```java
import components.simplereader.SimpleReader;

/**
 * This program illustrates the charming theory.
 *
 * @author Ansh Pachauri
 *
 */
public class ABCDGuesser2 {
    /**
     * Repeatedly asks the user for a positive real number until the user enters
     * one. Returns the positive real number.
     *
     * @param in
     *            the input stream
     * @param out
     *            the output stream
     * @return a positive real number entered by the user
     */
    private static double getPositiveDouble(SimpleReader in, SimpleWriter out) {
        double num = -1.0;
        while (num <= 0) {
            out.print("enter a positive number: ");
            String input = in.nextLine();
            if (FormatChecker.canParseDouble(input)) {
                num = Double.parseDouble(input);
            } else {
                out.println("enter a positive number ");
            }
        }
        return num;
    }

    /**
     * Repeatedly asks the user for a positive real number not equal to 1.0
     * until the user enters one. Returns the positive real
```

```
   number.
41      *
42      * @param in
43      *             the input stream
44      * @param out
45      *             the output stream
46      * @return a positive real number not equal to 1.0 entered
   by the user
47      */
48     private static double getPositiveDoubleNotOne(SimpleReader
   in,
49             SimpleWriter out) {
50
51         double num = -1.0;
52         while (num <= 1) {
53             out.print("enter a positive number other than 1 ");
54             String input = in.nextLine();
55             if (FormatChecker.canParseDouble(input)) {
56                 num = Double.parseDouble(input);
57             } else {
58                 out.println("enter a positive number ");
59             }
60         }
61         return num;
62
63     }
64
65     private static double getError(double mu, double w, double
   x, double y,
66             double z, double a, double b, double c, double d) {
67         int i = 0, j = 0, k = 0, l = 0;
68
69         double[] array1 = { -5, -4, -3, -2, -1, -1.0 / 2,
   -1.0 / 3, -1.0 / 4, 0,
70                 1.0 / 4, 1.0 / 3, 1.0 / 2, 1, 2, 3, 4, 5 };
71
72         double estimate = (Math.pow(w, a) * Math.pow(x, b) *
   Math.pow(y, c)
73                 * Math.pow(z, d));
```

```
74          double error = Math.abs(estimate - mu) / mu;
75          for (error = Math.abs(estimate - mu) / mu; error >
    0.01;) {
76              for (i = 0; i < 17; i++) {
77                  b = array1[i];
78                  estimate = (Math.pow(w, a) * Math.pow(x, b) *
    Math.pow(y, c)
79                          * Math.pow(z, d));
80                  error = Math.abs(estimate - mu) / mu;
81                  i++;
82                  j = 0;
83                  for (j = 0; j < 17; j++) {
84                      b = array1[j];
85                      estimate = (Math.pow(w, a) * Math.pow(x, b)
    * Math.pow(y, c)
86                              * Math.pow(z, d));
87                      error = Math.abs(estimate - mu) / mu;
88                      j++;
89                      k = 0;
90                      for (k = 0; k < 17; k++) {
91                          c = array1[k];
92                          estimate = (Math.pow(w, a) *
    Math.pow(x, b)
93                                  * Math.pow(y, c) * Math.pow(z,
    d));
94                          error = Math.abs(estimate - mu) / mu;
95                          k++;
96                          l = 0;
97                          for (l = 0; l < 17; l++) {
98                              d = array1[l];
99                              estimate = (Math.pow(w, a) *
    Math.pow(x, b)
100                                     * Math.pow(y, c) *
    Math.pow(z, d));
101                             error = Math.abs(estimate - mu) /
    mu;
102                             l++;
103                         }
104                     }
```

```
105                 }
106             }
107         }
108         return estimate;
109 
110     }
111 
112     /**
113      * main program.
114      *
115      * @param args
116      */
117     public static void main(String[] args) {
118         SimpleReader in = new SimpleReader1L();
119         SimpleWriter out = new SimpleWriter1L();
120         out.print("For the value of μ ");
121         double mu = getPositiveDouble(in, out);
122         out.print("For the value of w ");
123         double w = getPositiveDoubleNotOne(in, out);
124         out.print("For the value of x ");
125         double x = getPositiveDoubleNotOne(in, out);
126         out.print("For the value of y ");
127         double y = getPositiveDoubleNotOne(in, out);
128         out.print("For the value of z ");
129         double z = getPositiveDoubleNotOne(in, out);
130         double a = 0, b = 0, c = 0, d = 0;
131         double estimate = getError(mu, w, x, y, z, a, b, c, d);
132         out.println("the answer is " + estimate);
133         double error = (Math.abs(estimate - mu) / mu) * 100;
134         out.println("the value of a, b, c, d " + a + " " + b +
    " " + c + " " + d
135                 + "and the error percentage is " + error +
    "%");
136 
137     }
138 
139 }
140 
```