Nov 2021

First Term (Final Project 2)

Student Management System

Eng. Mazen Talaat
My profile: https://bit.ly/3DUmnyz

Mastering embedded system online diploma by Eng. Keroles Shenouda
www.learn-in-depth.com

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 student.

Approach

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

1. Add student details from 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 number of 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.

Idea

The software will consist of 4 files main.c, SYS_API.c, SYS_API.h and, Students.txt.

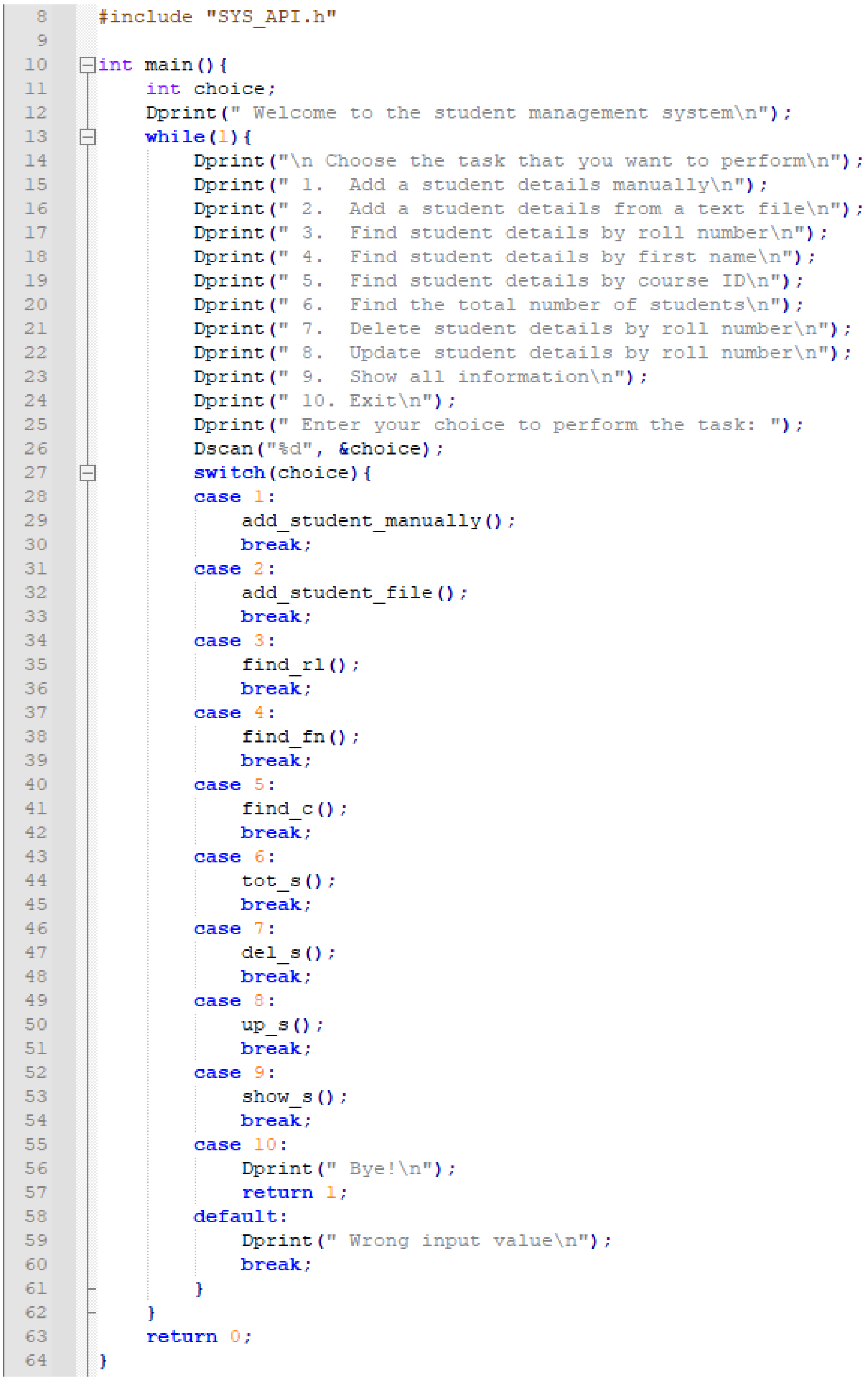
The main.c will contain the interface and calling the function, SYS_API.c will contain the global variables and the body of the functions, SYS_API.h will contain the definitions and the prototypes, Students.txt will contain the data.

1. We will have array of struct of 50 elements which is the max number of students.
2. Each struct contains the student details.
3. We will have a global index which refers to the number of students.
4. The index is initialized with zero and can go up to 50.

main.c

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

- Infinite while loop until the choice = 10.
- Dprint and Dscan function(a macro to prinf and fflush to fix eclipse console bug).
- Calling the program functions through switch cases.

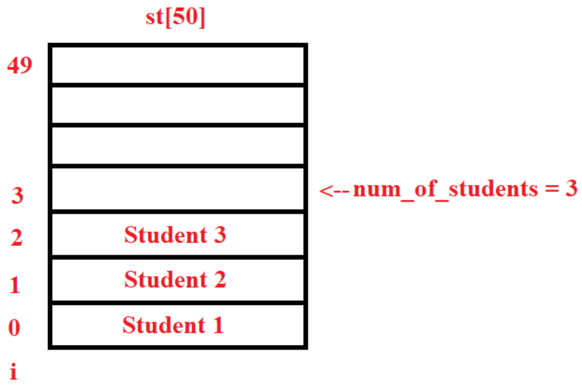


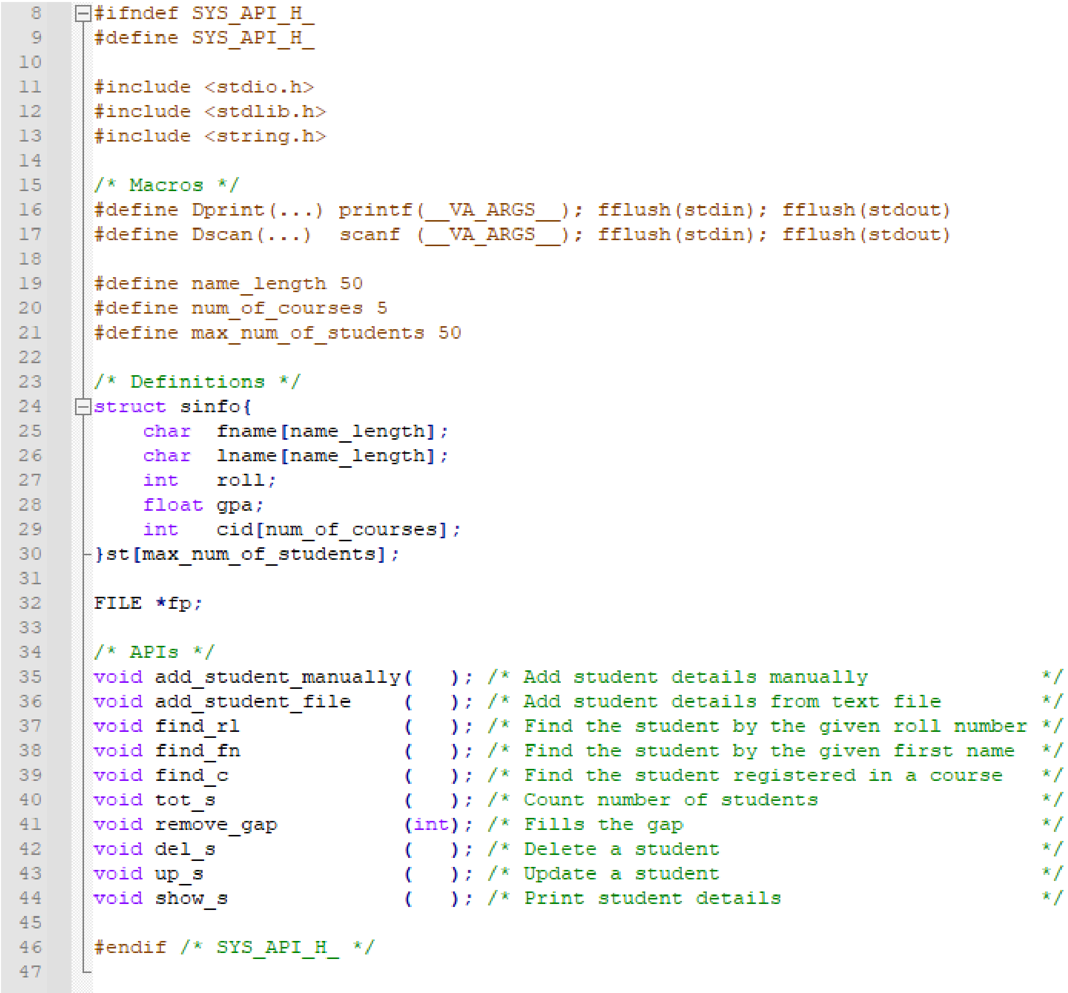
```c
#include "SYS_API.h"

int main(){
    int choice;
    Dprint(" Welcome to the student management system\n");
    while(1){
        Dprint("\n Choose the task that you want to perform\n");
        Dprint(" 1.  Add a student details manually\n");
        Dprint(" 2.  Add a student details from a text file\n");
        Dprint(" 3.  Find student details by roll number\n");
        Dprint(" 4.  Find student details by first name\n");
        Dprint(" 5.  Find student details by course ID\n");
        Dprint(" 6.  Find the total number of students\n");
        Dprint(" 7.  Delete student details by roll number\n");
        Dprint(" 8.  Update student details by roll number\n");
        Dprint(" 9.  Show all information\n");
        Dprint(" 10. Exit\n");
        Dprint(" Enter your choice to perform the task: ");
        Dscan("%d", &choice);
        switch(choice){
        case 1:
            add_student_manually();
            break;
        case 2:
            add_student_file();
            break;
        case 3:
            find_rl();
            break;
        case 4:
            find_fn();
            break;
        case 5:
            find_c();
            break;
        case 6:
            tot_s();
            break;
        case 7:
            del_s();
            break;
        case 8:
            up_s();
            break;
        case 9:
            show_s();
            break;
        case 10:
            Dprint(" Bye!\n");
            return 1;
        default:
            Dprint(" Wrong input value\n");
            break;
        }
    }
    return 0;
}
```

SYS_API.h

The SYS_API.h consist of:

- Macros to make things easier.
- Struct sinfo which contains the details of a student.
- st[max_num_of_students] which is an array of 50 of struct sinfo.
- FILE *fp which is a pointer to a typedef FILE to manage reading a file.
- Functions prototypes with their usage.





