Jupiter is invading! Major cities have been destroyed by Jovian spacecrafts and humanity is fighting back. Nlogonia is spearheading the counter-offensive, by hacking into the spacecrafts' control system.

Unlike Earthling computers, in which usually a byte has 2^8 possible values, Jovian computers use bytes with B possible values, $\{0, 1, \ldots, B-1\}$. Nlogonian software engineers have reverse-engineered the firmware for the Jovian spacecrafts, and plan to sabotage it so that the ships eventually selfdestruct.

As a security measure, however, the Jovian spacecrafts run a supervisory program that periodically checks the integrity of the firmware, by hashing portions of it and comparing the result against known good values. To hash the portion of the firmware from the byte at position i to the byte at position j, the supervisor uses the hash function

$$H(f_i, \ldots, f_j) = \sum_{k=0}^{j-i} B^k f_{j-k} \pmod{P}$$

where P is a prime number. For instance, if $B = 20$ and $P = 139$, while bytes 2 to 5 of the firmware have the values $f_2 = 14$, $f_3 = 2$, $f_4 = 2$, and $f_5 = 4$, then

$$\begin{aligned} H(f_2, \ldots, f_5) &= B^0 f_5 + B^1 f_4 + B^2 f_3 + B^3 f_2 \pmod{P} \\ &= 20^0 \times 4 + 20^1 \times 2 + 20^2 \times 2 + 20^3 \times 14 \pmod{139} \\ &= 4 + 40 + 800 + 112000 \pmod{139} \\ &= 112844 \pmod{139} \\ &= 115 \end{aligned}$$

The Nlogonian cryptologists need to find a way to sabotage the firmware without tripping the supervisor. As a first step, you have been assigned to write a program to simulate the interleaving of two types of commands: editing bytes of the firmware by the Nlogonian software engineers, and computing hashes of portions of the firmware by the Jovian supervisory program. At the beginning of the simulation the value of every byte in the firmware is zero.

Input

Each test case is described using several lines. The first line contains four integers B, P, L and N, where B is the number of possible values of a Jovian byte, P is the modulus of the Jovian hash ($2 \le B < P \le 10^9$ and P prime), L is the length (number of Jovian bytes) of the spacecrafts' firmware, and N is the number of commands to simulate ($1 \le L, N \le 10^5$). At the beginning of the simulation the value of every byte in the firmware is $f_i = 0$ for $1 \le i \le L$. Each of the next N lines describes a command to simulate. Each command description starts with an uppercase letter that is either 'E' or 'H', with the following meanings.

'E' → The line describes an edit command. The letter is followed by two integers I and V indicating that the byte at position I of the firmware (that is, f_I) must receive the value V ($1 \le I \le L$ and $0 \le V \le B-1$).

'H' → The line describes a hash command. The letter is followed by two integers I and J indicating that $H(f_I, \ldots, f_J)$ must be computed ($1 \le I \le J \le L$).

The last test case is followed by a line containing four zeros.

Output

For each test case output the results of the hash commands in the input. In the i-th line write an integer representing the result of the i-th hash command. Print a line containing a single character '-' (hyphen) after each test case.

Sample Input

```
20 139 5 7
E 1 12
E 2 14
E 3 2
E 4 2
E 5 4
H 2 5
E 2 14
10 1000003 6 11
E 1 3
E 2 4
E 3 5
E 4 6
E 5 7
E 6 8
H 1 6
E 3 0
E 3 9
H 1 3
H 4 6
999999935 999999937 100000 7
E 100000 6
E 1 7
H 1 100000
E 50000 8
H 1 100000
H 25000 75000
H 23987 23987
0 0 0 0
```

Sample Output

```
115
-
345678
349
678
-
824973478
236724326
450867806
0
-
```