Closing

Mađarska je zemlja sa N gradova, numerisanih brojevima od 0 do $N-1$.

Gradovi su spojeni sa $N-1$ *dvosmjernih direktnih puteva*, numerisanih od 0 do $N-2$. Za svako j takvo da $0 \leq j \leq N-2$, direktni put j spaja grad $U[j]$ i grad $V[j]$ i ima dužinu $W[j]$, odnosno, moguće je putovati između gradova $V[j]$ i $W[j]$ u $W[j]$ jedinica vremena. Svaki direktni put spaja dva različita grada i svaki par gradova je povezan najviše jednim direktnim putem.

Put između dva različita grada a i b je sekvenca $p_0, p_1, \ldots, p_t$ različitih gradova, takva da:

- $p_0 = a$,
- $p_t = b$,
- za svako i ($0 \leq i < t$), postoji direktni put koja spaja gradove p_i i p_{i+1}.

Moguće je putovati iz bilo kojeg grada u bilo koji grad, tačnije, postoji put između bilo koja dva različita grada. Primjetite da je ova put jedinstven za svaki par različitih gradova.

Dužina puta $p_0, p_1, \ldots, p_t$ je ukupna suma dužina t direktnih puteva koja spajaju uzastopne gradove na putanji.

U Mađarskoj, mnogo ljudi putuje da bi prisustvovali Danu Osnivanja u dva velika grada. Nakon priređenih zabava se vraćaju kući. Vlada želi da ne ometa lokalno stanovništvo, pa planiraju da zaključaju svaki od gradova u odrećeno vrijeme. Vlada će svakom gradu dodijeliti ne-negativno **vrijeme zaključavanja**. Vlada je odlučila da suma svih vremena zaključavanja ne smije biti veća od K. Preciznije, za svako i između 0 i $N-1$ (inkluzivno), vrijeme zaključavanja dodjeljeo gradu i je ne-negativan cijeli broj $c[i]$. Suma svih $c[i]$ ne smije biti veća od K.

Uzmimo za primjer grad a i određena vremena zaključavanja. Možemo reći da se u grad b **može doći** iz grada a ako i samo ako je $b = a$, ili put $p_0, \ldots, p_t$ između ova dva grada (odnosno $p_0 = a$ i $p_t = b$) zadovoljava sljedeće uslove:

- dužina puta p_0, p_1 je najviše $c[p_1]$, i
- dužina puta p_0, p_1, p_2 je najviše $c[p_2]$, i
- $\ldots$
- dužina puta $p_0, p_1, p_2, \ldots, p_t$ je najviše $c[p_t]$.

Ove godine dva glavna grada festivala se odvijaju u gradovima X i u gradu Y. Za dati raspored vrijemna zaključavanja, **bodovi ugodnosti** su definisani kao suma sljedeća dva broja:

- Broj gradova dostupnih iz grada X.
- Broj gradova dostupnih iz grada Y.

Primjetite da ako je neki grad dostupan iz grada X i dostupan je iz grada Y, on se broji *dva puta* u *bodovima ugodnosti*.

Vaš zadatak je da nađete najveći mogući broj *bodova ugodnosti* među svim mogućim rasporedima zaključavanja.

Detalji implementacije

Potrebno je da implementirate sljedeću funkciju.

```
int max_score(int N, int X, int Y, int64 K, int[] U, int[] V, int[] W)
```

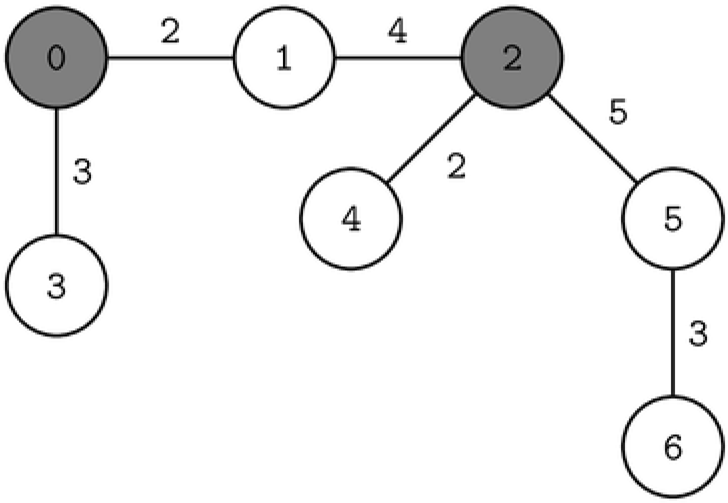
- N: broj gradova.
- X, Y: gradovi sa glavnim festivalima.
- K: najveća moguća suma svih vremena zaključavanja.
- U, V: nizovi dužine $N-1$ koji opisuju povezanost puteva.
- W: niz dužine $N-1$ koji opisuje dužine puteva.
- Ova funkcija treba da vrati najveći mogući broj *bodova ugodnosti* za neka vremena zaključavanja.
- Ova funkcija može biti pozvanai **više puta** u svakom testnom slučaju.

