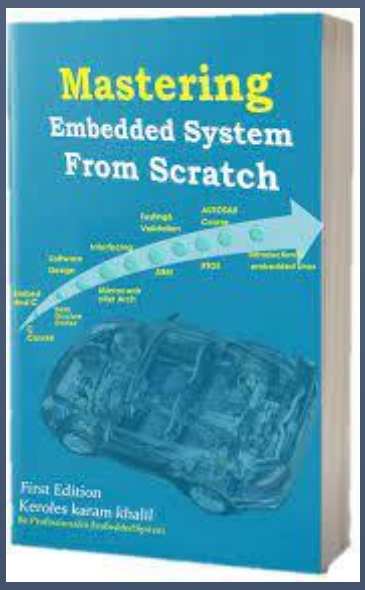


Student Management System

Mastering Embedded System Online Diploma

www.learn-in-depth.com

First Term Project (Final Project 2)

Eng. Ahmed Nabil Mahmoud

https://www.learn-in-depth.com/online-diploma/ahmed.nabil.9711%40gmail.com

Contents

1. case study

- Requirements

The client request is to implement a student management system through which students data could be stored and program user could have control over these data & some features like :-

1. students information could be added manually or from a text file.
2. student's information could be displayed by roll number or by first name .
3. Displaying number of students applied for a certain course.
4. Displaying total number of students and all students information.
5. delete or update student by roll number.

- Assumptions

1. Data will be erased when the program terminates.
2. maximum number of students could be added is 50.
3. Data validity is user's responsibility.
4. Data values that will be Entered have a logic limit.

