Documentation for "Sum Root to Leaf Numbers" Problem

Problem Description

You are only given the root of a binary tree containing digits from 0 to 9. Each root-to-leaf path in the tree represents a number. For example, the root-to-leaf path 1 -> 2 -> 3 represents the number 123. Your task is to return the total sum of all root-to-leaf numbers. Test cases are generated so that the answer will fit in a 32-bit integer. A leaf node is a node with no children.

Example 1:

Input: root = [1, 2, 3]

Output: 25

Explanation:

- The root-to-leaf path 1 -> 2 represents the number 12.
- The root-to-leaf path 1 -> 3 represents the number 13.
- Therefore, the total sum is 12 + 13 = 25.

Example 2:

Input: root = [4, 9, 0, 5, 1]

Output: 1026

Explanation:

- The root-to-leaf path 4 -> 9 -> 5 represents the number 495.
- The root-to-leaf path 4 -> 9 -> 1 represents the number 491.
- The root-to-leaf path 4 -> 0 represents the number 40.
- Therefore, the total sum is 495 + 491 + 40 = 1026.

Constraints

- The number of nodes in the tree is in the range [1, 1000].
- Each node's value is between 0 and 9 inclusive.
- The depth of the tree will not exceed 10.

Approach

1. **Depth-First Search (DFS):**
    - Use a helper function to perform a DFS traversal of the tree.
    - Maintain the current number being formed by the path from the root to the current node.
    - For each node, update the current number by appending the node's value to the right.
    - If a leaf node is reached, return the current number.
    - Recur for both the left and right subtrees and sum their values.

2. **Initialization:**
    - Start the DFS traversal with the initial number set to 0.

3. **Edge Cases:**
    - Handle cases where the tree has only one node or is skewed.

This approach ensures that we traverse all possible root-to-leaf paths, accumulate their corresponding numbers, and return the total sum.