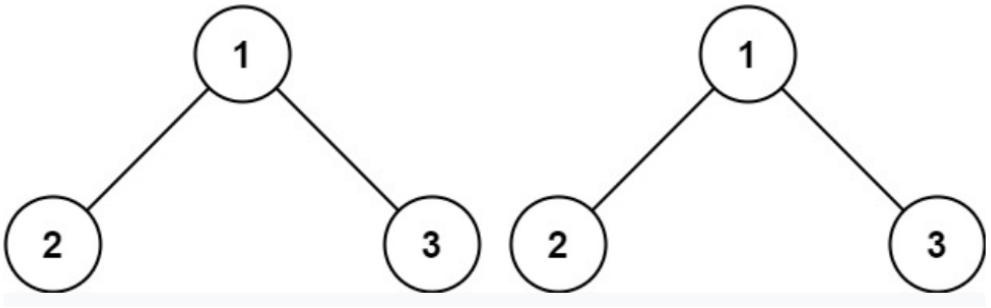
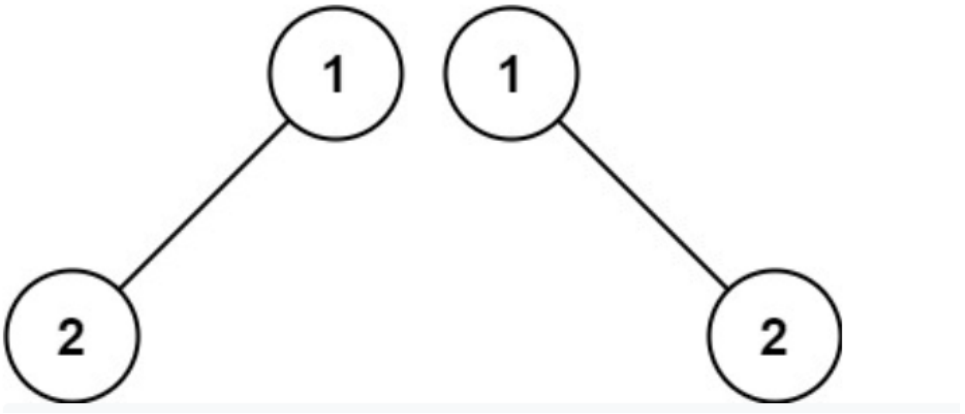
Exam 08 Question Set

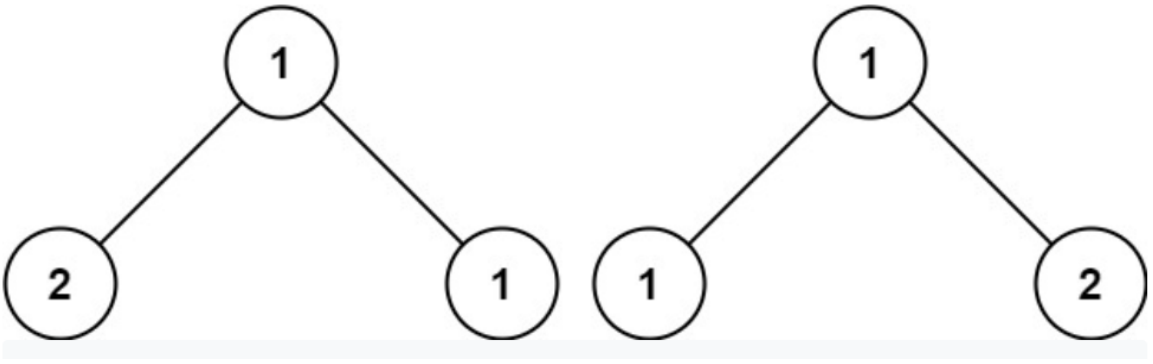
1. Given the roots of two binary trees, write a function named `bool isSame(Node* root1, Node* root2)` which will take two roots and check if they are the same or not. Two binary trees are considered the same if they are structurally identical, and the nodes have the same value.
   For example:



These two binary trees are the same.

