Werewolf

Terdapat N kota dan M jalan di Perfektur Ibaraki, Jepang. Kota-kota dinomori dari 0 hingga $N-1$ diurutkan berdasarkan semakin padatnya populasi. Setiap jalan menghubungkan sepasang kota berbeda, dan dapat dilewati secara dua arah. Anda dapat pergi dari kota apa pun ke kota lain menggunakan satu atau lebih jalan-jalan ini.

Anda merencanakan Q buah perjalanan, dinomori dari 0 hingga $Q-1$. Perjalanan i ($0 \leq i \leq Q-1$) adalah perjalanan dari kota S_i ke kota E_i.

Anda adalah *werewolf* (manusia serigala). Anda memiliki dua rupa: **rupa manusia** dan **rupa serigala**. Pada awal setiap perjalanan Anda dalam rupa manusia. Pada akhir setiap perjalanan, Anda harus dalam rupa serigala. Selama perjalanan Anda harus **berganti rupa** (berubah dari rupa manusia ke rupa serigala) tepat satu kali. Anda dapat berganti rupa hanya ketika Anda di suatu kota tertentu (bisa jadi S_i atau E_i).

Hidup sebagai *werewolf* tidaklah mudah. Anda harus menghindari kota-kota dengan populasi rendah ketika Anda dalam rupa manusia, dan menghindari kota-kota dengan populasi tinggi ketika Anda dalam rupa serigala. Untuk setiap perjalanan i ($0 \leq i \leq Q-1$), terdapat dua batas L_i dan R_i ($0 \leq L_i \leq R_i \leq N-1$) yang menyatakan kota-kota mana yang harus dihindari. Lebih spesifik, Anda harus menghindari kota-kota $0, 1, \ldots, L_i-1$ ketika Anda dalam rupa manusia, dan harus menghindari kota-kota $R_i+1, R_i+2, \ldots, N-1$ ketika Anda dalam rupa serigala. Ini artinya pada perjalanan i, Anda hanya dapat berganti rupa di salah satu dari kota-kota $L_i, L_i+1, \ldots, R_i$.

Tugas Anda adalah untuk menentukan, untuk setiap perjalanan, apakah mungkin untuk bepergian dari kota S_i ke kota E_i sedemikian rupa sehingga memenuhi batasan-batasan yang telah dijelaskan sebelumnya. Rute yang dapat Anda ambil dapat memiliki panjang berapa pun.

Detail implementasi

Anda harus mengimplementasi fungsi berikut:

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

- `N`: banyaknya kota.
- `X` dan `Y`: array dengan panjang M. Untuk setiap j ($0 \leq j \leq M-1$), kota `X[j]`

terhubung langsung dengan kota `Y[j]` dengan sebuah jalan.
- `S`, `E`, `L`, dan `R`: array dengan panjang Q, yang merepresentasikan perjalanan-perjalanan.

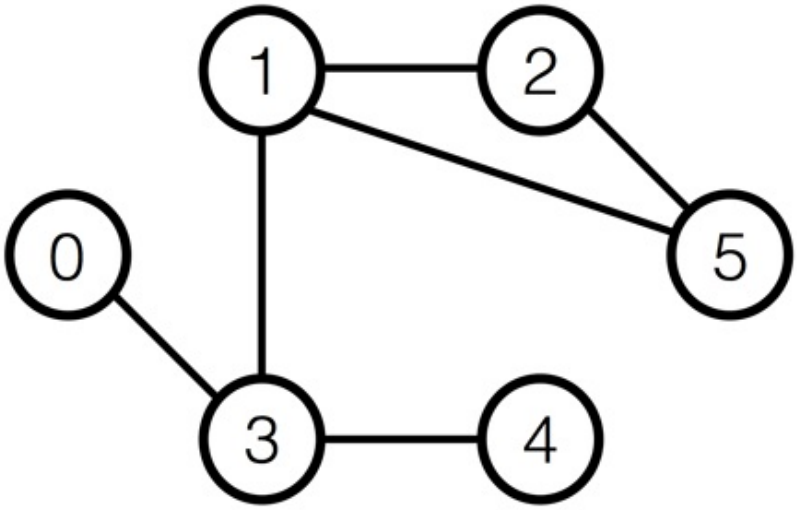
Perhatikan bahwa nilai dari M dan Q adalah panjang dari array, dan dapat diperoleh sebagaimana yang telah dijelaskan pada pengumuman implementasi.

Fungsi `check_validity` dipanggul tepat sekali untuk tiap kasus uji. Fungsi ini harus mengembalikan array integer A dengan panjang Q. Nilai dari A_i ($0 \leq i \leq Q-1$) harus `1` apabila perjalanan i mungkin dilakukan dengan memenuhi kondisi-kondisi yang telah dijelaskan sebelumnya, atau `0` jika tidak.

Contoh

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

Grader memanggil `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])`.



Untuk perjalanan 0, Anda dapat bepergian dari kota 4 ke 2 sebagai berikut:

- Mulai dari kota 4 (Anda dalam rupa manusia)
- Pergi ke kota 3 (Anda dalam rupa manusia)
- Pergi ke kota 1 (Anda dalam rupa manusia)
- Berganti rupa ke rupa serigala (Anda dalam rupa serigala)
- Pergi ke kota 2 (Anda dalam rupa serigala)

Untuk perjalanan 1 dan 2, Anda tidak dapat bepergian antara dua kota yang diberikan.

Sehingga, program Anda harus mengembalikan $[1, 0, 0]$.

Berkas `sample-01-in.txt` dan `sample-01-out.txt` di dalam lampiran paket zip berkoresponden dengan contoh ini. Paket ini juga berisi pasangan berkas-berkas

masukan/keluaran lain.

Batasan

- $2 \le N \le 200\,000$
- $N - 1 \le M \le 400\,000$
- $1 \le Q \le 200\,000$
- Untuk setiap $0 \le j \le M - 1$
  - $0 \le X_j \le N - 1$
  - $0 \le Y_j \le N - 1$
  - $X_j \ne Y_j$
- Anda dapat bepergian dari kota apa pun ke kota mana pun menggunakan jalan-jalan.
- Tiap pasang kota terhubung langsung oleh paling banyak satu jalan. Dengan kata lain, untuk semua $0 \le j < k \le M - 1$, $(X_j, Y_j) \ne (X_k, Y_k)$ dan $(Y_j, X_j) \ne (X_k, Y_k)$.
- Untuk setiap $0 \le i \le Q - 1$
  - $0 \le L_i \le S_i \le N - 1$
  - $0 \le E_i \le R_i \le N - 1$
  - $S_i \ne E_i$
  - $L_i \le R_i$

Subsoal

1. (7 poin) $N \le 100$, $M \le 200$, $Q \le 100$
2. (8 poin) $N \le 3\,000$, $M \le 6\,000$, $Q \le 3\,000$
3. (34 poin) $M = N - 1$ dan tiap kota terhubung langsung dengan paling banyak 2 jalan (kota-kota terhubung dalam sebuah garis)
4. (51 poin) Tidak ada batasan tambahan

Contoh grader

Contoh grader membaca masukan dengan format berikut:

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

Contoh grader mencetak nilai kembalian dari `check_validity` dengan format berikut:

- baris $1 + i$ ($0 \le i \le Q - 1$): A_i