Joel Edwards
Course: Java Programming 1
Homework 4
April 10, 2011

1)

Source:

DimensionOutOfRangeException.java:

```
public class DimensionOutOfRangeException
    extends RuntimeException
{
    public static final long serialVersionUID = 1L;

    public DimensionOutOfRangeException() {
        super();
    }

    public DimensionOutOfRangeException(String message) {
        super(message);
    }
}
```

Rectangle.java:

```
import java.awt.Color;
import java.awt.Graphics;
import java.util.UUID;

public class Rectangle
{
    public static final int MIN_WIDTH = 10;
    public static final int MAX_WIDTH = 100;

    public static final int MIN_LENGTH = 10;
    public static final int MAX_LENGTH = 100;

    public static final int DEFAULT_WIDTH  = 10;
    public static final int DEFAULT_LENGTH = 10;
    public static final Color DEFAULT_COLOR = Color.BLACK;

    private static int lastID = 1;
    public final int ID = lastID++;

    private int width = DEFAULT_WIDTH;
    private int length = DEFAULT_LENGTH;
```

```
    private Color color = DEFAULT_COLOR;

    public Rectangle() {;}

    public Rectangle(int width) throws DimensionOutOfRangeException {
        setWidth(width);
        setLength(width);
    }

    public Rectangle(int width, int length) throws
DimensionOutOfRangeException {
        setWidth(width);
        setLength(length);
    }

    public Rectangle(int width, int length, Color color) throws
DimensionOutOfRangeException {
        setWidth(width);
        setLength(length);
        this.color = color;
    }

    public void setWidth(int width)
        throws DimensionOutOfRangeException
    {
        if (width < MIN_WIDTH) {
            throw new
DimensionOutOfRangeException(String.format("Width is less than the
minimum allowed (width=%d  MIN_WIDTH=%d)", width, MIN_WIDTH));
        }
        if (width > MAX_WIDTH) {
            throw new
DimensionOutOfRangeException(String.format("Width is greater than the
maximum allowed (width=%d  MAX_WIDTH=%d)", width, MAX_WIDTH));
        }
        this.width = width;
    }

    public void setLength(int length)
        throws DimensionOutOfRangeException
    {
        if (length < MIN_LENGTH) {
            throw new
DimensionOutOfRangeException(String.format("Length is less than the
minimum allowed (length=%d  MIN_LENGTH=%d)", length, MIN_LENGTH));
        }
        if (length > MAX_LENGTH) {
            throw new
DimensionOutOfRangeException(String.format("Length is greater than
```

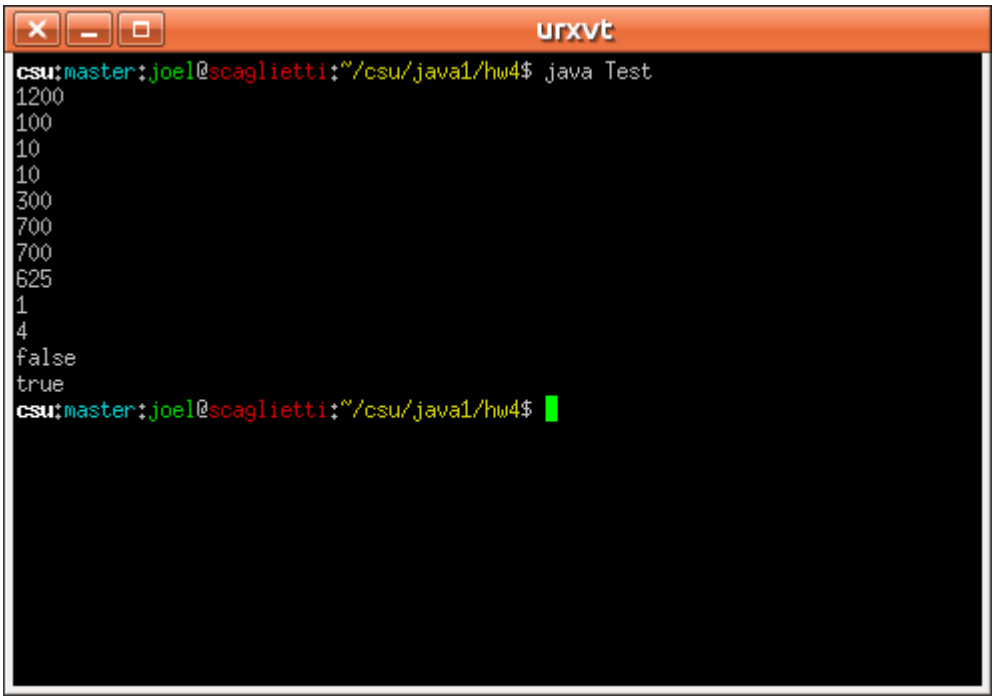
```
the maximum allowed (length=%d  MIN_LENGTH=%d)", length,
MAX_LENGTH));
        }
        this.length = length;
    }

    public void drawAt(Graphics g, int x, int y) {
        Color lastColor = g.getColor();
        g.setColor(color);
        g.fillRect(x, y, width, length);
        g.setColor(lastColor);
    }

    public boolean equals(Rectangle other) {
        if (width != other.width)
            return false;
        if (length != other.length)
            return false;
        if (!(color.equals(other.color)))
            return false;
        return true;
    }

    public int computeArea() {
        return width * length;
    }

    public int getWidth() {
        return width;
    }

    public int getLength() {
        return length;
    }

    public Color getColor() {
        return color;
    }
}
```

Test Source:

Test.java:

```
class Test{
    public static void main(String a[]){
        Rectangle r1=new Rectangle(30,40);
        Rectangle r2=new Rectangle();
        Rectangle r5=new Rectangle(25);
        Rectangle r3=new Rectangle(35,20);
        Rectangle r4=new Rectangle(35,20);
        System.out.println(r1.computeArea()); //1200.
        System.out.println(r2.computeArea()); //100.0
        System.out.println(r2.getWidth());   //10.0
        System.out.println(r2.getLength());  //10.0
        r2.setWidth(20);
        r2.setLength(15);
        System.out.println(r2.computeArea()); //300.0
        System.out.println(r3.computeArea()); //700.0
        System.out.println(r4.computeArea()); //700.0
        System.out.println(r5.computeArea()); //625.0
        System.out.println(r1.ID);      //1
        System.out.println(r3.ID);       //4
        //r2.ID = 99; will cause error
        System.out.println(r1.equals(r2));   //false
        System.out.println(r3.equals(r4));   //true
    }
}
```

Test Output:



DrawR Source:

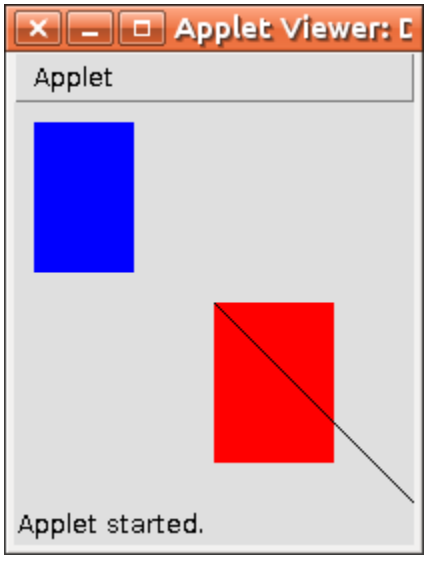
DrawR.java:

```
import java.awt.*;
import java.applet.Applet;
public class DrawR extends Applet{
    Rectangle r1=new Rectangle(50,75,Color.blue);
    Rectangle r2=new Rectangle(60,80,Color.red);
    public void paint(Graphics g) {
        r1.drawAt(g,10,10);
        r2.drawAt(g,100,100);
        g.drawLine(100,100,200,200);
    }

}
```

DrawR.html:

```
<html>
    <head>
        <title>A Simple Program</title>
        <meta http-equiv="pragma" content="no-cache" />
    </head>
    <body>
        Here is the output of my program:
        <applet code="DrawR.class" width="200" height="200">
        </applet>
    </body>
</html>
```

DrawR Output:



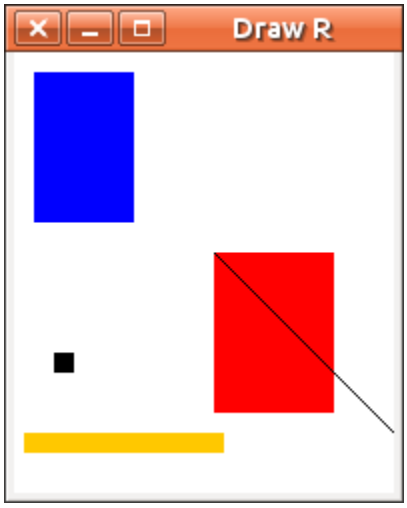
Below are a few other classes which I used to test the Rectangle class.