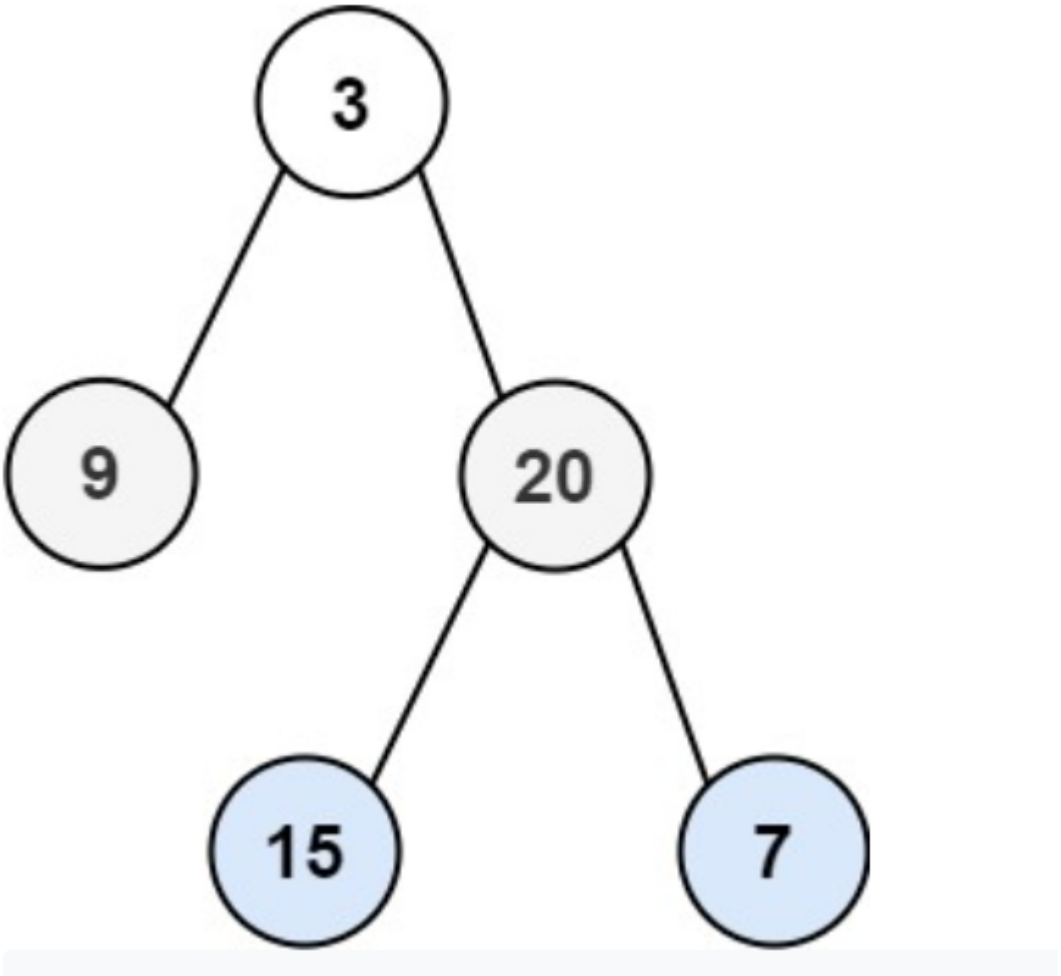


These two binary trees are not the same



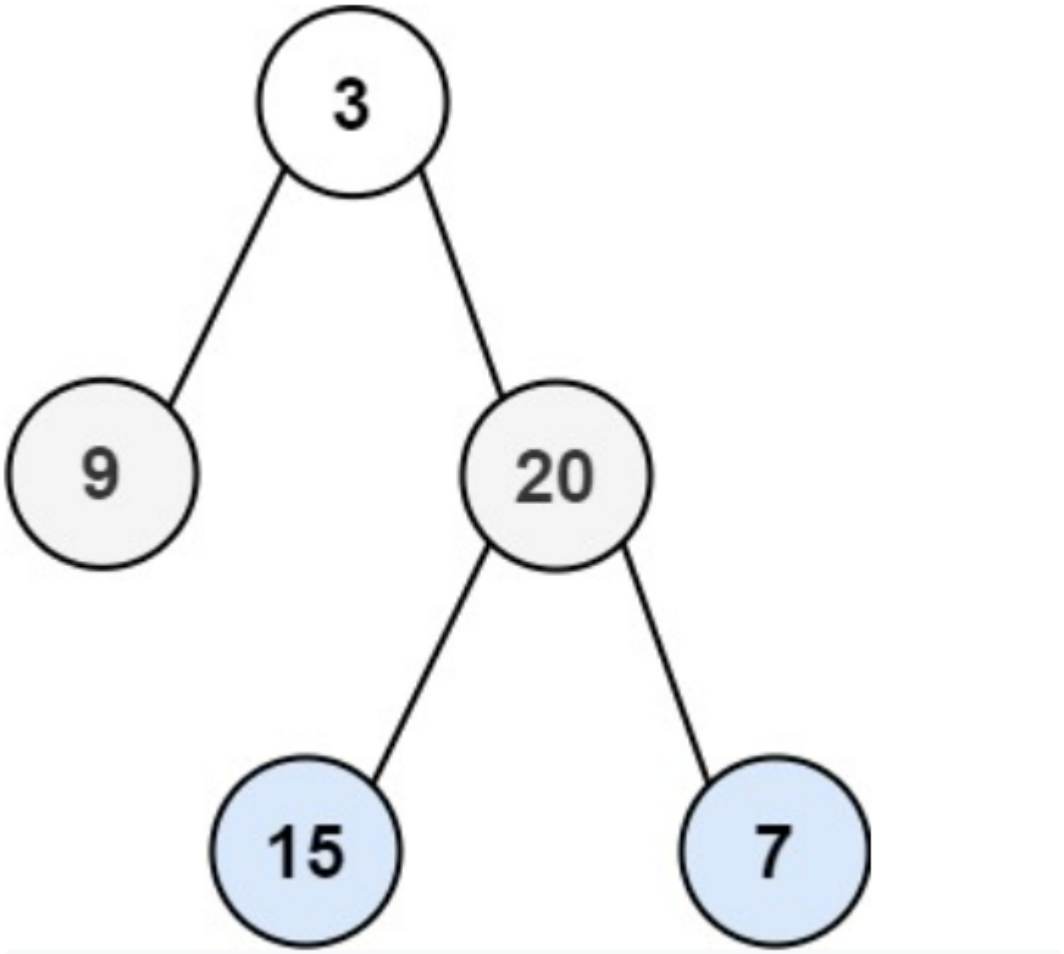
These two binary trees are not the same..

2. Given the root of a binary tree, make a function named `void level_order(Node* root)` and print the level order traversal of its nodes' values. (i.e., from left to right, level by level).
   For example:



