


Full Name: Micah Nicole Chu Im
Email: micah_chuim@yahoo.com
Test Name: **MP6 B-2**
Taken On: 13 Apr 2016 09:55:03 PHT
Time Taken: 43 min 56 sec/ 90 min
Invited by: Ryan
Invited on: 12 Apr 2016 18:53:54 PHT
Tags Score:




Candidate Feedback: sir ugh grabe lagi. nagarun sya perfectly sa command prompt then pagdito sa hackerrank kay may errors na hindi magets.

Recruiter/Team Comments:

No Comments.

Question Description	Time Taken	Score	Status
Q1 I Am, First and Foremost, a Student > Coding	33 min 14 sec	10/ 10	

QUESTION 1

 Correct Answer

 Score 10

I Am, First and Foremost, a Student > Coding
QUESTION DESCRIPTION

Create a structure that would represent a student. It should contain the following information:

- hyphenated student number
- first name
- middle name
- last name
- program
- year level

Assuming that we are keeping a record of all the students enrolled in UP Cebu, create another structure that contains an array of students (with 20 as capacity -- so much for keeping a record of all the students in UP Cebu) and of course, a size.

We should be able to do the following:

1. enroll a student
2. withdraw a student from the roll
3. display all students
4. search all students based on the year level
5. search all students based on the course
6. search all students based on the family name
7. search a student based on the student number

ALL THESE SHOULD BE IMPLEMENTED AS FUNCTIONS (this means, apart from the main, you should have at least 7 functions).

The first input is the number of operations, say T, to be executed followed T inputs which are a combination of type of input (1-7, refer to the listing above) and the actual input. The output is based on the input type. If the input type is 1, 2, or 3, the output is the display of all the students in the record. The output for all other input types is still a display but based only on the search criteria.

The display should print all the information of a student exactly like the following:

Student Number: 1997-65336
Name: Dulaca, Ryan Ciriaco Madolin
Program: BS Math Computer Science
Year Level: 4

If there exists more than one student in the list, separate the display with a blank line exactly like the one below:

Student Number: 1997-65336
Name: Dulaca, Ryan Ciriaco Madolin
Program: BS Math Computer Science
Year Level: 4

Student Number: 2015-55038
Name: Noynay, Danny Boy Anislag
Program: BS Computer Science
Year Level: 1

Sample Input:

3
1
1997-65336
Ryan Ciriaco
Madolin
Dulaca
BS Math Computer Science
4
3
1
2015-55038
Danny Boy
Noynay
Anislag
BS Computer Science
1

Output:

Student Number: 1997-65336
Name: Dulaca, Ryan Ciriaco Madolin
Program: BS Math Computer Science
Year Level: 4

Student Number: 1997-65336
Name: Dulaca, Ryan Ciriaco Madolin
Program: BS Math Computer Science
Year Level: 4

Name: Dulaca, Ryan Ciriaco Madolin
Program: BS Math Computer Science
Year Level: 4

Student Number: 2015-55038
Name: Noynay, Danny Boy Anislag
Program: BS Computer Science
Year Level: 1

