

Problem 333: Largest BST Subtree

Problem Information

Difficulty: Medium

Acceptance Rate: 45.74%

Paid Only: Yes

Tags: Dynamic Programming, Tree, Depth-First Search, Binary Search Tree, Binary Tree

Problem Description

Given the root of a binary tree, find the largest subtree, which is also a Binary Search Tree (BST), where the largest means subtree has the largest number of nodes.

A **Binary Search Tree (BST)** is a tree in which all the nodes follow the below-mentioned properties:

- * The left subtree values are less than the value of their parent (root) node's value.
- * The right subtree values are greater than the value of their parent (root) node's value.

Note: A subtree must include all of its descendants.

Example 1:



Input: root = [10,5,15,1,8,null,7] **Output:** 3 **Explanation:** The Largest BST Subtree in this case is the highlighted one. The return value is the subtree's size, which is 3.

Example 2:

Input: root = [4,2,7,2,3,5,null,2,null,null,null,null,1] **Output:** 2

Constraints:

- * The number of nodes in the tree is in the range `[0, 104]`.
- * `-104 <= Node.val <= 104`

****Follow up:**** Can you figure out ways to solve it with $O(n)$ time complexity?

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 largestBSTSubtree(TreeNode* root) {

    }
};
```

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 largestBSTSubtree(TreeNode root) {  
  
}  
}
```

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