Време на затворање

Унгарија е држава со N градови, нумерирани со целите броеви од 0 до $N-1$.

Градовите се поврзани со $N-1$ *дводимензионални* улици, нумерирани со целите броеви од 0 до $N-2$. За секое j такво што $0 \leq j \leq N-2$, улицата j го поврзува градот $U[j]$ и градот $V[j]$, и има должина $W[j]$, т.е., улицата овозможува да се патува помеѓу градовите за $W[j]$ временски единици. Секоја улица поврзува два различни града, и секој пар од градови е поврзан најмногу со една улица.

Пат помеѓу два различни града a и b е секвенца $p_0, p_1, \ldots, p_t$ од различни градови, таква што:

- $p_0 = a$,
- $p_t = b$,
- за секое i ($0 \leq i < t$), постои улица што ги поврзува градовите p_i и p_{i+1}.

Возможно е да се патува од кој било град до кој било друг град користејќи ги улиците, т.е., постои пат помеѓу секои два различни града. Може да се докаже дека овој пат е единствен за секој пар од различни градови.

Должина на пат $p_0, p_1, \ldots, p_t$ е збир на должините на t-те улици што поврзуваат последователни градови долж патот.

Во Унгарија, многу луѓе патуваат за да присуствуваат на прославата за Денот на основањето во два од поголемите градови. Штом завршат прославите, тие се враќаат во нивните домови. Владата сака да спречи толпата да го вознемирува локалното население, па планира да ги заклучи сите градови во одредени временски моменти. На секој град ќе му биде доделено ненегативно **време на затворање** од страна на владата. Владата одлучила дека збирот на сите времиња на затворање не смее да биде поголем од K. Попрецизно, за секое i помеѓу 0 и $N-1$, времето на затворање што е доделено на градот i е ненегативен цел број $c[i]$. Збирот на сите $c[i]$ не смее да биде поголем од K.

Да разгледаме еден град a и некое доделување на времиња на затворање. Ќе велиме дека градот b е **достиглив** од градот a ако и само ако или важи дека $b = a$, или пак дека патот $p_0, \ldots, p_t$ помеѓу овие два града (што значи дека $p_0 = a$ и $p_t = b$ во овој пат) ги задоволува следните услови:

- должината на патот p_0, p_1 е најмногу $c[p_1]$, и

- должината на патот p_0, p_1, p_2 е најмногу $c[p_2]$, и
- $\ldots$
- должината на патот $p_0, p_1, p_2, \ldots, p_t$ е најмногу $c[p_t]$.

Оваа година, двете главни локации на прославата се наоѓаат во градот X и во градот Y. За секое доделување на времиња на затворање, се дефинира **погодност (анг. convenience score)** како збир на следните два броја:

- Бројот на градови што се достигливи од градот X.
- Бројот на градови што се достигливи од градот Y.

Да забележиме дека ако некој град е достиглив од градот X и достиглив од градот Y, тоа се брои *два пати* во вредноста на погодноста.

Ваша задача е да ја пресметате максималната погодност што може да се постигне со некое доделување на времиња на затворање.

Имплементациски детали

Треба да ја имплементирате следната процедура:

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

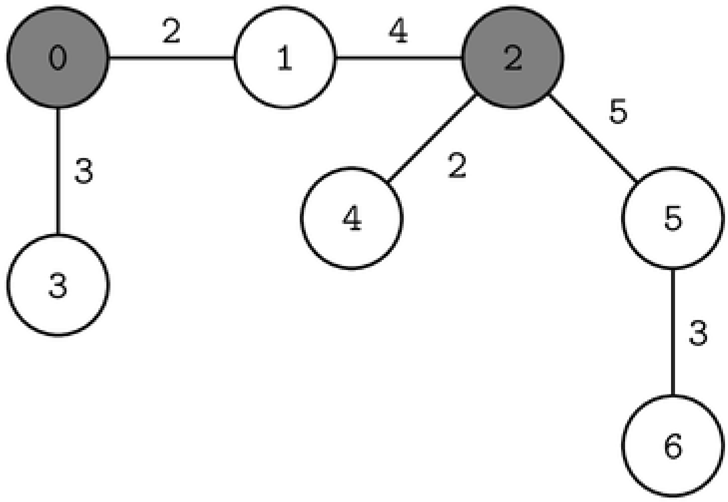
- N: бројот на градови.
- X, Y: градовите каде се наоѓаат главните локации на прославата.
- K: горната граница за збирот на времињата на затворање.
- U, V: низи со должина $N-1$ кои ги опишуваат уличните врски.
- W: низа со должина $N-1$ што ги опишува должините на улиците.
- Оваа процедура треба да ја враќа максималната погодност што може да се постигне со некое доделување на времиња на затворање.
- Оваа процедура може да биде повикана **повеќе пати** во секој тест случај.

