

## XMLTreeNNEvaluationEvaluator.java

```
1 import components.naturalnumber.NaturalNumber;
2 import components.naturalnumber.NaturalNumber2;
3 import components.simplereader.SimpleReader;
4 import components.simplereader.SimpleReader1L;
5 import components.simplewriter.SimpleWriter;
6 import components.simplewriter.SimpleWriter1L;
7 import components.xmltree.XMLTree;
8 import components.xmltree.XMLTree1;
9
10 /**
11  * Program to evaluate XMLTree expressions of {@code NaturalNumber}.
12  *
13  * @author Kevin Haller
14  *
15  */
16 public final class XMLTreeNNEvaluationEvaluator {
17
18     /**
19      * Private constructor so this utility class cannot be instantiated.
20      */
21     private XMLTreeNNEvaluationEvaluator() {
22     }
23
24     /**
25      * Evaluate the given expression.
26      *
27      * @param exp
28      *      the {@code XMLTree} representing the expression
29      * @return the value of the expression
30      * @requires <pre>
31      * [exp is a subtree of a well-formed XML arithmetic expression] and
32      * [the label of the root of exp is not "expression"]
33      * </pre>
34      * @ensures evaluate = [the value of the expression]
35      */
36     private static NaturalNumber evaluate(XMLTree exp) {
37
38         //Result NaturalNumber
39         NaturalNumber result = new NaturalNumber2(0);
40         //Constant NaturalNumber zero
41         NaturalNumber zero = new NaturalNumber2(0);
42
43         if (exp.label().equals("plus")) {
44             //Plus Operator
45             NaturalNumber childZero = evaluate(exp.child(0));
46             NaturalNumber childOne = evaluate(exp.child(1));
47             result.add(childZero);
48             result.add(childOne);
49         } else if (exp.label().equals("minus")) {
50             //Subtraction Operator
51             NaturalNumber childZero = evaluate(exp.child(0));
52             NaturalNumber childOne = evaluate(exp.child(1));
53
54             //Catch of illegal operation
55             if (childZero.compareTo(childOne) > 0) {
56                 childZero.subtract(childOne);
57                 result.add(childZero);
58             }
59         }
60     }
61 }
```

## XMLTreeNNEvaluationEvaluator.java

```

58         } else {
59             components.utilities.Reporter.fatalErrorToConsole(
60                 "ERROR! Violation of .subtract requires clause. child0 < child1");
61         }
62     }
63     } else if (exp.label().equals("times")) {
64         //Multiplication Operator
65         NaturalNumber childZero = evaluate(exp.child 0);
66         NaturalNumber childOne = evaluate(exp.child 1);
67         childZero.multiply(childOne);
68         result.add(childZero);
69     } else if (exp.label().equals("divide")) {
70         //Division Operator
71         NaturalNumber childZero = evaluate(exp.child 0);
72         NaturalNumber childOne = evaluate(exp.child 1);
73
74         //Catch of illegal operation
75         if (childOne.compareTo(zero) > 0) {
76             childZero.divide(childOne);
77             result.add(childZero);
78         } else {
79             components.utilities.Reporter.fatalErrorToConsole(
80                 "ERROR! Violation of .divide requires clause. child1 <= 0");
81         }
82     } else {
83         //Number
84         NaturalNumber numNode = new NaturalNumber2(
85             exp.attributeValue("value"));
86         result.copyFrom(numNode);
87     }
88
89     return result;
90 }
91
92 /**
93  * Main method.
94  *
95  * @param args
96  *         the command line arguments
97  */
98
99 public static void main(String[] args) {
100     SimpleReader in = new SimpleReader1L();
101     SimpleWriter out = new SimpleWriter1L();
102
103     out.print("Enter the name of an expression XML file: ");
104     String file = in.nextLine();
105     while (!file.equals("")) {
106         XMLTree exp = new XMLTree1(file);
107         out.println(evaluate(exp.child 0));
108         out.print("Enter the name of an expression XML file: ");
109         file = in.nextLine();
110     }
111
112     in.close();
113     out.close();
114 }

```

XMLTreeNNEvaluationEvaluator.java

115

116

117