【题目描述】

有 n 个二元组 (a_i, b_i)，编号为 1 到 n。

有一个初始为空的栈 S，向其中加入元素 (a_i, b_i) 时，先不断弹出栈顶元素直至栈空或栈顶元素 (a_j, b_j) 满足 $a_i \neq a_j$ 且 $b_i < b_j$，然后再将其加入栈中。

如果一个二元组入栈后栈内只有这一个元素，则称该二元组是“成功的”。

有 q 个询问 $[l_i, r_i]$，表示若将编号在 $[l_i, r_i]$ 中的二元组按编号从小到大依次入栈，会有多少个二元组是“成功的”。

询问之间相互独立。

【输入格式】

从文件 ***stack.in*** 中读入数据。

第一行两个正整数 n, q。

第二行 n 个正整数表示 a_i。

第三行 n 个正整数表示 b_i。

接下来 q 行，每行两个正整数 l_i, r_i，表示一组询问。

【输出格式】

输出到文件 ***stack.out*** 中。

q 行，每行一个自然数表示一组询问的答案。

【样例输入】

```
10 4
3 1 3 1 2 3 3 2 1 1
10 10 2 9 7 5 4 7 6 1
1 4
7 8
7 10
1 8
```

【样例输出】

```
3
2
2
3
```

【样例解释】

以第一次询问 $[1,4]$ 为例。

一开始栈为 $\{\}$。

加入 1 号二元组后栈为 $\{(3,10)\}$，栈中只有一个元素，该二元组是“成功的”。

加入 2 号二元组 $(1,10)$ 时，栈顶的 $(3,10)$ 的 b 值不大于 2 号二元组的，因此弹栈。此时栈空，2 号二元组入栈，栈为 $\{(1,10)\}$，该二元组是“成功的”。

加入 3 号二元组 $(3,2)$，此时栈顶元素与之 a 值不同，b 值比它更大，因而不需要弹栈，直接将 3 号二元组入栈，栈为 $\{(1,10),(3,2)\}$，栈中有多个元素，该二元组不是“成功的”。

加入 4 号二元组 $(1,9)$，此时栈顶元素 $(3,2)$ 的 b 值比它小，弹栈。弹栈后栈顶元素 $(1,10)$ 与 $(1,9)$ 的 a 值相同，继续弹栈。此时栈空，4 号二元组入栈，栈为 $\{(1,9)\}$，该二元组是“成功的”。

共有 3 个二元组是“成功的”，因而答案为 3。

【样例 2,3,4】

见选手目录下的 ***stack/stack*.in*** 与 ***stack/stack*.ans*** 。

【数据范围与提示】

对于所有测试点：$1 \le n,q \le 5 \times 10^5$，$1 \le a_i,b_i \le n$，$1 \le l_i \le r_i \le n$。

每个测试点的具体限制见下表：

测试点编号	特殊限制
$1 \sim 3$	$n,q \le 1000$
$4 \sim 6$	$n \le 5000$
$7 \sim 10$	$n,q \le 10^5$
$11 \sim 12$	$b_i = n-i+1$
$13 \sim 15$	$a_i = i$
$16 \sim 20$	无

【题目描述】

有 n 个人正在打模拟赛，模拟赛有 n 道题目。

有两人都会的题目并且没有人会的题目包含另一个人时，两者之间才会讨论。

（定义第 i 个人会的题目的集合为 S_i ，即当 $S_x \cap S_y \neq \varnothing \wedge S_x \nsubseteq S_y \wedge S_y \nsubseteq S_x$ 时，第 x 人和第 y 人会讨论）

为了让模拟赛的效果更好，希望你可以找出一对会讨论的人或判断不存在。

【输入格式】

从文件 ***discuss.in*** 中读入数据。

第一行一个正整数 T 表示数据组数，对于每组数据：

第一行一个正整数 n 表示人数和题目数量。

接下来 n 行，第 i 行第一个自然数 k_i 表示第 i 个人会 k_i 道题。接下来 k_i 个正整数，每个数 x 表示第 i 个人会第 x 道题。

【输出格式】

输出到文件 ***discuss.out*** 中。

对于每组数据：

如果没有会讨论的人，输出 NO。

否则第一行输出 YES，第二行输出两个正整数 x 和 y，表示第 x 人和第 y 人会讨论。

如果有多种方案，输出任意一种即可。

【样例 1 输入】

```
2
5
4 1 2 3 5
3 1 2 3
2 1 2
1 1
1 4
4
3 1 2 3
3 2 3 4
0
4 1 2 3 4
```

