

```
1 import components.simplereader.SimpleReader;
2 import components.simplereader.SimpleReader1L;
3 import components.simplewriter.SimpleWriter;
4 import components.simplewriter.SimpleWriter1L;
5 import components.utilities.FormatChecker;
6
7 /**
8  * This program illustrates the charming theory.
9  *
10 * @author Ansh Pachauri
11 *
12 */
13 public class ABCDGuesser1 {
14     /**
15      * Repeatedly asks the user for a positive real
16      * number until the user enters
17      * one. Returns the positive real number.
18      *
19      * @param in
20      *         the input stream
21      * @param out
22      *         the output stream
23      * @return a positive real number entered by the
24      *         user
25      */
26     private static double
27     getPositiveDouble(SimpleReader in, SimpleWriter out) {
28         double num = -1.0;
29         while (num <= 0) {
30             out.print("enter a positive number: ");
31             String input = in.nextLine();
32             if (FormatChecker.canParseDouble(input)) {
33                 num = Double.parseDouble(input);
34             } else {
```

```
32         out.println("enter a positive number
33     ");
34     }
35     return num;
36 }
37
38 /**
39  * Repeatedly asks the user for a positive real
40  * number not equal to 1.0
41  * until the user enters one. Returns the positive
42  * real number.
43  *
44  * @param in
45  *         the input stream
46  * @param out
47  *         the output stream
48  * @return a positive real number not equal to 1.0
49  *         entered by the user
50  */
51 private static double
52 getPositiveDoubleNotOne(SimpleReader in,
53                         SimpleWriter out) {
54     double num = -1.0;
55     while (num <= 1) {
56         out.print("enter a positive number other
57         than 1 ");
58         String input = in.nextLine();
59         if (FormatChecker.canParseDouble(input)) {
60             num = Double.parseDouble(input);
61         } else {
62             out.println("enter a positive number
63             ");
64         }
65     }
66 }
```

```
59     }
60 }
61     return num;
62
63 }
64
65 /**
66  * main program.
67  *
68  * @param args
69  */
70 public static void main(String[] args) {
71     SimpleReader in = new SimpleReader1L();
72     SimpleWriter out = new SimpleWriter1L();
73     out.print("For the value of  $\mu$  ");
74     double mu = getPositiveDouble(in, out);
75     out.print("For the value of w ");
76     double w = getPositiveDoubleNotOne(in, out);
77     out.print("For the value of x ");
78     double x = getPositiveDoubleNotOne(in, out);
79     out.print("For the value of y ");
80     double y = getPositiveDoubleNotOne(in, out);
81     out.print("For the value of z ");
82     double z = getPositiveDoubleNotOne(in, out);
83     double[] array1 = { -5, -4, -3, -2, -1, -1.0 /
84 2, -1.0 / 3, -1.0 / 4, 0,
85 1.0 / 4, 1.0 / 3, 1.0 / 2, 1, 2, 3, 4,
86 5 };
87     double a = 0, b = 0, c = 0, d = 0;
88     double estimate = (Math.pow(w, a) *
89 Math.pow(x, b) * Math.pow(y, c)
90 * Math.pow(z, d));
91     int i = 0, j = 0, k = 0, l = 0;
```

```
90         while ((Math.abs(estimate - mu) / mu) > 0.01)
91         {
92             while ((i < array1.length)
93                 && ((Math.abs(estimate - mu) / mu)
94 > 0.01)) {
95                 b = array1[i];
96                 estimate = (Math.pow(w, a) *
97 Math.pow(x, b) * Math.pow(y, c)
98                 * Math.pow(z, d));
99                 i++;
100                j = 0;
101                while ((j < array1.length)
102                    && ((Math.abs(estimate -
103 mu) / mu) > 0.01)) {
104                    b = array1[j];
105                    estimate = (Math.pow(w, a) *
106 Math.pow(x, b) * Math.pow(y, c)
107                    * Math.pow(z, d));
108                    j++;
109                    k = 0;
110                    while ((k < array1.length)
111                        && ((Math.abs(estimate -
112 mu) / mu) > 0.01)) {
113                        c = array1[k];
114                        estimate = (Math.pow(w, a) *
115 Math.pow(x, b)
116                        * Math.pow(y, c) *
117 Math.pow(z, d));
118                        k++;
119                        l = 0;
120                        while ((l < array1.length)
121                            && ((Math.abs(estimate
122 - mu) / mu) > 0.01)) {
123                            d = array1[l];
```

```
115         estimate = (Math.pow(w, a)
116         * Math.pow(x, b)
117         * Math.pow(y, c) *
118         Math.pow(z, d));
119         l++;
120     }
121 }
122 }
123 out.println("the answer is " + estimate);
124 double error = (Math.abs(estimate - mu) / mu)
125 * 100;
126 out.println("the value of a, b, c, d " + a + "
127 " + b + " " + c + " " + d
128 + "and the error percentage is " +
129 error + "%");
130 }
131 }
```