Kick Start 2019 - Round D

X or What?

Problem

Steven has an array of **N** non-negative integers. The i-th integer (indexed starting from 0) in the array is $\mathbf{A_i}$.

Steven really likes subintervals of **A** that are *xor-even*. Formally, a subinterval of **A** is a pair of indices (L, R), denoting the elements $\mathbf{A_L}$, $\mathbf{A_{L+1}}$, ..., $\mathbf{A_{R-1}}$, $\mathbf{A_R}$. The xor-sum of this subinterval is $\mathbf{A_L}$ xor $\mathbf{A_{L+1}}$ xor ... xor $\mathbf{A_{R-1}}$ xor $\mathbf{A_R}$, where xor is the bitwise exclusive or.

A subinterval is *xor-even* if its xor-sum has an even number of set bits in its binary representation.

Steven would like to make **Q** modifications to the array. The i-th modification changes the $\mathbf{P_i}$-th (indexed from 0) element to $\mathbf{V_i}$. Steven would like to know, what is the size of the xor-even subinterval of **A** with the most elements after each modification?

Input

The first line of the input gives the number of test cases, **T**. **T** test cases follow.

Each test case starts with a line containing two integers **N** and **Q**, denoting the number of elements in Steven's array and the number of modifications, respectively.

The second line contains **N** integers. The i-th of them gives $\mathbf{A_i}$ indicating the i-th integer in Steven's array.

Then, **Q** lines follow, describing the modifications. The i-th line contains $\mathbf{P_i}$ and $\mathbf{V_i}$, The i-th modification changes the $\mathbf{P_i}$-th element to $\mathbf{V_i}$. indicating that the i-th modification changes the $\mathbf{P_i}$-th (indexed from 0) element to $\mathbf{V_i}$.

Output

For each test case, output one line containing `Case #x: y_1 y_2 ... y_Q`, where `x` is the test case number (starting from 1) and `y_i` is the number of elements in the largest xor-even subinterval of **A** after the i-th modification. If there are no xor-even subintervals, then output 0.

Limits

Time limit: 40 seconds per test set.
Memory limit: 1GB.
$1 \leq \mathbf{T} \leq 100$.
$0 \leq \mathbf{A_i} < 1024$.
$0 \leq \mathbf{P_i} < \mathbf{N}$.
$0 \leq \mathbf{V_i} < 1024$.

Test set 1 (Visible)

$1 \le \mathbf{N} \le 100$.
$1 \le \mathbf{Q} \le 100$.

Test set 2 (Hidden)

$1 \le \mathbf{N} \le 10^5$.
$1 \le \mathbf{Q} \le 10^5$.

Sample

Sample Input

```
2
4 3
10 21 3 7
1 13
0 32
2 22
5 1
14 1 15 20 26
4 26
```

Sample Output

```
Case #1: 4 3 4
Case #2: 4
```

In Sample Case 1, **N** = 4 and **Q** = 3.

- After the 1st modification, **A** is [10, 13, 3, 7]. The subinterval (0, 3) has xor-sum 10 xor 13 xor 3 xor 7 = 3. In binary, the xor-sum is 11_2, which has an even number of 1 bits, so the subinterval is xor-even. This is the largest subinterval possible, so the answer is 4.
- After the 2nd modification, **A** is [32, 13, 3, 7]. The largest xor-even subinterval is (0, 2), which has xor-sum 32 xor 13 xor 3 = 46. In binary, this is 101110_2.
- After the 3rd modification, **A** is [32, 13, 22, 7]. The largest xor-even subinterval is (0, 3) again, which has xor-sum 32 xor 13 xor 22 xor 7 = 60. In binary, this is 111100_2.

In Sample Case 2, **N** = 5 and **Q** = 1. After the 1st modification, **A** is [14, 1, 15, 20, 26]. The largest xor-even subinterval is (1, 4), which has xor sum 1 xor 15 xor 20 xor 26 = 0. In binary, this is 0_2.