Segment Tree Test

Time limit: 0.5s
Java: 1.0s
Python: 2.5s

Memory limit: 16M
Python: 256M

Xyene is doing a contest. He comes across the following problem:

You have an array of N $(1 \leq N \leq 100\,000)$ elements, indexed from 1 to N. There are M $(1 \leq M \leq 500\,000)$ operations you need to perform on it.

Each operation is one of the following:

- `C x v` Change the x-th element of the array to v.
- `M l r` Output the minimum of all the elements from the l-th to the r-th index, inclusive.
- `G l r` Output the greatest common divisor of all the elements from the l-th to the r-th index, inclusive.
- `Q l r` Let G be the result of the operation `G l r` right now. Output the number of elements from the l-th to the r-th index, inclusive, that are equal to G.

At any time, every element in the array is between 1 and 10^9 (inclusive).

Xyene knows that one fast solution uses a Segment Tree. He practices that data structure every day, but still somehow manages to get it wrong. Will you show him a working example?

Input Specification

The first line has N and M.

The second line has N integers, the original array.

The next M lines each contain an operation in the format described above.

Output Specification

For each `M`, `G`, or `Q` operation, output the answer on its own line.

Sample Input 1

```
5 5
1 1 4 2 8
C 2 16
M 2 4
G 2 3
C 2 1
Q 1 5
```

Sample Output 1

```
2
4
2
```

Sample Input 2

```
5 2
1 1 2 2 2
Q 1 4
Q 3 5
```

Sample Output 2

```
2
3
```