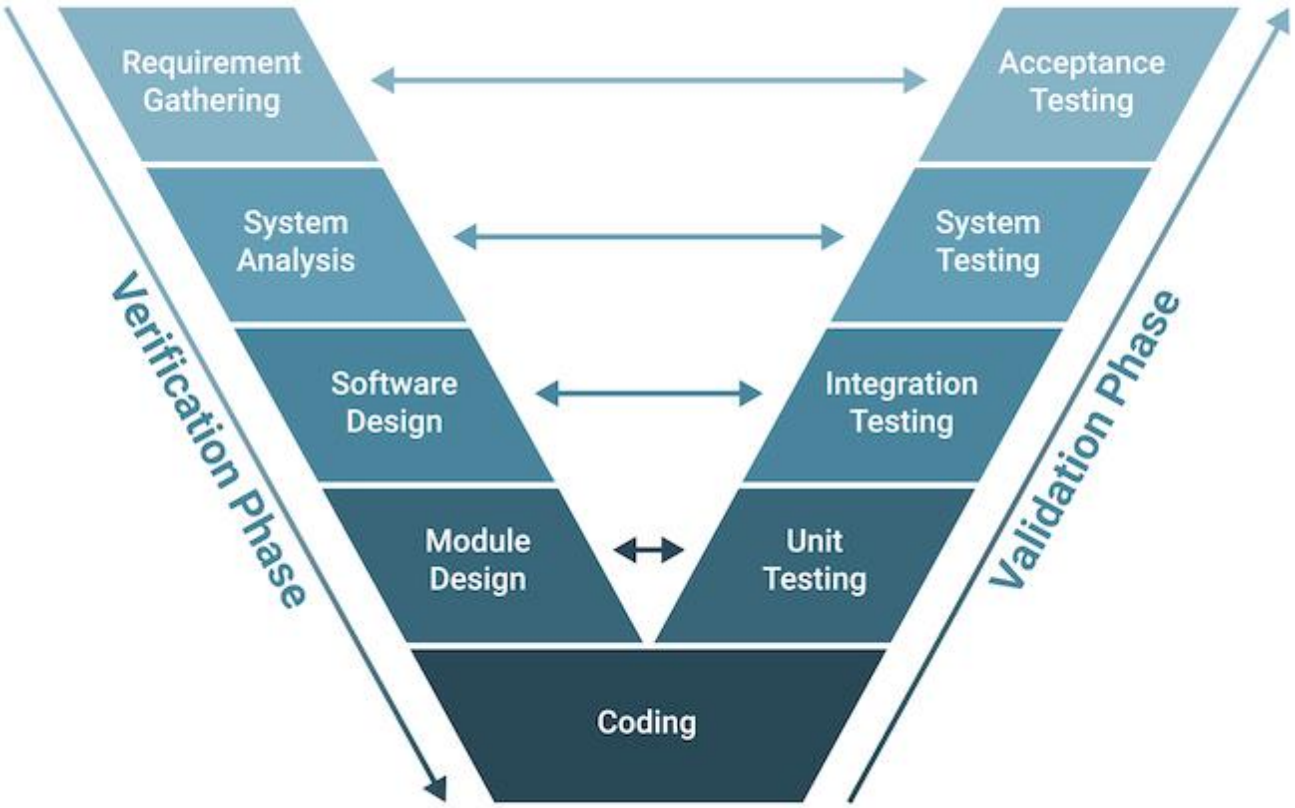
- Versioning

The probability of adding a feature to store students data in rom memory in the next version.

