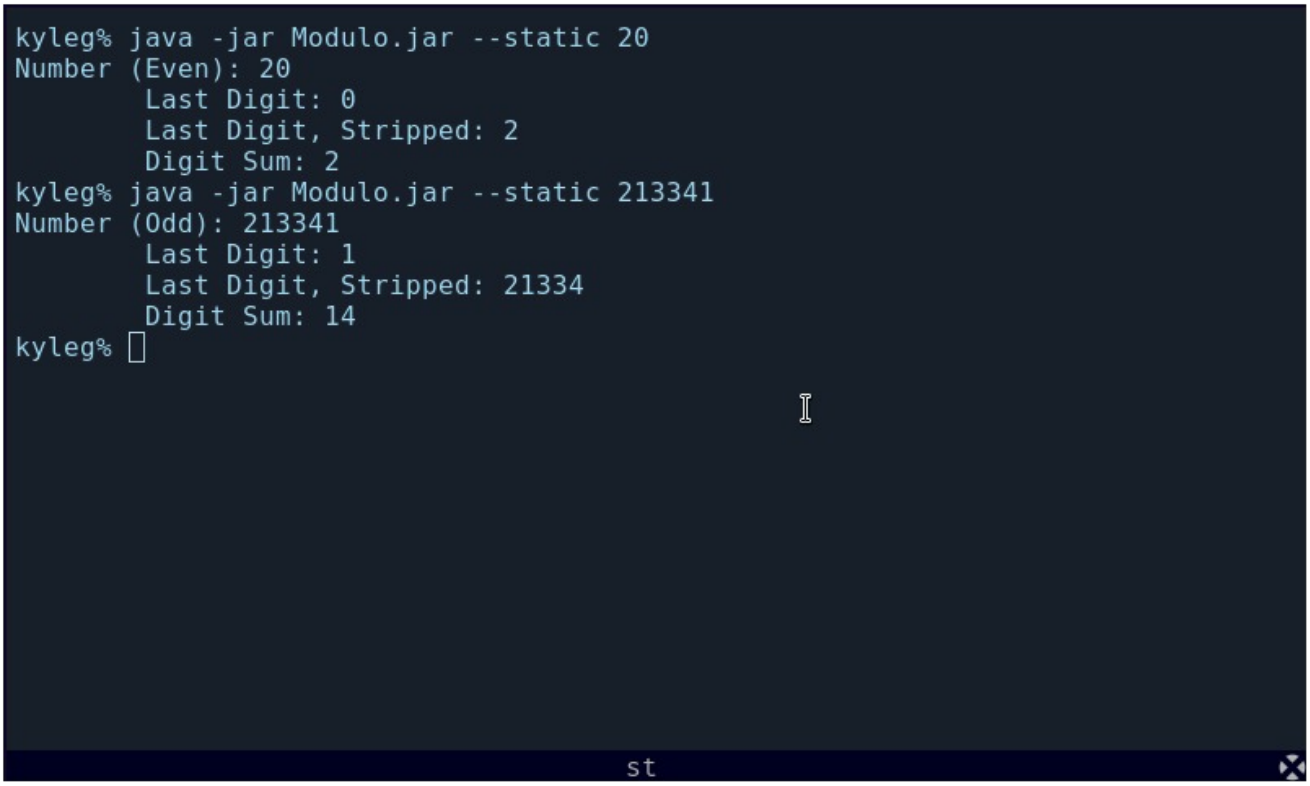
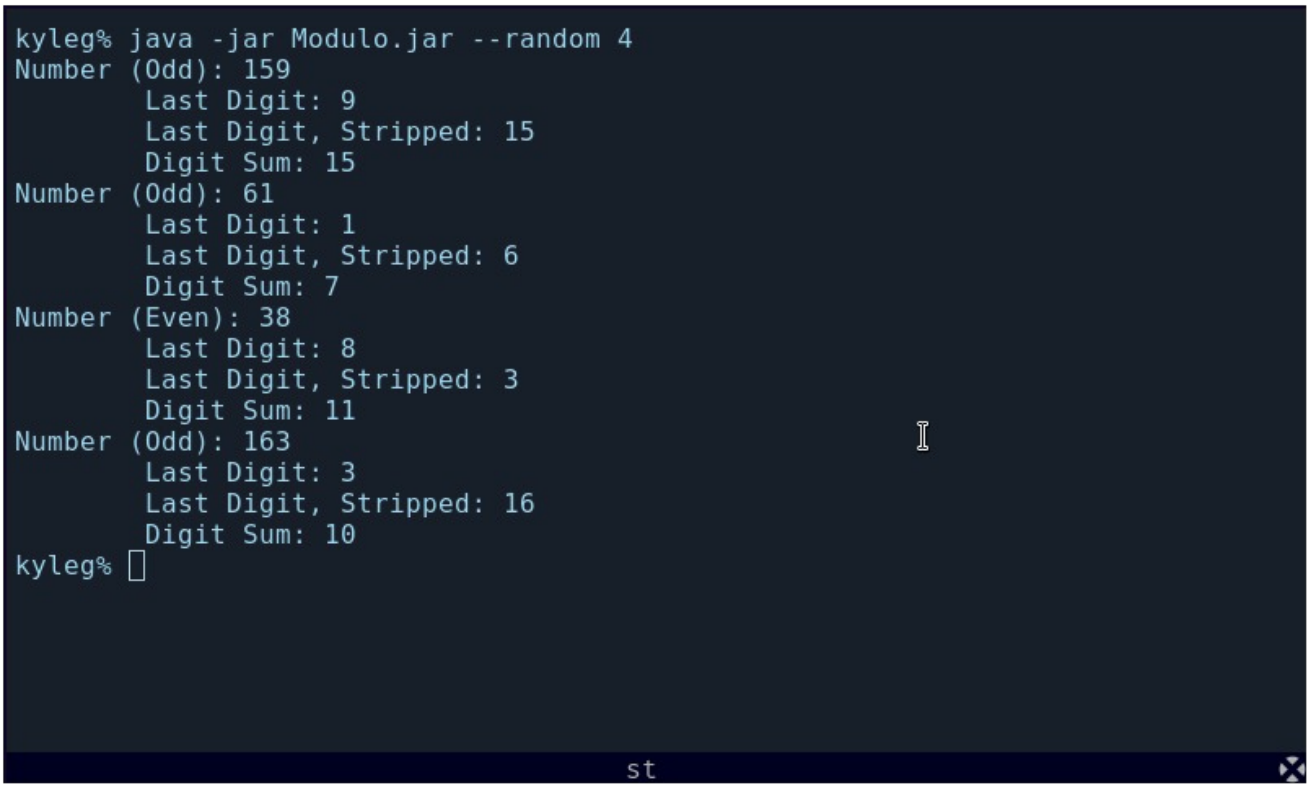
Project 4
Kyle Guarco

