
MASTERING EMBEDDED SYSTEM ONLINE DIPLOMA

First Term

PROJECT 2

ENG. OMAR SHAWKY MOHAMED

<https://www.learn-in-depth.com/online-diploma/omarshawky9@gmail.com>

STUDENT DATABASE

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 Functions to implement

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.

3 IMPLEMENTATION

3.1 main.c

```
/*
 * main.c
 * Created on 20-8-2023
 * Author: Omar Shawky
 */
#include "FIFO.h"

extern element_type Buffer[BUFFSIZE];
extern FIFO_Buf_t student_fifo;

void main(void)
{
    int choice = 1;
    char temp_text [10];
    FIFO_Init(&(student_fifo) , Buffer , BUFFSIZE);

    while(choice)
    {
        DPRINTF("=====Welcome to the program=====\\n");
        DPRINTF("Choose the number corresponding to the required option:\\n");
        DPRINTF("=====\\n");
        DPRINTF("1- Add New Student Manually\\n");
        DPRINTF("2- Add New Student/s From Text File\\n");
        DPRINTF("3- Find Student by Roll Number\\n");
        DPRINTF("4- Find Student by First Name\\n");
        DPRINTF("5- Find Student by Course ID\\n");
        DPRINTF("6- Total Number of Students\\n");
        DPRINTF("7- Delete Student by Roll Number\\n");
        DPRINTF("8- Update Student by Roll Number\\n");
        DPRINTF("9- View All Students Info\\n");
        DPRINTF("0- Exit program\\n");
        DPRINTF("=====\\n");
        DPRINTF("Please Enter your choice: ");
        gets(temp_text);
        choice = atoi(temp_text);
        DPRINTF("=====\\n");
    }
}
```

```
switch (choice)
{
case 0:
    break;
case 1:
    Add_student_manual();
    break;
case 2:
    Add_student_file();
    break;
case 3:
    Find_Student_RollN();
    break;
case 4:
    Find_Student_FName();
    break;
case 5:
    Find_Student_C_ID();
    break;
case 6:
    Student_count();
    break;
case 7:
    DLT_Student_RollN();
    break;
case 8:
    UPDT_Student_RollN();
    break;
case 9:
    View_All();
    break;
default:
    printf("Incorrent entry please try again\n");
    break;
}
printf("=====\n");
```

