```java
import components.simplereader.SimpleReader;

/**
 * The users are asked to type a constant number and 4 numbers which are
 * meaningful. When the list of the numbers are provided, the meaningful numbers
 * would combine with the list of the numbers to get the constant number that
 * are shown above.
 *
 * @author Yiming Cheng
 */
public final class ABCDGuesser1 {

    /**
     * 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) {
        String positive = in.nextLine();

        while (!(FormatChecker.canParseDouble(positive))) {
            out.println("Type a mathamatical constant");
            positive = in.nextLine();
        }
        double positiveNumber = Double.parseDouble(positive);
        while (positiveNumber > 0) {
            positive = in.nextLine();
            positiveNumber = Double.parseDouble(positive);
        }

        return positiveNumber;
    }

    /**
     * Repeatedly asks the user for a positive real number not equal to 1.0
     * 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 not equal to 1.0 entered by the user
     */
    private static double getPositiveDoubleNotOne(SimpleReader in,
            SimpleWriter out) {
        String real = in.nextLine();
        while (!(FormatChecker.canParseDouble(real))) {
            out.println("Type a numbers which is meaningful to you");
            real = in.nextLine();
        }
        double realNumber = Double.parseDouble(real);
        while (realNumber == 1.0) {
            real = in.nextLine();
            realNumber = Double.parseDouble(real);
        }
        return realNumber;

```

```
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          out.println("Type a mathamatical constant");
79          double u = getPositiveDouble(in, out);
80          /*
81           * People would be asked to type the numbers which are meaningful to
82           * them.
83           */
84          out.println("Type a numbers which is meaningful to you");
85          double w = getPositiveDoubleNotOne(in, out);
86          out.println("Type a numbers which is meaningful to you");
87          double x = getPositiveDoubleNotOne(in, out);
88          out.println("Type a numbers which is meaningful to you");
89          double y = getPositiveDoubleNotOne(in, out);
90          out.println("Type a numbers which is meaningful to you");
91          double z = getPositiveDoubleNotOne(in, out);
92          /*
93           * The list of numbers could combined with 4 meaningful numbers.
94           */
95          final double[] seriesNumber = { -5, -4, -3, -2, -1, -1.0 / 2, -1.0 / 3,
96                  -1.0 / 4, 0, 1.0 / 4, 1.0 / 3, 1.0 / 2, 1, 2, 3, 4, 5 };
97          double estimate = 0;
98          /*
99           * The numbers which are used to find the closer number for the constant
100          * number
101          */
102         int a = 0;
103         int b = 0;
104         int c = 0;
105         int d = 0;
106         /*
107          * The numbers which is related to the closet number are shown.
108          */
109         double fA = 0;
110         double fB = 0;
111         double fC = 0;
112         double fD = 0;
113         final int percentage = 100;
114         /*
115          * Find the closest number to the number which the users type.
116          */
117         while (a < seriesNumber.length) {
118             double num1 = Math.pow(w, seriesNumber[a]);
119             a++;
120             b = 0;
121             while (b < seriesNumber.length) {
122                 double num2 = Math.pow(x, seriesNumber[b]);
123                 b++;
124                 c = 0;
125                 while (c < seriesNumber.length) {
126                     double num3 = Math.pow(y, seriesNumber[c]);
127                     c++;
128                     d = 0;
```

```
129                     /*
130                      * find the estimate number which is the closest to the
131                      * number.
132                      */
133                     while (d < seriesNumber.length) {
134                         double num4 = Math.pow(z, seriesNumber[d]);
135                         double estimateNumber = num1 * num2 * num3 * num4;
136                         while (Math.abs(u - estimate) > Math
137                                 .abs(u - estimateNumber)) {
138                             estimate = estimateNumber;
139                             fA = seriesNumber[a - 1];
140                             fB = seriesNumber[b - 1];
141                             fC = seriesNumber[c - 1];
142                             fD = seriesNumber[d - 1];
143                         }
144                         d++;
145                     }
146                 }
147             }
148         }
149         out.println(fA);
150         out.println(fB);
151         out.println(fC);
152         out.println(fD);
153         out.println("The closest number would be " + estimate);
154         /*
155          * Print the relative error which is
156          */
157         out.println("The relative error would be "
158                 + Math.abs(percentage * (1 - estimate / u)) + "%.");
159 
160     }
161 }
162 
```