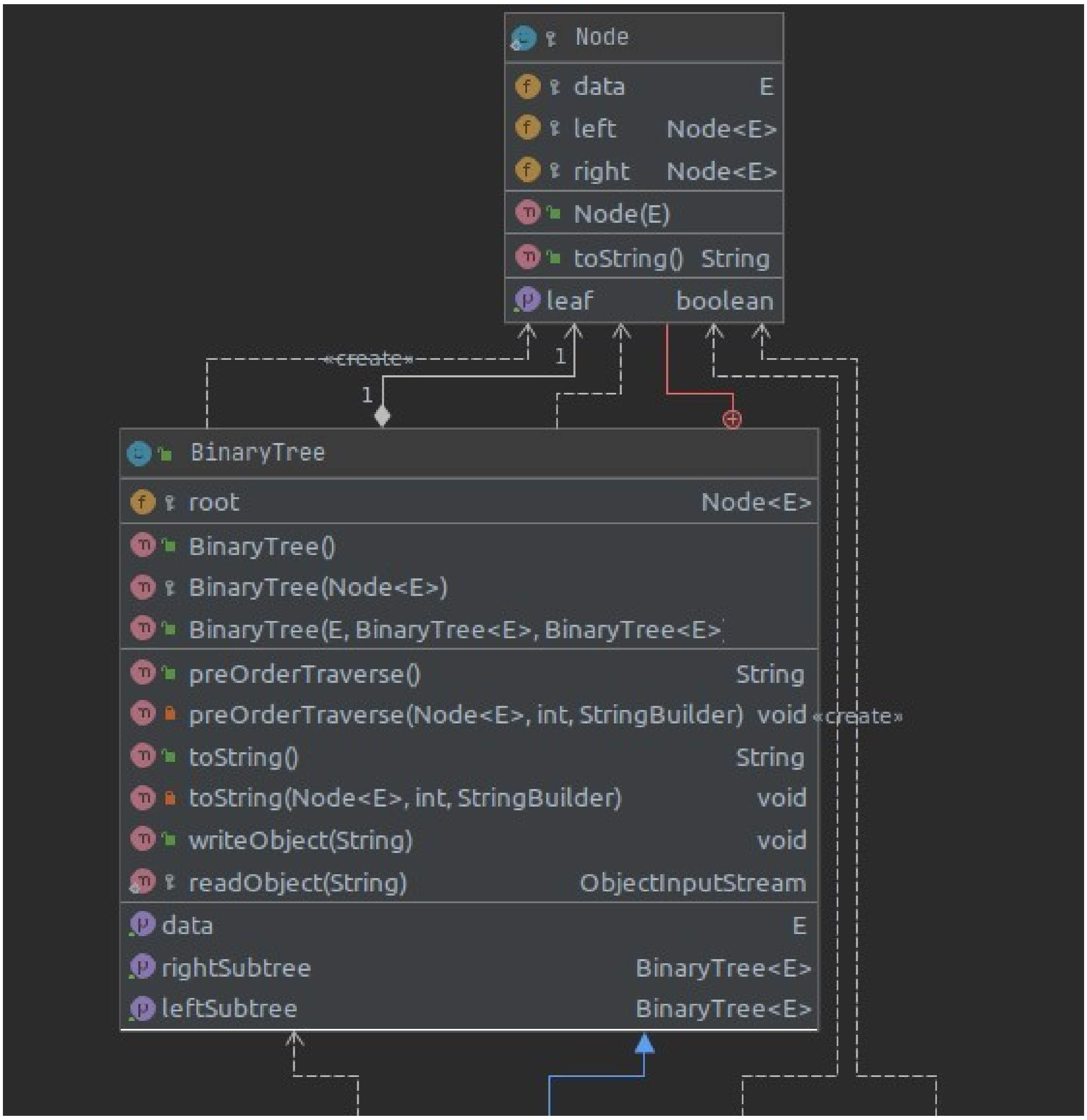
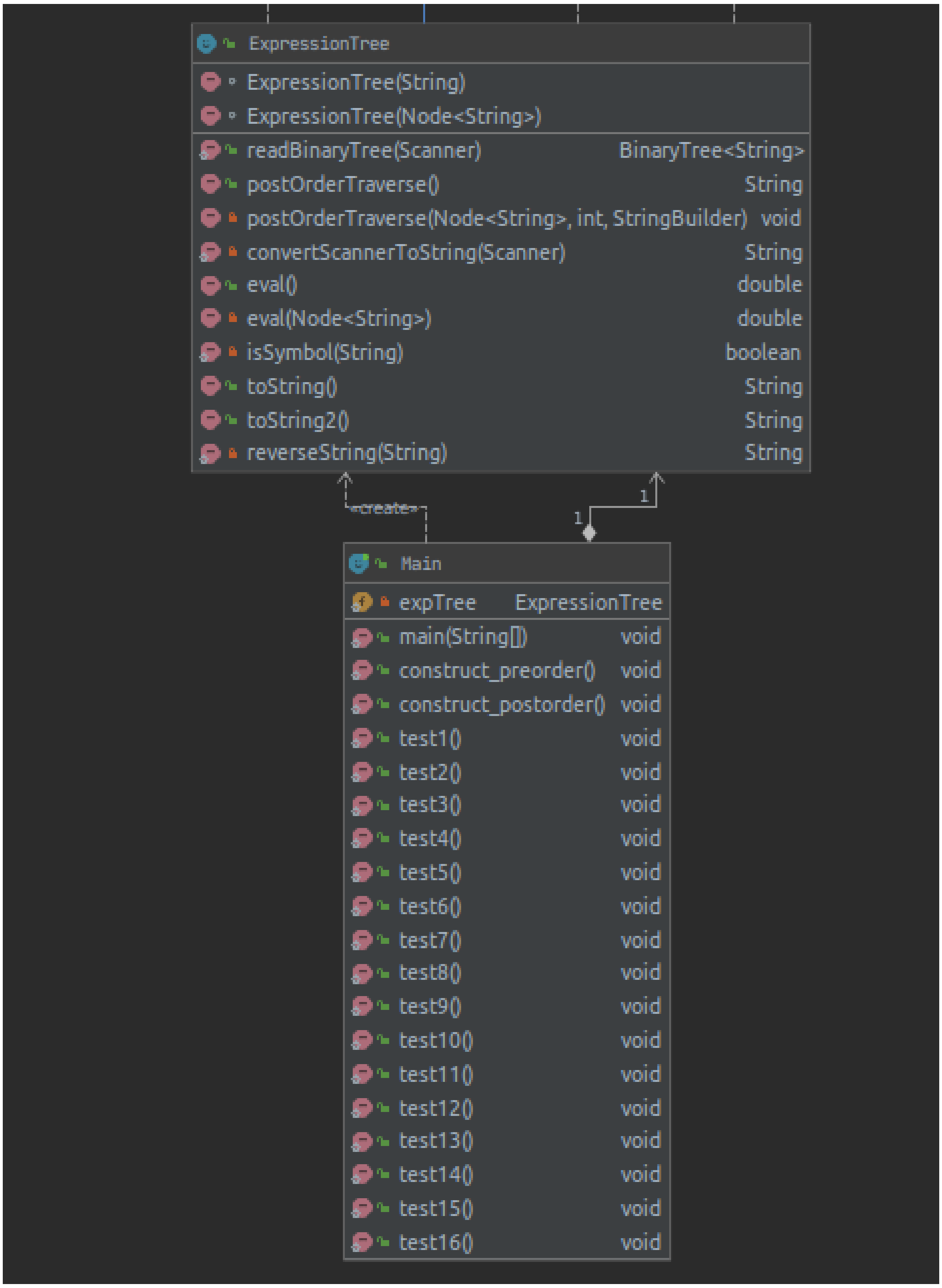
GIT Department of Computer Engineering
CSE 222/505 - Spring 2020
Homework 5 Question 3 Report

