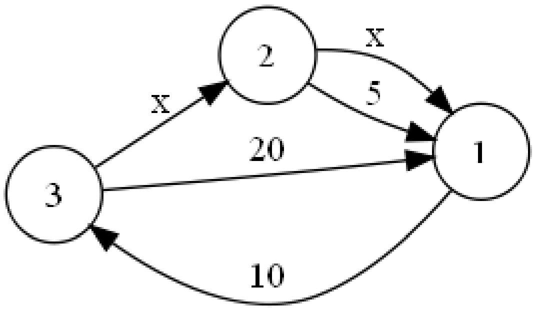
TASK	HIPERPROSTOR	ROTACIJE	SNJEGULJICA	SPIJUNI
input	standard input (*stdin*)			
output	standard output (*stdout*)			
time limit	1,5 sec	0,5 sec	1 sec	1 sec
memory limit	32 MB	256 MB	32 MB	32 MB
points	**100**	**100**	**100**	**100**
	400			

In the distant future, food is transported between planets via **one-way trade routes**. Each route directly connects two planets and has a known **transit time**.

The traders' guild plans to add some new routes utilizing a recently discovered technology – hyperspace travel. Travelling via hyperspace is also unidirectional. Since it is still experimental, hyperspace travel time is **not yet known**, however it is known not to depend on distances between planets, so each hyperspace route will take an **equal amount of time** to traverse.

The picture below shows an example of three interconnected planets with transit times shown. The planets are labelled with positive integers, and the hyperspace travel time is denoted by "**x**" (the picture corresponds to the second example input):



Transit time is measured in days and is always a **positive integer**.

The traders' guild wishes to analyze the consequences of introducing the new routes: for some two planets **A** and **B**, they want to know what are **all the possible values** of the **shortest path** total transit time from **A** to **B**, for all possible values of **x**. For example, in the situaton above, shortest path travel from planet 2 to planet 1 could take 5 (if $\mathbf{x} \geq 5$), 4, 3, 2, or 1 day (if $\mathbf{x} < 5$).

INPUT

The first line of input contains the two integers **P** and **R**, the number of planets and the number of routes, respectively ($1 \leq \mathbf{P} \leq 500$, $0 \leq \mathbf{R} \leq 10\ 000$).

Each of the following **R** lines contains two integers **C** and **D**, the planet labels ($1 \leq \mathbf{C}, \mathbf{D} \leq \mathbf{P}$, $\mathbf{C} \neq \mathbf{D}$), and **T**, the travel time from **C** to **D**. For conventional routes, **T** is an **integer** ($1 \leq \mathbf{T} \leq 1\ 000\ 000$), and for hyperspace routes, **T** is **the character "x"**. Multiple lines can exist between the same two planets.

The following line contains the integer **Q**, the number of queries ($1 \leq \mathbf{Q} \leq 10$).

Each of the following **Q** lines contains two integers **A** and **B**, the planet labels ($\mathbf{A} \neq \mathbf{B}$) representing a **query** by the traders' guild: "what are the possible values of shortest path transit time from **A** to **B**?".

OUTPUT

The output must contain **Q** rows, one per query.

Each row must contain two integers: the **number of different values** and **their sum**. If the number of different values is **unbounded**, output only "**inf**" in that row. If there is no path from **A** to **B**, the number of different values and their sum is 0.

SCORING

If the output is incorrect, but the first number in each of the **Q** rows is correct, the solution will be awarded 50% of points for that test case. *Note:* The output must contain both numbers in each row where the number of values is bounded in order to qualify.

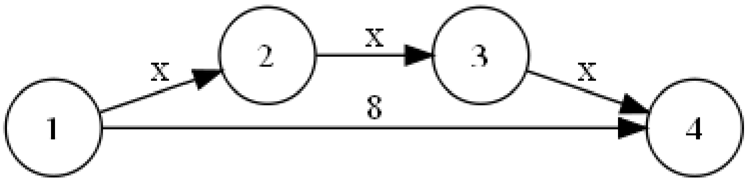
In test data worth a total of 50 points, the following constraints hold: **P** ≤ 30, **R** ≤ 300, and **T** ≤ 50.

SAMPLE TESTS

input	input
4 4 1 2 x 2 3 x 3 4 x 1 4 8 3 2 1 1 3 1 4	3 5 3 2 x 2 1 x 2 1 5 1 3 10 3 1 20 6 1 2 2 3 3 1 2 1 3 2 1 3
output	**output**
0 0 inf 3 17	inf 5 65 15 185 5 15 inf 1 10

Clarification of the first example:

1. There is no possible path from 2 to 1.
2. For any positive integer **x**, the shortest path from 1 to 3 takes 2**x** time, so the solution is "inf".
3. The shortest path from 1 to 4 can take 3 (for **x** = 1), 6 (for **x** = 2), or 8 (for **x** >= 3) time. 3+6+8=17



The famous archaeologist Diana Jones has discovered a secret passageway leading to hidden treasure near Nowhere, Kansas. The passageway is blocked by a stone gate which has an ancient unlocking mechanism chiselled into it. Fortunately, she has immediately recognized the chiselled symbols:

1. The unlocking mechanism is a table with **R** rows and **C** columns. Each cell contains a **unique positive integer** between 1 and **R*****C**, inclusive. At first glance, the numbers appear to be ordered randomly.
2. The mechanism contains cogwheels which Diana can use to rearrange the table cells. In one **move**, she can rotate any **2-by-2** group of adjacent cells clockwise by 90 degrees.
3. The gate will be unlocked when the numbers are rearranged in sorted row-major order (the upper left cell must contain 1, the cell to the right of it 2, and so on until the lower right cell, which must contain **R*****C**).

For example, for the initial arrangement shown in the first picture, two moves are sufficient to unlock the mechanism:

3	2	6
1	4	5

→

1	3	6
4	2	5

→

1	2	3
4	5	6

Write a program that, given the initial arrangement of cells, finds a **sequence of moves** that unlocks the mechanism. The number of moves needn't be optimal, however it **must not exceed 100 000**.

INPUT

The first line of input contains the two positive integers **R** and **C** ($2 \le R \le C \le 25$).

Each of the following **R** lines contains **C** positive integers Z_{ij} ($1 \le Z_{ij} \le R*C$), the numbers chiselled into the corresponding mechanism cells, which describes the initial arrangement.

OUTPUT

The output must contain the required sequence of moves, one per line. For each move, output two positive integers **M** and **N** ($1 \le M \le R-1$, $1 \le N \le C-1$) representing the row and column index of the **upper left cell** in the 2-by-2 group rotated in that move.

Note: For the given input data, a solution, not necessarily unique, **will always exist**.

SCORING

In test data worth a total of 40 points, **R*****C** will be at most 9.

In test data worth a total of 40 points, **R** will be equal to 2.

In test data worth a total of 60 points, at least one of the two constraints above will hold.

SAMPLE TESTS

input 2 3 3 2 6 1 4 5	**input** 3 3 1 2 3 4 6 9 7 5 8	**input** 2 4 1 2 7 3 5 6 8 4
output 1 1 1 2	**output** 2 2	**output** 1 3 1 3 1 3

Clarification of the first example: According to the picture in the problem description, the initial arrangement can be ordered in two moves: we first rotate the group with the upper left corner in row 1 and column 1, and then the group with the upper left corner in row 1 and column 2.

In a small village beyond seven hills and seven seas, Snow White lives together with **N** dwarves who spend all their time eating and playing League of Legends. Snow White wants to put an end to this, so she has organized gym classes for them.

At the beginning of each class, the dwarves must **stand in line**, ordered by their **height**. For the purposes of this task, assume that the dwarves have heights 1, 2, ..., **N** (each exactly once). However, the dwarves' intelligence has somewhat deteriorated from the unhealthy lifestyle, so they are incapable of ordering themselves by height. That's why Snow White helps them by issuing commands of the form:

- 1 **X Y** -- dwarves at positions **X** and **Y** in the line must switch places.