3.2 FIFO.h

```
/*
 * FIFO.h
 * Created on 20-8-2023
 * Author: Omar Shawky
 */

#ifndef FIFO_H_
#define FIFO_H_

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

#define BUFFSIZE 200

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

//User configurations

typedef struct{

    char        first_name[50] ;           //First Name of the student
    char        second_name[50] ;          //Second Name of the student
    uint32_t    Roll_num;                  //Roll Num of the student
    float       GPA ;                      //GPA of the student
    char        courses[5][50] ;           //Courses registered by the student
}
```

```
//User configurations
```

```
typedef struct{
```

```
    char        first_name[50] ;           //First Name of the student
    char        second_name[50] ;          //Second Name of the student
    uint32_t     Roll_num;                 //Roll Num of the student
    float        GPA ;                     //GPA of the student
    char        courses[5][50] ;           //Courses registered by the student
```

```
} Student_t;
```

```
//select element type (uint8_t , uint16_t , uint32_t .....)
```

```
#define element_type Student_t
```

```
extern element_type Buffer[BUFSIZE];
```

```
//create buffer 1
```

```
#define width1 5
```

```
// type definitions
```

```
typedef struct{
```

```
    element_type * head ;                  //Pointer to the head (last element enqueued)
    element_type * base ;                  //Pointer to the base (start of the buffer)
    element_type * tail ;                  //Pointer to the tail (last element dequeued)
    uint32_t length ;                      //Total length of the buffer
    uint32_t count ;                       //Total length of the buffer
```

```
} FIFO_Buf_t;
```

```
typedef enum{
```

```
    FIFO_no_error,                        //No error return from the calling the api
    FIFO_full,                             //Buffer is full
    FIFO_empty,                            //Buffer is empty
    FIFO_Null                              //Buffer is destroyed or not found
```

```
} FIFO_Status_t;
```

```
// APIS
```

```
/*Basic APIS*/
```

```
FIFO_Status_t FIFO_Enqueue_item ( FIFO_Buf_t * fbuf , element_type item );           //Add element to the buffer (Arguements: The buffer , the it
FIFO_Status_t FIFO_Dequeue_item ( FIFO_Buf_t * fbuf , element_type item );           //Get element to the buffer (Arguements: The buffer , the it
FIFO_Status_t FIFO_Init         ( FIFO_Buf_t * fbuf , element_type buff[] , unsigned int size); //Initialize the buffer
FIFO_Status_t FIFO_IS_FULL      ( FIFO_Buf_t * fbuf );                               //check if buffer is full
FIFO_Status_t FIFO_IS_EMPTY     ( FIFO_Buf_t * fbuf );                               //check if buffer is empty
```

```
/*Extra APIS*/
```

```
element_type FIFO_Read_item      ( FIFO_Buf_t * fbuf );                             //Read the last item in the buffer
void FIFO_Print                  ( FIFO_Buf_t * fbuf );                             //Read the last item in the buffer
element_type collect_Data        (void);
void Add_student_manual          (void);
void Add_student_file            (void);
void Student_count               (void);
void Find_Student_C_ID           (void);
void Find_Student_FName          (void);
void Find_Student_RollN          (void);
void UPDT_Student_RollN          (void);
void DLT_Student_RollN           (void);
void View_ALL                    (void);
```

```
#endif /* FIFO_H_ */
```

3.3 FIFO.c

```
4  /*
5   *   LIFO.c
6   *   Created on 20-8-2023
7   *   Author: Omar Shawky
8   */
9
10 #include "FIFO.h"
11
12 #ifndef NULL
13 #define NULL 0
14 #endif
15
16 element_type Buffer[BUFFSIZE] ;           //Static allocation of 5*(sizeof(unsigned int)) = 20
17 byte
18 FIFO_Buf_t student_fifo;
19
20 // APIS
21 /*Basic APIS*/
22 FIFO_Status_t FIFO_enqueue_item ( FIFO_Buf_t * fbuf , element_type item )           //Add
23 element to the buffer (Arguments: The buffer , the item to be added)
24 {
25     //Check if the FIFO is valid
26     if ( ! fbuf->head || ! fbuf->base || ! fbuf->tail ){
27         return FIFO_Null;
28     }
29     //Check if the LIFO is full
30     if ( fbuf->count == fbuf->length ){
31         return FIFO_full;
32     }
33     //queue the value (Cicular Queue)
34     *(fbuf->head) = item;
35     fbuf->count ++;
36
37     if(fbuf->head == (fbuf->base + fbuf->length))
38     {
39         fbuf->head = fbuf->base;
40     }
41     else{
42         fbuf->head ++;
43     }
44
45     return FIFO_no_error ;
46 }
```

```

46 FIFO_Status_t FIFO_dequeue_item ( FIFO_Buf_t * fbuf , element_type item ) //Get
   element to the buffer (Arguments: The buffer , the item to be gotten)
47 {
48     //Check if the LIFO is valid
49     if ( ! fbuf->head || ! fbuf->base || ! fbuf->tail ){
50         return FIFO_Null;
51     }
52     //Check if the LIFO is empty
53     if ( fbuf->count == 0 ){
54         return FIFO_empty;
55     }
56     //dequeue the value (Cicular Queue)
57     (item) = *(fbuf->tail);
58     fbuf->count --;
59     if(fbuf->tail == (fbuf->base + fbuf->length))
60     {
61         fbuf->tail = fbuf->base;
62     }
63     else{
64         fbuf->tail ++;
65     }
66
67     return FIFO_no_error ;
68 }
69
70 FIFO_Status_t FIFO_Init ( FIFO_Buf_t * fbuf , element_type buff[] , unsigned int
   size) //Initialize the buffer
71 {
72     if( buff == NULL ) {
73         return FIFO_Null;
74     }
75
76     fbuf->base = buff;
77     fbuf->head = buff;
78     fbuf->tail = buff;
79     fbuf->length = size;
80     fbuf->count = 0;
81
82     return FIFO_no_error ;
83 }
84
85 FIFO_Status_t FIFO_IS_FULL ( FIFO_Buf_t * fbuf
   ) //check if buffer is full
86 {
87     if ( ! fbuf->head || ! fbuf->base || ! fbuf->tail ){

