Branch Cutting (lumberjacks)

A tree was planted in the main square of Byteland many many years ago. It has been neglected in recent years and has grown incredibly large, causing an obstruction to traffic. In order to solve the situation, the Gardening Department of the city wants to cut some branches off the tree.

Figure 1: The tree in Byteland's square.

The tree has N branches numbered from 0 to $N-1$. The Gardening Department sends N lumberjacks to cut down the branches as quickly as possible. The lumberjacks devise the following method. Everyone climbs up a branch (each on a different branch) and start working at the same time: the lumberjack sitting on branch i starts cutting down branch $(i+1) \mod N$. Unfortunately, if the i-th branch is cut, the lumberjack on the i-th branch will fall. If a lumberjack falls, he cannot continue cutting a branch.

The Gardening Department wants to calculate the *safety value* of the cutting. The safety value is the number of lumberjacks who **does not** fall off the tree.

The branches take different times to cut. Initially, it takes A_i seconds to cut branch i.

You are given the initial A_i times, and you are also given Q queries in the form (X_i, C_i): following query i, the time to cut branch X_i becomes C_i seconds (the changes are permanent). Your task is to calculate the safety value initially, and after each query.

☞ Among the attachments of this task you may find a template file lumberjacks.* with a sample incomplete implementation.

Input

The input file consists of:

- a line containing integers N, Q.
- a line containing the N integers $A_0, \ldots, A_{N-1}$.
- Q lines, the i-th of which consisting of integers X_i, C_i.

Output

The output file must contain a single line consisting of $Q + 1$ integers: the initial *safety value* and the *safety value* after each one of the Q queries.

Constraints

- $2 \leq N \leq 200\,000$.
- $0 \leq Q \leq 200\,000$.
- $1 \leq A_i \leq 1\,000\,000\,000$ for each $i = 0 \ldots N - 1$.
- $0 \leq X_i < N$ for each $i = 0 \ldots Q - 1$.
- $1 \leq C_i \leq 1\,000\,000\,000$ for each $i = 0 \ldots Q - 1$.

Scoring

Your program will be tested against several test cases grouped in subtasks. In order to obtain the score of a subtask, your program needs to correctly solve all of its test cases.

– **Subtask 1** (0 points) Examples.

– **Subtask 2** (8 points) $N \leq 100$, $Q \leq 100$.

– **Subtask 3** (15 points) $1 \leq A_i \leq 2$ for each $i = 0 \ldots N - 1$.
$1 \leq C_i \leq 2$ for each $i = 0 \ldots Q - 1$.

– **Subtask 4** (10 points) $N \leq 1000$, $Q \leq 1000$.

– **Subtask 5** (25 points) $N \leq 8000$, $Q \leq 8000$.

– **Subtask 6** (42 points) No additional limitations.

Examples

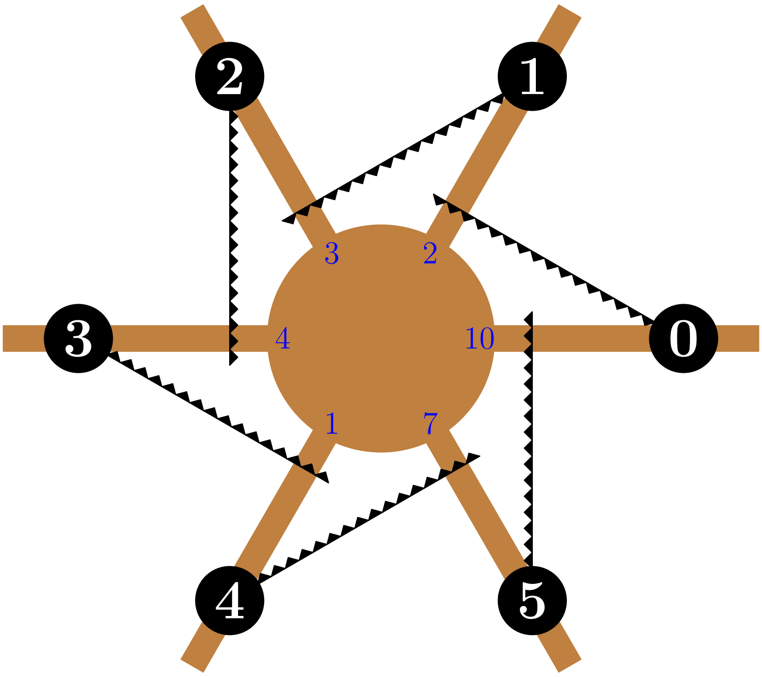
input	output

6 3
10 2 3 4 1 7
4 6
2 2
5 1
	2 3 2 2

8 2
3 3 3 3 2 1 4 5
7 1
2 5
	1 2 3

Explanation

In the **first sample case** the initial situation is the following.



- after 1 second branch 4 gets cut off, therefore branch 5 won't get cut off later.
- after 2 seconds branch 1 gets cut off, therefore branch 2 won't get cut off 1 second later.
- after 4 seconds branch 3 gets cut off.
- after 10 seconds branch 0 get cut off.

Only branches 2 and 5 did not get cut off, therefore the **initial** answer is 2.

After the **first** query branches 0, 2 and 4 won't get cut off, therefore the answer is 3.

After the **second** query branches 3 and 5 won't get cut off, therefore the answer is 2.

After the **third** query branches 0 and 3 won't get cut off, therefore the answer is 2.

In the **second sample case**.

Before the first query:

- after 1 second branch 5 gets cut off, therefore branch 6 won't get cut off later.
- after 2 seconds branch 4 gets cut off.
- after 3 seconds branches $0, 1, 2$ and 3 gets cut off simultaneously.
- after 5 seconds branch 7 gets cut off.

At the end only branch 6 did not get cut off, therefore the **initial** answer is 1.

After the **first** query branches 0 and 6 won't get cut off, therefore the answer is 2.

After the **second** query branches 0, 2 and 6 won't get cut off, therefore the answer is 3.