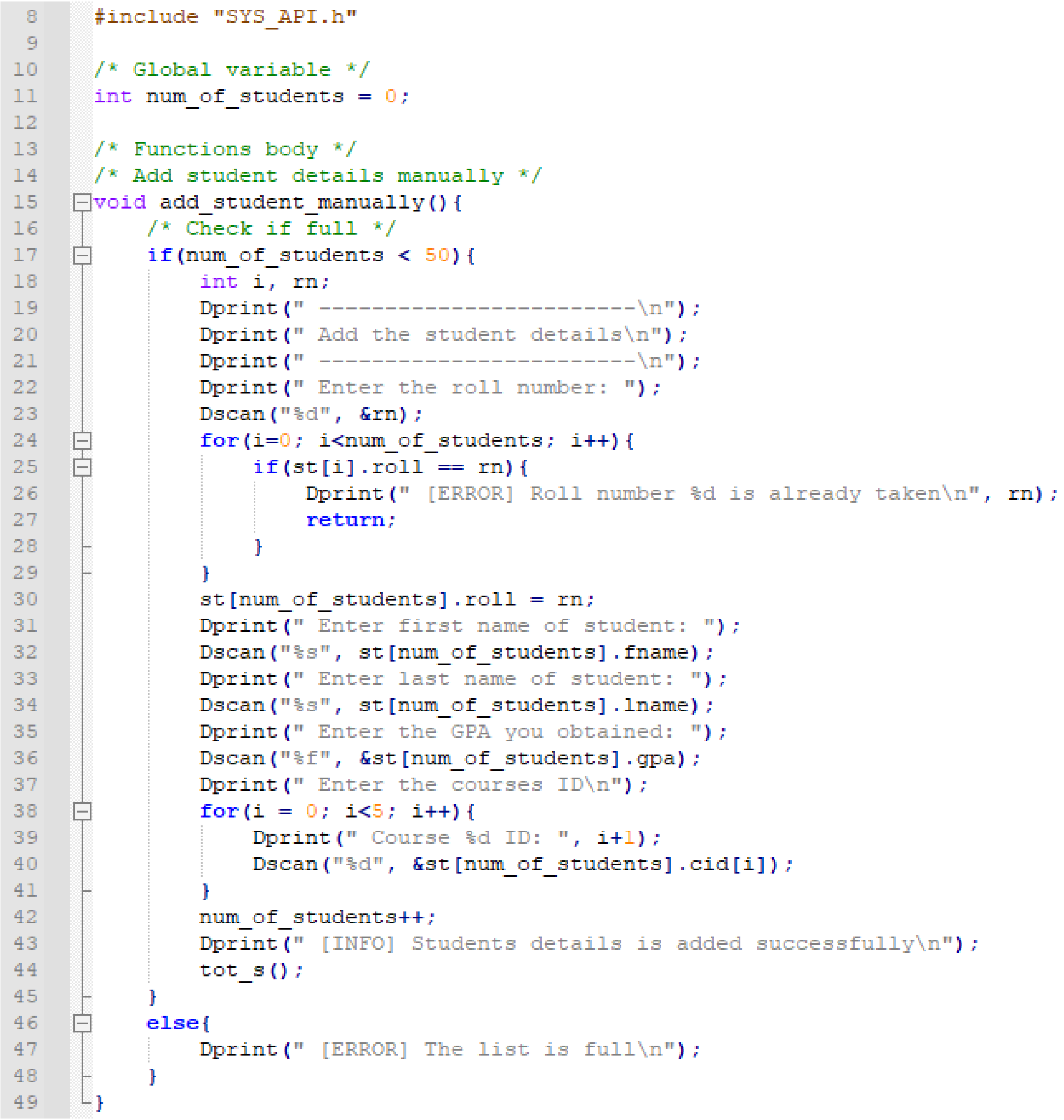
```c
#ifndef SYS_API_H_
#define SYS_API_H_

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

/* Macros */
#define Dprint(...) printf(__VA_ARGS__); fflush(stdin); fflush(stdout)
#define Dscan(...)  scanf (__VA_ARGS__); fflush(stdin); fflush(stdout)

#define name_length 50
#define num_of_courses 5
#define max_num_of_students 50

/* Definitions */
struct sinfo{
    char  fname[name_length];
    char  lname[name_length];
    int   roll;
    float gpa;
    int   cid[num_of_courses];
}st[max_num_of_students];

FILE *fp;

/* APIs */
void add_student_manually(   ); /* Add student details manually              */
void add_student_file    (   ); /* Add student details from text file        */
void find_rl             (   ); /* Find the student by the given roll number */
void find_fn             (   ); /* Find the student by the given first name  */
void find_c              (   ); /* Find the student registered in a course   */
void tot_s               (   ); /* Count number of students                  */
void remove_gap          (int); /* Fills the gap                             */
void del_s               (   ); /* Delete a student                          */
void up_s                (   ); /* Update a student                          */
void show_s              (   ); /* Print student details                     */

#endif /* SYS_API_H_ */
```

SYS_API.c

The SYS_API.h consist of:

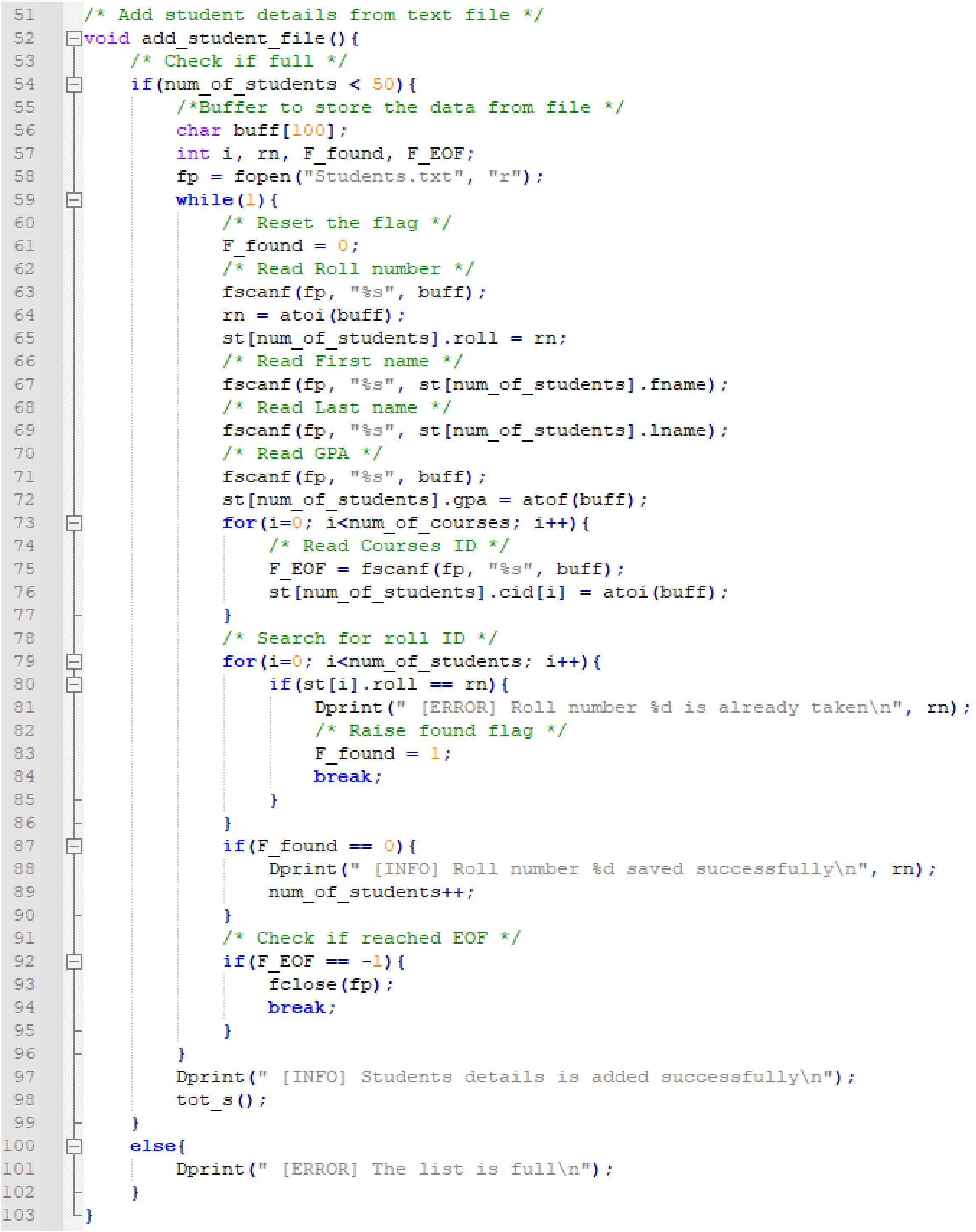
- Global variable num_of_students which is the index of the array st[] to track the data.
- Functions body.

1. Add student details manually
    - The function will start by checking the list if it is full.
    - Takes from user the roll number and checks if it is duplicated.
        a. If yes, it will print an error.
        b. If no, it will print success and add the student details to the location where the index is pointing to.
    - The number of students is then incremented by 1.
    - Printing the total, max and remaining number of students through the function tot_s().



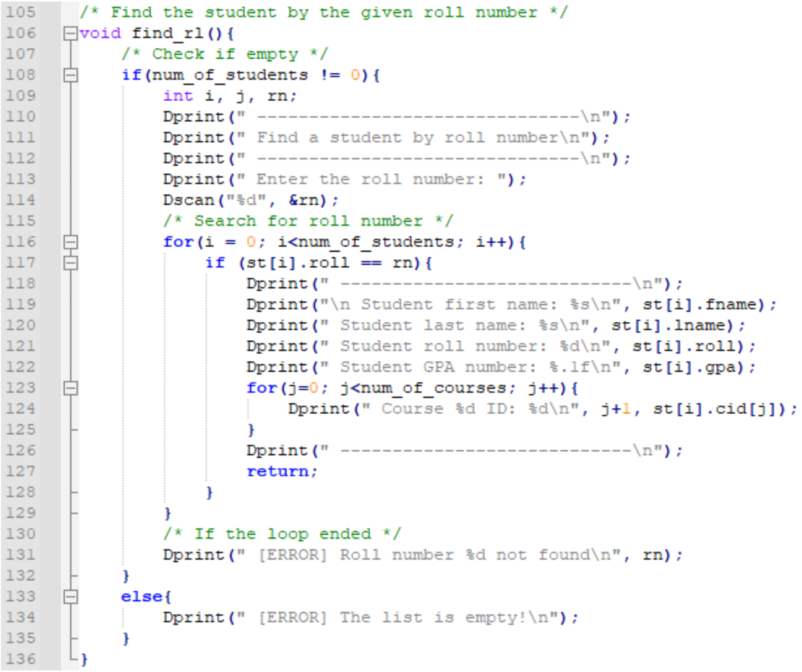
```c
#include "SYS_API.h"

/* Global variable */
int num_of_students = 0;

/* Functions body */
/* Add student details manually */
void add_student_manually(){
    /* Check if full */
    if(num_of_students < 50){
        int i, rn;
        Dprint(" ------------------------\n");
        Dprint(" Add the student details\n");
        Dprint(" ------------------------\n");
        Dprint(" Enter the roll number: ");
        Dscan("%d", &rn);
        for(i=0; i<num_of_students; i++){
            if(st[i].roll == rn){
                Dprint(" [ERROR] Roll number %d is already taken\n", rn);
                return;
            }
        }
        st[num_of_students].roll = rn;
        Dprint(" Enter first name of student: ");
        Dscan("%s", st[num_of_students].fname);
        Dprint(" Enter last name of student: ");
        Dscan("%s", st[num_of_students].lname);
        Dprint(" Enter the GPA you obtained: ");
        Dscan("%f", &st[num_of_students].gpa);
        Dprint(" Enter the courses ID\n");
        for(i = 0; i<5; i++){
            Dprint(" Course %d ID: ", i+1);
            Dscan("%d", &st[num_of_students].cid[i]);
        }
        num_of_students++;
        Dprint(" [INFO] Students details is added successfully\n");
        tot_s();
    }
    else{
        Dprint(" [ERROR] The list is full\n");
    }
}
```

2. Add student details from file
    - The function will start by checking the list if it is full.
    - Reads from the file the details, saves them, and checks if the roll number is duplicated.
        a. If yes, it will print that the roll number is already taken and continue to the next line in the file.
        b. If no, it will print success and increment the number of students by 1 (index).
    - If it reached the EOF(end of file) , a flag will be raised.
    - Printing the total, max and remaining number of students through the function tot_s().



