Student ID: 202112047

IT623 - Lab Assignment 6

1. **Create a binary tree as shown in the above figure.**
2. **Print In-order traversal of the above tree.**
3. **Print Pre-order traversal of the above tree.**
4. **Print Post-order traversal of the above tree.**

Code:

```
class Node {
	int key;
	Node left, right;

	Node(int item) {
		key = item;
		left = right = null;
	}
}

public class Program1 {
	Node root;

	void printPostorder(Node node) {
		if (node == null)
			return;

		printPostorder(node.left);
		printPostorder(node.right);
		System.out.print(node.key + " ");
	}
```

```
void printInorder(Node node) {
        if (node == null)
                return;

        printInorder(node.left);
        System.out.print(node.key + " ");
        printInorder(node.right);
}

void printPreorder(Node node) {
        if (node == null)
                return;

        System.out.print(node.key + " ");
        printPreorder(node.left);
        printPreorder(node.right);
}

public static void main(String[] args) {
        Program1 p = new Program1();

        p.root = new Node(1);

        p.root.left = new Node(2);
        p.root.right = new Node(3);

        p.root.left.left = new Node(4);
        p.root.left.right = new Node(5);
        p.root.right.left = new Node(6);
        p.root.right.right = new Node(7);

        p.root.left.left.left = new Node(8);
        p.root.left.left.right = new Node(9);
```

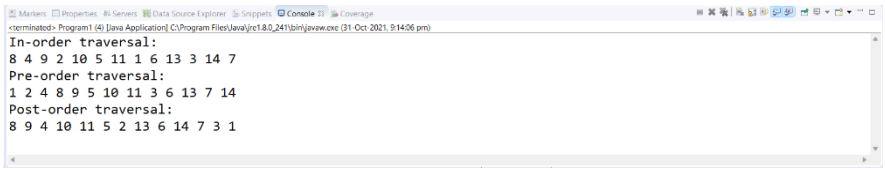
```
        p.root.left.right.left = new Node(10);
        p.root.left.right.right = new Node(11);
        p.root.right.left.right = new Node(13);
        p.root.right.right.left = new Node(14);

        System.out.println("In-order traversal: ");
        p.printInorder(p.root);

        System.out.println("\nPre-order traversal: ");
        p.printPreorder(p.root);

        System.out.println("\nPost-order traversal: ");
        p.printPostorder(p.root);
    }
}
```

Output Snapshot:



5. Sum of leaf nodes at each horizontal level in a binary tree.

Code:

```
import java.util.HashMap;
import java.util.LinkedList;
```

```
import java.util.Map;
import java.util.Queue;

public class Program5 {
	Node root = null;

	static class Pair {
		Node n;
		int i;

		Pair(Node n, int i) {
			this.n = n;
			this.i = i;
		}
	}

	static void printInorder(Node node) {
		if (node == null)
			return;

		printInorder(node.left);
		System.out.print(node.key + " ");
		printInorder(node.right);
	}

	static void printLevelSum(Node root) {
		if (root == null) {
			System.out.println("No node");
			return;
		}

		HashMap<Integer, Integer> map = new HashMap<>();

		Queue<Pair> q = new LinkedList<Pair>();
```

```
        q.add(new Pair(root, 1));

        Pair p;

        while (!q.isEmpty()) {
            p = q.peek();
            q.remove();

            if (!map.containsKey(p.i)) {
                map.put(p.i, 0);
            }

            if (p.n.left == null && p.n.right == null) {
                map.put(p.i, map.get(p.i) + p.n.key);
            }

            if (p.n.left != null) {
                q.add(new Pair(p.n.left, p.i + 1));
            }

            if (p.n.right != null) {
                q.add(new Pair(p.n.right, p.i + 1));
            }
        }

        int j = 0;
        for (Map.Entry<Integer,Integer> mapElement : map.entrySet()) {
            int value = ((int) mapElement.getValue());
            j++;
            System.out.println("Level " + j + " is " + value);
        }
}
```

```
    public static void main(String args[]) {
        Program5 p5 = new Program5();

        p5.root = new Node(1);

        p5.root.left = new Node(2);
        p5.root.right = new Node(3);

        p5.root.left.left = new Node(4);
        p5.root.left.right = new Node(5);
        p5.root.right.left = new Node(6);
        p5.root.right.right = new Node(7);

        p5.root.left.left.right = new Node(8);
        p5.root.left.right.right = new Node(9);
        p5.root.right.right.left = new Node(10);
        p5.root.right.right.right = new Node(11);

        p5.root.left.left.right.right = new Node(12);

        System.out.println("Sum of leaf nodes at each horizontal level : ");
        printLevelSum(p5.root);
    }
}
```

Output Snapshot: