Distributions de bonbons (Candies)

La tante Khong prépare n boîtes de bonbons pour les élèves de l'école voisine. Les boîtes sont numérotées de 0 à $n-1$ et sont initialement vides. La boîte i ($0 \leq i \leq n-1$) a une capacité maximale de $c[i]$ bonbons.

La tante Khong passe q jours à préparer les boîtes. On peut résumer son travail du jour j ($0 \leq j \leq q-1$) par trois entiers : $l[j]$, $r[j]$ et $v[j]$ avec $0 \leq l[j] \leq r[j] \leq n-1$ et $v[j] \neq 0$. Pour chaque boîte k satisfaisant $l[j] \leq k \leq r[j]$, on a :

- Si $v[j] > 0$ alors la tante Khong ajoute des bonbons à la boîte k, un par un, jusqu'à ce qu'elle ajoute exactement $v[j]$ bonbons ou que la boîte soit remplie. Autrement dit, si la boîte avait p bonbons au début de la journée, elle aura $\min(c[k], p+v[j])$ bonbons à la fin de la journée.
- Si $v[j] < 0$, alors la tante Khong va retirer des bonbons de la boîte k, un par un, jusqu'à ce quelle retire exactement $-v[j]$ bonbons ou que la boîte soit vide. Autrement dit, si la boîte avait p bonbons au début de la journée, elle aura $\max(0, p+v[j])$ bonbons en fin de journée.

Votre tâche est de déterminer le nombre de bonbons dans chaque boîte à la fin des q jours.

Détails d'implémentation

Vous devez implémenter la fonction suivante :

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

- c : un tableau de longueur n. Pour $0 \leq i \leq n-1$, $c[i]$ représente la capacité maximale de la boîte i.
- l, r et v: trois tableaux de taille q. Pour $0 \leq j \leq q-1$, les trois entiers $l[j]$, $r[j]$ et $v[j]$ résument le travail de la tante Khong au jour j, comme décrit plus haut.
- Cette fonction doit renvoyer un tableau de taille n. Si l'on nomme ce tableau s, alors pour $0 \leq i \leq n-1$, $s[i]$ doit correspondre au nombre de bonbons dans la boîte i à la fin des q jours.

Exemples

Exemple 1

Supposons que la fonction est appelée ainsi :

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

Cela signifie que la boîte 0 a une capacité maximale de 10 bonbons, la boîte 1 une capacité de 15 bonbons et la boîte 2 une capacité de 13 bonbons.

À la fin du jour 0, la boîte 0 a $\min(c[0], 0 + v[0]) = 10$ bonbons, la boîte 1 a $\min(c[1], 0 + v[0]) = 15$ bonbons et la boîte 2 a $\min(c[2], 0 + v[0]) = 13$ bonbons.

À la fin du jour 1, la boîte 0 a $\max(0, 10 + v[1]) = 0$ bonbons, la boîte 1 a $\max(0, 15 + v[1]) = 4$ bonbons. Comme $2 > r[1]$, le nombre de bonbons dans la boîte 2 ne change pas. Le nombre de bonbons à la fin de chaque jour est donné par le tableau suivant :

Jour	**Boîte** 0	**Boîte** 1	**Boîte** 2
0	10	15	13
1	0	4	13

La fonction doit donc renvoyer $[0, 4, 13]$.

Contraintes

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

Sous-tâches

1. (3 points) $n, q \leq 2000$
2. (8 points) $v[j] > 0$ (pour chaque $0 \leq j \leq q - 1$)
3. (27 points) $c[0] = c[1] = \ldots = c[n-1]$
4. (29 points) $l[j] = 0$ et $r[j] = n - 1$ (pour chaque $0 \leq j \leq q - 1$)
5. (33 points) Pas de contraintes additionnelles.

Évaluateur d'exemple

L'évaluateur d'exemple lit l'entrée au format suivant :

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

L'évaluateur d'exemple affiche les réponses au format suivant :

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