530 Minimum Absolute Difference in BST

2018年4月13日 20:18

Given a binary search tree with non-negative values, find the minimum absolute difference between values of any two nodes.

Example:

Input:

```
1
  \
   3
  /
 2
```

Output:

1

Explanation:

The minimum absolute difference is 1, which is the difference between 2 and 1 (or between 2 and 3).

Note: There are at least two nodes in this BST.

来自 <https://leetcode.com/problems/minimum-absolute-difference-in-bst/description/>

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。

Solution for Python3:

```python
# Definition for a binary tree node.
# class TreeNode:
#     def __init__(self, x):
#         self.val = x
#         self.left = None
#         self.right = None

class Solution1:
    def getMinimumDifference(self, root):
        """
        :type root: TreeNode
        :rtype: int
        """
        nums = self.inOrder(root)
        return min(nums[i+1] - nums[i] for i in range(len(nums)-1))

    def inOrder(self, root):
       return self.inOrder(root.left) + [root.val] +
self.inOrder(root.right) if root else []


class Solution2:
    def getMinimumDifference(self, root):
        """
        :type root: TreeNode
        :rtype: int
        """
        self.minVal, self.preVal = float('inf'), -1
        return self.inOrder(root)

    def inOrder(self, root):
        if not root:
           return self.minVal
        self.inOrder(root.left)
        if self.preVal != -1:
           self.minVal = min(self.minVal, root.val - self.preVal)
        self.preVal = root.val
        self.inOrder(root.right)
        return self.minVal
```

Solution for C++:

```
/**
 * Definition for a binary tree node.
 * struct TreeNode {
 *     int val;
 *     TreeNode *left;
 *     TreeNode *right;
 *     TreeNode(int x) : val(x), left(NULL), right(NULL) {}
 * };
 */
class Solution {
public:
    int getMinimumDifference(TreeNode* root) {
        if (root == NULL)
            return minVal;
        getMinimumDifference(root->left);
        if (preVal != -1){
            minVal = min(minVal, root->val - preVal);
        }
        preVal = root->val;
        getMinimumDifference(root->right);
        return minVal;
    }
private:
    int minVal = INT_MAX, preVal = -1;
};
```