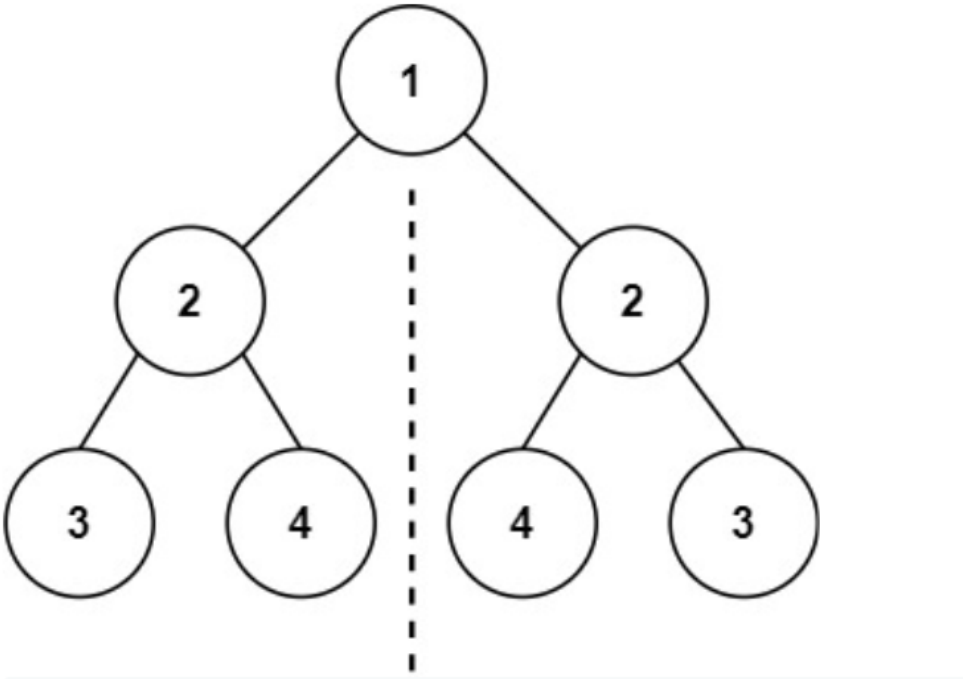
   Output: 3 9 20 15 7

3. Given the root of a binary tree, make a function named void `level_order_reverse(Node* root)` and output the bottom-up level order traversal of its nodes' values. (i.e., from left to right, level by level from leaf to root).
For example:

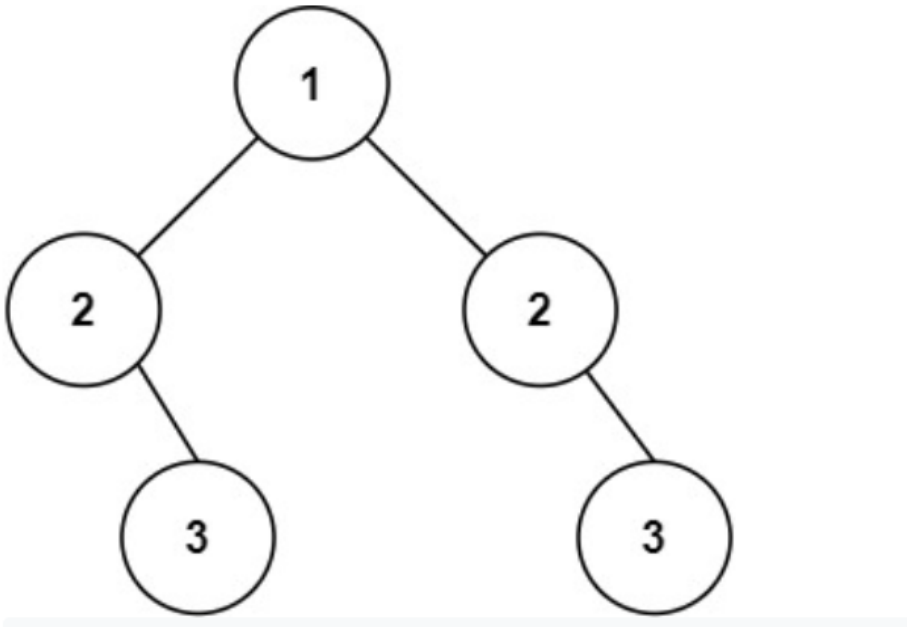


Output: 15 7 9 20 3

4. Given the root of a binary tree, make a function named `bool isSymmetric(Node* root)` check whether it is a mirror of itself (i.e., symmetric around its center).
   For example:

