MASTERING EMBEDDED SYSTEM ONLINE DIPLOMA

www.learn-in-depth.com

FIRST TERM - PROJECT 2

ENG. AHMED ESSAM ELDIN ELSHAFIE
https://www.learn-in-depth.com/online-diploma/ahmedessam020@gmail.com

Table of Contents

STUDENT INFO MANAGEMENT SYSTEM

1 PROBLEM STATEMENT

This project is mainly about implementing a software system to manage the students' information regarding the following:

- First Name.
- Last Name.
- GPA .
- Unique Roll Number.
- Current Enrolled Courses.

2 APPROACH

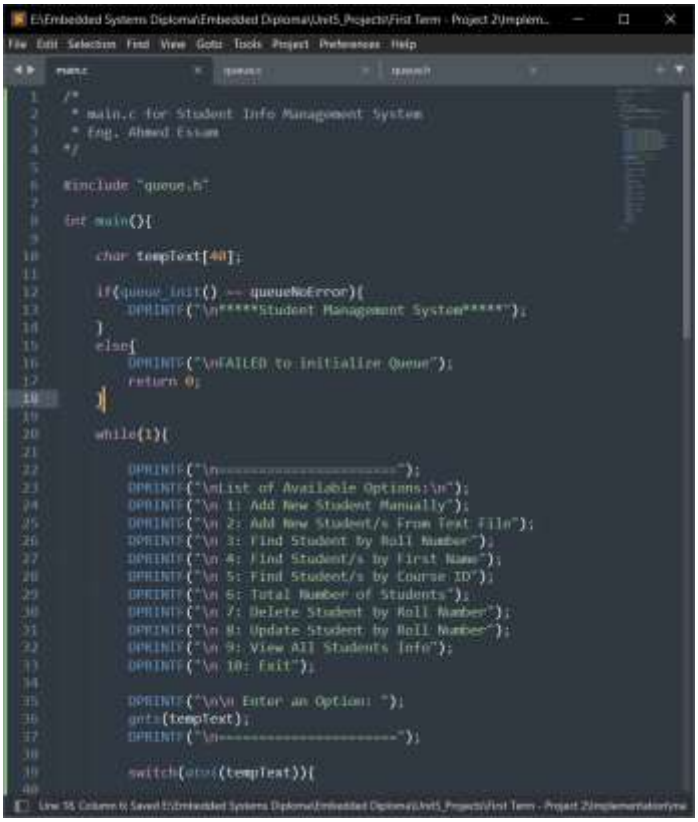
The idea is to form an individual function for each operation. All the functions are unified to form a software system. The functions needed to be implemented are expected to be as following:

1. Add Student Details from File.
2. Add Student Details manually.
3. Find Student by given Roll Number.
4. Find Student by given First Name.
5. Find Students enrolled in a course.
6. Count of students.
7. Delete a student.
8. Update a student.
9. View all info.

We will start implementing each of the following points as a callable functions to be called by user within entry.