She also checks their ordering by issuing queries of the form:

- 2 **A B** -- do dwarves with heights **A**, **A**+1, ..., **B** (not necessarily in that order) occupy a **contiguous** subsequence of the current line?

Help the doofus dwarves follow Snow White's instructions and respond to her queries.

INPUT

The first line of input contains the two positive integers **N** and **M**, the number of dwarves and the number of Snow White's requests, respectively (2 ≤ **N** ≤ 200 000, 2 ≤ **M** ≤ 200 000).

The following line contains **N** space-separated positive integers from 1 to **N**, each exactly once, representing the initial arrangement of the dwarves.

Each of the following **M** lines contains a single Snow White's request, of the form "1 **X Y**" (1 ≤ **X**, **Y** ≤ **N**, **X** ≠ **Y**) or “2 **A B**” (1 ≤ **A** ≤ **B** ≤ **N**), as described in the problem statement.

OUTPUT

The output must contain one line for each request of type 2, containing the reply to the query, either “YES” or “NO”.

SCORING

In test data worth a total of 50 points, for all requests of type 2, the constraint **B** - **A** ≤ 50 holds.

SAMPLE TESTS

input

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

output

```
NO
YES
```

input

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

output

```
YES
NO
YES
NO
YES
```

You are *M*, the head of the intelligence agency which employs **N** spies with codenames from 1 to **N**. Each of the spies has been assigned a different country and obtained an important piece of information there. Your task is the following:

1. Organize meetings between some of the spies. In each meeting, exactly two spies meet and exchange all information that they have obtained themselves or learned from other spies in previous meetings. Organizing a top-secret meeting between two spies in different countries is difficult, so each potential meeting has a price.
2. After all the meetings have concluded, select a subset of spies and send them together on the world-saving (and woman-romancing) assignment. Sending a spy **k** on the assignment costs $\mathbf{M_k}$. For the assignment to succeed, it is important that the spies **together** know all the information obtained by the remaining spies.

Find the minimum total price of preparing and executing this assignment.

INPUT

The first line of input contains the positive integer **N**, the number of spies ($2 \le \mathbf{N} \le 1000$).

Each of the following **N** lines contains **N** positive integers not exceeding 10^6. The number in row **k** and column **m** represents the price of a meeting between spies **k** and **m** and, of course, equals the number in row **m** and column **k** (for **k** = **m** the number will be 0).

The following line contains **N** positive integers $\mathbf{M_k}$ ($1 \le \mathbf{M_k} \le 10^6$), the prices of sending each spy on the assignment, in order for spies 1, 2, ..., **N**.

OUTPUT

The first and only line of output must contain the required minimum total price.

SCORING

In test data with $\mathbf{N} \le 30$, which is worth a total of 40 points, it will be optimal to send at most four spies on the assignment.

In test data worth a total of 50 points, all prices of sending spies on assignment are equal.

SAMPLE TESTS

<table>
<tr>
<td>

input

```
3
0 6 9
6 0 4
9 4 0
7 7 7
```

output

```
17
```

</td>
<td>

input

```
3
0 17 20
17 0 10
20 10 0
15 9 12
```

output

```
34
```

</td>
<td>

input

```
5
0 3 12 15 11
3 0 14 3 20
12 14 0 11 7
15 3 11 0 15
11 20 7 15 0
5 10 10 10 10
```

output

```
28
```

</td>
</tr>
</table>

Clarification of the first example: We will organize meetings between spies 1 and 2, then 2 and 3, and send spy 2 on the assignment.

Clarification of the second example: We will organize a meeting between spies 2 and 3, and send spies 1 and 2 on the assignment.

Clarification of the third example: We will organize meetings between spies 2 and 4, then 1 and 2, then 3 and 5, and send spies 1 and 3 (or 1 and 5) on the assignment.