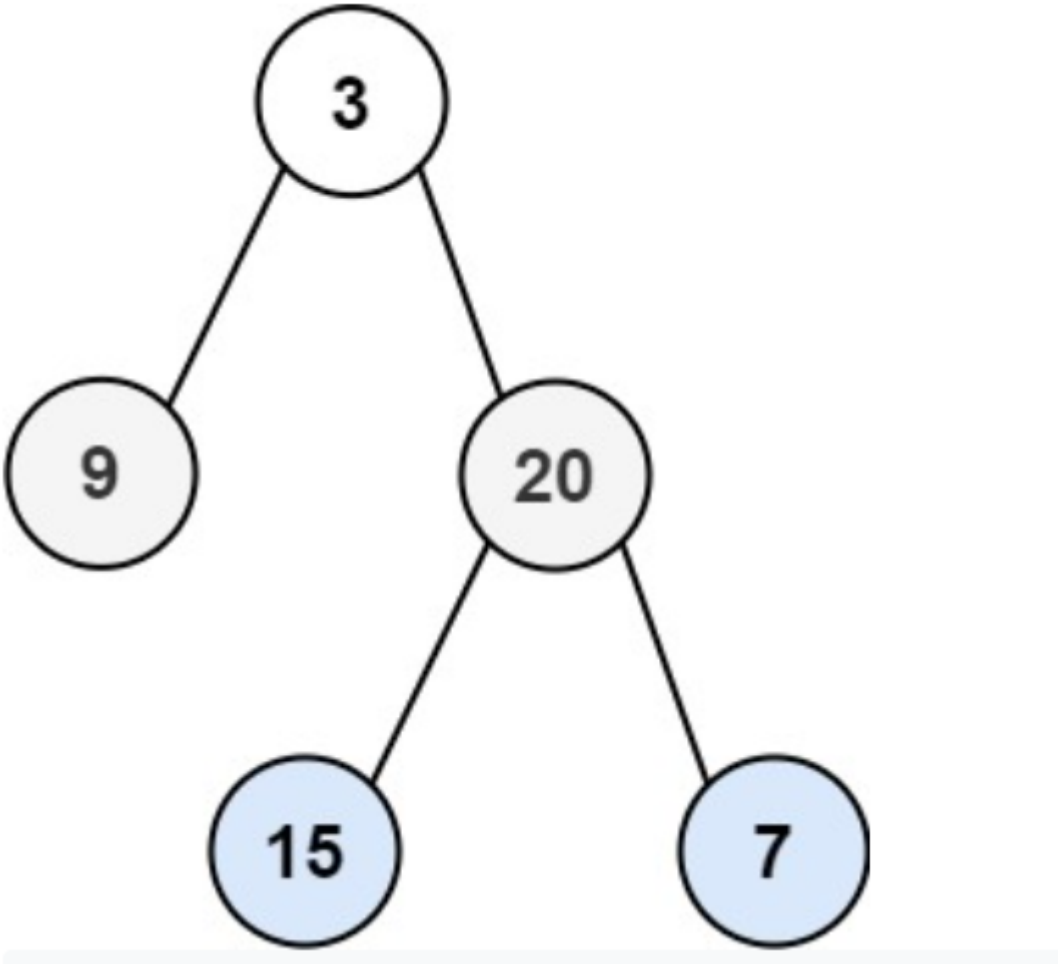


This tree is a symmetric tree.



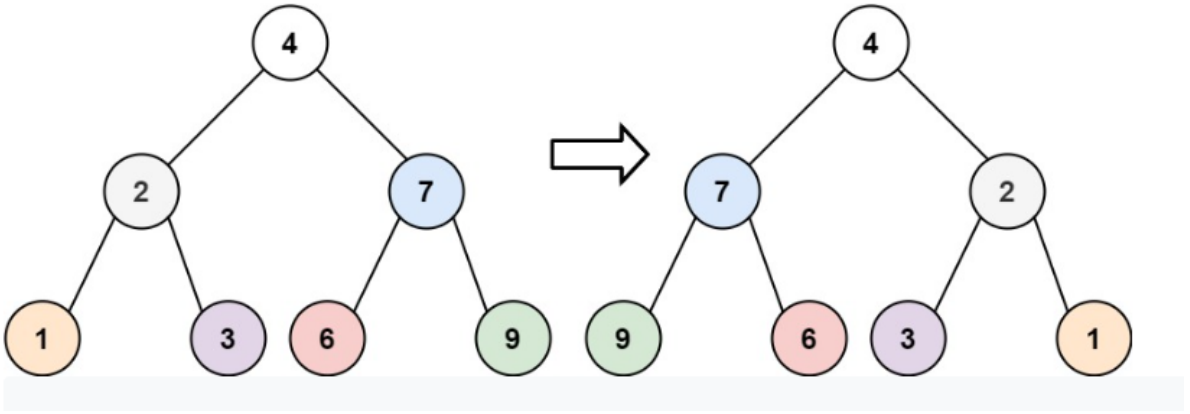
This tree is not a symmetric tree.