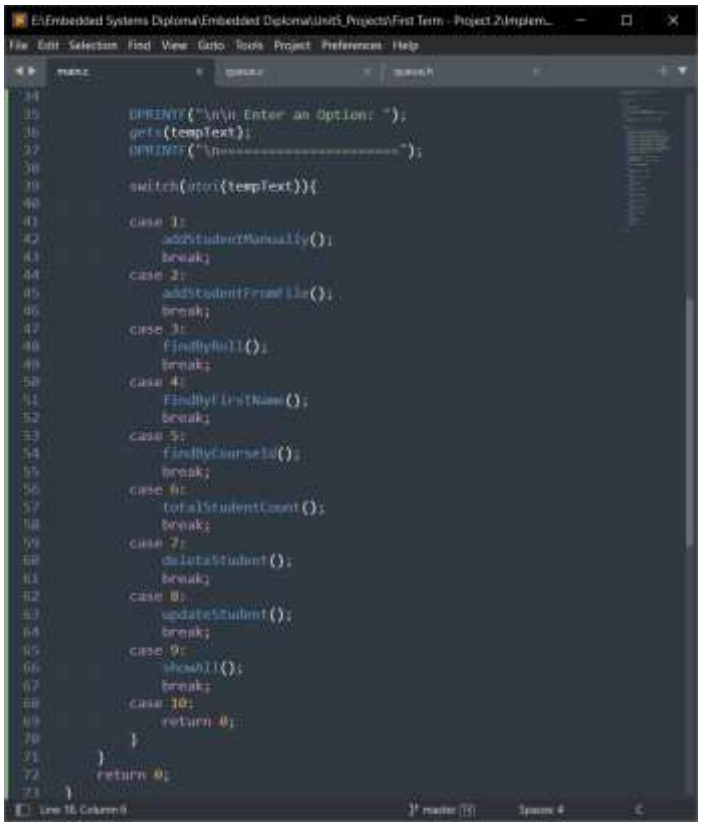
3 IMPLEMENTATION

3.1 main.c



```c
/*
 * main.c for Student Info Management System
 * Eng. Ahmed Essam
*/

#include "queue.h"

int main(){

    char tempText[40];

    if(queue_init() == queueNoError){
        DPRINTF("\n*****Student Management System*****");
    }
    else{
        DPRINTF("\nFAILED to initialize Queue");
        return 0;
    }

    while(1){

        DPRINTF("\n======================");
        DPRINTF("\nList of Available Options:\n");
        DPRINTF("\n 1: Add New Student Manually");
        DPRINTF("\n 2: Add New Student/s From Text File");
        DPRINTF("\n 3: Find Student by Roll Number");
        DPRINTF("\n 4: Find Student/s by First Name");
        DPRINTF("\n 5: Find Student/s by Course ID");
        DPRINTF("\n 6: Total Number of Students");
        DPRINTF("\n 7: Delete Student by Roll Number");
        DPRINTF("\n 8: Update Student by Roll Number");
        DPRINTF("\n 9: View All Students Info");
        DPRINTF("\n 10: Exit");

        DPRINTF("\n\n Enter an Option: ");
        gets(tempText);
        DPRINTF("\n======================");

        switch(atoi(tempText)){
```

Figure 1 - main.c 1/2



```c
        DPRINTF("\n\n Enter an Option: ");
        gets(tempText);
        DPRINTF("\n======================");

        switch(atoi(tempText)){

        case 1:
            addStudentManually();
            break;
        case 2:
            addStudentFromFile();
            break;
        case 3:
            findByRoll();
            break;
        case 4:
            findByFirstName();
            break;
        case 5:
            findByCourseId();
            break;
        case 6:
            totalStudentCount();
            break;
        case 7:
            deleteStudent();
            break;
        case 8:
            updateStudent();
            break;
        case 9:
            showAll();
            break;
        case 10:
            return 0;
        }
    }
    return 0;
}
```

Figure 2 - main.c 2/2

3.2 queue.h

```c
/*
 * queue.h for Student Info Management System
 * Eng. Ahmed Essam
*/

#ifndef QUEUE_H_
#define QUEUE_H_

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

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

#define size 50

//Sensitive Data
typedef struct{
    char fName[50];
    char lName[50];
    int roll;
    //float gpa;
    int courseId[10];
} sData;

extern sData buffer[size];

//Queue Initialization
typedef struct{
    unsigned int length;
    unsigned int count;
    sData* base;
    sData* head;
    sData* tail;
} queueBuffer_t;
```

Figure 3 - queue.h 1/2

```c
//Queue Initialization
typedef struct{
    unsigned int length;
    unsigned int count;
    sData* base;
    sData* head;
    sData* tail;
} queueBuffer_t;

//Queue Status
typedef enum{
    queueNoError,
    queueFull,
    queueEmpty,
    queueNull
} queueStatus;

//Main APIS to be called by user
void addStudentManually();
void addStudentFromFile();
void findByRoll();
void findByFirstName();
void findByCourseId();
void totalStudentCount();
void deleteStudent();
void updateStudent();
void showAll();

//Internal APIs
queueStatus queue_init();
queueStatus queue_isValid();
queueStatus queue_isEmpty();
queueStatus queue_isFull();
int roll_isUnique(int rollCheck);
void printInfo(sData* student);
void updateInfo(sData* student);

#endif /* QUEUE_H_ */
```

Figure 4 - queue.h 2/2

3.3 queue.c

```c
/*
 * queue.c for Student Info Management System
 * Eng. Ahmed Essam
 */

#include "queue.h"

sData buffer[size];
queueBuffer_t mainQueue;
queueBuffer_t* pQueue = &mainQueue;

queueStatus queue_init(){

    //Check Buffer availability
    if(buffer == NULL){
        return queueNull;
    }

    pQueue->base = buffer;
    pQueue->head = buffer;
    pQueue->tail = buffer;
    pQueue->length = size;
    pQueue->count = 0;

    return queueNoError;
}

```

Figure 5 - Global vars and queue_init()

```c
queueStatus queue_isValid(){

    if(!pQueue->base || !pQueue->head || !pQueue->tail)
    {
        DPRINTF("\n[ERROR] Queue is not Valid");
        return queueNull;
    }
    else
        return queueNoError;
}

queueStatus queue_isEmpty(){

    if(pQueue->count == 0){
        DPRINTF("\n[ERROR] Queue is Empty");
        return queueEmpty;
    }
    else
        return queueNoError;
}

queueStatus queue_isFull(){

    if(pQueue->count == pQueue->length)
    {
        DPRINTF("\n[ERROR] Queue is Full");
        return queueFull;
    }
    else
        return queueNoError;
}
```

Figure 6 – isValid(), isEmpty(), isFull()

E:\Embedded Systems Diploma\Embedded Diploma\Unit5_Projects\First Term - Project 2\Implem...

