By Alfred Fayez

Mastering Embedded Systems Online Diploma

www.learn-in-depth-store.com

First Term (Final Project 2)

Eng. Alfred Fayez Fahim

My Profile:

https://www.learn-in-depth-store.com/certificate/alfred.f.d646%40gmail.com

By Alfred Fayez

STUDENT MANAGEMENT SYSTEM

Creating FIFO(Queue), Source, And Header Files For A Student Management System Program

Contents

By Alfred Fayez

Table of Figures

By Alfred Fayez

1. Case Study

A simple student database that allows the administrator to:

- Add students from file.

- Add a student Manually.

- Find a student by roll number.

- Find a student by first name.

- Find Student that take a certain course.

- Find the total number of students.

- Delete a student.

- Update student's information.

- Show all the students in the database

2. Implementation

- Add students from file.

```
21  void add_student_file()
22  {
23      dprintf("\n----------------------\n \n");
24      FILE *p = fopen("StudentInfo.txt","r");
25      if (p == NULL){
26          dprintf("[Error] File can't be opened\n");
27      }
28      else{
29          element_type student;
30          while(fscanf(p, "%d %s %s %f %d %d %d %d %d", &student.roll, student.fname, student.lname,
31                  &student.GPA, &student.c_id[0], &student.c_id[1], &student.c_id[2],
32                  &student.c_id[3], &student.c_id[4])!= EOF)
33          {
34              if(fifo_check_roll(&student_fifo, (uint32)student.roll) == roll_available){
35                  fifo_enqueue(&student_fifo,&student);
36                  dprintf("[INFO] Roll number %d saved successfully\n", student.roll)
37              }
38              else{
39                  dprintf("[Error] roll number: %d is used before \n", student.roll);
40              }
41          }
42          total_s();
43      }
44      fclose(p);
45  }
```

Snippet 1 add_student_file fun

By Alfred Fayez

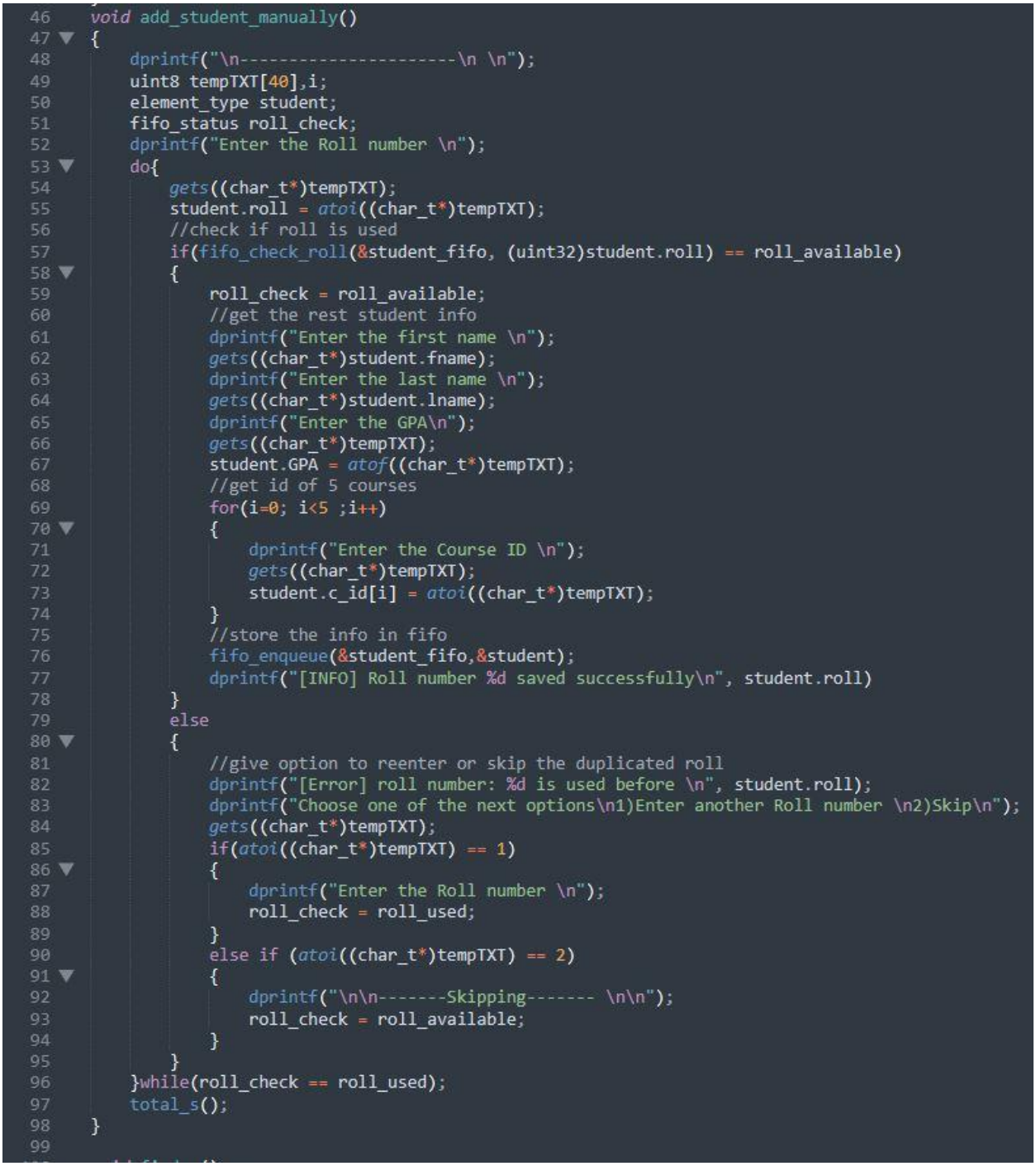
```
21  fifo_status fifo_enqueue(Sfifo_t* fifo, element_type* item)
22  {
23      //check fifo is valid
24      if(!fifo->base || !fifo->head || !fifo->tail)
25          return fifo_null;
26
27      //check fifo is full
28      if (fifo_is_full(fifo) == fifo_full)
29          return fifo_full;
30      *(fifo->head) = *item;
31      fifo->count++;
32      if (fifo->head == (fifo->base + (fifo->length * sizeof(element_type))))
33          fifo->head = fifo->base;
34      else
35          fifo->head++;
36      return fifo_no_error;
37  }
```

