แจกลูกกวาด

ป้าคงกำลังจัดเตรียมกล่องลูกกวาด n กล่องสำหรับนักเรียนจากโรงเรียนข้างเคียง แต่ละกล่องกำกับด้วยหมายเลข 0 ถึง $n-1$ และเป็นกล่องที่ว่างเปล่าในตอนเริ่มต้น โดยกล่อง i ($0 \leq i \leq n-1$) มีความจุสำหรับลูกกวาดจำนวน $c[i]$ เม็ด

ป้าคงใช้เวลา q วันสำหรับการจัดเตรียมกล่องลูกกวาดทั้งหมด สำหรับวันที่ j ($0 \leq j \leq q-1$) เธอดำเนินการตามที่ระบุด้วยจำนวนเต็มสามจำนวน $l[j]$, $r[j]$ และ $v[j]$ โดยที่ $0 \leq l[j] \leq r[j] \leq n-1$ และ $v[j] \neq 0$ สำหรับแต่ละกล่อง k ใด ๆ ที่ตรงกับเงื่อนไข $l[j] \leq k \leq r[j]$ นั้น

- ถ้า $v[j] > 0$ ป้าคงนำลูกกวาดใส่ลงในกล่อง k ทีละเม็ดจนกว่าเธอจะเพิ่มลูกกวาดครบเป็นจำนวน $v[j]$ เม็ดพอดี หรือจนกว่ากล่องนั้นจะเต็มความจุ กล่าวคือถ้ากล่องนั้นมีลูกกวาดจำนวน p เม็ดก่อนการดำเนินการ กล่องนั้นจะมีลูกกวาดจำนวน $\min(c[k], p+v[j])$ เม็ดหลังการดำเนินการ
- ถ้า $v[j] < 0$ ป้าคงนำลูกกวาดออกจากกล่อง k ทีละเม็ดจนกว่าเธอจะนำลูกกวาดออกครบเป็นจำนวน $-v[j]$ เม็ดพอดี หรือจนกว่ากล่องนั้นจะว่างเปล่า กล่าวคือถ้ากล่องนั้นมีลูกกวาดจำนวน p เม็ดก่อนการดำเนินการ กล่องนั้นจะมีลูกกวาดจำนวน $\max(0, p+v[j])$ เม็ดหลังการดำเนินการ

คุณได้รับมอบหมายให้หาจำนวนลูกกวาดในแต่ละกล่องหลังจาก q วันดังกล่าว

รายละเอียดการเขียนโปรแกรม

คุณจะต้องเขียนฟังก์ชันต่อไปนี้:

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

- c: อาร์เรย์ความยาว n โดยที่ $c[i]$ ระบุความจุของกล่องที่ i เมื่อ $0 \leq i \leq n-1$
- l, r และ v: อาร์เรย์ความยาว q จำนวนสามอาร์เรย์ ในวันที่ j สำหรับ $0 \leq j \leq q-1$ ป้าคงดำเนินการตามที่ระบุด้วยจำนวนเต็ม $l[j]$, $r[j]$ และ $v[j]$ ตามคำอธิบายข้างต้น
- ฟังก์ชันนี้จะต้องคืนค่าอาร์เรย์ความยาว n โดยเรียกอาร์เรย์นี้ว่า s สำหรับ $0 \leq i \leq n-1$, $s[i]$ ควรระบุจำนวนลูกกวาดในกล่องที่ i หลังจาก q วันดังกล่าว

ตัวอย่าง

ตัวอย่างที่ 1

พิจารณาการเรียกต่อไปนี้:

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

.

ซึ่งหมายความว่ากล่อง 0 มีความจุลูกกวาดจำนวน 10 เม็ด กล่อง 1 มีความจุลูกกวาดจำนวน 15 เม็ด และกล่อง 2 มีความจุลูกกวาดจำนวน 13 เม็ด

เมื่อจบวันที่ 0 กล่อง 0 มีลูกกวาดจำนวน $\min(c[0], 0 + v[0]) = 10$ เม็ด กล่อง 1 มีลูกกวาดจำนวน $\min(c[1], 0 + v[0]) = 15$ เม็ด และกล่อง 2 มีลูกกวาดจำนวน $\min(c[2], 0 + v[0]) = 13$ เม็ด

เมื่อจบวันที่ 1 กล่อง 0 มีลูกกวาดจำนวน $\max(0, 10 + v[1]) = 0$ เม็ด กล่อง 1 มีลูกกวาดจำนวน $\max(0, 15 + v[1]) = 4$ เม็ด เนื่องจาก $2 > r[1]$ จะไม่มีการเปลี่ยนแปลงจำนวนลูกกวาดในกล่อง 2 จำนวนลูกกวาดเมื่อจบแต่ละวันเป็นดังนี้

วัน	กล่อง 0	กล่อง 1	กล่อง 2
0	10	15	13
1	0	4	13

ดังนั้น ฟังก์ชันควรคืนค่า $[0, 4, 13]$

ข้อจำกัด

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

ปัญหาย่อย

1. (3 คะแนน) $n, q \le 2000$
2. (8 คะแนน) $v[j] > 0$ (สำหรับ $0 \le j \le q - 1$)
3. (27 คะแนน) $c[0] = c[1] = \ldots = c[n - 1]$
4. (29 คะแนน) $l[j] = 0$ และ $r[j] = n - 1$ (สำหรับ $0 \le j \le q - 1$)
5. (33 คะแนน) ไม่มีเงื่อนไขใด ๆ เพิ่มเติม

เกรดเดอร์ตัวอย่าง

เกรดเดอร์ตัวอย่างจะอ่านข้อมูลนำเข้าในรูปแบบต่อไปนี้:

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

เกรดเดอร์ตัวอย่างเขียนข้อมูลส่งออกในรูปแบบต่อไปนี้:

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