Werewolf

U cijeloj Ibaraki provinciji, Japan, nalazi se ukupno N gradova i M puteva. Gradovi su numerisani od 0 do $N-1$ u rastaćem redoslijedu prema broju stanovnika u njima. Svaki od puteva povezuje jedan par različitih gradova. Svakim putem moguće je putovati u oba smjera. Moguće je doći iz bilo kojeg grada u bilo koji grad koristeći postojeću mrežu puteva.

Vi planirate Q putovanja, numerisanih od 0 do $Q-1$. Putovanje i ($0 \leq i \leq Q-1$) je putovanje od grada S_i do grada E_i.

Vi ste, u stvari, vukodlak i možete biti u dva različita oblika, u **obliku čovjeka** i u **obliku vuka**. Na početku svakog od putovanja imate oblik čovjeka. Na kraju svakog od putovanja morate imati oblik vuka. Tokom putovanja morate se, dakle, **transformisati** (promijeniti se iz oblika čovjeka u oblik vuka) tačno jednom i to promjena se mora dogoditi dok se nalazite u nekom od gradova (moguće je da to budu i početni grad S_i ili krajnji grad E_i).

Život vukodlaka nije nimalo lagan. Znate da morate izbjegavati slabo naseljene gradove kada imate oblik čovjeka a isto tako morate izbjegavati gusto naseljene gradove kada imate oblik vuka. Tako uvijek izaberete, za svako putovanje i ($0 \leq i \leq Q-1$), dva cijela broja L_i i R_i ($0 \leq L_i \leq R_i \leq N-1$) koji označavaju gradove koje morate izbjegavati tokom putovanja. To znači da tokom putovanja i morate izbjegavati gradove $0, 1, \ldots, L_i-1$ dok imate oblik čovjeka i, na sličan način, morate izbjegavati gradove $R_i+1, R_i+2, \ldots, N-1$ kada imate oblik vuka. To znači, između ostalog, da ćete se, tokom putovanja i, transformisati iz čovjeka u vuka u nekom od gradova $L_i, L_i+1, \ldots, R_i$.

Vaš zadatak je da odredite, za svako putovanje, da li je moguće ili ne putovati od grada S_i do grada E_i tako da su svi spomenuti uslovi zadovoljeni. Dužina vašeg putovanje može biti proizvoljno velika.

Detalji implementacije

Vi treba da implementirate 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 svaki j ($0 \leq j \leq M-1$), grad `X[j]` je direktno povezan jednim putem sa gradom `Y[j]`.
- `S`, `E`, `L`, i `R`: nizovi dužine Q, koji predstavljaju niz putovanja.

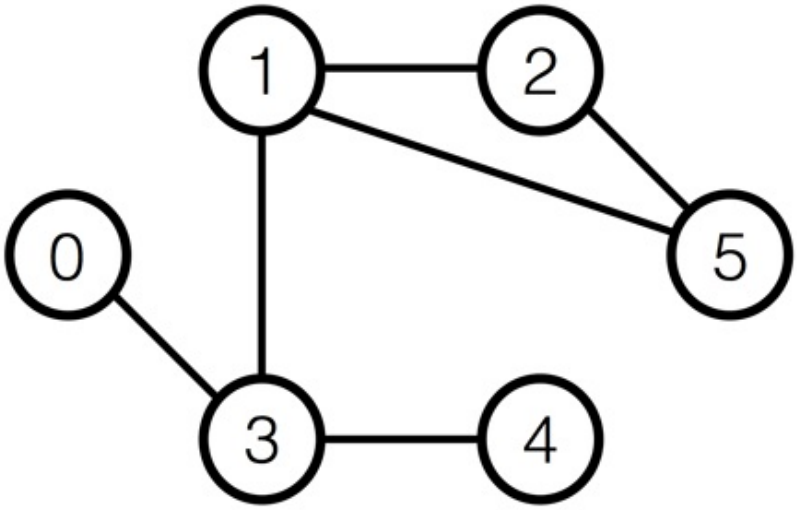
Primjetimo da su vrijednosti cijelih brojeva M i Q dužine nizova, i mogu biti dobivene kako je to opisano u napomeni o implementaciji.

Funkcija `check_validity` se poziva tačno jednomo za svaki od testnih primjera. Ova funkcija treba da vrati niz A cijelih brojeva dužine Q. Vrijednosti od brojeva A_i ($0 \leq i \leq Q-1$) moraju biti 1 ako je putovanje i moguće, to jest ako je moguće zadovoljiti sve gore postavljene uslove. Inače, vrijednost A_i je 0.

Example

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

Grader poziva `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 0 moguće je putovati od grada 4 to grada 2 na sljedeći način:

- Započeti u gradu 4 (Tada imate oblik čovjeka)
- Otputovati u grad 3 (Još uvijek ste u obliku čovjeka)
- Otputovati u grad 1 (Još uvijek ste u obliku čovjeka)
- Transformisati se u oblik vuka (Sada ste u obliku vuka)
- Otputovati u grad 2 (U obliku vuka ste završili putovanje)

Putovanja 1 i 2 nije moguće realizovati pod danim uslovima.

Prema tome, vaš program treba da vrati niz $[1, 0, 0]$.

Datoteke `sample-01-in.txt` and `sample-01-out.txt` u zipovanom formatu u prilogu

koji trebate preuzeti sa grader-a odgovaraju ovom primjeru. Još nekoliko parova jednostavnih ulaznih i izlaznih datoteka se nalazi u istom prilogu.

Ograničenja

- $2 \leq N \leq 200\,000$
- $N - 1 \leq M \leq 400\,000$
- $1 \leq Q \leq 200\,000$
- Za svaki $0 \leq j \leq M - 1$
  - $0 \leq X_j \leq N - 1$
  - $0 \leq Y_j \leq N - 1$
  - $X_j \neq Y_j$
- Dozvoljeno je putovati iz bilo kojeg grada u bilo koji grad kada postoji put izmedju njih.
- Svaki par gradova je direktno povezan sa najviše jednim putem. Drugim rječima za sve vrijedi $0 \leq j < k \leq M - 1$, $(X_j, Y_j) \neq (X_k, Y_k)$ i $(Y_j, X_j) \neq (X_k, Y_k)$.
- Za svaki $0 \leq i \leq Q - 1$
  - $0 \leq L_i \leq S_i \leq N - 1$
  - $0 \leq E_i \leq R_i \leq N - 1$
  - $S_i \neq E_i$
  - $L_i \leq R_i$

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 svaki grad je susjedan sa najviše 2 puta (gradovi se mogu smjestiti na jednu (pravu) liniju)
4. (51 bod) Nema dodatnih ograničenja

Testni grader

Testni grader čita na ulazu podatke u sljedećem formatu:

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

Testni grader štampa vraćenu vrijednost funkcije `check_validity` u sljedećem formatu:

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