Adding student from text file



```
Welcome to the student mangement system
Choose the task that you want to perform
1. Add the student details manually
2. Add the student details from text file
3. Find the student details by roll number
4. Find the student details by first name
5. Find the student details by course ID
6. Find the total number of students
7. Delete the students details by roll number
8. Update the students details by roll number
9. Show all information
10. To Exit
Enter your choice to perform the task: 2
[INFO] Roll number 1 saved successfully
[INFO] Roll number 2 saved successfully
[INFO] Roll number 3 saved successfully
[INFO] Roll number 5 saved successfully
[INFO] Students details were added successfully
--------------------------
[INFO] Total number of students is 4
[INFO] You can add up to 50 students
[INFO] You can add 46 more students
--------------------------
--------------------------
Choose the task that you want to perform
1. Add the student details manually
2. Add the student details from text file
3. Find the student details by roll number
4. Find the student details by first name
5. Find the student details by course ID
6. Find the total number of students
7. Delete the students details by roll number
8. Update the students details by roll number
9. Show all information
10. To Exit
Enter your choice to perform the task:
```

7Add_Student_Details_File رسم توضيحي

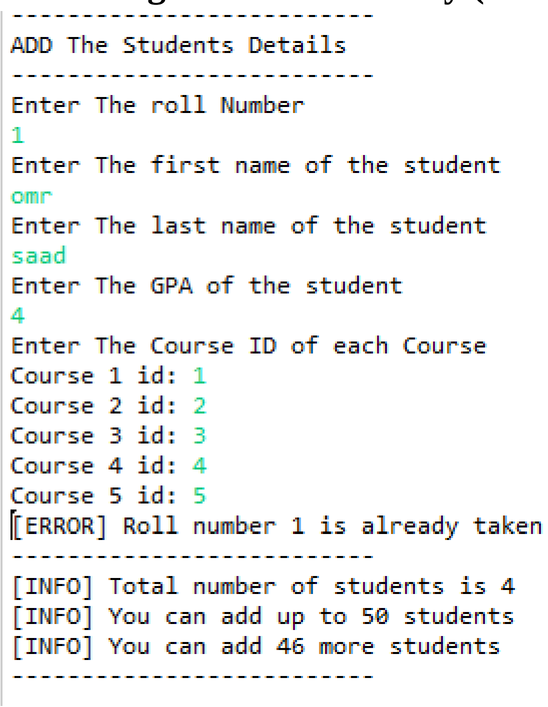
2.2Adding Student Manually:



```
--------------------------
Choose the task that you want to perform
1. Add the student details manually
2. Add the student details from text file
3. Find the student details by roll number
4. Find the student details by first name
5. Find the student details by course ID
6. Find the total number of students
7. Delete the students details by roll number
8. Update the students details by roll number
9. Show all information
10. To Exit
Enter your choice to perform the task:
1
--------------------------
ADD The Students Details
--------------------------
Enter The roll Number
8
Enter The first name of the student
omar
Enter The last name of the student
ahmed
Enter The GPA of the student
4
Enter The Course 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 was added successfully
--------------------------
[INFO] Total number of students is 5
[INFO] You can add up to 50 students
[INFO] You can add 45 more students
--------------------------
--------------------------
```