```



```

88     return FIFO_Null;
89 }
90 //Check if the LIFO is full
91 if ( fbuf->count == fbuf->length ){
92     return FIFO_full;
93 }
94 else{
95     return FIFO_no_error;
96 }
97 }
98
99 FIFO_Status_t  FIFO_IS_EMPTY      ( FIFO_Buf_t * fbuf
    )                                     //check if buffer is empty
100{
101     if ( ! fbuf->head || ! fbuf->base || ! fbuf->tail ){
102         return FIFO_Null;
103     }
104     //Check if the LIFO is empty
105     if ( fbuf->count == 0 ){
106         return FIFO_empty;
107     }
108     else{
109         return FIFO_no_error;
110     }
111 }
112
113 /*Extra APIS*/
114 element_type  FIFO_Read_item      ( FIFO_Buf_t * fbuf )           //Read
    the last item in the buffer
115{
116     //Check if the LIFO is valid
117     if ( ! fbuf->head || ! fbuf->base ){
118         //return FIFO_Null;
119     }
120     //Check if the LIFO is empty
121     if ( fbuf->count == 0 ){
122         //return FIFO_empty;
123     }
124
125     return (element_type)*(fbuf->head) ;
126 }
127
128 void          FIFO_Print          ( FIFO_Buf_t * fbuf )           //Read
    the last item in the buffer
129{

```

```

130     element_type * temp ;
131     int i ;
132     //Check if the FIFO is empty
133     if ( fbuf->count == 0 ){
134         printf("Fifo is empty \n");
135     }
136     else {
137         temp = fbuf->tail ;
138         printf("=====fifo print=====\\n");
139         for(i=0;i<(fbuf->count);i++)
140         {
141             printf("%d \\n",*temp);
142             if( temp == (fbuf->base + fbuf->length))
143             {
144                 temp = fbuf->base;
145             }
146             else{
147                 temp++;
148             }
149         }
150         printf("=====fifo end=====\\n");
151     }
152 }
153
154 void Add_student_manual(void)
155 {
156     Student_t s ;
157     char temp_text [40];
158     DPRINTF("Enter the student's first Name:\\n");
159     gets(temp_text);
160     //DPRINTF("%s\\n",temp_text);
161     strcpy(s.first_name , temp_text);
162
163     DPRINTF("Enter the student's last Name:\\n");
164     gets(temp_text);
165     strcpy(s.second_name , temp_text);
166
167     DPRINTF("Enter the student's Roll number:\\n");
168     gets(temp_text);
169     s.Roll_num = atoi(temp_text);
170
171     DPRINTF("Enter the student's GPA:\\n");
172     gets(temp_text);
173     s.GPA = atof(temp_text);
174

