CSE 2221 – Software 1: Software Components

Lecturer: Nyigel Spann

Project #7: XMLTree Expression Evaluator

The Ohio State University

College of Engineering

Columbus, Ohio

```
import components.naturalnumber.NaturalNumber;
import components.naturalnumber.NaturalNumber2;
import components.simplereader.SimpleReader;
import components.simplereader.SimpleReader1L;
import components.simplewriter.SimpleWriter;
import components.simplewriter.SimpleWriter1L;
import components.utilities.Reporter;
import components.xmltree.XMLTree;
import components.xmltree.XMLTree1;

/**
 * Program to evaluate XMLTree expressions of {@code int}.
 *
 * @author Danny Kan (kan.74@osu.edu)
 */
public final class XMLTreeNNExpressionEvaluator {

  /**
   * Private constructor so this utility class cannot be instantiated.
   */
  private XMLTreeNNExpressionEvaluator() {
  }

  /**
   * Evaluate the given expression.
   *
   * @param exp
   *          the {@code XMLTree} representing the expression
   * @return the value of the expression
```

```
 * @requires <pre>
 * [exp is a subtree of a well-formed XML arithmetic expression]  and
 *  [the label of the root of exp is not "expression"]
 * </pre>
 * @ensures evaluate = [the value of the expression]
 */
private static NaturalNumber evaluate(XMLTree exp) {
    assert exp != null : "Violation of: exp is not null";

    NaturalNumber numericalResult = new NaturalNumber2();
    if (exp.label().equals("number")) {
        int val = Integer.parseInt(exp.attributeValue("value"));
        numericalResult = new NaturalNumber2(val);
    } else {
        /*
         * Determine the value of the first and second child node by
         * evaluating the expression using a recursive method call.
         */
        NaturalNumber leftChildNode = new NaturalNumber2(
                evaluate(exp.child(0)));
        NaturalNumber rightChildNode = new NaturalNumber2(
                evaluate(exp.child(1)));

        if (exp.label().equals("times")) {
            leftChildNode.multiply(rightChildNode);
            numericalResult = leftChildNode;
        } else if (exp.label().equals("divide")) {
            if (rightChildNode.isZero()) {
                /*
```

```
            * If the second child node is 0, print the given error
            * message to the console and terminate the application.
            */
           Reporter.fatalErrorToConsole("ERROR - Cannot divide by 0.");
        }
        leftChildNode.divide(rightChildNode);
        numericalResult = leftChildNode;
    } else if (exp.label().equals("plus")) {
        leftChildNode.add(rightChildNode);
        numericalResult = leftChildNode;
    } else if (exp.label().equals("minus")) {
        /*
         * NaturalNumber (non-negative integers).
         */
        if (leftChildNode.compareTo(rightChildNode) < 0) {
            /*
             * When the right child node is greater than the left child
             * node, subtraction of right from left will result in the
             * evaluation of a negative conditional expression.
             *
             * ---> NaturalNumber MUST be non-negative integers.
             *
             * Hence, print the given error message to the console and
             * terminate the application.
             */
            Reporter.fatalErrorToConsole(
                    "ERROR - NaturalNumber MUST be non-negative.");
        }
        leftChildNode.subtract(rightChildNode);
```

```
            numericalResult = leftChildNode;
          }
        }
        return numericalResult;
    }

    /**
     * Main method.
     *
     * @param args
     *          the command line arguments
     */
    public static void main(String[] args) {
        SimpleReader in = new SimpleReader1L();
        SimpleWriter out = new SimpleWriter1L();

        out.print("Enter the name of an expression XML file: ");
        String file = in.nextLine();
        while (!file.equals("")) {
            XMLTree exp = new XMLTree1(file);
            out.println(evaluate(exp.child(0)));
            out.print("Enter the name of an expression XML file: ");
            file = in.nextLine();
        }

        in.close();
        out.close();
    }
}
```