```
import java.awt.Color;
import java.awt.Dimension;
import java.awt.Graphics;
import javax.swing.JPanel;
import javax.swing.JFrame;
import java.util.Random;


public class LineArt extends JPanel {

	// Unique version ID for this class to ensure saved objects can be
loaded safely
	private static final long serialVersionUID = 1L;

	// Initial width of height of the starting rectangle
	private static int width = 980;
	private static int height = 630;

	// main method to launch the program as a standalone application -
no need to
	// modify
	public static void main(String[] args) {
		LineArt panel = new LineArt();
		panel.setPreferredSize(new Dimension(width + 20, height +
20)); // content size window dimensions

		JFrame frame = new JFrame("Line Art"); // Title of frame
		frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		frame.add(panel);
		frame.pack();
		frame.setVisible(true);
	}

	/**
	 * Draw the four corners of line art.  Line art displays straight
lines inside a rectangle from one side to
	 * a perpendicular side. The lines must be drawn in such a way that
both the starting points of the lines on
	 * one side and the ending points on the other side are equi-
distant along the sides.  The size of the rectangle
	 * is 980 pixels wide by 630 pixels high.
	 *
	 * @param g the Graphics object used for drawing shapes, text, and
images
	 */
	public void drawLineArt(Graphics g) {
		//generate random color

		// Draw the initial rectangle
		g.drawRect(10, 10, width, height);
int ymax=640;
int xmax=990;
int ymin=10;
int xmin=10;
```

```
int i=1;
int i2= 1;
Random reddy= new Random();
int red;
Random bluey= new Random();
int blue;
Random greeny= new Random();
int green;

            while(i<=3) {
            // Draw bottom-left corner
            int change=10;
            int x=xmin;
            int y1=ymin;
            int z=0;
            int l=0;
            while(x<xmax & y1<ymax) {
                  red= reddy.nextInt(0,256);
                  blue= bluey.nextInt(0,256);
                  green= greeny.nextInt(0,256);
                  Color color1= new Color(red,green,blue);
            g.setColor(color1);
            g.drawLine(xmin, y1, x, ymax);
            x=x+change;
            y1=y1+change;
            z++;
      }

            // Draw bottom-right corner
            y1=ymin;
            x=xmax;
            while(x>xmin & y1<ymax) {

            g.drawLine(xmax, y1, x, ymax);
            x=x-change;
            y1=y1+change;
            l++;
      }


            // Draw top-right corner

            y1=ymax;
            x=xmax;
            while(x>xmin & y1>ymin) {
            red= reddy.nextInt(0,256);
            blue= bluey.nextInt(0,256);
            green= greeny.nextInt(0,256);
                  Color color2= new Color(red,green,blue);
                  g.setColor(color2);
            g.drawLine(xmax, y1, x, ymin);
            x=x-change;
            y1=y1-change;
```

```
	}

		// Draw top-left corner
		y1=ymax;
		x=xmin;
		while(x<xmax & y1>ymin) {
		g.drawLine(xmin, y1, x, ymin);
		x=x+change;
		y1=y1-change;
		}
		ymax=ymax-100;
		xmax=xmax-230/(i2);
		ymin=ymin+100;
		xmin=xmin+230/(i2);
		i++;
		i2=i2*2;
		}
	}

	/**
	 * Overrides JPanel's paintComponent method to perform custom
drawing.
	 *
	 * @param g the Graphics object used for drawing shapes, text, and
images
	 */
	@Override
	protected void paintComponent(Graphics g) {
		super.paintComponent(g); // Clears the panel before drawing
		drawLineArt(g);
	}

}
```