```

```

175     for(int i=0 ; i<5 ; i++)
176     {
177         DPRINTF("Enter the student's registered course number %d:\n",i+1);
178         gets(temp_text);
179         strcpy(s.courses[i] , temp_text);
180
181     }
182     DPRINTF("#####Data filled successfully#####\n");
183     FIFO_enqueue_item ( &student_fifo ,  s );
184}
185
186void Add_student_file()
187{
188
189     FILE *fp;
190     fp = fopen("StudentFile.txt", "r");
191
192     if(fp == NULL){
193         DPRINTF("\n[ERROR] Failed to open file");
194         return;
195     }
196
197     char line[100];
198     char *token;
199
200     while(fgets(line, sizeof(line), fp)){
201
202         int counter = 1;
203         int course = 0;
204         Student_t new_s;
205         token = strtok(line, " ");
206
207         while(token != NULL){
208             switch(counter){
209                 case 1:
210                     new_s.Roll_num = atoi(token);
211                     break;
212                 case 2:
213                     strcpy(new_s.first_name, token);
214                     break;
215                 case 3:
216                     strcpy(new_s.second_name, token);
217                     break;
218                 case 4:
219                     new_s.GPA = atof(token);

```

```

220         break;
221     case 5:
222         strcpy(new_s.courses[course],token);
223         course++;
224         counter--;
225         break;
226     }
227     token = strtok(NULL, " ");
228     counter++;
229 }
230 *(student_fifo.head) = new_s;
231 student_fifo.count++;
232
233 if(student_fifo.head == (student_fifo.base + (student_fifo.length)))
234     student_fifo.head = student_fifo.base;
235 else
236     student_fifo.head++;
237 }
238 fclose(fp);
239 DPRINTF("\n[INFO] Student Info added successfully\n");
240 }
241
242 void Student_count(void)
243 {
244     DPRINTF("DataBase size is %d students\n",student_fifo.count);
245 }
246
247 void View_All(void)
248 {
249     int size = student_fifo.count;
250     for(int i = 0 ; i<size;i++)
251     {
252         DPRINTF("the student's first Name:\n");
253         DPRINTF("%s\n",Buffer[i].first_name);
254         DPRINTF("the student's second Name:\n");
255         DPRINTF("%s\n",Buffer[i].second_name);
256         DPRINTF("the student's Roll number:\n");
257         DPRINTF("%d\n",Buffer[i].Roll_num);
258         DPRINTF("the student's GPA:\n");
259         DPRINTF("%.2lf\n",Buffer[i].GPA);
260         DPRINTF("the student's registered courses are:");
261         for (int j = 0 ; j<5; j++)
262         {
263             if(j==4){ DPRINTF("%s\n",Buffer[i].courses[j]); }
264             else{ DPRINTF("%s,",Buffer[i].courses[j]); }

```

```

265     }
266     DPRINTF("=====\n");
267 }
268}
269
270void Find_Student_FName(void)
271{
272     uint32_t size = student_fifo.count;
273     char temp_text [40];
274     DPRINTF("SEARCH : Enter the student's first Name:\n");
275     gets(temp_text);
276     DPRINTF("=====\n");
277     char foundflag = 0;
278
279     for(int i = 0 ; i<size;i++)
280     {
281         if(strcmp(temp_text,Buffer[i].first_name)==0){
282             DPRINTF("the student's first Name:\n");
283             DPRINTF("%s\n",Buffer[i].first_name);
284             DPRINTF("the student's second Name:\n");
285             DPRINTF("%s\n",Buffer[i].second_name);
286             DPRINTF("the student's Roll number:\n");
287             DPRINTF("%d\n",Buffer[i].Roll_num);
288             DPRINTF("the student's GPA:\n");
289             DPRINTF("%.2lf\n",Buffer[i].GPA);
290             DPRINTF("the student's registered courses are:");
291             for (int j = 0 ; j<5; j++)
292             {
293                 if(j==4){ DPRINTF("%s\n",Buffer[i].courses[j]); }
294                 else{ DPRINTF("%s,",Buffer[i].courses[j]); }
295             }
296             foundflag = 1;
297             DPRINTF("=====\n");
298         }
299     }
300     if(foundflag == 0) {DPRINTF("Couldn't find a match\n");}
301}
302
303void Find_Student_RollN(void)
304{
305     uint32_t size = student_fifo.count;
306     char temp_text [40];
307     DPRINTF("SEARCH : Enter the student's Roll Number:\n");
308     gets(temp_text);
309     uint32_t ID = atoi(temp_text);

