

# 专题分享

---

CJ FHR

# 抛砖引玉 1.0

---

给定数列  $A$ ，求数列中比  $A_i$  小的数的个数。

不好意思走错片场了。

# 抛砖引玉 2.0

---

LIS 问题：求最长上升 / 下降 / 不上升 / 不下降子序列。

惊哥表示可以闭上显示器打出来。

# 抛砖引玉 3.0

---

SDOI2011 拦截导弹。 CJOI 陌上菊开

CDQ 分治。

赏心悦目。

# 总结

---

一维排序

二维树状数组

三维 CDQ 分治

# 抛砖引玉 4.0

---

HZOI2016 偏序

题面真是裸到爆炸。

给定一个有 $n$ 个元素的序列，元素编号为 $1 \sim n$ ，每个元素有三个属性 $a, b, c$ ，求序列中满足 $i < j$ 且 $a_i < a_j$ 且 $b_i < b_j$ 且 $c_i < c_j$ 的数对 $(i, j)$ 的个数。

## 【输入格式】

第一行一个整数 $n$ ，表示序列长度。

第二行 $n$ 个整数，分别表示 $a_1 \sim a_n$ 。

第三行 $n$ 个整数，分别表示 $b_1 \sim b_n$ 。

第四行 $n$ 个整数，分别表示 $c_1 \sim c_n$ 。

对于100%的数据， $1 \leq n \leq 50000$ ，保证所有的 $a_i$ 、 $b_i$ 、 $c_i$ 分别组成三个 $1 \sim n$ 的排列。

# 抛砖引玉 4.0

---

4 维了。。。

CDQ 套 CDQ ?

树套树 ?

KD 树 ?

欢迎分享.....

# 如果，还来？

七

## 2580. [HZOI 2015]偏序 II

★★★★☆

输入文件：

partial\_order\_two.in

输出文件：

partial\_order\_two.out

简单对比

时间限制：3.5 s

内存限制：256 MB

### 【题目描述】

给定一个有 $n$ 个元素的序列，元素编号为 $1 \sim n$ ，每个元素有四个属性 $a, b, c, d$ ，求序列中满足 $i < j$ 且 $a_i < a_j$ 且 $b_i < b_j$ 且 $c_i < c_j$ 且 $d_i < d_j$ 的数对 $(i, j)$ 的个数。

### 【输入格式】

第一行一个整数 $n$ ，表示序列长度。

第二行 $n$ 个整数，分别表示 $a_1 \sim a_n$ 。

第三行 $n$ 个整数，分别表示 $b_1 \sim b_n$ 。

第四行 $n$ 个整数，分别表示 $c_1 \sim c_n$ 。

第五行 $n$ 个整数，分别表示 $d_1 \sim d_n$ 。

对于100%的数据， $1 \leq n \leq 50000$ ，保证所有的 $a_i$ 、 $b_i$ 、 $c_i$ 、 $d_i$ 分别组成四个 $1 \sim n$ 的排列。

请注意常数因子带来的程序效率上的影响。

# 可是，继续？

卡

## 2639. [HZOI 2015] 偏序++

★★★

输入文件：

partial\_order\_plus.in

输出文件：

partial\_order\_plus.out

简单对比

时间限制：4 s

内存限制：512 MB

### 【题目描述】

给定一个有  $n$  个元素的序列，元素编号为  $1 \sim n$ ，每个元素有  $k$  个属性  $p^1, p^2, p^3, \dots, p^k$ ，求序列中满足  $i < j$  且  $1 \leq t \leq k, p_i^t < p_j^t$  的数对  $(i, j)$  的个数。

### 【输入格式】

第一行两个整数  $n, k$ ，表示序列长度和属性个数。

接下来  $k$  行，每行  $n$  个整数，第  $t$  行表示  $n$  个元素的第  $p^t$  个属性。

对于100%的数据， $1 \leq n \leq 40000$ ， $k \leq 6$ 。保证对于所有元素的  $p^t$  属性组成一个  $1 \sim n$  的排列。

# 抛砖引玉 4.0

写得很烂的 KD 树.....

本地大概要 2.6S

Cogs 上只怕更加绝望

正在编译... 开始运行

点	结果	得分	运行时间	内存使用	返回
1	答案正确	10	0.002 s	18682 KB	0
2	答案正确	10	0.014 s	18682 KB	0
3	答案正确	10	0.092 s	18682 KB	0
4	超过时间限制	0	2.500 s	18682 KB	-1
5	超过时间限制	0	2.500 s	18682 KB	-1
6	超过时间限制	0	2.500 s	18682 KB	-1
7	超过时间限制	0	2.500 s	18682 KB	-1
8	超过时间限制	0	2.500 s	18682 KB	-1
9	超过时间限制	0	2.500 s	18682 KB	-1
10	超过时间限制	0	2.500 s	18682 KB	-1

# 如果，还来？

还是熟悉的配方

本地 3.2S

被卡成暴力分

正在编译... 开始运行

点	结果	得分	运行时间	内存使用	返回
1	答案正确	10	0.002 s	18682 KB	0
2	答案正确	10	0.013 s	18682 KB	0
3	答案正确	10	0.096 s	18682 KB	0
4	超过时间限制	0	3.500 s	18682 KB	-1
5	超过时间限制	0	3.500 s	18682 KB	-1
6	超过时间限制	0	3.500 s	18682 KB	-1
7	超过时间限制	0	3.500 s	18682 KB	-1
8	超过时间限制	0	3.500 s	18682 KB	-1
9	超过时间限制	0	3.500 s	18682 KB	-1
10	超过时间限制	0	3.500 s	18682 KB	-1

# 可是，继续？

还是原来的味道

???

