

```

1  /*
2  * FIFO.c
3  *
4  * Created on: 12 Sept 2022
5  * Author: MAshr
6  */
7  #include "FIFO.h"
8
9  FIFO_status FIFO_init(FIFO_buf_t *fifo, element_type *buf, unsigned int length) {
10
11      if (buf == NULL)
12          return FIFO_null;
13
14      fifo->base = buf;
15      fifo->tail = buf;
16      fifo->head = buf;
17      fifo->count = 0;
18      fifo->length = length;
19
20      return FIFO_no_error;
21 }
22
23 FIFO_status FIFO_enqueue(FIFO_buf_t *fifo, element_type item) {
24
25     if (!fifo->base || !fifo->tail || !fifo->head)
26         return FIFO_null;
27     if (FIFO_is_full(fifo) == FIFO_full)
28         return FIFO_full;
29
30     *(fifo->head) = item;
31     if (fifo->head == fifo->base + (fifo->length * sizeof(element_type)))
32         fifo->head = fifo->base;
33     else
34         fifo->head++;
35
36     fifo->count++;
37     return FIFO_no_error;
38 }
39
40
41 FIFO_status FIFO_dequeue(FIFO_buf_t *fifo, element_type *item) {
42
43     if (!fifo->base || !fifo->tail || !fifo->head)
44         return FIFO_null;
45     if (fifo->count == 0)
46         return FIFO_empty;
47
48     *item = *fifo->tail;
49     fifo->count--;
50
51     if (fifo->tail == fifo->base + (fifo->length * sizeof(element_type)))
52         fifo->tail = fifo->base;
53     else
54         fifo->tail++;
55
56
57     return FIFO_no_error;
58 }
59

```

```

60 FIFO_status FIFO_is_full(FIFO_buf_t *fifo) {
61
62     if (!fifo->base || !fifo->tail || !fifo->head)
63         return FIFO_null;
64     if (fifo->count == fifo->length)
65         return FIFO_full;
66     else return FIFO_no_error;
67 }
68
69 void FIFO_print(FIFO_buf_t *fifo) {
70
71     int i;
72     element_type *ptrtemp;
73
74     if (fifo->count == 0)
75     {
76         Dprintf("FIFO is empty\n");
77     }
78     else {
79         ptrtemp = fifo->tail;
80         Dprintf("\n===== print ===== \n");
81         for (i = 0; i < fifo->count; i++) {
82             // Dprintf("\t (%x) \n", *ptrtemp);
83             ptrPrinterFnc(ptrtemp);
84             ptrtemp++;
85         }
86     }
87 }
88
89 FIFO_status FIFO_delete_element(FIFO_buf_t* fifo, element_type* item)
90 {
91     if (!fifo->base || !fifo->tail || !fifo->head)
92         return FIFO_null;
93     if (fifo->count == 0)
94         return FIFO_empty;
95     int i= 0 ;
96
97     while (item < fifo->head)
98     {
99         *item=*(item+1);
100
101         if (item == fifo->base + (fifo->length * sizeof(element_type)))
102             item = fifo->base;
103         else
104             item++;
105     }
106     fifo->count-- ;
107
108     if (fifo->head == fifo->base)
109         fifo->head = fifo->base + (fifo->length * sizeof (element_type));
110     else
111         fifo->head--;
112     return FIFO_no_error ;
113 }
114

```

```

1  /*
2   * FIFO.h
3   *
4   * Created on: 12 Sept 2022
5   * Author: MAshr
6   */
7
8  #ifndef FIFO_H_
9  #define FIFO_H_
10
11 #include<stdio.h>
12 #include<stdint.h>
13
14 #include "FIFO_config.h"
15
16 #define Dprintf(...) printf(__VA_ARGS__);fflush(stdout);fflush(stdin)
17
18 //create buffer
19
20 element_type uart_buff[buffSize];
21
22 typedef struct {
23     unsigned int length ;
24     unsigned int count ;
25     element_type *head ;
26     element_type *base ;
27     element_type *tail ;
28 }FIFO_buf_t;
29
30 typedef enum {
31     FIFO_no_error ,
32     FIFO_full,
33     FIFO_empty,
34     FIFO_null
35 }FIFO_status;
36
37
38 // API's
39
40 FIFO_status FIFO_init(FIFO_buf_t* fifo , element_type* buf, unsigned int length) ;
41 FIFO_status FIFO_enqueue(FIFO_buf_t* fifo , element_type item) ;
42 FIFO_status FIFO_dequeue(FIFO_buf_t* fifo , element_type* item) ;
43 FIFO_status FIFO_is_full(FIFO_buf_t* fifo) ;
44 void FIFO_print(FIFO_buf_t* fifo) ;
45 FIFO_status FIFO_delete_element(FIFO_buf_t* fifo, element_type* item);
46
47
48 #endif /* FIFO_H_ */
49