82.2Adding Student Manually: رسم توضيحي

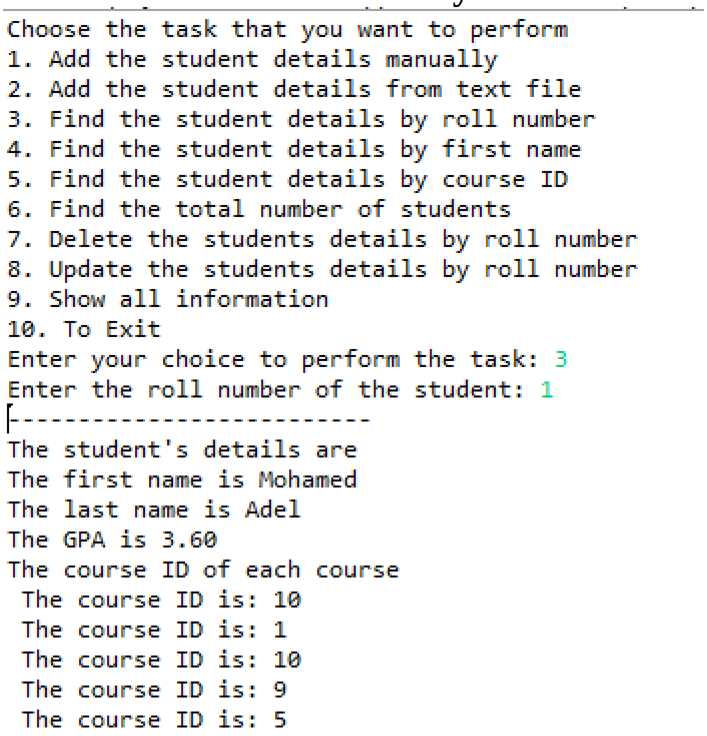
2.3Adding Student Manually (but roll number was repeated):



```
--------------------------
ADD The Students Details
--------------------------
Enter The roll Number
1
Enter The first name of the student
omr
Enter The last name of the student
saad
Enter The GPA of the student
4
Enter The Course ID of each Course
Course 1 id: 1
Course 2 id: 2
Course 3 id: 3
Course 4 id: 4
Course 5 id: 5
[ERROR] Roll number 1 is already taken
--------------------------
[INFO] Total number of students is 4
[INFO] You can add up to 50 students
[INFO] You can add 46 more students
--------------------------
```

92.2Adding Student Manually(roll number repated): رسم توضيحي

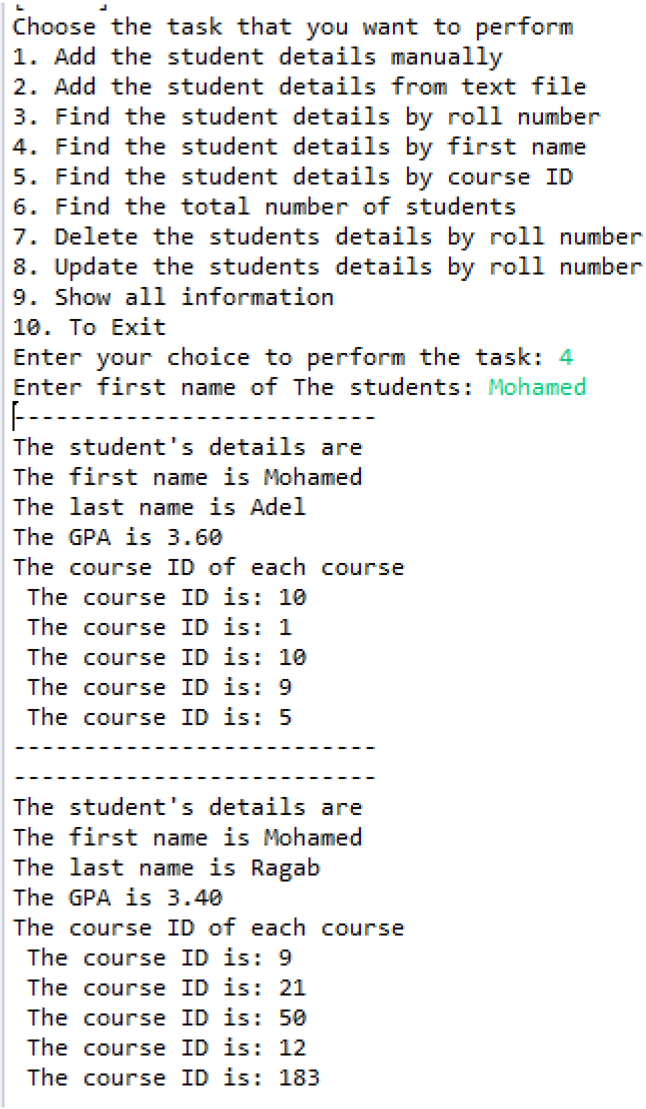
2.4Find the student details by roll number:



```
Choose the task that you want to perform
1. Add the student details manually
2. Add the student details from text file
3. Find the student details by roll number
4. Find the student details by first name
5. Find the student details by course ID
6. Find the total number of students
7. Delete the students details by roll number
8. Update the students details by roll number
9. Show all information
10. To Exit
Enter your choice to perform the task: 3
Enter the roll number of the student: 1
-------------------------
The student's details are
The first name is Mohamed
The last name is Adel
The GPA is 3.60
The course ID of each course
 The course ID is: 10
 The course ID is: 1
 The course ID is: 10
 The course ID is: 9
 The course ID is: 5
```