DrawRApp Source:
This is a simple extension of the DrawR Applet. It can be viewed as an Applet or an Application, and adds a few more shapes.

DrawRApp.java:

```
import java.awt.*;
import java.applet.Applet;
import javax.swing.JApplet;
import javax.swing.JFrame;

public class DrawRApp
    extends JApplet
{
    public static final long serialVersionUID = 1L;

    Rectangle r1 = new Rectangle(50,75,Color.blue);
    Rectangle r2 = new Rectangle(60,80,Color.red);
    Rectangle r3 = new Rectangle();
    Rectangle r4 = new Rectangle(100,10,Color.orange);

    public void start() {
        repaint();
    }

    public void paint(Graphics g) {
        r1.drawAt(g,10,10);
        r2.drawAt(g,100,100);
        r3.drawAt(g,20, 150);
        r4.drawAt(g,5,190);
        g.drawLine(100,100,200,200);
    }

    public static void main(String[] args) {
        JFrame frame = new JFrame("Draw R");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setPreferredSize(new Dimension(200, 250));
        frame.setMinimumSize(new Dimension(200,200));

        JApplet applet = new DrawRApp();
        applet.init();
        applet.start();
        frame.add("Center", applet);
        frame.pack();
        frame.setVisible(true);
    }
}
```

DrawRApp Output:



DrawBoard Source:
This class uses the rectangle class in order to draw the squares within a resizeable 8x8 game board. When a resize occurs, the board is buffered to an Image, which then overwrites the previous image. Does not use Java's built-in BufferStrategy class.

DrawBoard.java:

```
import java.applet.Applet;
import java.awt.Color;
import java.awt.Dimension;
import java.awt.event.ComponentListener;
import java.awt.event.ComponentEvent;
import java.awt.Graphics;
import java.awt.Image;
import java.awt.image.BufferedImage;
import java.awt.image.ImageObserver;
import javax.swing.JApplet;
import javax.swing.JFrame;

public class DrawBoard
    extends JApplet
{
    public static final long serialVersionUID = 1L;
    private Rectangle[][] squares = new Rectangle[8][8];

    private Color light  = Color.WHITE;
    private Color dark   = Color.BLACK;
    private Color border = Color.ORANGE;

    public void init() {
        // Here we handle the re-sizing of the Applet, and ignore all
other
        // events returned by the ComponentListener
        this.addComponentListener(new ComponentListener() {
            public void componentHidden(ComponentEvent e) { ; }
            public void componentMoved(ComponentEvent e) { ; }
            public void componentResized(ComponentEvent e) {
                ((DrawBoard)e.getComponent()).repaint();
            }
            public void componentShown(ComponentEvent e) { ; }
        });

        Color color;
        int count = 0;
        for (int i = 0; i < 8; i++) {
            for (int j = 0; j < 8; j++) {
                count++;
                if ((count % 2) == 0) {
                    color = dark;
                } else {
```

```
                    color = light;
                }
                squares[i][j] = new Rectangle(10,10,color);
            }
            count++;
        }

        setBackground(border);
    }

    public void start() {
        repaint();
    }

    public void paint(Graphics g) {
        int width   = getWidth();
        int height  = getHeight();
        int smaller = (width < height) ? width : height;
        int size    = smaller / 10; // size of squares

        BufferedImage frameBuffer = new BufferedImage(width, height,
BufferedImage.TYPE_INT_RGB);
        Graphics gf = frameBuffer.getGraphics();

        // Draw Board Items
        Rectangle square;
        gf.setColor(border);
        gf.fillRect(0, 0, width, height);
        for (int x = 0; x < 8; x++) {
            for (int y = 0; y < 8; y++) {
                square = squares[x][y];
                square.setWidth(size);
                square.setLength(size);
                square.drawAt(gf, size+(x*size), size+(y*size));
            }
        }

        g.drawImage(frameBuffer, 0, 0, border, new ImageObserver(){
            public boolean imageUpdate(Image img, int infoflags, int
x, int y, int width, int height) {return true;}
        });
    }

    public static void main(String[] args) {
        JFrame frame = new JFrame("Draw R");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setPreferredSize(new Dimension(400, 400));
        frame.setMinimumSize(new Dimension(200,200));
```

```
        JApplet applet = new DrawBoard();
        applet.init();
        applet.start();
        frame.add("Center", applet);
        frame.pack();
        frame.setVisible(true);
    }
}
```

DrawBoard Output: