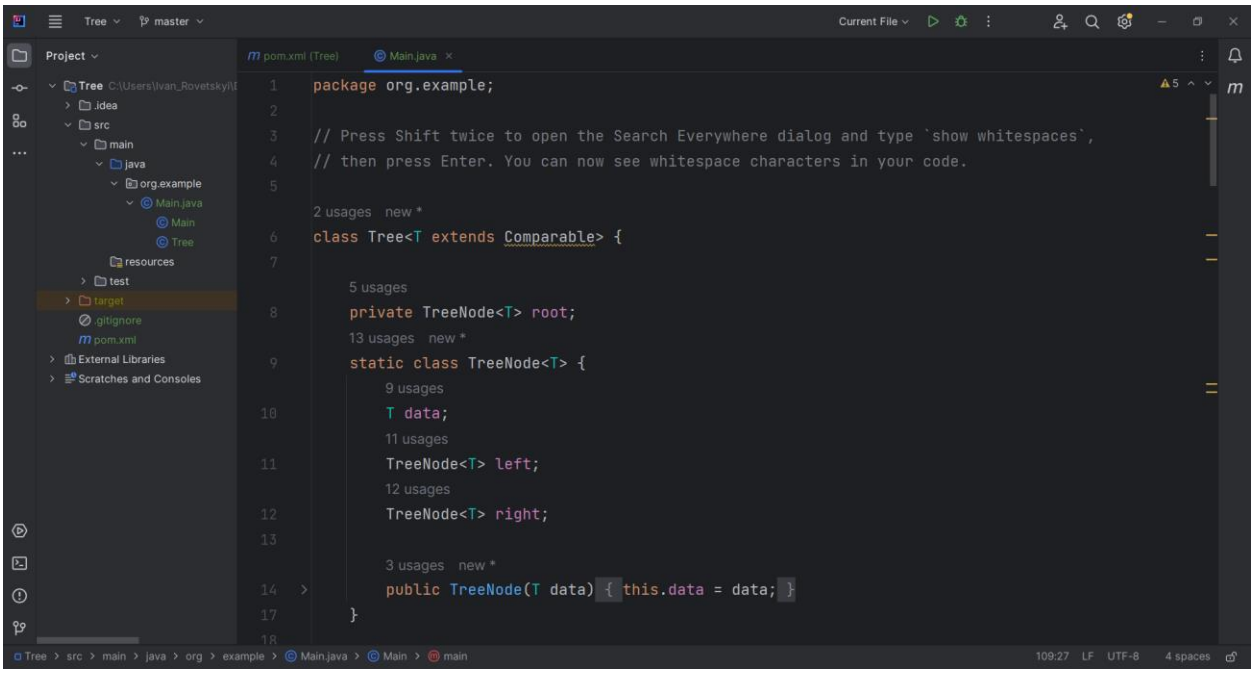
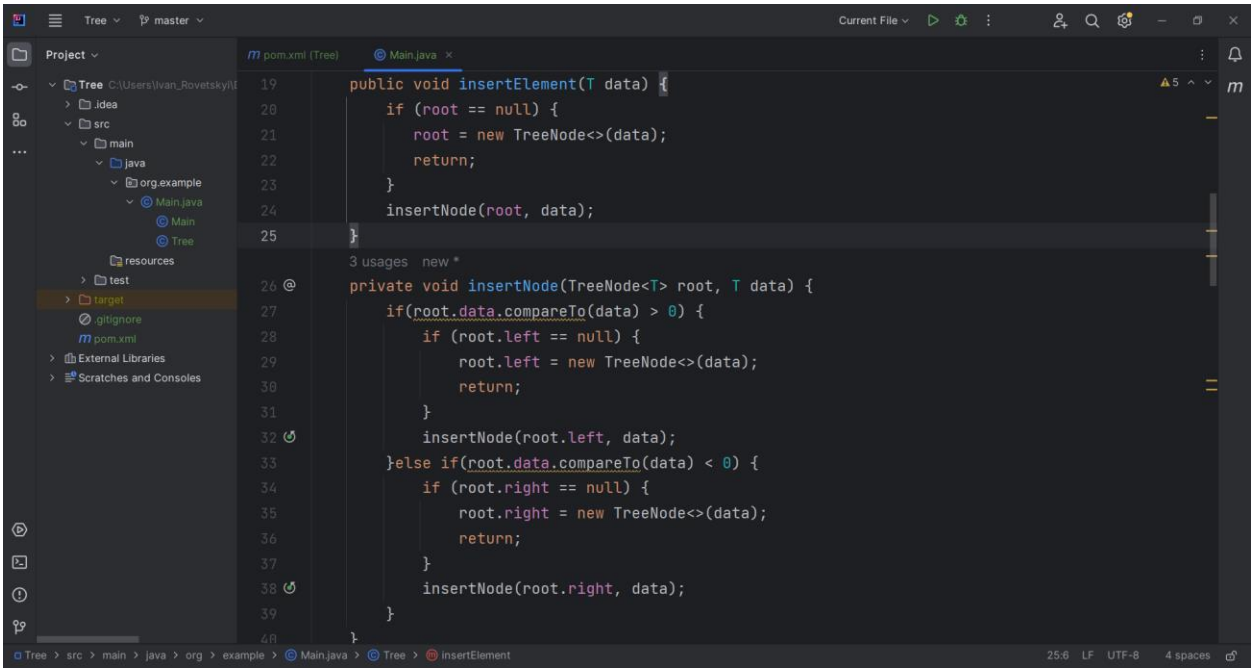
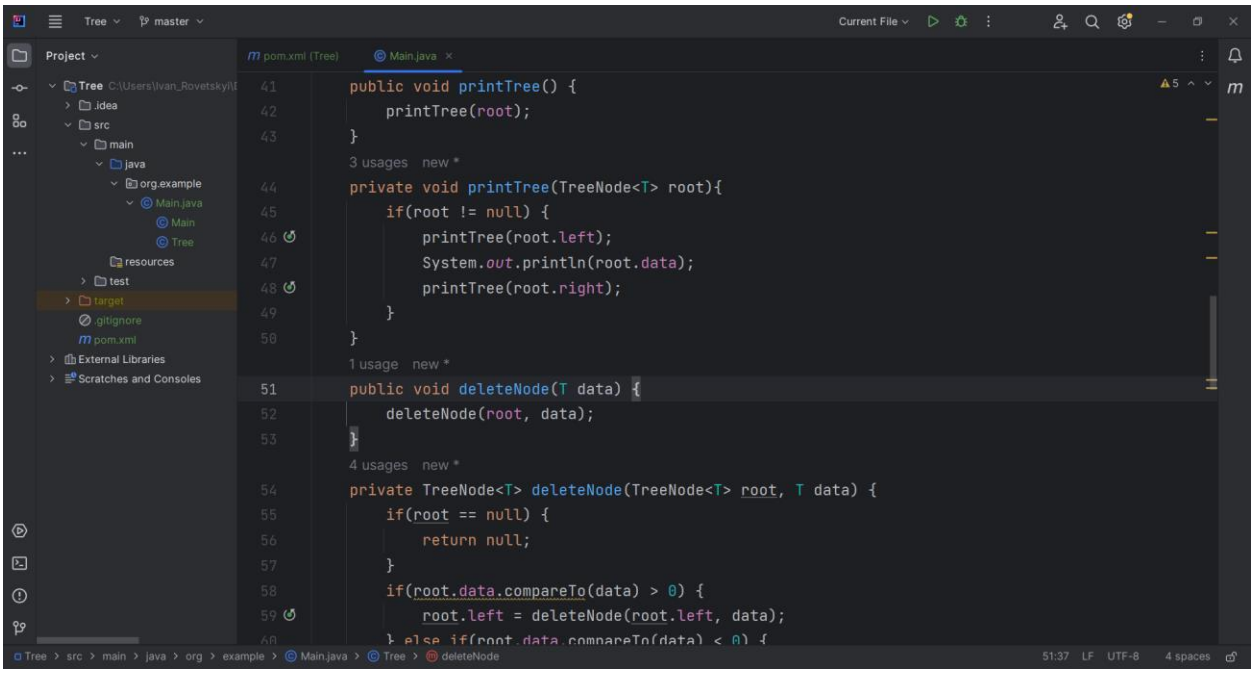
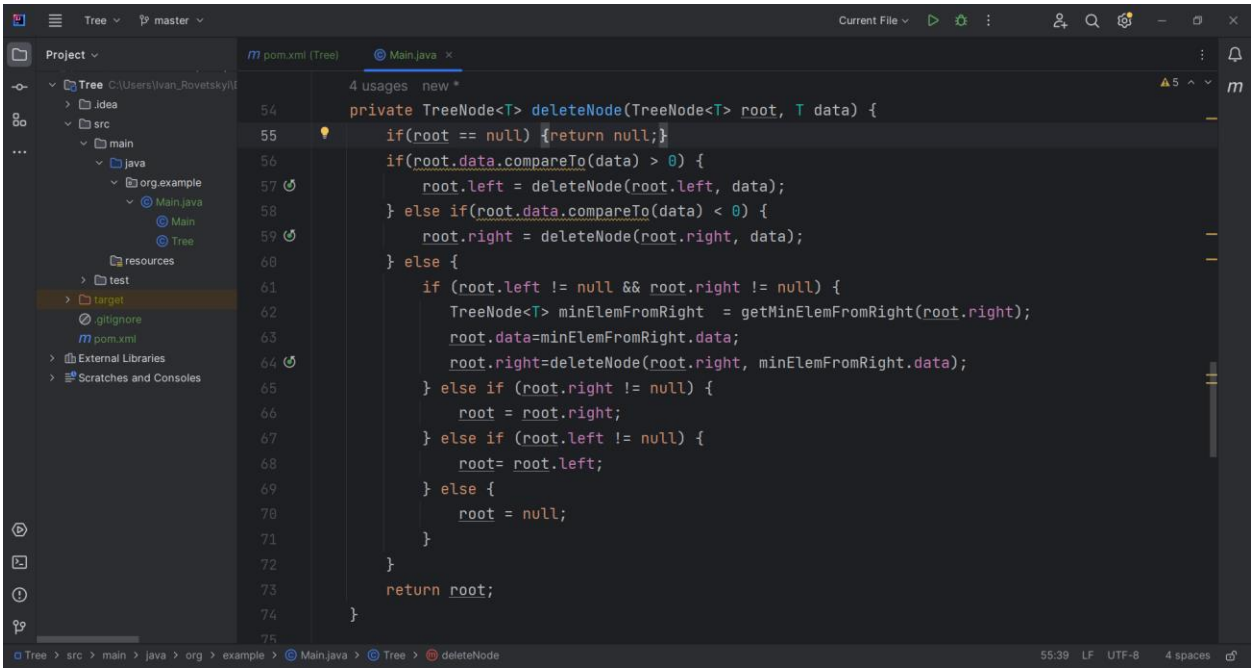

Tree
master
Current File
Project
pom.xml (Tree)
Main.java
Tree C:\Users\Ivan_Rovetskyi\D
.idea
src
main
java
org.example
Main.java
Main
Tree
resources
test
target
.gitignore
pom.xml
External Libraries
Scratches and Consoles
package org.example;
// Press Shift twice to open the Search Everywhere dialog and type `show whitespaces`,
// then press Enter. You can now see whitespace characters in your code.
