A. Valeriy and Deque

time limit per test: 6 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Recently, on the course of algorithms and data structures, Valeriy learned how to use a deque. He built a deque filled with n elements. The i-th element is a_i ($i = 1, 2, \ldots, n$). He gradually takes the first two leftmost elements from the deque (let's call them A and B, respectively), and then does the following: if $A > B$, he writes A to the beginning and writes B to the end of the deque, otherwise, he writes to the beginning B, and A writes to the end of the deque. We call this sequence of actions an operation.

For example, if deque was $[2, 3, 4, 5, 1]$, on the operation he will write $B = 3$ to the beginning and $A = 2$ to the end, so he will get $[3, 4, 5, 1, 2]$.

The teacher of the course, seeing Valeriy, who was passionate about his work, approached him and gave him q queries. Each query consists of the singular number m_j $(j = 1, 2, \ldots, q)$. It is required for each query to answer which two elements he will pull out on the m_j-th operation.

Note that **the queries are independent** and for each query the numbers A and B should be **printed in the order in which they will be pulled out of the deque**.

Deque is a data structure representing a list of elements where insertion of new elements or deletion of existing elements can be made from both sides.

Input

The first line contains two integers n and q ($2 \leq n \leq 10^5$, $0 \leq q \leq 3 \cdot 10^5$) — the number of elements in the deque and the number of queries. The second line contains n integers a_1, a_2, ..., a_n, where a_i $(0 \leq a_i \leq 10^9)$ — the deque element in i-th position. The next q lines contain one number each, meaning m_j ($1 \leq m_j \leq 10^{18}$).

Output

For each teacher's query, output two numbers A and B — the numbers that Valeriy pulls out of the deque for the m_j-th operation.

Examples

input

```
5 3
1 2 3 4 5
1
2
10
```

output

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

input

```
2 0
0 0
```

output

```
```

Note

Consider all 10 steps for the first test in detail:

1. $[1, 2, 3, 4, 5]$ — on the first operation, A and B are 1 and 2, respectively.
   So, 2 we write to the beginning of the deque, and 1 — to the end.

   We get the following status of the deque: $[2, 3, 4, 5, 1]$.

2. $[2, 3, 4, 5, 1] \Rightarrow A = 2, B = 3$.
3. $[3, 4, 5, 1, 2]$
4. $[4, 5, 1, 2, 3]$
5. $[5, 1, 2, 3, 4]$
6. $[5, 2, 3, 4, 1]$
7. $[5, 3, 4, 1, 2]$
8. $[5, 4, 1, 2, 3]$
9. $[5, 1, 2, 3, 4]$

10. $[5, 2, 3, 4, 1] \Rightarrow A = 5, B = 2$.

B. Tolik and His Uncle

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

This morning Tolik has understood that while he was sleeping he had invented an incredible problem which will be a perfect fit for Codeforces! But, as a "Discuss tasks" project hasn't been born yet (in English, well), he decides to test a problem and asks his uncle.

After a long time thinking, Tolik's uncle hasn't any ideas on how to solve it. But, he doesn't want to tell Tolik about his inability to solve it, so he hasn't found anything better than asking you how to solve this task.

In this task you are given a cell field $n \cdot m$, consisting of n rows and m columns, where point's coordinates (x, y) mean it is situated in the x-th row and y-th column, considering numeration from one ($1 \le x \le n, 1 \le y \le m$). Initially, you stand in the cell $(1, 1)$. Every move you can jump from cell (x, y), which you stand in, by any non-zero vector (dx, dy), thus you will stand in the $(x + dx, y + dy)$ cell. Obviously, you can't leave the field, but also there is one more important condition — you're not allowed to use one vector twice. Your task is to visit each cell of the field exactly once (the initial cell is considered as already visited).

Tolik's uncle is a very respectful person. Help him to solve this task!

Input

The first and only line contains two positive integers n, m ($1 \le n \cdot m \le 10^6$) — the number of rows and columns of the field respectively.

Output

Print "-1" (without quotes) if it is impossible to visit every cell exactly once.

Else print $n \cdot m$ pairs of integers, i-th from them should contain two integers x_i, y_i ($1 \le x_i \le n, 1 \le y_i \le m$) — cells of the field in order of visiting, so that all of them are distinct and vectors of jumps between them are distinct too.

