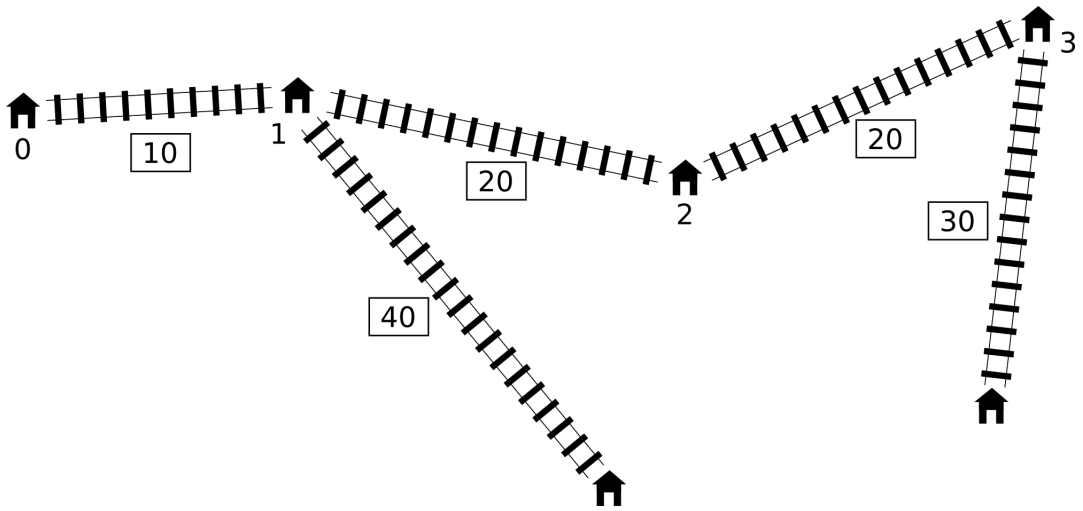
Trumpesnis kelias

Pavelas turi žaislinį geležinkelį. Jį sudaro viena pagrindinė linija, kurioje yra n stočių, nuosekliai sunumeruotų nuo 0 iki $n-1$. Atstumas tarp stočių i ir $i+1$ yra l_i centimetrų ($0 \leq i < n-1$).

Be pagrindinės linijos gali būti ir šalutinės linijos. Kiekviena šalutinė linija yra geležinkelio linija, jungianti vieną stotį, esančią pagrindinėje linijoje, ir vieną naują stotį, kuri nėra pagrindinėje linijoje. Šios naujos stotys yra nesunumeruotos. Iš bet kurios stoties, esančios pagrindinėje linijoje, gali būti nutiesta ne daugiau kaip viena šalutinė linija. Šalutinės linijos, prasidedančios stotyje i ilgis lygus d_i centimetrų. Žymėjimas $d_i = 0$ reiškia, kad iš stoties i neišeina šalutinė linija.



Pavelas nori nutiesti vieną greitąją liniją, kuri sujungtų dvi skirtingas (galima ir kaimynines) stotis, esančias **pagrindinėje linijoje**. Greitosios geležinkelio linijos ilgis visuomet bus c centimetrų, nesvarbu kurias dvi stotis ji sujungs.

Visomis geležinkelio linijoms, tame tarpe ir nauja greitąją linija, galima važiuoti abiem kryptimis. *Atstumu* tarp dviejų stočių vadinsime trumpiausio galimo maršruto nuo vienos iki kitos stoties ilgį. Viso geležinkelio tinklo *diametru* vadinsime maksimalų atstumą tarp visų galimų stočių porų. Kitaip sakant, jis lygus tokiam mažiausiam t, kad atstumas tarp bet kurios stočių poros neviršija t.

Patarkite Pavelui tarp kurių dviejų stočių nutiesti greitąją liniją, kad geležinkelių tinklo diametras būtų mažiausias galimas.