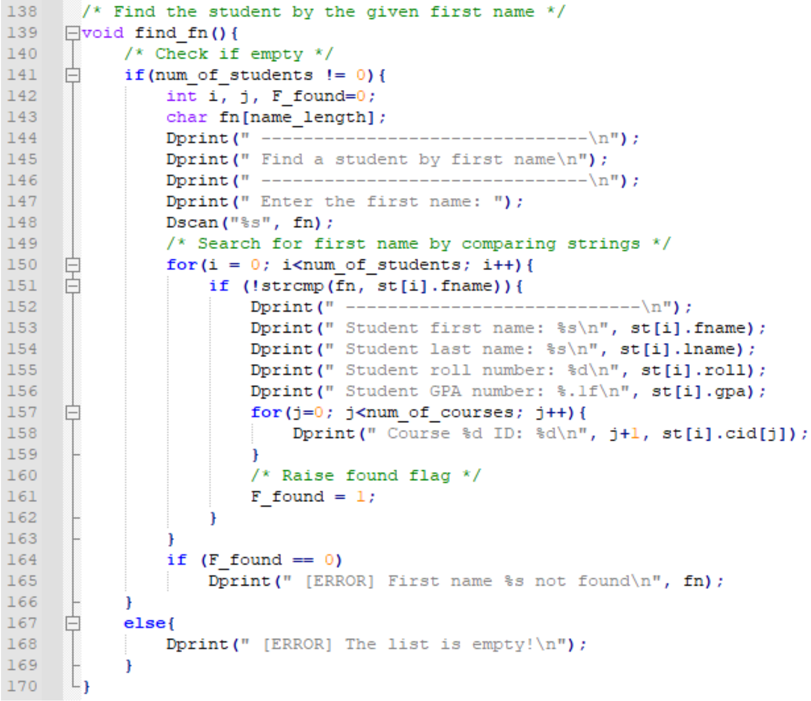
```c
/* Add student details from text file */
void add_student_file(){
    /* Check if full */
    if(num_of_students < 50){
        /*Buffer to store the data from file */
        char buff[100];
        int i, rn, F_found, F_EOF;
        fp = fopen("Students.txt", "r");
        while(1){
            /* Reset the flag */
            F_found = 0;
            /* Read Roll number */
            fscanf(fp, "%s", buff);
            rn = atoi(buff);
            st[num_of_students].roll = rn;
            /* Read First name */
            fscanf(fp, "%s", st[num_of_students].fname);
            /* Read Last name */
            fscanf(fp, "%s", st[num_of_students].lname);
            /* Read GPA */
            fscanf(fp, "%s", buff);
            st[num_of_students].gpa = atof(buff);
            for(i=0; i<num_of_courses; i++){
                /* Read Courses ID */
                F_EOF = fscanf(fp, "%s", buff);
                st[num_of_students].cid[i] = atoi(buff);
            }
            /* Search for roll ID */
            for(i=0; i<num_of_students; i++){
                if(st[i].roll == rn){
                    Dprint(" [ERROR] Roll number %d is already taken\n", rn);
                    /* Raise found flag */
                    F_found = 1;
                    break;
                }
            }
            if(F_found == 0){
                Dprint(" [INFO] Roll number %d saved successfully\n", rn);
                num_of_students++;
            }
            /* Check if reached EOF */
            if(F_EOF == -1){
                fclose(fp);
                break;
            }
        }
        Dprint(" [INFO] Students details is added successfully\n");
        tot_s();
    }
    else{
        Dprint(" [ERROR] The list is full\n");
    }
}
```

3. Find the student by the given roll number
    - The function will start by checking the list if it is empty.
    - Searches for the roll number received from user.
        a. If found, it will print that the details and return.
        b. If not, it will print an error.



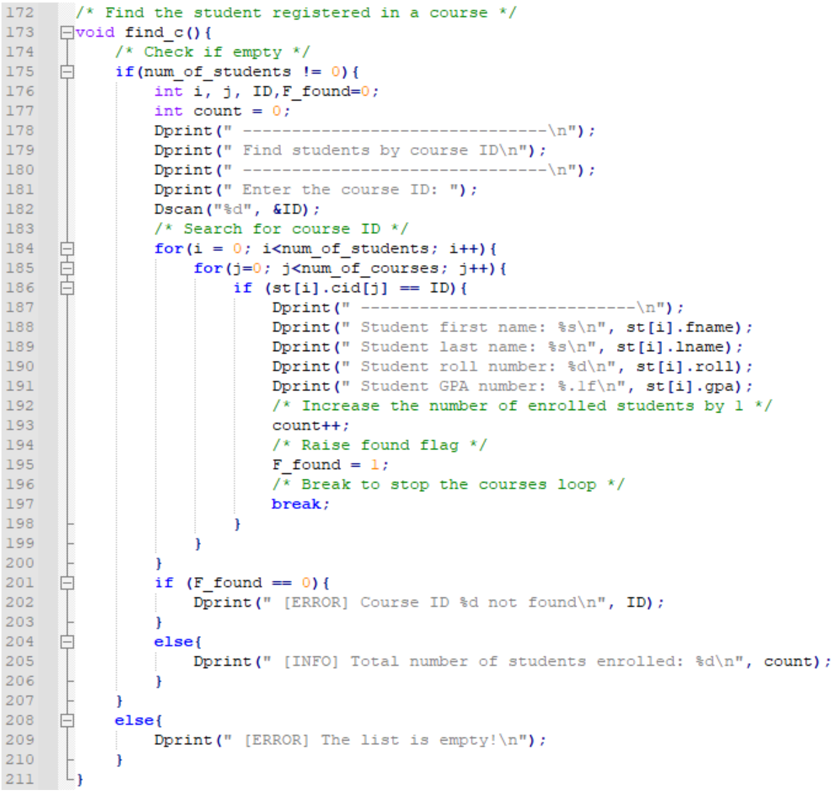
```c
/* Find the student by the given roll number */
void find_rl(){
    /* Check if empty */
    if(num_of_students != 0){
        int i, j, rn;
        Dprint(" -------------------------------\n");
        Dprint(" Find a student by roll number\n");
        Dprint(" -------------------------------\n");
        Dprint(" Enter the roll number: ");
        Dscan("%d", &rn);
        /* Search for roll number */
        for(i = 0; i<num_of_students; i++){
            if (st[i].roll == rn){
                Dprint(" ----------------------------\n");
                Dprint("\n Student first name: %s\n", st[i].fname);
                Dprint(" Student last name: %s\n", st[i].lname);
                Dprint(" Student roll number: %d\n", st[i].roll);
                Dprint(" Student GPA number: %.1f\n", st[i].gpa);
                for(j=0; j<num_of_courses; j++){
                    Dprint(" Course %d ID: %d\n", j+1, st[i].cid[j]);
                }
                Dprint(" ----------------------------\n");
                return;
            }
        }
        /* If the loop ended */
        Dprint(" [ERROR] Roll number %d not found\n", rn);
    }
    else{
        Dprint(" [ERROR] The list is empty!\n");
    }
}
```

4. Find the student by the given first name
    - The function will start by checking the list if it is empty.
    - Searches for the first name received from user.
        a. If found, it will print that the details and raise a flag.
        b. If not, it will print an error.
    - It will continue the search until reaching the end of the list.



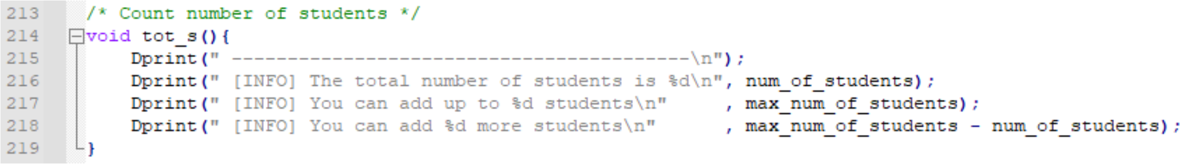
```c
/* Find the student by the given first name */
void find_fn(){
    /* Check if empty */
    if(num_of_students != 0){
        int i, j, F_found=0;
        char fn[name_length];
        Dprint(" -------------------------------\n");
        Dprint(" Find a student by first name\n");
        Dprint(" -------------------------------\n");
        Dprint(" Enter the first name: ");
        Dscan("%s", fn);
        /* Search for first name by comparing strings */
        for(i = 0; i<num_of_students; i++){
            if (!strcmp(fn, st[i].fname)){
                Dprint(" ----------------------------\n");
                Dprint(" Student first name: %s\n", st[i].fname);
                Dprint(" Student last name: %s\n", st[i].lname);
                Dprint(" Student roll number: %d\n", st[i].roll);
                Dprint(" Student GPA number: %.1f\n", st[i].gpa);
                for(j=0; j<num_of_courses; j++){
                    Dprint(" Course %d ID: %d\n", j+1, st[i].cid[j]);
                }
                /* Raise found flag */
                F_found = 1;
            }
        }
        if (F_found == 0)
            Dprint(" [ERROR] First name %s not found\n", fn);
    }
    else{
        Dprint(" [ERROR] The list is empty!\n");
    }
}
```

5. Find the student registered in a course
    - The function will start by checking the list if it is empty.
    - Searches for the course ID received from user.
        a. If found, it will print that the details and raise a flag.
        b. If not, it will print an error.
    - It will continue the search until reaching the end of the list and prints the total number of students enrolled.



```c
/* Find the student registered in a course */
void find_c(){
    /* Check if empty */
    if(num_of_students != 0){
        int i, j, ID,F_found=0;
        int count = 0;
        Dprint(" -------------------------------\n");
        Dprint(" Find students by course ID\n");
        Dprint(" -------------------------------\n");
        Dprint(" Enter the course ID: ");
        Dscan("%d", &ID);
        /* Search for course ID */
        for(i = 0; i<num_of_students; i++){
            for(j=0; j<num_of_courses; j++){
                if (st[i].cid[j] == ID){
                    Dprint(" ----------------------------\n");
                    Dprint(" Student first name: %s\n", st[i].fname);
                    Dprint(" Student last name: %s\n", st[i].lname);
                    Dprint(" Student roll number: %d\n", st[i].roll);
                    Dprint(" Student GPA number: %.1f\n", st[i].gpa);
                    /* Increase the number of enrolled students by 1 */
                    count++;
                    /* Raise found flag */
                    F_found = 1;
                    /* Break to stop the courses loop */
                    break;
                }
            }
        }
        if (F_found == 0){
            Dprint(" [ERROR] Course ID %d not found\n", ID);
        }
        else{
            Dprint(" [INFO] Total number of students enrolled: %d\n", count);
        }
    }
    else{
        Dprint(" [ERROR] The list is empty!\n");
    }
}
```

6. Count number of students
    - The function will print the total number of students by using the index_number_of_students.
    - Print the max number by using the macro max_number_of_students.



```c
/* Count number of students */
void tot_s(){
    Dprint(" ----------------------------------------\n");
    Dprint(" [INFO] The total number of students is %d\n", num_of_students);
    Dprint(" [INFO] You can add up to %d students\n"     , max_num_of_students);
    Dprint(" [INFO] You can add %d more students\n"      , max_num_of_students - num_of_students);
}
```

7. Delete a student

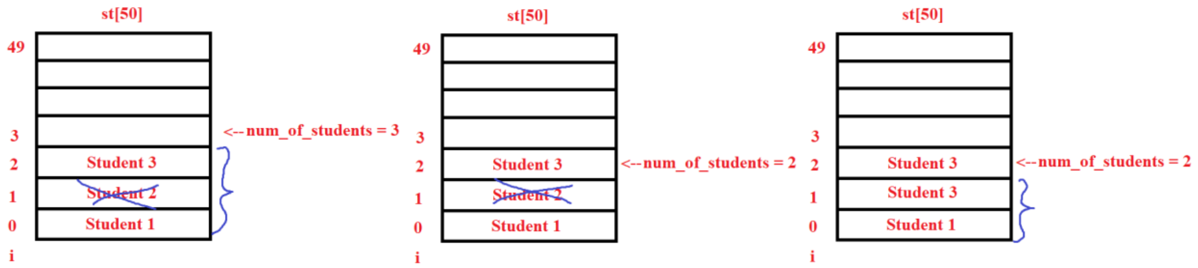
It consists of two functions: remove_gap and del_s.

