

Problem 998: Maximum Binary Tree II

Problem Information

Difficulty: Medium

Acceptance Rate: 69.97%

Paid Only: No

Tags: Tree, Binary Tree

Problem Description

A **maximum tree** is a tree where every node has a value greater than any other value in its subtree.

You are given the `root` of a maximum binary tree and an integer `val`.

Just as in the [previous problem](<https://leetcode.com/problems/maximum-binary-tree/>), the given tree was constructed from a list `a` (`root = Construct(a)`) recursively with the following `Construct(a)` routine:

* If `a` is empty, return `null`. * Otherwise, let `a[i]` be the largest element of `a`. Create a `root` node with the value `a[i]`. * The left child of `root` will be `Construct([a[0], a[1], ..., a[i - 1]])`. * The right child of `root` will be `Construct([a[i + 1], a[i + 2], ..., a[a.length - 1]])`. * Return `root`.

Note that we were not given `a` directly, only a root node `root = Construct(a)`.

Suppose `b` is a copy of `a` with the value `val` appended to it. It is guaranteed that `b` has unique values.

Return `Construct(b)`.

Example 1:

****Input:**** root = [4,1,3,null,null,2], val = 5 ****Output:**** [5,4,null,1,3,null,null,2] ****Explanation:**** a = [1,4,2,3], b = [1,4,2,3,5]

****Example 2:****

****Input:**** root = [5,2,4,null,1], val = 3 ****Output:**** [5,2,4,null,1,null,3] ****Explanation:**** a = [2,1,5,4], b = [2,1,5,4,3]

****Example 3:****

****Input:**** root = [5,2,3,null,1], val = 4 ****Output:**** [5,2,4,null,1,3] ****Explanation:**** a = [2,1,5,3], b = [2,1,5,3,4]

****Constraints:****

* The number of nodes in the tree is in the range `[1, 100]`. * `1 <= Node.val <= 100` * All the values of the tree are **unique**. * `1 <= val <= 100`

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* insertIntoMaxTree(TreeNode* root, int val) {  
    }  
};
```

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 insertIntoMaxTree(TreeNode root, int val) {  
        }  
}
```

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 insertIntoMaxTree(self, root: Optional[TreeNode], val: int) ->  
#         Optional[TreeNode]:
```