5. Given the root of a binary tree, make a function named `void zigzag_order(Node* root)` and output the zigzag level order traversal of its nodes' values. (i.e., from left to right, then right to left, then again left to right for the next level and this order will continue).
   For example:



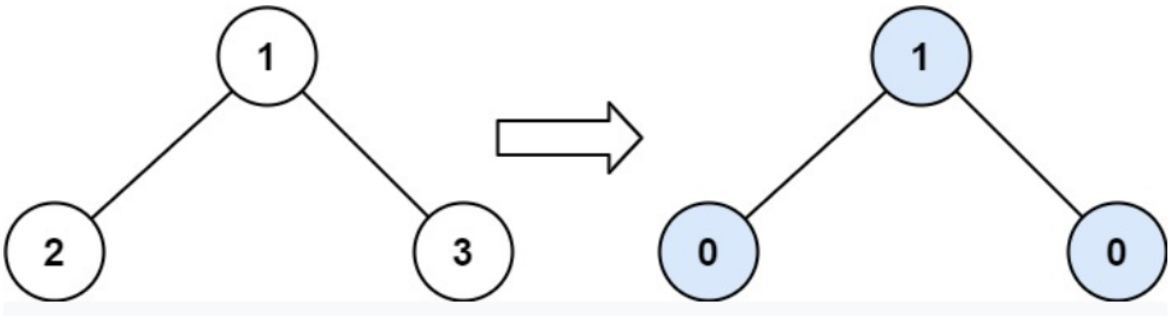
   Output: 3 20 9 15 7

6. Given the root of a binary tree, make a function named `Node* invert_tree(Node* root)` to invert the tree, and return its root
   For example:

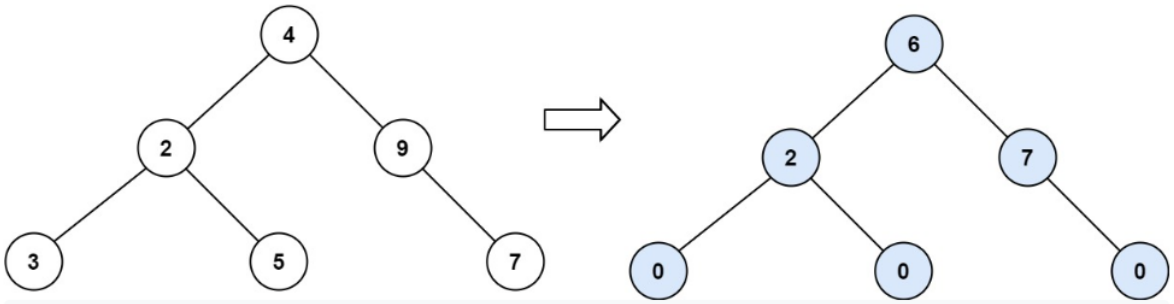


   You need to return the updated binary tree root.

   .
7. Given the root of a binary tree,make a function named `int findTilt(Node* root)` and return the sum of every tree node's tilt. The tilt of a tree node is the absolute difference between the sum of all left subtree node values and all right subtree node values.
   If a node does not have a left child, then the sum of the left subtree node values is treated as 0. The rule is similar if the node does not have a right child.
   For example:

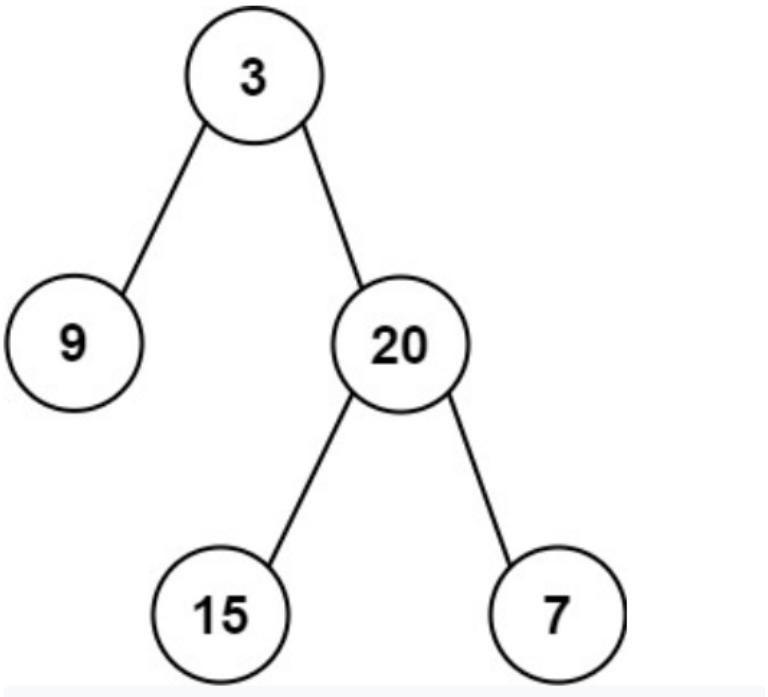


   Output: 1
   Explanation:
   Tilt of node 2 : |0-0| = 0 (no children)
   Tilt of node 3 : |0-0| = 0 (no children)
   Tilt of node 1 : |2-3| = 1 (left subtree is just left child, so sum is 2; right subtree is just right child, so sum is 3)
   Sum of every tilt : 0 + 0 + 1 = 1