File Edit Selection Find View Goto Tools Project Preferences Help

main.c | queue.c | queue.h

```c

void addStudentManually(){

    if(queue_isValid() != queueNoError && queue_isFull() != queueNoError)
        return;

    DPRINTF("\n----------------------");
    DPRINTF("\nFill out Student Info");
    DPRINTF("\n----------------------");

    sData studentInfo;
    char tempText[40];

    DPRINTF("\nEnter Roll Number: ");
    gets(tempText);
    studentInfo.roll = atoi(tempText);
    if(roll_isUnique(studentInfo.roll) == 0)
        return;

    DPRINTF("Enter First Name: ");
    gets(studentInfo.fName);

    DPRINTF("Enter Last Name: ");
    gets(studentInfo.lName);

    DPRINTF("Enter GPA: ");
    gets(tempText);
    studentInfo.gpa = atof(tempText);

    DPRINTF("Enter each Course ID \n");
    for(int i = 0; i < 5; i++){
        DPRINTF(" Course %d ID: ", i+1);
        gets(tempText);
        studentInfo.courseId[i] = atoi(tempText);
    }

    *(pQueue->head) = studentInfo;
    pQueue->count++;

    if(pQueue->head == (pQueue->base + (pQueue->length * sizeof(queueBuffer_t))))
        pQueue->head = pQueue->base;
    else
        pQueue->head++;
    DPRINTF("\n[INFO] Student Info added successfully\n");
}

```

Figure 7 – addStudentManually()

E:\Embedded Systems Diploma\Embedded Diploma\Unit5_Projects\First Term - Project 2\Implem...

File Edit Selection Find View Goto Tools Project Preferences Help

main.c | queue.c | queue.h

```c
void addStudentFromFile(){
    if(queue_isValid() != queueNoError && queue_isFull() != queueNoError)
        return;

    FILE *fp;
    fp = fopen("addStudentFromFile.txt", "r");
    if(fp == NULL){
        DPRINTF("\n[ERROR] Failed to open file");
        return;
    }

    char line[100];
    char *token;

    while(fgets(line, sizeof(line), fp)){

        int counter = 1;
        int course = 0;
        sData newStudent;
        token = strtok(line, " ");

        while(token != NULL){
            switch(counter){
            case 1:
                newStudent.roll = atoi(token);
                if(roll_isUnique(newStudent.roll) == 0)
                    return;
                break;
            case 2:
                strcpy(newStudent.fName, token);
                break;
            case 3:
                strcpy(newStudent.lName, token);
                break;
            case 4:
                newStudent.gpa = atof(token);
                break;
            case 5:
                newStudent.courseId[course] = atoi(token);
                course++;
                counter--;
                break;
            }
            token = strtok(NULL, " ");
            counter++;
        }
        *(pQueue->head) = newStudent;
        pQueue->count++;

        if(pQueue->head == (pQueue->base + (pQueue->length * sizeof(queueBuffer_t))))
            pQueue->head = pQueue->base;
        else
            pQueue->head++;
    }
    fclose(fp);
```

Line 110, Column 47; Saved E:\Embedded Systems Diploma\Embedded Diploma\Unit5_Projects\First Term - Project 2\Implementation\c

Figure 8 - addStudentFromFile()

E:\Embedded Systems Diploma\Embedded Diploma\Unit5_Projects\First Term - Project 2\Implem...

File Edit Selection Find View Goto Tools Project Preferences Help

main.c | queue.c | queue.h

```c
void findByRoll(){

    if(queue_isValid() != queueNoError && queue_isEmpty() != queueNoError)
        return;

    char tempText[40];
    int rollNum;

    DPRINTF("\nEnter Student's Roll Number: ");
    gets(tempText);
    rollNum = atoi(tempText);

    sData *student;
    student = pQueue->tail;
    for(int i = 0; i < pQueue->count; i++){
        if(student->roll == rollNum){
            printInfo(student);
            return;
        }
        student++;
    }
    DPRINTF("\n[ERROR] Roll Number not found");
}
```

Figure 9 - findByRoll()

E:\Embedded Systems Diploma\Embedded Diploma\Unit5_Projects\First Term - Project 2\Implem...