Пример

Да го разгледаме следниот повик:

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

Ова соодветствува на следната мрежа на улици:



Да претпоставиме дека времињата на затворање се доделени на следниот начин:

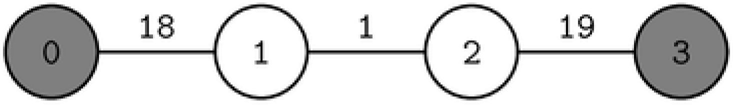
Град	0	1	2	3	4	5	6
Време на затворање	0	4	0	3	2	0	0

Да забележиме дека збирот на сите времиња на затворање е 9, што не е повеќе од $K = 10$. Градовите 0, 1 и 3 се достигливи од градот X ($X = 0$), додека градовите 1, 2 и 4 се достигливи од градот Y ($Y = 2$). Според тоа, погодноста изнесува $3 + 3 = 6$. Не постои доделување на времиња на затворање со погодност поголема од 6, па процедурата треба да врати резултат 6.

Да го разгледаме и следниот повик:

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

Ова соодветствува на следната мрежа на улици:



Да претпоставиме дека времињата на затворање се доделени на следниот начин:

Град	0	1	2	3
Време на затворање	0	1	19	0

Градот 0 е достиглив од градот X ($X = 0$), додека градовите 2 и 3 се достигливи од градот Y ($Y = 3$). Според тоа, погодноста изнесува $1 + 2 = 3$. Не постои доделување на времиња на затворање со погодност поголема од 3, па процедурата треба да врати резултат 3.

Ограничувања

- $2 \le N \le 200\,000$
- $0 \le X < Y < N$
- $0 \le K \le 10^{18}$
- $0 \le U[j] < V[j] < N$ (за секое j такво што $0 \le j \le N-2$)
- $1 \le W[j] \le 10^6$ (за секое j такво што $0 \le j \le N-2$)
- Возможно е да се патува од кој било град до кој било друг град користејќи ги улиците.
- $S_N \le 200\,000$, каде S_N е збирот на сите N низ сите повици до `max_score` во секој тест случај.

Подзадачи

За една мрежа на улици ќе велиме дека е **линеарна** ако улицата i ги поврзува градовите i и $i+1$ (за секое i такво што $0 \le i \le N-2$).

1. (8 поени) The length of the path from city X to city Y is greater than $2K$.
2. (9 поени) $S_N \le 50$, мрежата на улици е линеарна.
3. (12 поени) $S_N \le 500$, мрежата на улици е линеарна.
4. (14 поени) $S_N \le 3\,000$, мрежата на улици е линеарна.
5. (9 поени) $S_N \le 20$
6. (11 поени) $S_N \le 100$
7. (10 поени) $S_N \le 500$
8. (10 поени) $S_N \le 3\,000$
9. (17 поени) Без дополнителни ограничувања.

Пример-оценувач

Нека C го означува бројот на сценарија, т.е. бројот на повици до `max_score`. Пример-оценувачот го чита влезот во следниот формат:

- линија 1: C

Следуваат описите на C-те сценарија.

Пример-оценувачот го чита описот на секое сценарио во следниот формат:

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

Пример-оценувачот печати по една линија за секое сценарио, во следниот формат:

- линија 1: повратната вредност на процедурата `max_score`