Binary Indexed Tree Test

Time limit: 1.4s **Memory limit:** 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.
- `S l r` Output the sum of all the elements from the l-th to the r-th index, inclusive.
- `Q v` Output how many elements are less than or equal to v in the array.

At any time, every element in the array is between 1 and $100\,000$ (inclusive).

Xyene knows that one fast solution uses a Binary Indexed 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 `S` or `Q` operation, output the answer on its own line. Note that you may need to use 64-bit integers to store the answer.

Sample Input

```
10 10
4 8 4 5 6 3 2 2 8 1
C 7 6
Q 7
S 2 3
S 1 4
C 4 9
S 2 3
Q 6
C 3 9
S 6 7
Q 6
```

Sample Output

```
8
12
21
12
7
9
6
```