C. Fun Tour

Time limit	2 s
Memory limit	512 MB

Description

There are N attractions in the biggest theme park in Jakarta, numbered from 0 to $N-1$. These attractions are connected by $N-1$ bidirectional roads such that there is a unique path between any pair of attractions through the roads. The roads are numbered from 0 to $N-2$. The i-th road connects the $A[i]$-th attraction and the $B[i]$-th attraction and takes one hour to walk through. To avoid congestion, each attraction is an endpoint of at most three roads.

You would like to create a tour visiting all attractions exactly once. Going through many roads when going from an attraction to another is boring. To create a fun tour, you would like to find an ordering of all attractions, such that the time required to visit the next attraction is not longer than the time required to visit the previous attraction. In other words, you would like to find a sequence $P[0], P[1], \ldots, P[N-1]$ containing all integers from 0 to $N-1$ exactly once such that the time required to go from the $P[i]$-th attraction to the $P[i+1]$-th attraction is not longer than the time required to go from the $P[i-1]$-th attraction to the $P[i]$-th attraction, for $0 < i < N-1$.

You do not have the complete map of the attractions. Therefore, you have to ask several questions to the information centre to create a fun tour. You can ask at most Q questions, each with two parameters X and Y, where $0 \leq X, Y < N$. Each question is either of the following:

- How many hours are required to go from the X-th attraction to the Y-th attraction? In particular, if $X = Y$, the answer is 0.
- How many attractions Z such that you have to visit the Y-th attraction to go from the X-th attraction to the Z-th attraction? The Y-th attraction will be counted as well. In particular, if $X = Y$, the answer is N.

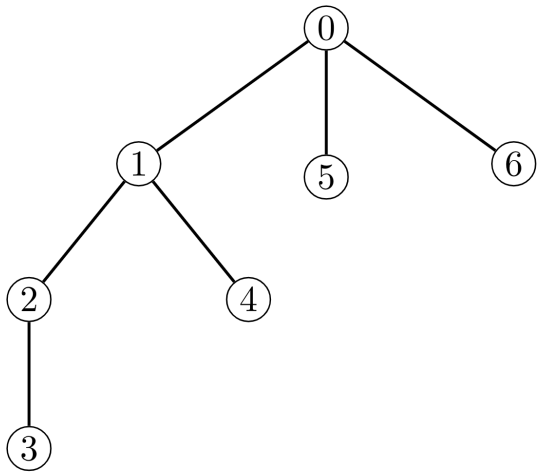
Task

You have to implement `createFunTour` function:

- `createFunTour(N, Q)` - This function will be called by the grader exactly once.
  - N: An integer representing the number of attractions.
  - Q: An integer representing the maximum number of questions.
  - This function is allowed to call two grader functions:
    - `hoursRequired(X, Y)`
      - X: An integer representing the first attraction.
      - Y: An integer representing the second attraction.
      - This function returns an integer representing hours required to go from the X-th attraction to the Y-th attraction.
      - If either X or Y is not an integer between 0 and $N-1$, then you will get a WA verdict.
    - `attractionsBehind(X, Y)`
      - X: An integer representing the first attraction.
      - Y: An integer representing the second attraction.
      - This function returns an integer representing the number of attractions Z such that you have to visit the Y-th attraction to go from the X-th attraction to the Z-th attraction.
      - If either X or Y is not an integer between 0 and $N-1$, then you will get a WA verdict.
  - This function must return an array of N integers representing the permutation of attractions in a fun tour.

Example

In the following example, $N = 7$, $Q = 400\,000$, $A = [0, 0, 0, 1, 1, 2]$, and $B = [1, 5, 6, 2, 4, 3]$. The example is illustrated by the following image:



Grader will call `createFunTour(7, 400000)`.

- If the contestant queries `hoursRequired(3, 5)`, then the function will return 4.
- If the contestant queries `hoursRequired(5, 4)`, then the function will return 3.
- If the contestant queries `attractionsBehind(5, 1)`, then the function will return 4. To go from the fifth attraction to the first, second, third, and fourth attractions, you will have to visit the first attraction.
- If the contestant queries `attractionsBehind(1, 5)`, then the function will return 1.
- The contestant can return $[3, 6, 4, 5, 2, 0, 1]$ since the hours required to visit the next attractions are $[4, 3, 3, 3, 2, 1]$ in order.

Constraints

- $2 \leq N \leq 100\,000$.
- $Q = 400\,000$.
- It is possible to travel between any pair of attractions through the roads.
- Each attraction is an endpoint of at most three roads.

Subtask 1 (10 points)

- $N \leq 17$.

Subtask 2 (16 points)

- $N \leq 500$.

Subtask 3 (21 points)

- There is a road connecting the i-th attraction and the $\lfloor \frac{i-1}{2} \rfloor$-th attraction, for all $1 \leq i < N$.

Subtask 4 (19 points)

- There is at least an attraction T such that for all $0 \leq i < N$, `hoursRequired(T, i)` < 30 and there exists an interval $[L[i], R[i]]$ $(0 \leq L[i] \leq i \leq R[i] < N)$ satisfying the following conditions:
  - You have to visit the i-th attraction to go from the T-th attraction to the j-th attraction if and only if $L[i] \leq j \leq R[i]$.
  - If $L[i] < i$, then there must be exactly one attraction X such that:
    - $L[i] \leq X < i$.

    - There is a road connecting the i-th attraction and the X-th attraction.
- If $i < R[i]$, then there must be exactly one attraction Y such that:
    - $i < Y \leq R[i]$.
    - There is a road connecting the i-th attraction and the Y-th attraction.

Subtask 5 (34 points)

- No additional constraints.

Sample Grader

The sample grader reads the input in the following format:

```
N Q
A[0] B[0]
A[1] B[1]
.
.
.
A[N-2] B[N-2]
```

The sample grader writes the integers returned by `createFunTour` if it correctly returns an array of N integers representing the permutation of attractions in a fun tour and calls both `hoursRequired` and `attractionsBehind` not more than Q times combined. Otherwise, it prints a wrong answer message.