Sum Of Left Leaves [LeetCode](https://leetcode.com/problems/sum-of-left-leaves/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:

5

/ \

3 7

/ \ \

11 1 6

/ \ / \

9 12 13 15

Output: The Sum of Left Leaves: 22

**Approach 1: Recursive function to calculate the sum of left leaves in a binary tree**

* In the **sumOfLeftLeavesHelper** function, you recursively calculate the sum of left leaves in the binary tree.
* For each node, you check if its left child is a leaf node (has no left or right children).
* If the left child is a leaf node, you add its value to the **sum** variable.
* You then recursively process the left and right subtrees.
* The final sum of left leaves is returned.

**Time Complexity:** **O(N), where N is the number of nodes in the binary tree. You visit each node once.**

**Space Complexity: O(H), where H is the height of the binary tree due to the function call stack.**

**Approach 2: Iterative function to calculate the sum of left leaves in a binary tree**

* In the **sumOfLeftLeavesIteratively** function, you calculate the sum of left leaves in the binary tree using an iterative approach with a stack.
* You start from the root and push nodes onto the stack while checking for left leaves.
* If a node has a left child that is a leaf node, you add its value to the **sum** variable.
* Otherwise, you continue processing the left and right children.
* The final sum of left leaves is returned.

**Time Complexity: O(N), where N is the number of nodes in the binary tree. You visit each node once.**

**Space Complexity: O(W), where W is the maximum width of the binary tree at any level.**