# A1.1.范围查询(Range)

**Descriptioin**

Let S be a set of n integral points on the x-axis. For each given interval [a, b], you are asked to count the points lying inside.

**Input**

The first line contains two integers: n (size of S) and m (the number of queries).

The second line enumerates all the n points in S.

Each of the following m lines consists of two integers a and b and defines an query interval [a, b].

**Output**

The number of points in S lying inside each of the m query intervals.

**Example**

Input

5 2

1 3 7 9 11

4 6

7 12

Output

0

3

**Restrictions**

0 <= n, m <= 5 \* 10^5

For each query interval [a, b], it is guaranteed that a <= b.

Points in S are distinct from each other.

Coordinates of each point as well as the query interval boundaries a and b are non-negative integers not greater than 10^7.

Time: 2 sec

Memory: 256 MB

**描述**

数轴上有n个点，对于任一闭区间 [a, b]，试计算落在其内的点数。

**输入**

第一行包括两个整数：点的总数n，查询的次数m。

第二行包含n个数，为各个点的坐标。

以下m行，各包含两个整数：查询区间的左、右边界a和b。

**输出**

对每次查询，输出落在闭区间[a, b]内点的个数。

**样例**

见英文题面

**限制**

0 ≤ n, m ≤ 5×105

对于每次查询的区间[a, b]，都有a ≤ b

各点的坐标互异

各点的坐标、查询区间的边界a、b，均为不超过10^7的非负整数

时间：2 sec

内存：256 MB

# A1.2.祖玛(Zuma)

**Description**

Let's play the game Zuma!

There are a sequence of beads on a track at the right beginning. All the beads are colored but no three adjacent ones are allowed to be with a same color. You can then insert beads one by one into the sequence. Once three (or more) beads with a same color become adjacent due to an insertion, they will vanish immediately.



Note that it is possible for such a case to happen for more than once for a single insertion. You can't insert the next bead until all the eliminations have been done.

Given both the initial sequence and the insertion series, you are now asked by the fans to provide a playback tool for replaying their games. In other words, the sequence of beads after all possible eliminations as a result of each insertion should be calculated.

**Input**

The first line gives the initial bead sequence. Namely, it is a string of capital letters from 'A' to 'Z', where different letters correspond to beads with different colors.

The second line just consists of a single interger n, i.e., the number of insertions.

The following n lines tell all the insertions in turn. Each contains an integer k and a capital letter Σ, giving the rank and the color of the next bead to be inserted respectively. Specifically, k ranges from 0 to m when there are currently m beads on the track.

**Output**

n lines of capital letters, i.e., the evolutionary history of the bead sequence.

Specially, "-" stands for an empty sequence.

**Example**

Input

ACCBA

5

1 B

0 A

2 B

4 C

0 A

Output

ABCCBA

AABCCBA

AABBCCBA

-

A

**Restrictions**

0 <= n <= 10^4

0 <= length of the initial sequence <= 10^4

Time: 2 sec

Memory: 256 MB

**Hints**

List

**描述**

祖玛是一款曾经风靡全球的游戏，其玩法是：在一条轨道上初始排列着若干个彩色珠子，其中任意三个相邻的珠子不会完全同色。此后，你可以发射珠子到轨道上并加入原有序列中。一旦有三个或更多同色的珠子变成相邻，它们就会立即消失。这类消除现象可能会连锁式发生，其间你将暂时不能发射珠子。

开发商最近准备为玩家写一个游戏过程的回放工具。他们已经在游戏内完成了过程记录的功能，而回放功能的实现则委托你来完成。

游戏过程的记录中，首先是轨道上初始的珠子序列，然后是玩家接下来所做的一系列操作。你的任务是，在各次操作之后及时计算出新的珠子序列。

**输入**

第一行是一个由大写字母'A'~'Z'组成的字符串，表示轨道上初始的珠子序列，不同的字母表示不同的颜色。

第二行是一个数字n，表示整个回放过程共有n次操作。

接下来的n行依次对应于各次操作。每次操作由一个数字k和一个大写字母Σ描述，以空格分隔。其中，Σ为新珠子的颜色。若插入前共有m颗珠子，则k ∈ [0, m]表示新珠子嵌入之后（尚未发生消除之前）在轨道上的位序。