```

```
1  /*
2  * main.c
3  *
4  * Created on: 12 Sept 2022
5  * Author: MAshr
6  */
7  #include "Management_System.h"
8  #include "stdlib.h"
9  int main(void) {
10     int choice;
11     systemCreate();
12     while (1) {
13         systemPrintMenu();
14         scanf("%d",&choice);
15         switch (choice) {
16             case 1:
17                 systemAddStudent_manually();
18                 break;
19             case 2:
20                 loadDataFile();
21                 break;
22             case 3:
23                 systemFindStudentRoll();
24                 break;
25             case 4:
26                 systemFindStudentName();
27                 break;
28             case 5:
29                 systemFindStudentCourseID();
30                 break;
31             case 6:
32                 systemStatistics();
33                 break;
34             case 7:
35                 systemDeleteStudent();
36                 break;
37             case 8:
38                 systemUpdateStudent();
39                 break;
40             case 9:
41                 systemPrintAll();
42                 break;
43             case 10:
44                 systemUpdateStudentFile();
45                 break;
46             case 11:
47                 exit(1);
48         }
49     }
50 }
51
```

1 1 Mina Karam 2.80 1 2 3 4 5
2 3 Leila Mustafa 3.40 54 12 55 81 64
3 4 Koky Samy 3.60 5 21 55 92 64
4 5 Esraa Hegazy 3.60 54 12 55 81 64
5 2 Mahmoud Ashraf 4.00 15 1 20 3 40
6

```
1 //
2 // Created by MAshr on 22/09/2022.
3 //
4
5 #ifndef STUDENT_MANAGEMENT_SYSTEM_FIFO_CONFIG_H
6 #define STUDENT_MANAGEMENT_SYSTEM_FIFO_CONFIG_H
7
8 #include "Management_system_types.h"
9
10 // USER configuration
11 //select element type
12 //fifo data types
13 #define element_type studentInfo_t
14 #define buffSize (unsigned int)50
15 extern systemStatus_t systemPrintStudent(studentInfo_t *);
16 extern systemStatus_t (*ptrPrinterFnc)(studentInfo_t*);
17 #endif //STUDENT_MANAGEMENT_SYSTEM_FIFO_CONFIG_H
18
```

```
1 cmake_minimum_required(VERSION 3.23)
2 project(Student_Management_System C)
3
4 set(CMAKE_C_STANDARD 11)
5
6 include_directories(.)
7
8 add_executable(Student_Management_System
9     FIFO.c
10    FIFO.h
11    main.c
12    Management_System.h
13    FIFO_config.h
14    Management_system_types.h
15    Management_System.c
16 )
17
```

| | | | | | | | | |
|---|---|-----------------|-----|----|----|----|----|----|
| 1 | 1 | Ahmed Gamal | 2.8 | 1 | 2 | 3 | 4 | 5 |
| 2 | 1 | Mohamed Fahmy | 3.3 | 5 | 21 | 73 | 92 | 36 |
| 3 | 3 | Mahmoud Mustafa | 3.4 | 54 | 12 | 55 | 81 | 64 |
| 4 | 4 | Ahmed Samy | 3.6 | 5 | 21 | 55 | 92 | 64 |
| 5 | 5 | Islam Hilal | 3.6 | 54 | 12 | 55 | 81 | 64 |


```

1 //
2 // Created by MAshr on 23/09/2022.
3 //
4 /*****
5  *   FILE DESCRIPTION
6  *   -----
7  **      \file   FileName.c
8  *      \brief
9  *
10 *      \details
11 *
12 *
13 *****/
14
15 /*****
16  *   INCLUDES
17 *****/
18 #include "Management_System.h"
19 #include <string.h>
20 #include <stdio.h>
21 #include <stdlib.h>
22
23 /*****
24  *   LOCAL MACROS CONSTANT\FUNCTION
25 *****/
26
27 /*****
28  *   LOCAL DATA
29 *****/
30
31 /*****
32  *   GLOBAL DATA
33 *****/
34 studentInfo_t mainBuff[buffSize];
35 FIFO_buf_t FIFO_info;
36
37 systemStatus_t (*ptrPrinterFnc)(studentInfo_t *);
38
39 /*****
40  *   LOCAL FUNCTION PROTOTYPES
41 *****/
42 studentInfo_t *systemCatchStudentData();
43
44 systemStatus_t checkRollID(int rollID);
45
46 systemStatus_t systemPrintStudent(studentInfo_t *);

