Problem 230: Kth Smallest Element in a BST

Problem Information

Difficulty: Medium
Acceptance Rate: 76.13%
Paid Only: No
Tags: Tree, Depth-First Search, Binary Search Tree, Binary Tree

Problem Description

Given the `root` of a binary search tree, and an integer `k`, return _the_ `kth` _smallest value (**1-indexed**) of all the values of the nodes in the tree_.

Example 1:

![](https://assets.leetcode.com/uploads/2021/01/28/kthtree1.jpg)

Input: root = [3,1,4,null,2], k = 1 **Output:** 1

Example 2:

![](https://assets.leetcode.com/uploads/2021/01/28/kthtree2.jpg)

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

Constraints:

* The number of nodes in the tree is `n`. * `1 <= k <= n <= 104` * `0 <= Node.val <= 104`

Follow up: If the BST is modified often (i.e., we can do insert and delete operations) and you need to find the kth smallest frequently, how would you optimize?

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:
int kthSmallest(TreeNode* root, int k) {

}
};
```

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 int kthSmallest(TreeNode root, int k) {

}
}
```

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 kthSmallest(self, root: Optional[TreeNode], k: int) -> int:
```