GIT Department of Computer Engineering

CSE 222/505 - Spring 2020

Homework #5 Part 2 Report

Murat YILDIZ

1801042004

PROBLEM SOLUTION APPROACH

STEP 1 – Extend BinaryTree in the book to implements ExpressionTree

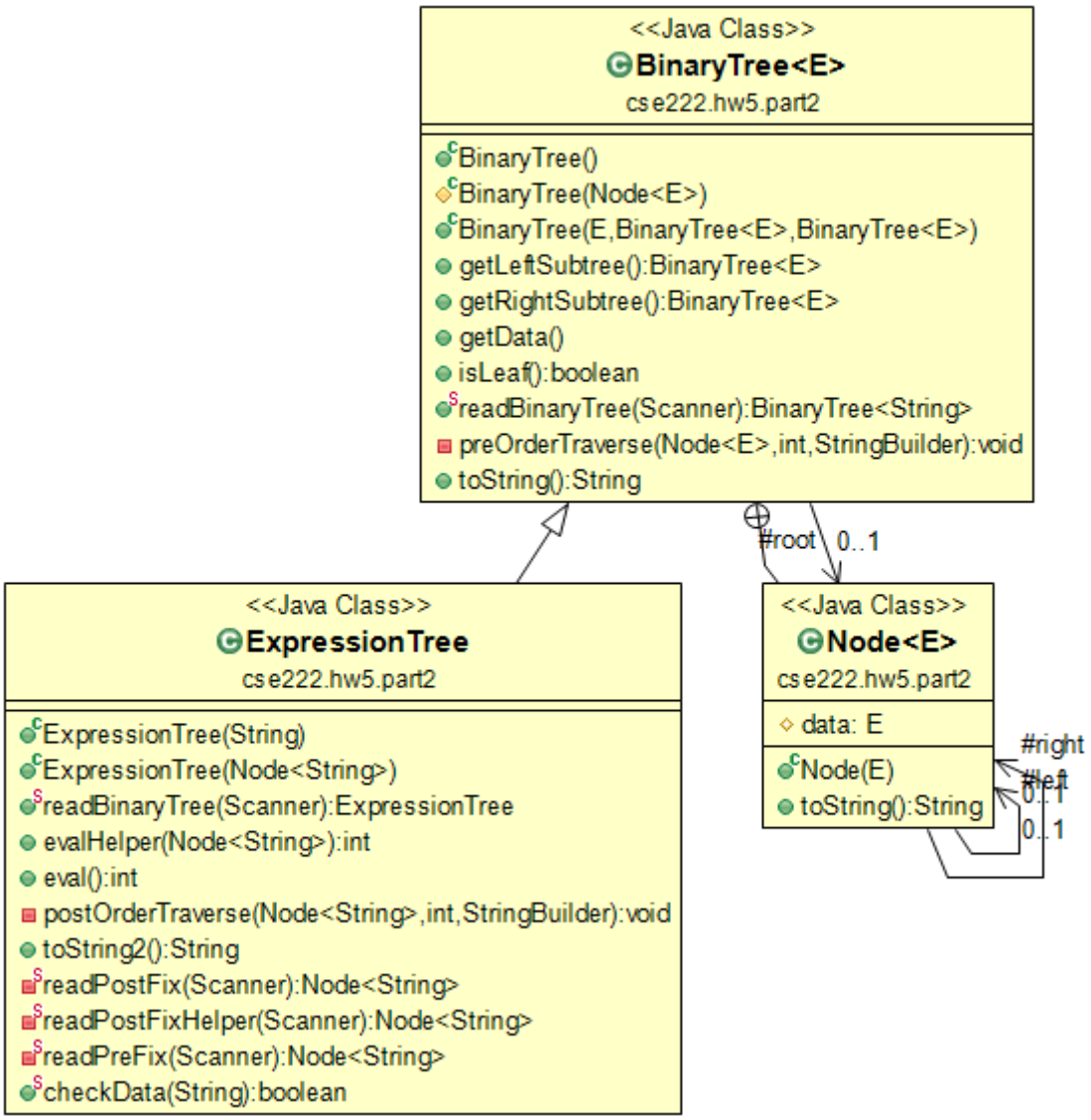
STEP 2 – After that implement readBinaryTree(), postOrderTraverse(), toString2(), eval() methods

STEP 3 – readBinaryTree() read an expression(postfix, prefix) and place it to tree

STEP 4 – toString() uses postOrderTraverse() which visit left – right – root

STEP 5 – eval() return the result of the expression

CLASS DIAGRAM



TEST CASES

TEST ID	Scenario	Test Data	Expected Results	Actual Results	Pass/ Fail
TEST01	ExpressionTree(null)	null	Exception Throwed	As Expected	Pass
TEST02	ExpressionTree()	/ 64 * 8 + 1 3	Successfully Created	As Expected	Pass
TEST03	toString() which uses preOrderTraverse()		Successfully Printed	As Expected	Pass
TEST04	eval()		Successfully Returned	As Expected	Pass
TEST05	ExpressionTree()	10 5 15 * + 20 +	Successfully Created	As Expected	Pass
TEST06	toString() which uses preOrderTraverse()		Successfully Printed	As Expected	Pass
TEST07	eval()		Successfully Returned	As Expected	Pass
TEST08	toString2() which uses postOrderTraverse()		Successfully Printed	As Expected	Pass

RUNNING AND RESULTS

```
TEST01 - Testing ExpressionTree() constructor with null

java.lang.NullPointerException


TEST02 - Testing ExpressionTree() constructor with prefix expression
 (/ 64 * 8 + 1 3)
```

```
 TEST03 - Testing toString() which uses preOrderTraverse

/
  64
    null
    null
  *
    8
      null
      null
    +
      1
        null
        null
      3
        null
        null
```

```
/
  64
    null
    null
  *
    8
      null
      null
    +
      1
        null
        null
      3
        null
        null
```

```
 TEST04 - Testing eval()

 RESULT: 2
```

```
TEST05 - Testing ExpressionTree() constructor with postfix expression
 (10 5 15 * + 20 +)


TEST06 - Testing toString() which uses preOrderTraverse

+
  +
    10
      null
      null
    *
      5
        null
        null
      15
        null
        null
  20
    null
    null
```

```
+
  +
    10
      null
      null
    *
      5
        null
        null
      15
        null
        null
  20
    null
    null
```

```
TEST07 - Testing eval()

RESULT: 105
```

```
TEST08 - Testing toString2() which uses postOrderTraverse

            null
      null
10
                  null
        null
5
               null
        null
15
*
+
      null
    null
20
+
```