```

```

310     DPRINTF("=====\n");
311     char foundflag = 0;
312
313     for(int i = 0 ; i<size;i++)
314     {
315         if(ID == Buffer[i].Roll_num){
316             DPRINTF("the student's first Name:\n");
317             DPRINTF("%s\n",Buffer[i].first_name);
318             DPRINTF("the student's second Name:\n");
319             DPRINTF("%s\n",Buffer[i].second_name);
320             DPRINTF("the student's Roll number:\n");
321             DPRINTF("%d\n",Buffer[i].Roll_num);
322             DPRINTF("the student's GPA:\n");
323             DPRINTF("%.2lf\n",Buffer[i].GPA);
324             DPRINTF("the student's registered courses are:");
325             for (int j = 0 ; j<5; j++)
326             {
327                 if(j==4){ DPRINTF("%s\n",Buffer[i].courses[j]); }
328                 else{ DPRINTF("%s,",Buffer[i].courses[j]); }
329             }
330             foundflag = 1;
331             DPRINTF("=====\n");
332         }
333     }
334     if(foundflag == 0) {DPRINTF("Couldn't find a match\n");}
335 }
336
337 void Find_Student_C_ID(void)
338 {
339     uint32_t size = student_fifo.count;
340     char temp_text [40];
341     DPRINTF("SEARCH : Enter the student's Course:\n");
342     gets(temp_text);
343     DPRINTF("=====\n");
344     char foundflag = 0;
345
346     for(int i = 0 ; i<size;i++)
347     {
348         for(int z = 0 ; z < 5 ; z++){
349             if(strcmp(temp_text,Buffer[i].courses[z]) == 0)
350             {
351                 DPRINTF("the student's first Name:\n");
352                 DPRINTF("%s\n",Buffer[i].first_name);
353                 DPRINTF("the student's second Name:\n");
354                 DPRINTF("%s\n",Buffer[i].second_name);

```

```

355         DPRINTF("the student's Roll number:\n");
356         DPRINTF("%d\n",Buffer[i].Roll_num);
357         DPRINTF("the student's GPA:\n");
358         DPRINTF("%.2lf\n",Buffer[i].GPA);
359         DPRINTF("the student's registered courses are:");
360         for (int j = 0 ; j<5; j++)
361         {
362             if(j==4){ DPRINTF("%s\n",Buffer[i].courses[j]); }
363             else{ DPRINTF("%s,",Buffer[i].courses[j]); }
364         }
365         foundflag = 1;
366         DPRINTF("=====\n");
367     }
368 }
369 }
370 if(foundflag == 0) {DPRINTF("Couldn't find a match\n");}
371}
372
373void UPDT_Student_RollN(void)
374{
375     uint32_t size = student_fifo.count;
376     char temp_text [40];
377     DPRINTF("SEARCH : Enter the student's Roll Number:\n");
378     gets(temp_text);
379     uint32_t ID = atoi(temp_text);
380     DPRINTF("=====\n");
381     char foundflag = 0;
382
383     for(int i = 0 ; i<size;i++)
384     {
385         if(ID == Buffer[i].Roll_num){
386             DPRINTF("the student's first Name:\n");
387             DPRINTF("%s\n",Buffer[i].first_name);
388             DPRINTF("the student's second Name:\n");
389             DPRINTF("%s\n",Buffer[i].second_name);
390             DPRINTF("the student's Roll number:\n");
391             DPRINTF("%d\n",Buffer[i].Roll_num);
392             DPRINTF("the student's GPA:\n");
393             DPRINTF("%.2lf\n",Buffer[i].GPA);
394             DPRINTF("the student's registered courses are:");
395             for (int j = 0 ; j<5; j++)
396             {
397                 if(j==4){ DPRINTF("%s\n",Buffer[i].courses[j]); }
398                 else{ DPRINTF("%s,",Buffer[i].courses[j]); }
399             }

