Werewolf

U prefekturi Ibaraki postoji N gradova i M puteva. Gradovi su označeni brojevima od 0 do $N-1$ u rastućem redosljedu po broju stanovnika. Svaki put spaja dva različita grada i dvosmjeran je. Moguće je stići od bilo kog do bilo kog drugog grada pomoću ovih puteva.

Pavle je isplanirao Q putovanja, koja su označena brojevima od 0 do $Q-1$. Cilj putovanja sa rednim brojem i ($0 \leq i \leq Q-1$) je da se dođe od grada S_i do grada E_i.

Pavle je vukodlak. Ovo u prevodu znači da on ima dva moguća oblika: **ljudski oblik** i **vučji oblik**. Na početku svakog putovanja Pavle je u ljudskom obliku. Na kraju svakog putovanja, Pavle mora biti u vučjem obliku. Ovo znači da će Pavle nekad u toku putovanja da se **transformiše** tj. da se pretvori iz ljudskog oblika u vučji oblik tačno jednom i to mora da se desi u nekom od gradova na putovanju (možda S_i ili E_i).

Živjeti kao vukodlak nije lako.

Neophodno je da Pavle zaobilazi gradove sa malom populacijom kada je u ljudskom obliku i da zaobilazi gradove sa velikom populacijom kada je u vučjem obliku. Tačnije, za putovanje i, postoje brojevi (ograničenja) L_i i R_i koji zadovoljavaju $0 \leq L_i \leq R_i \leq N-1$. Ovo znači da kod putovanja sa rednim brojem i Pavle zaobilazi gradove $0, 1, \ldots, L_i-1$ kada je u ljudskom obliku i zaobilazi gradove $R_i+1, R_i+2, \ldots, N-1$ kada je u vučjem obliku. Znači, Pavle će se transformisati u jednom od gradova $L_i, L_i+1, \ldots, R_i$.

Za svako putovanje, vaš zadatak je da odredite da li je moguće da Pavle otputuje od grada S_i do grada E_i tako da budu ispoštovana gore pomenuta ograničenja. Putanja kojim Pavle ide može biti proizvoljne dužine.

Detalji implementacije

Potrebno je implementirati sljedeću funkciju:

```
int[] check_validity(int N, int[] X, int[] Y, int[] S, int[] E, int[]
 L, int[] R)
```

- `N`: broj gradova.
- `X` i `Y`: nizovi dužine M. Za svako j ($0 \leq j \leq M-1$), grad `X[j]` je direktno povezan sa gradom `Y[j]` jednim putem.

- S, E, L i R: nizovi dužine Q koji predstavljaju putovanja.

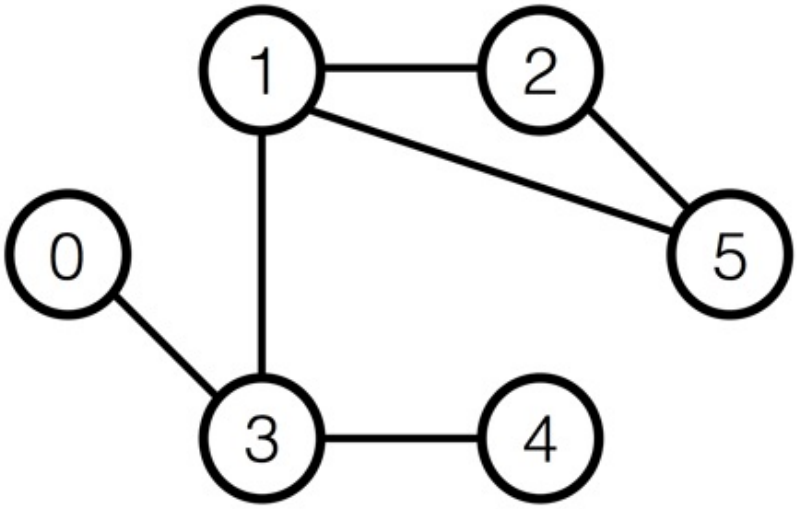
Obratite pažnju, vrijednosti M i Q su dužine nizova i mogu se dobiti onako kako je navedeno u obavještenju.

