Lab 11: Expression Trees

Professor: Ronaldo Menezes

TA: Ivan Bogun

Department of Computer Science
Florida Institute of Technology

October 28, 2013

1 General description

1.1 Legend

Expression trees are binary trees used to evaluate complex expressions [1] [2]. The problem is, given an expression in the standart form, transform it into a postfix notation [3] and build a binary expression tree.

1.2 Example

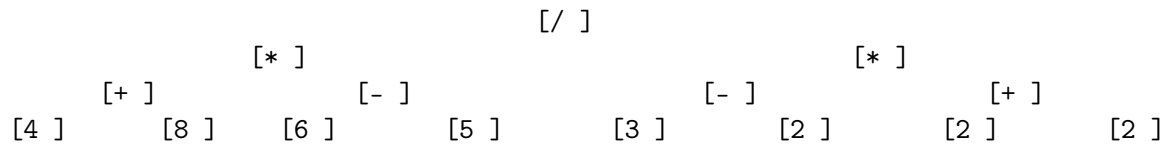
Assume the expression is:

$$(4 + 8) * (6 - 5)/((3 - 2) * (2 + 2)).$$

It's postfix notation is given by:

$$48 + 65 - *32 - 22 + */$$

Expression tree is:



2 Implementation

Reuse the class *BST.java* and implement *Expression.java*

```
// Expression.java
public class Expression {

        private String expression; // expression in the standart form

        public String toPostfix(){
        // get expression in the postfix notation
        }

        public BST getBinaryExpressionTree(){
        // create Binare Expression tree
        }

        }
```

In this lab you are only allowed to use the following import: *java.util.Stack*.

[1] http://en.wikipedia.org/wiki/Binary_expression_tree.
[2] More complex applications can be found like expression trees in C#
[3] http://en.wikipedia.org/wiki/Reverse_Polish_notation

3 Sample input-output

3.1 Input

Use the following main for testing. *showTree()* implementation is not requred. Instead of it print out inorder tree traversal.

```
	public static void main(String[] args) {
		// TODO Auto-generated method stub
		String s="(a+b)/(c+d)+(e-f-g)/(h+j)";
		Expression e=new Expression(s);

		String res=e.toPostfix();
		System.out.println(res);

		BST bst=e.getBinaryExpressionTree();
		bst.showTree();
	}
```

3.2 Output

```
ab+cd+/ef-g-hj+/+
                                        [+ ]
                    [/ ]                                        [/ ]
          [+ ]                [+ ]                    [- ]                    [+ ]
    [a ]      [b ]      [c ]      [d ]          [- ]      [g ]      [h ]        [j ]
                                          [e ]      [f ]
```

4 Grade breakdown

basis	grade
postfix form	(60)
expression tree	30
Hash table	30
Comments	(20)
Javadocs	10
General	10
Overall	(20)
Compiled	5
Style	5
Runtime	10
Total	100