```

```

47
48 systemStatus_t printUpdateMenu();
49
50 systemStatus_t updateStudentRoll(studentInfo_t *Student);
51
52 systemStatus_t updateStudentFname(studentInfo_t *Student);
53
54 systemStatus_t updateStudentLname(studentInfo_t *Student);
55
56 systemStatus_t updateStudentGPA(studentInfo_t *Student);
57
58 systemStatus_t updateStudentCourses(studentInfo_t *Student);
59
60 systemStatus_t FIFO_print_file(FIFO_buf_t *fifo);
61 /*****
    *****/
62 * LOCAL FUNCTIONS
63 *****/
64
65 studentInfo_t *systemCatchStudentData() {
66     static studentInfo_t temp;
67     char str[20];
68     Dprintf("Enter Student Roll ID\n");
69     scanf("%d", &temp.roll);
70     if (!checkRollID(temp.roll)) {
71         Dprintf("Enter Student first Name \n");
72         gets(str);
73         strcpy(temp.firstName, str);
74         Dprintf("Enter Student last Name \n");
75         gets(str);
76         strcpy(temp.lastName, str);
77         Dprintf("Enter Student GPA \n");
78         scanf("%f", &temp.GPA);
79         Dprintf("Enter Student Courses ID\n");
80         for (int i = 0; i < 5; i++) {
81             Dprintf("==> Enter Course %d ID \n", i + 1);
82             scanf("%d", &temp.courseID[i]);
83         }
84     } else {
85         return NULL;
86     }
87     return &temp;
88 }
89
90 /*-----*/
91 systemStatus_t printUpdateMenu() {
92     Dprintf("Choose which item to update : \n");
93     Dprintf("1- Roll ID \n");
94     Dprintf("2- First name \n");
95     Dprintf("3- Last name \n");
96     Dprintf("4- GPA \n");
97     Dprintf("5- Courses \n");
98     Dprintf("6- Back to main menu\n");
99 }
100
101 systemStatus_t systemPrintMenu() {
102     char menu[15][50] = {
103         "-----\n",

```

```

104         " Chose the task that you want to perform \n",
105         "1- Add the students data manually\n",
106         "2- Load Students data from file \n",
107         "3- Find student details by roll number \n",
108         "4- Find student details by first name \n",
109         "5- Find student details by course ID \n",
110         "6- Find total number of students\n",
111         "7- Delete student details by roll number \n",
112         "8- Update student details by roll number \n",
113         "9- Show all information\n",
114         "10- Save data to output file\n",
115         "11- Exit the system\n",
116         "==> Enter your choice number \n "
117     };
118     for (int i = 0; i < 15; i++) {
119         Dprintf("%s", menu[i]);
120     }
121
122     return System_noError;
123 }
124
125 /*-----
126 */
127
128 systemStatus_t checkRollID(int rollID) {
129     int i = 0;
130     while (i < FIFO_info.count) {
131         if (FIFO_info.tail[i].roll == rollID) {
132             Dprintf("-----\n");
133             Dprintf("[ERROR] : This Roll ID (%d) already exist \n", rollID);
134             Dprintf("-----\n");
135
136             systemPrintStudent(&FIFO_info.tail[i]);
137             return System_Error;
138         }
139         i++;
140     }
141
142     return System_noError;
143 }
144
145 /*-----
146 */
147
148 systemStatus_t systemPrintStudent(studentInfo_t *student) {
149     Dprintf("Student Roll Number : %d\n", student->roll);
150     Dprintf("First Name : %s\n", student->firstName);
151     Dprintf("Last Name : %s\n", student->lastName);
152     Dprintf("GPA: %f\n", student->GPA);
153     Dprintf("Registered Courses ID : [");
154     for (int i = 0; i < 5; i++) {
155         Dprintf(" %d ", student->courseID[i]);
156     }
157     Dprintf("]\n-----\n");
158     return System_noError;
159 }
160
161 systemStatus_t updateStudentRoll(studentInfo_t *Student) {
162     int nRollID;
163     Dprintf("Enter Student (New) Roll ID \n");
164     scanf("%d", &nRollID);