Melihcan Çilek
1801042092

1-)Class Diagram

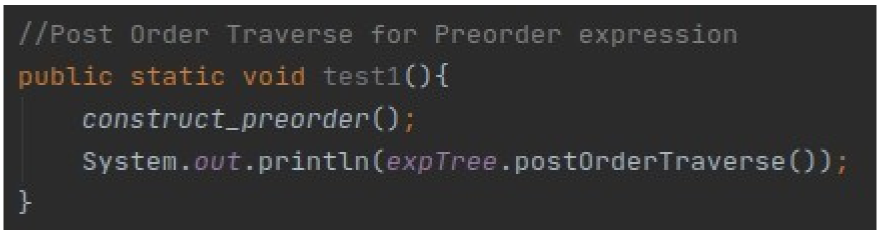

Node
data E
left Node<E>
right Node<E>
Node(E)
toString() String
leaf boolean
«create»
1
1
«create»
BinaryTree
root Node<E>
BinaryTree()
BinaryTree(Node<E>)
BinaryTree(E, BinaryTree<E>, BinaryTree<E>)
preOrderTraverse() String
preOrderTraverse(Node<E>, int, StringBuilder) void
toString() String
toString(Node<E>, int, StringBuilder) void
writeObject(String) void
readObject(String) ObjectInputStream
data E
rightSubtree BinaryTree<E>
leftSubtree BinaryTree<E>


ExpressionTree
ExpressionTree(String)
ExpressionTree(Node<String>)
readBinaryTree(Scanner) BinaryTree<String>
postOrderTraverse() String
postOrderTraverse(Node<String>, int, StringBuilder) void
convertScannerToString(Scanner) String
eval() double
eval(Node<String>) double
isSymbol(String) boolean
toString() String
toString2() String
reverseString(String) String
«create»
1
1
Main
expTree ExpressionTree
main(String[]) void
construct_preorder() void
construct_postorder() void
test1() void
test2() void
test3() void
test4() void
test5() void
test6() void
test7() void
test8() void
test9() void
test10() void
test11() void
test12() void
test13() void
test14() void
test15() void
test16() void

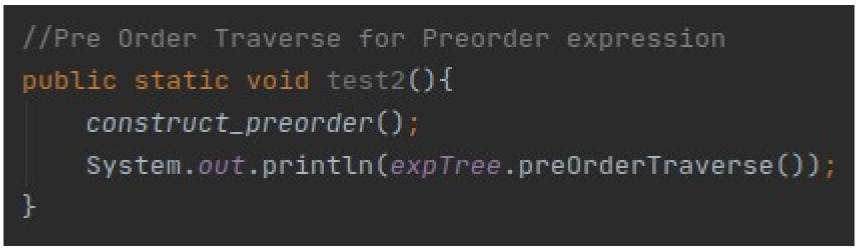
2-)Problem Solution Approach

If I have to summarize problem for questions 1, problem for question 2 is that implement ExpressionTree class of arithmetic operations which extends the BinaryTree class implementation given in your Data Structures book. First, I copied BinaryTree class from our Data Structures book. In our problem, we have to detect whether input is prefix or postfix. Because of our readBinaryTree file is taking Scanner so I have to convert it into string and determine whether input is prefix or postfix, then I convert string if input is postfix so it became prefix and it is really easy to convert prefix to tree. There was a problem after converting input to prefix version. Problem is that I constructed a while loop that uses scanner.hasNext() method but in this while loop, stop was occured at last number so last number at prefix expression wasn’t taken. For solve that problem, I’m adding “ .” end of the string so last number of prefix expression is scanned by scanner. At eval method, we weren’t able to use stack so I tried to solve prefix expression using recursion.

