**Problem Name:** Binary tree paths

**Topics:**

**Companies:**

**Level:** Hard

**Language:** C++

**Problem Statement:**

**Input Format:**

**Output Format:**

**Constraints:**

**Examples:**

**Approach one Solution:**

**Explanation:** The vector res will accumulate our results, while the main function will basically just call dfs, the core of our logic, and return the updated value once said function has finished running its course.

In dfs we have 2 terminal conditions: root == NULL, for which we just exit, since it means the initial tree had zero elements or we sent our function to try a left or right branch which is just not there, and when !root->left && !root->right, meaning the current node is a leaf, in which case we update res before leaving.

In all the other cases, we call dfs recursively on both left and right, going another level down the tree.

This problem is rather similar to at least 3-4 other tree-traversal problems here on LC, with the slight added inconvenience of having to interspace all the explored nodes' values with "->"; in order not to be distracted by that and/or to keep my code a bit more DRY, I put that logic into nextCur.

**Code:**

**Time Complexity**: O(N)

**Space Complexity:** O(height of tree)

**Approach two Solution:**

**Explanation:** The basic idea is to traverse the tree keeping the path in the string **curr** and when you find a leaf then substituting curr in the answer.We need vector **answer** to be same in all the calls in the call stack, therefore we use address operator while we need separate curr for each step,hence not using address operator.

**Code:**

**Time Complexity**: O(N)

**Space Complexity:** O(height of tree)