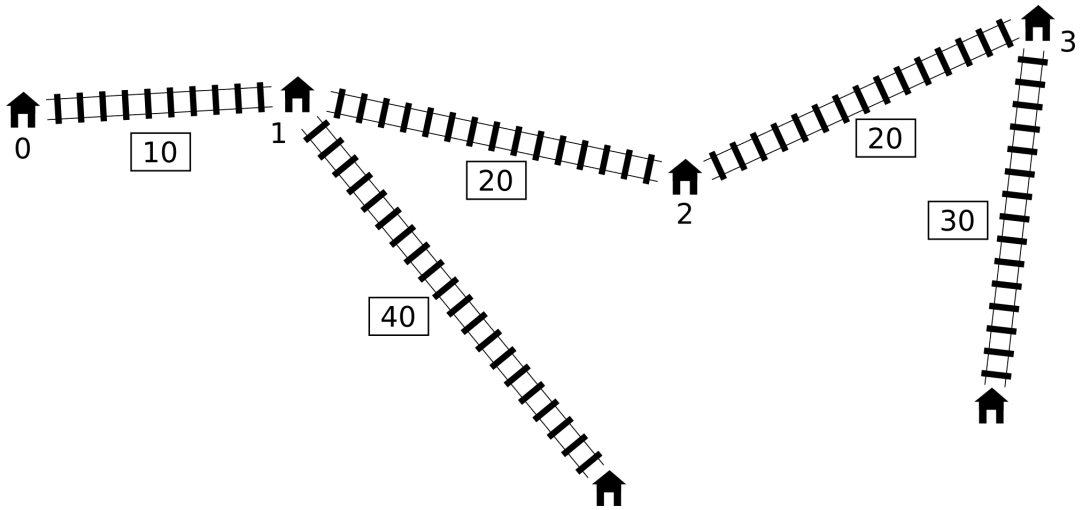
Shortcut

Pavel are un joc cu trenulețe. Acesta este foarte simplu. Există o singură linie principală constând în n stații consecutive, numerotate în ordine, de-a lungul liniei, cu numere de la 0 la $n-1$. Stațiile 0 și $n-1$ sunt situate la capetele liniei principale. Distanța dintre stațiile i și $i+1$ este de l_i centimetri ($0 \leq i < n-1$).

În afară de linia principală pot exista și linii secundare. Fiecare linie secundară este o linie de cale ferată între o stație de pe linia principală și o nouă stație nesituată pe linia principală. (Aceste stații noi nu sunt numerotate.) Din fiecare stație de pe linia principală poate pleca cel mult o linie secundară. Lungimea liniei secundare care pleaca din stația i este de d_i centimetri. Dacă $d_i = 0$ înseamnă că nu există linie secundară care pleacă din stația i.



Pavel își propune să construiască o scurtătură: o linie expres între două stații (posibil vecine) **ale liniei principale**. Linia expres va avea lungimea de exact c centimetri, indiferent care vor fi cele două stații pe care le va conecta.

Fiecare porțiune de cale ferată, inclusiv linia expres, poate fi parcursă în ambele sensuri. *Distanța* dintre două stații este cea mai mică lungime a unui treseu care unește cele două stații de-a lungul căii ferate. *Diametrul* întregii rețele de cale ferată este maximul distanțelor pentru oricare pereche de stații. Cu alte cuvinte, acesta este cel mai mic număr t, astfel încât distanța între oricare două stații este cel mult t.

Pavel dorește să construiască linia expres astfel încât diametrul rețelei rezultate să fie minimizat.

Detalii de implementare

Trebuie să implementezi funcția
`int64 find_shortcut(int n, int[] l, int[] d, int c)`

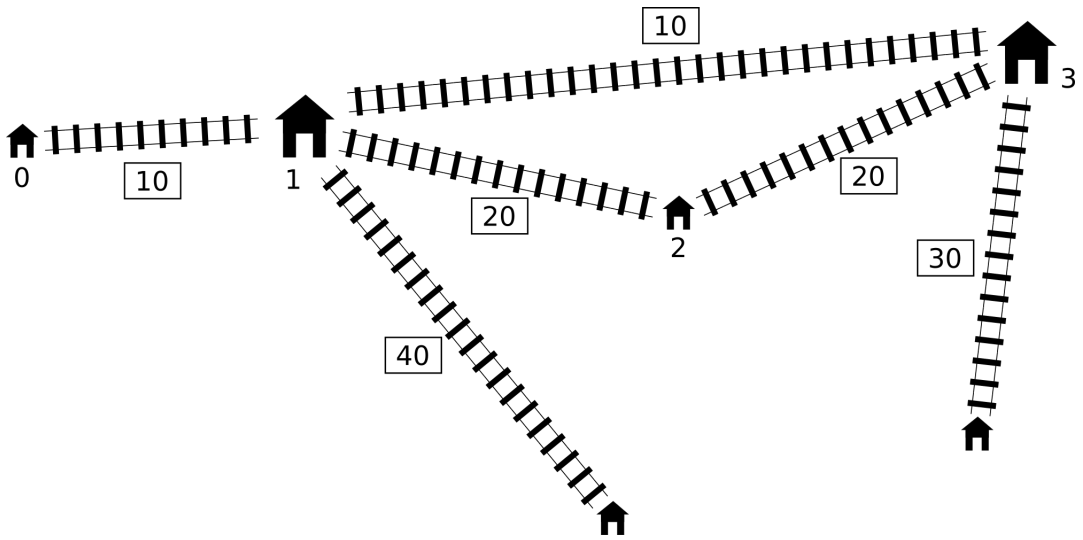
- `n`: numărul de stații de pe linia principala,
- `l`: distanțele dintre stațiile de pe linia principală (vector de lungime $n-1$),
- `d`: lungimile liniilor secundare (vector de lungime n),
- `c`: lungimea liniei expres.
- funcția trebuie să returneze cel mai mic diametru posibil al rețelei de cale ferată după adăugarea liniei expres.

Folosiți fișierele template oferite pentru detalii de implementare în limbajul vostru de programare.

Exemple

Exemplul 1

Pentru rețeaua de cale ferată de mai jos, graderul va face urmatorul apel:
`find_shortcut(4, [10, 20, 20], [0, 40, 0, 30], 10)`
Soluția optimă este să se construiască linia expres între stațiile 1 și 3, așa cum este arătat în figură.



Diametrul noii rețele de cale ferată este de 80 de centimetri, așa că funcția va trebui sa returneze 80.

Exemplul 2

Graderul face următorul apel:

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

Soluția optimă este să se conecteze stațiile 2 și 7, caz în care diametrul este 110.

Exemplul 3

Graderul face următorul apel:

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

Soluția optimă este să se conecteze stațiile 1 și 2, reducând diametrul la 21.

Exemplul 4

Graderul face următorul apel:

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

Oricum am conecta două stații cu o linie expres de lungime 3 nu se poate îmbunătăți diametrul rețelei inițiale de cale ferată care este 4.

Subtaskuri

Pentru toate subtaskurile $2 \leq n \leq 1\,000\,000$, $1 \leq l_i \leq 10^9$, $0 \leq d_i \leq 10^9$, $1 \leq c \leq 10^9$.

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

Sample grader

Sample grader-ul citește date de intrare în următorul format:

- linia 1: numerele întregi n și c,
- linia 2: numerele întregi $l_0, l_1, \ldots, l_{n-2}$,
- linia 3: numerele întregi $d_0, d_1, \ldots, d_{n-1}$.