```

```

400     foundflag = 1;
401     DPRINTF("=====\n");
402     DPRINTF("UPDATING INFO PLEASE FILL THE FOLLOWING:\n");
403     DPRINTF("=====\n");
404     DPRINTF("Enter the student's first Name:\n");
405     gets(temp_text);
406     //DPRINTF("%s\n",temp_text);
407     strcpy(Buffer[i].first_name , temp_text);
408     DPRINTF("Enter the student's last Name:\n");
409     gets(temp_text);
410     strcpy(Buffer[i].second_name , temp_text);
411
412     DPRINTF("Enter the student's Roll number:\n");
413     gets(temp_text);
414     Buffer[i].Roll_num = atoi(temp_text);
415
416     DPRINTF("Enter the student's GPA:\n");
417     gets(temp_text);
418     Buffer[i].GPA = atof(temp_text);
419     for(int i=0 ; i<5 ; i++)
420     {
421         DPRINTF("Enter the student's registered course number %d:\n",i+1);
422         gets(temp_text);
423         strcpy(Buffer[i].courses[i] , temp_text);
424
425     }
426
427 }
428 }
429 if(foundflag == 0) {DPRINTF("Couldn't find a match\n");}
430}
431
432void DLT_Student_RollN(void)
433{
434     char temp_text [40];
435     DPRINTF("SEARCH : Enter the student's Roll Number:\n");
436     gets(temp_text);
437     uint32_t ID = atoi(temp_text);
438
439     uint32_t size = student_fifo.count;
440
441     Student_t * s ;
442     Student_t * curr_s ;
443     Student_t * next_s ;
444     s = student_fifo.tail;

```



```

445     DPRINTF("=====\n");
446     char foundflag = 0;
447
448     for(int i = 0 ; i<size;i++)
449     {
450         if(ID == Buffer[i].Roll_num){
451             for(int j = i ; j<size;j++)
452             {
453                 curr_s = s;
454                 next_s = ++s;
455                 *curr_s = *next_s;
456             }
457             student_fifo.count-- ;
458             if(student_fifo.head == student_fifo.base)
459             {
460                 student_fifo.head = student_fifo.base + student_fifo.length;
461             }
462             else
463             {
464                 student_fifo.head--;
465             }
466             foundflag = 1;
467             DPRINTF("DELETED SUCCESSFULLY\n");
468         }
469         s++;
470     }
471     if(foundflag == 0) {DPRINTF("Couldn't find a match\n");}
472 }

```

4 OUTPUT

1)Add Manual

```
=====Welcome to the program=====
Choose the number corresponding to the required option:
=====
1- Add New Student Manually
2- Add New Student/s From Text File
3- Find Student by Roll Number
4- Find Student by First Name
5- Find Student 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
0- Exit program
=====
Please Enter your choice: 1
=====
Enter the student's first Name:
Mohanad
Enter the student's last Name:
Sherif
Enter the student's Roll number:
66
Enter the student's GPA:
3.1
Enter the student's registered course number 1:
Math
Enter the student's registered course number 2:
Machines
Enter the student's registered course number 3:
Production
```

