Codeforces Round #569 (Div. 2)

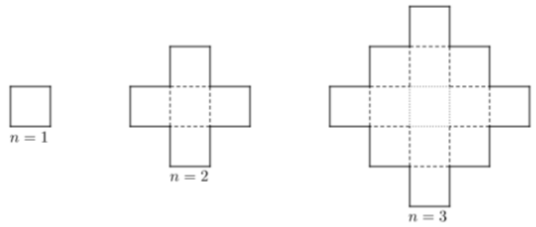
A. Alex and a Rhombus

1 second, 256 megabytes

While playing with geometric figures Alex has accidentally invented a concept of a n-*th order rhombus* in a cell grid.

A 1-*st order rhombus* is just a square 1×1 (i.e just a cell).

A n-*th order rhombus* for all $n \geq 2$ one obtains from a $n-1$-*th order rhombus* adding all cells which have a common side with it to it (look at the picture to understand it better).



Alex asks you to compute the number of cells in a n-*th order rhombus*.

Input

The first and only input line contains integer n ($1 \leq n \leq 100$) — order of a rhombus whose numbers of cells should be computed.

Output

Print exactly one integer — the number of cells in a n-*th order rhombus*.

input
1
output
1

input
2
output
5

input
3
output
13

Images of rhombus corresponding to the examples are given in the statement.

B. Nick and Array

1 second, 256 megabytes

Nick had received an awesome array of integers $a = [a_1, a_2, \ldots, a_n]$ as a gift for his 5 birthday from his mother. He was already going to explore its various properties but after unpacking he was disappointed a lot because the product $a_1 \cdot a_2 \cdot \ldots a_n$ of its elements seemed to him not large enough.

He was ready to throw out the array, but his mother reassured him. She told him, that array would not be spoiled after the following operation: choose any index i ($1 \leq i \leq n$) and do $a_i := -a_i - 1$.

For example, he can change array $[3, -1, -4, 1]$ to an array $[-4, -1, 3, 1]$ after applying this operation to elements with indices $i = 1$ and $i = 3$.

Kolya had immediately understood that sometimes it's possible to increase the product of integers of the array a lot. Now he has decided that he wants to get an array with the maximal possible product of integers using only this operation with its elements (possibly zero, one or more times, as many as he wants), it is not forbidden to do this operation several times for the same index.

Help Kolya and print the array with the maximal possible product of elements $a_1 \cdot a_2 \cdot \ldots a_n$ which can be received using only this operation in some order.

If there are multiple answers, print any of them.

Input

The first line contains integer n ($1 \leq n \leq 10^5$) — number of integers in the array.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($-10^6 \leq a_i \leq 10^6$) — elements of the array

Output

Print n numbers — elements of the array with the maximal possible product of elements which can be received using only this operation in some order from the given array.

If there are multiple answers, print any of them.

input
4 2 2 2 2
output
-3 -3 -3 -3

input
1 0
output
0

input
3 -3 -3 2
output
-3 -3 2

C. Valeriy and Deque

6 seconds, 256 megabytes

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 ($2 \le n \le 10^5$, $0 \le q \le 3 \cdot 10^5$) — 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 $(0 \le a_i \le 10^9)$ — the deque element in i-th position. The next q lines contain one number each, meaning m_j $(1 \le m_j \le 10^{18})$.

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.

input
5 3 1 2 3 4 5 1 2 10
output
1 2 2 3 5 2

input
2 0 0 0
output

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

D. Tolik and His Uncle

1 second, 256 megabytes

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.

input
2 3
output
1 1 1 3 1 2 2 2 2 3 2 1

input
1 1
output
1 1

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

E. Serge and Dining Room

4 seconds, 256 megabytes

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)

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
```

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.