Weekly Lab

Binary Search Tree

In this lab session, you will implement a classic data structure: the **Binary Search Tree (BST)**. All code must be written in a single Python file named `StudentID.py`.

Each node in the BST should be defined with the following structure:

```python
class Node:
    def __init__(self, key: int):
        self.key = key
        self.left = None
        self.right = None
```

Exercise 1: Basic BST Operations

Implement the following basic functions for a Binary Search Tree:

1. Create a new node from a given value.

   `def new_node(data: int) -> Node`

2. Insert a new value into the BST.

   `def insert(root: Node, data: int) -> Node`

3. Search for a node with a given value. Return `None` if not found.

   `def search(root: Node, data: int) -> Node`

4. Delete a node with a given value from the BST.

   `def delete(root: Node, data: int) -> Node`

5. Tree traversal methods:

   - Pre-order: `def preorder(root: Node) -> None`
   - In-order: `def inorder(root: Node) -> None`
   - Post-order: `def postorder(root: Node) -> None`
   - Level-order (BFS): `def level_order(root: Node) -> None`

Exercise 2: Additional BST Problems

Using the BST implementation from Exercise 1, solve the following problems:

1. Compute the height of the BST.

   ```
   def height(root: Node) -> int
   ```

2. Count the total number of nodes in the BST.

   ```
   def count_nodes(root: Node) -> int
   ```

3. Compute the sum of all node values in the BST.

   ```
   def sum_nodes(root: Node) -> int
   ```

4. Count the number of leaf nodes (nodes with no children).

   ```
   def count_leaves(root: Node) -> int
   ```

5. Count the number of nodes with values less than a given number `x`.

   ```
   def count_less(root: Node, x: int) -> int
   ```

6. Count the number of nodes with values greater than a given number `x`.

   ```
   def count_greater(root: Node, x: int) -> int
   ```

Note:
You are free to define any helper functions if necessary.
You may test your code in a `main` function with hardcoded examples. Test example:

- Initialize an empty BST Tree.
- Insert 8, 6, 5, 7, 10, 9.
- Show the tree in Pre-order, In-order, Post-order.
- Remove 8. Show the tree in Level Order.
- Solve the problems in Exercise 2. Test with x = 7 (if the problem requires key).

The end.