Problem 285: Inorder Successor in BST

Problem Information

Difficulty: Medium
Acceptance Rate: 51.03%
Paid Only: Yes
Tags: Tree, Depth-First Search, Binary Search Tree, Binary Tree

Problem Description

Given the `root` of a binary search tree and a node `p` in it, return _the in- order successor of that node in the BST_. If the given node has no in-order successor in the tree, return `null`.

The successor of a node `p` is the node with the smallest key greater than `p.val`.

Example 1:

Input: root = [2,1,3], p = 1 **Output:** 2 **Explanation:** 1's in-order successor node is 2. Note that both p and the return value is of TreeNode type.

Example 2:

Input: root = [5,3,6,2,4,null,null,1], p = 6 **Output:** null **Explanation:** There is no in-order successor of the current node, so the answer is null.

Constraints:

* The number of nodes in the tree is in the range `[1, 104]`. * `-105 <= Node.val <= 105` * All Nodes will have unique values.

Code Snippets

C++:

```
/**
* Definition for a binary tree node.
* struct TreeNode {
* int val;
* TreeNode *left;
* TreeNode *right;
* TreeNode(int x) : val(x), left(NULL), right(NULL) {}
* };
*/
class Solution {
public:
TreeNode* inorderSuccessor(TreeNode* root, TreeNode* p) {

}
};
```

Java:

```
/**
* Definition for a binary tree node.
* public class TreeNode {
* int val;
* TreeNode left;
* TreeNode right;
* TreeNode(int x) { val = x; }
* }
*/
class Solution {
public TreeNode inorderSuccessor(TreeNode root, TreeNode p) {

}
}
```

Python3:

```
# Definition for a binary tree node.
# class TreeNode:
# def __init__(self, x):
# self.val = x
```

```
# self.left = None
# self.right = None

class Solution:
def inorderSuccessor(self, root: TreeNode, p: TreeNode) ->
Optional[TreeNode]:
```