**Problem Statement**

**Title: Sum of Left Leaves**

**Problem Description:** Given the root of a binary tree, print 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.

**Input Format:**

* The first line contains an integer n, the number of nodes in the tree.
* The second line contains n nodes, space-separated, representing the tree in the following format: "root left-child right-child" of the current node.
* 'N' represents a null node.

**Constraints:**

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

**Output Format:**

* Print the sum of all left leaves.

**Sample Input:**

7

3 9 20 N N 15 7

**Sample Output:**

24

**Explanation:** There are two left leaves in the binary tree, with values 9 and 15 respectively.

**Extra Test Cases:**

**Test Case 1:**

Input:

7

3 9 20 N N 15 7

Output:

24

**Test Case 2:**

Input:

31

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Output:

184

**Test Case 3:**

Input:

11

3 5 1 6 2 0 8 N N 7 4

Output:

13

**Test Case 4:**

Input:

1

1

Output:

0

**Test Case 5:**

Input:

19

1 2 3 N 5 6 N 8 9 10 N 12 13 N 15 16 17 18 19

Output:

34

**Solution**

To solve this problem, we need to:

1. Parse the input string to construct the binary tree.
2. Traverse the tree to sum up all the left leaves.

Here's the Python implementation:

python

Copy code

class TreeNode:

def \_\_init\_\_(self, val=0, left=None, right=None):

self.val = val

self.left = left

self.right = right

def build\_tree(nodes):

if not nodes:

return None

val = nodes.pop(0)

root = TreeNode(int(val)) if val != 'N' else None

queue = [root]

while nodes:

current = queue.pop(0)

if current:

left\_val = nodes.pop(0)

right\_val = nodes.pop(0)

current.left = TreeNode(int(left\_val)) if left\_val != 'N' else None

current.right = TreeNode(int(right\_val)) if right\_val != 'N' else None

queue.append(current.left)

queue.append(current.right)

return root

def sum\_of\_left\_leaves(root):

if not root:

return 0

def is\_leaf(node):

return node and not node.left and not node.right

total\_sum = 0

stack = [root]

while stack:

node = stack.pop()

if node:

if is\_leaf(node.left):

total\_sum += node.left.val

if node.left:

stack.append(node.left)

if node.right:

stack.append(node.right)

return total\_sum

def main():

import sys

input = sys.stdin.read().strip().split()

n = int(input[0])

nodes = input[1:]

root = build\_tree(nodes)

result = sum\_of\_left\_leaves(root)

print(result)

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Explanation of the Code**

1. **TreeNode Class:** Defines the structure of a node in the binary tree.
2. **build\_tree Function:** Constructs the binary tree from the input list of nodes.
3. **sum\_of\_left\_leaves Function:** Traverses the binary tree and calculates the sum of all left leaves.
4. **main Function:** Reads the input, constructs the tree, and calculates and prints the sum of left leaves.