CPC '19 Contest 1 P6 - Maintaining Some Coins

Time limit: 2.5s **Memory limit:** 32M

Angie is maintaining some coins!

She begins with an array of N coins, and then assigns each coin with a number a_i. Next, she will perform Q operations of one of the following types on the coins.

- `I i x`: Insert a coin with value x directly after coin i. If $i = 0$ then the coin is inserted at the beginning
- `D j`: Delete coin j
- `C l r x`: Output how many coins between coin l and r (inclusive) have a value of x

Because the amount of coins can grow very large, she has decided to enlist your help to keep track of the coins. Can you help her do it?

Important Note: The test cases are *Online Enforced*. That means all numbers in the input will be encrypted with the formula $v \oplus lastAns$, where v is the original number in the query, $\oplus$ is the Bitwise XOR operation, and $lastAns$ is the answer to the last `C` operation given (or 0 if none have been answered yet).

Constraints

For all subtasks:

$1 \le a_i, x \le 10^6$

$0 \le i \le |a|$

$1 \le j \le |a|$

$1 \le l \le r \le |a|$

$|a|$ is the length of the coin array at the time of the query.

Subtask 1 [5%]

$1 \le N, Q \le 100$

Subtask 2 [20%]

All elements in the initial array and all inserted elements are distinct.

$1 \le N, Q \le 3 \times 10^5$

Subtask 3 [75%]

$1 \le N, Q \le 3 \times 10^5$

Input Specification

The first line of input will contain the space separated integers N and Q.

The second line will contain N space separated integers: $a_1, a_2, \ldots, a_N$.

The final Q lines will contain operations in the format described above.

Output Specification

Output the answer for each `C` query on its own line.

Sample Input

```
10 10
5 1 6 1 5 1 2 2 7 8
C 2 5 1
D 0
C 0 6 3
I 1 4
I 11 11
C 0 2 4
C 8 9 8
I 2 7
C 0 7 7
C 3 14 0
```

Sample Input (Not Encrypted)

For your convenience, here is a version of the sample input that is **NOT encrypted**. Remember, all of the real test files will be encrypted (like the input above).

```
10 10
5 1 6 1 5 1 2 2 7 8
C 2 5 1
D 2
C 2 4 1
I 0 5
I 10 10
C 1 3 5
C 10 11 10
I 3 6
C 1 6 6
C 1 12 2
```

Sample Output

```
2
1
2
1
2
2
```

Sample Explanation

This is what the array begins as: `5 1 6 1 5 1 2 2 7 8`

The first query asks for this range: `5 [1 6 1 5] 1 2 2 7 8`

The second query changes the array to this: `5 6 1 5 1 2 2 7 8`

The third query asks for this range: `5 [6 1 5] 1 2 2 7 8`

The fourth query changes the array to this: `5 5 6 1 5 1 2 2 7 8`

The fifth query changes the array to this: `5 5 6 1 5 1 2 2 7 8 10`

The sixth query asks for this range: `[5 5 6] 1 5 1 2 2 7 8 10`

The seventh query asks for this range: `5 5 6 1 5 1 2 2 7 [8 10]`

The eighth query changes the array to this: `5 5 6 6 1 5 1 2 2 7 8 10`

The ninth query asks for this range: `[5 5 6 6 1 5] 1 2 2 7 8 10`

The tenth and final query asks for the entire array.