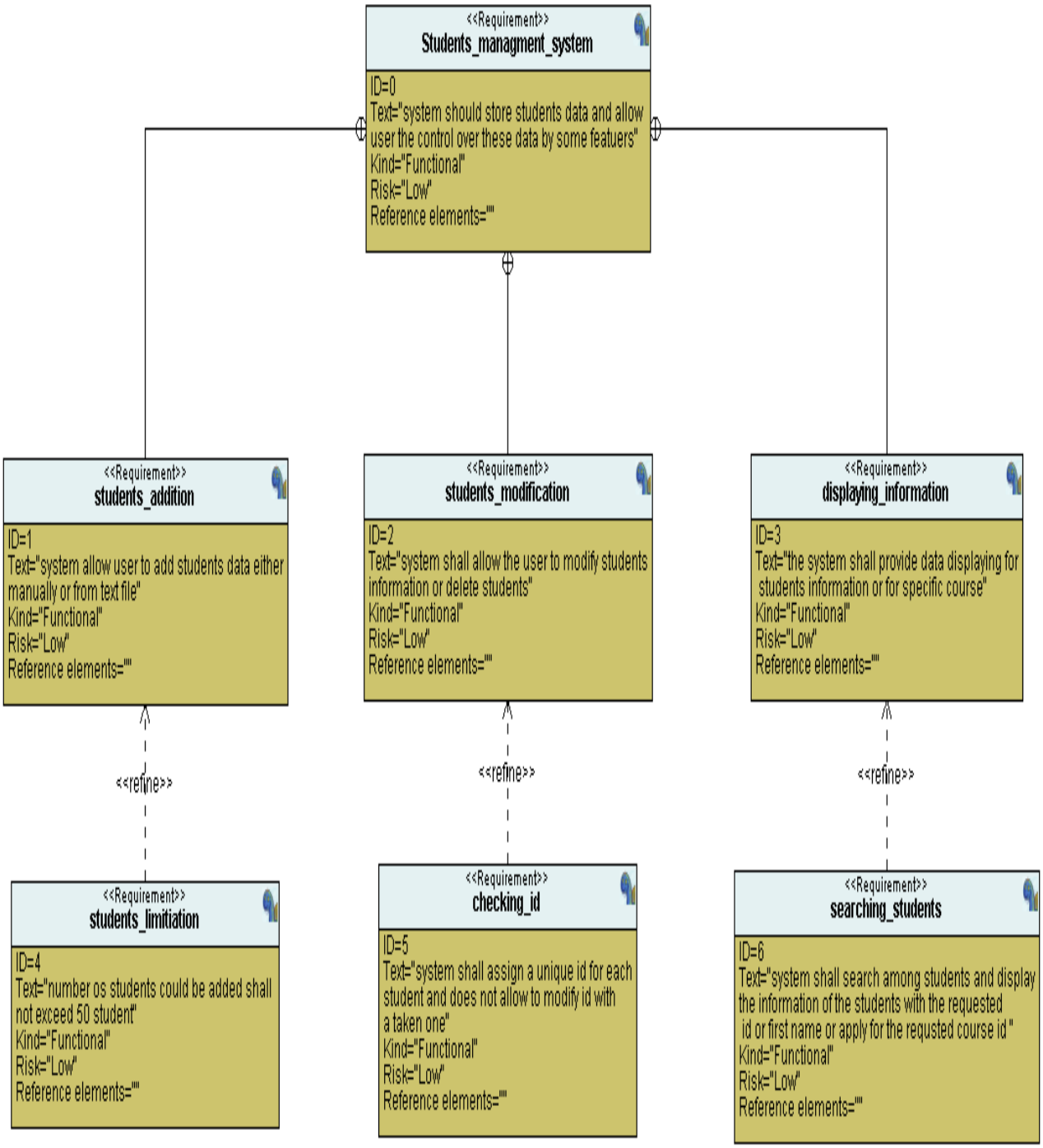
2. Method

- Software developing life cycle & software testing life cycle

The (SDLC) & (STLC) will be approached according to the V-Model.



3. Requirement Diagram



4. Space Exploration

The system will be run on either pc or laptop with these minimum standards :-

- Standards

CPU : core i3 7th generation.

RAM : 4GB.

