1025. PAT Ranking (25)

时间限制

200 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Programming Ability Test (PAT) is organized by the College of Computer Science and Technology of Zhejiang University. Each test is supposed to run simultaneously in several places, and the ranklists will be merged immediately after the test. Now it is your job to write a program to correctly merge all the ranklists and generate the final rank.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains a positive number N (<=100), the number of test locations. Then N ranklists follow, each starts with a line containing a positive integer K (<=300), the number of testees, and then K lines containing the registration number (a 13-digit number) and the total score of each testee. All the numbers in a line are separated by a space.

**Output Specification:**

For each test case, first print in one line the total number of testees. Then print the final ranklist in the following format:

registration\_number final\_rank location\_number local\_rank

The locations are numbered from 1 to N. The output must be sorted in nondecreasing order of the final ranks. The testees with the same score must have the same rank, and the output must be sorted in nondecreasing order of their registration numbers.

**Sample Input:**

2

5

1234567890001 95

1234567890005 100

1234567890003 95

1234567890002 77

1234567890004 85

4

1234567890013 65

1234567890011 25

1234567890014 100

1234567890012 85

**Sample Output:**

9

1234567890005 1 1 1

1234567890014 1 2 1

1234567890001 3 1 2

1234567890003 3 1 2

1234567890004 5 1 4

1234567890012 5 2 2

1234567890002 7 1 5

1234567890013 8 2 3

1234567890011 9 2 4

被说明时说的给坑了一下，因为当分数一样时应该用id来排序；

#include<iostream>

#include<string>

#include<algorithm>

#include<queue>

#include<vector>

#include<sstream>

#include<stack>

using namespace std;

struct student

{

string id;

int score;

int group;

int local\_rank;

int total\_rank;

};

int comp(student\* a, student\* b)

{

if (a->score > b->score)

return 1;

else if (a->score == b->score)

{

if (a->id < b->id)

return 1;

else

return 0;

}

else

return 0;

}

vector<vector<student\*>>all\_student;

vector<student\*>all;

int main()

{

int group;

cin >> group;

all\_student.resize(group);

for (int i = 0; i < group; i++)

{

int num;

scanf("%d", &num);

for (int j = 0; j < num; j++)

{

student\* cur;

cur = new student;

/\*char temp[14];\*/

int score;

cin >> cur->id;

scanf("%d",&score);

cur->group = i;

cur->score = score;

all\_student[i].push\_back(cur);

all.push\_back(cur);

}

sort(all\_student[i].begin(), all\_student[i].end(), comp);

all\_student[i][0]->local\_rank = 1;

for (int j = 1; j < num; j++)

{

if (all\_student[i][j]->score == all\_student[i][j - 1]->score)

all\_student[i][j]->local\_rank = all\_student[i][j - 1]->local\_rank;

else

all\_student[i][j]->local\_rank = j+1;

}

/\*int nowrank = -1;

int nowgrade = -1;

for (int j = 0; j < num; j++)

{

if (all\_student[i][j]->score == nowgrade)

{

all\_student[i][j]->local\_rank = nowrank;

}

else

{

nowgrade = all\_student[i][j]->score;

nowrank = j + 1;

all\_student[i][j]->local\_rank = j + 1;

}

}\*/

}

sort(all.begin(), all.end(), comp);

all[0]->total\_rank = 1;

int allnum = all.size();

for (int i = 1; i < allnum; i++)

{

if (all[i]->score == all[i - 1]->score)

all[i]->total\_rank = all[i - 1]->total\_rank;

else

all[i]->total\_rank = i + 1;

}

/\*int nowrank = -1;

int nowgrade = -1;

for (int j = 0; j < allnum; j++)

{

if (all[j]->score == nowgrade)

{

all[j]->total\_rank = nowrank;

}

else

{

nowgrade = all[j]->score;

nowrank = j + 1;

all[j]->total\_rank = j + 1;

}

}\*/

cout << allnum << endl;

for (int i = 0; i < allnum; i++)

{

printf("%s %d %d %d\n", all[i]->id.c\_str(), all[i]->total\_rank, all[i]->group + 1, all[i]->local\_rank);

}

return 0;

}