FIFO 1 enqueue fun

```
 94  fifo_status fifo_check_roll(Sfifo_t* fifo, uint32 roll)
 95  {
 96      uint32 i;
 97      element_type *temp;
 98      //check fifo is valid
 99      if(!fifo->base || !fifo->head || !fifo->tail)
100          return fifo_null;
101
102
103      //check fifo is full
104      if(fifo->count == 55)
105          return fifo_full;
106
107      else {
108          temp = fifo->tail;
109          for (i=0;i<fifo->count;i++){
110              if(roll == temp->roll)
111                  return roll_used;
112              temp++;
113          }
114      }
115      return roll_available;
116  }
117
```

FIFO 2 check_roll fun

- Add a student manually.



```
void add_student_manually()
{
    dprintf("\n----------------------\n \n");
    uint8 tempTXT[40],i;
    element_type student;
    fifo_status roll_check;
    dprintf("Enter the Roll number \n");
    do{
        gets((char_t*)tempTXT);
        student.roll = atoi((char_t*)tempTXT);
        //check if roll is used
        if(fifo_check_roll(&student_fifo, (uint32)student.roll) == roll_available)
        {
            roll_check = roll_available;
            //get the rest student info
            dprintf("Enter the first name \n");
            gets((char_t*)student.fname);
            dprintf("Enter the last name \n");
            gets((char_t*)student.lname);
            dprintf("Enter the GPA\n");
            gets((char_t*)tempTXT);
            student.GPA = atof((char_t*)tempTXT);
            //get id of 5 courses
            for(i=0; i<5 ;i++)
            {
                dprintf("Enter the Course ID \n");
                gets((char_t*)tempTXT);
                student.c_id[i] = atoi((char_t*)tempTXT);
            }
            //store the info in fifo
            fifo_enqueue(&student_fifo,&student);
            dprintf("[INFO] Roll number %d saved successfully\n", student.roll)
        }
        else
        {
            //give option to reenter or skip the duplicated roll
            dprintf("[Error] roll number: %d is used before \n", student.roll);
            dprintf("Choose one of the next options\n1)Enter another Roll number \n2)Skip\n");
            gets((char_t*)tempTXT);
            if(atoi((char_t*)tempTXT) == 1)
            {
                dprintf("Enter the Roll number \n");
                roll_check = roll_used;
            }
            else if (atoi((char_t*)tempTXT) == 2)
            {
                dprintf("\n\n-------Skipping------- \n\n");
                roll_check = roll_available;
            }
        }
    }while(roll_check == roll_used);
    total_s();
}
```

