[OOP]Assign3

Student ID: 20152008

Name: Kusdavletov Ernar

Date: 2017.03.30

1. Descriptions

The first line of input consists of number of cases. Then, comes blank line (there is also a blank line between two consecutive cases) and in each line: team number, problem number, submission time and decision. All of them separated by empty space.
Contestants are ranked first by the number of problems solved (the more the better), then by decreasing amounts of penalty time. If two or more contestants are tied in both problems solved and penalty time, they are displayed in order of increasing team numbers.
A problem is considered solved by a contestant if any of the submissions for that problem was judged correct. Penalty time is computed as the number of minutes it took until the first correct submission for a problem was received, plus 20 minutes for each incorrect submission prior to the correct solution. Unsolved problems incur no time penalties.
The program reads all inputs, computes how many problems each team solved, penalty time, time to solve problem, how many attempts was and was there any submission of any problem or not.
The output for each test case consists of a scoreboard, sorted by the criteria described above. Each line of output contains a contestant number, the number of problems solved by the contestant and the total time penalty
accumulated by the contestant. Since not all contestants are participating, in the output only those contestants
who have made a submission.
The output of two consecutive cases will be separated by a blank line.

2. Code

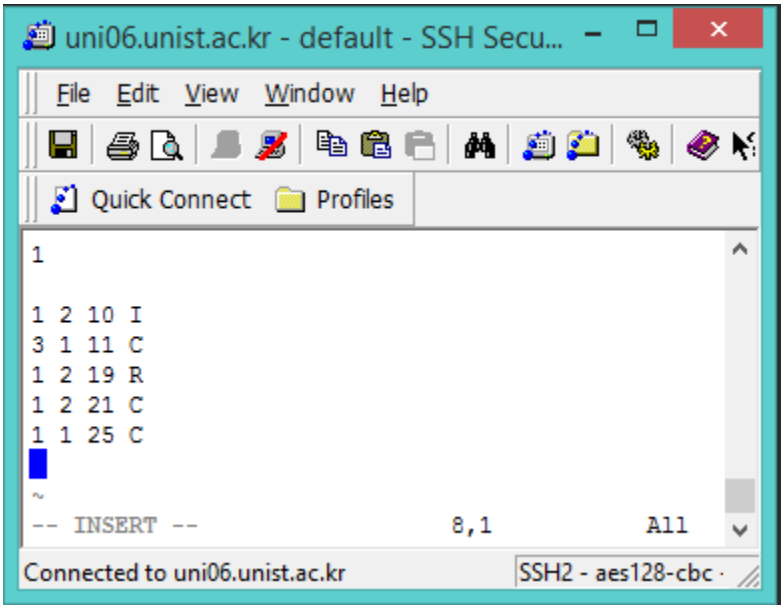
```
//Kusdavletov Ernar
#include <iostream>
using namespace std;

int main(){
   int cases_number;   //number of total tests
   cin >> cases_number;
   string checker;     //string to read lines
   getline(cin, checker);    //reading empty lines
   getline(cin, checker);
   for (int i = 0; i < cases_number; i++){
      if (i != 0){    //to make a blank line between different tests
         cout << endl;
      }
      int penalty[101][10], attempts[101][10], submission[101];   //variables to calculate penalties,
      for (int v = 1; v <= 100; v++){                    //determine attempts, and whether there was a submission or not
         for (int u = 1; u <= 9; u++){
            penalty[v][u] = 0;     //assign all of them to zero
            attempts[v][u] = 0;
            submission[v] = 0;
         }
      }
      while (true){              //while not a blank line
         getline(cin, checker);    //reading a line
```

```
   int team = 0, problem = 0, time = 0;     //variables for team number, problem number, time and decision
   char decision;
   if (checker.length() == 0){   //if blank line break
      break;
   }
   else{                    //else convert line to team number, problem number, time and decision
      int j = 1;
      team = checker[0] - 48;
      if (checker[j] != ' '){
         team = team * 10 + checker[j] - 48;
         j += 1;
         if (checker[j] != ' '){
            team = team * 10 + checker[j] - 48;
            j += 1;
         }
      }
      j += 1;
      problem = checker[j] - 48;
      j += 2;
      time = checker[j] - 48;
      j += 1;
      while (checker[j] != ' '){
         time = time * 10 + checker[j] - 48;
         j += 1;
      }
      j += 1;
      decision = checker[j];
      if (penalty[team][problem] == 0){    //if there is no penalty means that the problem still not solved
         if (decision == 'C'){          //if correct then compute time and assign to penalty
            penalty[team][problem] = time + 20 * attempts[team][problem];
         }
         else if (decision == 'I'){      //if incorrect increase attempts number by one, later used to calculate penalty
            attempts[team][problem] += 1;
         }
         submission[team] += 1;          //if there is any decision then that team submitted something
      }
   }
}
int problems[101][4];    //array to compute number of solved problems
for (int x = 1; x <= 100; x++){
   problems[x][0] = 0;  //number of solved problems
   problems[x][1] = 0;  //total time with penalty minutes
   problems[x][2] = x;  //team number
}
for (int t = 1; t <= 100; t++){
   if (submission[t] == 0){    //if there was not any submission then assign to -1
      problems[t][0] = -1;
   }
   else{
      for (int s = 1; s <= 9; s++){     //computing solved problems and time
         if (penalty[t][s] != 0){
            problems[t][0] += 1;
            problems[t][1] += penalty[t][s];
         }
      }
   }
}
for (int t = 1; t <= 100; t++){    //sorting by number of problems, time and team number
```