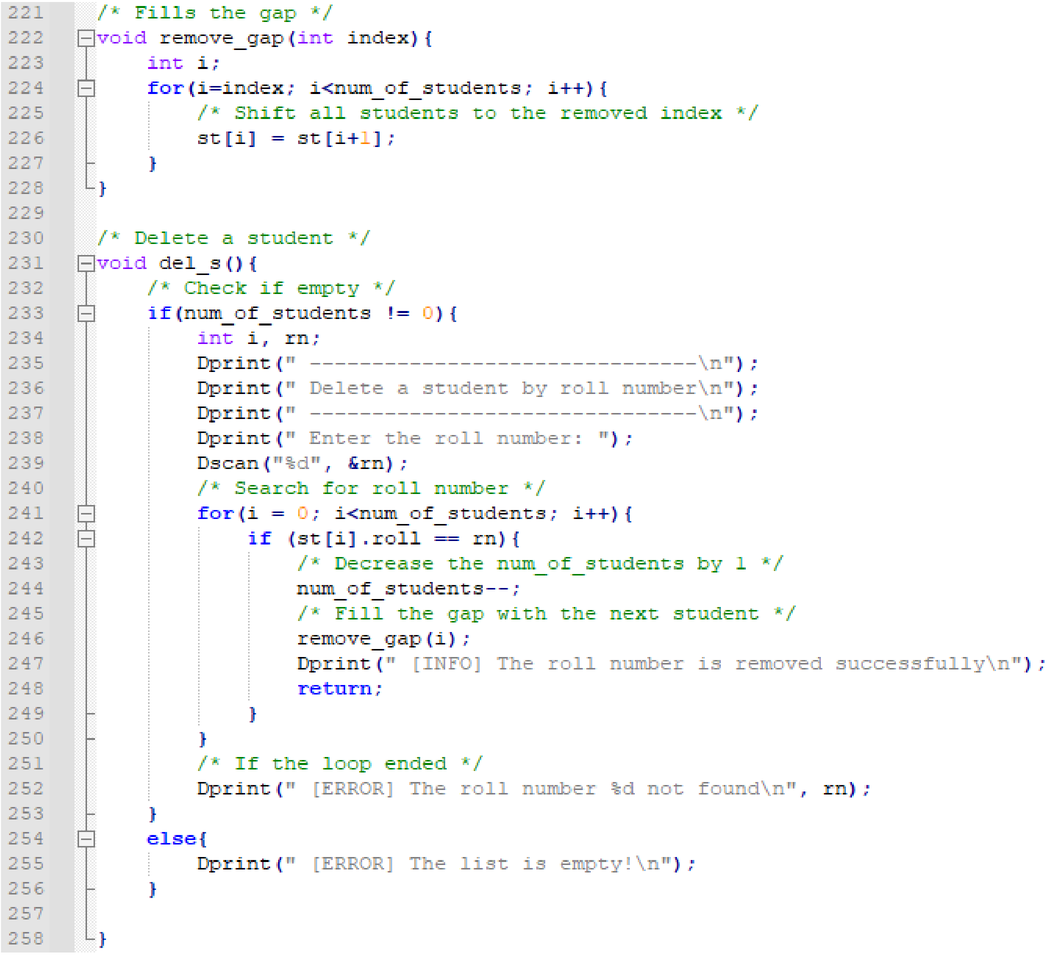
del_s:

- The function will start by checking the list if it is empty.
- Searches for the roll number received from user.
  a. If found, it will decrease the number of students, calls the function remove_gap and return.
  b. If not, it will print an error.

remove_gap:

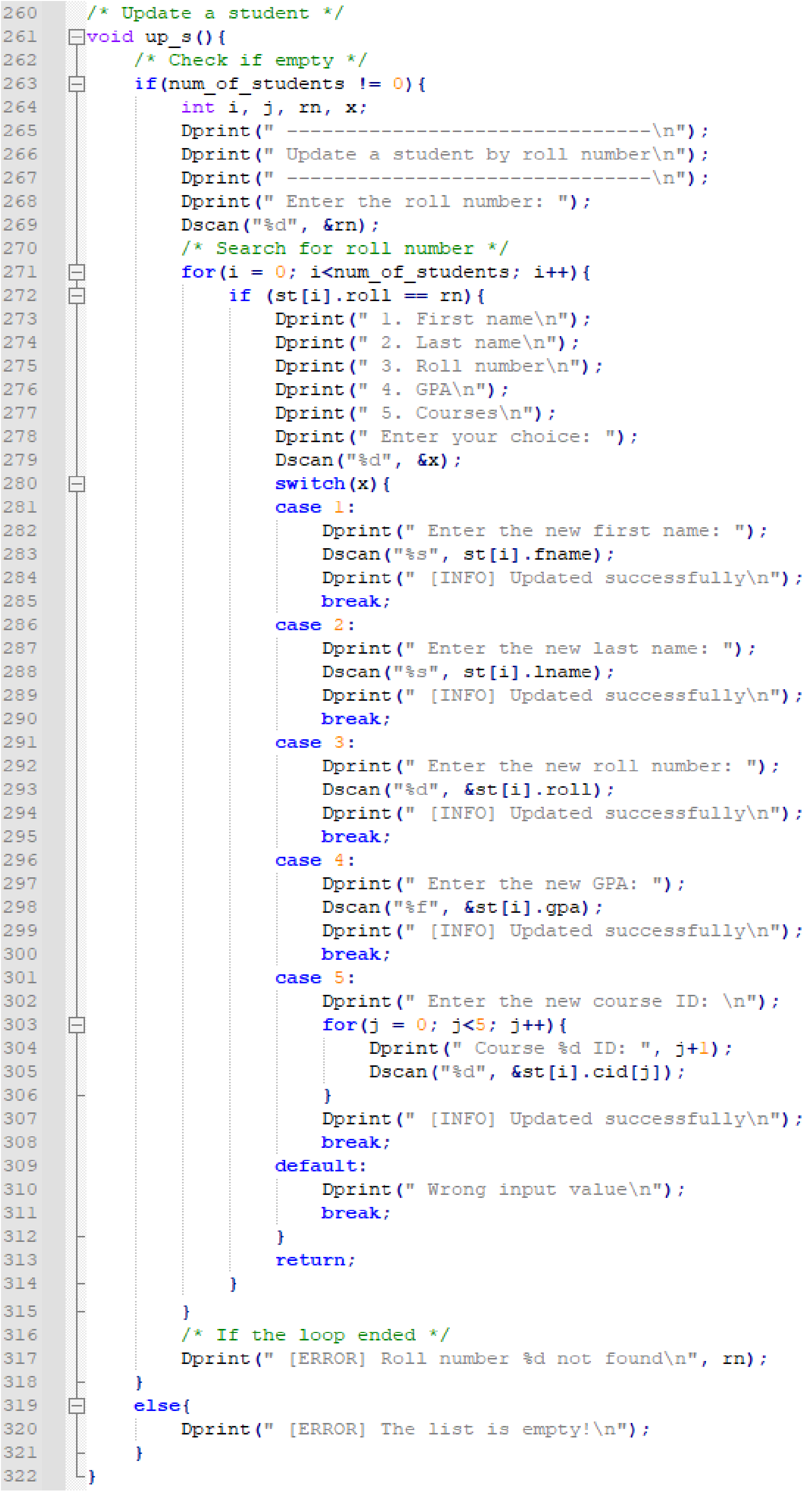
- The function will shift the elements starting from the index of the removed student to the start.





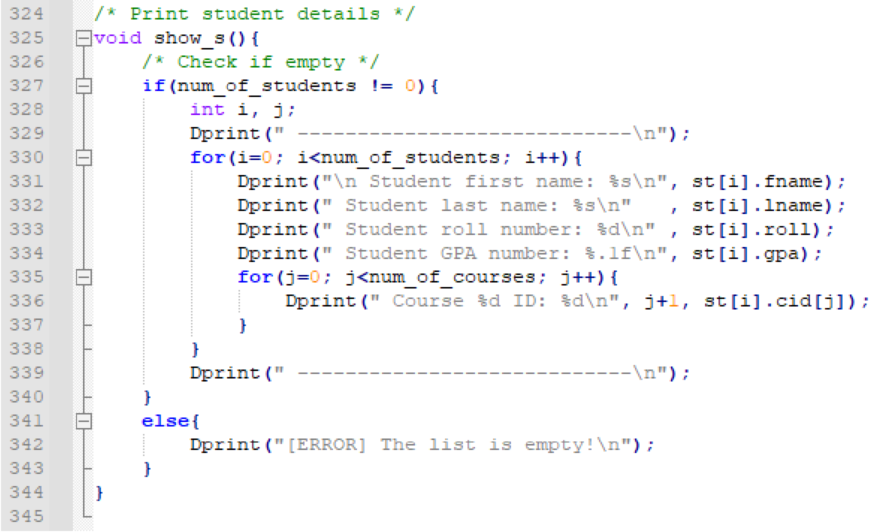
```c
/* Fills the gap */
void remove_gap(int index){
    int i;
    for(i=index; i<num_of_students; i++){
        /* Shift all students to the removed index */
        st[i] = st[i+1];
    }
}

/* Delete a student */
void del_s(){
    /* Check if empty */
    if(num_of_students != 0){
        int i, rn;
        Dprint(" -------------------------------\n");
        Dprint(" Delete a student by roll number\n");
        Dprint(" -------------------------------\n");
        Dprint(" Enter the roll number: ");
        Dscan("%d", &rn);
        /* Search for roll number */
        for(i = 0; i<num_of_students; i++){
            if (st[i].roll == rn){
                /* Decrease the num_of_students by 1 */
                num_of_students--;
                /* Fill the gap with the next student */
                remove_gap(i);
                Dprint(" [INFO] The roll number is removed successfully\n");
                return;
            }
        }
        /* If the loop ended */
        Dprint(" [ERROR] The roll number %d not found\n", rn);
    }
    else{
        Dprint(" [ERROR] The list is empty!\n");
    }

}
```

