Inorder Traversal of Binary Tree LeetCode

In-order traversal: Left subtree, current node, right subtree

Example:

```
    5
   / \
  3   7
 / \   \
1   4   9
```

Output: **[1, 3, 4, 5, 7, 9]**

Approach 1: Perform an in-order traversal of the binary tree using recursion

- The **solve** function is a recursive helper function that performs an in-order traversal.
- It follows the order: Left subtree, current node, Right subtree.
- The values of visited nodes are appended to the **ans** vector.
- The **inOrderTraversalRecursive** function initializes the **ans** vector and calls **solve** to perform the traversal.
- **Time Complexity: O(N), where N is the number of nodes in the binary tree.**
- **Space Complexity: O(N), as it uses additional space for the ans vector and the recursion call stack.**

Approach 2: Perform an in-order traversal of the binary tree using an iterative approach

- The **inOrderTraversalIterative** function uses an iterative approach with a stack to mimic the recursive in-order traversal.
- It starts at the root and traverses left as deep as possible while pushing nodes onto the stack.
- When a node with no left child is reached, it processes the node, pops it from the stack, and then moves to the right child.
- **Time Complexity: O(N), where N is the number of nodes in the binary tree.**
- **Space Complexity: O(H), where H is the height of the binary tree (H ≤ N), due to the stack space used for maintaining the traversal path.**