**输出**

输出共n行，依次给出各次操作（及可能随即发生的消除现象）之后轨道上的珠子序列。

如果轨道上已没有珠子，则以“-”表示。

**样例**

见英文题面

**限制**

0 ≤ n ≤ 10^4

0 ≤ 初始珠子数量 ≤ 10^4

时间：2 sec

内存：256 MB

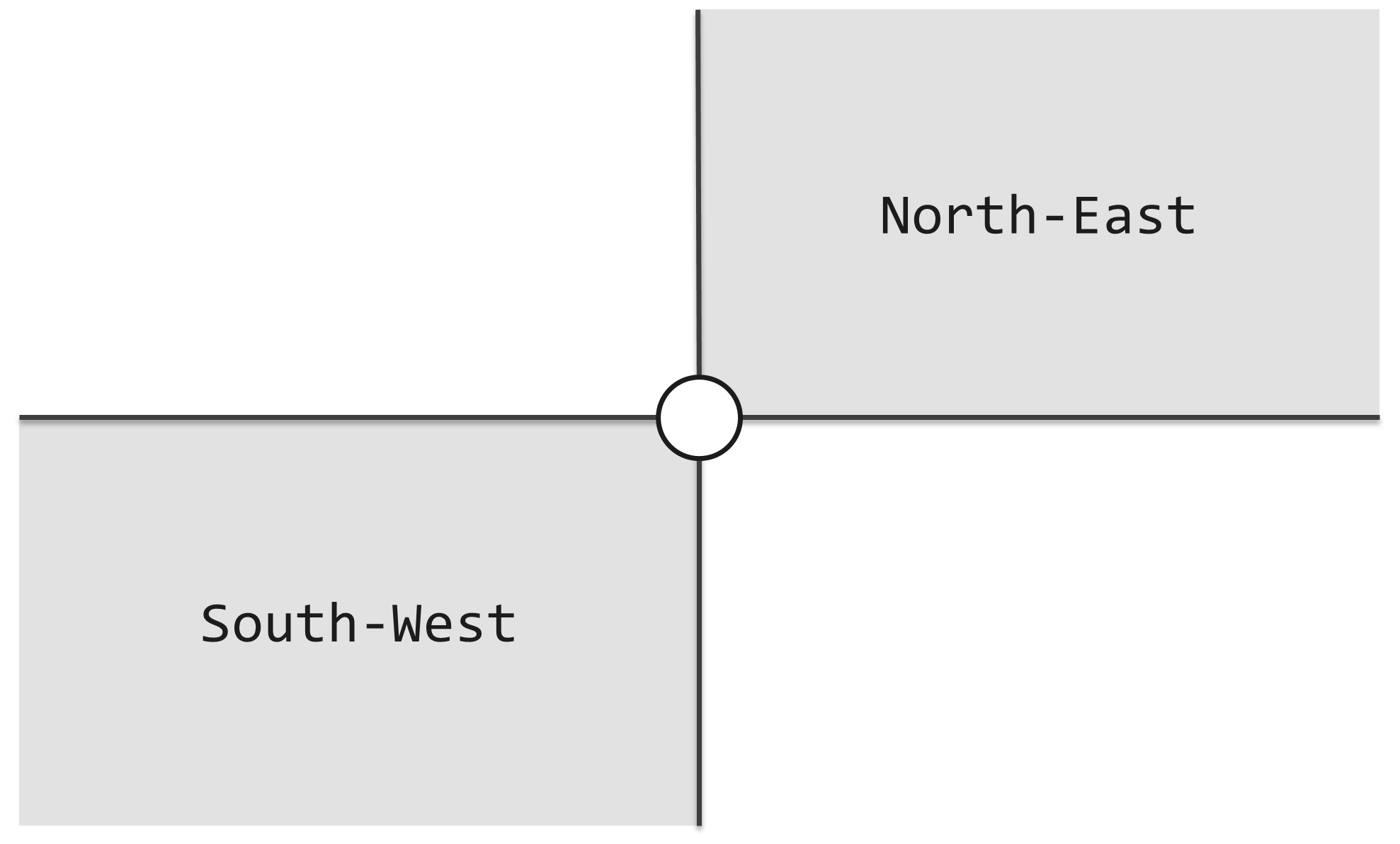
**提示**

列表

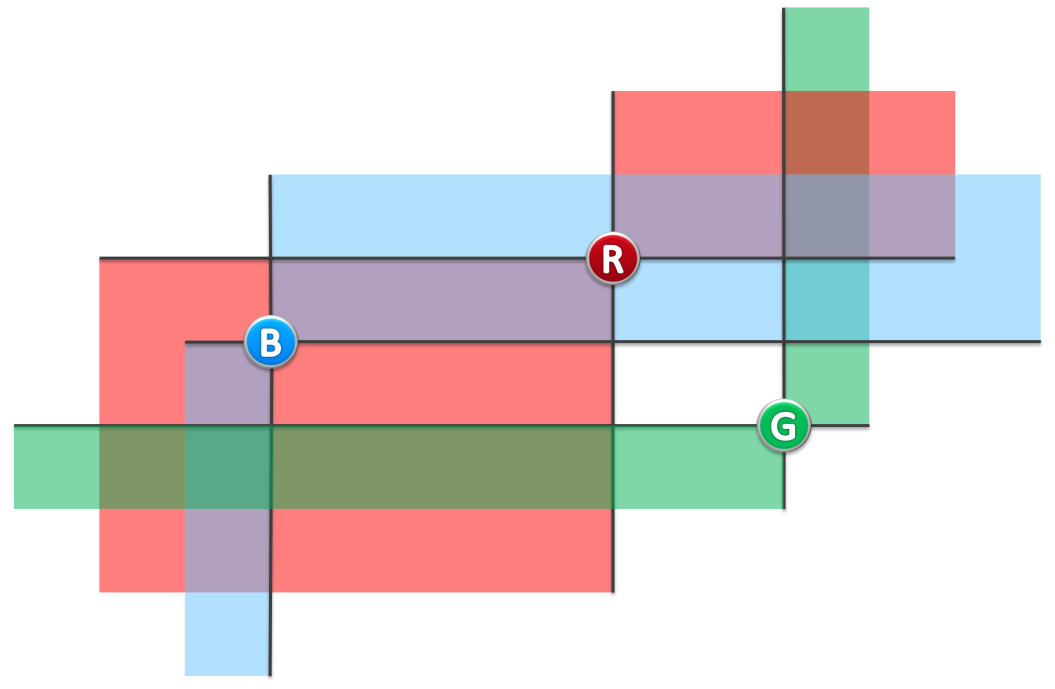
# A1.3.灯塔(LightHouse)

**Description**

As shown in the following figure, If another lighthouse is in gray area, they can beacon each other.