File Edit Selection Find View Goto Tools Project Preferences Help

main.c | queue.c | queue.h

```c
void findByFirstName(){

    if(queue_isValid() != queueNoError && queue_isEmpty() != queueNoError)
        return;

    char firstName[40];
    int flag = 0;

    DPRINTF("\nEnter Student's First Name: ");
    gets(firstName);

    sData *student;
    student = pQueue->tail;
    for(int i = 0; i < pQueue->count; i++){
        if(strcmp(student->fName, firstName) == 0){
            printInfo(student);
            flag = 1;
        }
        student++;
    }
    if(flag == 0)
        DPRINTF("\n[ERROR] Student's First Name not found");

}
```

Figure 10 - findByFirstName()

E:\Embedded Systems Diploma\Embedded Diploma\Unit5_Projects\First Term - Project 2\Implem...

File Edit Selection Find View Goto Tools Project Preferences Help

main.c | queue.c | queue.h

```c
void findByCourseId(){

    if(queue_isValid() != queueNoError && queue_isEmpty() != queueNoError)
        return;

    char tempText[40];
    int course;

    DPRINTF("\nEnter Course ID: ");
    gets(tempText);
    course = atoi(tempText);

    sData *student;
    student = pQueue->tail;
    int flag = 0;
    for(int i = 0; i < pQueue->count; i++){
        for(int j = 0; j < 5; j++){
            if(student->courseId[j] == course){
                printInfo(student);
                flag = 1;
            }
        }
        student++;
    }
    if(flag == 0)
        DPRINTF("\n[ERROR] Course ID not found");
}
```

Figure 11 – findByCourseId()

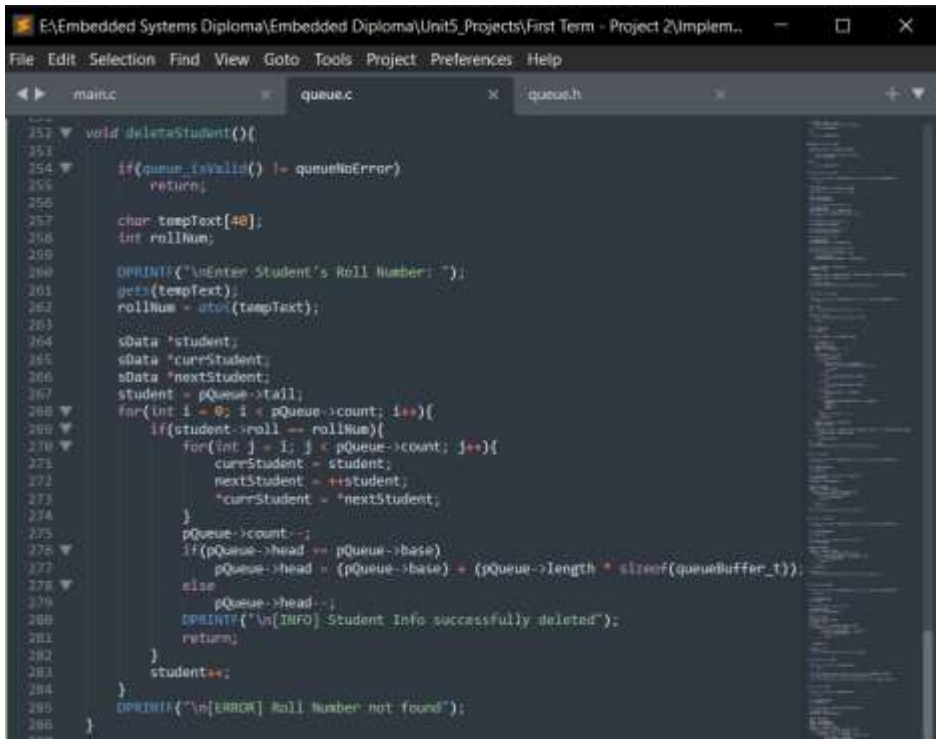
E:\Embedded Systems Diploma\Embedded Diploma\Unit5_Projects\First Term - Project 2\Implem...

