Júpiter Ataca!

Por Fidel I. Schaposnik 🇦🇷 Argentina

Timelimit: 3

Júpiter está invadindo! As principais cidades tem sido destruídas por espaçonaves Jovianas e a humanidade está lutando contra. Nlogônia está à frente da contraofensiva, invadindo os sistemas de controle das espaçonaves.

Diferente dos computadores Terráqueos, nos quais usalmente um byte possui 2^8 valores possíveis, os computadores Jovianos usam bytes com B possíveis valores, {0,1,...,B-1}. Os engenheiros de software Nlogonianos tem realizado engenharia reversa sobre o firmware das espaçonaves Jovianas, e planejam sabotá-lo de modo que as embarcações eventualmete autodestruam-se.

Como uma medida de segurança, entretanto, as espaçonaves Jovianas rodam um programa supervisor que periodicamente checa a integridade do firmware, aplicando hashing sobre porções dele e comparando o resultado contra valores bons conhecidos. Para aplicar o hashing sobre uma porção do firmware do byte na posição i até o byte na posição j, o supervisor usa a função de hashing

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

onde P é um número primo. Por exemplo, se B = 20 e P = 139, enquanto os bytes 2 ao 5 do firmware tem os valores $f_2 = 14$, $f_3 = 2$, $f_4 = 2$ e $f_5 = 4$ então

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

Os criptologistas Nlogonianos precisam encontrar um meio de sabotar o firmware sem esbarrar no supervisor. Como um primeiro passo, a você foi atribuída a função de escrever um programa para simular a intercalagem de dois tipos de comandos: edição de bytes do firmware pelos engenheiros de software Nlogonianos, e o cálculo de hashes de porções do firmware pelo program supervisor Joviano. No início da simulação o valor de cada byte é zero.

Entrada

Cada caso de teste é descrito usando várias linhas. A primeira linha contém quatro inteiros **B, P, L** e **N**, onde **B** é o número de possíveis valores de um byte Joviano, **P** é o módulo da hash Joviana ($2 \le B < P \le 10^9$e **P** primo), **L** é o comprimento (número de bytes Jovianos) do firmware das espaçonaves, e **N** é o número de comandos a simular ($1 \le L, N \le 10^5$). No início da simulação o valor de cada byte no firmware é $f_i = 0$ para $1 \le i \le L$. Cada uma das **N** linhas seguintes descreve um comando a simular. Cada descrição de comando começa com uma letra maiúscula que é ou um 'E' ou um 'H', com os seguintes significados.

- 'E': A linha descreve um comando de edição. A letra é seguida por dois inteiros **I** e **V** indicando que o byte na posição **I** do firmware (ou seja, f_i) deve receber o valor **V** ($1 \le I \le L$ e $0 \le V \le B-1$).
- 'H': A linha descreve um comando de hash. A letra é seguida por dois inteiro I e J indicando que

$H(f_i...f_j)$ deve ser computado ($1 \le I \le J \le L$).

O último caso de teste é seguido por uma linha contendo quatro zeros.

Saída

Para cada caso de teste imprima os resultados de cada comando de hashing na entrada. Na i-ésima linha escreva um inteiro representando o resultado do i-ésimo comando de hashing. Imprima uma linha contendo um único caractere '-' (hífen) após cada caso de teste.

Exemplo de Entrada	Exemplo de Saída
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	115 - 345678 349 678 - 824973478 236724326 450867806 0 -

ACM/ICPC South America Contest 2011.