LC 101. Symmetric Tree

Question

Given a binary tree, check whether it is a mirror of itself (ie, symmetric around its center).

For example, this binary tree `[1,2,2,3,4,4,3]` is symmetric:

```
    1
   / \
  2   2
 / \ / \
3  4 4  3
```

But the following `[1,2,2,null,3,null,3]` is not:

```
    1
   / \
  2   2
   \   \
   3    3
```

Solution

```
# Definition for a binary tree node.
# class TreeNode:
#     def __init__(self, x):
#         self.val = x
#         self.left = None
#         self.right = None

class Solution:
    def isSymmetric(self, root):
        """
        :type root: TreeNode
        :rtype: bool
        """
        #Solution
        if not root:
            return True
        stack = [(root.left, root.right)]
        while stack:
            n1, n2 = stack.pop()
            if not n1 and not n2:
                continue
```

```
            elif n1 and n2 and n1.val == n2.val:
                stack.append((n1.left, n2.right))
                stack.append((n1.right, n2.left))
            else:
                return False
        return True
```