D Choice III

Description

据说 sd0061 要选队友了？Thor 是一个十分八卦的人，他马上向 DPY 要了一份名单，名单里面 n 个队伍的信息，第 i 个队伍里面有 m[i]个人，每个人都有一个能力值 A[j]。现在，假如 sd0061 要挑两个人作队友，这两个队友的能力值的和一定要超过 s。而且，sd0061 不会在同一个队伍里面挑人。Thor 现在想要知道 sd0061 的选择方案有多少种。

Input

多组数据。
每组数据有 n+1 行。
第一行有 2 个整数 n,s(1<=n<=100,0<=s<=10000)。
接下来 n 行，每行第一个数为 m[i]，之后有 m[i]个整数，代表每个人的能力值 A[j](1<=m[i]<=100,0<=A[j]<=10000)。

Output

对于每组数据输出一行一个整数 Ans 表示 Thor 的选择方案数。

Sample Input

```
3 2
1 2
1 2
1 2
1 1
3 1 1 1
```

Sample Output

```
3
0
```

Hint

其实以上全都是 Thor 口胡的。
时限:100ms

解题分析

若没有题目里面的不能在同一个队里，那么题目就和 ChoiceII 是同一道题目。
如果我们用 Ability[i]表示 i 的能力，Belong[i]表示 i 所属的队伍，那么我们想要求的就是所有的 Ability[i]+Ability[j]>k && Belong[i] != Belong[j]的对数 sum。
如果只是求 Ability[i]+Ability[j]>k 的对数那么我们可以用 ChoiceII 里面的办法。复杂度是 O(nlogn)。
如果我们能够知道 Ability[i]+Ability[j]>k && Belong[i] == Belong[j]的对数 sub，那么答案不就是 sum-sub 了么。而对于 sub，就是对每个队用 ChoiceII 的做法得到的答案的和。
所以做法就是

1. 读入每个队的数据——O(n*m)
2. 把每个队伍排序——O(n*m^2)
3. 对每个队伍用 ChoiceII 的方法求出答案加到 sub 里面——O(n*1)
4. 将每一个队伍的数据记录到另一个数组 que 里面——O(n*m*m)
5. 处理出 sub 之后，对 que 数组进行 ChoiceII 的方法求出 sum——O(n*mlog(n*m))
6. 答案即 sum-sub

参考代码

```
#include <stdio.h>
const int M = 100+10;
const int N = 10000+10;
int n,k;
int cnt,temp[M];
int sum,data[N];
int aux[N];
int sub;

int count(int *a,int cnt) {     //对于一个排好序的长度为cnt 的数组a，统计其中相加大于k 的对数
    int answer = 0;
    int l = 0,r = cnt-1;
    while(l<r) {
        if(a[l]+a[r]<=k) {
            ++l;
        }
        else {
            answer += (r--)-l;
        }
    }
    return answer;
}
```

```
void sort(int* a,int cnt) {     //简单的排序
    for(int i = 0 ; i < cnt ; ++i) {
        for(int j = i+1 ; j < cnt ; ++j) {
            if(a[i]>a[j]) {
                int t = a[i];
                a[i] = a[j];
                a[j] = t;
            }
        }
    }
}

void Union(int *a,int cnt_a,int *b,int cnt_b,int *data,int &sum) {//对于两个排好序的数组 a,b 合并成另一个排好序的数组 data
    sum = 0;
    int i,j;
    for(i=0,j=0 ; i < cnt_a && j < cnt_b ; ) {
        data[sum++] = a[i]<b[j]?a[i++]:b[j++];
    }
    while(i<cnt_a)data[sum++] = a[i++];
    while(j<cnt_b)data[sum++] = b[j++];
}

int main() {
    while(scanf("%d%d",&n,&k)!=EOF) {
        sum = sub = 0;
        for(int i = 0 ; i < n ; ++i) {
            scanf("%d",&cnt);
            for (int j = 0; j < cnt; ++j)
            {
                scanf("%d",&temp[j]);
            }
            sort(temp,cnt);
            for(int j = 0 ; j < sum ; ++j)
                aux[j] = data[j];
            Union(aux,sum,temp,cnt,data,sum);
            sub += count(temp,cnt);
        }

        printf("%d\n",count(data,sum)-sub);
    }
}
```