Notice that the first cell should have $(1, 1)$ coordinates, according to the statement.

Examples

input

```
2 3
```

output

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

input

```
1 1
```

output

```
1 1
```

Note

The vectors from the first example in the order of making jumps are $(0, 2), (0, -1), (1, 0), (0, 1), (0, -2)$.

C. Serge and Dining Room

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Serge came to the school dining room and discovered that there is a big queue here. There are m pupils in the queue. He's not sure now if he wants to wait until the queue will clear, so he wants to know which dish he will receive if he does. As Serge is very tired, he asks you to compute it instead of him.

Initially there are n dishes with costs $a_1, a_2, \ldots, a_n$. As you already know, there are the queue of m pupils who have $b_1, \ldots, b_m$ togrogs respectively (pupils are enumerated by queue order, i.e the first pupil in the queue has b_1 togrogs and the last one has b_m togrogs)

Pupils think that the most expensive dish is the most delicious one, so every pupil just buys the most expensive dish for which he has money (every dish has a single copy, so when a pupil has bought it nobody can buy it later), and if a pupil doesn't have money for any dish, he just leaves the queue (so brutal capitalism...)

But money isn't a problem at all for Serge, so Serge is buying the most expensive dish if there is at least one remaining.

Moreover, Serge's school has a very unstable economic situation and the costs of some dishes or number of togrogs of some pupils can change. More formally, you must process q queries:

- change a_i to x. It means that the price of the i-th dish becomes x togrogs.
- change b_i to x. It means that the i-th pupil in the queue has x togrogs now.

Nobody leaves the queue during those queries because a saleswoman is late.

After every query, you must tell Serge price of the dish which he will buy if he has waited until the queue is clear, or -1 if there are no dishes at this point, according to rules described above.

Input

The first line contains integers n and m ($1 \le n, m \le 300\ 000$) — number of dishes and pupils respectively. The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^6$) — elements of array a. The third line contains m integers $b_1, b_2, \ldots, b_m$ ($1 \le b_i \le 10^6$) — elements of array b. The fourth line conatins integer q ($1 \le q \le 300\ 000$) — number of queries.

Each of the following q lines contains as follows:

- if a query changes price of some dish, it contains 1, and two integers i and x ($1 \le i \le n$, $1 \le x \le 10^6$), what means a_i becomes x.
- if a query changes number of togrogs of some pupil, it contains 2, and two integers i and x ($1 \le i \le m$, $1 \le x \le 10^6$), what means b_i becomes x.

Output

For each of q queries prints the answer as the statement describes, the answer of the i-th query in the i-th line (the price of the dish which Serge will buy or -1 if nothing remains)

Examples

input

```
1 1
1
1
1
1 1 100
```

output

```
100
```

input

```
1 1
1
1
1
2 1 100
```

output

```
-1
```

input

```
4 6
1 8 2 4
3 3 6 1 5 2
3
1 1 1
2 5 10
1 1 6
```

output

```
8
-1
4
```

Note

In the first sample after the first query, there is one dish with price 100 togrogs and one pupil with one togrog, so Serge will buy the dish with price 100 togrogs.

In the second sample after the first query, there is one dish with price one togrog and one pupil with 100 togrogs, so Serge will get nothing.

In the third sample after the first query, nobody can buy the dish with price 8, so Serge will take it. After the second query, all dishes will be bought, after the third one the third and fifth pupils will by the first and the second dishes respectively and nobody will by the fourth one.

D. Fedor Runs for President

time limit per test: 2 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

Fedor runs for president of Byteland! In the debates, he will be asked how to solve Byteland's transport problem. It's a really hard problem because of Byteland's transport system is now a tree (connected graph without cycles). Fedor's team has found out in the ministry of transport of Byteland that there is money in the budget only for one additional road. In the debates, he is going to say that he will build this road as a way to maximize the number of distinct simple paths in the country. A simple path is a path which goes through every vertex no more than once. Two simple paths are named distinct if sets of their edges are distinct.

But Byteland's science is deteriorated, so Fedor's team hasn't succeeded to find any scientists to answer how many distinct simple paths they can achieve after adding exactly one edge on the transport system?

Help Fedor to solve it.