File Edit Selection Find View Goto Tools Project Preferences Help

main.c | queue.c | queue.h

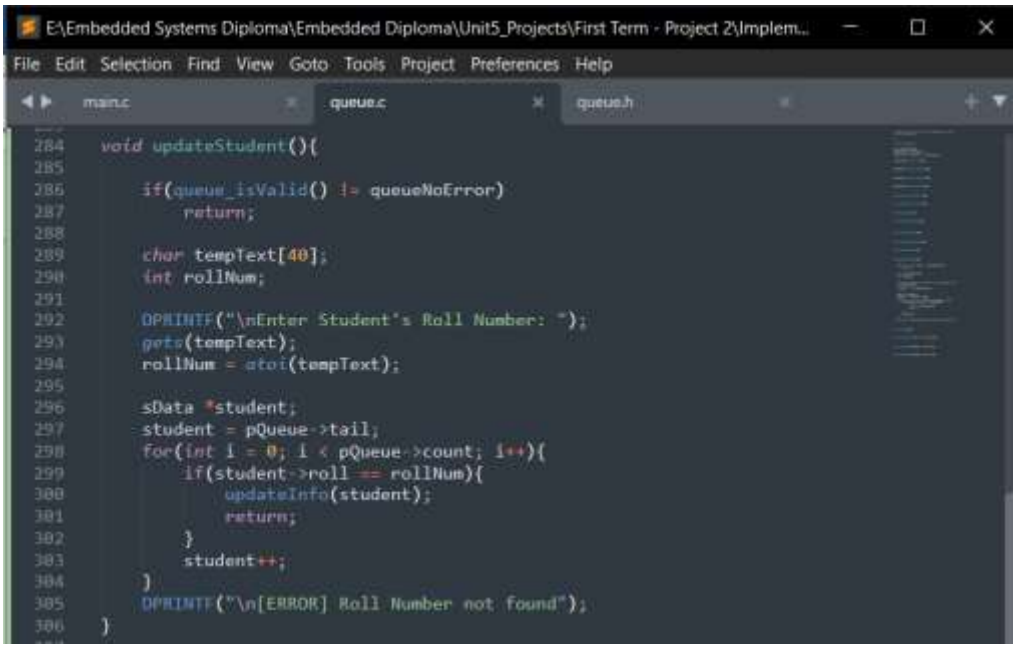
```c
void totalStudentCount(){

    if(queue_isValid() != queueNoError)
        return;

    DPRINTF("\n[INFO] Total Number of Students = %d", pQueue->count);
    DPRINTF("\n[INFO] You can add up to 50 Students");
    DPRINTF("\n[INFO] %d more slots are available", 50 - pQueue->count);
}
```

Figure 12 - totalStudentCount()




```
void deleteStudent(){

    if(queue_isValid() != queueNoError)
        return;

    char tempText[40];
    int rollNum;

    DPRINTF("\nEnter Student's Roll Number: ");
    gets(tempText);
    rollNum = atoi(tempText);

    sData *student;
    sData *currStudent;
    sData *nextStudent;
    student = pQueue->tail;
    for(int i = 0; i < pQueue->count; i++){
        if(student->roll == rollNum){
            for(int j = i; j < pQueue->count; j++){
                currStudent = student;
                nextStudent = ++student;
                *currStudent = *nextStudent;
            }
            pQueue->count--;
            if(pQueue->head == pQueue->base)
                pQueue->head = (pQueue->base) + (pQueue->length * sizeof(queueBuffer_t));
            else
                pQueue->head--;
            DPRINTF("\n[INFO] Student Info successfully deleted");
            return;
        }
        student++;
    }
    DPRINTF("\n[ERROR] Roll Number not found");
}
```

Figure 13 - deleteStudent()




```
void updateStudent(){

    if(queue_isValid() != queueNoError)
        return;

    char tempText[40];
    int rollNum;

    DPRINTF("\nEnter Student's Roll Number: ");
    gets(tempText);
    rollNum = atoi(tempText);

    sData *student;
    student = pQueue->tail;
    for(int i = 0; i < pQueue->count; i++){
        if(student->roll == rollNum){
            updateInfo(student);
            return;
        }
        student++;
    }
    DPRINTF("\n[ERROR] Roll Number not found");
}
```

Figure 13 - updateStudent()

```c
void showAll(){

    if(queue_isValid() != queueNoError)
        return;

    sData *student;
    student = pQueue->tail;
    for(int i = 0; i < pQueue->count; i++){
        printInfo(student);
        student++;
    }
}
```

Figure 14 - showAll()

E:\Embedded Systems Diploma\Embedded Diploma\Unit5_Projects\First Term - Project 2\Implem...

