# AcWing-4672：布料排序

每块布料包含三种属性：

- 颜色（$C$），一个由小写英文字母组成的字符串，表示布料的颜色。
- 耐久性（$D$），一个整数，表示布料的耐久性。
- 唯一标识符（$U$），一个整数，表示布料的 $ID$。

给定 $N$ 块布料，阿达和查尔斯需要对布料进行排序。

阿达按照颜色（$C$）字典序升序的顺序对布料进行排序，颜色相同的布料按唯一标识符（$U$）升序的顺序进行排序。

查尔斯按照耐久性（$D$）升序的顺序对布料进行排序，耐久性相同的布料按唯一标识符（$U$）升序的顺序进行排序。

请你计算，有多少块布料满足，无论是阿达还是查尔斯对布料进行排序，其最终顺位排名都相同。

**输入格式**

第一行包含整数 $T$，表示共有 $T$ 组测试数据。

每组数据第一行包含整数 $N$。

接下来 $N$ 行，每行包含一个字符串 $C_i$，一个整数 $D_i$，一个整数 $U_i$。

**输出格式**

每组数据输出一个结果，每个结果占一行。

结果表示为 `Case #x: y`，其中 $x$ 为组别编号（从 1 开始），$y$ 为满足条件的布料数量。

**数据范围**

$1 \le T \le 100$,
$1 \le N \le 10^3$,
$1 \le |C_i| \le 10$,
$1 \le D_i \le 100$,
$1 \le U_i \le 10^3$,
$C_i$ 只包含小写英文字母。
$U_i$ 两两不同。

## 输入样例1：

| | |
|---|---|
| 时/空限制： | 1s / 64MB |
| 总通过数： | 793 |
| 总尝试数： | 1294 |
| 来源： | Google Kickstart2022 Round F Problem A |
| 算法标签 | |

```
3
2
blue 2 1
yellow 1 2
2
blue 2 1
brown 2 2
1
red 1 1
```

### 输出样例1：

```
Case #1: 0
Case #2: 2
Case #3: 1
```

### 样例1解释

在 Case 1 中，按颜色排序时，布料（用唯一标识符表示）顺序为 1,2；按耐久性排序时，布料顺序为 2,1，所以 0 块布料具有相同的排名。

在 Case 2 中，按颜色排序时，布料（用唯一标识符表示）顺序为 1,2；按耐久性排序时，布料顺序为 1,2，所以 2 块布料具有相同的排名，这里需要注意，两块布料具有相同的耐久性，所以在查尔斯进行排序时，他将具有更小 ID 的 1 号布料排在前面。

在 Case 3 中，只有 1 块布料，所以无论如何其排名都不会有变化。

### 输入样例2：

```
1
5
blue 1 2
green 1 4
orange 2 5
red 3 6
yellow 3 7
```

### 输出样例2：

```
Case #1: 5
```

本题的主要思路就是实现两种排序方式，并遍历数组查看是否在同一位置有相同的元素。可以自定义排序函数`<`小于号，并使用`sort`函数进行排序。

时间复杂度为`O(nlogn)`，主要体现在排序算法上。

Code：

```cpp
//
//  main.cpp
//  4672-布料排序
//
//  Created by MacBook Pro on 2023/7/25.
//

#include <iostream>
#include <algorithm>
using namespace std;

struct node{
    string c;
    int d;
    int id;
}nums1[1005],nums2[1005];

bool sort1(node& x1,node& x2){
    if(x1.c==x2.c){
        return x1.id<x2.id;
    }
    return x1.c<x2.c;
}

bool sort2(node& x1,node& x2){
    if(x1.d==x2.d){
        return x1.id<x2.id;
    }
    return x1.d<x2.d;
}

int main() {
    int T;
    scanf("%d",&T);
    for(int cases=1;cases<=T;cases++){
        int n;
        scanf("%d",&n);
        for(int i=0;i<n;i++){
            cin>>nums1[i].c;
            scanf("%d%d",&nums1[i].d,&nums1[i].id);
            nums2[i].c=nums1[i].c;
            nums2[i].d=nums1[i].d;
            nums2[i].id=nums1[i].id;
        }
        //排序方法
        sort(nums1,nums1+n,sort1);
        sort(nums2,nums2+n,sort2);
        //检测重合个数
        int ans=0;
```

```
        for(int i=0;i<n;i++){
            if(nums1[i].id==nums2[i].id) ans++;
        }
        cout<<"Case #"<<cases<<": "<<ans<<endl;
    }
}
```

代码提交状态：Accepted