8. Update a student
   - The function will start by checking the list if it is empty.
   - Searches for the roll number received from user.
     a. If found, it will ask the user to choose what to update and print success if so.
     b. If not, it will print an error.



```c
/* Update a student */
void up_s(){
    /* Check if empty */
    if(num_of_students != 0){
        int i, j, rn, x;
        Dprint(" ------------------------------\n");
        Dprint(" Update a student by roll number\n");
        Dprint(" ------------------------------\n");
        Dprint(" Enter the roll number: ");
        Dscan("%d", &rn);
        /* Search for roll number */
        for(i = 0; i<num_of_students; i++){
            if (st[i].roll == rn){
                Dprint(" 1. First name\n");
                Dprint(" 2. Last name\n");
                Dprint(" 3. Roll number\n");
                Dprint(" 4. GPA\n");
                Dprint(" 5. Courses\n");
                Dprint(" Enter your choice: ");
                Dscan("%d", &x);
                switch(x){
                case 1:
                    Dprint(" Enter the new first name: ");
                    Dscan("%s", st[i].fname);
                    Dprint(" [INFO] Updated successfully\n");
                    break;
                case 2:
                    Dprint(" Enter the new last name: ");
                    Dscan("%s", st[i].lname);
                    Dprint(" [INFO] Updated successfully\n");
                    break;
                case 3:
                    Dprint(" Enter the new roll number: ");
                    Dscan("%d", &st[i].roll);
                    Dprint(" [INFO] Updated successfully\n");
                    break;
                case 4:
                    Dprint(" Enter the new GPA: ");
                    Dscan("%f", &st[i].gpa);
                    Dprint(" [INFO] Updated successfully\n");
                    break;
                case 5:
                    Dprint(" Enter the new course ID: \n");
                    for(j = 0; j<5; j++){
                        Dprint(" Course %d ID: ", j+1);
                        Dscan("%d", &st[i].cid[j]);
                    }
                    Dprint(" [INFO] Updated successfully\n");
                    break;
                default:
                    Dprint(" Wrong input value\n");
                    break;
                }
                return;
            }
        }
        /* If the loop ended */
        Dprint(" [ERROR] Roll number %d not found\n", rn);
    }
    else{
        Dprint(" [ERROR] The list is empty!\n");
    }
}
```

9. Print data
    - The function will start by checking the list if it is empty.
    - Loop through each element and print the data.

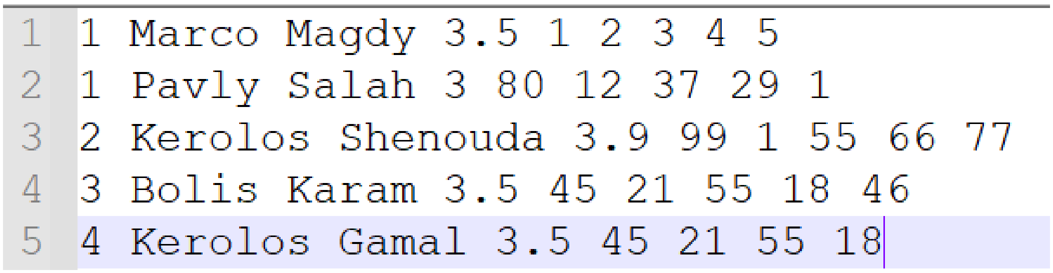


```c
/* Print student details */
void show_s(){
    /* Check if empty */
    if(num_of_students != 0){
        int i, j;
        Dprint(" ----------------------------\n");
        for(i=0; i<num_of_students; i++){
            Dprint("\n Student first name: %s\n", st[i].fname);
            Dprint(" Student last name: %s\n"   , st[i].lname);
            Dprint(" Student roll number: %d\n" , st[i].roll);
            Dprint(" Student GPA number: %.1f\n", st[i].gpa);
            for(j=0; j<num_of_courses; j++){
                Dprint(" Course %d ID: %d\n", j+1, st[i].cid[j]);
            }
        }
        Dprint(" ----------------------------\n");
    }
    else{
        Dprint("[ERROR] The list is empty!\n");
    }
}
```

Students.txt

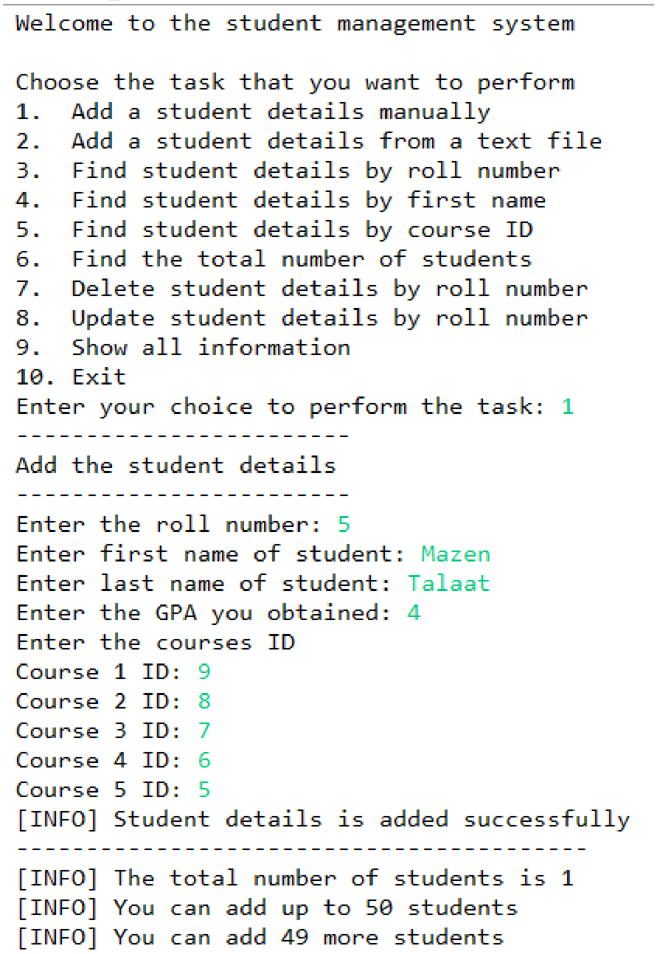
The format is:

Roll_Number First_Name Last_Name GPA C1_ID C2_ID C3_ID C4_ID C5_ID

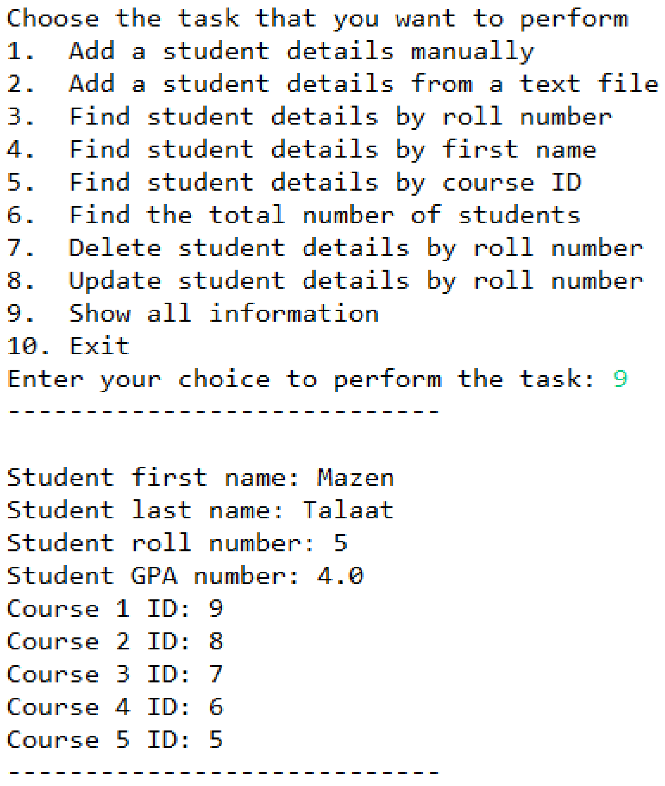


```
1 Marco Magdy 3.5 1 2 3 4 5
1 Pavly Salah 3 80 12 37 29 1
2 Kerolos Shenouda 3.9 99 1 55 66 77
3 Bolis Karam 3.5 45 21 55 18 46
4 Kerolos Gamal 3.5 45 21 55 18
```

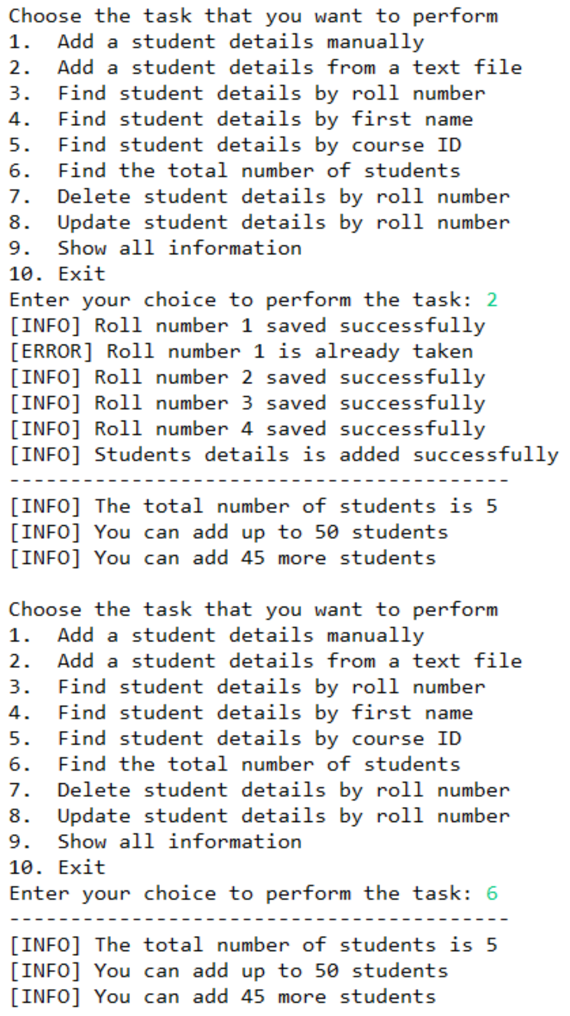
Implemented code output



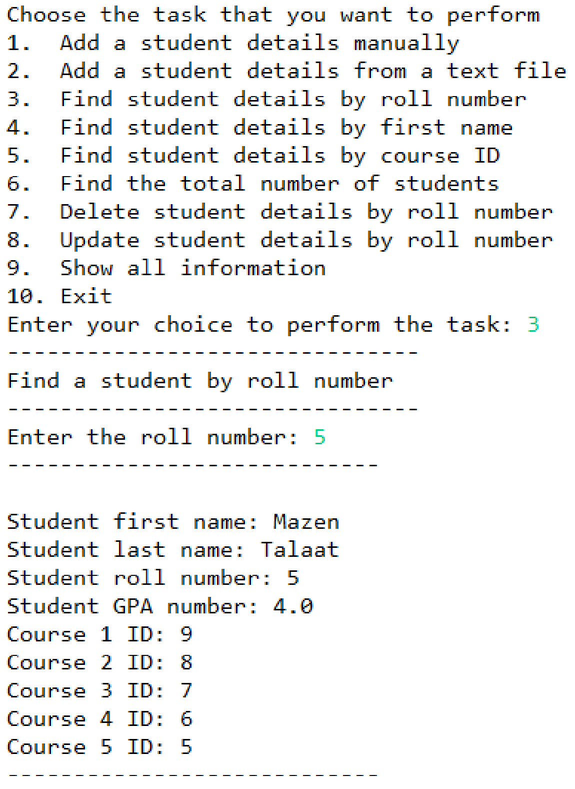
```
Welcome to the student management system

Choose the task that you want to perform
1.  Add a student details manually
2.  Add a student details from a text file
3.  Find student details by roll number
4.  Find student details by first name
5.  Find student details by course ID
6.  Find the total number of students
7.  Delete student details by roll number
8.  Update student details by roll number
9.  Show all information
10. Exit
Enter your choice to perform the task: 1
------------------------
Add the student details
------------------------
Enter the roll number: 5
Enter first name of student: Mazen
Enter last name of student: Talaat
Enter the GPA you obtained: 4
Enter the courses ID
Course 1 ID: 9
Course 2 ID: 8
Course 3 ID: 7
Course 4 ID: 6
Course 5 ID: 5
[INFO] Student details is added successfully
-----------------------------------------
[INFO] The total number of students is 1
[INFO] You can add up to 50 students
[INFO] You can add 49 more students
```