```

```

161     if (!checkRollID(nRollID)) {
162         Student->roll = nRollID;
163     }
164 }
165
166 systemStatus_t updateStudentFname(studentInfo_t *Student) {
167     char str[20];
168     Dprintf("Enter Student ( New ) first Name \n");
169     gets(str);
170     strcpy(Student->firstName, str);
171 }
172
173 systemStatus_t updateStudentLname(studentInfo_t *Student) {
174     char str[20];
175     Dprintf("Enter Student ( New ) last Name \n");
176     gets(str);
177     strcpy(Student->lastName, str);
178 }
179
180 systemStatus_t updateStudentGPA(studentInfo_t *Student) {
181     Dprintf("Enter Student (New) GPA \n");
182     scanf("%f", &Student->GPA);
183 }
184
185 systemStatus_t updateStudentCourses(studentInfo_t *Student) {
186     Dprintf("Enter Student (New) Courses ID\n");
187     for (int i = 0; i < 5; i++) {
188         Dprintf("==> Enter Course %d ID \n", i + 1);
189         scanf("%d", &Student->courseID[i]);
190     }
191 }
192
193 /*****
194  * GLOBAL FUNCTIONS
195  *****/
196
197
198 systemStatus_t systemCreate() {
199     FIFO_init(&FIFO_info, mainBuff, buffSize);
200     Dprintf("FIFO has been initialized successfully\n");
201     Dprintf("=== WELCOME TO STUDENT MANAGEMENT SYSTEM ===\n");
202     //systemPrintMenu();
203     return System_noError;
204 }
205
206
207 systemStatus_t systemAddStudent_manually() {
208     studentInfo_t *ptrStudent = systemCatchStudentData();
209     if (ptrStudent) {
210         FIFO_enqueue(&FIFO_info, *ptrStudent);
211         Dprintf("[INFO] student roll %d has been added successfully\n", ptrStudent->
roll);
212         Dprintf("System stored %d student remains : %d student \n", FIFO_info.count,
buffSize - FIFO_info.count);
213         return System_noError;
214     } else {
215         return System_Error;

```

```

216     }
217 }
218
219 systemStatus_t systemFindStudentRoll() {
220     int i = 0, id;
221     Dprintf("Enter Roll ID for the student \n");
222     scanf("%d", (int *) &id);
223     while (i < FIFO_info.count) {
224         if (FIFO_info.tail[i].roll == id) {
225             systemPrintStudent(&FIFO_info.tail[i]);
226             return System_noError;
227         }
228         i++;
229     }
230     Dprintf("-----\n");
231     Dprintf("[ERROR] : This ID doesn't exist \n");
232     Dprintf("-----\n");
233     return System_Error;
234 }
235 }
236
237 systemStatus_t systemFindStudentName() {
238     int i = 0;
239     char name[20];
240     Dprintf("Enter first name for the student \n");
241     gets(name);
242     while (i < FIFO_info.count) {
243         if (!strcmpi(name, FIFO_info.tail[i].firstName)) {
244             systemPrintStudent(&FIFO_info.tail[i]);
245             return System_noError;
246         }
247         i++;
248     }
249     Dprintf("-----\n");
250     Dprintf("[ERROR] : This name doesn't exist \n");
251     Dprintf("-----\n");
252     return System_Error;
253 }
254
255 systemStatus_t systemFindStudentCourseID() {
256     int i = 0, courseID, count = 0;
257     studentInfo_t *ptrStudent = NULL;
258     Dprintf("Enter Course ID \n");
259     scanf("%d", (int *) &courseID);
260     while (i < FIFO_info.count) {
261         for (int j = 0; j < 5; j++) {
262             if (FIFO_info.tail[i].courseID[j] == courseID) {
263                 ptrStudent = &FIFO_info.tail[i];
264                 Dprintf("The students registered this course ID are : \n");
265                 systemPrintStudent(ptrStudent);
266                 count++;
267             }
268         }
269         i++;
270     }
271     if (ptrStudent) {
272         Dprintf("Number of student register Course ID: (%d) are : (%d)\n", courseID,
count);
273         Dprintf("-----\n");

