Binary Beaver

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	512 megabytes

Busy Beaver has an array of N integers $a_1, a_2, \ldots, a_N$ with $1 \le a_i < 2^K$. As a coder, he loves writing numbers in binary!

Define the *roundness* of an array of positive integers as the total number of trailing 0's in the binary representations of all its elements. For example, the array $[2, 3, 6, 8, 10]$ is written in binary as $[1\underline{0}, 11, 11\underline{0}, 1\underline{000}, 101\underline{0}]$, so its roundness is $1 + 0 + 1 + 3 + 1 = 6$.

Busy Beaver may choose an integer x such that $0 \le x < 2^K$ and $x \neq a_i$ for all i, and then replace every element by $a_i \oplus x$, where $\oplus$ is bitwise XOR. For the current array, find the maximum possible roundness after this operation.

Then, Q updates will follow. Each query updates $a_i \leftarrow v$ for some $1 \le i \le N$ and $1 \le v < 2^K$. Note that **updates are persistent**, meaning that each update modifies the array, and subsequent updates should be applied to the current state.

Output the maximum roundness for the initial array, and again after each update.

Input

The first line contains a single integer T ($1 \le T \le 10^4$) — the number of test cases.

The first line of each test case contains three space-separated integers N, Q, and K ($1 \le N \le 2 \cdot 10^5$, $0 \le Q \le 2 \cdot 10^5$, $1 \le K \le 60$).

The second line of each test case contains N space-separated integers $a_1, \ldots, a_N$ ($1 \le a_i < 2^K$) — the members of the array a.

The next Q lines each contain two integers i and v ($1 \le i \le N$, $1 \le v < 2^K$), describing an update.

The sum of N across all test cases does not exceed $2 \cdot 10^5$.

The sum of Q across all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, output $Q + 1$ lines.

The first line should contain a single integer — the answer before any updates.

The i-th of the next Q lines should contain the answer after the i-th update.

Scoring

There are three subtasks for this problem. Let $\sum N$ denote the sum of N across all test cases and let $\sum Q$ denote the sum of Q across all test cases.

- (20 points): $\sum N, \sum Q \le 1000, K \le 25$. In addition, it is guaranteed that $K > 1$, $a_i < 2^{K-1}$ for all i, and $v < 2^{K-1}$ in all updates.
- (30 points): $\sum N, \sum Q \le 1000, K \le 25$.
- (50 points): No additional constraints.

Example

standard input	standard output
2 5 0 3 1 1 2 3 4 4 4 3 1 3 5 7 2 4 1 4 3 4 4 4	5 0 4 4 6 8

Note

For the first test case, the optimal value of x is $x = 5$, which gives $a \oplus 5 = [4, 4, 7, 6, 1]$. The roundness is 5.

In the second test case, before the first update, since $x \neq a_i$ for all i, x must be even, implying that all of $a \oplus x$ are odd, so the answer is 0. After the first update, the array becomes $a = [1, 4, 5, 7]$. Here, $x = 3$ provides the best result with $a \oplus 3 = [2, 7, 6, 4]$, with roundness 4.