File Edit Selection Find View Goto Tools Project Preferences Help

main.c queue.c queue.h

```c
void showAll(){ ...
}

int roll_isUnique(int rollCheck){

    sData *student;
    student = pQueue->tail;
    for(int i = 0; i < pQueue->count; i++){
        if(student->roll == rollCheck){
            DPRINTF("\n[ERROR] Roll Number is already taken");
            return 0;
        }
        student++;
    }
    return 1;
}
```

Figure 15 - roll_isUnique()

```c
void showAll(){
}

int roll_isUnique(int rollCheck){
}

void printInfo(sData* student){

    OPRINTF("\n\tRoll Number: %d", student->roll);
    OPRINTF("\n\tFirst Name: %s", student->fName);
    OPRINTF("\n\tLast Name: %s", student->lName);
    OPRINTF("\n\tGPA: %.2f", student->gpa);
    OPRINTF("\n\tCourses ID: ");
    for(int i = 0; i < 5; i++){
        OPRINTF("%d ", student->courseId[i]);
    }
    OPRINTF("\n----------------------");
}
```

Figure 16 - printInfo()

```c
void updateInfo(sData* student){

    char tempText[40];
    OPRINTF("\nChoose data to be updated: ");
    OPRINTF("\n 1: Roll Number");
    OPRINTF("\n 2: First Name");
    OPRINTF("\n 3: Last Name");
    OPRINTF("\n 4: GPA");
    OPRINTF("\n 5: Courses ID");
    OPRINTF("\n\nEnter an option: ");
    gets(tempText);

    switch(atoi(tempText)){
    case 1:
        OPRINTF("\nEnter Roll Number: ");
        gets(tempText);
        student->roll = atoi(tempText);
        if(!roll_isUnique(student->roll))
            return;
        else
            OPRINTF("\n\n[INFO] Data updated successfully");
        break;

    case 2:
        OPRINTF("\nEnter First Name: ");
        gets(student->fName);
        OPRINTF("\n\n[INFO] Data updated successfully");
        break;

    case 3:
        OPRINTF("\nEnter Last Name: ");
        gets(student->lName);
        OPRINTF("\n\n[INFO] Data updated successfully");
        break;

    case 4:
        OPRINTF("\nEnter GPA: ");
        gets(tempText);
        student->gpa = atof(tempText);
        OPRINTF("\n\n[INFO] Data updated successfully");
        break;

    case 5:
        OPRINTF("\nEnter each Course ID \n");
        for(int i = 0; i < 5; i++){
            OPRINTF(" Course %d ID: ", i+1);
            gets(tempText);
            student->courseId[i] = atoi(tempText);
        }
        OPRINTF("\n\n[INFO] Data updated successfully");
        break;
    }
}
```

Figure 17 - updateInfo()

4 OUTPUT

4.1 addStudentManually()

```
*****Student Management System*****
======================
List of Available Options:

 1: Add New Student Manually
 2: Add New Student/s From Text File
 3: Find Student by Roll Number
 4: Find Student/s by First Name
 5: Find Student/s by Course ID
 6: Total Number of Students
 7: Delete Student by Roll Number
 8: Update Student by Roll Number
 9: View All Students Info
 10: Exit

 Enter an Option: 1

======================
----------------------
Fill out Student Info
----------------------
Enter Roll Number: 5
Enter First Name: Ahmed
Enter Last Name: ElShafie
Enter GPA: 3.5
Enter each Course ID
 Course 1 ID: 6
 Course 2 ID: 7
 Course 3 ID: 8
 Course 4 ID: 9
 Course 5 ID: 1

[INFO] Student Info added successfully
```

Figure 18 - Output addStudentManually()

```
======================
List of Available Options:

 1: Add New Student Manually
 2: Add New Student/s From Text File
 3: Find Student by Roll Number
 4: Find Student/s by First Name
 5: Find Student/s by Course ID
 6: Total Number of Students
 7: Delete Student by Roll Number
 8: Update Student by Roll Number
 9: View All Students Info
 10: Exit

 Enter an Option: 9

======================
        Roll Number: 5
        First Name: Ahmed
        Last Name: ElShafie
        GPA: 3.50
        Courses ID: 6 7 8 9 1
----------------------
======================
```

Figure 19 - showAll() after adding.

4.2 addStudentFromFile()

```
=====================
List of Available Options:

 1: Add New Student Manually
 2: Add New Student/s From Text File
 3: Find Student by Roll Number
 4: Find Student/s by First Name
 5: Find Student/s by Course ID
 6: Total Number of Students
 7: Delete Student by Roll Number
 8: Update Student by Roll Number
 9: View All Students Info
 10: Exit

 Enter an Option: 2

=====================
[INFO] Student Info added successfully
```

Figure 20 - Output addStudentFromFile()

```
=====================
        Roll Number: 5
        First Name: Ahmed
        Last Name: ElShafie
        GPA: 3.50
        Courses ID: 6 7 8 9 1
---------------------
        Roll Number: 1
        First Name: Ahmed
        Last Name: Essam
        GPA: 3.00
        Courses ID: 1 2 3 4 5
---------------------
        Roll Number: 2
        First Name: Hagar
        Last Name: Karkar
        GPA: 2.50
        Courses ID: 11 22 3 44 55
---------------------
        Roll Number: 3
        First Name: Reem
        Last Name: Sadek
        GPA: 3.30
        Courses ID: 11 66 77 88 99
---------------------
        Roll Number: 4
        First Name: Zead
        Last Name: Hani
        GPA: 2.80
        Courses ID: 1 22 77 3 99
---------------------
```

Figure 21 - showAll() after adding.