【样例 1 输出】

NO

YES

1 2

【样例 2】

见选手目录下的 ***discuss/discuss2.in*** 与 ***discuss/discuss2.ans***。

【数据范围与提示】

对于所有测试点：令一组数据中 $m = \sum k_i$，则 $1 \le T \le 5$，$1 \le \sum n \le 10^6$，$1 \le \sum m \le 2 \times 10^6$，$0 \le k_i \le n$。

每个测试点的具体限制见下表：

测试点编号	特殊限制
1	$n \le 300$
$2 \sim 3$	$n \le 1000$
4	$n \le 5000$
$5 \sim 7$	$n \le 5 \times 10^4$
8	$k_i \le 10$
$9 \sim 10$	无

【题目描述】

有一个 $m \times n$ 的数组 $a_{i,j}$。

定义：

$$f(i,j) = \min_{k=1}^{m}(a_{k,i} + a_{k,j}) + \max_{k=1}^{m}(a_{k,i} + a_{k,j})$$

你需要求出 $\sum_{i=1}^{n}\sum_{j=1}^{n} f(i,j)$。

【输入格式】

从文件 ***sort.in*** 中读入数据。

第一行两个正整数 m, n。

接下来 m 行，每行 n 个正整数表示 $a_{i,j}$。

【输出格式】

输出到文件 ***sort.out*** 中。

一行一个正整数，表示答案。

【样例 1 输入】

```
3 5
1 7 2 2 7
9 10 4 10 3
7 7 8 10 2
```

【样例 1 输出】

```
564
```

【样例 1 解释】

以 $f(3,5)$ 为例:

$$\begin{aligned} f(3,5) &= \max(a_{1,3}+a_{1,5}, a_{2,3}+a_{2,5}, a_{3,3}+a_{3,5}) + \min(a_{1,3}+a_{1,5}, a_{2,3}+a_{2,5}, a_{3,3}+a_{3,5}) \\ &= \max(9,7,10) + \min(9,7,10) \\ &= 10+7 \\ &= 17 \end{aligned}$$

下面给出 $f(i,j)$ 的数表，第 i 行第 j 列表示 $f(i,j)$:

20	27	18	22	20
27	34	24	29	23
18	24	20	22	17
22	29	22	24	22
20	23	17	22	18

它们的和是答案 564。

【样例 2,3,4】

见选手目录下的 ***sort/sort*.in*** 与 ***sort/sort*.ans***。

【数据范围与提示】

对于所有测试点:

$2 \le m \le 4$，$1 \le n \le 2 \times 10^5$，$1 \le a_{i,j} \le 2 \times 10^5$。

每个测试点的具体限制见下表:

测试点编号	$m=$	$n \le$
1	4	3000
2	2	10^5
3 ~ 5	3	10^5
6 ~ 7	4	5×10^4
8 ~ 9		10^5
10		2×10^5

我们把询问 $[l,r]$ 过程中"成功"的对叫关键点。

观察询问 $[l,r]$ 相对于 $[l+1,r]$ 关键点集合的变化：

1.离 l 最近的点若与之 a 值不同，且 b 值比 b_l 小，那么它不再是关键点。去掉关键点后重复这一过程。

2.将 l 加入关键点集合。

对于一个 r，我们记 p_i 表示最小的 l，使得询问 $[l,r]$ 时 i 还是关键点。模拟上述过程可以 $O(n)$ 求出每个 p_i。发现对于不同的 r, p_i 是不变的。故可以令 $r=n$，$O(n)$ 求出每个 p_i。

