Radijo bokštai

Džakartoje yra N radijo bokštų. Bokštai yra vienoje linijoje sunumeruoti nuo 0 iki $N-1$ iš kairės į dešinę. i-ojo ($0 \leq i \leq N-1$) bokšto aukštis yra $H[i]$ metrų. Visų bokštų aukščiai yra **skirtingi**.

Duota teigiama trikdžių vertė δ. i-asis ir j-asis bokštai (kur $0 \leq i < j \leq N-1$) gali komunikuoti vienas su kitu tada ir tik tada, jei yra tarpinis bokštas k, toks, kad

- i-asis bokštas yra į kairę nuo k-ojo bokšto, o j-asis yra į dešinę nuo k-ojo bokšto, t.y., $i < k < j$, ir
- i-ojo ir j-ojo bokštų aukščiai yra ne didesni nei $H[k] - \delta$ metrų.

Pak Dengklek nori išsinuomoti radijo bokštus savo naujam radijo tinklui. Atsakykite į Q Pak Dengklek užklausų, kurios pateikiamos tokiu formatu: duoti parametrai L, R ir D ($0 \leq L \leq R \leq N-1$ ir $D > 0$) ir reikia rasti, kiek daugiausia bokštų Pak Dengklek gali išsinuomoti, jei:

- Pak Dengklek gali išsinuomoti tik tuos bokštus, kurių numeriai yra nuo L iki R (imtinai),
- trikdžių δ vertė lygi D, ir
- bet kuri Pak Dengklek išsinuomota radijo bokštų pora turi galėti komunikuoti tarpusavyje

Atkreipkite dėmesį, kad du išsinuomoti bokštai gali komunikuoti per tarpinį bokštą k nepriklausomai nuo to, ar k išnuomotas, ar ne.

Realizacija

Parašykite šias funkcijas:

```
void init(int N, int[] H)
```

- N: radijo bokštų skaičius.
- H: N ilgio masyvas, nusakantis bokštų aukščius.
- Ši funkcija iškviečiama lygiai vieną kartą prieš iškviečiant `max_towers`.

```
int max_towers(int L, int R, int D)
```

- L, R: galimų išsinuoti bokštų intervalas.
- D: δ vertė.

- Ši funkcija turi grąžinti didžiausią galimą radijo bokštų, kuriuos Pak Dengklek gali išsinuomoti skaičių, jei jis gali nuomotis nuo L-ojo iki R-ojo bokšto (imtinai), o δ vertė yra D.
- Ši funkcija iškviečiama lygiai Q kartų.

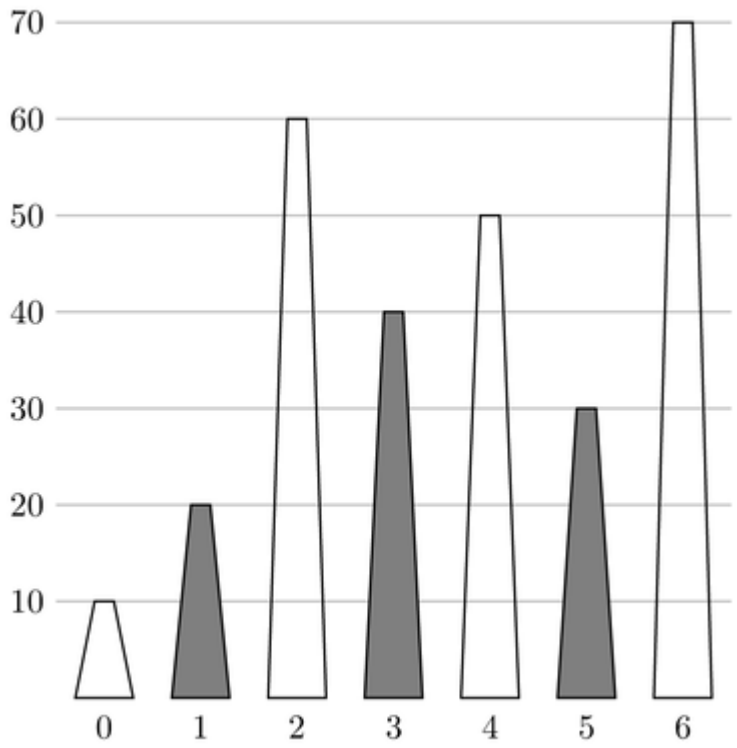
Pavyzdys

Panagrinėkime tokią užklausų seką:

```
init(7, [10, 20, 60, 40, 50, 30, 70])
```

```
max_towers(1, 5, 10)
```

Pak Dengklek gali išsinuomoti 1-ąjį, 3-iąjį ir 5-ąjį bokštus. Tai parodyta paveikslėlyje kur nuspalvintos trapecijos žymi išnuomotus bokštus.



3-iasis ir 5-asis bokštai komunikuoja per tarpinį 4-ąjį bokštą, nes $40 \leq 50 - 10$ ir $30 \leq 50 - 10$. 1-asis ir 3-iasis bokštai komunikuoja per tarpinį 2-ąjį bokštą. 1-asis ir 5-asis bokštai komunikuoja per tarpinį 3-iąjį bokštą. Nėra būdo išsinuomoti daugiau nei 3 bokštus, tad funkcija turi grąžinti 3.

```
max_towers(2, 2, 100)
```

Pak Dengklek gali išsinuomoti tik 1 bokštą, nes intervale yra tik 1 bokštas. Taigi, funkcija turi grąžinti 1.

```
max_towers(0, 6, 17)
```

Pak Dengklek gali išsinuomoti 1-ąjį ir 3-iąjį bokštus. 1-asis ir 3-iasis bokštai gali komunikuoti per tarpinį bokštą 2, nes $20 \leq 60 - 17$ ir $40 \leq 60 - 17$. Nėra galimybės išsinuomoti daugiau nei 2 bokštus, todėl funkcija turi grąžinti 2.

Ribojimai

- $1 \leq N \leq 100\ 000$
- $1 \leq Q \leq 100\ 000$
- $1 \leq H[i] \leq 10^9$ (kiekvienam i kur $0 \leq i \leq N - 1$)
- $H[i] \neq H[j]$ (kiekvienam i ir j kur $0 \leq i < j \leq N - 1$)
- $0 \leq L \leq R \leq N - 1$
- $1 \leq D \leq 10^9$

Dalinės užduotys

1. (4 taškai) Egzistuoja toks bokštas k ($0 \leq k \leq N - 1$), kad
    - visiems i kur $0 \leq i \leq k - 1$: $H[i] < H[i + 1]$, ir
    - visiems i kur $k \leq i \leq N - 2$: $H[i] > H[i + 1]$.
2. (11 taškų) $Q = 1$, $N \leq 2000$
3. (12 taškų) $Q = 1$
4. (14 taškų) $D = 1$
5. (17 taškų) $L = 0$, $R = N - 1$
6. (19 taškų) D vertė yra vienoda visose `max_towers` užklausose.
7. (23 taškai) Papildomų ribojimų nėra.

Pavyzdinė vertinimo programa

Pavyzdinė vertinimo programa skaito duomenis tokiu formatu:

- 1-oji eilutė: $N\ Q$
- 2-oji eilutė: $H[0]\ H[1]\ \ldots\ H[N - 1]$
- $(3 + j)$-oji eilutė ($0 \leq j \leq Q - 1$): $L\ R\ D$ skirti j-ajam klausimui

Pavyzdinė vertinimo programa išveda rezultatą tokiu formatu:

- $(1 + j)$-oji eilutė ($0 \leq j \leq Q - 1$): j-ajam klausimui grąžinama `max_towers` vertė