Validate Binary Search Tree LeetCode

Given the root of a binary tree, *determine if it is a valid binary search tree (BST)*.

A **valid BST** is defined as follows:

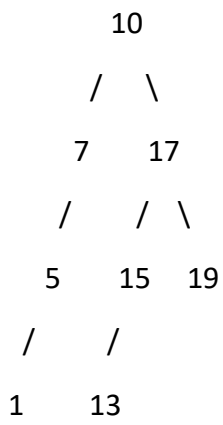
- The left

subtree

of a node contains only nodes with keys **less than** the node's key.

- The right subtree of a node contains only nodes with keys **greater than** the node's key.
- Both the left and right subtrees must also be binary search trees.

Example:



Output: The BST is Valid

Approach 1: Function to check if a binary tree is a valid binary search tree

- **Function Purpose:** This approach is used to check if a binary tree is a valid binary search tree.
- **Explanation:**
    1. The function performs an inorder traversal of the tree and stores the values in a vector.
    2. It then checks if the values in the vector are in ascending order.
- **Time Complexity: The time complexity of this approach is O(N), where N is the number of nodes in the tree, as it needs to visit all nodes.**
- **Space Complexity: The space complexity is O(N) due to the vector used to store the inorder traversal values.**

Approach 2: Function to check if a binary tree is a valid binary search tree

- **Function Purpose:** This approach is used to check if a binary tree is a valid binary search tree.
- **Explanation:**
    1. The function is a recursive approach that validates each node in the tree while keeping track of a valid range for each node.
    2. It checks if the value of the current node is within the valid range defined by its ancestors.
    3. It recursively checks the left and right subtrees while updating the valid range for each subtree.
- **Time Complexity: The time complexity of this approach is O(N), where N is the number of nodes in the tree, as it needs to visit all nodes.**
- **Space Complexity: The space complexity is O(H), where H is the height of the tree. In the worst case, it's O(N) for a skewed tree.**

Conclusion:

- Approach 1, which uses an inorder traversal to validate the BST, is straightforward and works for both balanced and skewed trees.
- Approach 2, which uses a recursive approach with a valid range, is also effective and may perform better for balanced trees.
- The choice between the two approaches depends on the specific requirements and characteristics of the tree being validated. Both approaches are valid and can accurately determine the validity of a BST.