4.3 findByRoll()

```
 Enter an Option: 3

======================
Enter Student's Roll Number: 1

        Roll Number: 1
        First Name: Ahmed
        Last Name: Essam
        GPA: 3.00
        Courses ID: 1 2 3 4 5
----------------------
======================
List of Available Options:

 1: Add New Student Manually
 2: Add New Student/s From Text File
 3: Find Student by Roll Number
 4: Find Student/s by First Name
 5: Find Student/s by Course ID
 6: Total Number of Students
 7: Delete Student by Roll Number
 8: Update Student by Roll Number
 9: View All Students Info
 10: Exit

 Enter an Option: 3

======================
Enter Student's Roll Number: 2

        Roll Number: 2
        First Name: Hagar
        Last Name: Karkar
        GPA: 2.50
        Courses ID: 11 22 3 44 55
----------------------
```

Figure 22 - Output findByRoll()

4.4 findByFirstName()

```
=====================
List of Available Options:

 1: Add New Student Manually
 2: Add New Student/s From Text File
 3: Find Student by Roll Number
 4: Find Student/s by First Name
 5: Find Student/s by Course ID
 6: Total Number of Students
 7: Delete Student by Roll Number
 8: Update Student by Roll Number
 9: View All Students Info
 10: Exit

 Enter an Option: 4

=====================
Enter Student's First Name: Ahmed

        Roll Number: 5
        First Name: Ahmed
        Last Name: ElShafie
        GPA: 3.50
        Courses ID: 6 7 8 9 1
---------------------
        Roll Number: 1
        First Name: Ahmed
        Last Name: Essam
        GPA: 3.00
        Courses ID: 1 2 3 4 5
---------------------
```

Figure 23 - Output findByFirstName()

4.5 findByCourseId()

```
=======================
List of Available Options:

 1: Add New Student Manually
 2: Add New Student/s From Text File
 3: Find Student by Roll Number
 4: Find Student/s by First Name
 5: Find Student/s by Course ID
 6: Total Number of Students
 7: Delete Student by Roll Number
 8: Update Student by Roll Number
 9: View All Students Info
 10: Exit

 Enter an Option: 5

=======================
Enter Course ID: 1

        Roll Number: 5
        First Name: Ahmed
        Last Name: ElShafie
        GPA: 3.50
        Courses ID: 6 7 8 9 1
-----------------------
        Roll Number: 1
        First Name: Ahmed
        Last Name: Essam
        GPA: 3.00
        Courses ID: 1 2 3 4 5
-----------------------
        Roll Number: 4
        First Name: Zead
        Last Name: Hani
        GPA: 2.80
        Courses ID: 1 22 77 3 99
-----------------------
```

Figure 24 - Output findByCourseId()

4.6 totalStudentCount()

```
=======================
List of Available Options:

 1: Add New Student Manually
 2: Add New Student/s From Text File
 3: Find Student by Roll Number
 4: Find Student/s by First Name
 5: Find Student/s by Course ID
 6: Total Number of Students
 7: Delete Student by Roll Number
 8: Update Student by Roll Number
 9: View All Students Info
 10: Exit

 Enter an Option: 6

=======================
[INFO] Total Number of Students = 5
[INFO] You can add up to 50 Students
[INFO] 45 more slots are available
=======================
```

Figure 25 - Output totalStudentCount()

4.7 deleteStudent()

```
=====================
List of Available Options:

 1: Add New Student Manually
 2: Add New Student/s From Text File
 3: Find Student by Roll Number
 4: Find Student/s by First Name
 5: Find Student/s by Course ID
 6: Total Number of Students
 7: Delete Student by Roll Number
 8: Update Student by Roll Number
 9: View All Students Info
 10: Exit

 Enter an Option: 7

=====================
Enter Student's Roll Number: 5

[INFO] Student Info successfully deleted
=====================
```

Figure 26 - Output deleteStudent()

```
 Enter an Option: 9

=====================
        Roll Number: 1
        First Name: Ahmed
        Last Name: Essam
        GPA: 3.00
        Courses ID: 1 2 3 4 5
---------------------
        Roll Number: 2
        First Name: Hagar
        Last Name: Karkar
        GPA: 2.50
        Courses ID: 11 22 3 44 55
---------------------
        Roll Number: 3
        First Name: Reem
        Last Name: Sadek
        GPA: 3.30
        Courses ID: 11 66 77 88 99
---------------------
        Roll Number: 4
        First Name: Zead
        Last Name: Hani
        GPA: 2.80
        Courses ID: 1 22 77 3 99
---------------------
=====================
```

