Werewolf

There are N cities and M roads in Ibaraki Prefecture, Japan. Cities are numbered from 0 through $N-1$ in increasing order of their population. Each road connects a pair of distinct cities, and can be traveled in both directions. You can travel from any city to any other city by using roads.

You planned Q trips, numbered from 0 through $Q-1$. The trip i ($0 \leq i \leq Q-1$) is to travel from the city S_i to the city E_i.

You are a werewolf. You have two forms: **human form** and **wolf form**. At the beginning of each trip you are in human form. At the end of each trip, you must be in wolf form. During the trip you have to **transform** (change from human form to wolf form) exactly once and it has to happen when you are in some city (possibly S_i or E_i).

Living as a werewolf is not easy. From your experience, you know that it is better to avoid low-populated cities when you are in human form and avoid highly-polulated cities when you are in wolf form. Specifically, for each trip i, you chose two integers L_i and R_i satisfying $0 \leq L_i \leq R_i \leq N-1$. For the trip i, you decided to avoid the cities $0, 1, \ldots, L_i - 1$ when you are in human form, and avoid the cities $R_i + 1, R_i + 2, \ldots, N-1$ when you are in wolf form. In particular, this means you will transform in one of the cities $L_i, L_i + 1, \ldots, R_i$.

For each trip, your task is to determine whether it is possible for you to travel from the city S_i to the city E_i, in a way that satisfies the aforementioned constraints. The route you take can have arbitrary length.

Implementation details

You should implement the following function:

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

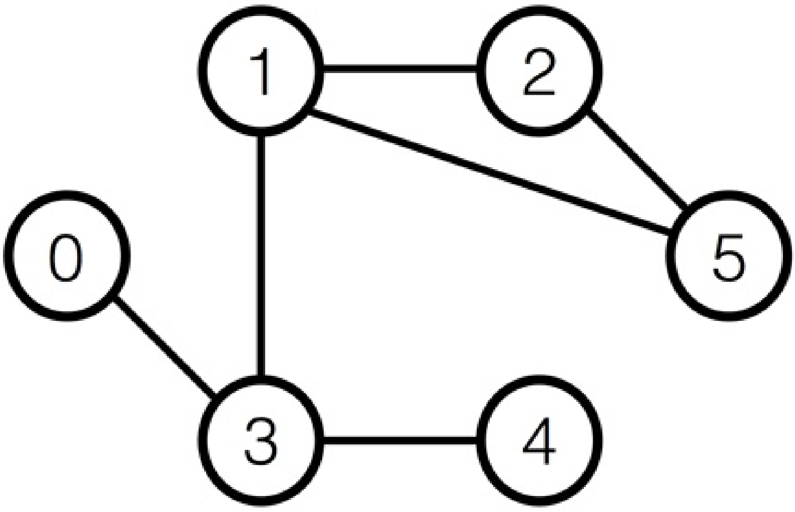
- `N`: the number of cities.
- `X` and `Y`: arrays of length M. For each j ($0 \leq j \leq M-1$), the city `X[j]` is directly connected to the city `Y[j]` by a road.
- `S`, `E`, `L`, and `R`: arrays of length Q, representing the trips.

The function `check_validity` is called exactly once for each test case. This function should return an array A of integers of length Q. The value of A_i ($0 \leq i \leq Q-1$) must be 1 if it is possible to travel from the city S_i to the city E_i, avoiding the cities $0, 1, \ldots, L_i - 1$ when you are in human form, and the cities $R_i + 1, R_i + 2, \ldots, N-1$ when you are in wolf form. Otherwise, it must be 0.

Example

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

The grader calls `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])`.



For the trip 0, you can travel from the city 4 to the city 2 as follows:

- Start at the city 4 (You are in human form)
- Move to the city 3 (You are in human form)
- Move to the city 1 (You are in human form)
- Transform yourself into wolf form (You are in wolf form)
- Move to the city 2 (You are in wolf form)

For the trips 1 and 2, you cannot travel between the given cities.

Hence, your program should return $[1, 0, 0]$.

The files `sample-01-in.txt` and `sample-01-out.txt` in the zipped attachment package correspond to this example. Other sample input/output are also available in the package.

Constraints

- $2 \leq N \leq 200\,000$
- $N - 1 \leq M \leq 400\,000$

- $1 \le Q \le 200\,000$
- $0 \le X_j \le N-1$ $(0 \le j \le M-1)$
- $0 \le Y_j \le N-1$ $(0 \le j \le M-1)$
- You can travel from any city to any other city by using roads.
- $X_j \neq Y_j$ $(0 \le j \le M-1)$
- $(X_j, Y_j) \neq (X_k, Y_k)$ and $(X_j, Y_j) \neq (Y_k, X_k)$ $(0 \le j < k \le M-1)$
- $0 \le S_i \le N-1$ $(0 \le i \le Q-1)$
- $0 \le E_i \le N-1$ $(0 \le i \le Q-1)$
- $S_i \neq E_i$ $(0 \le i \le Q-1)$
- $0 \le L_i \le R_i \le N-1$ $(1 \le i \le Q-1)$
- $L_i \le S_i$ $(0 \le i \le Q-1)$
- $E_i \le R_i$ $(0 \le i \le Q-1)$

Subtasks

1. (7 points) $N \le 100$, $M \le 200$, $Q \le 100$
2. (8 points) $N \le 3\,000$, $M \le 6\,000$, $Q \le 3\,000$
3. (34 points) $M = N-1$ and no city is directly connected to more than 2 cities (the cities are connected in a line)
4. (51 points) No additional constraints

Sample grader

The sample grader reads the input in the following format:

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

The sample grader prints the return value of `check_validity` in the following format:

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