2)Add from text

```
=====
=====Welcome to the program=====
Choose the number corresponding to the required option:
=====
1- Add New Student Manually
2- Add New Student/s From Text File
3- Find Student by Roll Number
4- Find Student by First Name
5- Find Student 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
0- Exit program
=====
Please Enter your choice: 2
=====

[INFO] Student Info added successfully
```

```
the student's first Name:
Mohanad
the student's second Name:
Sherif
the student's Roll number:
66
the student's GPA:
3.10
the student's registered courses are:[Math,Machines,Production,Manufacturing,English]
=====
the student's first Name:
Omar
the student's second Name:
Shawky
the student's Roll number:
1
the student's GPA:
3.80
the student's registered courses are:[Physiology,Ethics,Math,Chemistry,Design]
=====
the student's first Name:
Sameh
the student's second Name:
Saed
the student's Roll number:
2
the student's GPA:
2.50
the student's registered courses are:[Machines,Drawing,Electronics,Physics,Circuits
```

```
Mohamed
the student's second Name:
Ahmed
the student's Roll number:
3
the student's GPA:
3.30
the student's registered courses are:[Economics,Finance,Statistics,Math,English
]
=====
the student's first Name:
Seif
the student's second Name:
Ashraf
the student's Roll number:
4
the student's GPA:
2.80
the student's registered courses are:[Dynamics,Biology,Statics,Sports,Chemistry
]
=====
the student's first Name:
Karam
the student's second Name:
Islam
the student's Roll number:
5
the student's GPA:
3.20
the student's registered courses are:[Physics,Math,Chemistry,Arabic,English
]
=====
```

3)Find by Roll

```
=====Welcome to the program=====
Choose the number corresponding to the required option:
=====
1- Add New Student Manually
2- Add New Student/s From Text File
3- Find Student by Roll Number
4- Find Student by First Name
5- Find Student 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
0- Exit program
=====
Please Enter your choice: 3
=====
SEARCH : Enter the student's Roll Number:
66
=====
the student's first Name:
Mohanad
the student's second Name:
Sherif
the student's Roll number:
66
the student's GPA:
3.10
the student's registered courses are:[Math,Machines,Production,Manufacturing,English]
=====
=====
```

4)Find by First Name

```
2- Add New Student/s From Text File
3- Find Student by Roll Number
4- Find Student by First Name
5- Find Student 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
0- Exit program
=====
Please Enter your choice: 4
=====
SEARCH : Enter the student's first Name:
Omar
=====
the student's first Name:
Omar
the student's second Name:
Shawky
the student's Roll number:
1
the student's GPA:
3.80
the student's registered courses are:[Physiology,Ethics,Math,Chemistry,Design]
=====
```

5)Find by Course

```
7- Delete Student by Roll Number
8- Update Student by Roll Number
9- View All Students Info
0- Exit program
=====
Please Enter your choice: 5
=====
SEARCH : Enter the student's Course:
Machines
=====
the student's first Name:
Mohanad
the student's second Name:
Sherif
the student's Roll number:
66
the student's GPA:
3.10
the student's registered courses are:[Math,Machines,Production,Manufacturing,English]
=====
the student's first Name:
Sameh
the student's second Name:
Saed
the student's Roll number:
2
the student's GPA:
2.50
the student's registered courses are:[Machines,Drawing,Electronics,Physics,Circuits
]
=====
```

6)Print total number of students in the database

```
=====Welcome to the program=====
Choose the number corresponding to the required option:
=====
1- Add New Student Manually
2- Add New Student/s From Text File
3- Find Student by Roll Number
4- Find Student by First Name
5- Find Student 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
0- Exit program
=====
Please Enter your choice: 6
=====
DataBase size is 6 students
=====
```

