```java
import components.simplereader.SimpleReader;

/**
 * Program to evaluate XMLTree expressions of {@code int}.
 *
 * @author Yiming Cheng
 *
 */
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 evl = 0;

        if (exp.numberOfChildren() > 0) {
            //find the xml's label which is times, and do the corresponding actions
            if (exp.label().equals("times")) {
                evl = evaluate(exp.child(1)) * evaluate(exp.child(0));
                //find the xml's label which is divide, and do the corresponding actions
            } else if (exp.label().equals("divide")) {
                evl = evaluate(exp.child(0)) / evaluate(exp.child(1));
                //find the xml's label which is plus, and do the corresponding actions
            } else if (exp.label().equals("plus")) {
                evl = evaluate(exp.child(1)) + evaluate(exp.child(0));
                //find the xml's label which is minus, and do the corresponding actions
            } else if (exp.label().equals("minus")) {
                evl = -evaluate(exp.child(1)) + evaluate(exp.child(0));
            }
        } else {
            // this one would be run when the there is no subtree for exp to do actions
            String str1 = exp.attributeValue("value");
            evl = Integer.parseInt(str1);
        }

        return evl;
    }

    /**
     * Main method.
     *
     * @param args
     *            the command line arguments
     */
    public static void main(String[] args) {
```

```
68          SimpleReader in = new SimpleReader1L();
69          SimpleWriter out = new SimpleWriter1L();
70
71          out.print("Enter the name of an expression XML file: ");
72          String file = in.nextLine();
73          while (!file.equals("")) {
74              XMLTree exp = new XMLTree1(file);
75              out.println(evaluate(exp.child(0)));
76              out.print("Enter the name of an expression XML file: ");
77              file = in.nextLine();
78          }
79
80          in.close();
81          out.close();
82      }
83
84  }
85
```