```

```

274     return System_noError;
275 } else {
276     Dprintf("[INFO] : No student register this course \n");
277     return System_Error;
278 }
279 }
280
281 systemStatus_t systemPrintAll() {
282     ptrPrinterFnc = (systemStatus_t (*)(studentInfo_t *)) systemPrintStudent;
283     FIFO_print(&FIFO_info);
284     return System_noError;
285 }
286
287 systemStatus_t systemStatistics() {
288     Dprintf("Maximum size of storage is : %d \n", buffSize);
289     Dprintf("Number of students have been stored : %d student \n", FIFO_info.count);
290     Dprintf("Number of students remain : %d\n", buffSize - FIFO_info.count);
291     Dprintf("-----\n");
292 }
293
294 systemStatus_t loadDataFile() {
295     FILE *infile;
296     studentInfo_t tempStudent;
297     infile = fopen("StudentData.txt", "r");
298     if (infile == NULL) {
299         fprintf(stderr, "\nError opening file\n");
300         Dprintf("[ERROR] Can't open the file \n");
301     }
302     fscanf(infile, "%d", &tempStudent.roll);
303
304     while (!feof(infile)) {
305         fscanf(infile, "%d", &tempStudent.roll);
306         if (!checkRollID(tempStudent.roll)) {
307             fscanf(infile, "%s", tempStudent.firstName);
308             fscanf(infile, "%s", tempStudent.lastName);
309             fscanf(infile, "%f", &tempStudent.GPA);
310             for (int i = 0; i < 5; i++) {
311                 fscanf(infile, "%d", &tempStudent.courseID[i]);
312             }
313             FIFO_enqueue(&FIFO_info, tempStudent);
314             Dprintf("[INFO] student roll %d has been added successfully\n",
tempStudent.roll);
315             Dprintf("[INFO] System stored %d student remains : %d student \n",
FIFO_info.count,
buffSize - FIFO_info.count);
316         } else {
317             fscanf(infile, "%*[^\\n]");
318         }
319     }
320     fclose(infile);
321     return System_noError;
322 }
323
324
325 systemStatus_t systemDeleteStudent() {
326     int i = 0, rollID;
327     Dprintf("Enter Roll ID for the student \n");
328     scanf("%d", &rollID);
329     while (i < FIFO_info.count) {
330         if (FIFO_info.tail[i].roll == rollID) {

```

```

331         FIFO_delete_element(&FIFO_info, &FIFO_info.tail[i]);
332         Dprintf("[INFO] Student roll %d has been deleted successfully \n", rollID
);
333         return System_noError;
334     }
335     i++;
336 }
337 Dprintf("-----\n");
338 Dprintf("[ERROR] : This ID doesn't exist \n");
339 Dprintf("-----\n");
340 return System_Error;
341 }
342
343 systemStatus_t systemUpdateStudent() {
344     int i = 0, rollID, choice;
345     Dprintf("Enter Roll ID for the student \n");
346     scanf("%d", &rollID);
347     while (i < FIFO_info.count) {
348         if (FIFO_info.tail[i].roll == rollID) {
349             while (1) {
350                 printUpdateMenu();
351                 scanf("%d", &choice);
352                 switch (choice) {
353                     case 1:
354                         updateStudentRoll(&FIFO_info.tail[i]);
355                         break;
356                     case 2:
357                         updateStudentFname(&FIFO_info.tail[i]);
358                         break;
359                     case 3:
360                         updateStudentLname(&FIFO_info.tail[i]);
361                         break;
362                     case 4:
363                         updateStudentGPA(&FIFO_info.tail[i]);
364                         break;
365                     case 5:
366                         updateStudentCourses(&FIFO_info.tail[i]);
367                         break;
368                     case 6:
369                         break;
370                 }
371                 Dprintf("[INFO] Student information has been updated successfully \n"
);
372                 if (choice == 6) break;
373             }
374             return System_noError;
375         }
376         i++;
377     }
378     Dprintf("-----\n");
379     Dprintf("[ERROR] : This ID doesn't exist \n");
380     Dprintf("-----\n");
381     return System_Error;
382 }
383
384 systemStatus_t systemUpdateStudentFile() {
385
386     FIFO_print_file(&FIFO_info);
387     return System_noError;

