```java
 1 import components.simplereader.SimpleReader;
 8
 9 /**
10  * Program to evaluate XMLTree expressions of {@code int}.
11  *
12  * @author Vaishnavi Kasabwala
13  *
14  */
15
16 public final class XMLTreeIntExpressionEvaluator {
17
18     /**
19      * Private constructor so this utility class cannot be instantiated.
20      */
21     private XMLTreeIntExpressionEvaluator() {
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 int evaluate(XMLTree exp) {
37         assert exp != null : "Violation of: exp is not null";
38
39         int solution = 0;
40
41         // digit
42         if (exp.label().equals("number")) {
43             solution = Integer.parseInt(exp.attributeValue("value"));
44         }
45         // +
46         else if (exp.label().equals("plus")) {
47             solution = evaluate(exp.child(0)) + evaluate(exp.child(1));
48         }
49         // -
50         else if (exp.label().equals("minus")) {
51             solution = evaluate(exp.child(0)) - evaluate(exp.child(1));
52         }
53         // *
54         else if (exp.label().equals("times")) {
55             solution = evaluate(exp.child(0)) * evaluate(exp.child(1));
56         }
57         // "/"
58         else if (exp.label().equals("divide")) {
59             int dividend = evaluate(exp.child(1));
60             if (dividend == 0) {
61                 Reporter.fatalErrorToConsole("Error: Dividing by zero");
62             }
63             solution = evaluate(exp.child(0)) / dividend;
```

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