```
import components.simplereader.SimpleReader;
import components.simplereader.SimpleReader1L;
import components.simplewriter.SimpleWriter;
import components.simplewriter.SimpleWriter1L;
import components.utilities.FormatChecker;

/**
 * Project #3: Pseudoscience. The program prompts the user to enter a number and
 * then approximates it using four other numbers of personal meaning within 1%
 * relative error.
 *
 * @author Danny Kan (kan.74@osu.edu)
 * @version 02022022
 */

public final class ABCDGuesser1 {

    /**
     * Private constructor so this utility class cannot be instantiated.
     */
    private ABCDGuesser1() {
    }

    /**
     * Prints dashes to the length of characters in a string.
     *
     * @param str
     *          user input string
     * @param out
     *          the output stream
     */
    private static void printDash(String str, SimpleWriter out) {
        out.print(str + "\n");

        for (int i = 0; i < str.length(); i++) {
            out.print("-");
        }

        out.print("\n");
    }

    /**
     * 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) {
    out.print("Enter your selection here: ");
    String userInput = in.nextLine();
    double mu = 0.0;
    boolean valid = true;
    while (valid) {
        if (FormatChecker.canParseDouble(userInput)) {
            double userValue = Double.parseDouble(userInput);
            if (userValue > 0.0) {
                mu = userValue;
                valid = false;
            } else {
                out.print("Your selection MUST be a positive real number."
                        + "\n");
                out.print("Enter your selection here: ");
                userInput = in.nextLine();
            }
        } else {
            out.print(
                    "Input string cannot be parsed. Your selection MUST be a positive real number."
                            + "\n");
            out.print("Enter your selection here: ");
            userInput = in.nextLine();
        }
    }
    return mu;
}

/**
 * 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 userInput = in.nextLine();
        double value = 0.0;
        boolean valid = true;
        while (valid) {
            if (FormatChecker.canParseDouble(userInput)) {
                double userValue = Double.parseDouble(userInput);
                if (userValue > 0.0 && userValue != 1.0) {
                    value = userValue;
                    valid = false;
                } else {
                    out.print(
                            "Your selection MUST be a positive real number not equal to 1."
                                    + "\n");
                    out.print("Enter your selection here: ");
                    userInput = in.nextLine();
                }
            } else {
                out.print(
                        "Input string cannot be parsed. Your selection MUST be a positive real number not
equal to 1."
                                + "\n");
                out.print("Enter your selection here: ");
                userInput = in.nextLine();
            }
        }
        return value;
    }

    /**
     * Main method.
     *
     * @param args
     *            the command line arguments
     */
    public static void main(String[] args) {
        SimpleReader in = new SimpleReader1L();
        SimpleWriter out = new SimpleWriter1L();

        /*
         * initialize an array.
         */
        double[] myArray = { -5, -4, -3, -2, -1, (double) -1 / 2,
                (double) -1 / 3, (double) -1 / 4, 0, (double) 1 / 4,
                (double) 1 / 3, (double) 1 / 2, 1, 2, 3, 4, 5 };

        String str1 = "Select any positive real-valued universal physical or mathematical constant.";
        /**
         * method call.
         */
        printDash(str1, out);

        double mu = getPositiveDouble(in, out);

        for (int i = 0; i < 3; i++) {
            out.print("\n");
        }

        String str2 = "Select any 4 positive real numbers not equal to 1 that have personal
meaning.";
        /**
         * method call.
         */
        printDash(str2, out);

        /**
         * prompt the user to enter values.
         */
        out.print("Enter your selection here: w = ");
        double wValue = getPositiveDoubleNotOne(in, out);
        out.print("Enter your selection here: x = ");
        double xValue = getPositiveDoubleNotOne(in, out);
        out.print("Enter your selection here: y = ");
        double yValue = getPositiveDoubleNotOne(in, out);
        out.print("Enter your selection here: z = ");
        double zValue = getPositiveDoubleNotOne(in, out);

        int i = 0, length = myArray.length;
        final double RELATIVE_ERROR = 0.01;
        double a = 0.0, b = 0.0, c = 0.0, d = 0.0, bestApprox = 0.0;
        while (i < length) {
            int j = 0;
            while (j < length) {
                int k = 0;
                while (k < length) {
                    int l = 0;
```

```
               while (l < length) {
                  double current = Math.pow(wValue, myArray[i])
                        * Math.pow(xValue, myArray[j])
                        * Math.pow(yValue, myArray[k])
                        * Math.pow(zValue, myArray[l]);
                  if (Math.abs(mu - current) / mu < RELATIVE_ERROR) {
                     bestApprox = current;
                     a = myArray[i];
                     b = myArray[j];
                     c = myArray[k];
                     d = myArray[l];
                  }
                  l++;
               }
               k++;
            }
            j++;
         }
         i++;
      }

      for (int j = 0; j < 3; j++) {
         out.print("\n");
      }

      String str3 = "Results --->";
      printDash(str3, out);
      out.print("Best exponent combination: a = " + a + ", b = " + b
            + ", c = " + c + ", d = " + d + "\n");
      out.print("Best approximation: " + bestApprox + "\n");
      final int CONVERSION = 100;
      double error = CONVERSION * (Math.abs(mu - bestApprox) / mu);
      out.print("Relative error: " + error + "%");

      /**
       * close input and output streams.
       */
      in.close();
      out.close();
   }
}
```