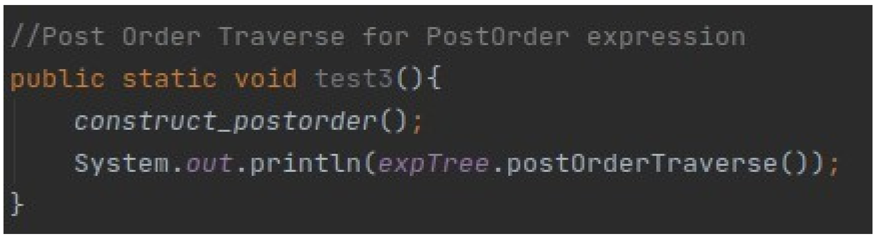
3-)Running Command And Results



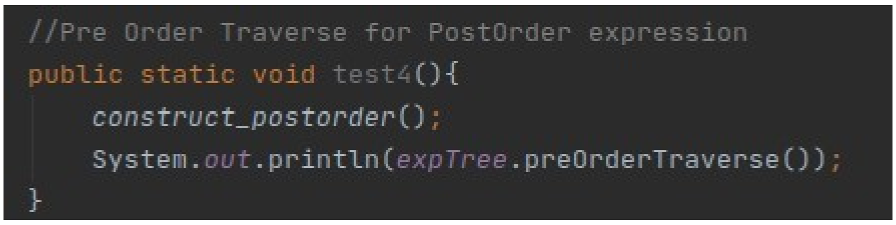
```java
//Post Order Traverse for Preorder expression
public static void test1(){
    construct_preorder();
    System.out.println(expTree.postOrderTraverse());
}
```



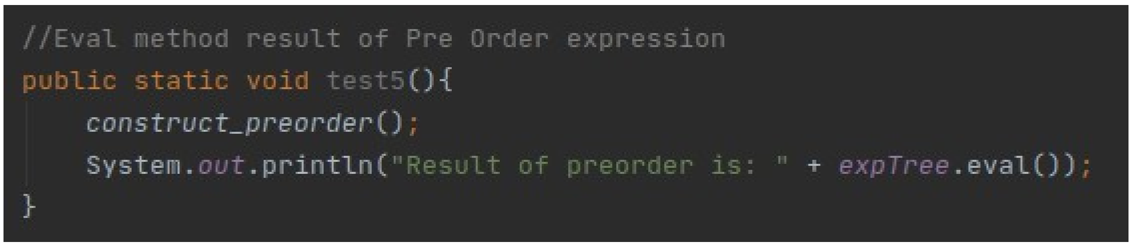
```java
//Pre Order Traverse for Preorder expression
public static void test2(){
    construct_preorder();
    System.out.println(expTree.preOrderTraverse());
}
```



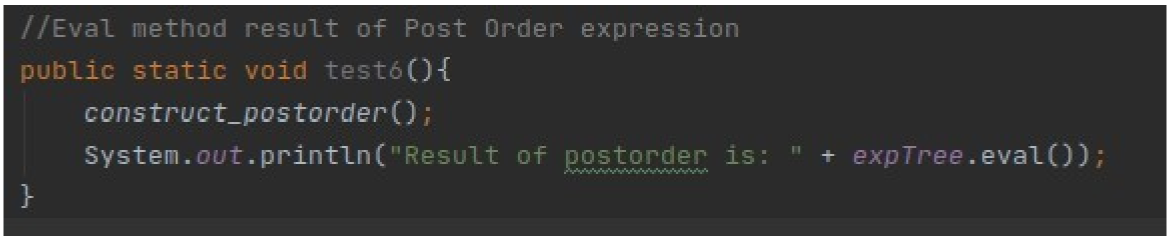
```java
//Post Order Traverse for PostOrder expression
public static void test3(){
    construct_postorder();
    System.out.println(expTree.postOrderTraverse());
}
```



