

Problem 226: Invert Binary Tree

Problem Information

Difficulty: Easy

Acceptance Rate: 79.56%

Paid Only: No

Tags: Tree, Depth-First Search, Breadth-First Search, Binary Tree

Problem Description

Given the `root` of a binary tree, invert the tree, and return _its root_.

Example 1:

Input: root = [4,2,7,1,3,6,9] **Output:** [4,7,2,9,6,3,1]

Example 2:

Input: root = [2,1,3] **Output:** [2,3,1]

Example 3:

Input: root = [] **Output:** []

Constraints:

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

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:  
    TreeNode* invertTree(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 TreeNode invertTree(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 invertTree(self, root: Optional[TreeNode]) -> Optional[TreeNode]:
```