Static Number Output



Arguments (Length 2): [“--static”, “<number>”]

Random Number Output



Arguments (Length 2): [“--random”, “<exit condition>”]

TestNumberUtils.java

```
import java.util.Random;

/**
* Plays with numbers, and program arguments.
*
* @author Kyle Guarco
*/
public class TestNumberUtils
{
    public static void main(String[] args)
    {
        // If there aren't two arguments, exit! They're important!
        if (args.length ≠ 2)
            errorOut();

        // Gets the second argument, and hope that it's a number. This is passed
        // onto the later functions, and serves as either an exit condition or
        // a constant number.
        int number = Integer.parseInt(args[1]);

        // Try out a few arguments.
        switch (args[0])
        {
            case "--static":
                testStatic(number);
                break;
            case "--random":
                testRandomizer(number);
                break;
            default:
                errorOut();
                break;
        }
    }

    /** This function is run when the '--static' case is provided */
    private static void testStatic(int number)
    {
        new NumberUtils(number).testAll();
    }

    /** This function is run when the '--random' case is provided */
    private static void testRandomizer(int repeats)
    {
        NumberUtils utils = new NumberUtils();
        Random generator = new Random();

        while (repeats ≠ 0)
        {
            // Range: 1 - 199
            utils.setNumber(generator.nextInt(199) + 1);
            utils.testAll();
```

```
            // Decrement 'repeats' until it reaches zero.
            repeats--;
        }
    }

    /** Exits the program, obnoxiously. */
    private static void errorOut()
    {
        System.out.println("WRONG! ERROR! BAD! AWFUL!");
        System.exit(-1);
    }
}
```

NumberUtils.java

```
/**
* Functions that play with numbers.
*
* @author Kyle Guarco
*/
public class NumberUtils
{
    private int number;

    public NumberUtils(int number)
    {
        this.number = number;
    }

    public NumberUtils()
    {
        this(0);
    }

    public boolean isOdd()
    {
        return number % 2 == 1;
    }

    public boolean isEven()
    {
        return !isOdd();
    }

    public void testAll()
    {
        // Perform the digit sum calculation beforehand.
        Int digitSum = getDigitSum();

        String oddEven = isOdd() ? "Odd" : "Even";

        String result = String.format(
                "Number (%s): %d\n\tLast Digit: %d\n\tLast Digit, Stripped: %d\
```

```
n\tDigit Sum: %d",
                oddEven, number, getLastDigit(number), stripLastDigit(number),
digitSum);

        System.out.println(result);
    }

    private int getLastDigit(int n)
    {
        // Lecture: If 100s, (n % 10) or (n - int(n / 10) * 10)
        // a%b = a-b*floor(a/b)
        return n % 10;
    }

    private int stripLastDigit(int n)
    {
        // Lecture: If 100s, int(n / 10)
        return n / 10;
    }

    private int getDigitSum()
    {
        // Save a copy of the number instance for the calculation.
        int number = this.number;
        int sum = 0;

        while (number > 0)
        {
            sum = getLastDigit(number);
            number = stripLastDigit(number);
        }

        return sum;
    }

    public void setNumber(int n)
    {
        this.number = n;
    }

    public int getNumber()
    {
        return number;
    }
}
```