10Find_Student_Details_(BY_ROLLNUMBER) رسم توضيحي

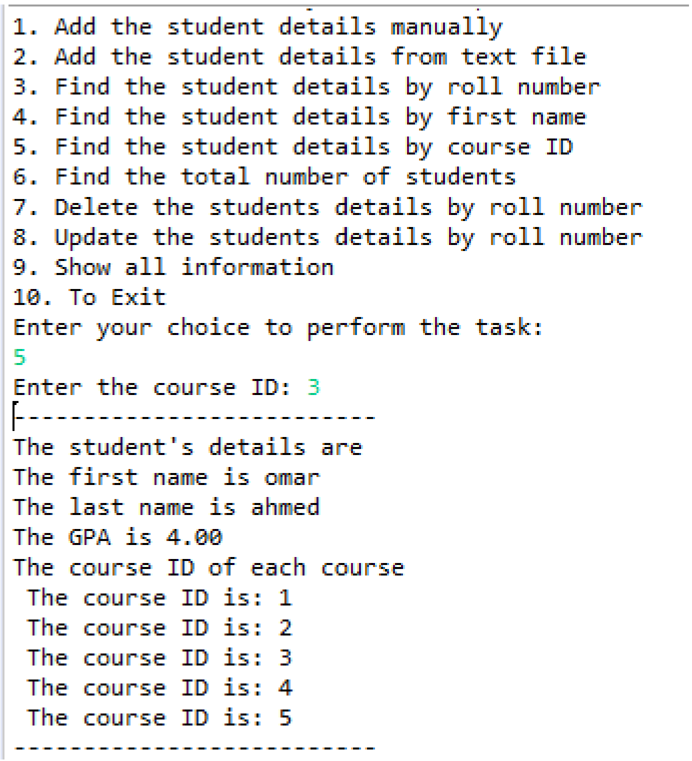
2.5Find the student by first name:



```
Choose the task that you want to perform
1. Add the student details manually
2. Add the student details from text file
3. Find the student details by roll number
4. Find the student details by first name
5. Find the student details by course ID
6. Find the total number of students
7. Delete the students details by roll number
8. Update the students details by roll number
9. Show all information
10. To Exit
Enter your choice to perform the task: 4
Enter first name of The students: Mohamed
--------------------------
The student's details are
The first name is Mohamed
The last name is Adel
The GPA is 3.60
The course ID of each course
 The course ID is: 10
 The course ID is: 1
 The course ID is: 10
 The course ID is: 9
 The course ID is: 5
--------------------------
--------------------------
The student's details are
The first name is Mohamed
The last name is Ragab
The GPA is 3.40
The course ID of each course
 The course ID is: 9
 The course ID is: 21
 The course ID is: 50
 The course ID is: 12
 The course ID is: 183
```

11Find_Student_Details_(BY_FNAME) رسم توضيحي

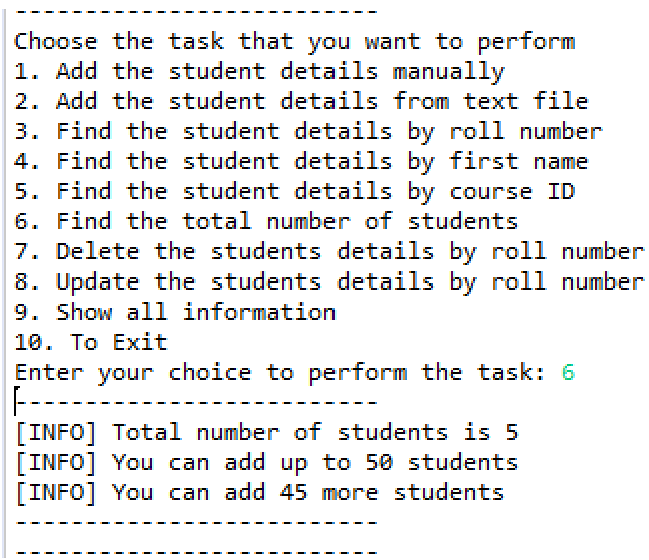
2.6Find the student by course ID:



```
1. Add the student details manually
2. Add the student details from text file
3. Find the student details by roll number
4. Find the student details by first name
5. Find the student details by course ID
6. Find the total number of students
7. Delete the students details by roll number
8. Update the students details by roll number
9. Show all information
10. To Exit
Enter your choice to perform the task:
5
Enter the course ID: 3
-------------------------
The student's details are
The first name is omar
The last name is ahmed
The GPA is 4.00
The course ID of each course
 The course ID is: 1
 The course ID is: 2
 The course ID is: 3
 The course ID is: 4
 The course ID is: 5
-------------------------
```

