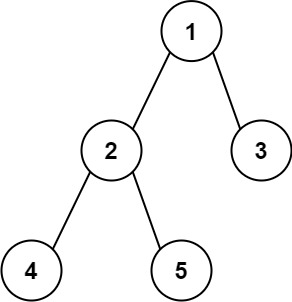
Given the root of a binary tree, return *the length of the****diameter****of the tree*.

The **diameter** of a binary tree is the **length** of the longest path between any two nodes in a tree. This path may or may not pass through the root.

The **length** of a path between two nodes is represented by the number of edges between them.

**Example 1:**



**Input:** root = [1,2,3,4,5]

**Output:** 3

**Explanation:** 3 is the length of the path [4,2,1,3] or [5,2,1,3].

**Example 2:**

**Input:** root = [1,2]

**Output:** 1

Solution:

/\*\*

\* Definition for a binary tree node.

\* public class TreeNode {

\* int val;

\* TreeNode left;

\* TreeNode right;

\* TreeNode() {}

\* TreeNode(int val) { this.val = val; }

\* TreeNode(int val, TreeNode left, TreeNode right) {

\* this.val = val;

\* this.left = left;

\* this.right = right;

\* }

\* }

\*/

class Solution {

public int diameterOfBinaryTree(TreeNode root) {

int[] diameter = new int[1];

height(root, diameter);

return diameter[0];

}

private int height(TreeNode node, int[] diameter){

if(node == null)

return 0;

int lh = height(node.left, diameter);

int rh = height(node.right, diameter);

diameter[0] = Math.max(diameter[0], lh+rh);

return 1+ Math.max(lh, rh);

}

}

T.C = O(N)

S.C= O(N)