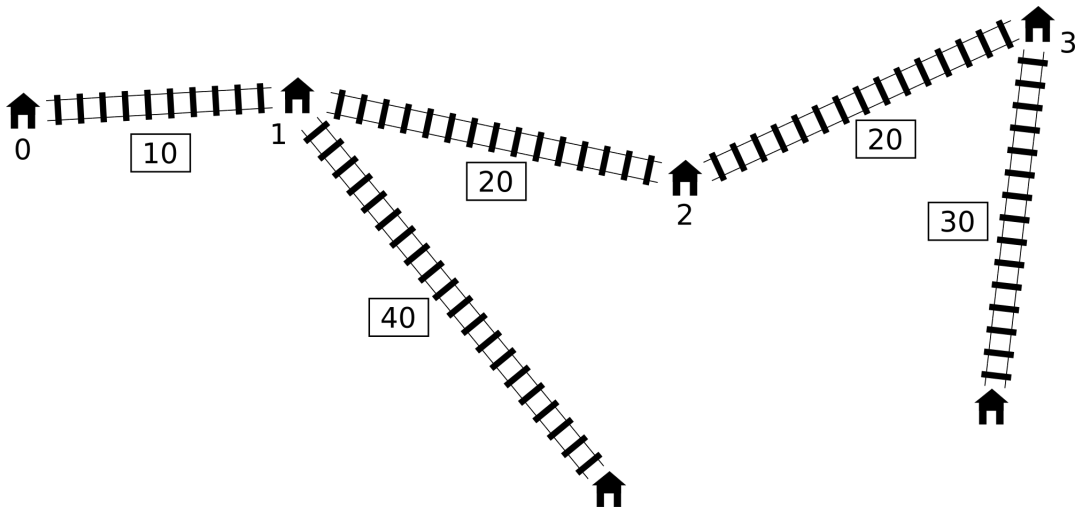
Shortcut

Sheldon se voli igrati vlakićima. Kupio je jednostavnu prugu koja se sastoji od n postaja koje su uzastopno označene od 0 do $n-1$. Postaje 0 i $n-1$ leže na krajevima glavne pruge. Udaljenost postaja i te $i+1$ iznosi l_i centimetara ($0 \leq i < n-1$).

Osim glavne pruge mogu postojati i sporedne pruge. Svaka sporedna pruga povezuje postaju na glavnoj pruzi i novu postaju koja ne leži na glavnoj pruzi. (Ove nove postaje nisu numerirane.) Najviše jedna sporedna pruga može početi na svakoj postaji glavne pruge. Duljina sporedne pruge koja počinje na postaji i iznosi d_i centimetara. Stavljamo $d_i = 0$ ako na postaji i nema sporedne pruge.



Sheldon želi izgraditi prečac: brzu prugu između dviju (možda i susjednih) postaja **glavne pruge**. Brza pruga bit će duga točno c centimetara, bez obzira koje će dvije postaje povezivati.

Sve su pruge dvosmjerne. *Udaljenost* dviju postaja je najmanja duljina dionice koja prugama ide od jedne do druge postaje. *Dijametar* cijele mreže je maksimalna udaljenost među svim parovima postaja. Drugim riječima, to je najmanji broj t takav da je udaljenost svakih dviju postaja najviše t.

Sheldon želi izgraditi brzu prugu tako da minimizira dijametar dobivene mreže.