2 usages new *
class Tree<T extends Comparable> {
5 usages
private TreeNode<T> root;
13 usages new *
static class TreeNode<T> {
9 usages
T data;
11 usages
TreeNode<T> left;
12 usages
TreeNode<T> right;
3 usages new *
public TreeNode(T data) { this.data = data; }
}
Tree > src > main > java > org > example > Main.java > Main > main
109:27 LF UTF-8 4 spaces

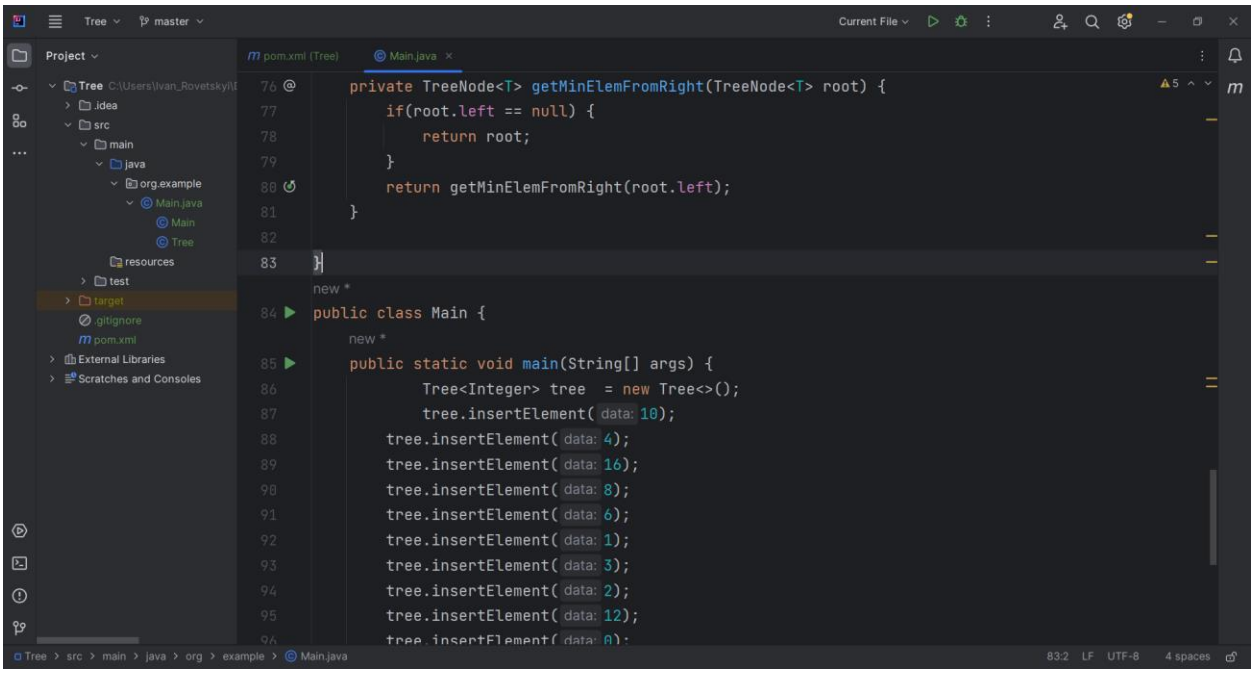

Tree
master
Current File
Project
pom.xml (Tree)
Main.java
Tree C:\Users\Ivan_Rovetskyi\D
.idea
src
main
java
org.example
Main.java
Main
Tree
resources
test
target
.gitignore
pom.xml
External Libraries
Scratches and Consoles
public void insertElement(T data) {
if (root == null) {
root = new TreeNode<>(data);
return;
}
insertNode(root, data);
}
3 usages new *
private void insertNode(TreeNode<T> root, T data) {
if(root.data.compareTo(data) > 0) {
if (root.left == null) {
root.left = new TreeNode<>(data);
return;
}
insertNode(root.left, data);
}else if(root.data.compareTo(data) < 0) {
if (root.right == null) {
root.right = new TreeNode<>(data);
return;
}
insertNode(root.right, data);
}
}
Tree > src > main > java > org > example > Main.java > Tree > insertElement
25:6 LF UTF-8 4 spaces



```
public void printTree() {
    printTree(root);
}
3 usages  new *
private void printTree(TreeNode<T> root){
    if(root != null) {
        printTree(root.left);
        System.out.println(root.data);
        printTree(root.right);
    }
}
1 usage  new *
public void deleteNode(T data) {
    deleteNode(root, data);
}
4 usages  new *
private TreeNode<T> deleteNode(TreeNode<T> root, T data) {
    if(root == null) {
        return null;
    }
    if(root.data.compareTo(data) > 0) {
        root.left = deleteNode(root.left, data);
    } else if(root.data.compareTo(data) < 0) {
```

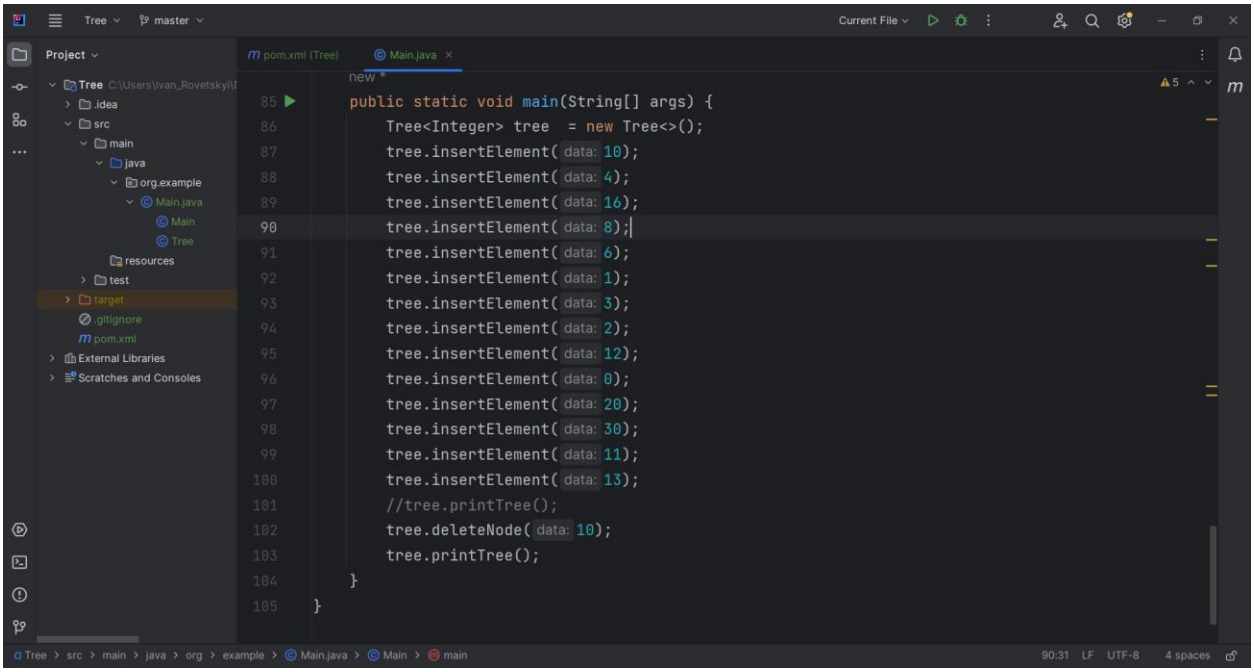


```
4 usages  new *
private TreeNode<T> deleteNode(TreeNode<T> root, T data) {
    if(root == null) {return null;}
    if(root.data.compareTo(data) > 0) {
        root.left = deleteNode(root.left, data);
