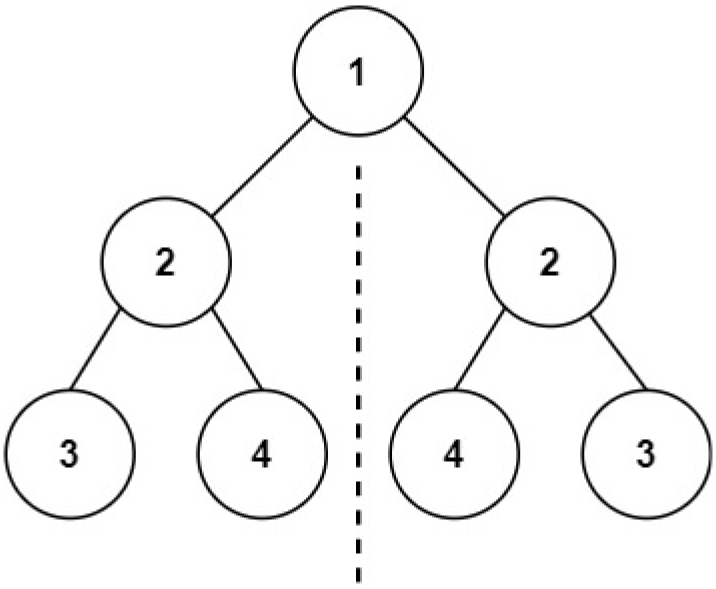
101. Symmetric Tree

Given the root of a binary tree, *check whether it is a mirror of itself* (i.e., symmetric around its center).

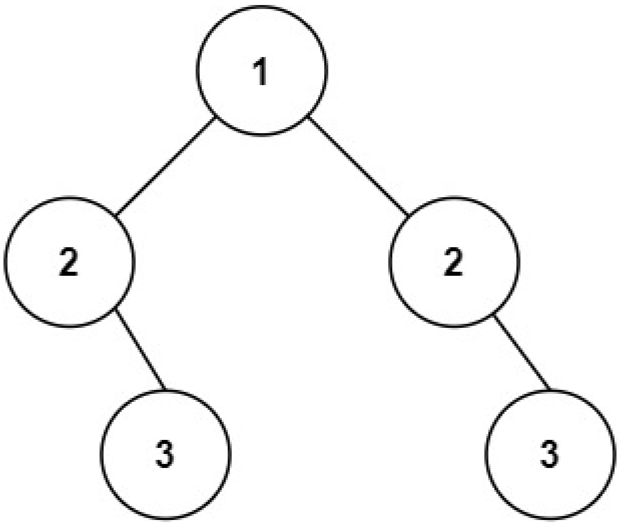
Example 1:



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

Output: true

Example 2:



Input: root = [1,2,2,null,3,null,3]

Output: false

Constraints:

- The number of nodes in the tree is in the range [1, 1000].
- -100 <= Node.val <= 100

```python
# Definition for a binary tree node.
# class TreeNode(object):
#     def __init__(self, val=0, left=None, right=None):
#         self.val = val
#         self.left = left
#         self.right = right
class Solution(object):
    def isSymmetric(self, root):
        """
        :type root: TreeNode
        :rtype: bool
        """
        if not root:
            return True

        stack1 = [root.left]
        stack2 = [root.right]

        while stack1 and stack2:
            left, right = stack1.pop(0),stack2.pop(0)
            if not left and not right:
                continue
            if not left or not right:
                return False
            if left.val != right.val:
                return False

            stack1.append(left.left)
            stack1.append(left.right)
            stack2.append(right.right)
            stack2.append(right.left)

        return True
```