

Problem 897: Increasing Order Search Tree

Problem Information

Difficulty: Easy

Acceptance Rate: 78.86%

Paid Only: No

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

Problem Description

Given the `root` of a binary search tree, rearrange the tree in **in-order** so that the leftmost node in the tree is now the root of the tree, and every node has no left child and only one right child.

Example 1.

 (https://assets.leetcode.com/uploads/2020/11/17/ex1.jpg)

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

Example 2.

 (https://assets.leetcode.com/uploads/2020/11/17/ex2.jpg)

Input: root = [5,1,7] **Output:** [1,null,5,null,7]

Constraints:

* The number of nodes in the given tree will be in the range `[1, 100]`. * `0 <= Node.val <= 1000`

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