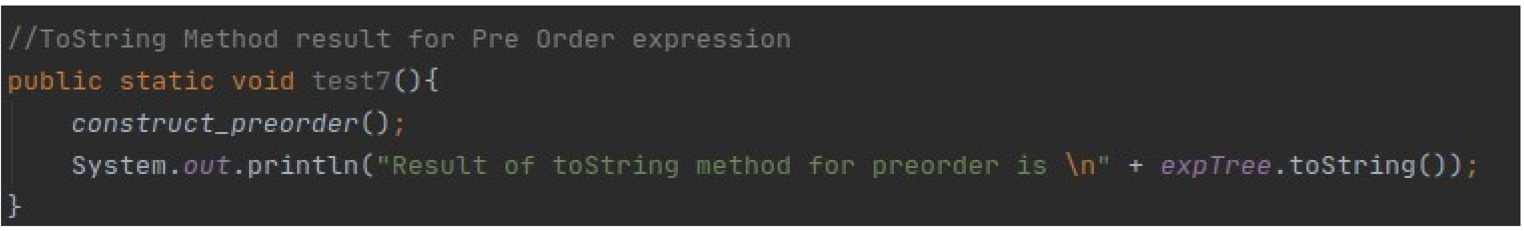
```java
//Pre Order Traverse for PostOrder expression
public static void test4(){
    construct_postorder();
    System.out.println(expTree.preOrderTraverse());
}
```



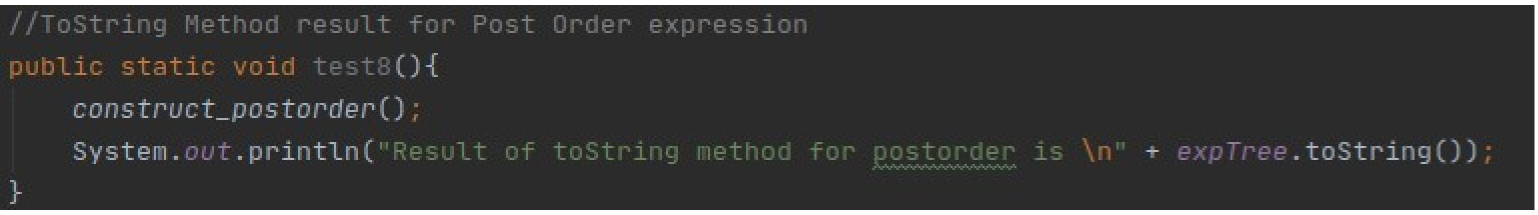
```java
//Eval method result of Pre Order expression
public static void test5(){
    construct_preorder();
    System.out.println("Result of preorder is: " + expTree.eval());
}
```



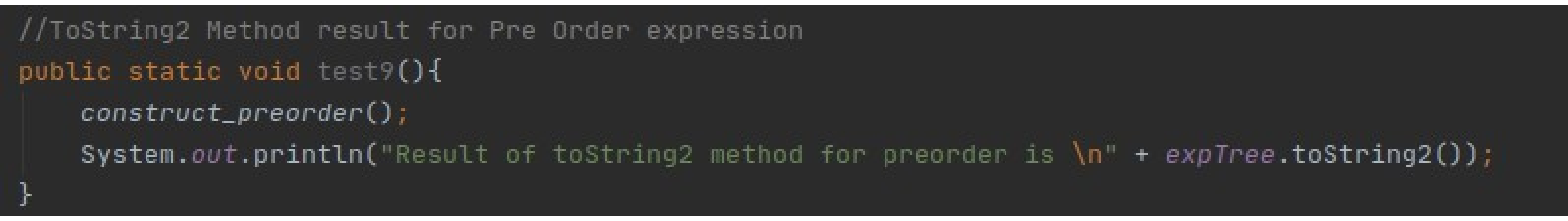
```java
//Eval method result of Post Order expression
public static void test6(){
    construct_postorder();
    System.out.println("Result of postorder is: " + expTree.eval());
}
```



