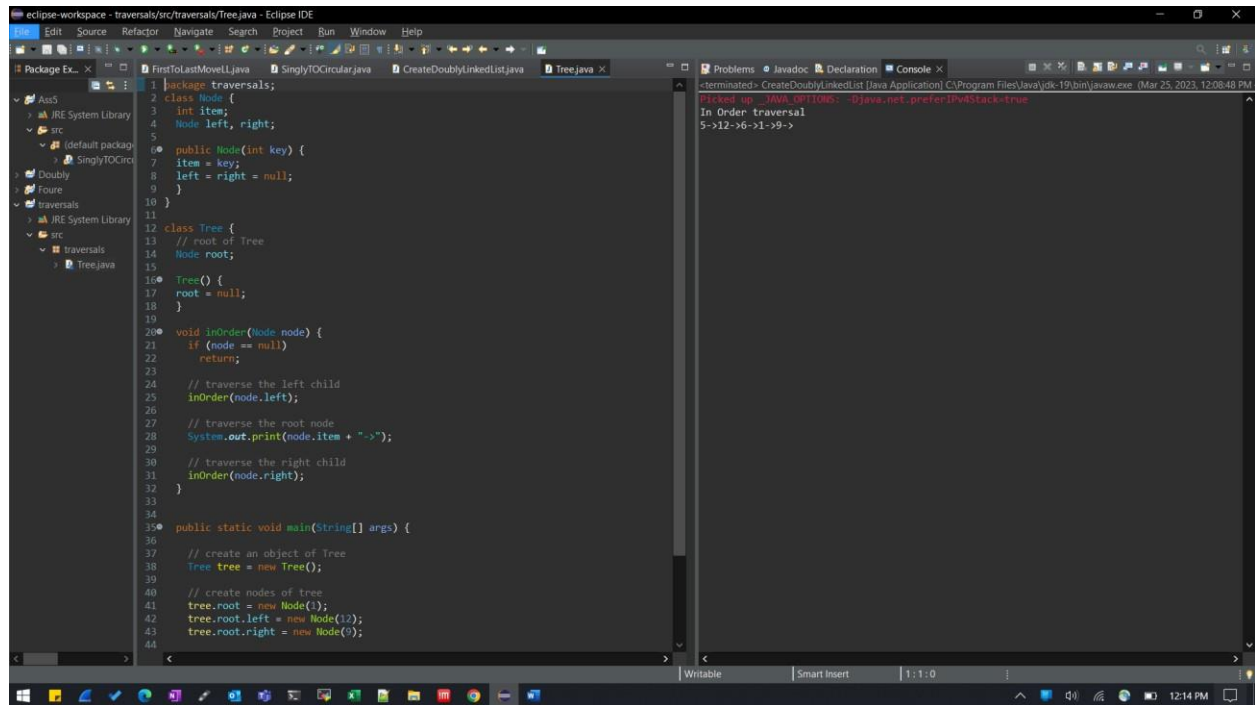


LAB SHEET 6

Aim: To understand the concepts of Tree data structure and its traversals.

1. Implement a program for different tree traversals.



```
package traversals;

class Node {
    int item;
    Node left, right;

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

class Tree {
    // root of Tree
    Node root;

    Tree() {
        root = null;
    }

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

        // traverse the left child
        inOrder(node.left);

        // traverse the root node
        System.out.print(node.item + "->");

        // traverse the right child
        inOrder(node.right);
    }

    public static void main(String[] args) {
        // create an object of Tree
        Tree tree = new Tree();

        // create nodes of tree
        tree.root = new Node(1);
        tree.root.left = new Node(12);
        tree.root.right = new Node(6);
    }
}
```

Console Output:

```
-terminated> CreateDoublyLinkedList [Java Application] C:\Program Files\Java\jdk-19\bin\javaw.exe (Mar 25, 2023, 12:08:48 PM)
Picked up _JAVA_OPTIONS: -Djava.net.preferIPv4Stack=true
In Order traversal
5->12->6->1->9->
```

```
package traversals;

class Node {

int item;

Node left, right;

public Node(int key) {

item = key;

left = right = null;

}

}

class Tree {
```

```
// root of Tree

Node root;

Tree() {
    root = null;
}

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

    // traverse the left child
    inOrder(node.left);

    // traverse the root node
    System.out.print(node.item + "->");

    // traverse the right child
    inOrder(node.right);
}

public static void main(String[] args) {
    // create an object of Tree
    Tree tree = new Tree();

    // create nodes of tree
    tree.root = new Node(1);
    tree.root.left = new Node(12);
    tree.root.right = new Node(9);

    // create child nodes of left child
    tree.root.left.left = new Node(5);
    tree.root.left.right = new Node(6);
}
```

```
System.out.println("In Order traversal");  
  
tree.inOrder(tree.root);  
  
}  
  
}
```