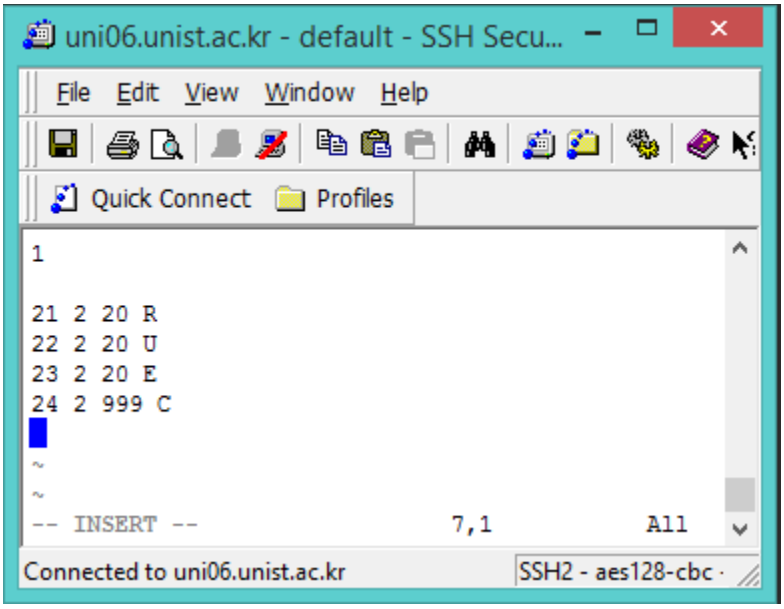
```
            for (int s = t + 1; s <= 100; s++){
                if (problems[t][0] < problems[s][0]){
                    int temp;
                    temp = problems[t][0];
                    problems[t][0] = problems[s][0];
                    problems[s][0] = temp;
                    temp = problems[t][1];
                    problems[t][1] = problems[s][1];
                    problems[s][1] = temp;
                    temp = problems[t][2];
                    problems[t][2] = problems[s][2];
                    problems[s][2] = temp;
                }
                else if (problems[t][0] == problems[s][0] && problems[t][1] > problems[s][1]){
                    int temp;
                    temp = problems[t][0];
                    problems[t][0] = problems[s][0];
                    problems[s][0] = temp;
                    temp = problems[t][1];
                    problems[t][1] = problems[s][1];
                    problems[s][1] = temp;
                    temp = problems[t][2];
                    problems[t][2] = problems[s][2];
                    problems[s][2] = temp;
                }
                else if (problems[t][0] == problems[s][0] && problems[t][1] == problems[s][1] && problems[t][2] > problems[s][2]){
                    int temp;
                    temp = problems[t][0];
                    problems[t][0] = problems[s][0];
                    problems[s][0] = temp;
                    temp = problems[t][1];
                    problems[t][1] = problems[s][1];
                    problems[s][1] = temp;
                    temp = problems[t][2];
                    problems[t][2] = problems[s][2];
                    problems[s][2] = temp;
                }
            }
        }
        for (int t = 1; t <= 100; t++){    //printing output
            if (problems[t][0] != -1){
                cout << problems[t][2] << " " << problems[t][0] << " " << problems[t][1] << endl;
            }
        }
    }
    return 0;
}
```

3. Sample input

First sample input

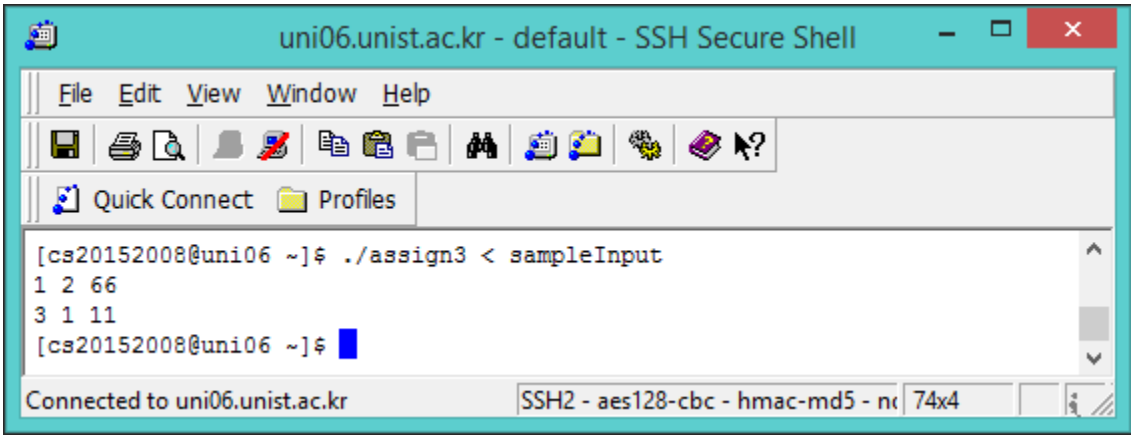


Second sample input



4. Sample output

First sample output



Second sample output