

The screenshot shows the NetBeans IDE interface with a Java file named `BinaryTreeTraversals.java` open. The code defines a `Node` class with data, left, and right pointers, and a `BinaryTree` class with a root pointer. It contains three traversal methods: `inorder`, `preorder`, and `postorder`, each taking a node and a list of integers as parameters. The `inorder` method performs an in-order traversal, `preorder` performs a pre-order traversal, and `postorder` performs a post-order traversal. A tooltip "Activate Windows" is visible in the bottom right corner.

```
1 package binarytreetraversals;
2 import java.util.*;
3
4 class Node {
5     int data;
6     Node left;
7     Node right;
8 }
9
10 public Node(int data) {
11     this.data = data;
12 }
13
14 class BinaryTree {
15     Node root;
16 }
17
18 public void inorder(Node n, List<Integer> out) {
19     if (n == null)
20         return;
21     inorder(n.left, out);
22     out.add(n.data);
23     inorder(n.right, out);
24 }
25
26 public void preorder(Node n, List<Integer> out) {
27     if (n == null)
28         return;
29     out.add(n.data);
30     preorder(n.left, out);
31     preorder(n.right, out);
32 }
33
34 public void postorder(Node n, List<Integer> out) {
35     if (n == null)
36         return;
37     postorder(n.left, out);
38     postorder(n.right, out);
39     out.add(n.data);
40 }
41
42 public List<Integer> levelOrder(Node n) {
43     List<Integer> out = new ArrayList<>();
44
45     if (n == null)
46         return out;
47
48     Queue<Node> q = new ArrayDeque<>();
49     q.add(n);
50
51     while (!q.isEmpty()) {
52         Node cur = q.remove();
53         out.add(cur.data);
54         if (cur.left != null)
55             q.add(cur.left);
56         if (cur.right != null)
57             q.add(cur.right);
58     }
59
60     return out;
61 }
62
63 public class BinaryTreeTraversals {
64     public static void main(String[] args) {
65
66         BinaryTree t = new BinaryTree();
67
68         t.root = new Node(10);
69         t.root.left = new Node(5);
70     }
71 }
```

This screenshot shows the NetBeans IDE interface with the same `BinaryTreeTraversals.java` file open. In addition to the code editor, the left side features the Project, Services, and Navigator panes. The Navigator pane displays the class hierarchy and method signatures. A tooltip "Activate Windows" is visible in the bottom right corner.

```
1 package binarytreetraversals;
2 import java.util.*;
3
4 class Node {
5     int data;
6     Node left;
7     Node right;
8 }
9
10 public Node(int data) {
11     this.data = data;
12 }
13
14 class BinaryTree {
15     Node root;
16 }
17
18 public void inorder(Node n, List<Integer> out) {
19     if (n == null)
20         return;
21     inorder(n.left, out);
22     out.add(n.data);
23     inorder(n.right, out);
24 }
25
26 public void preorder(Node n, List<Integer> out) {
27     if (n == null)
28         return;
29     out.add(n.data);
30     preorder(n.left, out);
31     preorder(n.right, out);
32 }
33
34 public void postorder(Node n, List<Integer> out) {
35     if (n == null)
36         return;
37     postorder(n.left, out);
38     postorder(n.right, out);
39     out.add(n.data);
40 }
41
42 public List<Integer> levelOrder(Node n) {
43     List<Integer> out = new ArrayList<>();
44
45     if (n == null)
46         return out;
47
48     Queue<Node> q = new ArrayDeque<>();
49     q.add(n);
50
51     while (!q.isEmpty()) {
52         Node cur = q.remove();
53         out.add(cur.data);
54         if (cur.left != null)
55             q.add(cur.left);
56         if (cur.right != null)
57             q.add(cur.right);
58     }
59
60     return out;
61 }
62
63 public class BinaryTreeTraversals {
64     public static void main(String[] args) {
65
66         BinaryTree t = new BinaryTree();
67
68         t.root = new Node(10);
69         t.root.left = new Node(5);
70     }
71 }
```

BinaryTreeTraversals.java

```
01     return out;
02 }
03 }
04
05 public class BinaryTreeTraversals {
06     public static void main(String[] args) {
07         BinaryTree t = new BinaryTree();
08
09         t.root = new Node(10);
10        t.root.left = new Node(5);
11        t.root.right = new Node(15);
12        t.root.left.left = new Node(2);
13        t.root.left.right = new Node(7);
14        t.root.right.right = new Node(12);
15
16        List<Integer> in = new ArrayList<>();
17        List<Integer> pre = new ArrayList<>();
18        List<Integer> post = new ArrayList<>();
19
20        t.inorder(t.root, in);
21        t.preorder(t.root, pre);
22        t.postorder(t.root, post);
23        List<Integer> level = t.levelOrder(t.root);
24
25        System.out.println("Inorder: " + in);
26        System.out.println("Preorder: " + pre);
27        System.out.println("Postorder: " + post);
28        System.out.println("Level order: " + level);
29    }
30 }
```

Flood warning In effect

Snipping Tool

Screenshot copied to clipboard
Automatically saved to screenshots folder.
Activate Windows
Go to Settings to activate Windows.

00:11 26/09/2025 6:39 am INS Windows (CRLF)

BinaryTreeTraversals.java

```
01     return out;
02 }
03 }
04
05 public class BinaryTreeTraversals {
06     public static void main(String[] args) {
07         BinaryTree t = new BinaryTree();
08
09         t.root = new Node(10);
10        t.root.left = new Node(5);
11        t.root.right = new Node(15);
12        t.root.left.left = new Node(2);
13        t.root.left.right = new Node(7);
14        t.root.right.right = new Node(12);
15
16        List<Integer> in = new ArrayList<>();
17        List<Integer> pre = new ArrayList<>();
18        List<Integer> post = new ArrayList<>();
19
20        t.inorder(t.root, in);
21        t.preorder(t.root, pre);
22        t.postorder(t.root, post);
23        List<Integer> level = t.levelOrder(t.root);
24
25        System.out.println("Inorder: " + in);
26        System.out.println("Preorder: " + pre);
27        System.out.println("Postorder: " + post);
28        System.out.println("Level order: " + level);
29    }
30 }
```

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
Go to Settings to activate Windows.

Output Finished building BinaryTreeTraversals (run).

Flood warning In effect

00:11 26/09/2025 6:40 am INS Windows (CRLF)