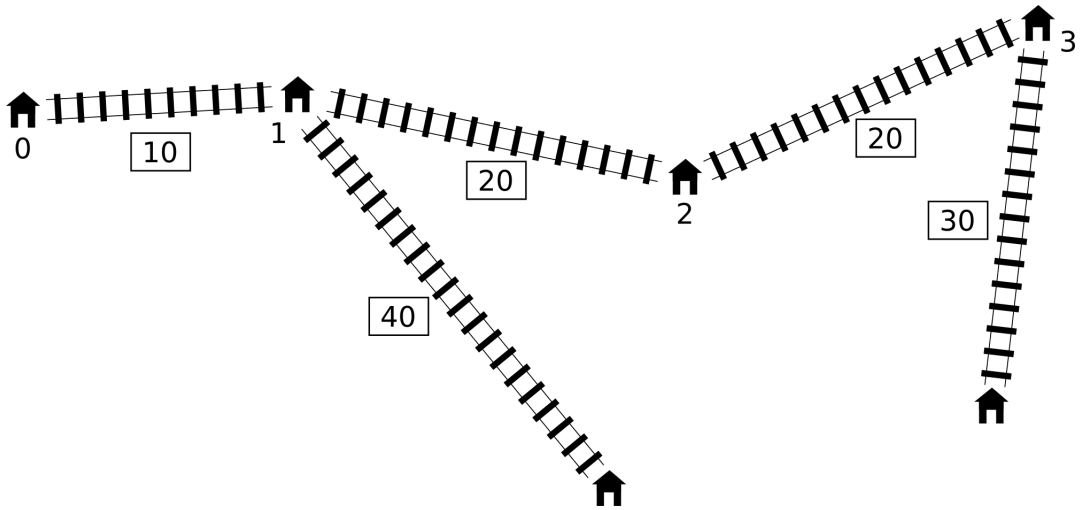
Snelkoppeling ("Shortcut")

Pavel heeft een speelgoed spoorweg. Deze is erg eenvoudig. Er is een enkele hoofdlijn die uit n stations bestaat. Deze stations zijn genummerd van 0 tot en met n-1 in volgorde langsheen de spoorlijn. De afstand tussen stations i en $i+1$ is l_i centimeter ($0 \leq i < n-1$).

Behalve de hoofdlijn kunnen er ook enkele zijsporen zijn. Elk zijspoor is een spoorlijn tussen een station van de hoofdlijn en een nieuw station dat niet op de hoofdlijn ligt. (Deze nieuwe stations zijn niet genummerd.) Er kan hoogstens één zijspoor vertrekken vanaf elk station van de hoofdlijn. De lengte van het zijspoor dat vertrekt vanaf station i is d_i centimeter. We gebruiken $d_i = 0$ om aan te geven dat er geen zijspoor is dat vertrekt vanaf station i.



Pavel heeft nu het plan opgevat om één snelkoppeling te bouwen: een expresslijn tussen twee verschillende (mogelijk naburige) stations van **de hoofdlijn**. De expresslijn zal een lengte hebben van precies c centimeter, onafhankelijk van welke twee stations het zal verbinden.

Elk stuk van de spoorweg, inclusief de nieuwe expresslijn, kan in beide richtingen gebruikt worden. De *afstand* tussen twee stations is de kleinste lengte van een route die gaat van het ene station naar het andere via de spoorwegen. De *diameter* van het hele spoorwegennetwerk is de maximale afstand tussen alle paren van stations. Met andere woorden, dit is het kleinste getal t, zo dat de afstand tussen elk paar stations hoogstens t is.

Pavel wilt de expresslijn op zo een manier bouwen dat de diameter van het resulterende netwerk minimaal is.

Implementatie details

Je moet de volgende functie implementeren:

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

- `n`: aantal stations op de hoofdlijn,
- `l`: afstanden tussen stations op de hoofdlijn (array van lengte $n-1$),
- `d`: lengtes van zijsporen (array van lengte n),
- `c`: lengte van de nieuwe expresslijn.
- De functie moet de kleinst mogelijke diameter van het spoorwegennetwerk (na het toevoegen van de expresslijn) terug geven.

Gelieve de voorziene template files te raadplegen voor implementatiedetails in jouw programmeertaal.

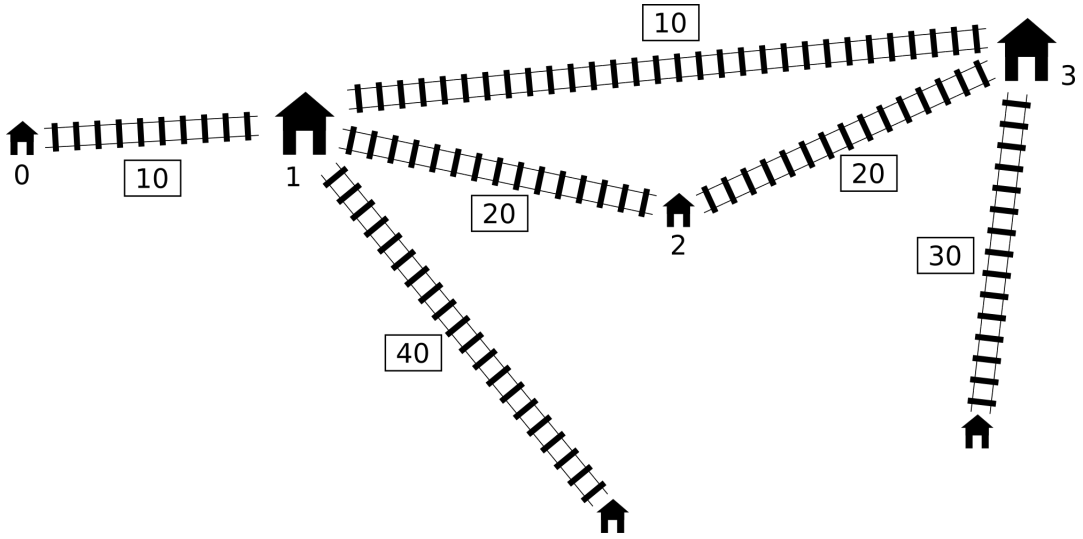
Voorbeelden

Voorbeeld 1

Voor bovenstaand spoorwegennetwerk zou de grader de volgende functie oproep maken:

`find_shortcut(4, [10, 20, 20], [0, 40, 0, 30], 10)`

De optimale oplossing is om een expresslijn te bouwen tussen stations 1 en 3, zoals hier onder getoond.



De diameter van het nieuwe spoorwegennetwerk is 80 centimeter, dus de functie moet 80 terug geven.

Voorbeeld 2

De grader maakt de volgende functie oproep:

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

De optimale oplossing is om stations 2 en 7 te verbinden, zo dat de diameter 110 is.

Voorbeeld 3

De grader maakt de volgende functie oproep:

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

De optimale oplossing is om stations 1 en 2 te verbinden, zo dat de diameter verkleind wordt tot 21.

Voorbeeld 4

De grader maakt de volgende functie oproep:

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

De aanvankelijke diameter (van lengte 4) van het spoorwegennetwerk kan niet verbeterd worden door twee stations te verbinden met behulp van de expresslijn van lengte 3.

Subtaken

In alle subtaken geldt: $2 \leq n \leq 1\,000\,000$, $1 \leq l_i \leq 10^9$, $0 \leq d_i \leq 10^9$, $1 \leq c \leq 10^9$.

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

Voorbeeldgrader

De voorbeeldgrader leest de invoer in het volgende formaat:

- regel 1: de integers n en c,
- regel 2: de integers $l_0, l_1, \ldots, l_{n-2}$,
- regel 3: de integers $d_0, d_1, \ldots, d_{n-1}$.