7)Delete student using roll number

```
=====Welcome to the program=====
Choose the number corresponding to the required option:
=====
1- Add New Student Manually
2- Add New Student/s From Text File
3- Find Student by Roll Number
4- Find Student by First Name
5- Find Student 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
0- Exit program
=====
Please Enter your choice: 7
=====
SEARCH : Enter the student's Roll Number:
66
=====
DELETED SUCCESSFULLY
=====
```

```
=====
Please Enter your choice: 9
=====
the student's first Name:
Omar
the student's second Name:
Shawky
the student's Roll number:
1
the student's GPA:
3.80
the student's registered courses are:[Physiology,Ethics,Math,Chemistry,Design]
=====
the student's first Name:
Sameh
the student's second Name:
Saed
the student's Roll number:
2
the student's GPA:
2.50
the student's registered courses are:[Machines,Drawing,Electronics,Physics,Circuits
]
=====
the student's first Name:
Mohamed
the student's second Name:
Ahmed
the student's Roll number:
3
the student's GPA:
```

```
the student's GPA:
3.30
the student's registered courses are:[Economics,Finance,Statistics,Math,English
]
=====
the student's first Name:
Seif
the student's second Name:
Ashraf
the student's Roll number:
4
the student's GPA:
2.80
the student's registered courses are:[Dynamics,Biology,Statics,Sports,Chemistry
]
=====
the student's first Name:
Karam
the student's second Name:
Islam
the student's Roll number:
5
the student's GPA:
3.20
the student's registered courses are:[Physics,Math,Chemistry,Arabic,English
]
=====
=====
```


8)Update student using roll number

```
SEARCH : Enter the student's Roll Number:
2
=====
the student's first Name:
Sameh
the student's second Name:
Saed
the student's Roll number:
2
the student's GPA:
2.50
the student's registered courses are:[Machines,Drawing,Electronics,Physics,Circuits
]
=====
UPDATING INFO PLEASE FILL THE FOLLOWING:
=====
Enter the student's first Name:
Salem
Enter the student's last Name:
Samir
Enter the student's Roll number:
2
Enter the student's GPA:
2.60
Enter the student's registered course number 1:
Machines
Enter the student's registered course number 2:
Drawing
Enter the student's registered course number 3:
Circuits
Enter the student's registered course number 4:
Math
```

```
the student's Roll number:
1
the student's GPA:
3.80
the student's registered courses are:[Machines,Ethics,Math,Chemistry,Design]
=====
the student's first Name:
Salem
the student's second Name:
Samir
the student's Roll number:
2
the student's GPA:
2.60
the student's registered courses are:[Machines,Drawing,Electronics,Physics,Circuits
]
=====
the student's first Name:
Mohamed
the student's second Name:
Ahmed
the student's Roll number:
3
the student's GPA:
3.30
the student's registered courses are:[Economics,Finance,Circuits,Math,English
]
=====
the student's first Name:
Seif
the student's second Name:
Ashraf
```

9)View the entire Data base

```
=====Welcome to the program=====
Choose the number corresponding to the required option:
=====
1- Add New Student Manually
2- Add New Student/s From Text File
3- Find Student by Roll Number
4- Find Student by First Name
5- Find Student 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
0- Exit program
=====
Please Enter your choice: 9
=====
the student's first Name:
Omar
the student's second Name:
Shawky
the student's Roll number:
1
the student's GPA:
3.80
the student's registered courses are:[Machines,Ethics,Math,Chemistry,Design]
=====
the student's first Name:
Salem
the student's second Name:
Samir
the student's Roll number:
```

```
the student's GPA:
2.60
the student's registered courses are:[Machines,Drawing,Electronics,Physics,Circuits
]
```

```
=====
```

```
the student's first Name:
Mohamed
the student's second Name:
Ahmed
the student's Roll number:
3
```

```
the student's GPA:
3.30
the student's registered courses are:[Economics,Finance,Circuits,Math,English
]
```

```
=====
```

```
the student's first Name:
Seif
the student's second Name:
Ashraf
the student's Roll number:
4
```

```
the student's GPA:
2.80
the student's registered courses are:[Dynamics,Biology,Statics,Math,Chemistry
]
```

```
=====
```

```
the student's first Name:
Karam
the student's second Name:
Islam
the student's Roll number:
```

```
=====
```

```
the student's first Name:
Karam
the student's second Name:
Islam
the student's Roll number:
5
```

```
the student's GPA:
3.20
the student's registered courses are:[Physics,Math,Chemistry,Arabic,Physics]
```

```
=====
```