A. Alexey and Train

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Alexey is travelling on a train. Unfortunately, due to the bad weather, the train moves slower that it should!

Alexey took the train at the railroad terminal. Let's say that the train starts from the terminal at the moment 0. Also, let's say that the train will visit n stations numbered from 1 to n along its way, and that Alexey destination is the station n.

Alexey learned from the train schedule n integer pairs (a_i, b_i) where a_i is the expected time of train's arrival at the i-th station and b_i is the expected time of departure.

Also, using all information he has, Alexey was able to calculate n integers $tm_1, tm_2, \dots, tm_n$ where tm_i is the extra time the train need to travel from the station $i-1$ to the station i. Formally, the train needs exactly $a_i - b_{i-1} + tm_i$ time to travel from station $i-1$ to station i (if $i = 1$ then b_0 is the moment the train leave the terminal, and it's equal to 0).

The train leaves the station i, if both conditions are met:

1. it's on the station for at least $\left\lceil \frac{b_i - a_i}{2} \right\rceil$ units of time (division with ceiling);
2. current time $\ge b_i$.

Since Alexey spent all his energy on prediction of time delays, help him to calculate the time of **arrival** at the station n.

Input

The first line contains one integer t $(1 \le t \le 100)$ — the number of test cases.

The first line of each test case contains the single integer n $(1 \le n \le 100)$ — the number of stations.

Next n lines contain two integers each: a_i and b_i $(1 \le a_i < b_i \le 10^6)$. It's guaranteed that $b_i < a_{i+1}$.

Next line contains n integers $tm_1, tm_2, \dots, tm_n$ $(0 \le tm_i \le 10^6)$.

Output

For each test case, print one integer — the time of Alexey's arrival at the last station.

Example

input

```
2
2
2 4
10 12
0 2
5
1 4
7 8
9 10
13 15
19 20
1 2 3 4 5
```

output

```
12
32
```

Note

In the first test case, Alexey arrives at station 1 without any delay at the moment $a_1 = 2$ (since $tm_1 = 0$). After that, he departs at moment $b_1 = 4$. Finally, he arrives at station 2 with $tm_2 = 2$ extra time, or at the moment 12.

In the second test case, Alexey arrives at the first station with $tm_1 = 1$ extra time, or at moment 2. The train, from one side, should stay at the station at least $\left\lceil \frac{b_1 - a_1}{2} \right\rceil = 2$ units of time and from the other side should depart not earlier than at moment $b_1 = 4$. As a result, the trains departs right at the moment 4.

Using the same logic, we can figure out that the train arrives at the second station at the moment 9 and departs at the moment 10; at the third station: arrives at 14 and departs at 15; at the fourth: arrives at 22 and departs at 23. And, finally, arrives at the fifth station at 32.

B. Napoleon Cake

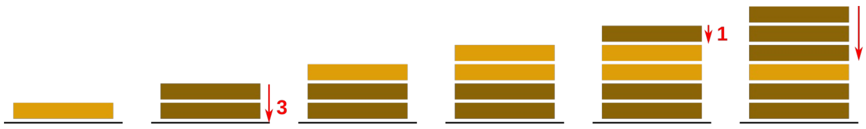
time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

This week Arkady wanted to cook some pancakes (to follow ancient traditions) and make a problem about that. But then he remembered that one can't make a problem about stacking pancakes without working at a specific IT company, so he decided to bake the Napoleon cake instead.

To bake a Napoleon cake, one has to bake n dry layers first, and then put them on each other in one stack, adding some cream. Arkady started with an empty plate, and performed the following steps n times:

- place a new cake layer on the top of the stack;
- after the i-th layer is placed, pour a_i units of cream on top of the stack.

When x units of cream are poured on the top of the stack, top x layers of the cake get drenched in the cream. If there are less than x layers, all layers get drenched and the rest of the cream is wasted. If $x = 0$, no layer gets drenched.



The picture represents the first test case of the example.

Help Arkady determine which layers of the cake eventually get drenched when the process is over, and which don't.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 20\,000$). Description of the test cases follows.

