Distributing Candies

Леля Конг е приготвила n кутии с бонбони за учениците от близкото училище. Кутиите са номерирани с числата от 0 до $n-1$ и в началото са празни. Кутия с номер i ($0 \leq i \leq n-1$) има капацитет от $c[i]$ бонбони.

Леля Конг прекарва q дни за подготвяне на кутиите. За ден j ($0 \leq j \leq q-1$), тя извършва действие, характеризирано от три цели числа $l[j]$, $r[j]$ и $v[j]$, където $0 \leq l[j] \leq r[j] \leq n-1$ и $v[j] \neq 0$. За всяка кутия с номер k, удовлетворяващ $l[j] \leq k \leq r[j]$:

- Ако $v[j] > 0$, то леля Конг добавя бонбони към кутията с номер k, един по един, докато тя не е добавила точно $v[j]$ бонбона или кутията е станала пълна. Казано по друг начин, ако кутията има p бонбона преди това действие, то тя ще има $\min(c[k], p + v[j])$ бонбона след действието.
- Ако $v[j] < 0$, то леля Конг маха бонбони от кутията с номер k, един по един, докато тя не махне точно $-v[j]$ бонбона или кутията не остане празна. Казано по друг начин, ако кутията има p бонбона преди това действие, то тя ще има $\max(0, p + v[j])$ бонбона след действието.

Вашата задача е да определите броя бонбони във всяка кутия след q-те дни.

Детайли по реализацията

Трябва да напишете следната функция:

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

- c: масив с дължина n. За всяко $0 \leq i \leq n-1$, $c[i]$ представлява капацитета на кутията с номер i.
- l, r и v: три масива с дължина q. За ден j, където $0 \leq j \leq q-1$, леля Конг извършва действие, характеризирано с целите числа $l[j]$, $r[j]$ и $v[j]$, както е описано по-горе.
- Тази функция трябва да върне масив с дължина n. Нека го означим с s. За всяко $0 \leq i \leq n-1$, $s[i]$ трябва да е броят бонбони в кутия i след q-те дни.

Пример

Нека имаме следното извикване:

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

Това означава, че кутия 0 има капацитет от 10 бонбона, кутия 1 има капацитет от 15 бонбона, а кутия 2 има капацитет от 13 бонбона.

В края на ден 0, кутия 0 има $\min(c[0], 0 + v[0]) = 10$ бонбона, кутия 1 има $\min(c[1], 0 + v[0]) = 15$ бонбона, а кутия 2 има $\min(c[2], 0 + v[0]) = 13$ бонбона.

В края на ден 1, кутия 0 има $\max(0, 10 + v[1]) = 0$ бонбона, кутия 1 има $\max(0, 15 + v[1]) = 4$ бонбона. Понеже $2 > r[1]$, няма промяна в броя бонбони в кутия 2. Броят бонбони в края на всеки ден е изобразен в следната таблица:

Ден	Кутия 0	Кутия 1	Кутия 2
0	10	15	13
1	0	4	13

Затова, функцията трябва да върне $[0, 4, 13]$.

Ограничения

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

Подзадачи

1. (3 точки) $n, q \leq 2000$
2. (8 точки) $v[j] > 0$ (за всяко $0 \leq j \leq q - 1$)
3. (27 точки) $c[0] = c[1] = \ldots = c[n - 1]$
4. (29 точки) $l[j] = 0$ и $r[j] = n - 1$ (за всяко $0 \leq j \leq q - 1$)
5. (33 точки) няма допълнителни ограничения.

Примерен грейдър

Примерният грейдър чете от стандартния вход в следния формат:

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

Примерният грейдър отпечатва вашите отговори в следния формат:

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