GTU Department of Computer Engineering

CSE 222/505 - Spring 2020

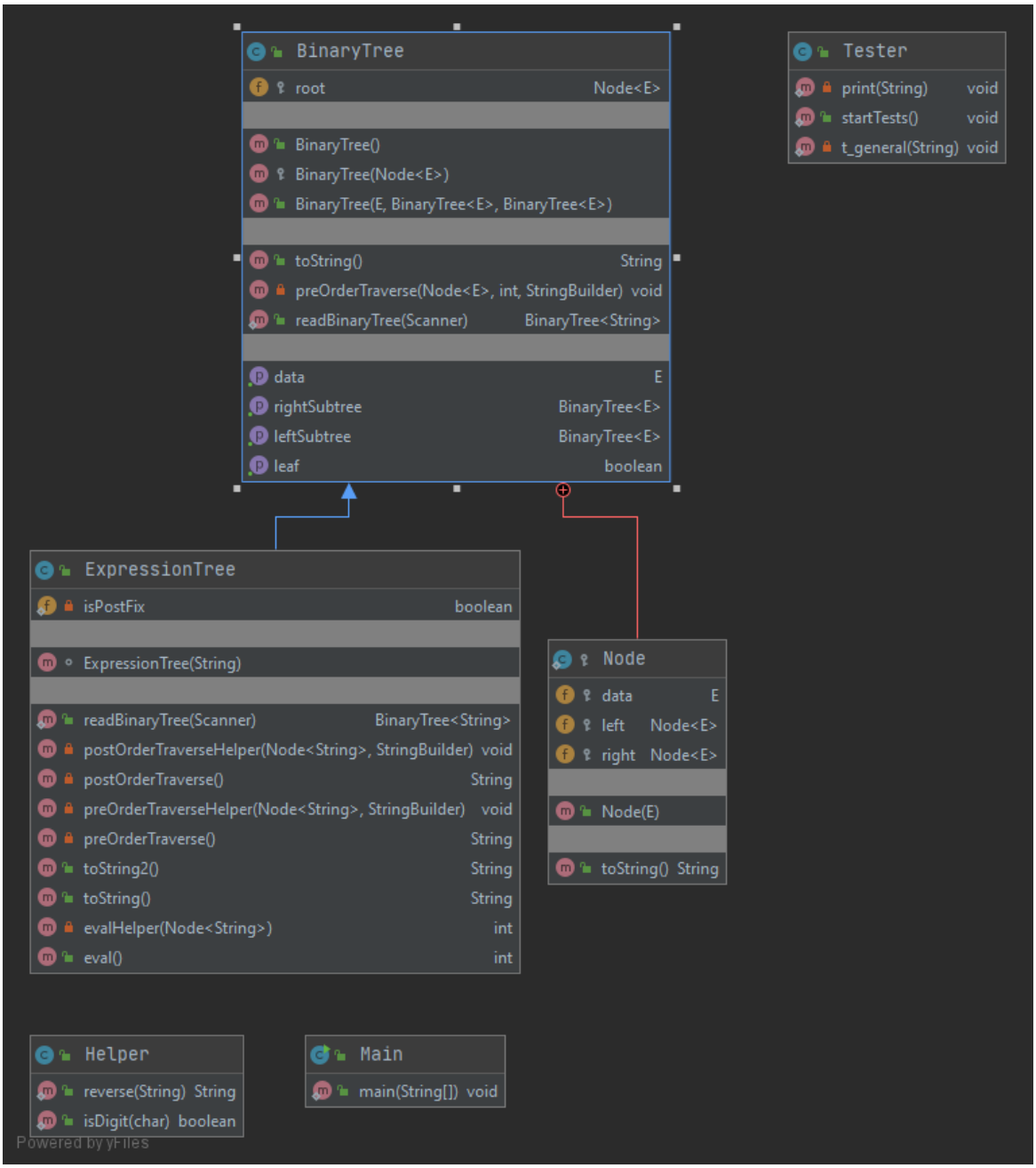
Homework 4 Report

Q2

Buğra Eren Yılmaz

1801042669

1. Class Diagrams



2.Problem Solution approach

The problem at hand was to create expression trees from either a Postfix or Prefix expression string.

I extended the BinaryTree implementation from the book, and build upon it.

First of I kind of made an recursive algorithm for constructing Prefix expressions without using stacks, which just utilizes 1 frame of process and constructs the prefix expression.

After that, to make it work with Postfix expressions I made the Helper class which just holds a sentence reverser method, which reverses given sentence, word wise. And then used that to reverse the expression and apply the prefix solution with little to no change.

The tree traversals was as expected, implemented them for printing and debugging

mainly. But also they serve the purpose of searching.

For evaluating a expression tree, I came up with a recursive solution.

The related errors are mostly non-valid system paths.

3.Test Cases

Test ID	Test Scenario	Test Steps	Test Data	Expected Results	Actual Results	Pass/Fail
T00	Get the expression string and do pre-post order prints and evaluations	Create the expression tree from given string, call the toString and toString2 in order, then evaluate the tree	Tree constructed from given expression string (post or pre, does not matter)	Correct evaluation value. If it is prefix then the pre order will be same with the expression tree, and vice versa for the post fix.	As expected	pass

4.Running results

Testing for expression: + + 10 * 5 15 20

toString (pre-order): + + 10 * 5 15 20

toString2 (post-order): 10 5 15 * + 20 +

eval: 105

Testing for expression: 10 5 15 * + 20 +

toString (pre-order): + + 10 * 5 15 20

toString2 (post-order): 10 5 15 * + 20 +

eval: 105

Testing for expression: + 5 2

toString (pre-order): + 5 2

toString2 (post-order): 5 2 +

eval: 7

Testing for expression: 5 2 +

toString (pre-order): + 5 2

toString2 (post-order): 5 2 +

eval: 7