CANDIDATE ANSWER

Language used: C

```
1 #include <math.h>
2 #include <stdio.h>
3 #include <string.h>
4 #include <stdlib.h>
5 #include <assert.h>
6 #include <limits.h>
7 #include <stdbool.h>
8 #define MAX 20
9
10 typedef struct{
11     int yearLevel;
12     char firstName[30],middleName[30],lastName[30],studentnumber[11], program[30];
13 }student;
14
15 typedef struct{
16     int size;
17     student record[20];
18 }list;
19
20
21 void init(list*);
22 int enroll(list*,student);
23 int withdraw(list*,char*);
24 void display(list);
25 int search(list ,char*);
26 void searchyearl(list,int);
27 void searchC(list,char*);
28 void searchfamily(list,char*);
29 void searchSN(list,char*);
30 int remo(list*,int);
31 int check(list ,char*);
32
33 int main() {
34     int t,inputtype,c;
35     student s;
36     list l;
37
38     init(&l);
39     scanf("%d",&t);
40
41     do{
42         scanf("%d",&inputtype);
43
44         if(inputtype==1){
45             scanf("%[^\\n]s", s.studentnumber);
46             scanf("%[^\\n]s", s.firstName);
47             scanf("%[^\\n]s", s.middleName);
48             scanf("%[^\\n]s", s.lastName);
49             scanf("%[^\\n]s", s.program);
50             scanf("%d", &s.yearLevel);
51
52             c=check(l,s.studentnumber);
53
54             if(c!=1)
55                 enroll(&l,s);
56
57             else
58                 printf("THE STUDENT NUMBER %s ALREADY EXISTS.\\n\\n",s.studentnumber);
59
60             display(l);
61             //printf("\\n");
62         }
63         if(inputtype==2){
64             scanf("%[^\\n]s", s.studentnumber);
65             withdraw(&l,s.studentnumber);
66
67             display(l);
68         }
69
70         else if(inputtype==3){
```

```

71         display(l);
72     }
73     else if(inputtype==4){
74         //search all students based on yearlevel
75         scanf("%d",&s.yearLevel);
76         searchyearl(l,s.yearLevel);
77     }
78     else if(inputtype==5){
79         //search based on course
80         scanf(" %[^\\n]s",s.program);
81         searchC(l,s.program);
82     }
83     else if(inputtype==6){
84         //search all students based on the family name
85         scanf(" %[^\\n]s",s.lastName);
86         searchfamily(l,s.lastName);
87     }
88     else if(inputtype==7){
89         //search a student based on the student number
90         scanf(" %[^\\n]s",s.studentnumber);
91         searchSN(l,s.studentnumber);
92     }
93     t--;
94 }while(t>0);
95
96
97
98     return 0;
99 }
100
101
102 void init(list* l){
103     l->size = 0;
104 }
105
106 int enroll(list* l,student s){
107     if(l->size == MAX)
108         return 0;
109     else{
110         l->record[l->size++]=s;
111         return 1;
112     }
113 }
114
115 void display(list l){
116     int i;
117     for(i=0; i<l.size; i++){
118         if(i>0)
119             printf("\\n");
120
121         printf("Student Number: %s\\n",l.record[i].studentnumber);
122         printf("Name: %s, %s
123 %s\\n",l.record[i].lastName,l.record[i].firstName,l.record[i].middleName);
124         printf("Program: %s\\n",l.record[i].program);
125         printf("Year Level: %d\\n",l.record[i].yearLevel);
126
127     }
128     printf("\\n");
129 }
130
131 int withdraw(list* l,char * sn){
132     int s= search(*l,sn);
133
134     if(s!=-1){
135         remo(l,s);
136         return 1;
137     }
138     else
139         return 0;
140 }
141
142 int search(list l,char* t){
143     int i;
144     for(i=0; i<l.size; i++)
145         if(strcmp(l.record[i].studentnumber,t)==0)