For example, in following figure, (B, R) is a pair of lighthouse which can beacon each other, while (B, G), (R, G) are NOT.



**Input**

1st line: N

2nd ~ (N + 1)th line: each line is X Y, means a lighthouse is on the point (X, Y).

**Output**

How many pairs of lighthourses can beacon each other

( For every lighthouses, X coordinates won't be the same , Y coordinates won't be the same )

**Example**

Input

3

2 2

4 3

5 1

Output

1

**Restrictions**

For 90% test cases: 1 <= n <= 3 \* 105

For 95% test cases: 1 <= n <= 106

For all test cases: 1 <= n <= 4 \* 106

For every lighthouses, X coordinates won't be the same , Y coordinates won't be the same.

1 <= x, y <= 10^8

Time: 2 sec

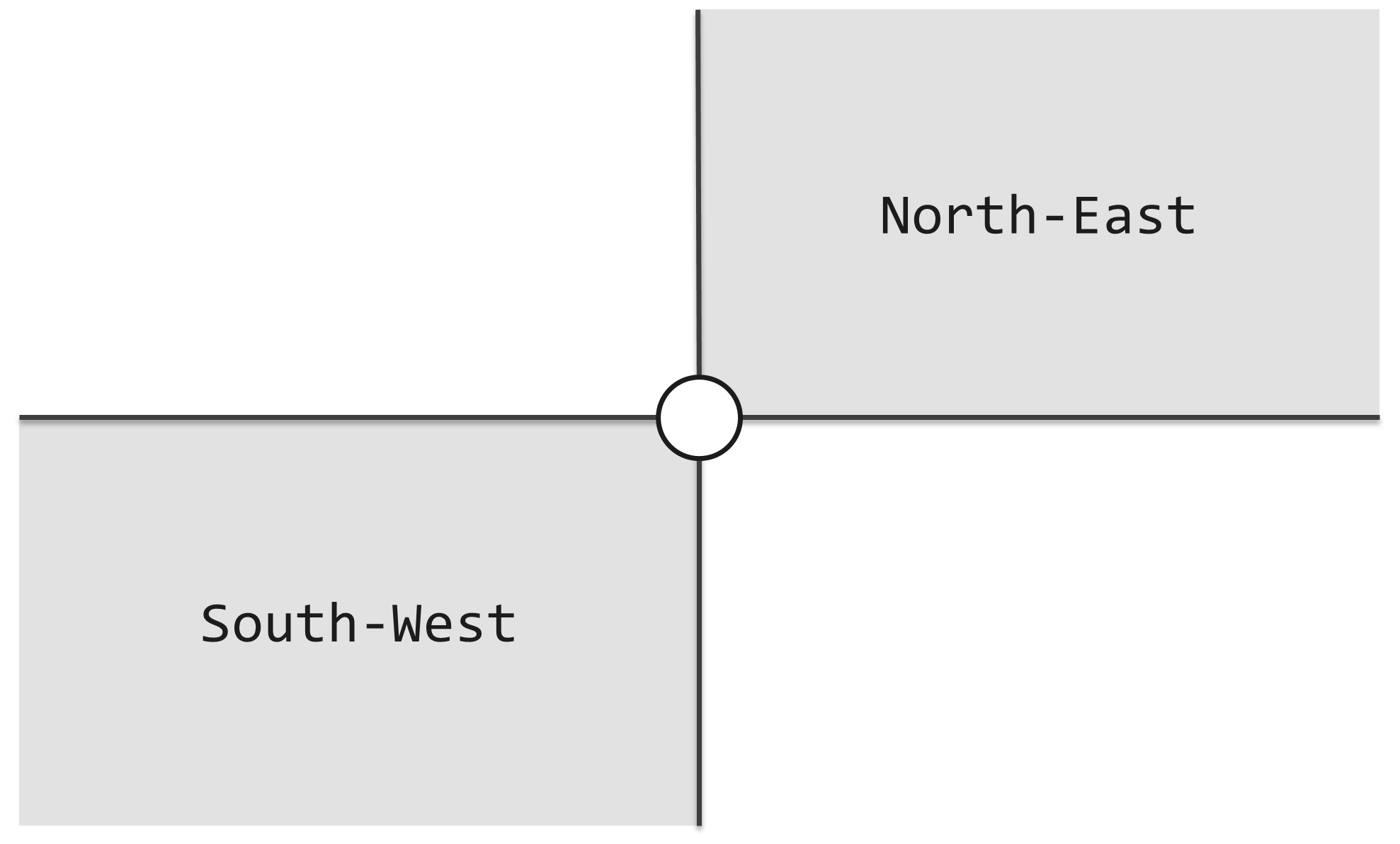
Memory: 256 MB

**Hints**

The range of **int** is usually [-231, 231 - 1], it may be too small.