Implementacijski detalji

Implementirajte funkciju:
`int64 find_shortcut(int n, int[] l, int[] d, int c)`

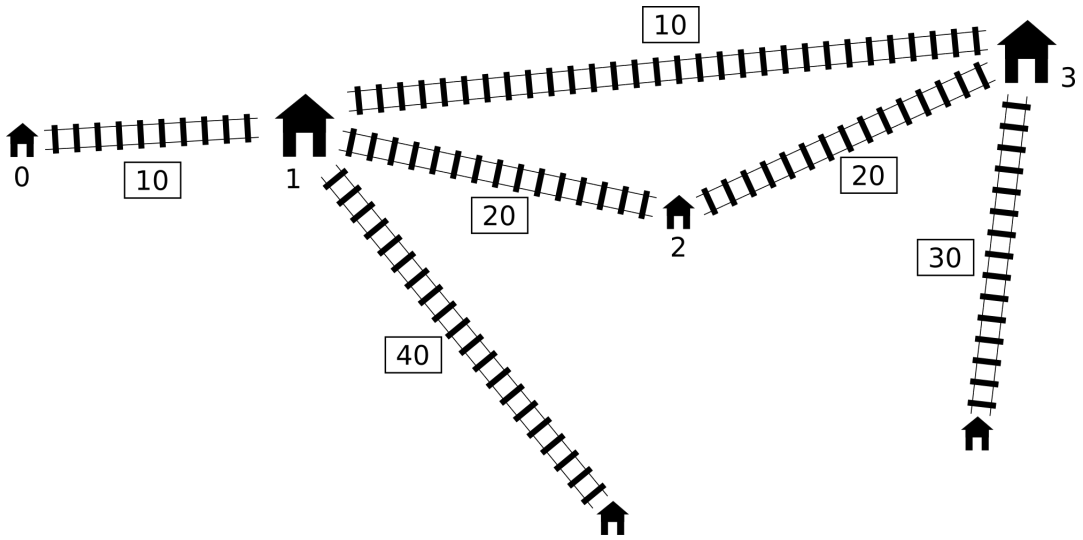
- `n`: broj postaja na glavnoj pruzi,
- `l`: udaljenosti među uzastopnim postajama na glavnoj pruzi (niz duljine $n-1$),
- `d`: duljine sporednih pruga (niz duljine n),
- `c`: duljina nove brze pruge.
- Funkcija treba vratiti najmanji mogući dijametar mreže nakon dodavanja brze pruge.

Za implementacijske detalje koristite dane template datoteke.

Primjeri

Primjer 1

Za mrežu s gornje slike grader će pozvati:
`find_shortcut(4, [10, 20, 20], [0, 40, 0, 30], 10)`
Optimalno je rješenje izgraditi brzu prugu između postaja 1 i 3, kao na slici ispod.



Dijametar nove mreže iznosi 80 centimetara, tj. funkcija treba vratiti 80.

Primjer 2

Grader poziva:

```
find_shortcut(9, [10, 10, 10, 10, 10, 10, 10, 10],
  [20, 0, 30, 0, 0, 40, 0, 40, 0], 30)
```

Optimalno rješenje povezuje postaje 2 i 7, a dijametar u tom slučaju iznosi 110.

Primjer 3

Grader poziva:

```
find_shortcut(4, [2, 2, 2],
  [1, 10, 10, 1], 1)
```

Optimalno rješenje povezuje postaje 1 i 2, smanjujući dijametar na 21.

Primjer 4

Grader poziva:

```
find_shortcut(3, [1, 1],
  [1, 1, 1], 3)
```

Povezivanje bilo kojih dviju postaja brzom prugom duljine 2 ne poboljšava početni dijametar mreže koji iznosi 4.

Podzadatci

U svim podzadatcima, $2 \leq n \leq 1\,000\,000$, $1 \leq l_i \leq 10^9$, $0 \leq d_i \leq 10^9$, $1 \leq c \leq 10^9$.

1. (9 bodova) $2 \leq n \leq 10$,
2. (14 bodova) $2 \leq n \leq 100$,
3. (8 bodova) $2 \leq n \leq 250$,
4. (7 bodova) $2 \leq n \leq 500$,
5. (33 boda) $2 \leq n \leq 3000$,
6. (22 boda) $2 \leq n \leq 100\,000$,
7. (4 boda) $2 \leq n \leq 300\,000$.
8. (3 boda) $2 \leq n \leq 1\,000\,000$.

Priloženi grader

Priloženi grader učitava ulaz u sljedećem obliku:

- redak 1: cijeli brojevi n i c,
- redak 2: cijeli brojevi $l_0, l_1, \ldots, l_{n-2}$,
- redak 3: cijeli brojevi $d_0, d_1, \ldots, d_{n-1}$.