Problem 145: Binary Tree Postorder Traversal

Problem Information

Difficulty: Easy
Acceptance Rate: 77.02%
Paid Only: No
Tags: Stack, Tree, Depth-First Search, Binary Tree

Problem Description

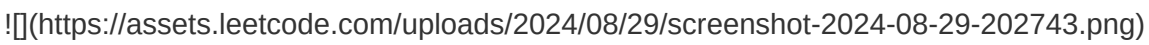
Given the `root` of a binary tree, return _the postorder traversal of its nodes ' values_.

Example 1:

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

Output: [3,2,1]

Explanation:

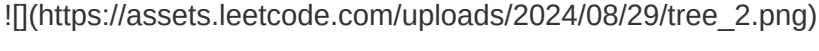


Example 2:

Input: root = [1,2,3,4,5,null,8,null,null,6,7,9]

Output: [4,6,7,5,2,9,8,3,1]

Explanation:



Example 3:

Input: root = []

Output: []

Example 4:

Input: root = [1]

Output: [1]

Constraints:

* The number of the nodes in the tree is in the range `[0, 100]`. * `-100 <= Node.val <= 100`

Follow up: Recursive solution is trivial, could you do it iteratively?

Code Snippets

C++:

```
/**
* Definition for a binary tree node.
* struct TreeNode {
* int val;
* TreeNode *left;
* TreeNode *right;
* TreeNode() : val(0), left(nullptr), right(nullptr) {}
* TreeNode(int x) : val(x), left(nullptr), right(nullptr) {}
* TreeNode(int x, TreeNode *left, TreeNode *right) : val(x), left(left),
right(right) {}
* };
*/
class Solution {
public:
vector<int> postorderTraversal(TreeNode* root) {

}
};
```

Java:

```
/**
* 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 List<Integer> postorderTraversal(TreeNode root) {

}
}
```

Python3:

```
# Definition for a binary tree node.
# class TreeNode:
# def __init__(self, val=0, left=None, right=None):
# self.val = val
# self.left = left
# self.right = right
class Solution:
def postorderTraversal(self, root: Optional[TreeNode]) -> List[int]:
```