Problem 450: Delete Node in a BST

Problem Information

Difficulty: Medium
Acceptance Rate: 53.80%
Paid Only: No
Tags: Tree, Binary Search Tree, Binary Tree

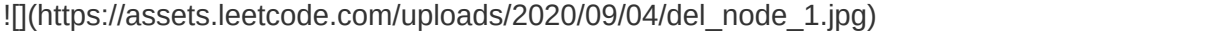
Problem Description

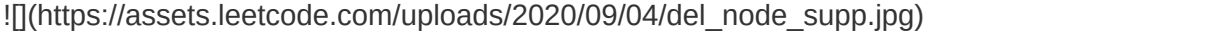
Given a root node reference of a BST and a key, delete the node with the given key in the BST. Return _the**root node reference** (possibly updated) of the BST_.

Basically, the deletion can be divided into two stages:

1. Search for a node to remove. 2. If the node is found, delete the node.

Example 1:



Input: root = [5,3,6,2,4,null,7], key = 3 **Output:** [5,4,6,2,null,null,7] **Explanation:** Given key to delete is 3. So we find the node with value 3 and delete it. One valid answer is [5,4,6,2,null,null,7], shown in the above BST. Please notice that another valid answer is [5,2,6,null,4,null,7] and it's also accepted.


Example 2:

Input: root = [5,3,6,2,4,null,7], key = 0 **Output:** [5,3,6,2,4,null,7] **Explanation:** The tree does not contain a node with value = 0.

Example 3:

Input: root = [], key = 0 **Output:** []

Constraints:

* The number of nodes in the tree is in the range `[0, 104]`. * `-105 <= Node.val <= 105` * Each node has a **unique** value. * `root` is a valid binary search tree. * `-105 <= key <= 105`

Follow up: Could you solve it with time complexity `O(height of tree)`?

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* deleteNode(TreeNode* root, int key) {

}
};
```

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 deleteNode(TreeNode root, int key) {

}
}
```

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 deleteNode(self, root: Optional[TreeNode], key: int) ->
Optional[TreeNode]:
```