It's normal to feel worried and tense the day before a programming contest. To relax, you went out for a drink with some friends in a nearby pub. To keep your mind sharp for the next day, you decided to play the following game. To start, your friends will give you a sequence of N integers $X_1, X_2, \ldots, XN$. Then, there will be K rounds; at each round, your friends will issue a command, which can be:

- a *change* command, when your friends want to change one of the values in the sequence; or
- a *product* command, when your friends give you two values I, J and ask you if the product $X_I \times X_{I+1} \times \ldots \times X_{J-1} \times X_J$ is positive, negative or zero.

Since you are at a pub, it was decided that the penalty for a wrong answer is to drink a pint of beer. You are worried this could affect you negatively at the next day's contest, and you don't want to check if Ballmer's peak theory is correct. Fortunately, your friends gave you the right to use your notebook. Since you trust more your coding skills than your math, you decided to write a program to help you in the game.

Input

Each test case is described using several lines. The first line contains two integers N and K, indicating respectively the number of elements in the sequence and the number of rounds of the game ($1 \leq N, K \leq 10^5$). The second line contains N integers X_i that represent the initial values of the sequence ($-100 \leq X_i \leq 100$ for $i = 1, 2, \ldots, N$). Each of the next K lines describes a command and starts with an uppercase letter that is either '`C`' or '`P`'. If the letter is '`C`', the line describes a *change* command, and the letter is followed by two integers I and V indicating that X_I must receive the value V ($1 \leq I \leq N$ and $-100 \leq V \leq 100$). If the letter is '`P`', the line describes a *product* command, and the letter is followed by two integers I and J indicating that the product from X_I to X_J, inclusive must be calculated ($1 \leq I \leq J \leq N$). Within each test case there is at least one *product* command.

Output

For each test case output a line with a string representing the result of all the *product* commands in the test case. The i-th character of the string represents the result of the i-th *product* command. If the result of the command is positive the character must be '`+`' (plus); if the result is negative the character must be '`-`' (minus); if the result is zero the character must be '`0`' (zero).

Sample Input

```
4 6
-2 6 0 -1
C 1 10
P 1 4
C 3 7
P 2 2
C 4 -5
P 1 4
5 9
1 5 -2 4 3
P 1 2
P 1 5
C 4 -5
P 1 5
P 4 5
C 3 0
P 1 5
C 4 -5
C 4 -5
```

Sample Output

```
0+-
+-+-0
```