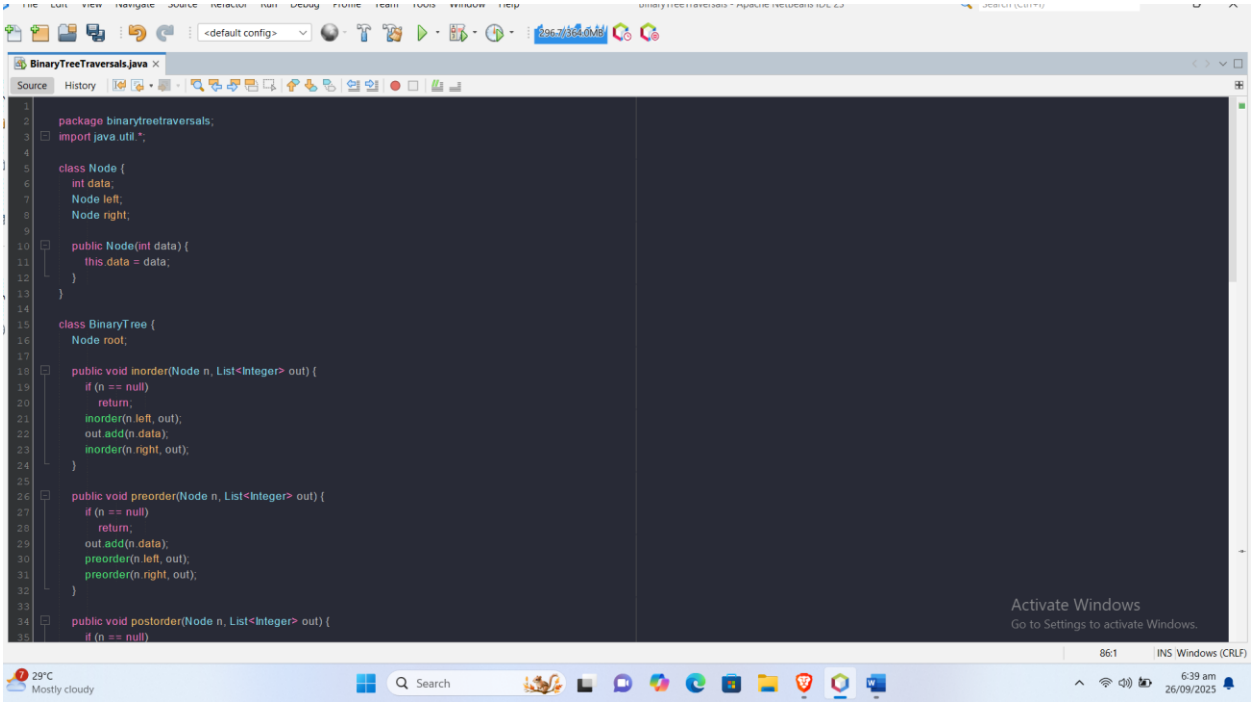
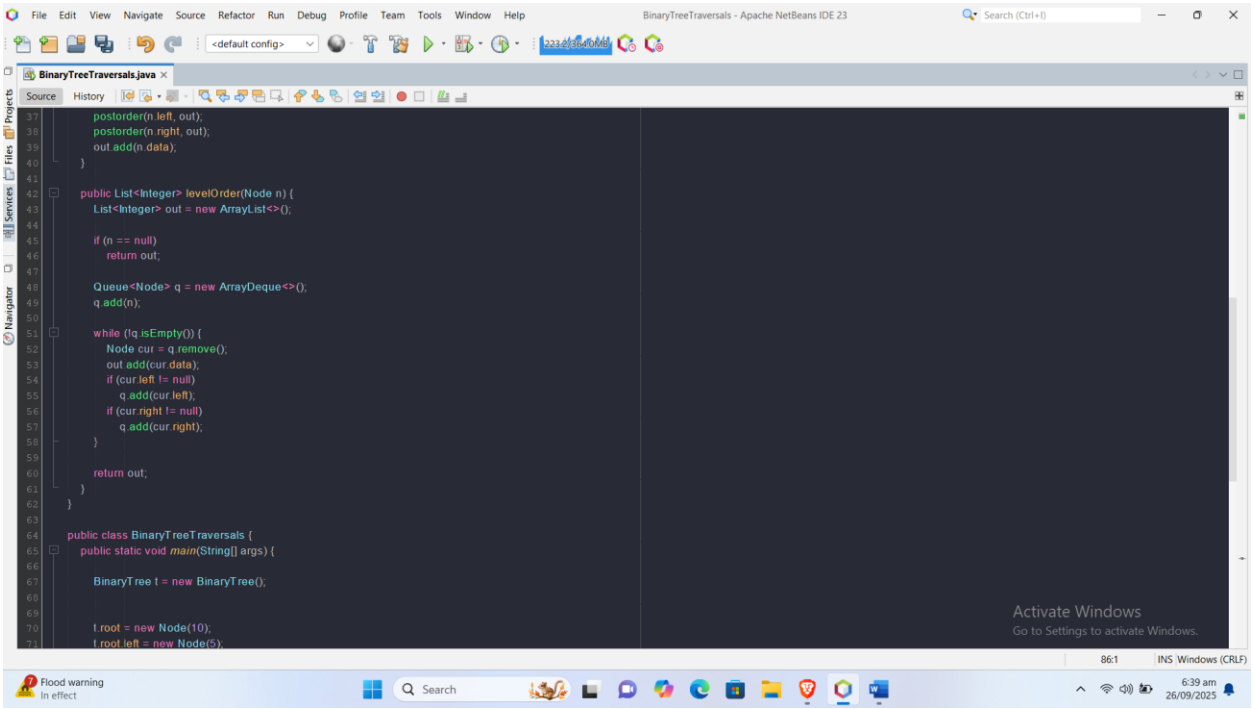
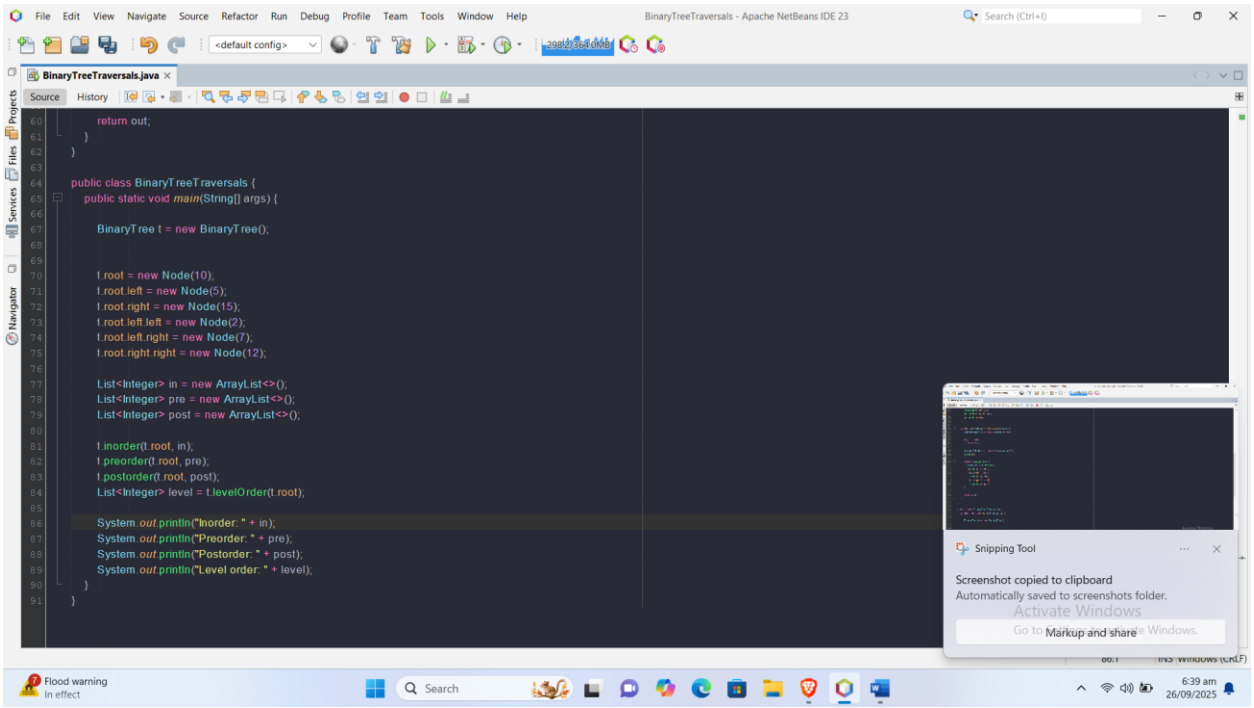
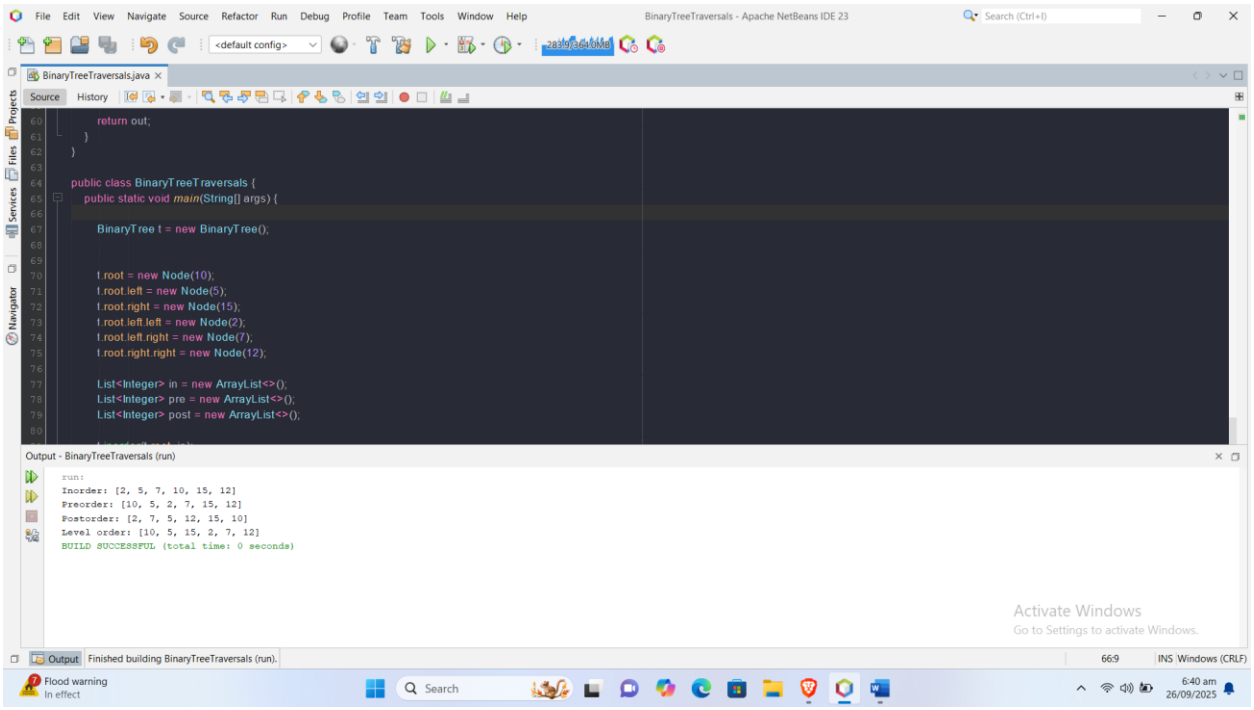


```java
package binarytreetraversals;
import java.util.*;

class Node {
    int data;
    Node left;
    Node right;

    public Node(int data) {
        this.data = data;
    }
}

class BinaryTree {
    Node root;

    public void inorder(Node n, List<Integer> out) {
        if (n == null)
            return;
        inorder(n.left, out);
        out.add(n.data);
        inorder(n.right, out);
    }

    public void preorder(Node n, List<Integer> out) {
        if (n == null)
            return;
        out.add(n.data);
        preorder(n.left, out);
        preorder(n.right, out);
    }

    public void postorder(Node n, List<Integer> out) {
        if (n == null)
```



```java
        postorder(n.left, out);
        postorder(n.right, out);
        out.add(n.data);
    }

    public List<Integer> levelOrder(Node n) {
        List<Integer> out = new ArrayList<>();

        if (n == null)
            return out;

        Queue<Node> q = new ArrayDeque<>();
        q.add(n);

        while (!q.isEmpty()) {