12Find_Student_Details_(BY_COURSE_ID) رسم توضيحي

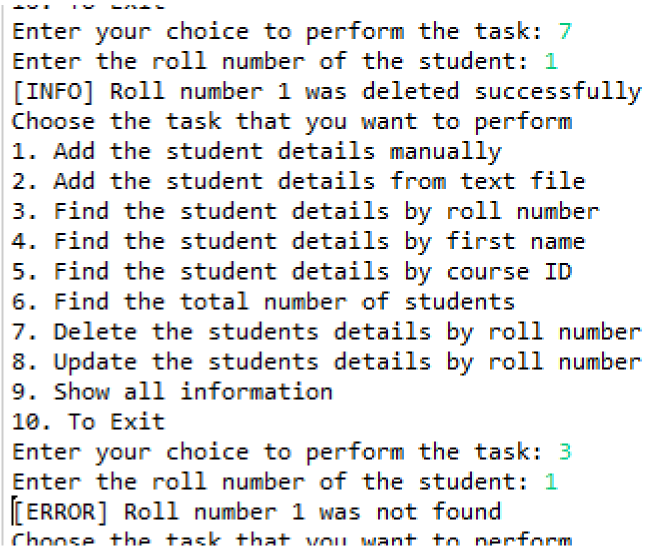
2.7Find the total number of students:



```
-------------------------
Choose the task that you want to perform
1. Add the student details manually
2. Add the student details from text file
3. Find the student details by roll number
4. Find the student details by first name
5. Find the student details by course ID
6. Find the total number of students
7. Delete the students details by roll number
8. Update the students details by roll number
9. Show all information
10. To Exit
Enter your choice to perform the task: 6
-------------------------
[INFO] Total number of students is 5
[INFO] You can add up to 50 students
[INFO] You can add 45 more students
-------------------------
-------------------------
```

