

Training for the Marathon (marathon)

William is training for the marathon! Some running skills are always useful... who knows who may chase you in the future.



Figure 1: William after realizing that he has taken the wrong direction.

William trains in the park near his newly found home, which is a connected graph composed by N intersections (numbered from 1 to N) and $N - 1$ roads connecting two intersections each. Due to bad weather, some roads may be temporarily closed. William, however, has consulted the weather forecast and knows exactly which roads will be closed at any given moment.

Before training, William has prepared a timeline consisting of Q events of two types:

- type 1: if the road from intersection x to intersection y is closed it will be reopened, if the road is open it will be closed;
- type 2: William will train by running the path from the intersection x to the intersection y .

Notice that due to bad weather, the chosen path in a type 2 event may involve some temporarily closed roads: in such case, William will put on a trekking outfit (outfit 0) and walk directly on the grass. Otherwise, he will put on the running outfit (outfit 1) and run on the path.

Giorgio, a freelance investigator, managed to get a photo of William's timeline, and wants to use it to catch him. However, William is very smart, and he wrote down the timeline in an encrypted fashion! For type 2 events, instead of writing the true intersections x and y describing it, he wrote the encrypted x_e, y_e computed as:

$$x = [(x_e + \text{sum}) \bmod N] + 1$$

$$y = [(y_e + \text{sum}) \bmod N] + 1$$

where **sum** is the sum of all the outfit numbers he will be wearing for the type 2 events preceding the current one.

Help Giorgio catch William, by computing the outfit numbers corresponding to each type 2 event!

🔔 The *modulo* operation ($a \bmod m$) can be written in C/C++/Python as `(a % m)` and in Pascal as `(a mod m)`.

🔔 Among the attachments of this task you may find a template file `marathon.*` with a sample incomplete implementation.

Input

The first line contains the two integer N and Q : the number of intersections of the park and the number of events.

The next $N - 1$ lines contain two integers x and y each, representing a road between intersection x and intersection y .

The next Q lines contain three integers representing an event:

- 1 x y for a type 1 event;
- 2 x_e y_e for a type 2 event.

Output





For every type 2 event in the input, you should write a line with an integer corresponding to the outfit number he will use.

Constraints

- $1 \leq N, Q \leq 250\,000$.
- $1 \leq x, y \leq N$ and $1 \leq x_e, y_e \leq N$ for each event.

Scoring

Your program will be tested against several test cases grouped in subtasks. In order to obtain the score of a subtask, your program needs to correctly solve all of its test cases.

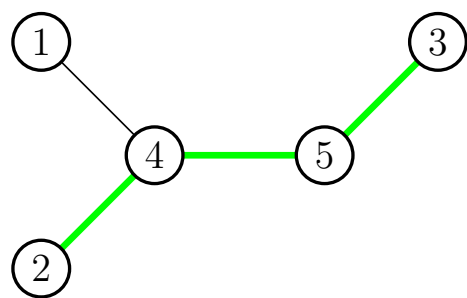
- **Subtask 1** (0 points) Examples.

- **Subtask 2** (45 points) $N, Q \leq 1000$.

- **Subtask 3** (20 points) $N \leq 1000$.

- **Subtask 4** (35 points) No additional limitations.


Examples

input	output
5 5	1
1 4	0
4 2	1
5 4	
3 5	
2 1 2	
1 4 5	
2 4 1	
1 4 5	
2 3 2	

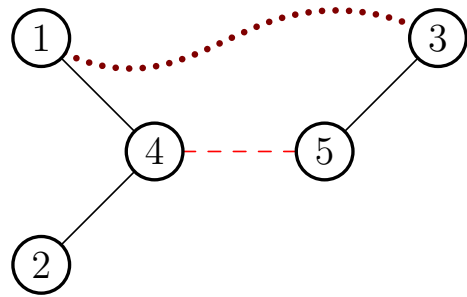
Explanation

In the first event, William runs from 2 to 3 through intersections 4 and 5 with the running outfit 1.



In the second event, the road from 4 to 5 is closed.

In the third event, there is no path between 1 and 3, therefore William walks on grass with the trekking outfit 0.



In the fourth event, the road from 4 to 5 is reopened.

In the fifth event, William runs from 5 to 4 with outfit 1.

