

Problem 404: Sum of Left Leaves

Problem Information

Difficulty: Easy

Acceptance Rate: 62.15%

Paid Only: No

Tags: Tree, Depth-First Search, Breadth-First Search, Binary Tree

Problem Description

Given the `root` of a binary tree, return _the sum of all left leaves._

A **leaf** is a node with no children. A **left leaf** is a leaf that is the left child of another node.

Example 1:



Input: root = [3,9,20,null,null,15,7] **Output:** 24 **Explanation:** There are two left leaves in the binary tree, with values 9 and 15 respectively.

Example 2:

Input: root = [1] **Output:** 0

Constraints:

* The number of nodes in the tree is in the range `[1, 1000]`. * `-1000 <= Node.val <= 1000`

Code Snippets

C++:

```

/**
 * 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 {
public:
    int sumOfLeftLeaves(TreeNode* root) {
}
};


```

Java:

```

/**
 * Definition for a binary tree node.
 * public class TreeNode {
 *     int val;
 *     TreeNode left;
 *     TreeNode right;
 *     TreeNode() {}
 *     TreeNode(int val) { this.val = val; }
 *     TreeNode(int val, TreeNode left, TreeNode right) {
 *         this.val = val;
 *         this.left = left;
 *         this.right = right;
 *     }
 * }
 */
class Solution {
    public int sumOfLeftLeaves(TreeNode root) {
}
};


```

Python3:

```
# Definition for a binary tree node.
# class TreeNode:
#     def __init__(self, val=0, left=None, right=None):
#         self.val = val
#         self.left = left
#         self.right = right
class Solution:

    def sumOfLeftLeaves(self, root: Optional[TreeNode]) -> int:
```