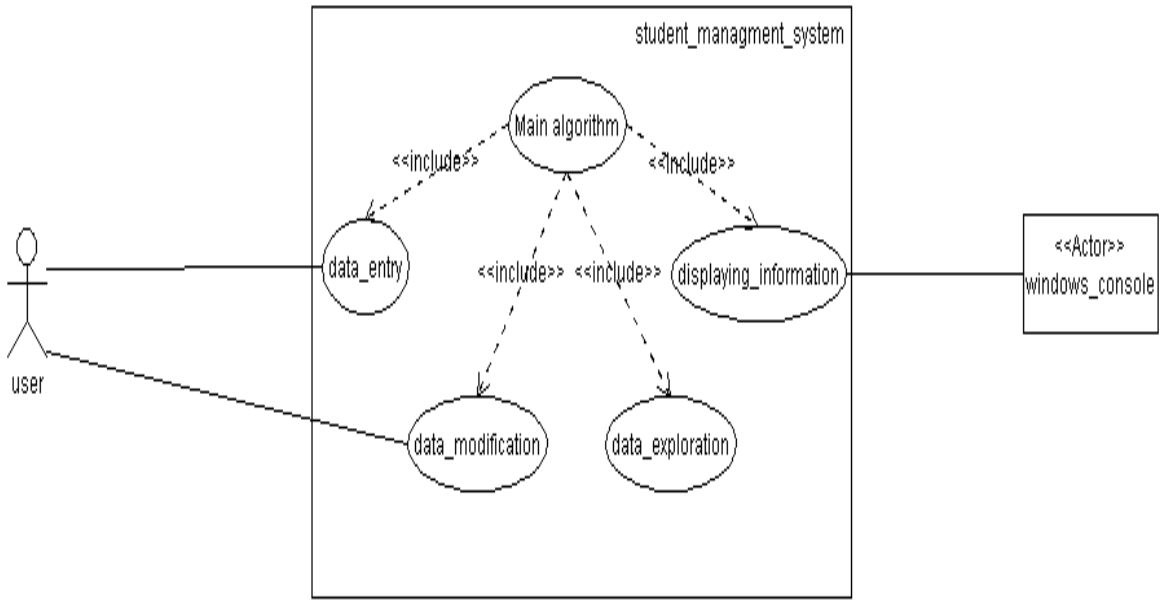
Compiler : gcc

OS : windows 10

5. System Analysis

- Use Case Diagram

pr



6. C codes

- Students.h

```c
/*
 * students.h
 *
 *  Created on: 23/10/2023
 *      Author: ahmed nabil
 */

#ifndef STUDENTS_H_
#define STUDENTS_H_
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

struct sinfo
{
    unsigned int roll_number;
    char first_name[50];
    char last_name[50];
    float gpa;
    unsigned int cid[5];
    }st[50];

     enum
    {
        fifo_full,
        fifo_empty,
        fifo_not_empty,
        fifo_null
    }fifo_status_t;

 struct Fifo
{
        struct sinfo * base;
        struct sinfo * head;
        unsigned char count;
        unsigned char length;
        }student_queue;


FILE *fpointer;

// function to initialize Fifo and assign it array of students struct
void queue_init();

// function to add students details from text file
void add_student_file();

// function to add students details manually by the user
void add_student_manually();

// function to find students details by student's roll number
void find_rl();

// function to find students details by student's first name
void find_fn();

// function to find students details by course id they are applied in
void find_c();

// function to find total number of students and number of students can be added to the list
void tot_s();

// function to delete student details by student's roll number
void del_s();

// function to update students details by student's roll number
void up_s();

// function to display all students informations in the list
void show_s();

#endif /* STUDENTS_H_ */
```

- Add student from file

```c
void add_student_file()
{
    //check if queue is full or not
    if(student_queue.length == student_queue.count)
    {
        printf("[Error] : list is full\n");
        return;
    }

    //open file
    fpointer = fopen("students.txt","r");

    // Check if the file exist ]
    if(fpointer == NULL)
    {
        printf("\n [ERROR] student.txt file not found. \n");
        fifo_status_t = fifo_null;
        return;
    }

    // Reading data until the end of the file
    ll: while(! feof(fpointer) )
    {
        if(fifo_status_t == fifo_empty)                                 //adding 1st element
        {
            fscanf(fpointer,"%d",&student_queue.head->roll_number);
            fscanf(fpointer,"%s",student_queue.head->first_name);
            fscanf(fpointer,"%s",student_queue.head->last_name);
            fscanf(fpointer,"%f",&student_queue.head->gpa);
            for(i=0; i<5; i++)
            {
                fscanf(fpointer,"%d",&student_queue.head->cid[i]);

            }
            printf("[INFO] roll number %d is added successfully\n",student_queue.head->roll_number);
            fifo_status_t = fifo_not_empty;
            student_queue.count ++;
            student_queue.head ++;
        }
        fscanf(fpointer,"%d",&student_queue.head->roll_number);
        for(i=0; i<student_queue.count; i++)
        {
            //checking if roll number is taken before
            if(student_queue.head->roll_number == (student_queue.base + i)->roll_number)
            {
                printf("[Error] roll number %d is already taken\n",student_queue.head->roll_number);
                fscanf(fpointer, "%*[^\n]");                    //ignore the rest of line
                goto ll;
            }
            fscanf(fpointer,"%s",student_queue.head->first_name);
            fscanf(fpointer,"%s",student_queue.head->last_name);
            fscanf(fpointer,"%f",&student_queue.head->gpa);
            fscanf(fpointer,"%f",&student_queue.head->gpa);
            for(i=0; i<5; i++)
            {
                fscanf(fpointer,"%d",&student_queue.head->cid[i]);

            }
            printf("[INFO] roll number %d is added successfully\n",student_queue.head->roll_number);
            fifo_status_t = fifo_not_empty;
            student_queue.count ++;
            student_queue.head ++;
            if(student_queue.count == student_queue.length)
            {
                fifo_status_t = fifo_full;
                student_queue.head --;


            }
        }

    }
    //close the file
    fclose(fpointer);
    printf("[INFO] total number of students : %d\n",student_queue.count);
    printf("[INFO] you can add up to %d students\n",50);
    printf("[INFO] you can add more %d students\n",50-student_queue.count);
    printf("-----------------------------------------\n");

}
```