Primjer

Uzmimo za primjer sljedeći poziv ove funkcije:

```
max_score(7, 0, 2, 10,
          [0, 0, 1, 2, 2, 5], [1, 3, 2, 4, 5, 6], [2, 3, 4, 2, 5, 3])
```

Ovo odgovara sljedećoj mreži puteva:



Pretpostavimo da su vremena zaključavanja sljedeća:

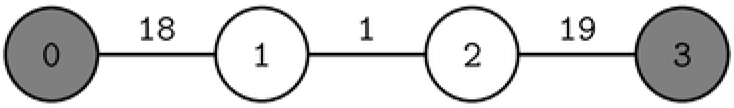
Grad	0	1	2	3	4	5	6
Vrijeme zaključavanja	0	4	0	3	2	0	0

Primijetite da je suma svih vremena zaključavanja 9, što nije više od $K = 10$. Gradovi 0, 1, i 3 su dostupni iz grada X ($X = 0$), dok gradovi 1, 2, i 4 su dostupni iz grada Y ($Y = 2$). Dakle, bodovi ugodnosti su $3 + 3 = 6$. Ne postoje vrijednosti vremena zaključavanja takva da je broj bodova ugodnosti veći od 6, tako da u ovom slučaju funkcija treba vratiti 6.

Također razmotrimo sljedeći poziv funkcije:

```
max_score(4, 0, 3, 20, [0, 1, 2], [1, 2, 3], [18, 1, 19])
```

Što odgovara sljedećoj mreži puteva:



Pretpostavimo da su vremena zaključavanja sljedeća:

City	0	1	2	3
Closing time	0	1	19	0

Grad 0 je dostupan iz grada X ($X = 0$), dok su gradovi 2 i 3 dostupni iz grada Y ($Y = 3$). Dakle bodovi ugodnosti su $1 + 2 = 3$. Ne postoje vrijednosti vremena zaključavanja takva da je broj bodova ugodnosti veći od 3, tako da bi u ovom slučaju funkcija trebala da vrati rezultat 3.

Ograničenja

- $2 \leq N \leq 200\,000$
- $0 \leq X < Y < N$
- $0 \leq K \leq 10^{18}$
- $0 \leq U[j] < V[j] < N$ (za svako j takvo da $0 \leq j \leq N-2$)
- $1 \leq W[j] \leq 10^6$ (za svako j takvo da $0 \leq j \leq N-2$)
- Između svaka dva grada postoji put
- $S_N \leq 200\,000$, gdje S_N je suma svih N tokom svih poziva funkcije `max_score` u jednom testnom primjeru.

Podzadaci

Mrežu puteva nazivamo **linearom** ako direktni put i spaja gradove i i $i+1$ (za svako i tako da $0 \leq i \leq N-2$).

1. (8 bodova) Dužina puta od X do Y je veća od $2K$.
2. (9 bodova) $S_N \leq 50$, mreža puteva je linearna.
3. (12 bodova) $S_N \leq 500$, mreža puteva je linearna.
4. (14 bodova) $S_N \leq 3\,000$, mreža puteva je linearna.
5. (9 bodova) $S_N \leq 20$
6. (11 bodova) $S_N \leq 100$
7. (10 bodova) $S_N \leq 500$
8. (10 bodova) $S_N \leq 3\,000$
9. (17 bodova) Bez dodatnih ograničenja.

Primjer gradera

Neka neko C predstavlja broj scenarija, tačnije, broj poziva na `max_score`. Primjer gradera čita ulaz u sljedećem formatu:

- linija 1: C

Slijede opisi scenarija C.

Primjer gradera čita opise svakog scenarija u sljedećem formatu:

- linija 1: $N\ X\ Y\ K$
- linija $2+j$ ($0 \leq j \leq N-2$): $U[j]\ V[j]\ W[j]$

Primjer gradera štampa jednu liniju za svaki scenario u sljedećem formatu:

- linija 1: vraća vrijednost `max_score`