An edge can be added between vertices that are already connected, but it can't be a loop.

In this problem, we consider only simple paths of length at least two.

Input

The first line contains one integer n ($2 \le n \le 500\ 000$) — number of vertices in Byteland's transport system.

Each of the following $n-1$ lines contains two integers v_i and u_i ($1 \le v_i, u_i \le n$). It's guaranteed that the graph is tree.

Output

Print exactly one integer — a maximal number of simple paths that can be achieved after adding one edge.

Examples

input

```
2
1 2
```

output

```
2
```

input

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

output

```
11
```

input

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

output

```
29
```

E. Alesya and Discrete Math

time limit per test: 5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

We call a function *good* if its domain of definition is some set of integers and if in case it's defined in x and $x-1$, $f(x) = f(x-1) + 1$ or $f(x) = f(x-1)$.

Tanya has found n *good* functions $f_1, \ldots, f_n$, which are defined on all integers from 0 to 10^{18} and $f_i(0) = 0$ and $f_i(10^{18}) = L$ for all i from 1 to n. It's an notorious coincidence that n is a divisor of L.

She suggests Alesya a game. Using one question Alesya can ask Tanya a value of any single function in any single point. To win Alesya must choose integers l_i and r_i ($0 \le l_i \le r_i \le 10^{18}$), such that $f_i(r_i) - f_i(l_i) \ge \frac{L}{n}$ (here $f_i(x)$ means the value of i-th function at point x) for all i such that $1 \le i \le n$ so that for any pair of two functions their segments $[l_i, r_i]$ don't intersect (but may have one common point).

Unfortunately, Tanya doesn't allow to make more than $2 \cdot 10^5$ questions. Help Alesya to win!

It can be proved that it's always possible to choose $[l_i, r_i]$ which satisfy the conditions described above.

It's guaranteed, that Tanya doesn't change functions during the game, i.e. interactor is not adaptive

Input

The first line contains two integers n and L ($1 \le n \le 1000$, $1 \le L \le 10^{18}$, n is a divisor of L) — number of functions and their value in 10^{18}.

Output

When you've found needed l_i, r_i, print $"!"$ without quotes on a separate line and then n lines, i-th from them should contain two integers l_i, r_i divided by space.

Interaction

To ask $f_i(x)$, print symbol "?" without quotes and then two integers i and x ($1 \le i \le n$, $0 \le x \le 10^{18}$). Note, you must flush your output to get a response.

After that, you should read an integer which is a value of i-th function in point x.

You're allowed not more than $2 \cdot 10^5$ questions.

To flush you can use (just after printing an integer and end-of-line):

- fflush(stdout) in C++;
- System.out.flush() in Java;
- stdout.flush() in Python;
- flush(output) in Pascal;
- See the documentation for other languages.

Hacks:

Only tests where $1 \le L \le 2000$ are allowed for hacks, for a hack set a test using following format:

The first line should contain two integers n and L ($1 \le n \le 1000$, $1 \le L \le 2000$, n is a divisor of L) — number of functions and their value in 10^{18}.

Each of n following lines should contain L numbers l_1, l_2, ... , l_L ($0 \le l_j < 10^{18}$ for all $1 \le j \le L$ and $l_j < l_{j+1}$ for all $1 < j \le L$), in i-th of them l_j means that $f_i(l_j) < f_i(l_j + 1)$.

Example

input

```
5 5
? 1 0
? 1 1
? 2 1
? 2 2
? 3 2
? 3 3
? 4 3
? 4 4
? 5 4
? 5 5
!
0 1
1 2
2 3
3 4
4 5
```

output

```
0
1
1
2
2
3
3
4
4
4
5
```

Note

In the example Tanya has 5 same functions where $f(0) = 0$, $f(1) = 1$, $f(2) = 2$, $f(3) = 3$, $f(4) = 4$ and all remaining points have value 5.

Alesya must choose two integers for all functions so that difference of values of a function in its points is not less than $\frac{L}{n}$ (what is 1 here) and length of intersection of segments is zero.

One possible way is to choose pairs $[0, 1]$, $[1, 2]$, $[2, 3]$, $[3, 4]$ and $[4, 5]$ for functions 1, 2, 3, 4 and 5 respectively.