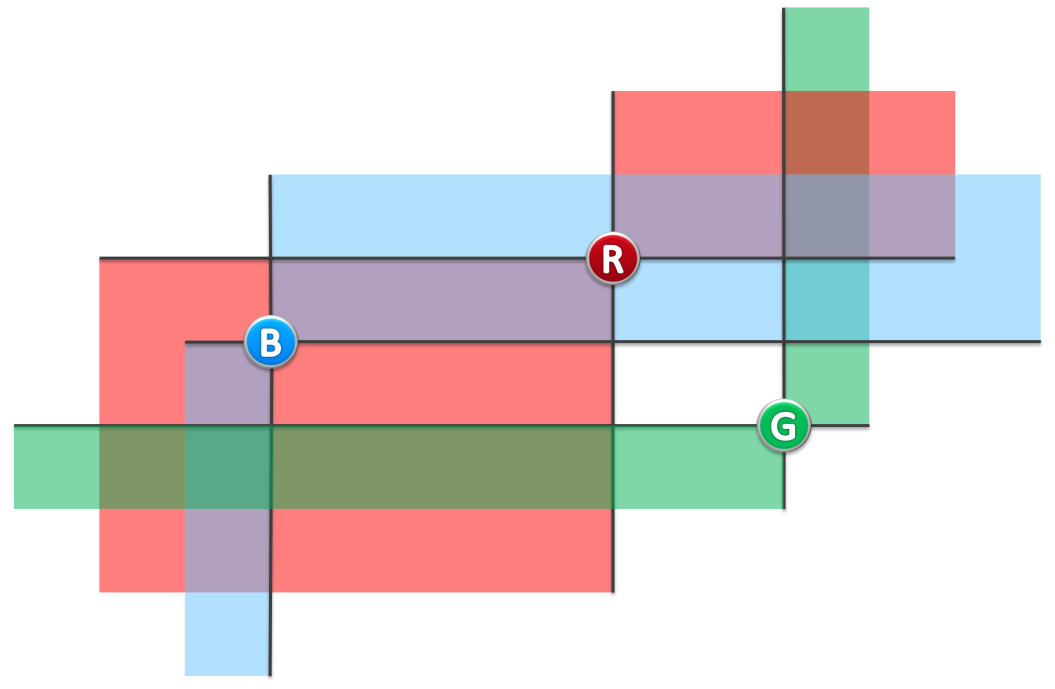
**描述**

海上有许多灯塔，为过路船只照明。



（图一）

如图一所示，每个灯塔都配有一盏探照灯，照亮其东北、西南两个对顶的直角区域。探照灯的功率之大，足以覆盖任何距离。灯塔本身是如此之小，可以假定它们不会彼此遮挡。



（图二）

若灯塔A、B均在对方的照亮范围内，则称它们能够照亮彼此。比如在图二的实例中，蓝、红灯塔可照亮彼此，蓝、绿灯塔则不是，红、绿灯塔也不是。

现在，对于任何一组给定的灯塔，请计算出其中有多少对灯塔能够照亮彼此。

**输入**

共n+1行。

第1行为1个整数n，表示灯塔的总数。

第2到n+1行每行包含2个整数x, y，分别表示各灯塔的横、纵坐标。

**输出**

1个整数，表示可照亮彼此的灯塔对的数量。

**样例**

见英文题面

**限制**

对于90%的测例：1 ≤ n ≤ 3×105

对于95%的测例：1 ≤ n ≤ 106

全部测例：1 ≤ n ≤ 4×106

灯塔的坐标x, y是整数，且不同灯塔的x, y坐标均互异

1 ≤ x, y ≤ 10^8

时间：2 sec

内存：256 MB

**提示**

注意机器中整型变量的范围，C/C++中的int类型通常被编译成32位整数，其范围为[-231, 231 - 1]，不一定足够容纳本题的输出。

# A2.1.列车调度(Train)

### ****Description****

Figure 1 shows the structure of a station for train dispatching.

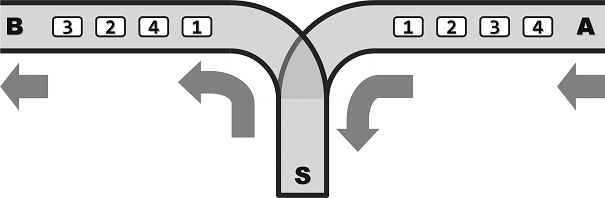


Figure 1

In this station, A is the entrance for each train and B is the exit. S is the transfer end. All single tracks are one-way, which means that the train can enter the station from A to S, and pull out from S to B. Note that the overtaking is not allowed. Because the compartments can reside in S, the order that they pull out at B may differ from that they enter at A. However, because of the limited capacity of S, no more that m compartments can reside at S simultaneously.

Assume that a train consist of n compartments labeled {1, 2, …, n}. A dispatcher wants to know whether these compartments can pull out at B in the order of {a1, a2, …, an} (a sequence). If can, in what order he should operate it?

### ****Input****

Two lines:

1st line: two integers n and m;

2nd line: n integers separated by spaces, which is a permutation of {1, 2, …, n}. This is a compartment sequence that is to be judged regarding the feasibility.

### ****Output****

If the sequence is feasible, output the sequence. “Push” means one compartment goes from A to S, while “pop” means one compartment goes from S to B. Each operation takes up one line.

If the sequence is infeasible, output a “no”.