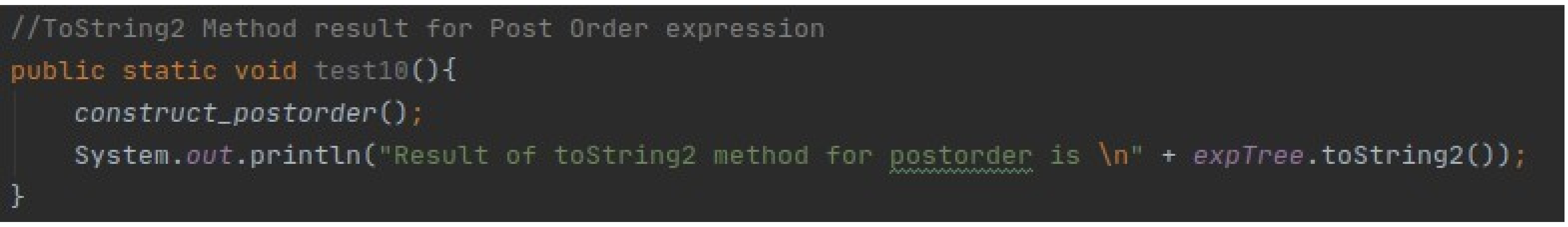
```
//ToString Method result for Pre Order expression
public static void test7(){
    construct_preorder();
    System.out.println("Result of toString method for preorder is \n" + expTree.toString());
}
```



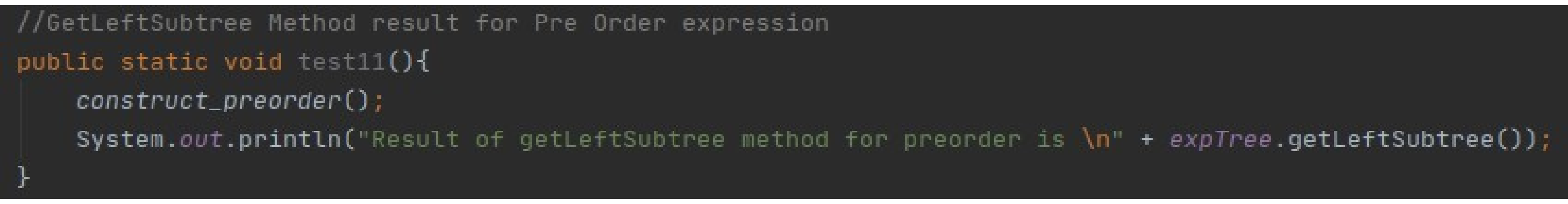
```
//ToString Method result for Post Order expression
public static void test8(){
    construct_postorder();
    System.out.println("Result of toString method for postorder is \n" + expTree.toString());
}
```



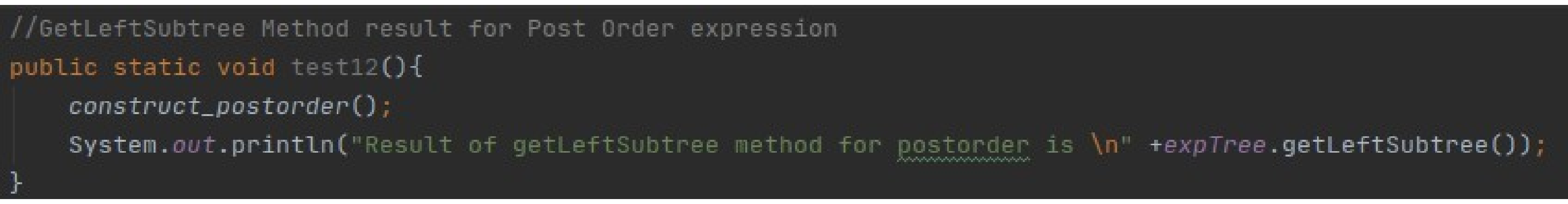
```
//ToString2 Method result for Pre Order expression
public static void test9(){
    construct_preorder();
    System.out.println("Result of toString2 method for preorder is \n" + expTree.toString2());
}
```



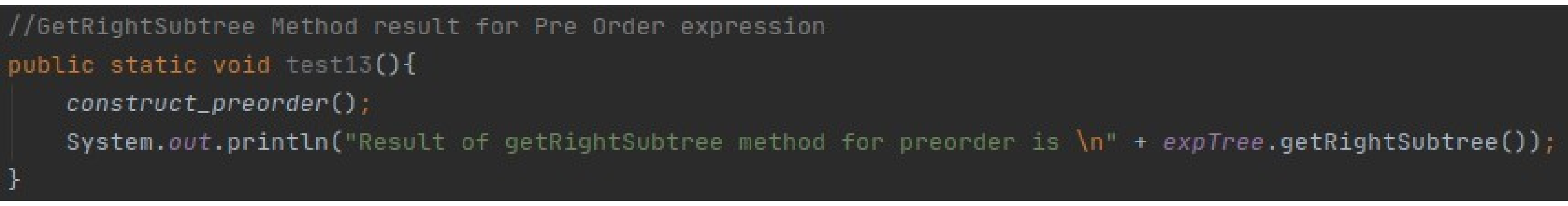
```
//ToString2 Method result for Post Order expression
public static void test10(){
    construct_postorder();
    System.out.println("Result of toString2 method for postorder is \n" + expTree.toString2());
}
```



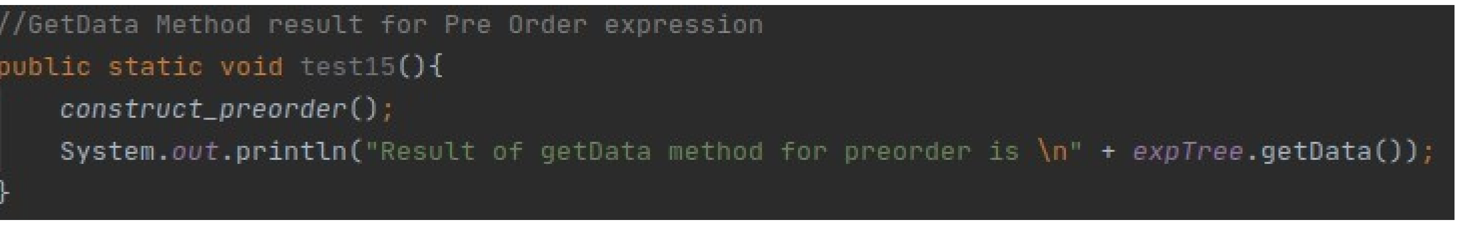
```
//GetLeftSubtree Method result for Pre Order expression
public static void test11(){
    construct_preorder();
    System.out.println("Result of getLeftSubtree method for preorder is \n" + expTree.getLeftSubtree());
}
```



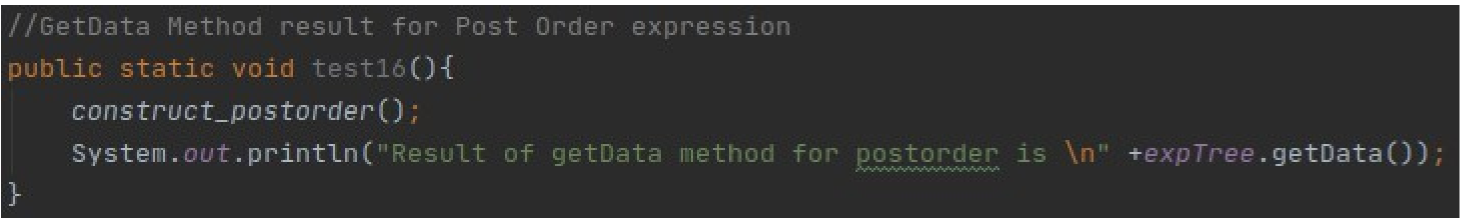
```
//GetLeftSubtree Method result for Post Order expression
public static void test12(){
    construct_postorder();
    System.out.println("Result of getLeftSubtree method for postorder is \n" +expTree.getLeftSubtree());
}
```



```
//GetRightSubtree Method result for Pre Order expression
public static void test13(){
    construct_preorder();
    System.out.println("Result of getRightSubtree method for preorder is \n" + expTree.getRightSubtree());
}
```



```
//GetData Method result for Pre Order expression
public static void test15(){
    construct_preorder();
    System.out.println("Result of getData method for preorder is \n" + expTree.getData());
}
```



```
//GetData Method result for Post Order expression
public static void test16(){
    construct_postorder();
    System.out.println("Result of getData method for postorder is \n" +expTree.getData());
}
```