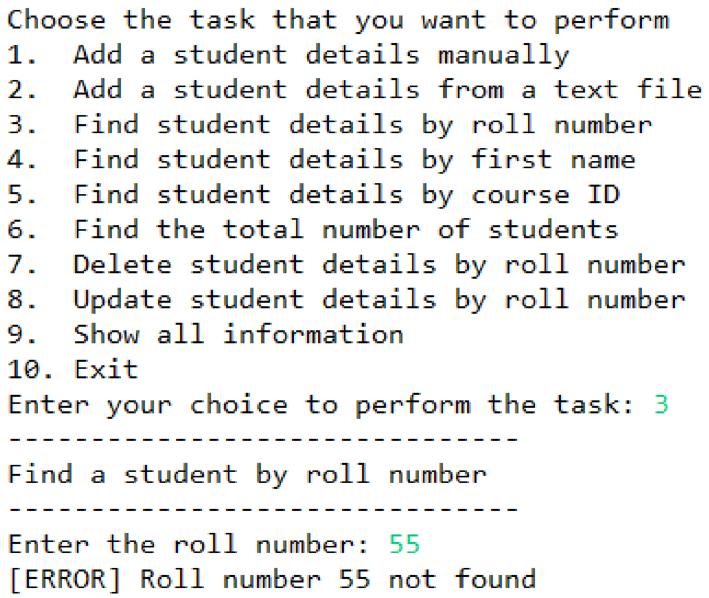
```
Choose the task that you want to perform
1.  Add a student details manually
2.  Add a student details from a text file
3.  Find student details by roll number
4.  Find student details by first name
5.  Find student details by course ID
6.  Find the total number of students
7.  Delete student details by roll number
8.  Update student details by roll number
9.  Show all information
10. Exit
Enter your choice to perform the task: 9
----------------------------

Student first name: Mazen
Student last name: Talaat
Student roll number: 5
Student GPA number: 4.0
Course 1 ID: 9
Course 2 ID: 8
Course 3 ID: 7
Course 4 ID: 6
Course 5 ID: 5
----------------------------
```



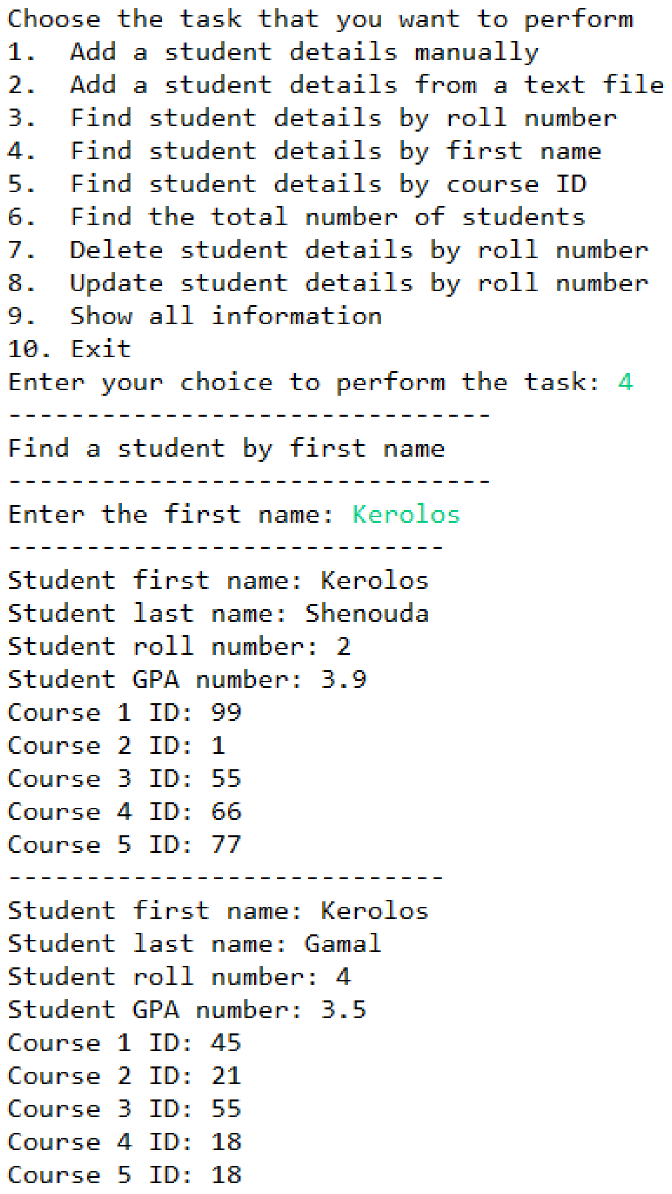
```
Choose the task that you want to perform
1.  Add a student details manually
2.  Add a student details from a text file
3.  Find student details by roll number
4.  Find student details by first name
5.  Find student details by course ID
6.  Find the total number of students
7.  Delete student details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

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



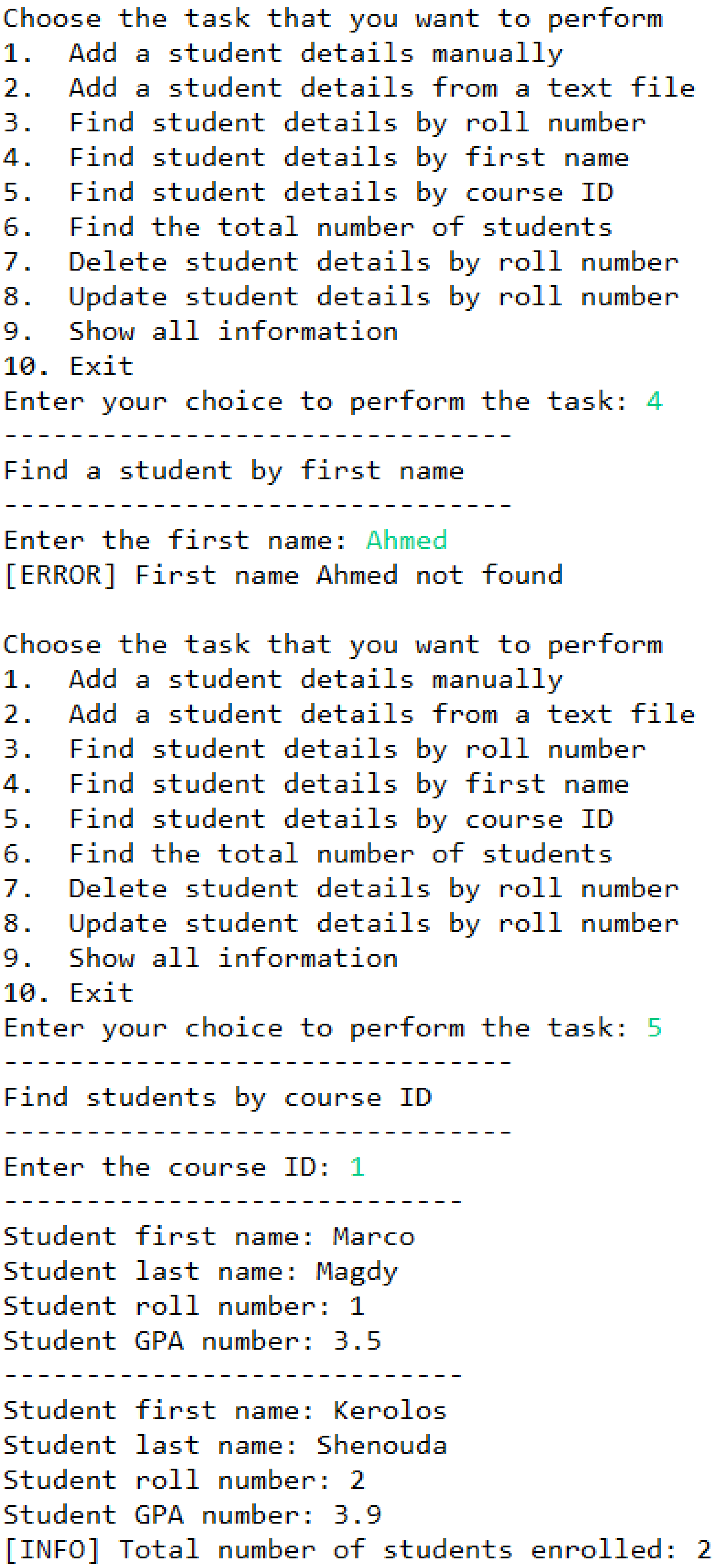
```
Choose the task that you want to perform
1.  Add a student details manually
2.  Add a student details from a text file
3.  Find student details by roll number
4.  Find student details by first name
5.  Find student details by course ID
6.  Find the total number of students
7.  Delete student details by roll number
8.  Update student details by roll number
9.  Show all information
10. Exit
Enter your choice to perform the task: 3
-------------------------------
Find a student by roll number
-------------------------------
Enter the roll number: 5
----------------------------