رسم توضيحي 13Find_Total_Number_Students

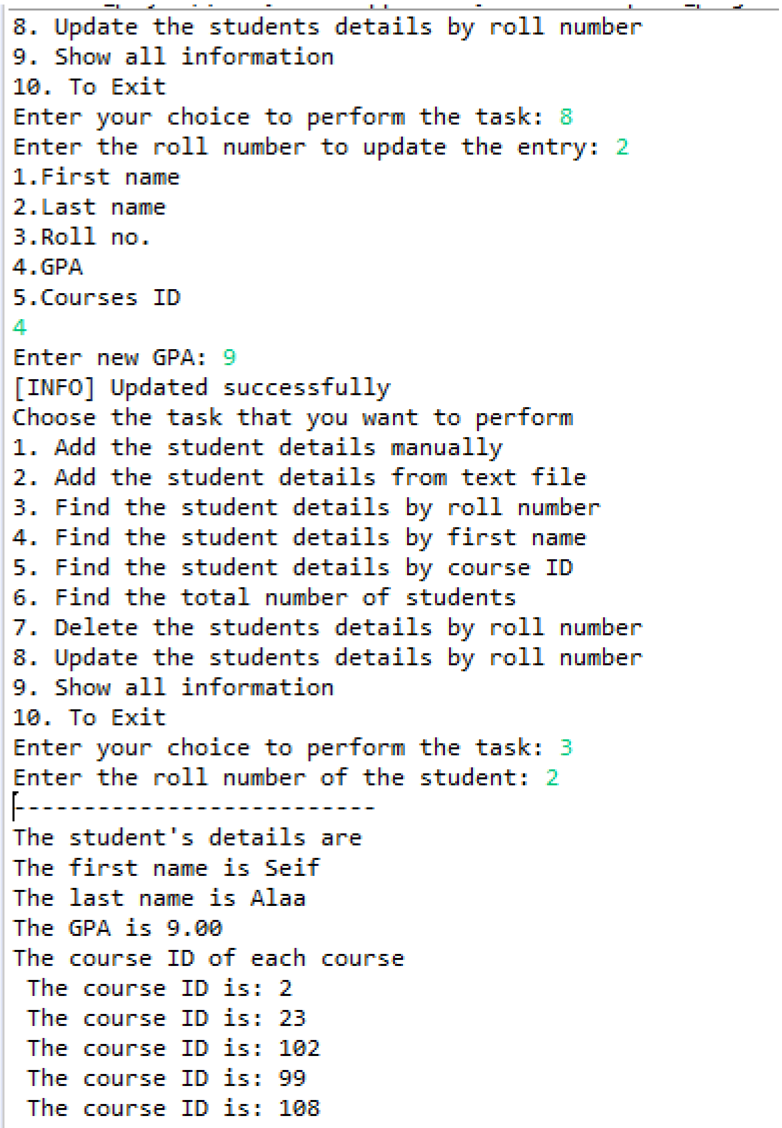
2.8Delete a student:



```
10. To Exit
Enter your choice to perform the task: 7
Enter the roll number of the student: 1
[INFO] Roll number 1 was deleted successfully
Choose the task that you want to perform
1. Add the student details manually
2. Add the student details from text file
3. Find the student details by roll number
4. Find the student details by first name
5. Find the student details by course ID
6. Find the total number of students
7. Delete the students details by roll number
8. Update the students details by roll number
9. Show all information
10. To Exit
Enter your choice to perform the task: 3
Enter the roll number of the student: 1
[ERROR] Roll number 1 was not found
Choose the task that you want to perform
```

رسم توضيحي 14elete_Student(BY_ROLLNUMBER)

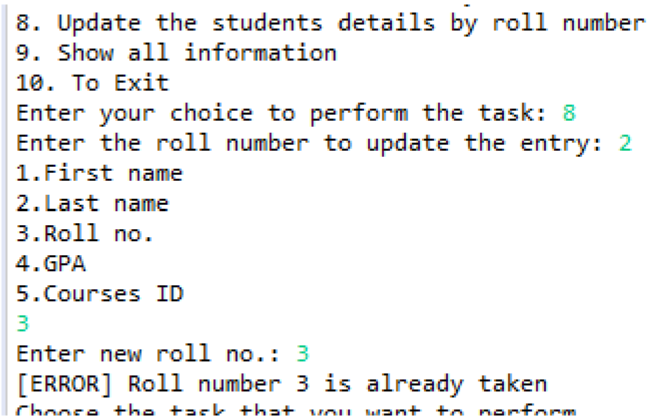
2.9Update student details:



```
8. Update the students details by roll number
9. Show all information
10. To Exit
Enter your choice to perform the task: 8
Enter the roll number to update the entry: 2
1.First name
2.Last name
3.Roll no.
4.GPA
5.Courses ID
4
Enter new GPA: 9
[INFO] Updated successfully
Choose the task that you want to perform
1. Add the student details manually
2. Add the student details from text file
3. Find the student details by roll number
4. Find the student details by first name
5. Find the student details by course ID
6. Find the total number of students
7. Delete the students details by roll number
8. Update the students details by roll number
9. Show all information
10. To Exit
Enter your choice to perform the task: 3
Enter the roll number of the student: 2
-------------------------
The student's details are
The first name is Seif
The last name is Alaa
The GPA is 9.00
The course ID of each course
 The course ID is: 2
 The course ID is: 23
 The course ID is: 102
 The course ID is: 99
 The course ID is: 108
```

رسم توضيحي 15Update_Student(BY_ROLLNUMBER)

2.10Update student details (roll number repeated):



```
8. Update the students details by roll number
9. Show all information
10. To Exit
Enter your choice to perform the task: 8
Enter the roll number to update the entry: 2
1.First name
2.Last name
3.Roll no.
4.GPA
5.Courses ID
3
Enter new roll no.: 3
[ERROR] Roll number 3 is already taken
Choose the task that you want to perform
```

رسم توضيحي 16Update_Student(!ROLLNUMBER)TAKEN