Lab 7

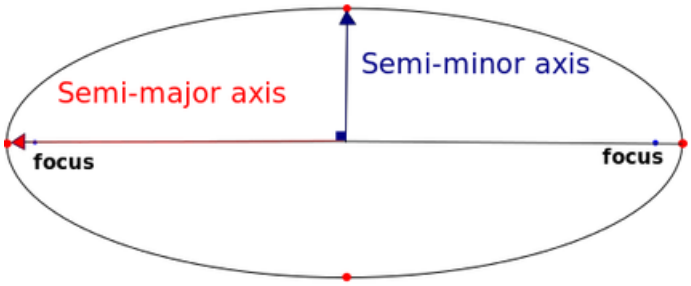
1. The Lesson 5 Demo in `lesson5.lecture.intfaces2` shows how to polymorphically compute the average perimeter of a list of geometric objects by requiring each to implement the `ClosedCurve` interface. Notice that when a closed curve happens to be a polygon, computing the perimeter is especially easy – you just add up the lengths of the sides.

   If we create an interface `Polygon` having method `double[] getSides()` (which will return the length of each side of the polygon in an array), we could replace `ClosedCurve` in our example with an interface `Polygon` – *if* we didn't have to take into account the computation of the perimeter of *non*-polygons, like `Circles`. In this problem, you will find a way to make use of both `ClosedCurve` and `Polygon`.

   Startup code for this problem is in the package lesson7.labs.prob2; it contains classes Circle and Rectangle, the interface ClosedCurve, and a DataMiner class that contains a main method that loads a few of these geometric objects into an array and computes the averagePerimeter. Begin by creating a new `Polygon` interface. Then think of a way to make use of both `ClosedCurve` and `Polygon` so that, when `computePerimeter` is called on one of the geometric objects that implements the `Polygon` interface (like `Rectangle`), the side lengths are added up, but when the object is not a polygon, a different computation of perimeter is done (as in the case of a Circle). *Hint.* Create a `default` method in `Polygon`. The idea is that you try to use the generic computation for computing perimeter, available in Polygon, whenever it is possible.

   Expand your code by adding two new `ClosedCurve`s to your package: `EquilateralTriangle` and `Ellipse` (an equilateral triangle is a triangle in which all side lengths are equal). Modify `DataMiner` so that it includes in the `objects` list instances of these new classes.

   Hint. The perimeter (or circumference) of an ellipse is $4aE$ where a is the length of the semi-major axis and E is the value of the elliptic integral evaluated at the ellipse's eccentricity. You do not need to know these technical concepts; just include a and E as instance variables in your class, of type `double`, and include them as arguments to the `Ellipse` constructor.



2. In the lesson7.labs.prob3 package, there is a class called ForEachExample that specifies, in its main method, a list of Strings. Use the Java 8 forEach method within the main method to print out the list so that *all Strings are in upper case*. To do this, you will need to define your own implementation of the Consumer interface.