Balanced Problem

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

There is an array a consisting of n integers. Initially, all elements of a are equal to 0.

Kevin can perform several operations on the array. Each operation is one of the following two types:

- Prefix addition — Kevin first selects an index x ($1 \le x \le n$), and then for each $1 \le j \le x$, increases a_j by 1;
- Suffix addition — Kevin first selects an index x ($1 \le x \le n$), and then for each $x \le j \le n$, increases a_j by 1.

In the country of KDOI, people think that the integer v is balanced. Thus, Iris gives Kevin an array c consisting of n integers and defines the *beauty* of the array a as follows:

- Initially, set $b = 0$;
- For each $1 \le i \le n$, if $a_i = v$, add c_i to b;
- The beauty of a is the final value of b.

Kevin wants to maximize the beauty of a after all the operations. However, he had already performed m operations when he was sleepy. Now, he can perform an arbitrary number (possibly zero) of new operations.

You have to help Kevin find the maximum possible beauty if he optimally performs the new operations.

However, to make sure that you are not just rolling the dice, Kevin gives you an integer V, and you need to solve the problem for each $1 \le v \le V$.

Input

Each test contains multiple test cases. The first line of the input contains a single integer t ($1 \le t \le 1000$) — the number of test cases. The description of test cases follows.

The first line of each test case contains three integers n, m, and V ($1 \le n, m \le 2 \cdot 10^5$, $1 \le V \le 2000$) — the length of the array a, the number of initial operations, and the number that Kevin gives you.

The second line contains n integers $c_1, c_2, \ldots, c_n$ ($1 \le c_i \le 10^9$) — the elements in the array c.

Then m lines follow, the i-th line containing a character op and an integer x ($op =$ `L` or `R`, $1 \le x \le n$) — the type of the i-th operation and the selected index.

- If $op =$ `L`, this operation is a prefix addition on index x;
- If $op =$ `R`, this operation is a suffix addition on index x.

It is guaranteed that:

- the sum of n over all test cases does not exceed $2 \cdot 10^5$;
- the sum of m over all test cases does not exceed $2 \cdot 10^5$;
- the sum of V^2 over all test cases does not exceed $4 \cdot 10^6$.

Output

For each test case, output V integers in a single line, the i-th integer denoting the maximum possible beauty after Kevin performs some new operations when $v = i$.

Example

standard input	standard output
5 3 3 2 1 2 4 L 3 R 3 L 1 3 3 2 5 1 4 L 3 R 3 L 1 5 4 5 1 1 1 1 1 L 3 R 2 L 5 L 4 10 12 9 10 9 8 7 6 5 4 3 2 1 L 2 L 4 R 4 R 4 L 6 R 8 L 3 L 2 R 1 R 10 L 8 L 1 1 1 4 1000000000 L 1	2 6 1 9 0 1 3 5 5 0 0 0 6 25 32 35 44 51 1000000000 1000000000 1000000000 1000000000

Note

In the first test case, the array a changes as follows for the initial operations: $[0,0,0] \xrightarrow{\text{L 3}} [1,1,1] \xrightarrow{\text{R 3}} [1,1,2] \xrightarrow{\text{L 1}} [2,1,2]$.

- For $v = 1$, it is optimal to not perform any new operations, and the beauty is $b = c_2 = 2$;
- For $v = 2$, it is optimal to perform a prefix addition operation on index 2. After that, a becomes $[3,2,2]$, and the beauty is $b = c_2 + c_3 = 6$.

In the second test case, for both $v = 1$ and $v = 2$, it is optimal to not perform any new operations.