Programming Assignment 4

<Code Skeleton>

The following is the code skeleton. It is translated from the UML diagram given in the assignment, plus a couple of comments. As I mentioned in class, you're encouraged to write more fields and methods, such as a name field for the Shape class and constructors for every class.

```
public class LastFirst4 {
    public static void main(String[] args) {
        // check command line
        // open file
        // read SVG header
        // while file has more tokens
        //     read command token
        //     read line, rect, or circle attributes
        //     do
        //         read tokenOne
        //         if tokenOne is not "end"
        //             read tokenTwo
        //             add style
        //     while tokenOne is not "end"
        // render SVG to "System.out"
    }
}

class Svg {
    private ArrayList<Shape> shapes = new ArrayList<Shape>();
    private double height;
    private double width;

    void addShape(Shape shape) {
        // add "shape" to the "shapes" list
    }

    void render(PrintStream out) {
        // render this "Svg" to "out" e.g.,
        // <svg width='300.0' height='300.0'>
        // <rect x='5.0' y='5.0' width='290.0' height='290.0' style='fill:#f8f8f8;'/>
        // </svg>
    }
}

abstract class Shape {
    private ArrayList<String> styles = new ArrayList<String>();

    public void addStyle(String key, String value) {
        // add to the "styles" list.
    }

    void render(PrintStream out) {
        // render this "Shape" to "out" e.g.,
        // <rect x='5.0' y='5.0' width='290.0' height='290.0' style='fill:#f8f8f8;'/>
    }

    abstract void renderAttributes(PrintStream out);
}

class Line extends Shape {
    private double x1;
    private double y1;
    private double x2;
    private double y2;

    void renderAttributes(PrintStream out) {
        // render class fields as attributes to "out" e.g.,
        // x1='5.0' y1='295.0' x2='295.0' y2='5.0'
    }
}

class Rectangle extends Shape {
    private double x;
    private double y;
    private double width;
    private double height;
    void renderAttributes(PrintStream out) {
        // render class fields as attributes to "out" e.g.,
        // x='5.0' y='5.0' width='290.0' height='290.0'
    }
}

class Circle extends Shape {
    private double cx;
    private double cy;
    private double r;

    void renderAttributes(PrintStream out) {
        // render class fields as attributes to "out" e.g.,
        // cx='150.0' cy='150.0' r='75.0'
    }
}
```

Let's use the following as an example input file.

```
svg 300 300

rect 5 5 290 290
stroke          #c0c0c0
stroke-width    2
end
```

First read the SVG header,

```
// read SVG header,              --> "svg 300 300"
```

Create a `Svg` object, something like:

```
Svg mySvg = new Svg(width, height);
```

Then read the shapes.

Try to read a shape command and its attributes:

```
// while file has more tokens      --> true
//   read command token            --> "rect"
//   read shape attributes         --> "5 5 290 290"
//   :
```

Create a `Rectangle` object, something like:

```
Rect myRect = new Rect(x, y, width, height);
```

Add myRect to `mySvg`, something like:

```
mySvg.addShape(myRect);
```

Then read the shape styles:

1. Iteration #1:

   ```
   //   do
   //     read tokenOne             --> "stroke"
   //     if tokenOne is not "end"   --> true
   //       read tokenTwo           --> "#c0c0c0"
   //       add style               --> "stroke:#c0c0c0"
   //   while tokenOne is not "end"  --> true
   ```

   Add a style to myRect object, something like:

   ```
   myRect.addStyle("stroke", "#c0c0c0");
   ```

2. Iteration #2:

   ```
   //   do
   //     read tokenOne             --> "stroke-width"
   //     if tokenOne is not "end"   --> true
   //       read tokenTwo           --> "2"
   //       add style               --> "stroke-width:2"
   //   while tokenOne is not "end"  --> true
   ```

   Add a style to myRect object, something like:

   ```
   myRect.addStyle("stroke-width", "2");
   ```

3. Iteration #3:

```
//   do
//     read tokenOne              --> "end"
//     if tokenOne is not "end"   --> false
//       read tokenTwo
//       add style
//   while tokenOne is not "end"  --> false
```

Try to read shape command and shape attributes:

```
// while file has more tokens     --> false
```

After the while loop ends.

```
// render SVG to "System.out"
```

We get:

```
  <svg width='300.0' height='300.0'>
  <rect x='5.0' y='5.0' width='290.0' height='290.0' style='stroke:#c0c0c0;stroke-
width:2;'/>
  </svg>
```