The design tries to maximize the amount of code reuse. Each class should utilize the inherited methods to limit the amount of code in each class.

```java
public interface Polygon extends Comparable<Polygon> {  
   /**
    * Returns the number of sides of the polygon.
    * @return The number of sides of the polygon.
    */
   public int numberOfSides();
   /**
    * Returns the perimeter of the polygon.
    * @return The perimeter of the polygon.
    */
   public float perimeter();
   /**
    * Calculates the area of the polygon.
    * @return The area of the polygon.
    */
   public float area();
}
```
Triangle
You can utilize Heron’s Formula to calculate the area from the length of the three sides:

\[ A = \sqrt{s(s-a)(s-b)(s-c)} \]

Where \( s \) is half the triangles perimeter. 

\[ s = \frac{a + b + c}{2} \]
Quadrilateral
This is an abstract class that has a concrete method for the *numberOfSides()*, but the other methods are declared abstract.
Rectangle
This is a subclass of Quadrilateral and must implement all of the inherited abstract methods.
Square
This is a subclass of Rectangle and will have a difference constructor, and possibly override methods?
Trapezoid

This is a subclass of Quadrilateral and must implement all of the inherited abstract methods.

Remember the area is calculated by the average of the two bases multiplied by the height.