Figure 27 - showAll() after deleting

4.8 updateStudent()

```
=====================
List of Available Options:

 1: Add New Student Manually
 2: Add New Student/s From Text File
 3: Find Student by Roll Number
 4: Find Student/s by First Name
 5: Find Student/s by Course ID
 6: Total Number of Students
 7: Delete Student by Roll Number
 8: Update Student by Roll Number
 9: View All Students Info
 10: Exit

 Enter an Option: 8

=====================
Enter Student's Roll Number: 1

Choose data to be updated:
 1: Roll Number
 2: First Name
 3: Last Name
 4: GPA
 5: Courses ID

Enter an option: 4

Enter GPA: 3.7


[INFO] Data updated successfully
=====================
```

Figure 28 - Output updateStudent()

```
 Enter an Option: 9

=====================
        Roll Number: 1
        First Name: Ahmed
        Last Name: Essam
        GPA: 3.70
        Courses ID: 1 2 3 4 5
---------------------
        Roll Number: 2
        First Name: Hagar
        Last Name: Karkar
        GPA: 2.50
        Courses ID: 11 22 3 44 55
---------------------
        Roll Number: 3
        First Name: Reem
        Last Name: Sadek
        GPA: 3.30
        Courses ID: 11 66 77 88 99
---------------------
        Roll Number: 4
        First Name: Zead
        Last Name: Hani
        GPA: 2.80
        Courses ID: 1 22 77 3 99
---------------------
```

Figure 29 - showAll() after updating

4.9 showAll()

```
 Enter an Option: 9

======================
        Roll Number: 1
        First Name: Ahmed
        Last Name: Essam
        GPA: 3.70
        Courses ID: 1 2 3 4 5
----------------------
        Roll Number: 2
        First Name: Hagar
        Last Name: Karkar
        GPA: 2.50
        Courses ID: 11 22 3 44 55
----------------------
        Roll Number: 3
        First Name: Reem
        Last Name: Sadek
        GPA: 3.30
        Courses ID: 11 66 77 88 99
----------------------
        Roll Number: 4
        First Name: Zead
        Last Name: Hani
        GPA: 2.80
        Courses ID: 1 22 77 3 99
----------------------
======================
```

Figure 30 - Output showAll()

4.10 Other Features

```
=======================
List of Available Options:

 1: Add New Student Manually
 2: Add New Student/s From Text File
 3: Find Student by Roll Number
 4: Find Student/s by First Name
 5: Find Student/s by Course ID
 6: Total Number of Students
 7: Delete Student by Roll Number
 8: Update Student by Roll Number
 9: View All Students Info
 10: Exit

 Enter an Option: 1

=======================
-----------------------
Fill out Student Info
-----------------------
Enter Roll Number: 1

[ERROR] Roll Number is already taken
=======================
```

Figure 30 - Adding duplicate Roll Number

```
=======================
List of Available Options:

 1: Add New Student Manually
 2: Add New Student/s From Text File
 3: Find Student by Roll Number
 4: Find Student/s by First Name
 5: Find Student/s by Course ID
 6: Total Number of Students
 7: Delete Student by Roll Number
 8: Update Student by Roll Number
 9: View All Students Info
 10: Exit

 Enter an Option: 3

=======================
Enter Student's Roll Number: 7

[ERROR] Roll Number not found
=======================
```

Figure 31 - Roll Number not found

```
=======================
List of Available Options:

 1: Add New Student Manually
 2: Add New Student/s From Text File
 3: Find Student by Roll Number
 4: Find Student/s by First Name
 5: Find Student/s by Course ID
 6: Total Number of Students
 7: Delete Student by Roll Number
 8: Update Student by Roll Number
 9: View All Students Info
 10: Exit

 Enter an Option: 4

======================
Enter Student's First Name: Omar

[ERROR] Student's First Name not found
=======================
```

Figure 32 - First Name not found

```
=======================
List of Available Options:

 1: Add New Student Manually
 2: Add New Student/s From Text File
 3: Find Student by Roll Number
 4: Find Student/s by First Name
 5: Find Student/s by Course ID
 6: Total Number of Students
 7: Delete Student by Roll Number
 8: Update Student by Roll Number
 9: View All Students Info
 10: Exit

 Enter an Option: 5

======================
Enter Course ID: 17

[ERROR] Course ID not found
=======================
```

Figure 33 - Course ID not found