原来询问只有 40000

所以过去了.....

所以 KD 树是优秀的暴力

正在编译... 开始运行

点	结果	得分	运行时间	内存使用	返回
1	答案正确	10	0.041 s	18682 KB	0
2	答案正确	10	0.061 s	18682 KB	0
3	答案正确	10	0.080 s	18682 KB	0
4	答案正确	10	1.923 s	18682 KB	0
5	答案正确	10	2.197 s	18682 KB	0
6	答案正确	10	2.370 s	18682 KB	0
7	答案正确	10	2.370 s	18682 KB	0
8	答案正确	10	2.667 s	18682 KB	0
9	答案正确	10	3.292 s	18682 KB	0
10	答案正确	10	2.888 s	18682 KB	0

# 如何解决多维偏序问题

---

**十维以上 不如  $n$  方  
暴力进队 乱搞 AC**

**如果维数稍微低一点，还是有优秀一点的做法的。**

# 以为这种东西不会考？

Kyle is a student of Programming Monkey Elementary School. Just as others, he is deeply concerned with his grades.

Last month, the school held an examination including five subjects, without any doubt, Kyle got a perfect score in every single subject.

There are  $n$  students took part in this examination(not including Kyle), and everyone got an integer between 1 to  $m$  as the score of one subject.

Now, looking at the grade table of these  $n$  students, Kyle wants to know how many students still did no better than him even if his scores are something else – Here, “no better” means there is no subject in which the student got strictly greater score than Kyle.

The first line of each test case contains two integers,  $n, m$  ( $n, m \leq 50,000$ ), which are the number of students and the perfect score of each subject.

In the next  $n$  lines, each line consists of five integers, indicating a student's scores.

Then one line follows. This line contains an integer  $q$  ( $q \leq 50,000$ ) indicating the number of queries.

In the next  $q$  lines, each line contains five integers as well, representing a query. Each query indicates a set of scores, and for each query, you should figure out that if Kyle's grade is this set of scores, how many students still did no better than him. But for the sake of security, only the first query is in its original form, and other queries are encrypted. To decrypt a query, you must let each integer in the query do xor operation with the answer of last query. It's guaranteed that all the decrypted queries contain integers between 1 and 50000.

标签

题目来源

ACM-ICPC国际大学生  
程序设计竞赛北京赛区  
(2015)网络赛

不仅出了  
而且还强制在线  
不会怎么办？  
(ノ`□')ノ へー

# 怎么优化

---

50000 , KD 树已经证实了轻松被卡。

数据结构 ( 不可能 , 不然早就被推广了 )

分治 (3 层 CDQ ? 想想就..... )

而且分治还不能被强制在线

# 暴力怎么优化

---

提示：当然是分块辣

分成 $\sqrt{n}$ 块，维护每块的东西？不好做。

单独考虑每一维。

# 暴力怎么优化

---

如果我们知道对于一个  $k$  维向量  $(p_1, p_2, \dots, p_k)$

对于每一维  $p_i$  , 比它小的数构成的集合是  $S_i$

那么答案集合就是  $S = S_1 \cap S_2 \cap \dots \cap S_k$  。

有种东西叫做 bitset 不知道大家写没写过.....

# Bitset 连击

---

bitset 是 C++ 的一种类型，需要头文件 `bitset`，可以看成是一个长长的 01 串集合。

`bitset<S>lj;` 申请一个有  $S$  个元素的 bitset。

`lj.set();/lj.reset();` 把 bitset 里面的所有元素赋为 1/0。

`lj.set(i);/lj.reset(i);` 把 bitset 里面的单个元素赋为 1/0。

`lj.count();` 返回集合中 1 的个数。

`lj.flip();/lj.flip(i);` 把所有 / 单个元素取反。

`lj[i];` 访问第  $i$  个元素。

支持两个 bitset 的按位与、或、异或等操作。

复杂度是  $O(\text{还算可以})/O(\text{跑得过})$ 。

# 如此一来问题就完美地被解决了

---

```
inline void solve(){  
    bitset<50000> ans;ans.set();  
    for(int i=1;i<=k;++i)  
        ans&=getset(k);  
    printf("%d\n",ans.count());  
}
```

谢谢大家！

getset() 怎么搞？

# 设想一下美好未来

---

如果我们知道第  $k$  维为  $i$  时，小于它的向量的编号集合。

可以用一个长度为  $n$  的 bitset 表示是吧。

查询的时候就直接 return 这个 bitset 就好了嘛。

# 面对一下残酷现实

---

```
bitset<50000>S[K][50000];
```

你当 bitset 不要空间啊.....

