Codeforces Round #707 (Div. 1, based on Moscow Open Olympiad in Informatics)

A. 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 ($4 \leq n \leq 200\,000$) — the size of the array.

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

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 $a_1 + a_2 = 1 + 3 = 3 + 1 = a_2 + a_3$

B. 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.

C. 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.

D. 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.

E. Subset Trick

time limit per test: 3 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Vanya invented an interesting trick with a set of integers.

Let an illusionist have a set of positive integers S. He names a positive integer x. Then an audience volunteer must choose some subset (possibly, empty) of S without disclosing it to the illusionist. The volunteer tells the illusionist the size of the chosen subset. And here comes the trick: the illusionist guesses whether the sum of the subset elements does not exceed x. The sum of elements of an empty subset is considered to be 0.

Vanya wants to prepare the trick for a public performance. He prepared some set of **distinct** positive integers S. Vasya wants the trick to be successful. He calls a positive number x *unsuitable*, if he can't be sure that the trick would be successful for every subset a viewer can choose.

Vanya wants to count the number of unsuitable integers for the chosen set S.

Vanya plans to try different sets S. He wants you to write a program that finds the number of unsuitable integers for the initial set S, and after each change to the set S. Vanya will make q changes to the set, and each change is one of the following two types:

- add a new integer a to the set S, or
- remove some integer a from the set S.

Input
The first line contains two integers n, q ($1 \le n, q \le 200\,000$) — the size of the initial set S and the number of changes.

The next line contains n **distinct** integers $s_1, s_2, \ldots, s_n$ ($1 \le s_i \le 10^{13}$) — the initial elements of S.

Each of the following q lines contain two integers t_i, a_i ($1 \le t_i \le 2$, $1 \le a_i \le 10^{13}$), describing a change:

- If $t_i = 1$, then an integer a_i is added to the set S. It is guaranteed that this integer is not present in S before this operation.
- If $t_i = 2$, then an integer a_i is removed from the set S. In is guaranteed that this integer is present in S before this operation.

Output
Print $q + 1$ lines.

In the first line print the number of unsuitable integers for the initial set S. In the next q lines print the number of unsuitable integers for S after each change.

Example

input

```
3 11
1 2 3
2 1
1 5
1 6
1 7
2 6
2 2
2 3
1 10
2 5
2 7
2 10
```

output

```
4
1
6
12
19
13
8
2
10
3
0
0
```

Note
In the first example the initial set is $S = \{1, 2, 3\}$. For this set the trick can be unsuccessful for $x \in \{1, 2, 3, 4\}$. For example, if $x = 4$, the volunteer can choose the subset $\{1, 2\}$ with sum $3 \le x$, and can choose the subset $\{2, 3\}$ with sum $5 > x$. However, in both cases the illusionist only know the same size of the subset (2), so he can't be sure answering making a guess. Since there is only one subset of size 3, and the sum of each subset of smaller size does not exceed 5, all $x \ge 5$ are suitable.

F. Cupboards Jumps

time limit per test: 6 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

In the house where Krosh used to live, he had n cupboards standing in a line, the i-th cupboard had the height of h_i. Krosh moved recently, but he wasn't able to move the cupboards with him. Now he wants to buy n new cupboards so that they look as similar to old ones as possible.

Krosh does not remember the exact heights of the cupboards, but for every three consecutive cupboards he remembers the height difference between the tallest and the shortest of them. In other words, if the cupboards' heights were $h_1, h_2, \ldots, h_n$, then Krosh remembers the values $w_i = \max(h_i, h_{i+1}, h_{i+2}) - \min(h_i, h_{i+1}, h_{i+2})$ for all $1 \le i \le n-2$.

Krosh wants to buy such n cupboards that all the values w_i remain the same. Help him determine the required cupboards' heights, or determine that he remembers something incorrectly and there is no suitable sequence of heights.

Input
The first line contains two integers n and C ($3 \le n \le 10^6$, $0 \le C \le 10^{12}$) — the number of cupboards and the limit on possible w_i.

The second line contains $n-2$ integers $w_1, w_2, \ldots, w_{n-2}$ ($0 \le w_i \le C$) — the values defined in the statement.

Output
If there is no suitable sequence of n cupboards, print "NO".

Otherwise print "YES" in the first line, then in the second line print n integers $h'_1, h'_2, \ldots, h'_n$ ($0 \le h'_i \le 10^{18}$) — the heights of the cupboards to buy, from left to right.

We can show that if there is a solution, there is also a solution satisfying the constraints on heights.

If there are multiple answers, print any.

Examples

input

```
7 20
4 8 12 16 20
```

output

```
YES
4 8 8 16 20 4 0
```

input

```
11 10
5 7 2 3 4 5 2 1 8
```

output

```
YES
1 1 6 8 6 5 2 0 0 1 8
```

input

```
6 9
1 6 9 0
```

output

```
NO
```

Note

Consider the first example:

- $w_1 = \max(4, 8, 8) - \min(4, 8, 8) = 8 - 4 = 4$
- $w_2 = \max(8, 8, 16) - \min(8, 8, 16) = 16 - 8 = 8$
- $w_3 = \max(8, 16, 20) - \min(8, 16, 20) = 20 - 8 = 12$
- $w_4 = \max(16, 20, 4) - \min(16, 20, 4) = 20 - 4 = 16$
- $w_5 = \max(20, 4, 0) - \min(20, 4, 0) = 20 - 0 = 20$

There are other possible solutions, for example, the following: $0, 1, 4, 9, 16, 25, 36$.