**Name:** Safyan Anwar

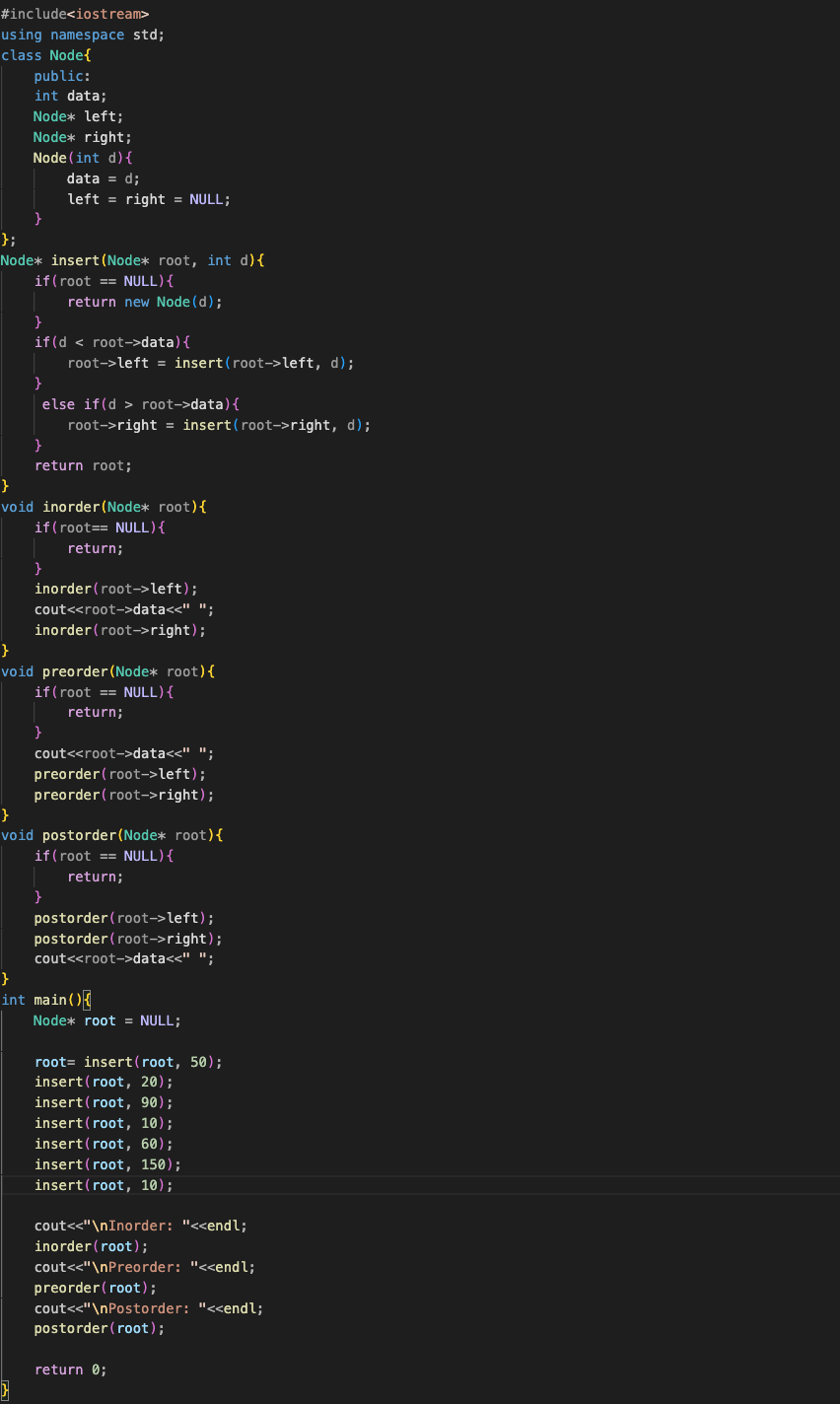
**Roll no:** -018

**Section:** 3A-BSSE

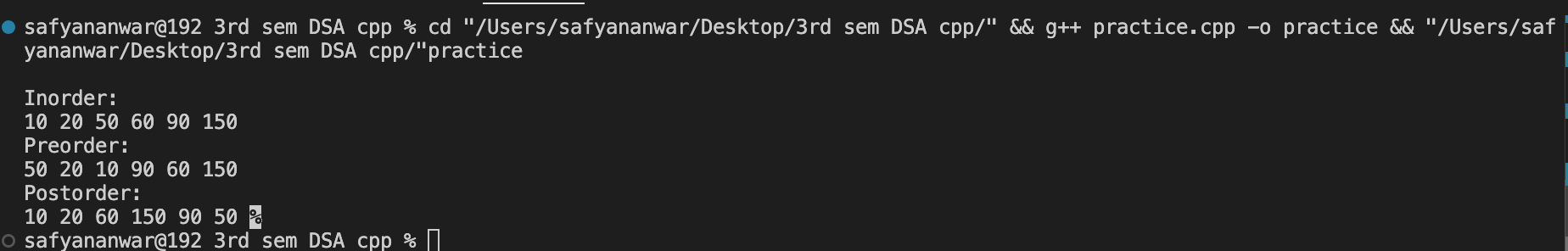
**LAB TASK:**

**Q#1:**

**Code:**

****

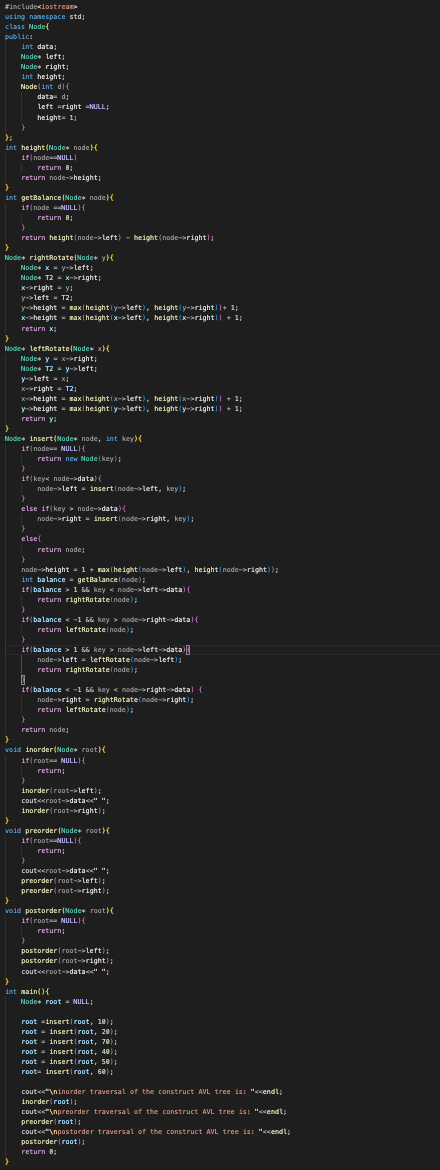
**Output:**

****

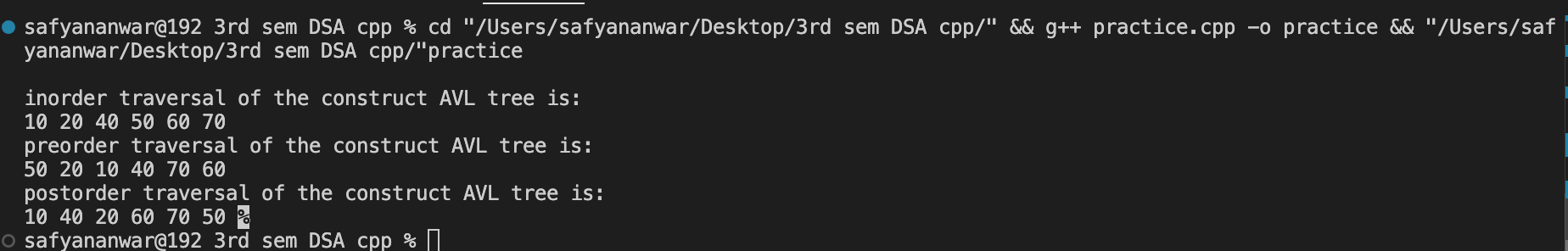
**Explanation:**

1. Node class:
   * Represent:
     + data: The value stored in the node.
     + left: Pointer to the left child.
     + right: Pointer to the right child.
   * Constructor initializes data and sets left and right to NULL.
2. Insert function:
   * Node: Insert a new value data into the BST.
   * If the root is null it creates a new node with the value data.
   * If data is less than the current node's data it recursively inserts into the left subtree.
   * If data is greater it inserting into the right subtree.
   * Return the update root of the tree.
3. Traversal functions:
   * Inorder: Visit node in ascending order.
   * Preorder:Visit node in the order root, left, right.
   * Postorder: Visit node in the order.
4. Main function:
   * Initialize the root of the bst to null.
   * Insert several values into the bst.
   * Call the traversal functions:
     + Inorder: Display values in sorted order.
     + Preorder: Display values starting from the root.
     + Postorder: Displays values after visiting children.

**Q2: code:**

****

**Output:**

****

**Explanation:**

1. Node class:
   * Represent.
     + data: The value stored in the node.
     + left and right: Pointers to the left and right children.
     + height: The height of the node initialized to 1.
2. Height function:
   * Returns the height of a given node. If the node is null it return 0.
3. Balance function:
   * Calculate the balance factor of a node which is the difference between the height of the left and right subtrees.
4. Rotation level:
   * Right: Perform a right rotation on a subtree to maintain balance.
   * Left: Perform a left rotation on a subtree to maintain balance.
5. Insert function:
   * Insert a new key into the AVL Tree.
   * After insertion it updates the height of the node and checks the balance factor.
   * If the tree becomes unbalance it performs the necessary rotations to restore balance.
6. Traversal functions:
   * Inorder: Visit nodes in ascending order i.e; left, root, right.
   * Preorder: Visit nodes in the order root i.e; left, right.
   * Postorder: Visit nodes in the order i.e; left, right, root.
7. Main function:
   * Initializes the root of the AVL Tree to null.
   * Inserts several values into the AVL Tree.
   * Calls the traversal functions to print the nodes in different order:
     + Inorder: Display value in sorted order.
     + Preorder: Display value starting from the root.
     + Postorder: Display value after visiting children.