    } else if(root.data.compareTo(data) < 0) {
        root.right = deleteNode(root.right, data);
    } else {
        if (root.left != null && root.right != null) {
            TreeNode<T> minElemFromRight  = getMinElemFromRight(root.right);
            root.data=minElemFromRight.data;
            root.right=deleteNode(root.right, minElemFromRight.data);
        } else if (root.right != null) {
            root = root.right;
        } else if (root.left != null) {
            root= root.left;
        } else {
            root = null;
        }
    }
    return root;
}
```



```
    private TreeNode<T> getMinElemFromRight(TreeNode<T> root) {
        if(root.left == null) {
            return root;
        }
        return getMinElemFromRight(root.left);
    }

}
new *
public class Main {
    new *
    public static void main(String[] args) {
            Tree<Integer> tree  = new Tree<>();
            tree.insertElement( data: 10);
        tree.insertElement( data: 4);
        tree.insertElement( data: 16);
        tree.insertElement( data: 8);
        tree.insertElement( data: 6);
        tree.insertElement( data: 1);
        tree.insertElement( data: 3);
        tree.insertElement( data: 2);
        tree.insertElement( data: 12);
```



```
    new *
    public static void main(String[] args) {
        Tree<Integer> tree  = new Tree<>();
        tree.insertElement( data: 10);
        tree.insertElement( data: 4);
        tree.insertElement( data: 16);
        tree.insertElement( data: 8);
        tree.insertElement( data: 6);
        tree.insertElement( data: 1);
        tree.insertElement( data: 3);
        tree.insertElement( data: 2);
        tree.insertElement( data: 12);
        tree.insertElement( data: 0);
        tree.insertElement( data: 20);
        tree.insertElement( data: 30);
        tree.insertElement( data: 11);
        tree.insertElement( data: 13);
        //tree.printTree();
        tree.deleteNode( data: 10);
        tree.printTree();
    }
}
```