**538. Convert BST to Greater Tree**

Easy

132987FavoriteShare

Given a Binary Search Tree (BST), convert it to a Greater Tree such that every key of the original BST is changed to the original key plus sum of all keys greater than the original key in BST.

**Example:**

**Input:** The root of a Binary Search Tree like this:

5

/ \

2 13

**Output:** The root of a Greater Tree like this:

18

/ \

1. 13

首先这道题注意是一个BST，BST定义在这：<https://zh.wikipedia.org/wiki/%E4%BA%8C%E5%85%83%E6%90%9C%E5%B0%8B%E6%A8%B9>

总而言之，就是按中序遍历是一个升序数组。至于什么是中序请看题目144讲解。严重推荐在看这题讲解前，看一下题目144讲解。因为这题也会用到Morris算法

<https://t123456ll.github.io/LeetCode-144-Binary-Tree-Preorder-Traversal.html#more>

这里推荐一下我写的博客，如果大家乐意，可以没事来看看，帮我增加一下点击率（打个广告）。

Ok。首先这题的意思是给出一个二分查找树BTS，更改里面的节点值，使得更改后的节点值为原来的节点加上所有比它大的节点的值。比如例子中的节点5，因为只有节点13比它大，所以更改的值为5+13=18，而对于节点2，因为节点5和节点13都比它大，所以更改的值为2+5+13=20。二分查找树的特点是中序遍历的结果是一个升序序列。而我们做加法时最简单的顺序就是从最大的值开始，降序地做下去，这里就需要用到逆向中序遍历。然后正好之前研究了一下morris traversal，然后直觉告诉我把算法里的左右颠倒一下，似乎就可以达到逆向遍历的效果。所以这就是这道题的解法来历。

1. Morris Traversal

总结一下题目144里对Morris Traversal的介绍。首先此算法空间复杂度O(1)，时间复杂度O(n),但是不知道为什么用python写memory usage并不是最优的，可能这题用此算法不是最优的，但不妨碍我们用此算法解题，多练一练总是有好处的。

这一题的和144唯一不同的就是把morris 算法里所有的左枝换成右枝，右枝换成左枝。然后访问时的处理变成cur.val += Sum； Sum = cur.val就行了。