- Add student from manually

```
101
102 void add_student_manually()
103 {
104     //check if queue is full or not
105     if(student_queue.length == student_queue.count)
106     {
107         printf("[Error] : list is full\n");
108         return;
109     }
110     if(fifo_status_t == fifo_empty)                                 //adding 1st element
111     {
112         printf("Enter student roll number : ");
113         fflush(stdin); fflush(stdout);
114         scanf("%d",&student_queue.head->roll_number);
115         printf("Enter student first name : ");
116         fflush(stdin); fflush(stdout);
117         scanf("%s",student_queue.head->first_name);
118         printf("Enter student last name : ");
119         fflush(stdin); fflush(stdout);
120         scanf("%s",student_queue.head->last_name);
121         printf("Enter student GPA : ");
122         fflush(stdin); fflush(stdout);
123         scanf("%f",&student_queue.head->gpa);
124         printf("Enter students courses id : ");
125         fflush(stdin); fflush(stdout);
126         for(i=0; i<5; i++)
127         {
128             scanf("%d",&student_queue.head->cid[i]);
129
130         }
131         printf("[INFO] student of roll number : %d is added successfully\n",student_queue.head->roll_number);
132         printf("----------------------------------------\n");
133         fifo_status_t = fifo_not_empty;
134         student_queue.count ++;
135         student_queue.head ++;
136     }
137     printf("Enter student roll number : ");
138     fflush(stdin); fflush(stdout);
139     scanf("%d",&student_queue.head->roll_number);
140     //checking if roll number is taken before
141     for(i=0; i<student_queue.count; i++)
142     {
143         if(student_queue.head->roll_number == (student_queue.base + i)->roll_number)
144         {
145             printf("[Error] roll number %d is already taken\n",student_queue.head->roll_number);
146             printf("----------------------------------------\n");
147             return;
148         }
149     }
150     printf("Enter student first name : ");
151     fflush(stdin); fflush(stdout);
152     scanf("%s",student_queue.head->first_name);
153     printf("Enter student last name : ");
154     fflush(stdin); fflush(stdout);
155      scanf("%s",student_queue.head->last_name);
156      printf("Enter student GPA : ");
157      fflush(stdin); fflush(stdout);
158      scanf("%f",&student_queue.head->gpa);
159      printf("Enter the course id for each course :- \n");
160      for(i=0; i<5; i++)
161      {
162          printf("for course %d id : ",i+1);
163          fflush(stdin); fflush(stdout);
164          scanf("%d",&student_queue.head->cid[i]);
165
166      }
167      printf("[INFO] student of roll number : %d is added successfully\n",student_queue.head->roll_number);
168      printf("----------------------------------------\n");
169      fifo_status_t = fifo_not_empty;
170      student_queue.count ++;
171      student_queue.head ++;
172      printf("[INFO] total number of students : %d\n",student_queue.count);
173      printf("[INFO] you can add up to %d students\n",50);
174      printf("[INFO] you can add more %d students\n",50-student_queue.count);
175      printf("----------------------------------------\n");
176
177      if(student_queue.count == student_queue.length)
178      {
179          fifo_status_t = fifo_full;
180          student_queue.head --;
181      }
182
183
184 }
```

- Find student by roll number

```c
void find_rl()
{
    if(fifo_status_t == fifo_empty)         // check if list is empty
    {
        printf("the list is empty there is no informations to show.\n");
        printf("---------------------------------------\n");
        return;
    }
    int roll,j;
    char flag =0;
    printf("Enter the roll number of student u want to view : ");
    fflush(stdin); fflush(stdout);
    scanf("%d",&roll);
    for(i=0; i<student_queue.count; i++)
    {
        if( (student_queue.base + i)->roll_number == roll)
        {
            printf("for student of roll number : %d\n",(student_queue.base + i)->roll_number);
            printf("student first name : %s\n",(student_queue.base + i)->first_name);
            printf("student last name : %s\n",(student_queue.base + i)->last_name);
            printf("student GPA : %.2f\n",(student_queue.base + i)->gpa);
            printf("students courses id : ");
            for(j=0; j<5; j++)
            {
                printf("%d ",(student_queue.base + i)->cid[j]);
            }
            flag =1;
            printf("\n");
            break;
        }
    }
    if(flag ==0 )
        printf("[Error] the roll number u Entered does not exist.\n");
    printf("---------------------------------------\n");

}
```

