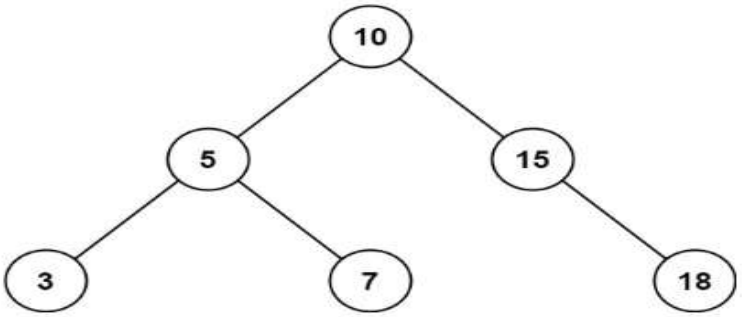
Given the `root` node of a binary search tree and two integers `low` and `high`, return *the sum of values of all nodes with a value in the* ***inclusive*** *range* `[low, high]`.

Example 1:




```
Input: root = [10,5,15,3,7,null,18], low = 7, high = 15
Output: 32
```

```
/**
 * 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 {
    int sum = 0;
public:
    int rangeSumBST(TreeNode* root, int low, int high) {
        if (root){
            rangeSumBST(root->left, low, high);
            if (root->val >= low && root->val <= high)
                sum += root->val;
            cout << root->val << " ";
            rangeSumBST(root->right, low, high);
        }
        return sum;
    }
};
```