Snippet 2 add_manually fun

- Find a student by roll number.

```
100  void find_r()
101  {
102      uint8 tempTXT[40],i;
103      dprintf("\n---------------------\n \n");
104      dprintf("Enter the Roll number \n");
105      gets((char_t*)tempTXT);
106      element_type* student = fifo_find_roll(&student_fifo, atoi((char_t*)tempTXT));
107      if(student == NULL)
108      {
109          dprintf("Cannot find the student \n");
110      }
111      else
112      {
113          dprintf("The student's details are: \n");
114          dprintf("The first name: %s\n", student->fname);
115          dprintf("The last name: %s\n", student->lname);
116          dprintf("The student's GPA: %0.2f\n", student->GPA);
117          dprintf("Student's courses: \n");
118          for (i =0 ; i < 5; i++){
119              dprintf("Course %d ID: %d\n", i+1, student->c_id[i]);
120          }
121          dprintf("--------------------------------\n");
122      }
123  }
```

Snippet 3 find by roll number fun

```
118  element_type* fifo_find_roll(Sfifo_t* fifo, uint32 roll)
119  {
120      uint32 i;
121      element_type *temp;
122      //check fifo is valid
123      if(!fifo->base || !fifo->head || !fifo->tail)
124          return (element_type*)NULL;
125
126      //check fifo is empty
127      if(fifo->count == 0)
128          return (element_type*)NULL;
129
130      else {
131          temp = fifo->tail;
132          for (i=0;i<fifo->count;i++){
133              if(roll == temp->roll)
134                  return temp;
135              temp++;
136          }
137      }
138      return 0;
139  }
140
```

FIFO 3 find roll fun

- Find a student by first name.

```
124  void find_fn()
125  {
126      uint8 tempTXT[40],i;
127      dprintf("\n----------------------\n \n");
128      dprintf("Enter the first name \n");
129      gets((char_t*)tempTXT);
130      element_type* student = fifo_find_fname(&student_fifo, (char_t*)tempTXT);
131      if(student == NULL)
132      {
133          dprintf("Cannot find the student \n");
134      }
135      else
136      {
137          dprintf("The student's details are: \n");
138          dprintf("The first name: %s\n", student->fname);
139          dprintf("The last name: %s\n", student->lname);
140          dprintf("The Roll number: %d\n", student->roll);
141          dprintf("The student's GPA: %0.2f\n", student->GPA);
142          dprintf("Student's courses: \n");
143          for (i =0 ; i < 5; i++){
144              dprintf("Course %d ID: %d\n", i+1, student->c_id[i]);
145          }
146          dprintf("--------------------------------\n");
147      }
148  }
```

Snippet 4 find by first name fun

```
141  element_type* fifo_find_fname(Sfifo_t* fifo, char_t* name)
142  {
143      uint32 i;
144      element_type *temp;
145      //check fifo is valid
146      if(!fifo->base || !fifo->head || !fifo->tail)
147          return (element_type*)NULL;
148
149      //check fifo is empty
150      if(fifo->count == 0)
151          return (element_type*)NULL;
152
153      else {
154          temp = fifo->tail;
155          for (i=0;i<fifo->count;i++){
156              if(stricmp(name,(char_t*)temp->fname) == 0)
157                  return temp;
158              temp++;
159          }
160      }
161      return temp;
162  }
163
```

FIFO 4 find by first name fun

By Alfred Fayez

- Find Student that take a certain course.

```
149   void find_c()
150   {
151       uint8 tempTXT[40];
152       dprintf("\n----------------------\n \n");
153       dprintf("Enter the class ID \n");
154       gets((char_t*)tempTXT);
155       fifo_find_class(&student_fifo, atoi((char_t*)tempTXT));
156   }
```

Snippet 5 find num of students in class fun

```
164   void fifo_find_class(Sfifo_t* fifo, uint32 ID)
165   {
166       uint32 i,j,k=1,flag=0;
167       element_type *temp;
168       //check fifo is valid
169       if(!fifo->base || !fifo->head || !fifo->tail)
170       {
171           dprintf("[Error] The FIFO is invalid: \n");
172       }
173       //check fifo is empty
174       if(fifo->count == 0){
175           dprintf("The list is empty: \n");
176       }
177       else
178       {
179           temp = fifo->tail;
180           for (i=0;i<fifo->count;i++){
181               for(j=0;j<5;j++)
182               {
183                   if(ID == temp->c_id[j])
184                   {
185                       dprintf("%d): \n", k++);
186                       dprintf("The first name: %s\n", temp->fname);
187                       dprintf("The last name: %s\n", temp->lname);
188                       dprintf("The Roll number: %d\n", temp->roll);
189                       dprintf("The student's GPA: %0.2f\n", temp->GPA);
190                       dprintf("--------------------------------\n");
191                       flag=1;
192                   }
193               }
194               temp++;
195           }
196           if(flag == 0)
197           {
198               dprintf("No one is enrolled in class: %d \n", ID);
199           }
200       }
201   }
202
```