### ****Example 1****

Input

5 2

1 2 3 5 4

Output

push

pop

push

pop

push

pop

push

push

pop

pop

### ****Example 2****

Input

5 5

3 1 2 4 5

Output

No

### ****Restrictions****

1 <= n <= 1,600,000

0 <= m <= 1,600,000

Time: 2 sec

Memory: 256 MB

### ****描述****

某列车调度站的铁道联接结构如Figure 1所示。

其中，A为入口，B为出口，S为中转盲端。所有铁道均为单轨单向式：列车行驶的方向只能是从A到S，再从S到B；另外，**不允许超车**。因为车厢可在S中驻留，所以它们从B端驶出的次序，可能与从A端驶入的次序不同。不过S的容量有限，同时驻留的车厢不得超过m节。

设某列车由编号依次为{1, 2, ..., n}的n节车厢组成。调度员希望知道，按照以上交通规则，这些车厢能否以{a1, a2, ..., an}的次序，重新排列后从B端驶出。如果可行，应该以怎样

的次序操作?

### ****输入****

共两行。

第一行为两个整数n，m。

第二行为以空格分隔的n个整数，保证为{1, 2, ..., n}的一个排列，表示待判断可行性的驶出序列{a1，a2，...，an}。

### ****输出****

若驶出序列可行，则输出操作序列，其中push表示车厢从A进入S，pop表示车厢从S进入B，每个操作占一行。

若不可行，则输出No。

### ****样例****

见英文题面

### ****限制****

1 ≤ n ≤ 1,600,000

0 ≤ m ≤ 1,600,000

时间：2 sec

空间：256 MB

# A2.2.真二叉树重构(Proper Rebuild)

### ****Description****

In general, given the preorder traversal sequence and postorder traversal sequence of a binary tree, we cannot determine the binary tree.

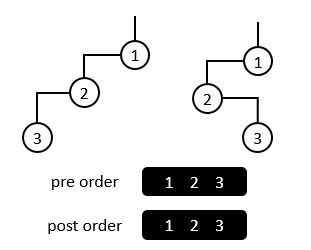


Figure 1

In Figure 1 for example, although they are two different binary tree, their preorder traversal sequence and postorder traversal sequence are both of the same.

But for one proper binary tree, in which each internal node has two sons, we can uniquely determine it through its given preorder traversal sequence and postorder traversal sequence.

Label n nodes in one binary tree using the integers in [1, n], we would like to output the inorder traversal sequence of a binary tree through its preorder and postorder traversal sequence.

### ****Input****

The 1st line is an integer n, i.e., the number of nodes in one given binary tree,

The 2nd and 3rd lines are the given preorder and postorder traversal sequence respectively.

### ****Output****

The inorder traversal sequence of the given binary tree in one line.

### ****Example****

Input

5

1 2 4 5 3

4 5 2 3 1

Output

4 2 5 1 3

### ****Restrictions****

For 95% of the estimation, 1 <= n <= 1,000,00

For 100% of the estimation, 1 <= n <= 4,000,000

The input sequence is a permutation of {1,2...n}, corresponding to a legal binary tree.

Time: 2 sec

Memory: 256 MB

### ****Hints****

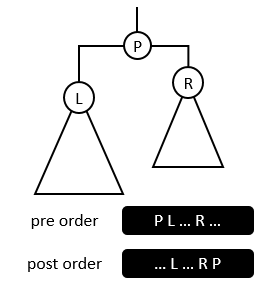
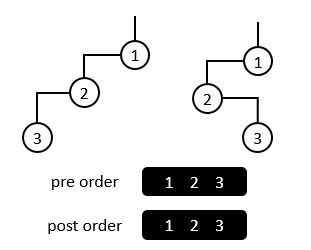


Figure 2

In Figure 2, observe the positions of the left and right children in preorder and postorder traversal sequence.

### ****描述****

一般来说，给定二叉树的先序遍历序列和后序遍历序列，并不能确定唯一确定该二叉树。



（图一）

比如图一中的两棵二叉树，虽然它们是不同二叉树，但是它们的先序、后序遍历序列都是相同的。

但是对于“真二叉树”（每个内部节点都有两个孩子的二叉树），给定它的先序、后序遍历序列足以完全确定它的结构。

将二叉树的n个节点用[1, n]内的整数进行编号，输入一棵真二叉树的先序、后序遍历序列，请输出它的中序遍历序列。

### ****输入****

第一行为一个整数n，即二叉树中节点的个数。

第二、三行为已知的先序、后序遍历序列。

### ****输出****

仅一行，给定真二叉树的中序遍历序列。

### ****样例****

见英文题面

### ****限制****

对于95%的测例：1 ≤ n ≤ 1,000,000

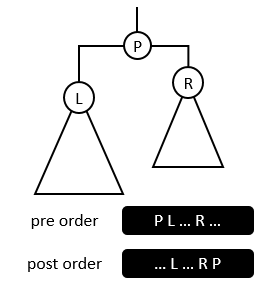
对于100%的测例：1 ≤ n ≤ 4,000,000

输入的序列是{1,2...n}的排列，且对应于一棵合法的真二叉树

时间：2 sec

空间：256 MB

### ****提示****



观察左、右孩子在先序、后序遍历序列中的位置

重温视频05e5-3

# A2.2.旅行商(TSP)

### ****Description****

Shrek is a postman working in the mountain, whose routine work is sending mail to n villages. Unfortunately, road between villages is out of repair for long time, such that some road is one-way road. There are even some villages that can’t be reached from any other village. In such a case, we only hope as many villages can receive mails as possible.

