BOI 2026 Day 2

Language: en

task	type	time limit	memory limit
A Distances	standard	1.00 s	512 MB
B Hamilton	interactive	10.00 s	512 MB
C Sort	standard	1.00 s	512 MB

A Distances

You are given integers n and k. Your goal is to pick n distinct integer points on the xy-plane such that for exactly k pairs of points, the Euclidean distance between the points is an integer. Recall that the Euclidean distance between points (x_1, y_1) and (x_2, y_2) is

$$\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}.$$

It can be shown that a solution always exists under the constraints of this task.

Input

The only line contains two integers, n and k.

Output

Print n lines with the ith line containing two integers: the x and y coordinates of the ith point. The absolute value of every coordinate must be at most 10^9.

If there are multiple solutions, you can print any of them.

Constraints

- $1 \leq n \leq 100$
- $0 \leq k \leq n(n-1)/2$

Example

Input:
```
3 2
```

Output:
```
1 1
1 2
2 2
```

Explanation: The Euclidean distance between $(1, 1)$ and $(1, 2)$ is 1. The distance between $(1, 2)$ and $(2, 2)$ is also 1. However, the distance between $(1, 1)$ and $(2, 2)$ is $\sqrt{2}$, which is not an integer.

Scoring

Subtask	Constraints	Points
1	$n \leq 4$	11
2	$k = n(n-1)/2$	4
3	$k = 0$	6
4	$k \leq n$	19
5	$k \leq n(n-1)/8$	22
6	No additional constraints	38

B Hamilton

Consider a directed graph with n nodes numbered $1, 2, \ldots, n$. The graph is called a *tournament* if there is exactly one edge between all pairs of nodes in either direction. That is, for any pair of distinct nodes u and v, there is either an edge from u to v or from v to u.

A Hamiltonian cycle is a sequence $c_1, c_2, \ldots, c_n$ that visits all nodes and returns back to the start, following edges in the graph. For all $1 \leq i \leq n-1$, there must be an edge from c_i to c_{i+1}. Additionally, there must be an edge from c_n to c_1.

You can freely construct a tournament graph of n nodes. The numbering of the nodes is then shuffled. By making queries on the edge directions in the shuffled graph, can you find a Hamiltonian cycle?

Interaction

This is an interactive problem. Start by reading two integers n and t: the number of nodes and the number of test cases.

Next, print n lines to describe the tournament graph. On the uth of these lines, print n characters "`0`" or "`1`". A character "`1`" at position v indicates that there is an edge from u to v. There should be no edge from u to itself.

Then, t test cases follow. Each test case uses the same graph you provided, but the numbering is shuffled and kept secret by the grader. You may make some number of queries, after which you should report a Hamiltonian cycle.

To make a query, print "`? u v`", where $1 \leq u, v \leq n$ are distinct nodes in the shuffled graph. The grader responds with either "`>`" if the edge is from u to v, or "`<`" if the edge is from v to u.

Once you have found a Hamiltonian cycle, print "`!`" followed by n integers $c_1, c_2, \ldots, c_n$. Note that the numbers c_i should follow the shuffled numbering. The next test case begins immediately after you have printed the answer.

A testing script can be downloaded here. The beginning of the script contains instructions on how to use it.

