```
/*
给定二叉搜索树的根结点 root，返回 L 和 R（含）之间的所有结点的值的和。

二叉搜索树保证具有唯一的值。

示例 1：

输入：root = [10,5,15,3,7,null,18], L = 7, R = 15
输出：32

示例 2：

输入：root = [10,5,15,3,7,13,18,1,null,6], L = 6, R = 10
输出：23



提示：

    树中的结点数量最多为 10000 个。
    最终的答案保证小于 2^31。

来源：力扣（LeetCode）
链接：https://leetcode-cn.com/problems/range-sum-of-bst
著作权归领扣网络所有。商业转载请联系官方授权，非商业转载请注明出处。
*/
```

分析:

- 二叉搜索树本身满足一些性质:当前节点左侧的值不大于当前节点的值，当前节点右侧的值不小于当前节点的值，又题设中"二叉搜索树保证具有唯一的值。",所以性质中都取不到等号.
- 根据以上性质:可以使用递归方法先序遍历的方法遍历二叉搜索树中在[L,R]之间的元素，并累加返回即可.
- 同理使用栈结构辅助即可使用迭代方法遍历二叉搜索树.

方法一:C_递归遍历

```
/**
 * Definition for a binary tree node.
 * struct TreeNode {
 *     int val;
 *     struct TreeNode *left;
 *     struct TreeNode *right;
 * };
 */

int ret_val = 0;


void __preOrderTravel(struct TreeNode* node , int L , int R)
{
    if(node==NULL)
```

```
    {
        return ;
    }
    if(node->val < L)
    {
        __preOrderTravel(node->right,L,R);
    }
    else if(node->val > R)
    {
        __preOrderTravel(node->left,L,R);
    }
    else
    {
        ret_val += node->val;
        __preOrderTravel(node->left,L,R);
        __preOrderTravel(node->right,L,R);
    }

    return ;
}


int rangeSumBST(struct TreeNode* root, int L, int R)
{
    ret_val = 0;
    __preOrderTravel(root,L,R);
    return ret_val;

}

/*
执行结果:
通过
显示详情
执行用时 :128 ms, 在所有 C 提交中击败了54.15% 的用户
内存消耗 :43.8 MB, 在所有 C 提交中击败了100.00%的用户
*/
```

方法二:C++_借助栈迭代遍历

```

/**
 * 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 rangeSumBST(TreeNode* root, int L, int R)
        {
            stack<TreeNode*>    stn             ;
```

```
            int                 ret_val = 0     ;
            TreeNode*           temp    = NULL  ;
            if(root!=NULL)
            {
                stn.push(root);
                while(!stn.empty())
                {
                    temp = stn.top();
                    stn.pop();
                    if(temp == NULL)
                    {
                        continue;
                    }

                    if(temp->val < L )
                    {
                        stn.push(temp->right);
                    }
                    else if(temp->val > R)
                    {
                        stn.push(temp->left);
                    }
                    else
                    {
                        ret_val += temp->val;
                        stn.push(temp->right);
                        stn.push(temp->left);
                    }
                }
            }
            return ret_val;
        }
};


/*
执行结果:
通过
显示详情
执行用时 :276 ms, 在所有 C++ 提交中击败了28.21% 的用户
内存消耗 :41.1 MB, 在所有 C++ 提交中击败了91.02%的用户
*/



class Solution
{
    public:
        int rangeSumBST(TreeNode* root, int L, int R)
        {
            stack<TreeNode*>    stn             ;
            int                 ret_val = 0     ;
            TreeNode*           temp    = NULL  ;
            if(root!=NULL)
            {
                stn.push(root);
                while(!stn.empty())
                {
```

```
                    temp = stn.top();
                    stn.pop();

                    if(temp->val < L )
                    {
                        if(temp->right)
                        {
                            stn.push(temp->right);
                        }
                    }
                    else if(temp->val > R)
                    {
                        if(temp->left)
                        {
                            stn.push(temp->left);
                        }
                    }
                    else
                    {
                        ret_val += temp->val;
                        if(temp->right)
                        {
                            stn.push(temp->right);
                        }
                        if(temp->left)
                        {
                            stn.push(temp->left);
                        }
                    }
                }
            }
            return ret_val;
        }
};


/*
执行结果:
通过
显示详情
执行用时 :276 ms, 在所有 C++ 提交中击败了28.21% 的用户
内存消耗 :41.1 MB, 在所有 C++ 提交中击败了87.11%的用户
*/
```

AlimyBreak
2019.08.29