Programming 1 Assignment 4.

Due Date: 11 October 2024.

Submit a single .txt file containing all your solution to Lea.

Question 1:

How many times do you have to roll a pair of dice before they come up snake eyes? You could do the experiment by rolling the dice by hand. Write a computer program that simulates the experiment. The program should report the number of rolls that it makes before the dice come up snake eyes. (Note: "Snake eyes" means that both dice show a value of 1.)

Question 2:

Which integer between 1 and 10000 has the largest number of divisors, and how many divisors does it have? Write a program to find the answers and print out the results. It is possible that several integers in this range have the same, maximum number of divisors. Your program only has to print out one of them.

Appendix 1 on the last page of this assignment contains a sample code that calculates how many divisors a number has

You might need some hints about how to find a maximum value. The basic idea is to go through all the integers, keeping track of the largest number of divisors that you've seen *so far*. Also, keep track of the integer that had that number of divisors.

Question 3:

Write a program that will evaluate simple expressions such as 17 + 3 and 3.14159 * 4.7. The expressions are to be typed in by the user. The input always consists of a number, followed by an operator, followed by another number. The operators that are allowed are +, -, *, and /. Your program should read an expression, print its value, read another expression, print its value, and so on.

The program should end when the user enters 0 as the first number on the line such as 0 + 3 or 0 * 4.7..

Question 4 (4 points) :

Write a program that reads one line of input text and breaks it up into words. The words should be output one per line. A word is defined to be a sequence of letters. Any characters in the input that are not letters should be discarded. For example, if the user inputs the line

```
He said, "That's not a good idea."
```

then the output of the program should be for 2 points

```
He
said
That
s
not
a
good
idea
```

An improved version of the program for the last 2 points would list "that's" as a single word. An apostrophe can be considered to be part of a word if there is a letter on each side of the apostrophe.

To test whether a character is a letter, you might use (ch >= 'a' && ch <= 'z') || (ch >= 'A' && ch <= 'Z'). There are other ways that this can also be achieved.

Appendix 1.

```
/**
 * This program reads a positive integer from the user.
 * It counts how many divisors that number has, and
 * then it prints the result.
 *
 * (Note: This program works for any integer in the range
 * 1 to 2147483646, but it crashes with a division-by-zero
 * when run with input 2147483647, the largest value of type
 * int.  This happens because 2147483647 + 1 is -2147483648,
 * the smallest value of type int.   So, after processing
 * testDivisor = 2147483647, testDivisor becomes -2147483648,
 * and the loop continues.  Eventually, testDivisor will be
 * zero, and the expression  N % testDivisor is trying to
 * divide N by zero.)
 */

public class CountDivisors {

    public static void main(String[] args) {

        int N;  // A positive integer entered by the user.
                // Divisors of this number will be counted.

        int testDivisor;  // A number between 1 and N that is a
                          // possible divisor of N.

        int divisorCount;  // Number of divisors of N that have been found.

        int numberTested;  // Used to count how many possible divisors
                           // of N have been tested.  When the number
                           // reaches 10000000, a period is output and
                           // the value of numberTested is reset to zero.

        /* Get a positive integer from the user. */

        while (true) {
            System.out.print("Enter a positive integer: ");
            N = TextIO.getlnInt();
            if (N > 0)
                break;
            System.out.println("That number is not positive.  Please try again.");
        }

        /* Count the divisors, printing a "." after every 10000000 tests. */

        divisorCount = 0;
        numberTested = 0;

        for (testDivisor = 1; testDivisor <= N; testDivisor++) {
            if ( N % testDivisor == 0 )
                divisorCount++;
            numberTested++;
            if (numberTested == 10000000) {
                System.out.print('.');
                numberTested = 0;
            }
        }

        /* Display the result. */

        System.out.println();
```

```
        System.out.println("The number of divisors of " + N
                + " is " + divisorCount);

    } // end main()

} // end class CountDivisors
```