Output: 15
Explanation:
Tilt of node 3 : |0-0| = 0 (no children)
Tilt of node 5 : |0-0| = 0 (no children)
Tilt of node 7 : |0-0| = 0 (no children)
Tilt of node 2 : |3-5| = 2 (left subtree is just left child, so sum is 3; right subtree is just right child, so sum is 5)
Tilt of node 9 : |0-7| = 7 (no left child, so sum is 0; right subtree is just right child, so sum is 7)
Tilt of node 4 : |(3+5+2)-(9+7)| = |10-16| = 6 (left subtree values are 3, 5, and 2, which sums to 10; right subtree values are 9 and 7, which sums to 16)
Sum of every tilt : 0 + 0 + 0 + 2 + 7 + 6 = 15

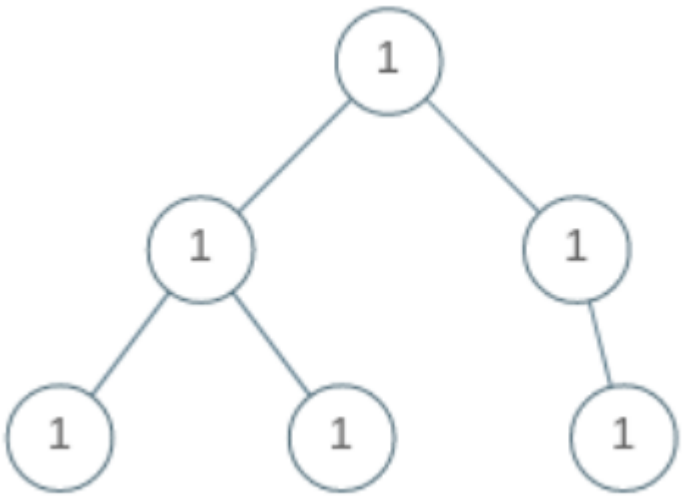
8. Given the root of a binary tree, make a function named `void average_level(Node* root)` and print the average value of the nodes on each level.
For example:



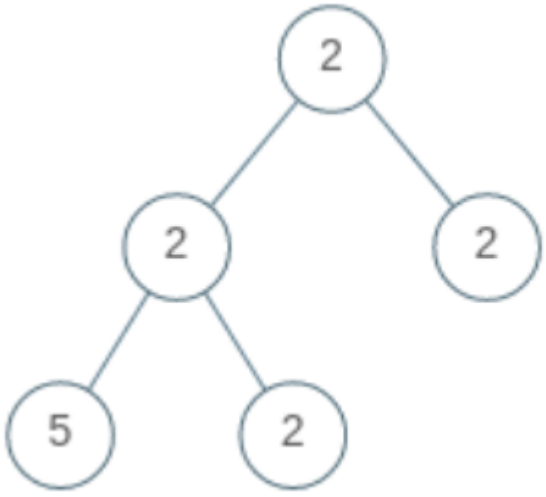
Output: 3 14.5 11
Explanation:

The average value of nodes on level 0: 3
The average value of nodes on level 1: (9+20)/2 = 14.5
The average value of nodes on level 2: (15+7)/2 = 11

9. A binary tree is uni-valued if every node in the tree has the same value. Given the root of a binary tree, make a function named `bool is_unival(Node* root)` and return true if the given tree is uni-valued, or false otherwise.
   For example:

