### Programming Guide for “Trees”

#### 1. Insert into BST

**Overview:** Insert a new value into a Binary Search Tree (BST).

**Pseudo Code:**

FUNCTION insert\_into\_bst(root, value)  
 IF root is None  
 CREATE a new TreeNode with value  
 RETURN the new node  
 IF value < root.value  
 RECURSIVELY insert into the left subtree  
 ELSE  
 RECURSIVELY insert into the right subtree  
 RETURN root  
END FUNCTION

**Implementation Guide:** - Start by checking if the current node (root) is None. If so, create a new TreeNode with the given value. - If the value is less than the current node’s value, recursively insert it into the left subtree. - Otherwise, insert it into the right subtree. - Finally, return the root to maintain the tree structure.

#### 2. Validate BST

**Overview:** Determine if a given binary tree is a valid BST.

**Pseudo Code:**

FUNCTION is\_valid\_bst(root, lower\_bound, upper\_bound)  
 IF root is None  
 RETURN True  
 IF root.value is not between lower\_bound and upper\_bound  
 RETURN False  
 RETURN is\_valid\_bst(root.left, lower\_bound, root.value) AND  
 is\_valid\_bst(root.right, root.value, upper\_bound)  
END FUNCTION

**Implementation Guide:** - A BST is valid if, for every node, all nodes in its left subtree are smaller and all nodes in its right subtree are larger. - Check if the current node’s value falls within the allowed range (initially -infinity to +infinity). - Recursively validate the left and right subtrees, updating the bounds accordingly. - An empty tree or reaching a leaf node indicates a valid path, so return True in such cases.

#### 3. Inorder Traversal

**Overview:** Perform an in-order traversal of a binary tree and return the values in a sequential order.

**Pseudo Code:**

FUNCTION inorder\_traversal(root)  
 IF root is None  
 RETURN an empty list  
 RETURN inorder\_traversal(root.left) + [root.value] + inorder\_traversal(root.right)  
END FUNCTION

**Implementation Guide:** - In-order traversal visits the left subtree, then the root node, and finally the right subtree. - If the current node is None, return an empty list. - Concatenate and return the list from the left subtree, the current node’s value, and the list from the right subtree.