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

/**
 * Asks the user to input a mathematical constant, and which every 4 numbers
 * they wish to input. Then uses the de Jager formula to calculate an
 * approximation of the constant using the 4 numbers the user inputted. Also
 * calculated the percent error between the approximation and the actual
 * constant.
 *
 * @author Shyam Sai Bethina
 *
 */
public final class ABCDGuesser2 {

    /**
     * Private constructor so this utility class cannot be instantiated.
     */
    private ABCDGuesser2() {
    }

    /**
     * Main method.
     *
     * @param args
     *            the command line arguments
     */

    public static void main(String[] args) {
        SimpleReader in = new SimpleReader1L();
        SimpleWriter out = new SimpleWriter1L();
        double[] abcd = new double[] { -5, -4, -3, -2, -1, -(1 / (double) 2),
                -(1 / (double) 3), -(1 / (double) 4), 0, 1 / (double) 4,
                1 / (double) 3, 1 / (double) 2, 1, 2, 3, 4, 5 };

        out.println("Input a constant to be approximated: ");
        double constant = getPositiveDouble(in, out);

        out.println("Input the first positive real number not equal to 1.0: ");
        double w = getPositiveDoubleNotOne(in, out);

        out.println("Input the second positive real number not equal to 1.0: ");
        double x = getPositiveDoubleNotOne(in, out);

        out.println("Input the third positive real number not equal to 1.0: ");
        double y = getPositiveDoubleNotOne(in, out);

        out.println("Input the fourth positive real number not equal to 1.0: ");
        double z = getPositiveDoubleNotOne(in, out);

        int aCounter = 0;
        int bCounter = 0;
        int cCounter = 0;
        int dCounter = 0;

        double a = abcd[aCounter];
        double b = abcd[bCounter];
        double c = abcd[cCounter];
        double d = abcd[cCounter];

        double constantApprox = Math.pow(w, a) * Math.pow(x, b) * Math.pow(y, c)
                * Math.pow(z, d);
        double error = error(constantApprox, constant);
        double finalApprox = constantApprox;
        double finalError = error;

        for (aCounter = 0; aCounter < abcd.length; aCounter++) {
            a = abcd[aCounter];
            for (bCounter = 0; bCounter < abcd.length; bCounter++) {
                b = abcd[bCounter];
                for (cCounter = 0; cCounter < abcd.length; cCounter++) {
                    c = abcd[cCounter];
                    for (dCounter = 0; dCounter < abcd.length; dCounter++) {
                        d = abcd[dCounter];

                        constantApprox = Math.pow(w, a) * Math.pow(x, b)
                                * Math.pow(y, c) * Math.pow(z, d);
                        error = error(constantApprox, constant);

                        if (error < finalError) {
                            finalApprox = constantApprox;
                            finalError = error;

                        }
                    }
                }
            }
        }

        out.println("Final approximation of constant is " + finalApprox);
        out.print("Final error is: ");
        out.print(finalError, 2, false);

    }

    /**
     * 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 userInput = in.nextLine();
        while (!FormatChecker.canParseDouble(userInput)) {
            out.println("Input a positive real number: ");
            userInput = in.nextLine();
```

```java
        }

        return Double.parseDouble(userInput);
    }

    /**
     * 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();
        final double one = 1.0;
        while (!FormatChecker.canParseDouble(userInput)) {
            out.println("Input a real number not equal to 1.0");
            userInput = in.nextLine();
        }

        while (Double.parseDouble(userInput) == one) {
            out.println("Input a real number not equal to 1.0");
            userInput = in.nextLine();
        }

        return Double.parseDouble(userInput);
    }

    /**
     * Calculates the error between the constant approximation and the constant.
     * Returns the resulting error.
     *
     * @param constantApprox
     *            the approximation of the constant
     * @param constant
     *            the original constant the program is
```

```java
approximating to
     * @return an percentage of the error based on the original constant
     */
    private static double error(double constantApprox, double constant) {
        final double hundred = 100;
        double error = (Math.abs(constantApprox - constant) / constant)
                * hundred;

        return error;
    }
}

```