# 分块做法

---

因为是单独考虑每一维的集合的一个前缀。

所以块尾储存关于大小的前缀集合就可以了。

即：  $\text{Bit}[p][i]$  储存第  $p$  维权值小于等于  $i \cdot \text{block}$  的向量集合。

剩下的不到  $\sqrt{n}$  个东西就暴力  $\text{set}()$  一下就好了嘛。

所以 bitset 只要开块尾个 ( 根号  $n$  个 ) 储存信息。

```
bitset<50000>Bit[7][240];
```

经实际验证没 (bu) 问 (hui) 题 (zha) 。

下面给出代码实现：

# 暴力三连

---

```
for(int j=0;j<=k;++j){
    bitset<N>w;w.reset();
    for(int i=1;i<=n;++i){
        w.set(bin[j][i]);
        if(bel[i]!=bel[i+1])Bit[j][bel[i]]=w;
    }
}
```

预处理

```
for(int i=1;i<=n;++i){
    bitset<N>ans;ans.set();
    for(int j=0;j<=k;++j)
        ans&=getbst(j,f[j][i]-1);
    Ans+=ans.count();
}
```

枚举

```
inline bitset<N> getbst(int p,int val){
    int st=bel[val]-1;bitset<N>ans;ans.reset();
    if(st<0)return ans;ans=Bit[p][st];
    for(int i=st*bal+1;i<=val;++i)ans.set(bin[p][i]);
    return ans;
}
```

求集合

讲完了 谢谢

---

# 附录：运行效率

## 四维偏序

点	结果	得分	运行时间
1	答案正确	10	0.002 s
2	答案正确	10	0.000 s
3	答案正确	10	0.097 s
4	超过时间限制	0	2.500 s
5	超过时间限制	0	2.500 s
6	超过时间限制	0	2.500 s
7	超过时间限制	0	2.500 s
8	超过时间限制	0	2.500 s
9	超过时间限制	0	2.500 s
10	超过时间限制	0	2.500 s

点	结果	得分	运行时间
1	答案正确	10	0.004 s
2	答案正确	10	0.018 s
3	答案正确	10	0.066 s
4	答案正确	10	0.689 s
5	答案正确	10	0.687 s
6	答案正确	10	0.696 s
7	答案正确	10	0.687 s
8	答案正确	10	0.697 s
9	答案正确	10	0.696 s
10	答案正确	10	0.692 s

# 附录：运行效率

## 五维偏序

点	结果	得分	运行时间
1	答案正确	10	0.002 s
2	答案正确	10	0.013 s
3	答案正确	10	0.096 s
4	超过时间限制	0	3.500 s
5	超过时间限制	0	3.500 s
6	超过时间限制	0	3.500 s
7	超过时间限制	0	3.500 s
8	超过时间限制	0	3.500 s
9	超过时间限制	0	3.500 s
10	超过时间限制	0	3.500 s

点	结果	得分	运行时间
1	答案正确	10	0.003 s
2	答案正确	10	0.024 s
3	答案正确	10	0.084 s
4	答案正确	10	0.890 s
5	答案正确	10	0.887 s
6	答案正确	10	0.887 s
7	答案正确	10	0.934 s
8	答案正确	10	0.966 s
9	答案正确	10	0.885 s
10	答案正确	10	0.884 s

# 附录：运行效率

## K 维偏序

点	结果	得分	运行时间
1	答案正确	10	0.041 s
2	答案正确	10	0.061 s
3	答案正确	10	0.080 s
4	答案正确	10	1.923 s
5	答案正确	10	2.197 s
6	答案正确	10	2.370 s
7	答案正确	10	2.370 s
8	答案正确	10	2.667 s
9	答案正确	10	3.292 s
10	答案正确	10	2.888 s

点	结果	得分	运行时间
1	答案正确	10	0.032 s
2	答案正确	10	0.066 s
3	答案正确	10	0.099 s
4	答案正确	10	0.270 s
5	答案正确	10	0.299 s
6	答案正确	10	0.314 s
7	答案正确	10	0.317 s
8	答案正确	10	0.391 s
9	答案正确	10	0.543 s
10	答案正确	10	0.820 s

# 附录 2 ：手写 bitset

Cogs 冲榜全靠它 ~

其实就是简单的压位。

```
inline void Init(){for(int i=0;i<65536;++i) Table[i]=Table[i>>1]+(i&1);}
inline int Query(const unsigned int& x){
    return Table[x>>16]+Table[x&65535];
}
struct BITSET{
    unsigned int Set[BLOCK],size;
    BITSET(){size=0;}
    inline void operator |= (const int& a){
        int p=a/M; int q=a-M*p;
        Set[p]|=(1u<<q);
        size++;
    }
    inline void operator &= (const BITSET& a){
        size=0;
        for(int i=0;i<BLOCK;++i) Set[i]&=a.Set[i],size+=Query(Set[i]);
    }
    inline void empty(){
        for(int i=0;i<BLOCK;++i) Set[i]=0;
        size=0;
    }
}Bit[7][bal+1],p,q,np;
```