```swift
import Foundation
/// **剑指 Offer 68 - II. 二叉树的最近公共祖先**
/// 给定一个二叉树, 找到该树中两个指定节点的最近公共祖先。
///
/// 百度百科中最近公共祖先的定义为：“对于有根树 T 的两个结点 p、q，
/// 最近公共祖先表示为一个结点 x，满足 x 是 p、q 的祖先且 x 的深度尽
/// 可能大（一个节点也可以是它自己的祖先）。”
///
/// 示例 1:
/// 输入: root = [3,5,1,6,2,0,8,null,null,7,4], p = 5, q = 1
/// 输出: 3
/// 解释: 节点 5 和节点 1 的最近公共祖先是节点 3。
class Solution {
    /// 寻找二叉树中两个节点的最近公共祖先
    func lowestCommonAncestor(_ root: TreeNode?, _ p: TreeNode, _ q: TreeNode)
     -> TreeNode? {
        /// 若当前节点为空或者当前节点为 p 或 q 中的一个，直接返回当前节点作为公共祖先
        if root == nil || root === p || root === q {
            return root
        }
        /// 在左子树中寻找 p 和 q 的最近公共祖先
        let left = lowestCommonAncestor(root?.left, p, q)
        /// 在右子树中寻找 p 和 q 的最近公共祖先
        let right = lowestCommonAncestor(root?.right, p, q)
        /// 若左子树中没有找到最近公共祖先，则返回右子树中找到的公共祖先
        if left == nil { return right }
        /// 若右子树中没有找到最近公共祖先，则返回左子树中找到的公共祖先
        if right == nil { return left }
        /// 若左右子树均找到了最近公共祖先，则当前节点为最近公共祖先
        return root
    }
}

public class TreeNode {
    public var val: Int
    public var left: TreeNode?
    public var right: TreeNode?

    public init() { self.val = 0; self.left = nil; self.right = nil; }
    public init(_ val: Int) { self.val = val; self.left = nil; self.right =
     nil; }
    public init(_ val: Int, _ left: TreeNode?, _ right: TreeNode?) {
        self.val = val
        self.left = left
        self.right = right
    }
}
```