- Find student by First name

```
223  void find_fn()
224  {
225      if(fifo_status_t == fifo_empty)         // check if list is empty
226      {
227          printf("the list is empty there is no informations to show.\n");
228          return;
229      }
230      int j;
231      char flag =0;
232      char name[50];
233      printf("Enter the first name of student u want to view : ");
234      fflush(stdin); fflush(stdout);
235      scanf("%s",name);
236      for(i=0; i<student_queue.count; i++)
237      {
238          if( ! stricmp( (student_queue.base + i)->first_name , name) )
239          {
240              printf("for student of first name : %s\n",(student_queue.base + i)->first_name);
241              printf("student roll number : %d\n",(student_queue.base + i)->roll_number);
242              printf("student first name : %s\n",(student_queue.base + i)->first_name);
243              printf("student last name : %s\n",(student_queue.base + i)->last_name);
244              printf("student GPA : %.2f\n",(student_queue.base + i)->gpa);
245              printf("students courses id : ");
246              for(j=0; j<5; j++)
247              {
248                  printf("%d ",(student_queue.base + i)->cid[j]);
249              }
250              printf("\n");
251              flag =1;
252          }
253      }
254      if(flag ==0 )
255          printf("[Error] the first name u Entered does not exist.\n");
256      printf("---------------------------------------\n");
257  }
258
```

- Find student by Course id

```
259  void find_c()
260  {
261      if(fifo_status_t == fifo_empty)         // check if list is empty
262      {
263          printf("the list is empty there is no informations to show.\n");
264          return;
265      }
266      int course_id,j;
267      char counter =0;
268      char flag =0;
269
270      printf("Enter the course id  : ");
271      fflush(stdin); fflush(stdout);
272      scanf("%d",&course_id);
273      for(i=0; i<student_queue.count; i++)
274      {
275          for(j=0; j<5; j++)
276          {
277              if( (student_queue.base+i)->cid[j] == course_id )
278              {
279                  counter++;
280                  flag = 1;
281                  printf("the student details are :-\n");
282                  printf("student roll number : %d\n",(student_queue.base + i)->roll_number);
283                  printf("student first name : %s\n",(student_queue.base + i)->first_name);
284                  printf("student last name : %s\n",(student_queue.base + i)->last_name);
285                  printf("student GPA : %.2f\n",(student_queue.base + i)->gpa);
286                  printf("---------------------------------------\n");
287              }
288          }
289      }
290      if(flag ==0 )
291          printf("[Error] the course id you Entered does not exist.\n");
292      else
293          printf("[INFO] total number of students in this course = %d\n",counter);
294      printf("---------------------------------------\n");
295
296  }
```

- Find total number of students

```
297
298  void tot_s()
299
300  {
301      if(fifo_status_t == fifo_empty)        // check if list is empty
302      {
303          printf("the list is empty there is no informations to show.\n");
304          return;
305      }
306      printf("[INFO] total number of students : %d\n",student_queue.count);
307      printf("[INFO] you can add up to %d students\n",50);
308      printf("[INFO] you can add more %d students\n",50-student_queue.count);
309      printf("-----------------------------------\n");
310
311
312  }
```

- Delete student

```
314  void del_s()
315  {
316      if(fifo_status_t == fifo_empty)         // check if list is empty
317      {
318          printf("the list is empty there is no informations to show.\n");
319          return;
320      }
321      int roll,j;
322      char flag=0;
323      printf("Enter the roll number of student you want to delete : ");
324      fflush(stdin); fflush(stdout);
325      scanf("%d",&roll);
326      for(i=0; i<student_queue.count; i++)
327      {
328          if (  (student_queue.base + i)->roll_number == roll )
329          {
330              for(j=i; j<=(student_queue.count-i); j++)
331              {
332
333                  if(j == student_queue.length-1)
334                  {
335                      student_queue.base[j].roll_number = 0;
336                      break;
337                  }
338                  student_queue.base[j] = student_queue.base[j+1];
339
340              }
341              student_queue.count--;
342              flag =1;
343          }
344      }
345      if(!flag)
346          printf("[Error] the roll number you Entered does not Exist\n");
347      else
348      {
349          printf("the student of roll number %d is deleted successfully\n",roll);
350          printf("[INFO] total number of students : %d\n",student_queue.count);
351          printf("[INFO] you can add up to %d students\n",50);
352          printf("[INFO] you can add more %d students\n",50-student_queue.count);
353      }
354      printf("----------------------------------------\n");
355
356  }
```