Shrek hopes to choose a village A as starting point (He will be air-dropped to this location), then pass by as many villages as possible. Finally, Shrek will arrived at village B. In the travelling process, each villages is only passed by once. You should help Shrek to design the travel route.

### ****Input****

There are 2 integers, n and m, in first line. Stand for number of village and number of road respectively.

In the following m line, m road is given by identity of villages on two terminals. From v1 to v2. The identity of village is in range [1, n].

### ****Output****

Output maximum number of villages Shrek can pass by.

### ****Example****

Input

4 3

1 4

2 4

4 3

Output

3

### ****Restrictions****

1 <= n <= 1,000,000

0 <= m <= 1,000,000

These is no loop road in the input.

Time: 2 sec

Memory: 256 MB

### ****Hints****

Topological sorting

### ****描述****

Shrek是一个大山里的邮递员，每天负责给所在地区的n个村庄派发信件。但杯具的是，由于道路狭窄，年久失修，村庄间的道路都只能单向通过，甚至有些村庄无法从任意一个村庄到达。这样我们只能希望尽可能多的村庄可以收到投递的信件。

Shrek希望知道如何选定一个村庄A作为起点（我们将他空投到该村庄），依次经过尽可能多的村庄，路途中的每个村庄都经过仅一次，最终到达终点村庄B，完成整个送信过程。这个任务交给你来完成。

### ****输入****

第一行包括两个整数n，m，分别表示村庄的个数以及可以通行的道路的数目。

以下共m行，每行用两个整数v1和v2表示一条道路，两个整数分别为道路连接的村庄号，道路的方向为从v1至v2，n个村庄编号为[1, n]。

### ****输出****

输出一个数字，表示符合条件的最长道路经过的村庄数。

### ****样例****

见英文题面

### ****限制****

1 ≤ n ≤ 1,000,000

0 ≤ m ≤ 1,000,000

**输入保证道路之间没有形成环**

时间：2 sec

空间：256 MB

### ****提示****

拓扑排序

# A3.1.无线广播(Broadcast)

### ****描述****

某广播公司要在一个地区架设无线广播发射装置。该地区共有n个小镇，每个小镇都要安装一台发射机并播放各自的节目。

不过，该公司只获得了FM104.2和FM98.6两个波段的授权，而使用同一波段的发射机会互相干扰。已知每台发射机的信号覆盖范围是以它为圆心，20km为半径的圆形区域，因此，如果距离小于20km的两个小镇使用同样的波段，那么它们就会由于波段干扰而无法正常收听节目。现在给出这些距离小于20km的小镇列表，试判断该公司能否使得整个地区的居民正常听到广播节目。

### ****输入****

第一行为两个整数n，m，分别为小镇的个数以及接下来小于20km的小镇对的数目。 接下来的m行，每行2个整数，表示两个小镇的距离小于20km（编号从1开始）。

### ****输出****

如果能够满足要求，输出1，否则输出-1。

### ****输入样例****

4 3

1 2

1 3

2 4

### ****输出样例****

1

### ****限制****

1 ≤ n ≤ 10000

1 ≤ m ≤ 30000

不需要考虑给定的20km小镇列表的空间特性，比如是否满足三角不等式，是否利用传递性可以推出更多的信息等等。

时间：2 sec

空间：256MB

### ****提示****

BFS

# A3.2.平均气温(Temperature)

### ****题目描述****

某气象台每天都要从遍布于各地的观察站采集气温数据，并通过互联网为远程用户提供统计查询服务。其中最常见的一类查询是，根据用户指定矩形区域内所有观察站的观测值计算出平均气温。随着更多观察站的不断建立，原始数据本身的规模急剧膨胀。另外，尽管可以假设每天采集的数据相对固定，但随着用户群体的扩大，查询的频率也日益激增。鉴于传统蛮力算法的效率已无法满足实用要求，气象台只好请你帮忙，通过改进数据结构和算法，提高查询的效率。

借助气象台提供的一组函数接口，服务器端可访问已采集到的所有数据，并报告查询结果。

### ****接口说明****

int GetNumOfStation(void);

该函数必须首先调用，返回现有观察站的总数n。

void GetStationInfo(int no, int \*x, int \*y, int \*temp);

获得第no个（0 ≤ no < n）观察站的信息：其地理坐标(\*x,\*y)及其所测温度值\*temp。各观测站的测量精度统一以0.01℃为基准单位，比如12.34℃表示为整数1234。

int GetQuery(int \*x1, int \*y1, int \*x2, int \*y2);

接收下一查询请求。返回值1对应于一次有效的查询。矩阵区域的四边分别与x或y轴平行，(\*x1,\*y1)和(\*x2,\*y2)分别为其西南角和东北角的坐标。恰好被矩形边界穿过的观察站，也视作落在其中。若返回0，则表示没有更多的查询，你的程序可以退出。

void Response(int temp);

针对当前的查询，在计算出对应的平均气温后，你可通过这一接口报告所得数值(截断取整，比如12.345℃输出为1234，-12.345℃输出为-1234)。

**特别注意**：每调用GetQuery()接收一次查询后，若未能通过Response()函数报告该次查询的结果就再次调用GetQuery()接收下一查询，则将因为前次查询的结果无法报告而注定输出错误。也就是说，GetQuery()和Response()必须交替调用，各n次。

### ****测试说明****

为便于你调试和测试，随题还附带有temperature.h和temperature\_lib.c文件。前者约定了上述接口，后者是这组接口的一种实现——OJ上的实现与之不同，但接口完全一致。调试时可将它们与你的代码一同编译，但在线测试时不必提交；即便提交，OJ也会自动忽略它们。

