```java
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 Jeng Zhuang
 *
 */
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]
     */
```

```
37      private static NaturalNumber evaluate(XMLTree exp) {
38          assert exp != null : "Violation of: exp is not null";
39
40          // Variable to store the result
41          NaturalNumber num = new NaturalNumber2();
42
43          // Base case: if the current node is a number, return
    its value
44          if (exp.label().equals("number")) {
45              // Get the value attribute
46              String valueStr = exp.attributeValue("value");
47              // Convert the value to an integer
48              int value = Integer.parseInt(valueStr);
49              // Set the NaturalNumber value
50              num.setFromInt(value);
51              return num;
52          } else {
53
54              // Recursive case: evaluate the left and right
    subtrees
55              // Evaluate the left child
56              NaturalNumber left = evaluate(exp.child(0));
57              // Evaluate the right child
58              NaturalNumber right = evaluate(exp.child(1));
59              // Variable to store the result of the operation
60              NaturalNumber result = new NaturalNumber2();
61
62              // Determine the operator and compute the result
63              String operator = exp.label();
64              if (operator.equals("plus")) {
65                  result.copyFrom(left); // Copy the left operand
66                  result.add(right); // Perform addition
67
68              } else if (operator.equals("minus")) {
69                  // Check if subtraction would result in a
    negative number
70                  if (left.compareTo(right) < 0) {
71                      Reporter.fatalErrorToConsole("Subtraction
    result would be negative.");
```

```
 72                 }
 73                 result.copyFrom(left); // Copy the left operand
 74                 result.subtract(right); // Perform subtraction
 75
 76             } else if (operator.equals("times")) {
 77                 result.copyFrom(left); // Copy the left operand
 78                 result.multiply(right); // Perform
    multiplication
 79
 80             } else if (operator.equals("divide")) {
 81                 // Check for division by zero
 82                 if (right.isZero()) {
 83                     Reporter.fatalErrorToConsole("Division by
    zero.");
 84                 }
 85                 result.copyFrom(left); // Copy the left operand
 86                 result.divide(right); // Perform division
 87             }
 88             return result;
 89         }
 90     }
 91
 92     /**
 93      * Main method.
 94      *
 95      * @param args
 96      *             the command line arguments
 97      */
 98     public static void main(String[] args) {
 99         SimpleReader in = new SimpleReader1L();
100         SimpleWriter out = new SimpleWriter1L();
101
102         out.print("Enter the name of an expression XML file:
    ");
103         String file = in.nextLine();
104         while (!file.equals("")) {
105             XMLTree exp = new XMLTree1(file);
106             out.println(evaluate(exp.child(0)));
107             out.print("Enter the name of an expression XML
```

```
    file: ");
108             file = in.nextLine();
109         }
110
111         in.close();
112         out.close();
113     }
114
115 }
116
```