Description | Solution | Discuss (237) | Submissions

510. Inorder Successor in BST II

Medium 583 29 Add to List Share

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;
}
```

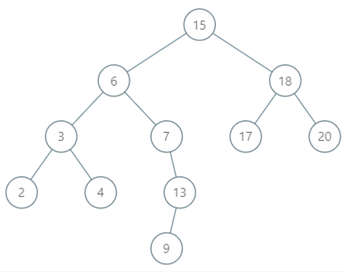
Example 1:


```
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:


```
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.
```

Example 3:




```
Input: tree = [15,6,18,3,7,17,20,2,4,null,13,null,null,null,null,null,null,null,null,9], node = 15
Output: 17
```

Example 4:


```
Input: tree = [15,6,18,3,7,17,20,2,4,null,13,null,null,null,null,null,null,null,null,9], node = 13
Output: 15
```

Example 5:

```
Input: tree = [0], node = 0
Output: null
```

Constraints:

- The number of nodes in the tree is in the range `[1, 10^4]`.
- `-10^5 <= Node.val <= 10^5`
- All Nodes will have unique values.

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

Accepted 40,980 | Submissions 66,628

Seen this question in a real interview before? Yes No

Companies

0 ~ 6 months 6 months ~ 1 year 1 year ~ 2 years

Microsoft | 3 Facebook | 2

Related Topics

Tree Binary Search Tree Binary Tree

Similar Questions

Inorder Successor in BST Medium

Java Autocomplete

```java
class Solution {
  public Node inorderSuccessor(Node x) {
    // the successor is somewhere lower in the right subtree
    if (x.right != null) {
      x = x.right;
      while (x.left != null) x = x.left;
      return x;
    }

    // the successor is somewhere upper in the tree
    while (x.parent != null && x == x.parent.right) x = x.parent;
    return x.parent;
  }
}
```