FIFO 5 find num of students in class fun

- Find the total number of students.

```
157  void total_s()
158  {
159      dprintf("\n\n-----------------------------------\n\n");
160      dprintf("The total number of students: %d\n", student_fifo.count);
161      dprintf("You can add up to %d students\n", student_fifo.length);
162      dprintf("You can add %d more students\n", student_fifo.length - student_fifo.count);
163      dprintf("\n-------------------------------------------------\n");
164  }
```

Snippet 6 Find the total number of students.fun

- Delete a student.

```
165  uint8 del_s()
166  {
167      uint8 tempTXT[40];
168      dprintf("\n----------------------\n \n");
169      dprintf("Enter the Roll number \n");
170      gets((char_t*)tempTXT);
171      element_type* student = fifo_find_roll(&student_fifo, atoi((char_t*)tempTXT));
172      if(student == 0){
173          dprintf("can't find roll number %d \n", atoi((char_t*)tempTXT));
174      }
175      else{
176      fifo_delete(&student_fifo, student);
177      }
178      return 0;
179  }
```

Snippet 7 Delete a student.fun

```
203  void fifo_delete(Sfifo_t* fifo, element_type* item)
204  {
205      uint32 roll = item->roll;
206      element_type* p = item;
207      while(p<fifo->head)
208      {
209          p++;
210          *item = *p;
211          item=p;
212      }
213      dprintf("Roll number %d is removed successfully\n", roll);
214      fifo->head--;
215      fifo->count--;
216  }
```

FIFO 6 delete a student fun

- Update student's information.

```
}
void up_S()
{
    uint8 tempTXT[40];
    dprintf("\n----------------------\n \n");
    dprintf("Enter the Roll number \n");
    gets((char_t*)tempTXT);
    element_type* student = fifo_find_roll(&student_fifo, atoi((char_t*)tempTXT));
    if(student == NULL)
    {
        dprintf("Cannot find the student \n");
    }
    else
    {
        dprintf("Enter the value you want to change: \n");
        dprintf("1. First name\n");
        dprintf("2. Last name\n");
        dprintf("3. Roll NO. \n");
        dprintf("4. GPA \n");
        dprintf("5. Courses\n");
        gets((char_t*)tempTXT);

        switch(atoi((char_t*)tempTXT))
        {
        case 1:
            dprintf("Enter the new first name: ");
            scanf("%s", student->fname);
            break;
        case 2:
            dprintf("Enter the new last name: ");
            scanf("%s", student->lname);
            break;
        case 3:
            dprintf("Enter the new roll number: ");
            scanf("%d", &student->roll);
            break;
        case 4:
            dprintf("Enter the new GPA: ");
            scanf("%f", &student->GPA);
            break;
        case 5:
            dprintf("Enter the course number you want to update: ");
            uint32 course;
            scanf("%d", &course);
            dprintf("Enter the new course id: ");
            scanf("%d", &student->c_id[course-1]);
            break;
        default:
            dprintf("INVALID OPTION!!!!\n");

        }
        dprintf("Student's details updated successfully\n");
    }
    dprintf("--------------------------------\n");
}

```

Snippet 8 Update student's information. Fun

- Show all the students in the database.

```
234
235   void show_s()
236   {
237       fifo_print(&student_fifo);
238       total_s();
239   }
240
```

Snippet 9 Show all the students in the database fun

```
65  fifo_status fifo_print(Sfifo_t* fifo)
66  {
67      element_type *temp;
68      uint32 i,j;
69      //check fifo is valid
70      if(!fifo->base || !fifo->head || !fifo->tail)
71          return fifo_null;
72      //check fifo is empty
73      if (fifo->count == 0)
74          return fifo_empty;
75      else {
76          temp = fifo->tail;
77          printf("\n\n ======The student's details are======\n\n");
78          for (i=0;i<fifo->count;i++){
79              dprintf("%d): \n", i+1);
80              dprintf("The first name: %s\n", temp->fname);
81              dprintf("The last name: %s\n", temp->lname);
82              dprintf("The Roll number: %d\n", temp->roll);
83              dprintf("The student's GPA: %0.2f\n", temp->GPA);
84              dprintf("Student's courses: \n");
85              for (j =0 ; j < 5; j++){
86                  dprintf("Course %d ID: %d\n", i+1, temp->c_id[i]);
87              }
88              dprintf("--------------------------------\n");
89              temp++;
90          }
91      }
92      return fifo_no_error;
93  }
```

FIFO 7 Show all the students in the database fun