- Update student

```
358  void up_s()
359  {
360      if(fifo_status_t == fifo_empty)          // check if list is empty
361      {
362          printf("the list is empty there is no informations to show.\n");
363          printf("----------------------------------------\n");
364          return;
365      }
366      int roll,options,j,new_roll,z;
367      char flag=0;
368      printf("Enter the roll number to update the Entry : ");
369      fflush(stdin); fflush(stdout);
370      scanf("%d",&roll);
371      for(i=0; i<student_queue.count; i++)
372      {
373          if ( (student_queue.base + i)->roll_number == roll )
374          {
375              flag =1;
376              printf("1. First name : \n");
377              printf("2. last name : \n");
378              printf("3. roll number : \n");
379              printf("4. GPA : \n");
380              printf("5. courses : \n");
381              ll: printf("Enter your choice : ");
382              fflush(stdin); fflush(stdout);
383              scanf("%d",&options);
384              switch(options)
385              {
386              case 1 :
387                  printf("new first name : ");
388                  fflush(stdin); fflush(stdout);
389                  scanf("%s", (student_queue.base+i)->first_name);
390                  break;
391              case 2 :
392                  printf("new first name : ");
393                  fflush(stdin); fflush(stdout);
394                  scanf("%s", (student_queue.base+i)->last_name);
395                  break;
396              case 3 :
397                  printf("Enter new roll number : ");
398                  aa: fflush(stdin); fflush(stdout);
399                  scanf("%d", &new_roll);
400                  for(z=0; z<student_queue.count; z++)
401                  {
402                      if ( (student_queue.base+z)->roll_number == new_roll )
403                      {
404                          printf("[Error] this roll number is already taken\n");
405                          printf("please Enter a valid roll number : ");
406                          goto aa;
407                      }
409                  }
410                  (student_queue.base+i)->roll_number =  new_roll;
411                  break;
412              case 4 :
413                  printf("new GPA : ");
414                  fflush(stdin); fflush(stdout);
415                  scanf("%f", & ((student_queue.base+i)->gpa));
416                  break;
417              case 5 :
418                  for(j=0; j<5; j++)
419                  {
420                      printf("for course %d id : ",j+1);
421                      fflush(stdin); fflush(stdout);
422                      scanf("%d",&student_queue.head->cid[j]);
423                  }
424                  break;
425              default :
426                  printf("[Error] : Enter correct number : ");
427                  goto ll;
428                  break;
429              }
430          }
431
432      }
433      if(flag)
434          printf("[INFO] updated successfully\n");
435      else
436          printf("[ERROR] the roll number u Entered does not exist\n");
437      printf("----------------------------------------\n");
438  }
439
```

- Show all information

```
440   void show_s()
441   {
442       if(fifo_status_t == fifo_empty)         // check if list is empty
443       {
444           printf("the list is empty there is no informations to show.\n");
445           return;
446       }
447       int j;
448       printf("---------------------------------------\n");
449
450       for(i=0; i<student_queue.count; i++)
451       {
452           printf("student roll number : %d\n",(student_queue.base + i)->roll_number);
453           printf("student first name : %s\n",(student_queue.base + i)->first_name);
454           printf("student last name : %s\n",(student_queue.base + i)->last_name);
455           printf("student GPA : %.2f\n",(student_queue.base + i)->gpa);
456           printf("student courses id : ");
457           for(j=0; j<5; j++)
458           {
459               printf("%d ",(student_queue.base+i)->cid[j]);
460           }
461           printf("\n---------------------------------------\n");
462       }
463       printf("[INFO] total number of students : %d\n",student_queue.count);
464       printf("[INFO] you can add up to %d students\n",50);
465       printf("[INFO] you can add more %d students\n",50-student_queue.count);
466       printf("---------------------------------------\n");
467
468   }
469
```