Problem 510: Inorder Successor in BST II

Problem Information

Difficulty: Medium
Acceptance Rate: 61.07%
Paid Only: Yes
Tags: Tree, Binary Search Tree, Binary Tree

Problem Description

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

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

You will have direct access to the node but not to the root of the tree. Each node will have a reference to its parent node. Below is the definition for `Node`:

class Node { public int val; public Node left; public Node right; public Node parent; }

Example 1:

![](https://assets.leetcode.com/uploads/2019/01/23/285_example_1.PNG)

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

Example 2:

![](https://assets.leetcode.com/uploads/2019/01/23/285_example_2.PNG)

Input: tree = [5,3,6,2,4,null,null,1], node = 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.

Follow up: Could you solve it without looking up any of the node's values?

Code Snippets

C++:

```
/*
// Definition for a Node.
class Node {
public:
int val;
Node* left;
Node* right;
Node* parent;
};
*/

class Solution {
public:
Node* inorderSuccessor(Node* node) {

}
};
```

Java:

```
/*
// Definition for a Node.
class Node {
public int val;
public Node left;
public Node right;
public Node parent;
};
*/

class Solution {
public Node inorderSuccessor(Node node) {
}
}
```

Python3:

```
"""
# Definition for a Node.
class Node:
def __init__(self, val):
self.val = val
self.left = None
self.right = None
self.parent = None
"""

class Solution:
def inorderSuccessor(self, node: 'Node') -> 'Optional[Node]':
```