The first line of each test case contains a single integer n ($1 \le n \le 2 \cdot 10^5$) — the number of layers in the cake.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le n$) — the amount of cream poured on the cake after adding each layer.

It is guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, print a single line with n integers. The i-th of the integers should be equal to 1 if the i-th layer from the bottom gets drenched, and 0 otherwise.

Example

input

```
3
6
0 3 0 0 1 3
10
0 0 0 1 0 5 0 0 0 2
3
0 0 0
```

output

```
1 1 0 1 1 1
0 1 1 1 1 1 0 0 1 1
0 0 0
```

C. Going Home

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

It was the third month of remote learning, Nastya got sick of staying at dormitory, so she decided to return to her hometown. In order to make her trip more entertaining, one of Nastya's friend presented her an integer array a.

Several hours after starting her journey home Nastya remembered about the present. To entertain herself she decided to check, are there four **different** indices x, y, z, w such that $a_x + a_y = a_z + a_w$.

Her train has already arrived the destination, but she still hasn't found the answer. Can you help her unravel the mystery?

Input

The first line contains the single integer n ($4 \le n \le 200\,000$) — the size of the array.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 2.5 \cdot 10^6$).

Output

Print "YES" if there are such four indices, and "NO" otherwise.

If such indices exist, print these indices x, y, z and w ($1 \le x, y, z, w \le n$).

If there are multiple answers, print any of them.

Examples

input

```
6
2 1 5 2 7 4
```

output

```
YES
2 3 1 6
```

input

```
5
1 3 1 9 20
```

output

```
NO
```

Note

In the first example $a_2 + a_3 = 1 + 5 = 2 + 4 = a_1 + a_6$. Note that there are other answer, for example, 2 3 4 6.

In the second example, we can't choose four indices. The answer 1 2 2 3 is wrong, because indices should be different, despite that $a_1 + a_2 = 1 + 3 = 3 + 1 = a_2 + a_3$

D. Two chandeliers

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Vasya is a CEO of a big construction company. And as any other big boss he has a spacious, richly furnished office with two crystal chandeliers. To stay motivated Vasya needs the color of light at his office to change every day. That's why he ordered both chandeliers that can change its color cyclically. For example: red – brown – yellow – red – brown – yellow and so on.

There are many chandeliers that differs in color set or order of colors. And the person responsible for the light made a critical mistake — they bought two different chandeliers.

Since chandeliers are different, some days they will have the same color, but some days — different. Of course, it looks poor and only annoys Vasya. As a result, at the k-th time when chandeliers will light with different colors, Vasya will become very angry and, most probably, will fire the person who bought chandeliers.

Your task is to calculate the day, when it happens (counting from the day chandeliers were installed). You can think that Vasya works every day without weekends and days off.

Input

The first line contains three integers n, m and k ($1 \le n, m \le 500\,000$; $1 \le k \le 10^{12}$) — the number of colors in the first and the second chandeliers and how many times colors should differ to anger Vasya.

The second line contains n **different** integers a_i ($1 \le a_i \le 2 \cdot \max(n, m)$) that describe the first chandelier's sequence of colors.

The third line contains m **different** integers b_j ($1 \le b_i \le 2 \cdot \max(n, m)$) that describe the second chandelier's sequence of colors.

At the i-th day, the first chandelier has a color a_x, where $x = ((i-1) \mod n) + 1)$ and the second one has a color b_y, where $y = ((i-1) \mod m) + 1)$.

It's guaranteed that sequence a differs from sequence b, so there are will be days when colors of chandeliers differs.

Output

Print the single integer — the index of day when Vasya will become angry.

Examples

input

```
4 2 4
4 2 3 1
2 1
```

output

```
5
```

input

```
3 8 41
1 3 2
```

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

output

```
47
```

input

```
1 2 31
1
1 2
```

output

```
62
```

Note

In the first example, the chandeliers will have different colors at days 1, 2, 3 and 5. That's why the answer is 5.

E. Matrix Sorting

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given two tables A and B of size $n \times m$.

We define a *sorting by column* as the following: we choose a column and reorder the rows of the table by the value in this column, from the rows with the smallest value to the rows with the largest. In case there are two or more rows with equal value in this column, their relative order does not change (such sorting algorithms are called *stable*).

