P1786 帮贡排序

题目描述

目前帮派内共最多有一位帮主，两位副帮主，两位护法，四位长老，七位堂主，二十五名精英，帮众若干。

现在absi2011要对帮派内几乎所有人的职位全部调整一番。他发现这是个很难的事情。于是要求你帮他调整。

他给你每个人的以下数据：

他的名字(长度不会超过30)，他的原来职位，他的帮贡，他的等级。

他要给帮贡最多的护法的职位，其次长老，以此类推。

可是，乐斗的显示并不按帮贡排序而按职位和等级排序。

他要你求出最后乐斗显示的列表(在他调整过职位后)：职位第一关键字，等级第二关键字。

注意：absi2011无权调整帮主、副帮主的职位，包括他自己的(这不是废话么..)

他按原来的顺序给你(所以，等级相同的，原来靠前的现在也要靠前，因为经验高低的原因，但此处为了简单点省去经验。)

输入格式

第一行一个数n，表示星月家园内帮友的人数。

下面n行每行两个字符串两个整数，表示每个人的名字、职位、帮贡、等级。

输出格式

一共输出n行，每行包括排序后乐斗显示的名字、职位、等级。

输入样例

```
9
DrangonflyKang BangZhu 100000 66
RenZaiJiangHu FuBangZhu 80000 60
absi2011 FuBangZhu 90000 60
BingQiLingDeYanLei HuFa 89000 58
Lcey HuFa 30000 49
BangYou3 ZhangLao 1000 1
BangYou1 TangZhu 100 40
BangYou2 JingYing 40000 10
BangYou4 BangZhong 400 1
```

输出样例

```
DrangonflyKang BangZhu 66
RenZaiJiangHu FuBangZhu 60
absi2011 FuBangZhu 60
BingQiLingDeYanLei HuFa 58
BangYou2 HuFa 10
Lcey ZhangLao 49
BangYou1 ZhangLao 40
BangYou3 ZhangLao 1
BangYou4 ZhangLao 1
```

解析

本题的一个坑点就是隐含的需要我们进行两次排序。第一次，我们按照帮贡和职位将所有角色进行排序。但是，这个时候很可能出现帮贡高的角色等级较低，但是会显示在前排。因此，第二轮排序则需要按照等级进行排序，将相同职位的等级高的优先显示。

编码

```cpp
#include <bits/stdc++.h>

using namespace std;

//无穷大
const long long inf = INT_MAX;
//各个职位的名字，按照从大到小的顺序排列
const string JobTitle[] = {"BangZhu", "FuBangZhu",
                           "HuFa", "ZhangLao",
                           "TangZhu", "JingYing", "BangZhong"};

//这个数组存各个职位的大小关系（职位小的值小，职位大的值大）
const int JobRank[] = {7, 6, 5, 4, 3, 2, 1};

//按照从大到小的顺序，每个职位需排到多少名额
const long long JobNum[] = {1, 3, 5, 9, 16, 41, inf};
```

```cpp
//一个结构体，存每个人的信息
struct player {
    //名称和职位
    string name, work;
    //帮贡和等级
    long long help, level;
    //在输入中排的位置
    int index;
    //职位的排名
    int rank;
} arr[150];

//总人数
int n;

//第一遍排序要用的函数
bool com(player a, player b) {
    if (a.help == b.help) {
        //帮贡相同，按第二关键字输入顺序排
       return a.index < b.index;
    } else {
        //按帮贡排
       return a.help > b.help;
    }
}

//第二遍排序
bool com2(player a, player b) {
    if (a.work == b.work) {//第一关键字相等
       if (a.level == b.level) {
            //第二关键字也相等，按输入顺序
          return a.index < b.index;
        } else {//按第二关键字等级排
          return a.level > b.level;
        }
    } else {//按第一关键字职位排
       return a.rank > b.rank;
    }
}


int main() {
    //初始化map
    cin >> n;
    for (int i = 1; i <= n; ++i) {
        //输入每个人的信息
       cin >> arr[i].name >> arr[i].work >> arr[i].help
        >> arr[i].level;
```

```
            //是帮主，帮贡设为无穷大，保证在最前
        if (arr[i].work == JobTitle[0]) {
                arr[i].help = inf;
            }
                //是副帮主，帮贡比无穷大小一点，保证在帮主后，其他人前
        else if (arr[i].work == JobTitle[1]) {
                arr[i].help = inf - 5;

            }
            arr[i].index = i;
            //第i个输入的
    }
     //第一遍排序，按照帮贡进行排序
    sort(arr + 1, arr + n + 1, com);

     //将每个人的职位分配好
    for (int i = 1; i <= n; ++i) {
            //根据排名分别title
            for (int j = 0; j < 7; ++j) {
                //排名符合标准
              if (i <= JobNum[j]) {
                    //分配职务
                 arr[i].work = JobTitle[j];
                    arr[i].rank = JobRank[j];
                    //分配成功就跳出，避免重复分配
                 break;
                }
            }
     }
     //在这一次排序，我们将忽略帮贡，直接按照等级进行排序
    sort(arr + 1, arr + n + 1, com2);

     //输出结果
    for (int i = 1; i <= n; ++i) {
            cout << arr[i].name << " " << arr[i].work << " " <<
            arr[i].level << endl;
     }
     return 0;
}
```