Bitwise Party (andxor)

The new year has come and just like all the competitive programmers, our hero Stefan received an array with positive integers as a Christmas present as well! Now he wants to explore the array and to find new interesting properties as he always likes doing.
For this year, he wants to find out the maximum number of values we can take from the array such that both the bitwise *AND* and the bitwise *XOR* are different from zero[1]. Since this is too easy for him, he also modifies his array and wants you to answer to the same question after each update.

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

Input

The first line contains two integers: N, the number of integers in the array, and Q, the number of queries.

The next line contains N integers, v_i, the initial values of the array, each one separated by a space.

The next Q lines contain a query each. Each query is composed of two integers: *pos*, the position of the element to change, and *val*, the new value of the element.

Output

You need to write $Q + 1$ lines.

The first line should contain the answer before the first query.

In the i-th line (with $i \geq 1$) you should output the answer after the first i queries.

[1] You can find more information about bitwise operations at `https://en.wikipedia.org/wiki/Bitwise_operation#Bitwise_operators`. In C, C++ and Python the operators for bitwise-and and bitwise-xor are `&` and `^`, respectively. In Pascal they are `and` and `xor`.

Constraints

- $1 \leq N \leq 200\,000$.
- $1 \leq Q \leq 200\,000$.
- $1 \leq v_i, val < 2^{20}$.
- $1 \leq pos \leq N$.

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** (10 points) $1 \leq v_i, val \leq 2$.

– **Subtask 3** (20 points) $1 \leq N, Q, v_i, val < 2^7$.

– **Subtask 4** (20 points) $1 \leq N, Q, v_i, val < 2^{10}$.

– **Subtask 5** (50 points) No additional limitations.

Examples

input	output
3 1 11 5 14 3 15	2 3
5 8 3 2 4 1 5 1 1 2 3 3 3 4 2 3 6 4 5 5 2 1 6	3 3 4 5 4 3 4 3 4

Explanation

In the **first sample case**, before any query all the possible ways to take the elements are:

v_1	v_2	v_3			
11	5	14	`and`	`xor`	valid
-	-	-	0000	0000	no
1011	-	-	1011	1011	yes
-	0101	-	0101	0101	yes
-	-	1110	1110	1110	yes
1011	0101	-	0001	1110	yes
1011	-	1110	1010	0101	yes
-	0101	1110	0100	1011	yes
1011	0101	1110	0000	0000	no

There are 6 ways to select elements such that the bitwise *AND* and the bitwise *XOR* are non-zero. At most 2 numbers can be selected doing so.

After the first update, the possible ways to take the elements are:

v_1	v_2	v_3			
11	5	15	`and`	`xor`	valid
-	-	-	0000	0000	no
1011	-	-	1011	1011	yes
-	0101	-	0101	0101	yes
-	-	1111	1111	1111	yes
1011	0101	-	0001	1110	yes
1011	-	1111	1011	0100	yes
-	0101	1111	0101	1010	yes
1011	0101	1111	0001	0001	yes

It's possible to take all 3 elements.