```java
import components.simplereader.SimpleReader;

/**
 * Put a short phrase describing the program here.
 *
 * @author Put your name here
 *
 */
public final class ABCDGuesser2 {

    /**
     * No argument constructor--private to prevent instantiation.
     */
    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[] exp = { -5, -4, -3, -2, -1, -0.5, -0.333, -0.25, 0, 0.25,
                0.333, 0.5, 1, 2, 3, 4, 5 };
        double total = -1, closest = -999999999;
        double bestW = -1, bestX = -1, bestY = -1, bestZ = -1;

        // get u
        double u = getPositiveDouble(in, out);

        // get w x y z
        double w = getPositiveDoubleNotOne(in, out);
        double x = getPositiveDoubleNotOne(in, out);
        double y = getPositiveDoubleNotOne(in, out);
        double z = getPositiveDoubleNotOne(in, out);

        // big boy loop
        for (int i = 0; i <= 16; i++) {
            for (int j = 0; j <= 16; j++) {
                for (int k = 0; k <= 16; k++) {
                    for (int l = 0; l <= 16; l++) {
                        // does calculations
                        total = Math.pow(w, exp[i]);
                        total += Math.pow(x, exp[j]);
                        total += Math.pow(y, exp[k]);
                        total += Math.pow(z, exp[l]);

                        // if total is the closest to u so far
                        if (Math.abs(u - total) < Math.abs(u - closest)) {
                            closest = total;
                            bestW = exp[i];
                            bestX = exp[j];
                            bestY = exp[k];
                            bestZ = exp[l];
                        }
                    }
                }
            }
        }

        // do some math
        double percentError = Math.abs((closest - u) / u * 100);

        // print results
        System.out.println("u = " + u);
        System.out.println("(" + w + "^" + bestW + ")" + " + " + "(" + x + "^"
                + bestX + ")" + " + " + "(" + y + "^" + bestY + ")" + " + "
                + "(" + z + "^" + bestZ + ")" + " = "
                + String.format("%.2f", closest));
        System.out.println(
                "Percent Error: " + String.format("%.2f", percentError) + "%");

        // close stuff
        in.close();
        out.close();
    }

    /**
     * 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) {
        System.out.print("Enter a positive number: ");
        String input = in.nextLine();
        boolean looping = true;
        double num = -1;

        while (looping) {
            if (FormatChecker.canParseInt(input)) {
                num = Integer.parseInt(input);
                if (num > 0) {
                    looping = false;
                }
            } else {
                System.out.print("Enter a positive number: ");
                input = in.nextLine();
            }
        }

        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.
     *
     * @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) {
        boolean looping = true;
        double num = -1;

        while (looping) {
            num = getPositiveDouble(in, out);
            if (num != 1.0) {
                looping = false;
            }
        }

        return num;
    }

}

```