此时我们发现答案就是 $[l,r]$ 中 $p_i \le l$ 的 i 个数，差分后二维数点即可。

复杂度 $O((n+q)\log n)$。

$10-30$ 分做法

对任意两个集合判断，可以使用 `bitset` ，时间复杂度 $O(\frac{n^3}{w})$ 。

100 分做法

首先将所有集合按集合大小从小到大排序。

考虑枚举位置 x，对于所有包含 x 的考虑：

如果相邻两个没有包含关系，那么它们一定满足条件。

这样做用 `bitset` 加速是 $O(\frac{n^3}{w})$ 的。考虑维护它们的包含关系。

对于每个 x ，包含关系都是一条链，需要维护的包含关系一定是树形结构，否则已经找出了一对满足条件的集合。

考虑把这条链加到森林上，如果能够成功加入那么考虑下一个位置 x 。

否则从下至上，第一个不和森林匹配的点和它的前驱一定满足条件。

只会判断 $O(n)$ 次包含关系，用 `bitset` 判断时间复杂度 $O(m+\frac{n^2}{w})$ 。

可以通过 70 分。

发现对于这个森林，只会判断父亲是否包含儿子，所以把儿子包含的所有题目向父亲的集合中查询，用哈希表维护总复杂度是 $O(m)$ 。

因此每组数据的复杂度为 $O(n+m)$ 。

为了方便描述，记 $g(i,j,k)$ 表示 $a_{k,i}+a_{k,j}$。

$k=2$ 和 $n\le 3000$ 的 20 分很容易拿到，不再赘述。

算法一

考虑 $k=3$ 时怎么做。

对于每个排列 p_1,p_2,p_3，统计满足 $g(i,j,p_1)<g(i,j,p_2)<g(i,j,p_3)$ 的对 (i,j) 的 $a_{p_2,i}+a_{p_2,j}$ 之和即可（如果不等式不会取到等号的话）。

发现 $g(i,j,p_1)<g(i,j,p_2)$ 可以写成 $a_{p_1,i}-a_{p_2,i}<a_{p_2,j}-a_{p_1,j}$。从而可以用 $3!$ 次二维偏序解决原问题。

但对于不等式取到等号的对，不管怎么算都会漏或者重。一种简单的处理方式是把每行乘 3，然后第 i 行加 $i-1$ 来避免取等。

结合前 20 分对应的算法可得 50 分。

算法二

对于 $k=4$ 的情况，记 $h(i,j,k)$ 表示三个满足 $s\neq k$ 的 $g(i,j,s)$ 的中位数。

可以注意到答案等于

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{m}\left(g(i,j,k)-\frac{1}{2}h(i,j,k)\right)$$

g 的和是容易计算的，h 的和套用 $k=3$ 的做法即可。

这是一个大常数 $O(n\log n)$ 做法（瓶颈是 24 次二维偏序）。

直接写可能常数稍大有点卡，不过把排列 p_1,p_2,p_3 和 p_3,p_2,p_1 在一个归并排序里做就可以卡掉近一半常数，可以保证通过。

Bonus

测试点 $6-9$ 留给常数较大的正解。

也可以用 $min-max$ 容斥将 $k=4$ 的问题转化为 $k=3$ 的问题，细节处理可能与本题解不同。

关于我们

北京高考在线创办于 2014 年，隶属于北京太星网络科技有限公司，是北京地区极具影响力的中学升学服务平台。主营业务涵盖：北京新高考、高中生涯规划、志愿填报、强基计划、综合评价招生和学科竞赛等。

北京高考在线旗下拥有网站门户、微信公众平台等全媒体矩阵生态平台。平台活跃用户 40W+，网站年度流量数千万量级。用户群体立足于北京，辐射全国 31 省市。

北京高考在线平台一直秉承 '' 精益求精、专业严谨 '' 的建设理念，不断探索“K12 教育+互联网+大数据”的运营模式，尝试基于大数据理论为广大中学和家长提供新鲜的高考资讯、专业的高考政策解读、科学的升学规划等，为广大高校、中学和教科研单位提供“衔接和桥梁纽带”作用。

平台自创办以来，为众多重点大学发现和推荐优秀生源，和北京近百所中学达成合作关系，累计举办线上线下升学公益讲座数百场，帮助数十万考生顺利通过考入理想大学，在家长、考生、中学和社会各界具有广泛的口碑影响力

未来，北京高考在线平台将立足于北京新高考改革，基于对北京高考政策研究及北京高校资源优势，更好的服务全国高中家长和学生。