```

```








146         return i;
147     return -1;
148 }
149
150 int remo(list* l,int pos){
151     int i = pos;
152     if(i<0 || i >= l->size)
153         return 0;
154     else{
155         for(; i<l->size-1; i++){
156             //strcpy(l->record[i],l->record[i+1]);
157             l->record[i]=l->record[i+1];
158             l->size--;
159             return 1;
160         }
161     }
162
163 void searchyearl(list l,int y){
164     int i,flag=0;
165     for(i=0; i<l.size; i++){
166         if(l.record[i].yearLevel==y){
167             if(i>0)
168                 printf("\n");
169
170             printf("Student Number: %s\n",l.record[i].studentnumber);
171             printf("Name: %s, %s",
172 %s\n",l.record[i].lastName,l.record[i].firstName,l.record[i].middleName);
173             printf("Program: %s\n",l.record[i].program);
174             printf("Year Level: %d\n",l.record[i].yearLevel);
175
176             flag=1;
177         }
178     }
179
180     //printf("\n");
181
182     if(flag==0){
183         printf("NO STUDENT WITH YEAR LEVEL %d EXISTS.\n",y);
184         printf("\n");
185     }
186 }
187
188 void searchC(list l,char* t){
189     int i,flag=0;
190     for(i=0; i<l.size; i++){
191         if(strcmp(l.record[i].program,t)==0){
192             if(i>0)
193                 printf("\n");
194
195             printf("Student Number: %s\n",l.record[i].studentnumber);
196             printf("Name: %s, %s",
197 %s\n",l.record[i].lastName,l.record[i].firstName,l.record[i].middleName);
198             printf("Program: %s\n",l.record[i].program);
199             printf("Year Level: %d\n",l.record[i].yearLevel);
200
201             flag=1;
202         }
203     }
204     //printf("\n");
205     if(flag==0){
206         printf("NO STUDENT WITH THE COURSE %s EXISTS.\n",t);
207         //printf("\n");
208     }
209 }
210
211 void searchfamily(list l,char* t){
212     int i,flag=0;
213     for(i=0; i<l.size; i++){
214         if(strcmp(l.record[i].lastName,t)==0){
215             if(i>0)
216                 printf("\n");
217
218             printf("Student Number: %s\n",l.record[i].studentnumber);
219             printf("Name: %s, %s",
220 %s\n",l.record[i].lastName,l.record[i].firstName,l.record[i].middleName);

```

```

220 %s\n", l.record[i].lastName, l.record[i].firstName, l.record[i].middleName);
221     printf("Program: %s\n", l.record[i].program);
222     printf("Year Level: %d\n", l.record[i].yearLevel);
223
224     flag=1;
225 }
226 }
227 //printf("\n");
228 if(flag==0){
229     printf("NO STUDENT WITH THE FAMILY NAME %s EXISTS.\n", t);
230     // printf("\n");
231 }
232 }
233
234 void searchSN(list l, char* t){
235     int i, flag=0;
236     for(i=0; i<l.size; i++){
237         if(strcmp(l.record[i].studentnumber, t)==0){
238             if(i>0)
239                 printf("\n");
240
241             printf("Student Number: %s\n", l.record[i].studentnumber);
242             printf("Name: %s, %s",
243 %s\n", l.record[i].lastName, l.record[i].firstName, l.record[i].middleName);
244             printf("Program: %s\n", l.record[i].program);
245             printf("Year Level: %d\n", l.record[i].yearLevel);
246
247             flag=1;
248         }
249     }
250 }
251 //printf("\n");
252 if(flag==0){
253     printf("NO SUCH STUDENT WITH STUDENT NUMBER %s EXISTS.\n", t);
254     //printf("\n");
255 }
256 }
257
258 int check(list l, char* t){
259     int i;
260     for(i=0; i<l.size; i++){
261         if(strcmp(l.record[i].studentnumber, t)==0)
262             return 1;
263     }
264     return -1;
265 }
266
267
268 ///enroll a student
269 //withdraw a student from the roll
//display all students
//search all students based on the year level
//search all students based on the course
//search all students based on the family name
//search a student based on the student number

```

TESTCASE	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	 Success	1	0.0 sec	2.53 MB
Testcase 1	Easy	 Success	2	0.0 sec	2.29 MB
Testcase 2	Easy	 Success	2	0.0 sec	2.28 MB
Testcase 3	Easy	 Success	1	0.0 sec	2.28 MB
Testcase 4	Easy	 Success	1	0.0 sec	2.28 MB
Testcase 5	Easy	 Success	1	0.0 sec	2.28 MB
Testcase 6	Easy	 Success	1	0.0 sec	2.28 MB
Testcase 7	Easy	 Success	1	0.0 sec	2.29 MB

No Comments