Problem 701: Insert into a Binary Search Tree

Problem Information

Difficulty: Medium
Acceptance Rate: 73.32%
Paid Only: No
Tags: Tree, Binary Search Tree, Binary Tree

Problem Description

You are given the `root` node of a binary search tree (BST) and a `value` to insert into the tree. Return _the root node of the BST after the insertion_. It is **guaranteed** that the new value does not exist in the original BST.

Notice that there may exist multiple valid ways for the insertion, as long as the tree remains a BST after insertion. You can return **any of them**.

Example 1:

Input: root = [4,2,7,1,3], val = 5 **Output:** [4,2,7,1,3,5] **Explanation:** Another accepted tree is:

Example 2:

Input: root = [40,20,60,10,30,50,70], val = 25 **Output:** [40,20,60,10,30,50,70,null,null,25]

Example 3:

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

Constraints:

* The number of nodes in the tree will be in the range `[0, 104]`. * `-108 <= Node.val <= 108` * All the values `Node.val` are **unique**. * `-108 <= val <= 108` * It's **guaranteed** that `val` does not exist in the original BST.

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* insertIntoBST(TreeNode* root, int val) {

}
};
```

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 insertIntoBST(TreeNode root, int val) {

}
}
```

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 insertIntoBST(self, root: Optional[TreeNode], val: int) ->
Optional[TreeNode]:
```