Constraints

- $4 \leq n \leq 500$
- $1 \leq t \leq 200$

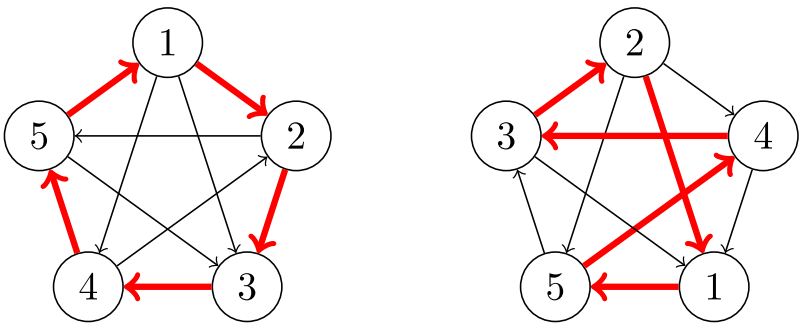
Example interaction

```
5 2

01110
00101
00010
01001
10100

? 1 2
>
? 2 3
>
? 3 4
>
? 4 5
>
? 5 1
>
! 1 2 3 4 5

? 1 2
<
? 1 5
>
? 4 3
>
? 4 5
<
? 3 2
>
! 1 5 4 3 2
```

Explanation: The nodes happen to be shuffled to the original order in the first test case and therefore $1, 2, 3, 4, 5$ is a Hamiltonian cycle.

In the second case, the node numbers $1, 2, 3, 4, 5$ are shuffled to $2, 4, 1, 5, 3$, in this order. The sequence $1, 5, 4, 3, 2$ is indeed a Hamiltonian cycle because $3, 4, 2, 5, 1$ was one in the original graph.

In the figure below, the original graph is shown on the left and the shuffled graph from the second test case is shown on the right. The two Hamiltonian cycles are highlighted in red.



Scoring

There is only one test input in each subtask, with $t = 200$ test cases. In each test case, the numbering of the graph is shuffled uniformly at random. Making more than 10^4 queries in a single test case results in the verdict WRONG ANSWER.

Let Q be the average number of queries made by your program among all test cases belonging to a subtask. You receive points for the subtask if Q is no greater than the specified limit.

Subtask	Constraints	Points
1	$n = 4, Q \leq 12$	5
2	$n = 50, Q \leq 1225$	7
3	$n = 50, Q \leq 300$	12
4	$n = 500, Q \leq 1500$	1–76

In subtask 4, you receive points according to the following formula:

$$\left\lfloor \frac{25\,000}{\max(750, Q) - 500} - 24 \right\rfloor$$

For example, if the average number of queries made by your program is $Q = 1500$, you get 1 point from the subtask. If $Q = 1000$, you get 26 points, and if $Q = 750$, you get 76 points.

C Sort

You are given an array $x_1, x_2, \ldots, x_n$ of n integers. Your task is to answer q queries (a, b). In one operation, you are allowed to do one of the following:

- sort the first a numbers in nondecreasing order, or
- sort the last b numbers in nondecreasing order.

What is the minimum number of operations needed to sort the whole array in nondecreasing order? In each query, the array starts from the initial values $x_1, x_2, \ldots, x_n$.

Input

The first line contains two integers n and q: the length of the array and the number of queries.

The second line contains n integers $x_1, x_2, \ldots, x_n$: the contents of the array.

The next q lines describe the queries. Each line contains two integers a and b.

Output

Print q lines with the answers to each query. If it is impossible to sort the array, print "-1".

Constraints

- $1 \leq n, q \leq 2 \cdot 10^5$
- $1 \leq x_i \leq 10^9$
- $1 \leq a, b \leq n$ in all queries

Example

Input:
```
6 3
3 1 4 1 5 9
4 1
3 3
2 5
```

Output:
```
1
-1
2
```

Explanation: In the first query, the array can be sorted in one operation by sorting the first 4 numbers.

The array cannot be sorted with the available operations in the second query.

In the third query, the array can be sorted in two operations: start by sorting the first 2 numbers and then sort the last 5 numbers.

Scoring

Subtask	Constraints	Points
1	$n, q \leq 10$ and $a + b \leq n$ in all queries	6
2	$n, q \leq 10$	5
3	$a + b \leq n$ in all queries	7
4	$1 \leq x_i \leq 2$	14
5	$n, q \leq 5000$ and the array is a permutation of the numbers $1, 2, \ldots, n$	23
6	$n, q \leq 5000$	12
7	No additional constraints	33