You can find this behavior of sorting by column in many office software for managing spreadsheets. Petya works in one, and he has a table A opened right now. He wants to perform zero of more sortings by column to transform this table to table B.

Determine if it is possible to do so, and if yes, find a sequence of columns to sort by. Note that you **do not need** to minimize the number of sortings.

Input

The first line contains two integers n and m ($1 \le n, m \le 1500$) — the sizes of the tables.

Each of the next n lines contains m integers $a_{i,j}$ ($1 \le a_{i,j} \le n$), denoting the elements of the table A.

Each of the next n lines contains m integers $b_{i,j}$ ($1 \le b_{i,j} \le n$), denoting the elements of the table B.

Output

If it is not possible to transform A into B, print -1.

Otherwise, first print an integer k ($0 \le k \le 5000$) — the number of sortings in your solution.

Then print k integers $c_1, \ldots, c_k$ ($1 \le c_i \le m$) — the columns, by which Petya needs to perform a sorting.

We can show that if a solution exists, there is one in no more than 5000 sortings.

Examples

input

```
2 2
2 2
1 2
1 2
2 2
```

output

```
1
1
```

input

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

output

```
2
1 2
```

input

```
2 2
1 1
```

```
2 1
2 1
1 1
```

output

```
-1
```

input

```
4 1
2
2
2
1
1
2
2
2
```

output

```
1
1
```

Note

Consider the second example. After the sorting by the first column the table becomes

$$\begin{matrix} 1 & 3 & 3 \\ 1 & 1 & 2 \\ 2 & 3 & 2. \end{matrix}$$

After the sorting by the second column the table becomes

$$\begin{matrix} 1 & 1 & 2 \\ 1 & 3 & 3 \\ 2 & 3 & 2, \end{matrix}$$

and this is what we need.

In the third test any sorting does not change anything, because the columns are already sorted.

F. Tiles for Bathroom

time limit per test: 5 seconds

memory limit per test: 512 megabytes

input: standard input

output: standard output

Kostya is extremely busy: he is renovating his house! He needs to hand wallpaper, assemble furniture throw away trash.

Kostya is buying tiles for bathroom today. He is standing in front of a large square stand with tiles in a shop. The stand is a square of $n \times n$ cells, each cell of which contains a small tile with color $c_{i,\,j}$. The shop sells tiles in packs: more specifically, you can only buy a subsquare of the initial square.

A subsquare is any square part of the stand, i. e. any set $S(i_0, j_0, k) = \{c_{i,\,j} \mid i_0 \le i < i_0 + k, j_0 \le j < j_0 + k\}$ with $1 \le i_0, j_0 \le n - k + 1$.

Kostya still does not know how many tiles he needs, so he considers the subsquares of all possible sizes. He doesn't want his bathroom to be too colorful. Help Kostya to count for each $k \le n$ the number of subsquares of size $k \times k$ that have at most q different colors of tiles. Two subsquares are considered different if their location on the stand is different.

Input

The first line contains two integers n and q ($1 \le n \le 1500$, $1 \le q \le 10$) — the size of the stand and the limit on the number of distinct colors in a subsquare.

Each of the next n lines contains n integers $c_{i,\,j}$ ($1 \le c_{i,\,j} \le n^2$): the j-th integer in the i-th line is the color of the tile in the cell $(i,\,j)$.

Output

For each k from 1 to n print a single integer — the number of subsquares of size $k \times k$ with no more than q different colors.

Examples

input

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

output

```
9
```

```
4
0
```

input

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

output

```
16
9
4
0
```

Note

In the first example all colors are distinct. Kostya doesn't want the subsquare have more than 4 colors, so he can buy any subsquare of size 1×1 or 2×2, but he can't buy a subsquare of size 3×3.

In the second example there are colors that appear multiple times. Because $q = 8$, Kostya can buy any subsquare of size 1×1 and 2×2, and any subsquare 3×3, because of such subsquare has 7 different colors. He can't buy the whole stand 4×4, because there are 9 colors.