Werewolf

Hay N ciudades y M caminos en la Prefectura de Ibaraki en Japón. Las ciudades están numeradas de 0 a $N-1$ en orden ascendente respecto a su población. Cada camino conecta 2 ciudades diferentes y puede ser recorrido en ambas direcciones. Puedes viajar desde cualquier ciudad a cualquier otra ciudad usando uno o más de estos caminos.

Planeaste Q viajes, numerados de 0 a $Q-1$. El viaje i ($0 \leq i \leq Q-1$) es para viajar de la ciudad S_i a la ciudad E_i.

Eres un hombre lobo que tiene dos formas: **forma humana** y **forma de lobo**. Al principio de cada viaje estás en forma humana. Al final de cada viaje debes estar en forma de lobo. Durante el viaje te tienes que **transformar** (cambiar de forma humana a forma de lobo) exactamente una vez. Sólo te puedes transformar cuando estés en alguna ciudad (incluyendo S_i o E_i).

No es fácil vivir como hombre lobo. Debes evitar ciudades poco pobladas cuando estás en forma humana y evitar ciudades muy pobladas cuando estás en forma de lobo. Para cada viaje i ($0 \leq i \leq Q-1$), existen dos enteros L_i y R_i ($0 \leq L_i \leq R_i \leq N-1$) que indican qué ciudades debes evitar. Específicamente debes evitar las ciudades $0, 1, \ldots, L_i - 1$ si estás en forma humana y evitar las ciudades $R_i + 1, R_i + 2, \ldots, N-1$ si estás en forma de lobo. Esto quiere decir que en el viaje i sólo te puedes transformar cuando estés en alguna de las ciudades $L_i, L_i + 1, \ldots, R_i$.

Tu tarea es determinar, para cada viaje, si es posible que viajes de la ciudad S_i a la ciudad E_i de tal manera que cumplas con las condiciones anteriores. No importa la longitud de la ruta que tomes.

Detalles de implementación

Debes implementar la siguiente función:

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

- `N`: el número de ciudades.
- `X` y `Y`: arreglos de longitud M. Para cada j ($0 \leq j \leq M-1$), la ciudad `X[j]` está conectada por un camino directo a la ciudad `Y[j]`.

- S, E, L, y R: arreglos de longitud Q, representando los viajes.

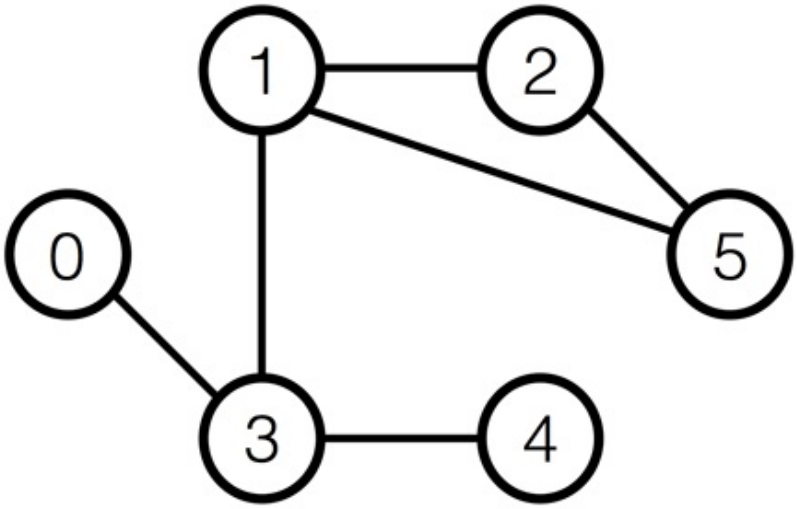
Date cuenta que los valores de M y Q son la longitud de los arreglos, para obtenerla puedes ver en el Aviso de Implementación.

La función `check_validity` se llama exactamente una vez para cada caso de prueba. Esta función debe regresar un arreglo A de enteros, de longitud Q. El valor de A_i ($0 \leq i \leq Q-1$) debe ser 1 si el viaje i se puede realizar cumpliendo con las condiciones anteriores. En caso de no ser posible, el valor de A_i debe ser 0.

Ejemplo

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

El evaluador llama `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])`.



Para el viaje 0 puedes viajar de la ciudad 4 a la ciudad 2 de la siguiente manera:

- Empiezas en la ciudad 4 (estás en forma humana)
- Viajas a la ciudad 3 (estás en forma humana)
- Viajas a la ciudad 1 (estás en forma humana)
- Te transformas en lobo (estás en forma de lobo)
- Viajas a la ciudad 2 (estás en forma de lobo)

Para los viajes 1 y 2 no puedes viajar entre las ciudades dadas.

Para este caso, tu programa debe regresar $[1, 0, 0]$.

Los archivos `sample-01-in.txt` y `sample-01-out.txt` en el ZIP adjunto corresponden a este ejemplo. Puedes encontrar otro par de ejemplos de entrada/salida en este ZIP.

Restricciones

- $2 \leq N \leq 200\,000$
- $N - 1 \leq M \leq 400\,000$
- $1 \leq Q \leq 200\,000$
- Para cada $0 \leq j \leq M - 1$
  - $0 \leq X_j \leq N - 1$
  - $0 \leq Y_j \leq N - 1$
- Puedes viajar de cualquier ciudad a cualquier otra ciudad usando caminos.
- Cada par de ciudades están conectadas por a lo más un camino. Esto es, para cada $0 \leq j < k \leq M - 1$, $(X_j, Y_j) \neq (X_k, Y_k)$ y $(Y_j, X_j) \neq (X_k, Y_k)$.
- Para cada $0 \leq i \leq Q - 1$
  - $0 \leq L_i \leq S_i \leq N - 1$
  - $0 \leq E_i \leq R_i \leq N - 1$
  - $S_i \neq E_i$ $(0 \leq i \leq Q - 1)$
  - $L_i \leq R_i$

Subtareas

1. (7 puntos) $N \leq 100$, $M \leq 200$, $Q \leq 100$
2. (8 puntos) $N \leq 3\,000$, $M \leq 6\,000$, $Q \leq 3\,000$
3. (34 puntos) $M = N - 1$ y de cada ciudad salen a lo más 2 caminos hacia otras ciudades (las ciudades están conectadas en una línea)
4. (51 puntos) No hay restricciones adicionales

Evaluador de ejemplo

El evaluador de ejemplo lee la entrada en el siguiente formato:

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

El evaluador de ejemplo imprime el valor de retorno de `check_validity` en el siguiente formato:

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