Realizacija

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

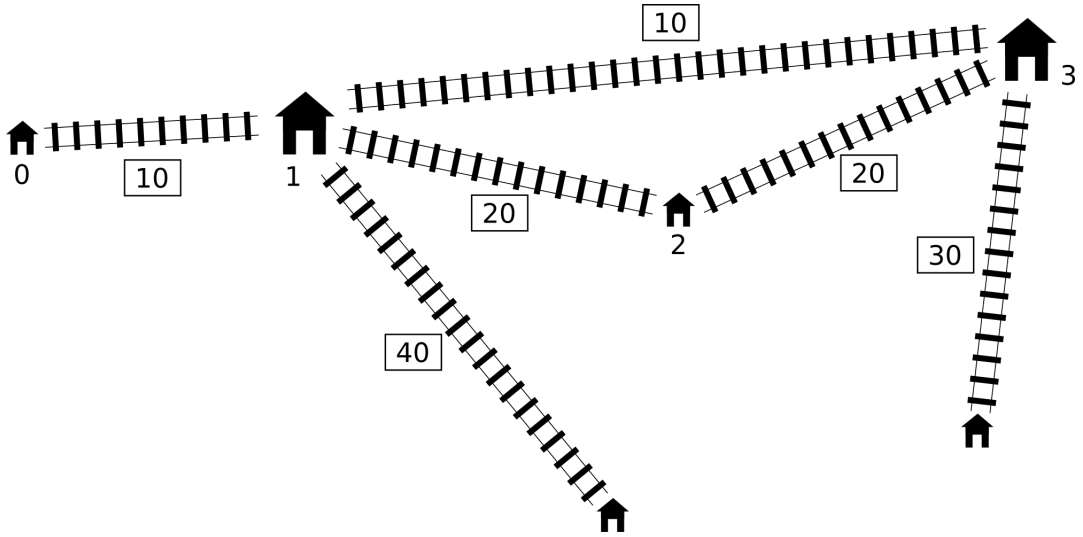
- `n`: stočių skaičius pagrindinėje linijoje,
- `l`: atstumai tarp stočių pagrindinėje linijoje ($n-1$ ilgio masyvas),
- `d`: šalutinių linijų ilgiai (n ilgio masyvas),
- `c`: naujos greitosios linijos ilgis,
- funkcija turi grąžinti mažiausią galimą geležinkelių tinklo diametrą, gautą nutiesus greitąją liniją.

Pateiktuose failų šablonuose rasite realizacijai reikalingą informaciją. Pasirinkite šabloną, atitinkantį jūsų programavimo kalbą.

Pavyzdys

Pavyzdys nr. 1

Aukščiau pateiktam pavyzdžiui, vertinimo programa šitaip iškvies funkciją:
`find_shortcut(4, [10, 20, 20], [0, 40, 0, 30], 10)`
Optimaliu atveju reikia nutiesti greitąją liniją tarp stočių nr. 1 ir nr. 3, taip, kaip parodyta paveiksle žemiau.



Šio geležinkelių tinklo diametras lygus 80 centimetrų, taigi funkcija turi grąžinti 80.

Pavyzdys nr. 2

Vertinimo programa šitaip iškviečia funkciją:

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

Optimaliame sprendinyje reikia sujungti stotį nr. 2 su nr. 7, ir šiuo atveju diametras lygus 110.

Pavyzdys nr. 3

Vertinimo programa šitaip iškviečia funkciją:

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

Optimaliame sprendinyje reikia sujungti stotį nr. 1 su nr. 2, ir diametras sumažės iki 21.

Pavyzdys nr. 4

Vertinimo programa šitaip iškviečia funkciją:

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

Jungiant bet kurias dvi stotis greitąją linija, kurios ilgis 3 nepavyks sumažinti pradinio tinklo diametro, kuris yra lygus 4.

Dalinės užduotys

Visose dalinėse užduotyse galioja: $2 \leq n \leq 1\,000\,000$, $1 \leq l_i \leq 10^9$, $0 \leq d_i \leq 10^9$, $1 \leq c \leq 10^9$.

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

Pavyzdinė vertinimo programa

Pavyzdinė vertinimo programa duomenis skaito tokiu formatu:

- 1-oji eilutė: sveikieji skaičiai n ir c,
- 2-oji eilutė: sveikieji skaičiai $l_0, l_1, \ldots, l_{n-2}$,
- 3-oji eilutė: sveikieji skaičiai $d_0, d_1, \ldots, d_{n-1}$.