Exercises: Day 1

Exercise 1:  Create a program that prints out all the prime numbers smaller than \( n \), a number read from the command line, using the sieve of Eratosthenes. Begin by creating an array of \( n \) boolean values and initializing them to be true. Then set the values indexed by multiplies of 2 to false, the multiples of 3 to false and so forth. There are several possible ways to organize your program.

Exercise 2:  There is a Division Algorithm for the Gaussian integers: that is, given Gaussian integers \( a \) and \( b \) where \( b \neq 0 \), there are (not necessarily unique) Gaussian integers \( q \) and \( r \) such that

\[
a = bq + r
\]

\[|r| < |b|\]

In fact, to find \( q \) you may view \( b \) as a complex number and find the closest Gaussian integer to the complex number \( a/b \).

Beginning with the GaussianInteger class we constructed in the lecture, add an instance method

```java
public GaussianInteger[] dividedBy(GaussianInteger b)
```

that returns an array with two GaussianIntegers, \( q \) and \( r \), that result from dividing the Gaussian integer represented by the instance by \( b \). You will want to use the expression `(int) Math.round(double x)` to find the closest integer to a real number \( x \). You may wish to add in a few auxiliary methods to help out.

Also construct a program that reads two GaussianIntegers from the command line and prints the quotient and remainder:

```
$ java DivisionAlgorithm 10 5 1 1
q = 8 + -2 i, r = 0 + -1 i
```

Extra Credit

Exercise 3:  Design a class called Rational that represents a rational number. There should be two constructors,

```java
public Rational(int n)
public Rational(int p, int q)
```

the first of which represents an integer. Include methods, such as `add`, `sub`, `mul`, and `div`, for doing arithmetic with rationals. (You may wish, for the time being, to ignore any problems you might encounter when dividing by 0.)

Include a method to determine when two instances of Rational represent equal rational numbers and another to determine when one rational is larger than another.

Over
Finally, construct a method

```
public boolean isIntegerMultipleOf(Rational r)
```

such that

```
x.isIntegerMultipleOf(r)
```

returns `true` if \( x = nr \) where \( n \) is an integer.

Construct another program to test your `Rational` class.

**Exercise 4:** Implement the Euclidean Algorithm to find the greatest common divisor of two Gaussian integers by creating a class method

```
public static GaussianInteger gcd(GaussianInteger a, GaussianInteger b)
```

Here is a helpful fact: given two Gaussian integers \( a \) and \( b \), and Gaussian integers \( q \) and \( r \) such that

\[
a = bq + r,
\]

the greatest common divisor of \( a \) and \( b \) is the same as the greatest common divisor of \( b \) and \( r \). This fact leads to the Euclidean Algorithm, presented for the integers in Chapter 1 of the notes.

**Exercise 5:** Study the class `java.lang.String` and write a program that reads a String from the command line and writes it backward.

```
$ java Backward Hello
olleH
```