# Arcadia Computing Contest 1 P4 - Cyclic Sorting

**Time limit:** 3.0s **Memory limit:** 256M

Robert likes computer science! In his computer science course, he is learning about *cyclic shifts*. A *cyclic shift* of a list is the new list formed by moving the last element to the front, or moving the first element to the back. For example, after performing one cyclic shift, the possible lists created from $[1, 2, 3, 4]$ are $[4, 1, 2, 3]$ and $[2, 3, 4, 1]$.

Robert strolls into his computer science classroom. Unbeknownst to him, his teacher is giving the class a pop quiz! The quiz consists of only $1$ question:

> You are given an array $A$ with length $N$. Determine the minimum number of cyclic shifts to sort the array in non-decreasing order.

Robert finishes the quiz in a breeze. However, he still has $40$ minutes to kill before the end of class! So, he challenges himself to solve the same problem, but with $Q$ updates. In each update, he asks himself:

> If we replace $A_i$ with $x$, what is the minimum number of cyclic shifts to sort the list in non-decreasing order?

For each update, output the minimum number of cyclic shifts. If it is not possible to sort the list, output `-1`.

Note that updates are persistent. This means that if $A_i$ is changed to $x$, during the next update, $A_i$ will remain equal to $x$.

## Constraints

$1 \leq N, Q \leq 2 \times 10^5$

$1 \leq A_i \leq 10^9$

**Subtask 1 [20%]**

$1 \leq N, Q \leq 1000$

**Subtask 2 [80%]**

No additional constraints.

## Input Specification

The first line will contain two space-separated integers, $N$ and $Q$.

The second line will contain $N$ space-separated integers, denoting the elements of the array $A$.

The next $Q$ lines will each contain two integers: $i$ and $x$, denoting the parameters to the question. $i$ is one-indexed, meaning that $i = 1$ corresponds to the first element of the array.

## Output Specification

For each query, output a single integer: the minimum number of cyclic shifts to sort the array, or `-1` if it is not possible.

## Sample Input

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

## Sample Output

```
0
1
-1
```

## Explanation for Sample

After the first update, our array becomes:

$\{1, 2, 3, 3, 5\}$

Clearly, this array is already sorted. So, we don't need any operations to sort it.

After the second update, our array becomes:

$\{6, 2, 3, 3, 5\}$

We can cycle this array one time, obtaining the array $\{2, 3, 3, 5, 6\}$ which is sorted.

After the third update, our array becomes:

$\{6, 2, 1, 3, 5\}$

Clearly, no matter how many times we cycle the array, it will remain unsorted.