Distributing Candies

Tante Khong vult n dozen snoep voor leerlingen van een lokale school. De dozen zijn genummerd van 0 tot en met $n-1$ en zijn aanvankelijk leeg. Doos i ($0 \leq i \leq n-1$) heeft een capaciteit van $c[i]$ snoepjes.

Tante Khong is q dagen bezig met het vullen van de dozen. Op dag j ($0 \leq j \leq q-1$), voert ze een actie uit, aangegeven door drie gehele getallen $l[j]$, $r[j]$ en $v[j]$ met $0 \leq l[j] \leq r[j] \leq n-1$ en $v[j] \neq 0$. Voor iedere doos k waar $l[j] \leq k \leq r[j]$:

- Als $v[j] > 0$, voegt tante Khong snoepjes toe aan doos k, één voor één, tot ze precies $v[j]$ snoepjes heeft toegevoegd of de doos vol is. In andere woorden, als de doos p snoepjes had voor de actie, dan heeft die $\min(c[k], p + v[j])$ snoepjes na de actie.
- Als $v[j] < 0$, verwijdert tante Khong snoepjes uit doos k, één voor één, tot ze precies $-v[j]$ snoepjes heeft verwijderd of de doos leeg is. Met andere woorden: als de doos p snoepjes had voor de actie, dan heeft die $\max(0, p + v[j])$ snoepjes na de actie.

Het is jouw taak om vast te stellen hoeveel snoepjes er in iedere doos zitten na q dagen.

Implementatiedetails

Je moet de volgende functie implementeren:

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

- c: een array van lengte n. Voor elke $0 \leq i \leq n-1$, geeft $c[i]$ de capaciteit van doos i aan.
- l, r en v: drie arrays van lengte q. Op dag j, voert tante Khong voor $0 \leq j \leq q-1$ een actie uit, aangegeven door integers $l[j]$, $r[j]$ en $v[j]$, zoals hierboven omschreven.
- Deze functie moet een array van lengte n teruggeven. Noem de array s. Voor $0 \leq i \leq n-1$, moet $s[i]$ het aantal snoepjes in doos i na q dagen zijn.

Voorbeelden

Voorbeeld 1

Neem de volgende aanroep:

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

Dit betekent: doos 0 heeft een capaciteit van 10 snoepjes; doos 1 heeft een capaciteit van 15 snoepjes; en doos 2 heeft een capaciteit van 13 snoepjes.

Aan het einde van dag 0 bevat doos 0 $\min(c[0], 0 + v[0]) = 10$ snoepjes; doos 1 bevat $\min(c[1], 0 + v[0]) = 15$ snoepjes en doos 2 bevat $\min(c[2], 0 + v[0]) = 13$ snoepjes.

Aan het einde van dag 1 bevat doos 0 $\max(0, 10 + v[1]) = 0$ snoepjes; doos 1 bevat $\max(0, 15 + v[1]) = 4$ snoepjes. Omdat $2 > r[1]$, is er geen verandering in het aantal snoepjes in doos 2. Het aantal snoepjes aan het einde van iedere dag is hier onder samengevat:

Dag	Doos 0	Doos 1	Doos 2
0	10	15	13
1	0	4	13

De functie moet dus $[0, 4, 13]$ teruggeven.

Randvoorwaarden

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

Subtaken

1. (3 punten) $n, q \leq 2000$
2. (8 punten) $v[j] > 0$ (voor iedere $0 \leq j \leq q - 1$)
3. (27 punten) $c[0] = c[1] = \ldots = c[n - 1]$
4. (29 punten) $l[j] = 0$ and $r[j] = n - 1$ (voor iedere $0 \leq j \leq q - 1$)
5. (33 punten) Geen aanvullende randvoorwaarden.

Voorbeeldgrader

De voorbeeldgrader leest de invoer in het volgende formaat:

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

De voorbeeldgrader schrijft naar de uitvoer in het volgende formaat:

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