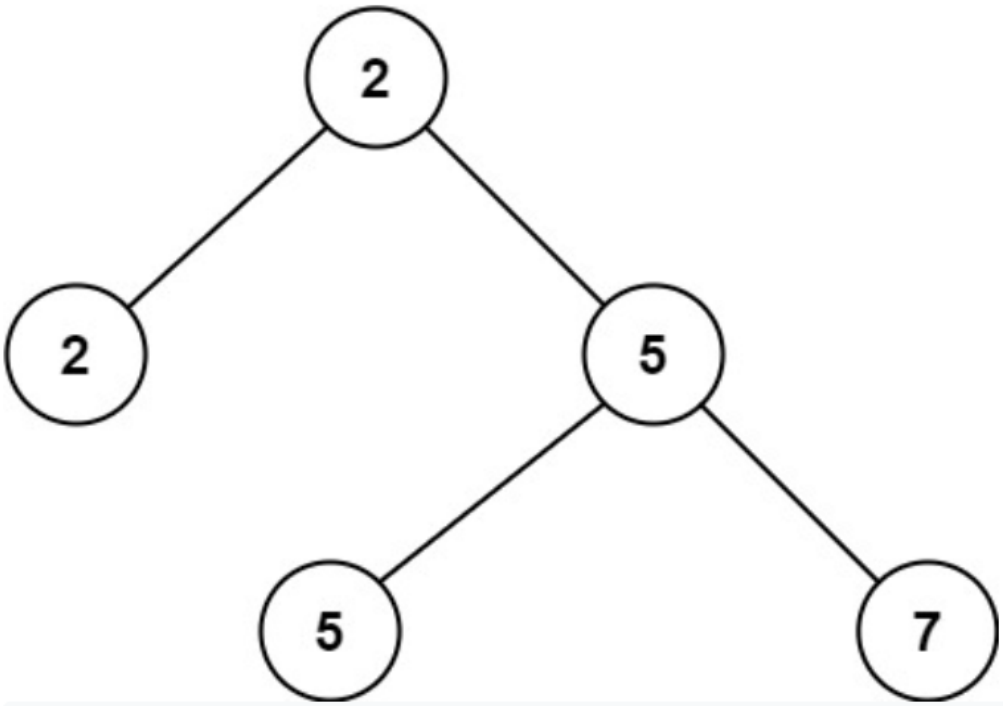


This is an uni-valued tree.

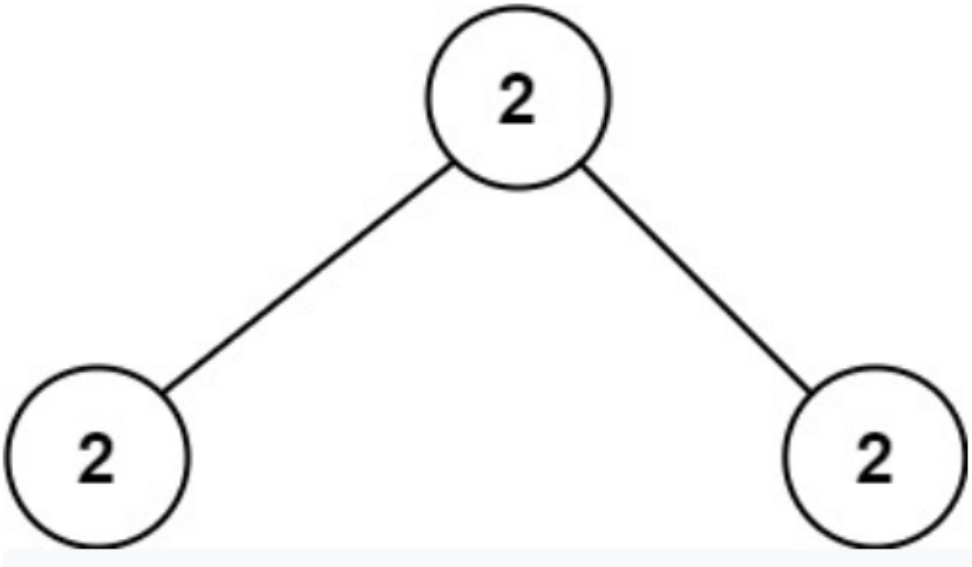


This is not an uni-valued tree.

10. Given a binary tree, you need to make a function named int second_minimum(Node* root) and return the second minimum value in all the nodes' value in the whole tree. If no such second minimum value exists, output -1 instead.
For example:



Output: 5



Output: -1