

lab11

```
1 // EE231002 Lab11. Academic Competition
2 // 108061112, 林靖
3 // Date: Dec. 7, 2019
4
5 #include <stdio.h>           // Standard input and output library
6
7 typedef enum {FALSE, TRUE} Bool; // Boolean type
8
9 struct STU {                // structure definition for each students
10     char fName[15];         //     first name
11     char lName[15];         //     last name
12     double math, sci, lit;   //     test scores
13     double min;             //     minimum subject score
14 } ;
15 struct STU list[100];       // Subjects were tested among 100 students
16
17 Bool Find_Max_and_Print(const int *rank, double score[100]);
18                             // Find and print out the award winner with
19                             // max score. Return FALSE if have
20                             // printed out all the winners.
21
22 /*****
23 * Read all the data of the students into list[100], and simultaneously, *
24 * for each student, find the min score among three subjects. *
25 * After reading all the data, the prizes "Grand", "Math", "Sci" and "Lit" *
26 * were processed and printed separately. *
27 *****/
28 int main(void)
29 {
30     int i;                   // Index for looping
31     double score[100];       // Scores of candidate winners
32     int rank;                // Ranking of the winner
33
34     scanf("FirstName LastName Math Science Literature\n");
35     // To skip the first line in the data file
36
37     for (i = 0; i < 100; i++) { // For each student,
38         scanf("%s %s %lf %lf %lf\n", list[i].fName, // Read all the data
39             list[i].lName, // into list[100].
40             &list[i].math,
41             &list[i].sci,
42             &list[i].lit);
43         list[i].min = list[i].math; // Find the min score
44         if (list[i].min > list[i].sci) // among three subjects
45             list[i].min = list[i].sci; // and save it to
46         if (list[i].min > list[i].lit) // list[i].min
47             list[i].min = list[i].lit;
```

```

48     }
49
50     puts("Grand Prize:");
51     for (i = 0; i < 100; i++) {           // For each student
52         if (80 <= list[i].min)           // with all scores >= 80
53             score[i] = list[i].math + list[i].sci + list[i].lit; // Copy score.
54         else                               // Otherwise,
55             score[i] = 0;                 // flag ineligible.
56     }
57     for (rank = 1; Find_Max_and_Print(&rank, score); rank++) ;
58                                     // Print all winners out
59     puts("Math Prize:");
60     for (i = 0; i < 100; i++) {           // For each student
61         if (60 <= list[i].min && list[i].min < 80) // with all scores >= 60
62             score[i] = list[i].math;     // Copy score.
63         else                               // Otherwise,
64             score[i] = 0;                 // flag ineligible.
65     }
66     for (rank = 1; rank <= 10 && Find_Max_and_Print(&rank, score); rank++) ;
67                                     // Print all winners out
68     puts("Science Prize:");
69     for (i = 0; i < 100; i++) {           // For each student
70         if (60 <= list[i].min && list[i].min < 80) // with all scores >= 60
71             score[i] = list[i].sci;     // Copy score.
72         else                               // Otherwise,
73             score[i] = 0;                 // flag ineligible.
74     }
75     for (rank = 1; rank <= 10 && Find_Max_and_Print(&rank, score); rank++) ;
76                                     // Print all winners out
77     puts("Literature Prize:");
78     for (i = 0; i < 100; i++) {           // For each student
79         if (60 <= list[i].min && list[i].min < 80) // with all scores >= 60
80             score[i] = list[i].lit;     // Copy score
81         else                               // Otherwise,
82             score[i] = 0;                 // flag ineligible.
83     }
84     for (rank = 1; rank <= 10 && Find_Max_and_Print(&rank, score); rank++) ;
85                                     // Print all winners out
86     return 0; // Normal program termination
87 }
88
89 /*****
90 * Find the max score in score[100] given, and print out the data of the *
91 * student with the max score. Return FALSE if have printed out all the *
92 * winners. Otherwise, return TRUE. *
93 *****/
94 Bool Find_Max_and_Print(const int *rank, double score[100])
95 {
96     int i; // Index for looping
97     double score_max = 0; // The highest score found

```

```

98     int index_max;                // Index of the highest score
99
100    for (i = 0; i < 100; i++) {    // For each score,
101        if (score_max < score[i]) { // find the highest score
102            score_max = score[i];   // and save it to score_max.
103            index_max = i;          // Save index of the highest score
104        }
105    }
106
107    if (score_max == 0)             // If have printed out all the
108        return FALSE;              // winners, return FALSE.
109
110    printf(" %d: %s %s %.1f\n", *rank, // Print out the data
111        list[index_max].fName, // of the student with
112        list[index_max].lName, // the heghest score.
113        score_max);
114    score[index_max] = 0;           // Flag that this score
115                                    // has been printed out.
116    return TRUE;                   // There may be scores that have not
117 }                                  // yet been printed, so return TRUE.

```

[Format] can be improved.

[Coding] lab11.c spelling errors: heghest(1)

[Extra] array score is not needed.

Score: 93