Point Class:

- A point is represented as a pair of Cartesian coordinates.
- Instance Variables:
  - x and y representing x and y coordinates (double)
- Instance Methods:
  - A default constructor: sets the x and y coordinates to 0.0
  - Constructor that takes two doubles and initializes x and y
  - A copy constructor: takes a Point object as parameter and creates a new Point object that is an exact copy of the parameter.
  - get methods getX and getY
  - set methods setX and setY
  - distance method that returns the distance between this Point and another point. This method takes a Point object as a parameter and computes the length of the line segment between this Point and the parameter and returns the value as a double. The length of the line segment is defined as Math.sqrt($(this.x - p.x)^2 + (this.y - p.y)^2$) where p is the parameter
  - *print*(PrintWriter out) method that that prints the point in the [x,y] format to the output file

Circle Class:

- A circle **is a** TwoDimensionalShape and is represented by its radius and center (center is a Point object).

- Instance Variables:
  - radius (represented by a *double*) – radius cannot be -ve
  - center which is a Point object

- Instance Methods:
  - Constructor – takes a Point object and a *double* and sets the center and radius
  - Copy constructor that takes a Circle and creates an exact copy (see Point class above)
  - Set methods for radius and center
  - Get methods for radius, center
  - Overridden methods *getArea* and *getPerimeter*
    - area (represented by a *double*) – Formula: Math.PI * radius * radius
    - perimeter (represented by a *double*) – Formula: 2 * Math.PI * radius
  - Overridden *print*(PrintWriter out) method that prints the circle object. See the output file for the format.

Triangle Class:

- A Triangle **is a** TwoDimensionalShape and is represented by its 3 corner points (which are all Point objects).
- Instance Variables:
  - 3 Points p1, p2 and p3 (all Point objects)
- Instance Methods:
  - Constructor – takes 3 Point objects and initializes the three corner points
  - Copy constructor
  - Set methods for 3 corner points – setPoint1, setPoint2, setPoint3 – takes a Point object as parameter and sets the appropriate corner point
  - Get methods for 3 corner points – getPoint1, getPoint2 and getPoint3
  - Overridden methods *getArea* and *getPerimeter*
  - Overridden *print*(PrintWriter out) method that that prints the Triangle object. See the output file for the format.
- See http://www.mathopenref.com/heronsformula.html for formula on how to compute the area of a triangle given 3 sides.
- The perimeter is the length of the 3 line segments – use the distance method from the Point class to compute the length of line segment between each pair of 3 corner points.

Rectangle Class:

Instance Variables:

- length and width (both double)

Instance Methods:

- ◦ Constructor – takes 2 doubles and initializes the length and width
- ◦ Copy constructor
- ◦ set and get methods for length and width
- ◦ Overridden methods *getArea* and *getPerimeter*
- ◦ Overridden *print*(PrintWriter out) method that prints a Rectangle object on the screen. See the output file for the format.
- ◦ *isSquare*() method that returns true if both length and width are the same.
- ◦ See the output file for the format.

Cone Class:

- A Cone **is a** ThreeDimensionalShape and is represented by its radius and height
  - radius (represented by a *double*);
  - height (represented by a *double*)

- Instance Methods:
  - Constructor – takes radius and height
  - Copy constructor
  - Set methods for radius and height
  - Get methods for radius, height
  - Overridden methods *getSurfaceArea* and *getVolume*
  - Overridden *print*(PrintWriter out) method that prints a Cone object. See the output file for the format.
  - getVolume() – Formula $(\pi r^2 h)/3$ (r is the radius and h is the height)
  - getSurfaceArea() – Formula $\pi r * (r + sqrt(r^2 + h^2))$
  - http://chemistry.about.com/od/chemistry101/ss/3dformulas_2.htm

Sphere Class:

- A Sphere **is a** ThreeDimensionalShape and is represented by its radius
  - ◦ radius (represented by a *double*);

- Instance Methods: similar to Cone and Cube
  - ◦ getSurfaceArea – Formula: 4 * PI * radius * radius
  - ◦ getVolume – Formula: 4 * PI * radius * radius * radius / 3

Cylinder Class:

- A Cylinder **is a** ThreeDimensionalShape and is represented by its radius and height
  - ◦ radius (represented by a *double*);
  - ◦ height (represented by a *double*)

- Instance Methods: similar to Cone and Cube and Sphere
  - ◦ getSurfaceArea – Formula: 2 * PI * radius * height
  - ◦ getVolume – Formula: PI * radius * radius * height

Cube Class:

- A Cube **is a** ThreeDimensionalShape and is represented by its side length
  - ◦ length (represented by a *double*);

- Instance Methods: similar to Cone
  - ◦ getSurfaceArea – Formula: 6 * length * length
  - ◦ getVolume – Formula: length * length * length