```java
import components.simplereader.SimpleReader;
import components.simplereader.SimpleReader1L;
import components.simplewriter.SimpleWriter;
import components.simplewriter.SimpleWriter1L;
import components.xmltree.XMLTree;
import components.xmltree.XMLTree1;

/**
 * Program to evaluate XMLTree expressions of {@code int}.
 *
 * @author Robert Frenken
 *
 */
public final class XMLTreeIntExpressionEvaluator {

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

    /**
     * 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 int evaluate(XMLTree exp) {
        assert exp != null : "Violation of: exp is not null";

        int num = 0;
        // base case
        if (exp.label().equals("number")) {
            String val = exp.attributeValue("value");
            num = Integer.parseInt(val);

        } else {
            int first = 0;
            int second = 0;
            if (exp.numberOfChildren() > 1) {
                first = evaluate(exp.child(0));
                second = evaluate(exp.child(1));

                // determine operation
                if (exp.label().equals("plus")) {
                    num = first + second;
                } else if (exp.label().equals("minus")) {
                    num = first - second;
                } else if (exp.label().equals("times")) {
                    num = first * second;
                } else {
```

```
58                      num = first / second;
59                  }
60              } else {
61                  first = evaluate(exp.child(0));
62                  num = first;
63              }
64
65          }
66
67          return num;
68      }
69
70      /**
71       * Main method.
72       *
73       * @param args
74       *            the command line arguments
75       */
76      public static void main(String[] args) {
77          SimpleReader in = new SimpleReader1L();
78          SimpleWriter out = new SimpleWriter1L();
79
80          out.print("Enter the name of an expression XML file: ");
81          String file = in.nextLine();
82          while (!file.equals("")) {
83              XMLTree exp = new XMLTree1(file);
84              out.println(evaluate(exp.child(0)));
85              out.print("Enter the name of an expression XML file: ");
86              file = in.nextLine();
87          }
88
89          in.close();
90          out.close();
91      }
92
93  }
```