2773. Height of Special Binary Tree Premium

Medium Topics Hint

You are given a `root`, which is the root of a **special** binary tree with `n` nodes. The nodes of the special binary tree are numbered from `1` to `n`. Suppose the tree has `k` leaves in the following order: $b_1 < b_2 < \ldots < b_k$.

The leaves of this tree have a **special** property! That is, for every leaf b_i, the following conditions hold:

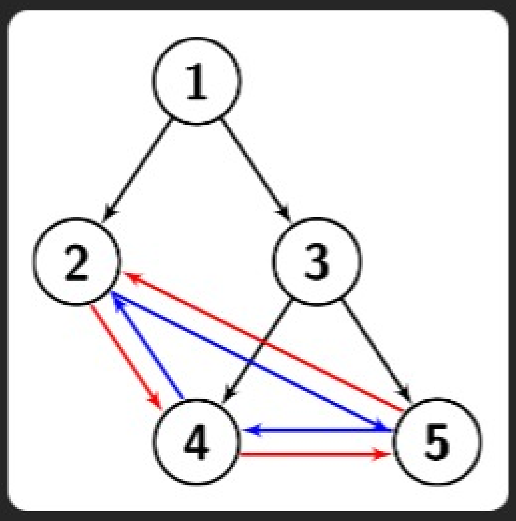
- The right child of b_i is b_{i+1} if $i < k$, and b_1 otherwise.
- The left child of b_i is b_{i-1} if $i > 1$, and b_k otherwise.

Return *the height of the given tree.*

Note: The height of a binary tree is the length of the **longest path** from the root to any other node.

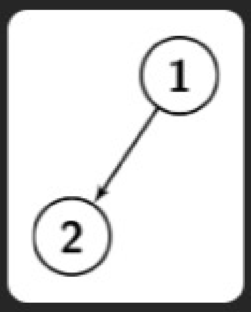
Example 1:

```
Input: root = [1,2,3,null,null,4,5]
Output: 2
Explanation: The given tree is shown in the following picture. Each leaf's left child is the leaf to its left (shown with the blue edges). Each leaf's right child is the leaf to its right (shown with the red edges). We can see that the graph has a height of 2.
```



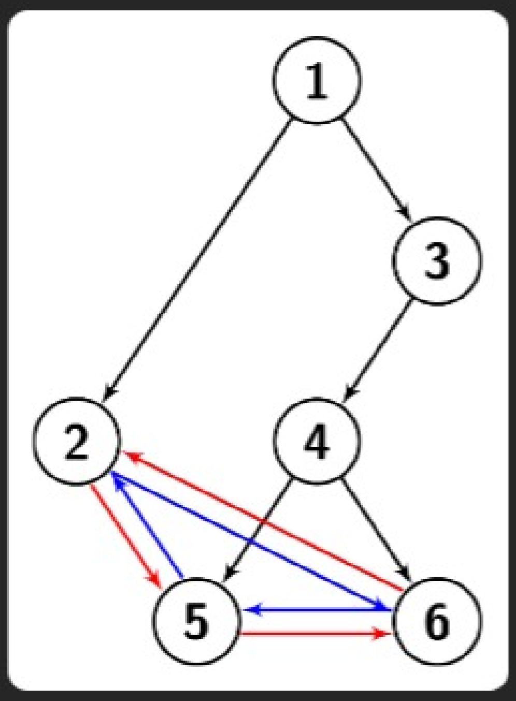
Example 2:

```
Input: root = [1,2]
Output: 1
Explanation: The given tree is shown in the following picture. There is only one leaf, so it doesn't have any left or right child. We can see that the graph has a height of 1.
```



Example 3:

```
Input: root = [1,2,3,null,null,4,null,5,6]
Output: 3
Explanation: The given tree is shown in the following picture. Each leaf's left child is the leaf to its left (shown with the blue edges). Each leaf's right child is the leaf to its right (shown with the red edges). We can see that the graph has a height of 3.
```



Constraints:

- `n == number of nodes in the tree`
- $2 <= n <= 10^4$
- `1 <= node.val <= n`
- The input is generated such that each `node.val` is unique.

Seen this question in a real interview before? 1/5

Yes No

Accepted 889 | Submissions 1.2K | Acceptance Rate 72.7%

Topics

Tree Depth-First Search Breadth-First Search Binary Tree

Hint 1

To solve the problem, we must first distinguish leaves from internal nodes.

Hint 2

For some node v, if v.left == null or v.right == null, then v is not a leaf.

Hint 3

If the previous condition does not hold, and v.left.right == v, then v is a leaf node.

Hint 4

Now that we can check if some node is a leaf, we can make the function "heightOfTree" a recursive function that returns the tree's height in which its input is the root of that subtree.

Discussion (5)