            Node cur = q.remove();
            out.add(cur.data);
            if (cur.left != null)
                q.add(cur.left);
            if (cur.right != null)
                q.add(cur.right);
        }

        return out;
    }
}

public class BinaryTreeTraversals {
    public static void main(String[] args) {

        BinaryTree t = new BinaryTree();


        t.root = new Node(10);
        t.root.left = new Node(5);
```


File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
BinaryTreeTraversals - Apache NetBeans IDE 23
BinaryTreeTraversals.java
Source History
return out;
}
}
public class BinaryTreeTraversals {
public static void main(String[] args) {
BinaryTree t = new BinaryTree();
t.root = new Node(10);
t.root.left = new Node(5);
t.root.right = new Node(15);
t.root.left.left = new Node(2);
t.root.left.right = new Node(7);
t.root.right.right = new Node(12);
List<Integer> in = new ArrayList<>();
List<Integer> pre = new ArrayList<>();
List<Integer> post = new ArrayList<>();
t.inorder(t.root, in);
t.preorder(t.root, pre);
t.postorder(t.root, post);
List<Integer> level = t.levelOrder(t.root);
System.out.println("Inorder: " + in);
System.out.println("Preorder: " + pre);
System.out.println("Postorder: " + post);
System.out.println("Level order: " + level);
}
}
Snipping Tool
Screenshot copied to clipboard
Automatically saved to screenshots folder.
Markup and share
Activate Windows
Flood warning
In effect
Search
6:39 am
26/09/2025


File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
BinaryTreeTraversals - Apache NetBeans IDE 23
BinaryTreeTraversals.java
Source History
return out;
}
}
public class BinaryTreeTraversals {
public static void main(String[] args) {
BinaryTree t = new BinaryTree();
t.root = new Node(10);
t.root.left = new Node(5);
t.root.right = new Node(15);
t.root.left.left = new Node(2);
t.root.left.right = new Node(7);
t.root.right.right = new Node(12);
List<Integer> in = new ArrayList<>();
List<Integer> pre = new ArrayList<>();
List<Integer> post = new ArrayList<>();
Output - BinaryTreeTraversals (run)
run:
Inorder: [2, 5, 7, 10, 15, 12]
Preorder: [10, 5, 2, 7, 15, 12]
Postorder: [2, 7, 5, 12, 15, 10]
Level order: [10, 5, 15, 2, 7, 12]
BUILD SUCCESSFUL (total time: 0 seconds)
Activate Windows
Output
Finished building BinaryTreeTraversals (run).
66:9
Flood warning
In effect
Search
6:40 am
26/09/2025