[下载接口文件](https://dsa.cs.tsinghua.edu.cn/oj/attachment/60fe/60fec26fef5ccada8e4e8845b808985ad9080785.zip)

### ****输入****

脱机调试时，temperature\_lib.c所实现的三个输入接口，实际上是从当前目录下的temperature.in文件读入数据，因此通过按如下格式更改该文件，即可设定不同的输入数据：

第一行为两个整数：观察站总数n，所需查询的总次数m

以下n行分别描述各观察站：位置坐标为整数(x, y)，该站所测得温度值为整数t

再以下m行分别对应于各次查询操作，整数(x1, y1)和(x2, y2)分别表示其西南角和东北角

### ****输出****

脱机调试时，temperature\_lib.c所实现的Response()接口会在程序运行后，将所有的输出结果写入temperature.out文件。

文件共m行，各含1个整数，表示每次查询所得平均温度。

若查询区域不含任何观测站，则输出0。

### ****输入样例****

4 2

0 0 1000

1 1 1300

2 2 1600

3 3 1100

0 0 1 1

0 0 10 10

### ****输出样例****

1150

1250

### ****限制****

0 ≤n ≤ 50,000

0 ≤ m ≤ 500,000

观测站坐标取值范围是[-2^31, 2^31)

查询区域的坐标 x1 ≤ x2 且 y1 ≤ y2

时间限制：10秒

内存限制：256 MB

### ****提示****

温度计算请使用64位整数，以保证累加不致溢出

kd-tree

range tree

本题的规则较一般题目更为复杂，若对题面有疑问，请在讨论区提问

# A3.3.重名剔除(Deduplicate)

### ****Description****

Mr. Epicure is compiling an encyclopedia of food. He had collected a long list of candidates nominated by several belly-gods. As candidates in list are nominated by several people, duplication of name is inevitable. Mr. Epicure pay you a visit for help. He request you to remove all duplication, which is thought an easy task for you. So please hold this opportunity to be famous to all belly-gods.

### ****Input****

1 integer in fist line to denote the length of nomination list. In following n lines, each nomination is given in each line.

### ****Output****

All the duplicated nomination (only output once if duplication appears more multiple times), which is sorted in the order that duplication appears firstly.

### ****Example****

Input

10

brioche

camembert

cappelletti

savarin

cheddar

cappelletti

tortellni

croissant

brioche

mapotoufu

Output

cappelletti

brioche

### ****Restrictions****

1 < n < 6 \* 10^5

All nominations are only in lowercase. No other character is included. Length of each item is not greater than 40.

Time: 2 sec

Memory: 256 MB

### ****Hints****

Hash

### ****描述****

Epicure先生正在编撰一本美食百科全书。为此，他已从众多的同好者那里搜集到了一份冗长的美食提名清单。既然源自多人之手，其中自然不乏重复的提名，故必须予以筛除。Epicure先生因此登门求助，并认定此事对你而言不过是“一碟小菜”，相信你不会错过在美食界扬名立万的这一良机

### ****输入****

第1行为1个整数n，表示提名清单的长度。以下n行各为一项提名

### ****输出****

所有出现重复的提名（多次重复的仅输出一次），且以其在原清单中首次出现重复（即第二次出现）的位置为序

### ****样例****

见英文题面

### ****限制****

1 < n < 6 \* 10^5

提名均由小写字母组成，不含其它字符，且每项长度不超过40

时间：2 sec

空间：256 MB

### ****提示****

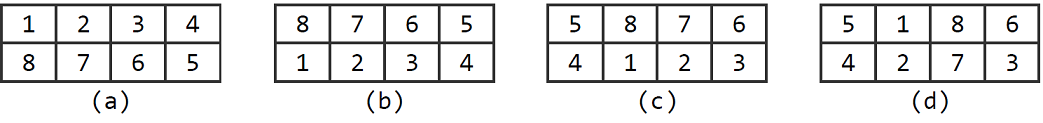
散列

# A4.1.玩具(Toy)

### ****描述****

ZC神最擅长逻辑推理，一日，他给大家讲述起自己儿时的数字玩具。

该玩具酷似魔方，又不是魔方。具体来说，它不是一个3 \* 3 \* 3的结构，而是4 \* 2的结构。



按照该玩具约定的玩法，我们可反复地以如下三种方式对其做变换：

**A． 交换上下两行。比如，图(a)经此变换后结果如图(b)所示。**

**B． 循环右移（ZC神从小就懂得这是什么意思的）。比如，图(b)经此变换后结果如图(c)所示。**

**C． 中心顺时针旋转。比如，图(c)经此变换后结果如图(d)所示。**

ZC神自小就是这方面的天才，他往往是一只手还没揩干鼻涕，另一只手已经迅速地将处于任意状态的玩具复原至如图(a)所示的初始状态。物质极其匮乏的当年，ZC神只有一个这样的玩具；物质极大丰富的今天，你已拥有多个处于不同状态的玩具。现在，就请将它们全部复原吧。

### ****输入****

第一行是一个正整数，即你拥有的魔方玩具总数N。

接下来共N行，每行8个正整数，表示该玩具的当前状态。

这里，魔方状态的表示规则为：前四个数自左向右给出魔方的第一行，后四个数自右向左给出第二行。比如，初始状态表示为“1 2 3 4 5 6 7 8”。

### ****输出****

共N行，各含一个整数，依次对应于复原各玩具所需执行变换的最少次数。

特别地，若某个玩具不可复原，则相应行输出-1。

### ****输入样例****

2

1 2 3 4 5 6 7 8

8 6 3 5 4 2 7 1

### ****输出样例****

0

2

### ****限制****

对于60%的数据，N = 1

对于100%的数据，1 <= N <= 1,000

时间：1 sec

空间：20MB

### ****提示****

状态转换图及其搜索

# A4.2.任务调度(Schedule)

### ****Description****

A HPS cluster is equipped with a unique task scheduler. To be simple, it is assumed that this cluster doesn’t support multiple tasks running at the same time, such that only one task is allowed to be in running state at any moment. Initially, the priority of ever task is denoted by an integer which is called priority number. **The smaller priority number stands for high priority. If two tasks have same task number, the priority is decided in the ASCII order of task name.**Following this policy, resources, such as CPU, are always occupied by the task with minimum priority number. When one task is finished, the one with minimum priority number in the rest tasks is picked to execute. The finished task won’t quit immediately. The priority number is doubled and put back to the task set. Once the priority number is greater or equal to **2^32**, this task is deleted from the task set.

Given initial priority setting of every task, your job is to predict the running order of a batch of tasks.

### ****Input****

First line contains two integers, says n and m. n stands for the number of tasks in initial state. m stands for the length of predicted sequence. Every line is ended by a line break symbol. In each one of the following n lines, an integer and a string are included. This string is shorter than 8, which only contains lowercase letters and numbers. The integer is priority number and the string is the task name. The integer and string is separated by space.

### ****Output****

At most m lines, each one contains a string. Output the name of tasks according to the order that tasks are executed. If the number of executed tasks is less than m, then output all the executed tasks.

### ****Example****

Input

3 3

1 hello

2 world

10 test

Output

hello

hello

world

### ****Restrictions****

0 <= n <= 4,000,000

0 <= m <= 2,000,000

0 < Priority number < 2^32

No tasks have same name

Time: 2 sec

Memory: 512 MB

### ****Hints****

Priority queue

### ****描述****

某高性能计算集群（HPC cluster）采用的任务调度器与众不同。为简化起见，假定该集群不支持多任务同时执行，故同一时刻只有单个任务处于执行状态。初始状态下，每个任务都由称作优先级数的一个整数指定优先级，**该数值越小优先级越高**；**若优先级数相等，则任务名ASCII字典顺序低者优先**。此后，CPU等资源总是被优先级数最小的任务占用；每一任务计算完毕，再选取优先级数最小下一任务。不过，这里的任务在计算结束后通常并不立即退出，而是将优先级数加倍（加倍计算所需的时间可以忽略）并继续参与调度；只有在优先级数不小于**2^32**时，才真正退出

你的任务是，根据初始优先级设置，按照上述调度原则，预测一批计算任务的执行序列。

### ****输入****

第一行为以空格分隔的两个整数n和m，n为初始时的任务总数，m为所预测的任务执行序列长度，每行末尾有一个换行符

以下n行分别包含一个整数和一个由不超过8个小写字母和数字组成的字符串。前者为任务的初始优先级数，后者为任务名。数字和字符串之间以空格分隔

### ****输出****

最多m行，各含一个字符串。按执行次序分别给出执行序列中前m个任务的名称，若执行序列少于m，那么输出调度器的任务处理完毕前的所有任务即可。

### ****样例****

见英文题面

### ****限制****

0 ≤ n ≤ 4,000,000

0 ≤ m ≤ 2,000,000

0 < 每个任务的初始优先级 < 2^32

不会有重名的任务

时间：2 sec

内存：512 MB

### ****提示****

优先级队列

# A4.3. 循环移位(Cycle)

### ****Description****

Cycle shifting refers to following operation on the sting. Moving first letter to the end and keeping rest part of the string. For example, apply cycle shifting on ABCD will generate BCDA. Given any two strings, to judge if arbitrary times of cycle shifting on one string can generate the other one.

### ****Input****

There m lines in the input, while each one consists of two strings separated by space. Each string only contains uppercase letter 'A'~'Z'.

### ****Output****

For each line in input, output YES in case one string can be transformed into the other by cycle shifting, otherwise output NO.

### ****Example****

Input

AACD CDAA

ABCDEFG EFGABCD

ABCD ACBD

ABCDEFEG ABCDEE

Output

YES

YES

NO

NO

### ****Restrictions****

0 <= m <= 5000

1 <= |S1|, |S2| <= 10^5

Time: 2 sec

Memory: 256 MB

### ****描述****

所谓循环移位是指。一个字符串的首字母移到末尾, 其他字符的次序保持不变。比如ABCD经过一次循环移位后变成BCDA

给定两个字符串，判断它们是不是可以通过若干次循环移位得到彼此

### ****输入****

由m行组成，每行包含两个由大写字母'A'~'Z'组成的字符串，中间由空格隔开

### ****输出****

对于每行输入，输出这两个字符串是否可以通过循环移位得到彼此：YES表示是，NO表示否

### ****样例****

见英文题面

### ****限制****

0 ≤ m ≤ 5000

1 ≤ |S1|, |S2| ≤ 10^5

时间：2 sec

内存：256 MB