Homework 1

(Due: February 21, 2022)

The purpose of this homework is to get you started with programming in Java and Python. For each question hand in a solution (i.e., program source code + program output) in Java, and then a second solution (i.e., program source code + program output) coded in Python.

Question 1: 10 points

The fragment of code:

```
import java.lang.Math;

public class EulerMath {
   public static boolean isPrime( long num ) {

      if (num < 2 || (num % 2 == 0 && num != 2))
          return false;

      for (int i = 3; i <= Math.sqrt(num); i += 2)
          if (num % i == 0)
              return false;

      return true;
   }
}
```

defines a class called EulerMath. The method `isPrime()` determines whether or not an integer (actually a long integer) `num` is prime. The method will return `true` if `num` is prime; otherwise it will return `false`.

Notice that the method declaration includes the keyword static. This makes `isPrime()` a class method, meaning that it can be called without first having to create an object. To call the method we simply write:

```
EulerMath.isPrime ( ... );
```

and the result with either be true or false. (e.g., EulerMath.isPrime(4) evaluates to false).

Write a Java program to find and print all of the prime numbers less than 1000 in a tidy table. Then, repeat exercise using Python. To see how static methods work in Python, Google: python static method.

Question 2: 10 points

(Brain Teaser): The modulo operator, `%`, computes the remainder that occurs after an integer `m` has been divided by a second integer `n`. For example,

```
m = 5, n = 3, 5 = 1*3 + 2  --> 5%3 evaluates to 2
m = 6, n = 3, 6 = 2*3 + 0  --> 6%3 evaluates to 0
m = 7, n = 3, 7 = 2*3 + 1  --> 7%3 evaluates to 1
m = 8, n = 3, 8 = 2*3 + 3  --> 8%3 evaluates to 2
```

and so forth. It is important to notice that `m%n` will always return an integer between 0 and (n-1). Now let A be a (7×7) matrix whose elements are given by

$$\mathrm{A}(i,j) = \left[(i-1)^2 + (j-1)^2)\right] \%7. \tag{1}$$

Things to do:

1. Write a short Java program to evaluate and print equation 1.

2. Now let `p` and `q` be integers that cover the interval 0 through 100. Extend your Java program to find combinations of `p` and `q` where $p^2 + q^2$ will be divisible by 7.

3. Prove that if $p^2 + q^2$ is divisible by 7, then it will also be divisible by 49.

Hint. The last part of this problem is not as difficult as it looks. Write `p` as $7 * p_1 + r_1$ and `q` as $7 * q_1 + r_2$ and then an expression for $p^2 + q^2$. The result follows directly from the expression and the matrix element values in equation 1.

Question 3: 10 points

Figure 1 is a schematic of an irregular polygon having seven sides.

Suppose that the x and y vertex coordinates are stored as two columns of information in the array

```
float faaPolygon [ 7 ][ 2 ] = { { 1.0, 1.0 },
                                { 1.0, 5.0 },
                                { 6.0, 5.0 },
                                { 7.0, 3.0 },
                                { 4.0, 3.0 },
                                { 3.0, 2.0 },
                                { 3.0, 1.0 } };
```

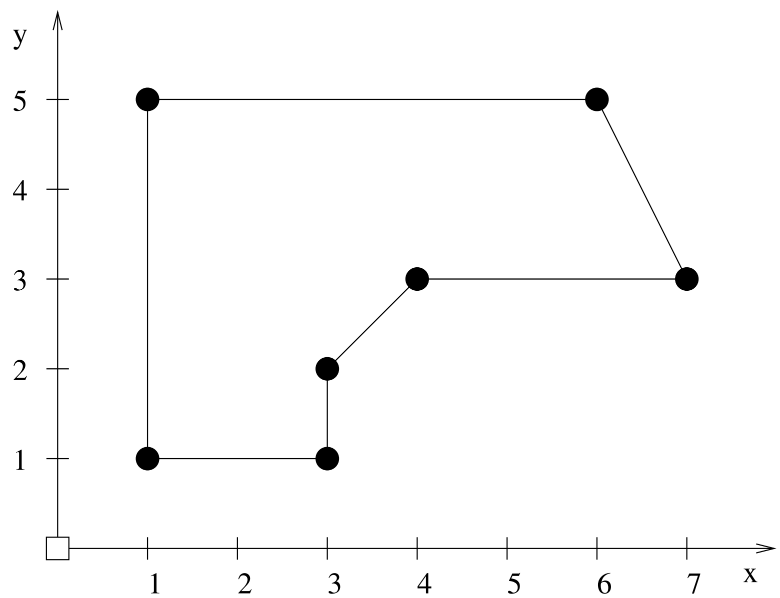

Figure 1: Seven-sided irregular polygon.

Write a Java program that will compute and print

1. The minimum and maximum polygon coordinates in both the x and y directions.
2. The minimum and maximum distance of the polygon vertices from the coordinate system origin.
3. The perimeter and area of the polygon.

Note. For Parts 1 and 2, use the `Math.max()` and `Math.min()` methods in java.lang.Math. In Part 3, use the fact that the vertices have been specified in a clockwise manner.

Question 4: 10 points

Write a program that will print a list of points (x, y) on the graph of the equation

$$y(x) = \left[\frac{x^4 + \left[\frac{x}{sin(x)} \right]}{x - 2} \right] \tag{2}$$

for the range $-4 \leq x \leq 10$ in intervals of 0.25.

Note: You can approach this problem in one of two ways:

1. Detect a numerical problem before it occurs and print out an appropriate message, or

2. Evaluate $y(x)$ for all values of x and then test for various types of error.

You should find that $y(0)$ and $y(2)$ evaluate to not-a-number (NaN) and positive infinity, respectively.

If you have ant working on your computer, create plots of y(x) with JFreeChart.

Note. These quantities can be tested for via the error condition constants `Double.POSITIVE_INFINITY` and `Double.NaN` on the Java side.

Python implements these values in its standard library. For more info and examples Google: python largest double. You might also try creating a plot of y(x) vs x to see how Python handles these error conditions?