Funkcija `check_validity` se poziva tačno jednom za svaki test primjer. Ova funkcija treba da vrati niz A cijelih brojeva dužine Q. Vrijednost A_i ($0 \leq i \leq Q-1$) mora biti jednaka 1 ako je moguće da Pavle otputuje od grada S_i do grada E_i zaobilazeći gradove $0, 1, \ldots, L_i - 1$ dok je u ljudskom obliku, i gradove $R_i + 1, R_i + 2, \ldots, N - 1$ dok je u vučjem obliku. U suprotnom, ova vrijednost mora biti jednaka 0.

Primjer

Neka je $N = 6$, $M = 6$, $Q = 3$, $X = [5, 1, 1, 3, 3, 5]$, $Y = [1, 2, 3, 4, 0, 2]$, $S = [4, 4, 5]$, $E = [2, 2, 4]$, $L = [1, 2, 3]$ i $R = [2, 2, 4]$.

Program za ocjenjivanje (grader) poziva funkciju `check_validity(6, [5, 1, 1, 3, 3, 5], [1, 2, 3, 4, 0, 2], [4, 4, 5], [2, 2, 4], [1, 2, 3], [2, 2, 4])`.



Za putovanje pod rednim brojem 0, Pavle može da otputuje od grada 4 do grada 2 na sljedeći način:

- Pavle započinje u gradu 4 (trenutno je u ljudskom obliku)
- Pomjeri se u grad 3 (i dalje je u ljudskom obliku)
- Pomjeri se u grad 1 (još uvijek je u ljudskom obliku)
- Transformiše se u vukodlaka (sada je u vučjem obliku)
- Pomjeri se u grad 2 (i sad je u vučjem obliku)

Za putovanja pod rednim brojem 1 i 2 nije moguće doći od početnog do krajnjeg grada.

Dakle, vaša funkcija treba da vrati niz $[1, 0, 0]$.

U zip-fajlu uz zadatak, fajlovi `sample-01-in.txt` i `sample-01-out.txt` odgovaraju ovom primjeru. Drugi primjeri ulaza/izlaza su također dostupni u ovom dodatku.

Ograničenja

- $2 \leq N \leq 200\,000$
- $N - 1 \leq M \leq 400\,000$
- $1 \leq Q \leq 200\,000$
- $0 \leq X_j \leq N - 1$ $(0 \leq j \leq M - 1)$
- $0 \leq Y_j \leq N - 1$ $(0 \leq j \leq M - 1)$
- Moguće je putovati između bilo koja dva grada pomoću datih puteva.
- $X_j \neq Y_j$ $(0 \leq j \leq M - 1)$
- $(X_j, Y_j) \neq (X_k, Y_k)$ i $(X_j, Y_j) \neq (Y_k, X_k)$ $(0 \leq j < k \leq M - 1)$
- $0 \leq S_i \leq N - 1$ $(0 \leq i \leq Q - 1)$
- $0 \leq E_i \leq N - 1$ $(0 \leq i \leq Q - 1)$
- $S_i \neq E_i$ $(0 \leq i \leq Q - 1)$
- $0 \leq L_i \leq R_i \leq N - 1$ $(1 \leq i \leq Q - 1)$
- $L_i \leq S_i$ $(0 \leq i \leq Q - 1)$
- $E_i \leq R_i$ $(0 \leq i \leq Q - 1)$

Podzadaci

1. (7 bodova) $N \leq 100$, $M \leq 200$, $Q \leq 100$
2. (8 bodova) $N \leq 3\,000$, $M \leq 6\,000$, $Q \leq 3\,000$
3. (34 boda) $M = N - 1$ i nijedan grad nije direktno povezan sa više od 2 grada (putevi su povezani u liniju)
4. (51 bod) Bez dodatnih ograničenja

Primjer programa za ocjenjivanje (sample grader)

Program za ocjenjivanje (grader) učitava podatke u sljedećem formatu:

- red 1: N M Q
- red $(2 + j)$ $(0 \leq j \leq M - 1)$: X_j Y_j
- red $(2 + M + i)$ $(0 \leq i \leq Q - 1)$: S_i E_i L_i R_i

Program za ocjenjivanje (grader) štampa povratnu vrijednost funkcije `check_validity` u sljedećem formatu:

- red $(1 + i)$, $0 \leq i \leq Q - 1$): A_i