Student first name: Mazen
Student last name: Talaat
Student roll number: 5
Student GPA number: 4.0
Course 1 ID: 9
Course 2 ID: 8
Course 3 ID: 7
Course 4 ID: 6
Course 5 ID: 5
----------------------------
```



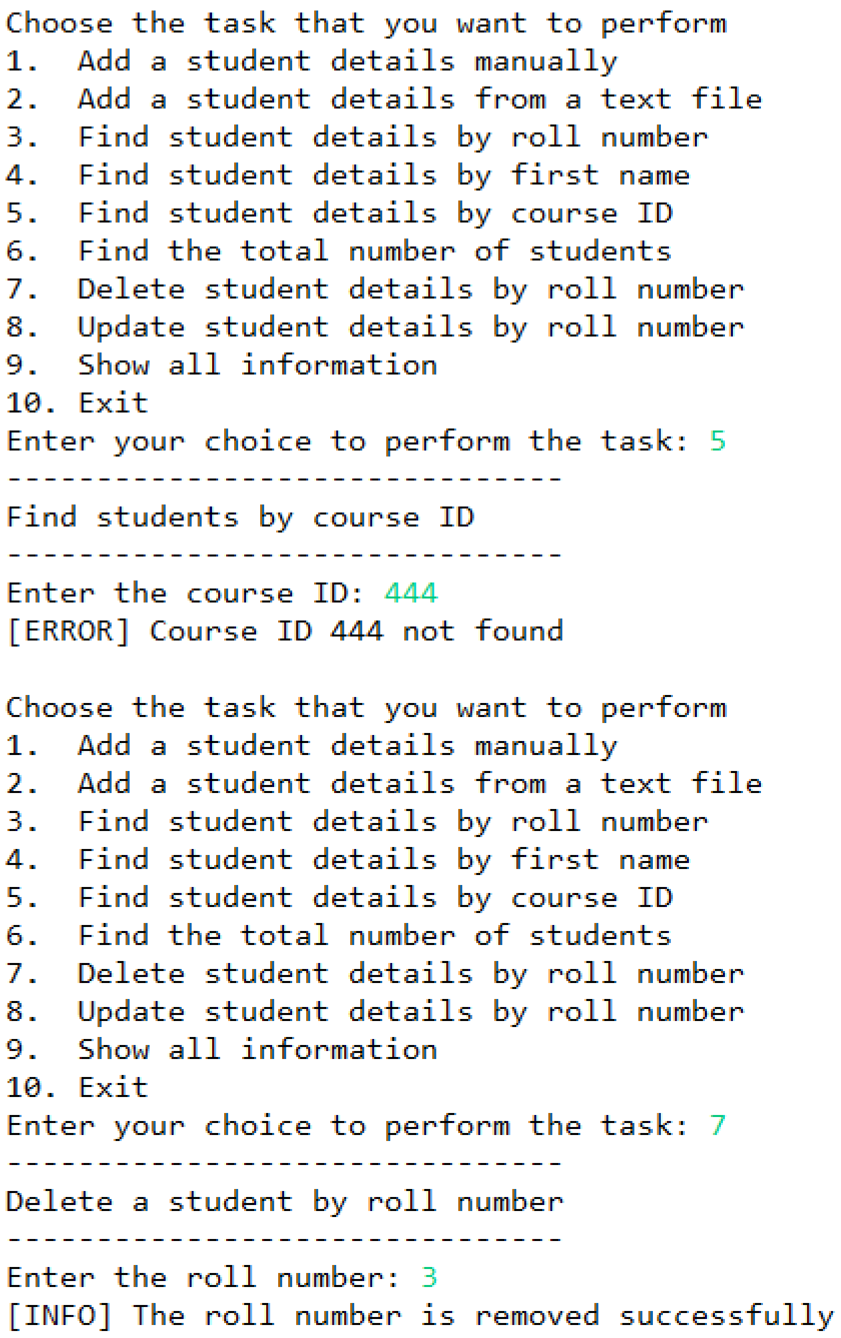
```
Choose the task that you want to perform
1.  Add a student details manually
2.  Add a student details from a text file
3.  Find student details by roll number
4.  Find student details by first name
5.  Find student details by course ID
6.  Find the total number of students
7.  Delete student details by roll number
8.  Update student details by roll number
9.  Show all information
10. Exit
Enter your choice to perform the task: 3
-------------------------------
Find a student by roll number
-------------------------------
Enter the roll number: 55
[ERROR] Roll number 55 not found
```



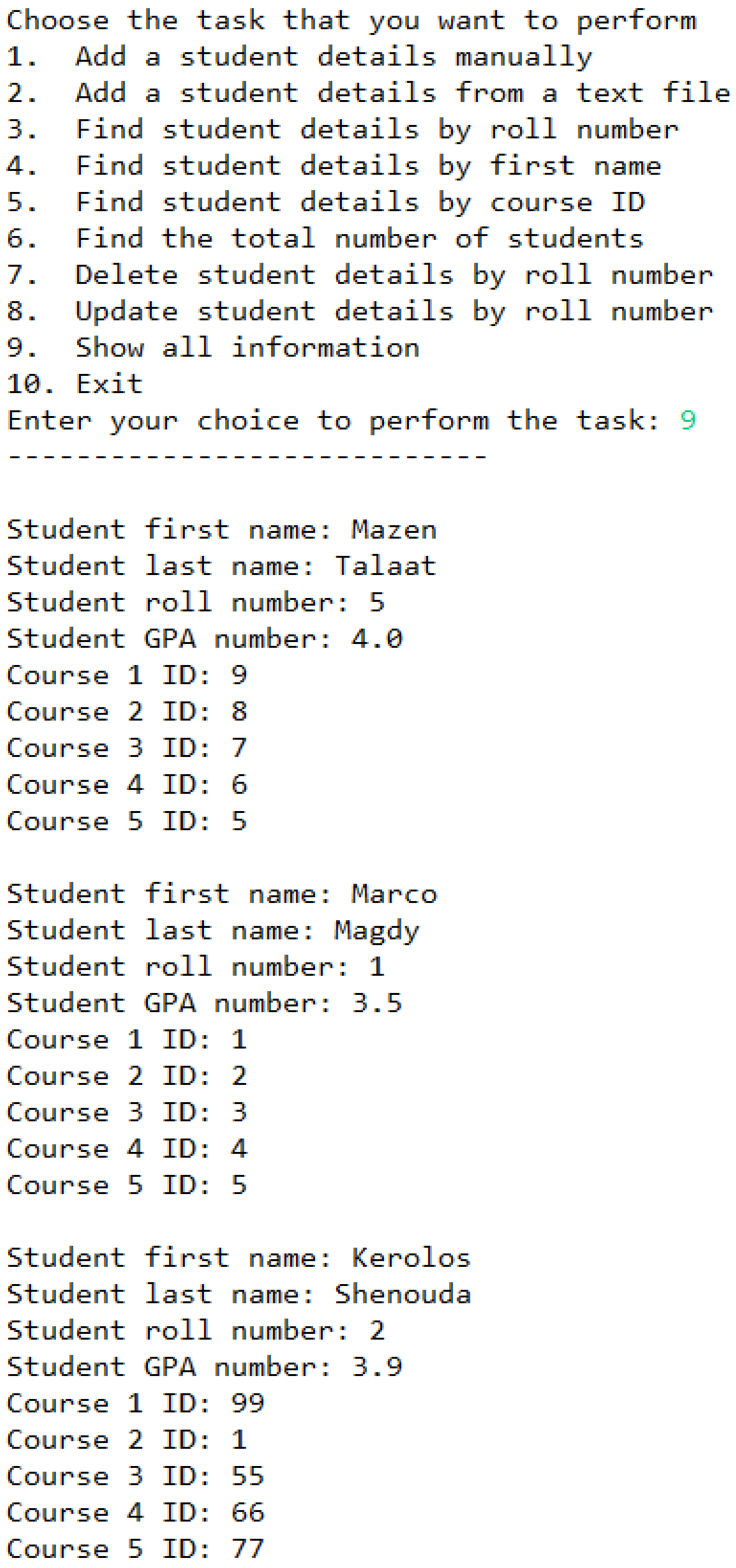
```
Choose the task that you want to perform
1.  Add a student details manually
2.  Add a student details from a text file
3.  Find student details by roll number
4.  Find student details by first name
5.  Find student details by course ID
6.  Find the total number of students
7.  Delete student details by roll number
8.  Update student details by roll number
9.  Show all information
10. Exit
Enter your choice to perform the task: 4
-------------------------------
Find a student by first name
-------------------------------
Enter the first name: Kerolos
----------------------------
Student first name: Kerolos
Student last name: Shenouda
Student roll number: 2
Student GPA number: 3.9
Course 1 ID: 99
Course 2 ID: 1
Course 3 ID: 55
Course 4 ID: 66
Course 5 ID: 77
----------------------------
Student first name: Kerolos
Student last name: Gamal
Student roll number: 4
Student GPA number: 3.5
Course 1 ID: 45
Course 2 ID: 21
Course 3 ID: 55
Course 4 ID: 18
Course 5 ID: 18
```



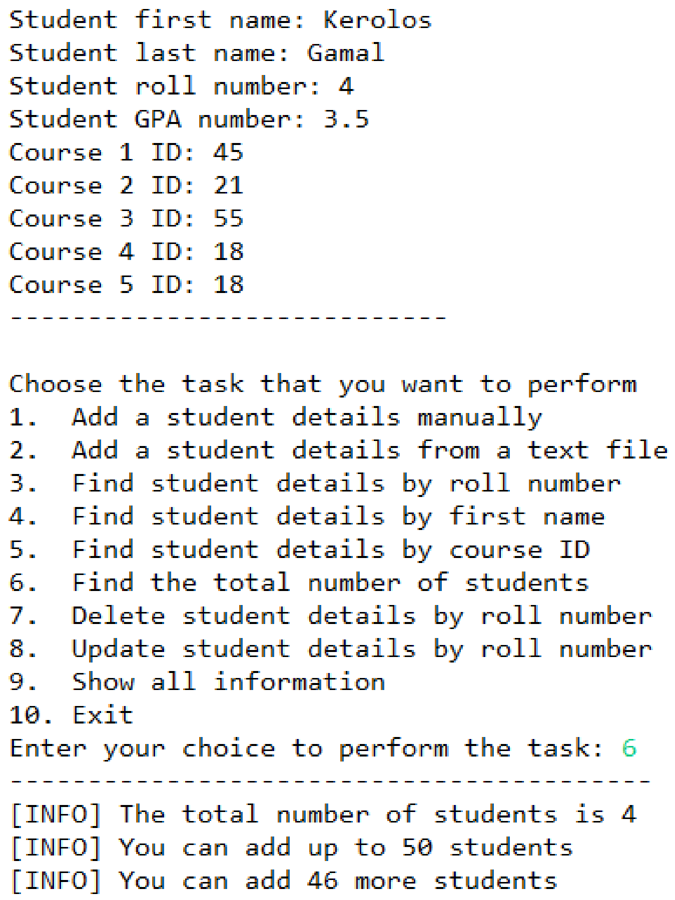
```
Choose the task that you want to perform
1.  Add a student details manually
2.  Add a student details from a text file
3.  Find student details by roll number
4.  Find student details by first name
5.  Find student details by course ID
6.  Find the total number of students
7.  Delete student details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

Choose the task that you want to perform
1.  Add a student details manually
2.  Add a student details from a text file
3.  Find student details by roll number
4.  Find student details by first name
5.  Find student details by course ID
6.  Find the total number of students
7.  Delete student details by roll number
8.  Update student details by roll number
9.  Show all information
10. Exit
Enter your choice to perform the task: 5
-------------------------------
Find students by course ID
-------------------------------
Enter the course ID: 1
----------------------------
Student first name: Marco
Student last name: Magdy
Student roll number: 1
Student GPA number: 3.5
----------------------------
Student first name: Kerolos
Student last name: Shenouda
Student roll number: 2
Student GPA number: 3.9
[INFO] Total number of students enrolled: 2
```



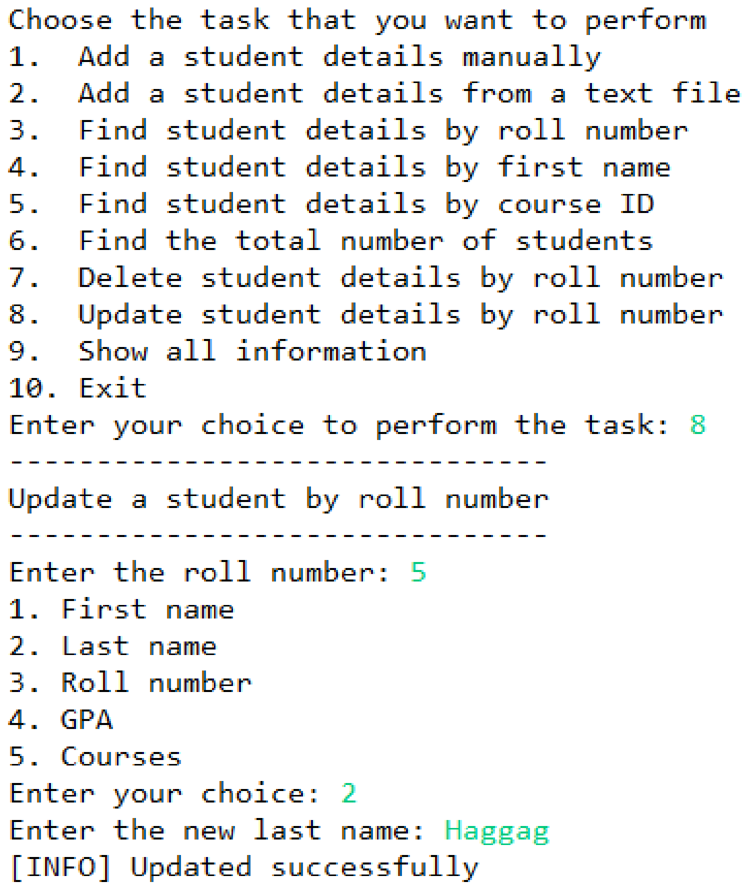
```
Choose the task that you want to perform
1.  Add a student details manually
2.  Add a student details from a text file
3.  Find student details by roll number
4.  Find student details by first name
5.  Find student details by course ID
6.  Find the total number of students
7.  Delete student details by roll number
8.  Update student details by roll number
9.  Show all information
10. Exit
Enter your choice to perform the task: 5
-------------------------------
Find students by course ID
-------------------------------
Enter the course ID: 444
[ERROR] Course ID 444 not found

