Distributing Candies

A tia Khong está a preparar n caixas de doces para os estudantes de uma escola próxima. As caixas são numeradas de 0 a $n-1$ e estão inicialmente vazias. A caixa i ($0 \leq i \leq n-1$) tem uma capacidade de $c[i]$ doces.

A tia Khong gasta q dias para preparar as caixas. No dia j ($0 \leq j \leq q-1$), ela faz uma ação especificada por três inteiros $l[j]$, $r[j]$ e $v[j]$ onde $0 \leq l[j] \leq r[j] \leq n-1$ e $v[j] \neq 0$. Para cada caixa k satisfazendo $l[j] \leq k \leq r[j]$:

- Se $v[j] > 0$, a tia Khong adiciona doces à caixa k, um por um, até ela ter adicionado exatamente $v[j]$ doces ou a caixa ficar cheia. Por outras palavras, se a caixa tiver p doces antes de ação, passará a ter $\min(c[k], p + v[j])$ depois da ação.
- Se $v[j] < 0$, a tia Khong remove doces da caixa k, um por um, até ela ter removido exatamente $-v[j]$ doces ou a caixa ficar vazia. Por outras palavras, se a caixa tiver p doces antes de ação, passará a ter $\max(0, p + v[j])$ depois da ação.

A tua tarefa é determinar o número de doces de cada caixa depois dos q dias.

Detalhes de Implementação

Deves implementar a seguinte função:

```
int[] distribute_candies(int[] c, int[] l, int[] r, int[] v)
```

- c: um array de tamanho n. Para $0 \leq i \leq n-1$, $c[i]$ indica a capacidade da caixa i.
- l, r and v: três arrays de tamanho q. No dia j, para $0 \leq j \leq q-1$, a tia Khong faz uma ação especificada pelos inteiros $l[j]$, $r[j]$ e $v[j]$, como atrás descrito.
- Esta função deve devolver um array de tamanho n. Seja este array indicado por s. Para $0 \leq i \leq n-1$, $s[i]$ deve ser o número de doces na caixa i depois de q dias.

Exemplos

Exemplo 1

Considera a seguinte chamada:

```
distribute_candies([10, 15, 13], [0, 0], [2, 1], [20, -11])
```

Isto significa que a caixa 0 tem capacidade para 10 doces, a caixa 1 tem capacidade para 15 doces e a caixa 2 tem capacidade para 13 doces.

No final do dia 0, a caixa 0 tem $\min(c[0], 0 + v[0]) = 10$ doces, a caixa 1 tem $\min(c[1], 0 + v[0]) = 15$ doces e a caixa 2 tem $\min(c[2], 0 + v[0]) = 13$ doces.

No final do 1, a caixa 0 tem $\max(0, 10 + v[1]) = 0$ doces e a caixa 1 tem $\max(0, 15 + v[1]) = 4$ doces. Como $2 > r[1]$, não existe alteração no número de doces da caixa 2. O número de doces no final de cada diz é resumido na seguinte tabela:

Dia	Caixa 0	Caixa 1	Caixa 2
0	10	15	13
1	0	4	13

Como tal, a função deve devolver $[0, 4, 13]$.

Restrições

- $1 \leq n \leq 200\,000$
- $1 \leq q \leq 200\,000$
- $1 \leq c[i] \leq 10^9$ (para $0 \leq i \leq n - 1$)
- $0 \leq l[j] \leq r[j] \leq n - 1$ (para $0 \leq j \leq q - 1$)
- $-10^9 \leq v[j] \leq 10^9, v[j] \neq 0$ (para $0 \leq j \leq q - 1$)

Subtarefas

1. (3 pontos) $n, q \leq 2000$
2. (8 pontos) $v[j] > 0$ (para $0 \leq j \leq q - 1$)
3. (27 pontos) $c[0] = c[1] = \ldots = c[n - 1]$
4. (29 pontos) $l[j] = 0$ e $r[j] = n - 1$ (para $0 \leq j \leq q - 1$)
5. (33 pontos) Sem restrições adicionais.

Avaliador Exemplo

O avaliador exemplo lê o input no seguinte formato:

- linha 1: n
- linha 2: $c[0]$ $c[1]$ $\ldots$ $c[n - 1]$
- linha 3: q
- linha $4 + j$ ($0 \leq j \leq q - 1$): $l[j]$ $r[j]$ $v[j]$

O avaliador exemplo escreve as respostas no seguinte formato:

- linha 1: $s[0]$ $s[1]$ $\ldots$ $s[n - 1]$