1.

XYZ Island has many cities. In the summer, many travelers will come to the island and attend festive events in different cities. The festive events in XYZ Island are crazy. Once it starts, it will never end. In the following sentences, the cities which have festive events are called festive cities. At the beginning, only city No. 1 is festive city. If a new city becomes festive city, the government will tell the information center about this news. Every day, the information center will receive many inquiries from travelers from different cities of this land. They want to know the closest festive city, and calculate the distance (If current city has festive event, the distance is 0). Due to the growing number of the travelers, the information center is overloaded. The government wants to fix the problem by developing a system to handle the inquiries automatically. As a fact, cities in XYZ Island are connected with highways (bidirectional, length of every highway is 1). Any two cities are connected directly or indirectly, and there is ONLY one path between any 2 cities.

Input

There are two integers in the first line, n (2 ≤ n 10^5) and m (1 ≤ m ≤ 10^5), n is the number of cities in the XYZ Island and m is the number of queries. The coming n-1 lines are the highways which connect two cities. In the line, there are two integers a and b representing two cities. ((1 ≤ a, b ≤ n), a ≠ b) Each line means the highway connecting the two cities. Next m lines are inquiries from travelers or news from government. Each line has two integers q and c (1 ≤ q ≤ 2, 1 ≤ c ≤ n). If q= 1, the government announce a new festive city c. If q= 2, you have to find and print the shortest distance from the city c to the closest festive city.

Output

Results from each (q = 2) Questions. Print every result with a new line.

Sample Test

Input 5 5 1 2 1 3 3 4 3 5 2 5 2 3 1 3 2 3 2 4

Output 2 1 0 1

2.

A small frog wants to get to the other side of the road. The frog is currently located at position X and

wants to get to a position greater than or equal to Y. The small frog always jumps a fixed distance, D.

Count the minimal number of jumps that the small frog must perform to reach its target.

Write a function:

int solution(int X, int Y, int D)

{

//Write your code here

}

That, given three integers X, Y and D, returns the minimal number of jumps from position X to a position equal to or greater than Y.

For example, given:

X = 10 Y = 85 D = 30

The function should return 3, because the frog will be positioned as follows:

- after the first jump, at position 10 + 30 = 40
- after the second jump, at position 10 + 30 + 30 = 70
- after the third jump, at position 10 + 30 + 30 + 30 = 100

Write an efficient algorithm for the following assumptions:

- X, Y and D are integers within the range [1..1,000,000,000];
- X ≤ Y.

3.

Now a small frog wants to get to the other side of a river. The frog is initially located on one bank of the river (position 0) and wants to get to the opposite bank (position X+1). Leaves fall from a tree onto the surface of the river.

You are given an array A consisting of N integers representing the falling leaves. A[K] represents the position where one leaf falls at time K, measured in seconds.

The goal is to find the earliest time when the frog can jump to the other side of the river. The frog can cross only when leaves appear at every position across the river from 1 to X (that is, we want to find the earliest moment when all the positions from 1 to X are covered by leaves). You may assume that the speed of the current in the river is negligibly small, i.e. the leaves do not change their positions once they fall in the river.

For example, you are given integer X = 5 and array A such that:

A[0] = 1 A[1] = 3 A[2] = 1 A[3] = 4 A[4] = 2 A[5] = 3 A[6] = 5 A[7] = 4

In second 6, a leaf falls into position 5. This is the earliest time when leaves appear in every position across the river.

Write a function:

class Solution { public int solution(int X, int[] A); }

that, given a non-empty array A consisting of N integers and integer X, returns the earliest time when the frog can jump to the other side of the river.

If the frog is never able to jump to the other side of the river, the function should return −1.

For example, given X = 5 and array A such above, the function should return 6, as explained above.

Write an efficient algorithm for the following assumptions:

- N and X are integers within the range [1..100,000];
- Each element of array A is an integer within the range [1..X].