```

```

388
389 }
390
391 systemStatus_t FIFO_print_file(FIFO_buf_t *fifo) {
392
393     int i;
394     element_type *ptrtemp;
395
396     if (fifo->count == 0) {
397         Dprintf("FIFO is empty\n");
398     } else {
399         ptrtemp = fifo->tail;
400         Dprintf("\n===== printing in file ===== \n");
401
402         FILE *outfile;
403         outfile = fopen("outfile.txt", "w");
404         if (outfile == NULL) {
405             fprintf(stderr, "\nError opening file\n");
406             Dprintf("[ERROR] Can't open the file \n");
407         }
408         for (i = 0; i < fifo->count; i++) {
409
410             fprintf(outfile, "%d ", ptrtemp->roll);
411             fprintf(outfile, "%s ", ptrtemp->firstName);
412             fprintf(outfile, "%s ", ptrtemp->lastName);
413             fprintf(outfile, "%.2f ", ptrtemp->GPA);
414             for (int j = 0; j < 5; j++) {
415                 fprintf(outfile, "%d ", ptrtemp->courseID[j]);
416             }
417
418             fprintf(outfile, "\n");
419
420             if (ptrtemp == fifo->base + (fifo->length * sizeof(element_type)))
421                 ptrtemp = fifo->base;
422             else
423                 ptrtemp++;
424
425         }
426         Dprintf("[INFO] Data saved to file successfully \n") ;
427         fclose(outfile);
428     }
429 }
430
431
432 /*****
433  * END OF FILE: Management_System.c
434  *****/
435

```



```

1 //
2 // Created by MAshr on 22/09/2022.
3 //
4
5 #ifndef STUDENT_MANAGEMENT_SYSTEM_MANAGEMENT_SYSTEM_H
6 #define STUDENT_MANAGEMENT_SYSTEM_MANAGEMENT_SYSTEM_H
7
8 /*****
9 *****/
10 * FILE DESCRIPTION
11 * -----
12 *
13 * File: <Management_system.h>
14 *
15 * Module: -
16 *
17 * Description: < >
18 *
19 *****/
20 /*****
21 *****/
22 * INCLUDES
23 *****/
24 #include "FIFO.h"
25 #include "Management_system_types.h"
26 /*****
27 *****/
28 * GLOBAL CONSTANT MACROS
29 *****/
30 /*****
31 *****/
32 * GLOBAL FUNCTION MACROS
33 *****/
34 /*****
35 *****/
36 * GLOBAL DATA TYPES AND STRUCTURES
37 *****/
38
39
40 /*****
41 *****/
42 * GLOBAL DATA PROTOTYPES
43 *****/
44 extern studentInfo_t mainBuff[buffSize];
45 extern FIFO_buf_t FIFO_info;
46 /*****

```

```

46 *****
47 * GLOBAL FUNCTION PROTOTYPES
48 *****
   *****/
49 systemStatus_t systemCreate();
50 systemStatus_t systemPrintMenu();
51 systemStatus_t loadDataFile();
52 systemStatus_t systemAddStudent_manually();
53 systemStatus_t systemFindStudentRoll();
54 systemStatus_t systemFindStudentName();
55 systemStatus_t systemFindStudentCourseID();
56 systemStatus_t systemStatistics();
57 systemStatus_t systemDeleteStudent();
58 systemStatus_t systemUpdateStudent();
59 systemStatus_t systemPrintAll();
60 systemStatus_t systemUpdateStudentFile();
61 /*****
   *****/
62 * END OF FILE: Std_Types.h
63 *****/
   *****/
64
65 #endif //STUDENT_MANAGEMENT_SYSTEM_MANAGEMENT_SYSTEM_H
66

```

```
1 //
2 // Created by MAshr on 22/09/2022.
3 //
4
5 #ifndef STUDENT_MANAGEMENT_SYSTEM_MANAGEMENT_SYSTEM_TYPES_H
6 #define STUDENT_MANAGEMENT_SYSTEM_MANAGEMENT_SYSTEM_TYPES_H
7
8 /*****
9  * GLOBAL DATA PROTOTYPES
10 *****/
11
12 typedef struct {
13     int roll ;
14     char firstName[20];
15     char lastName[20];
16     float GPA ;
17     int courseID[5];
18 }studentInfo_t;
19
20 typedef enum {
21     System_noError,
22     System_Error
23 }systemStatus_t;
24 #endif //STUDENT_MANAGEMENT_SYSTEM_MANAGEMENT_SYSTEM_TYPES_H
25
26
```