Leetcode(python) 二叉搜索树 简单

530二叉搜索树的最小绝对差

题目：

给定一个所有节点为非负值的二叉搜索树，求树中任意两节点的差的绝对值的最小值。

**示例 :**

**输入:**

1

\

3

/

2

**输出:**

1

**解释:**

最小绝对差为1，其中 2 和 1 的差的绝对值为 1（或者 2 和 3）。

**注意:** 树中至少有2个节点。

思路：中序遍历将BST中的节点按照递增顺序输出

代码：

1. # Definition for a binary tree node.
2. # class TreeNode(object):
3. #     def \_\_init\_\_(self, x):
4. #         self.val = x
5. #         self.left = None
6. #         self.right = None
8. **class** Solution(object):
9. **def** getMinimumDifference(self, root):
10. """
11. :type root: TreeNode
12. :rtype: int
13. """
14. **if** root **is** None:
15. **return**
16. temp=[]
17. node=self.inorder(root,temp)
18. temp2=min([abs(node[i+1]-node[i]) **for** i **in** range(len(node)-1)])
19. **return** temp2
21. **def** inorder(self,root,temp):
22. **if** root==None:
23. **return**
24. self.inorder(root.left,temp)
25. temp.append(root.val)
26. self.inorder(root.right,temp)
27. **return** temp

783二叉搜索树结点最小距离（与530一样）

题目：

给定一个二叉搜索树的根结点 root, 返回树中任意两节点的差的最小值。

**示例：**

**输入:** root = [4,2,6,1,3,null,null]

**输出:** 1

**解释:**

注意，root是树结点对象(TreeNode object)，而不是数组。

给定的树 [4,2,6,1,3,null,null] 可表示为下图:

4

/ \

2 6

/ \

1 3

最小的差值是 1, 它是节点1和节点2的差值, 也是节点3和节点2的差值。

**注意：**

1. 二叉树的大小范围在 2 到 100。
2. 二叉树总是有效的，每个节点的值都是整数，且不重复。

思路：看见BST想中序遍历是有序的啊~所以先进性中序遍历，得到有序列表，然后找出相邻的两个节点差值的最小值即可

代码：

1. # Definition for a binary tree node.
2. # class TreeNode(object):
3. #     def \_\_init\_\_(self, x):
4. #         self.val = x
5. #         self.left = None
6. #         self.right = None
8. **class** Solution(object):
9. **def** minDiffInBST(self, root):
10. """
11. :type root: TreeNode
12. :rtype: int
13. """
14. vals = []
15. **def** inOrder(root):
16. **if** **not** root:
17. **return**
18. inOrder(root.left)
19. vals.append(root.val)
20. inOrder(root.right)
21. inOrder(root)
22. **return** min([vals[i + 1] - vals[i] **for** i **in** xrange(len(vals) - 1)])