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



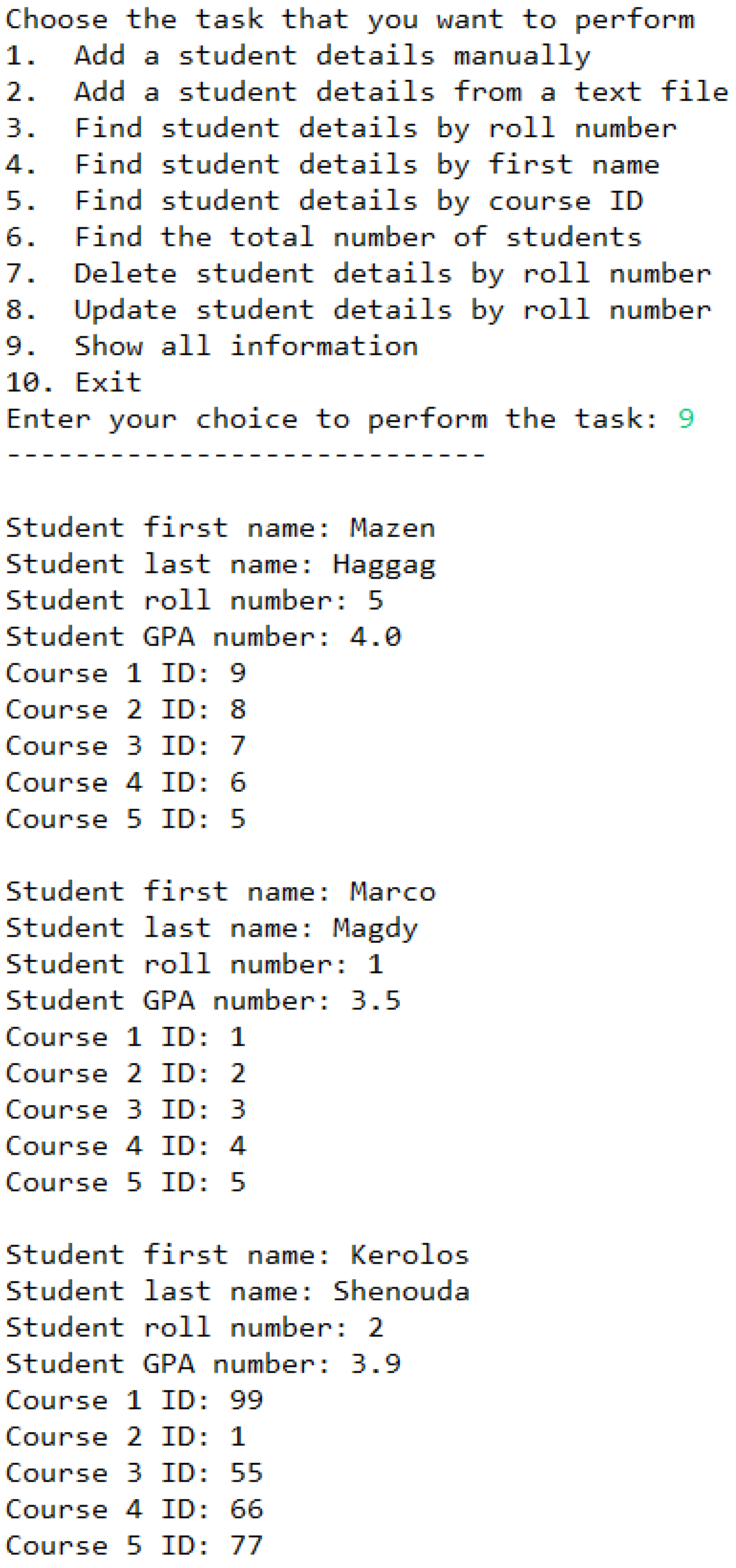
```
Choose the task that you want to perform
1.  Add a student details manually
2.  Add a student details from a text file
3.  Find student details by roll number
4.  Find student details by first name
5.  Find student details by course ID
6.  Find the total number of students
7.  Delete student details by roll number
8.  Update student details by roll number
9.  Show all information
10. Exit
Enter your choice to perform the task: 9
----------------------------

Student first name: Mazen
Student last name: Talaat
Student roll number: 5
Student GPA number: 4.0
Course 1 ID: 9
Course 2 ID: 8
Course 3 ID: 7
Course 4 ID: 6
Course 5 ID: 5

Student first name: Marco
Student last name: Magdy
Student roll number: 1
Student GPA number: 3.5
Course 1 ID: 1
Course 2 ID: 2
Course 3 ID: 3
Course 4 ID: 4
Course 5 ID: 5

Student first name: Kerolos
Student last name: Shenouda
Student roll number: 2
Student GPA number: 3.9
Course 1 ID: 99
Course 2 ID: 1
Course 3 ID: 55
Course 4 ID: 66
Course 5 ID: 77
```



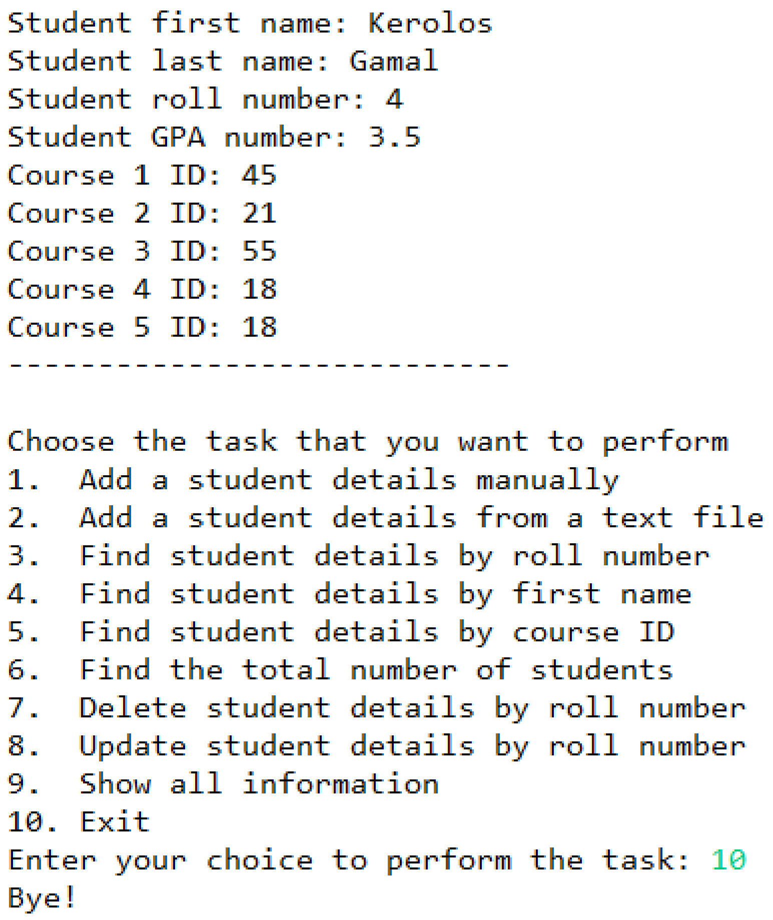
```
Student first name: Kerolos
Student last name: Gamal
Student roll number: 4
Student GPA number: 3.5
Course 1 ID: 45
Course 2 ID: 21
Course 3 ID: 55
Course 4 ID: 18
Course 5 ID: 18
----------------------------

Choose the task that you want to perform
1.  Add a student details manually
2.  Add a student details from a text file
3.  Find student details by roll number
4.  Find student details by first name
5.  Find student details by course ID
6.  Find the total number of students
7.  Delete student details by roll number
8.  Update student details by roll number
9.  Show all information
10. Exit
Enter your choice to perform the task: 6
-----------------------------------------
[INFO] The 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 a student details manually
2.  Add a student details from a text file
3.  Find student details by roll number
4.  Find student details by first name
5.  Find student details by course ID
6.  Find the total number of students
7.  Delete student details by roll number
8.  Update student details by roll number
9.  Show all information
10. Exit
Enter your choice to perform the task: 8
-------------------------------
Update a student by roll number
-------------------------------
Enter the roll number: 5
1. First name
2. Last name
3. Roll number
4. GPA
5. Courses
Enter your choice: 2
Enter the new last name: Haggag
[INFO] Updated successfully
```



```
Choose the task that you want to perform
1.  Add a student details manually
2.  Add a student details from a text file
3.  Find student details by roll number
4.  Find student details by first name
5.  Find student details by course ID
6.  Find the total number of students
7.  Delete student details by roll number
8.  Update student details by roll number
9.  Show all information
10. Exit
Enter your choice to perform the task: 9
----------------------------

Student first name: Mazen
Student last name: Haggag
Student roll number: 5
Student GPA number: 4.0
Course 1 ID: 9
Course 2 ID: 8
Course 3 ID: 7
Course 4 ID: 6
Course 5 ID: 5

Student first name: Marco
Student last name: Magdy
Student roll number: 1
Student GPA number: 3.5
Course 1 ID: 1
Course 2 ID: 2
Course 3 ID: 3
Course 4 ID: 4
Course 5 ID: 5

Student first name: Kerolos
Student last name: Shenouda
Student roll number: 2
Student GPA number: 3.9
Course 1 ID: 99
Course 2 ID: 1
Course 3 ID: 55
Course 4 ID: 66
Course 5 ID: 77
```



```
Student first name: Kerolos
Student last name: Gamal
Student roll number: 4
Student GPA number: 3.5
Course 1 ID: 45
Course 2 ID: 21
Course 3 ID: 55
Course 4 ID: 18
Course 5 ID: 18
----------------------------

Choose the task that you want to perform
1.  Add a student details manually
2.  Add a student details from a text file
3.  Find student details by roll number
4.  Find student details by first name
5.  Find student details by course ID
6.  Find the total number of students
7.  Delete student details by roll number
8.  Update student details by roll number
9.  Show all information
10. Exit
Enter your choice to perform the task: 10
Bye!
```