Activity 8.2

Ramzy Izza Wardhana - 21/472698/PA/20322

BinaryTreeNode.java

```java
public class BinaryTreeNode {
    String label;
    BinaryTreeNode left;
    BinaryTreeNode right;

    BinaryTreeNode(String label, BinaryTreeNode left, BinaryTreeNode right) {
        this.label = label;
        this.left = left;
        this.right = right;
    }

    public String toString() {
        return "[" + label + ", " + left + ", " + right + "]";
    }
}
```

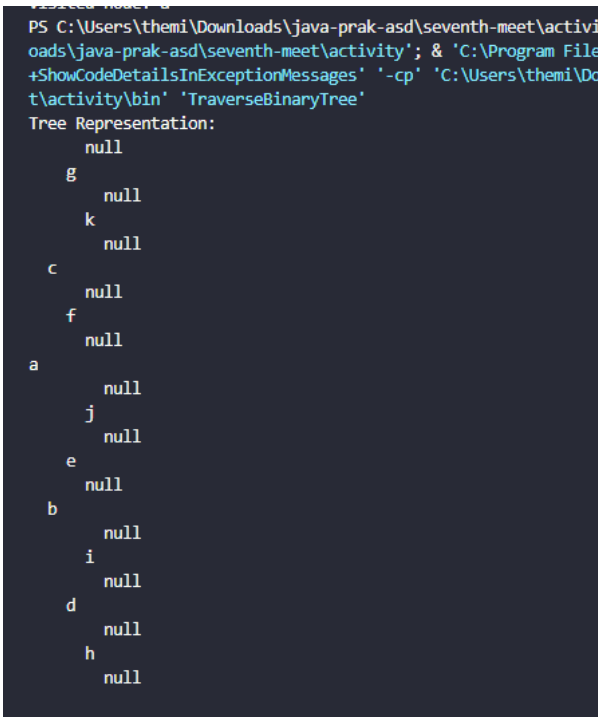
TraverseBinaryTree.java

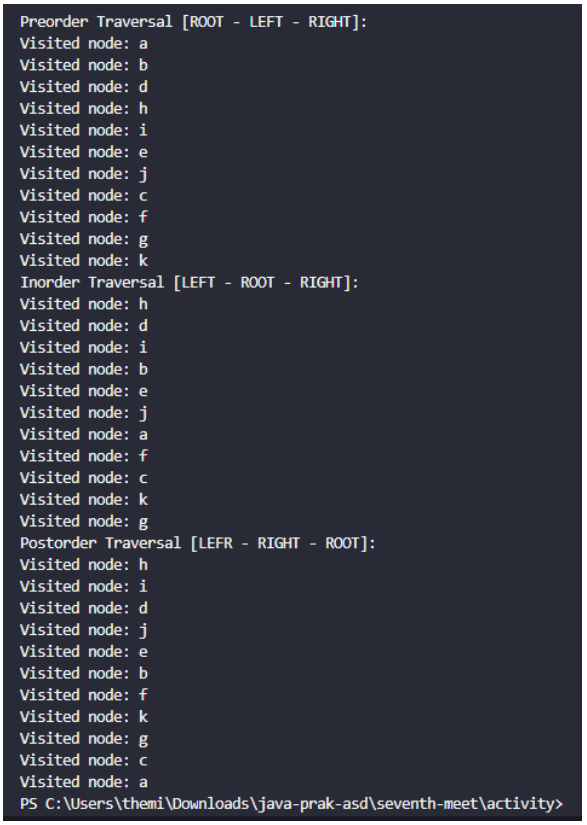
```java
public class TraverseBinaryTree {
    public static void traversePreorder(BinaryTreeNode t) {
        if(t == null) return;
        System.out.println("Visited node: " + t.label);
        traversePreorder(t.left);
        traversePreorder(t.right);
    }

    public static void traverseInorder(BinaryTreeNode t) {
        if (t == null) return;
        traverseInorder(t.left);
        System.out.println("Visited node: " + t.label);
        traverseInorder(t.right);
    }

    public static void traversePostorder(BinaryTreeNode t) {
        if (t  == null) return;
        traversePostorder(t.left);
        traversePostorder(t.right);
        System.out.println("Visited node: " + t.label);
    }
```

```
public static  String getSpaces(int n) {
    String s = "";
    for(int i = 0; i < n; i++)
        s += " ";
    return s;
}

public static String prettyPrint(BinaryTreeNode node) {
    return prettyPrint(0, node);
}

public static String prettyPrint(int n, BinaryTreeNode node) {
    if (node == null)
        return getSpaces(n) + "null\n";
    String s = "";
    s += prettyPrint(n+2, node.right);
    s += getSpaces(n) + node.label + "\n";
    s += prettyPrint(n+2, node.left);
    return s;
}

//Activity
public static void main(String[] args) {
    BinaryTreeNode tree, p , q, r;
    p = new BinaryTreeNode("h", null, null);
    q = new BinaryTreeNode("i", null, null);
    p = new BinaryTreeNode("d", p, q);
    q = new BinaryTreeNode("j", null, null);
    q = new BinaryTreeNode("e", null, q);
    r = new BinaryTreeNode("b", p, q);

    p = new BinaryTreeNode("f", null, null);
    q = new BinaryTreeNode("k", null, null);
    q = new BinaryTreeNode("g", q, null);
    p = new BinaryTreeNode("c", p, q);

    tree = new BinaryTreeNode("a", r, p);


    System.out.println("Tree Representation: \n" + prettyPrint(tree));
    System.out.print("Preorder Traversal [ROOT - LEFT - RIGHT]: ");
```

```
        traversePreorder(tree);
        System.out.print("Inorder Traversal [LEFT - ROOT - RIGHT]: ");
        traverseInorder(tree);
        System.out.print("Postorder Traversal [LEFR - RIGHT - ROOT]: ");
        traversePostorder(tree);
    }
}
```

Output and Evaluation





Answer:

Notice both screenshots given above and below, that the result of tree representation drawn by the source code and also three different traversals method (pre-order, in-order, post-order) yield the same result, implying that both have the same tree structure.

Manual Drawn Tree (Activity 8.1)