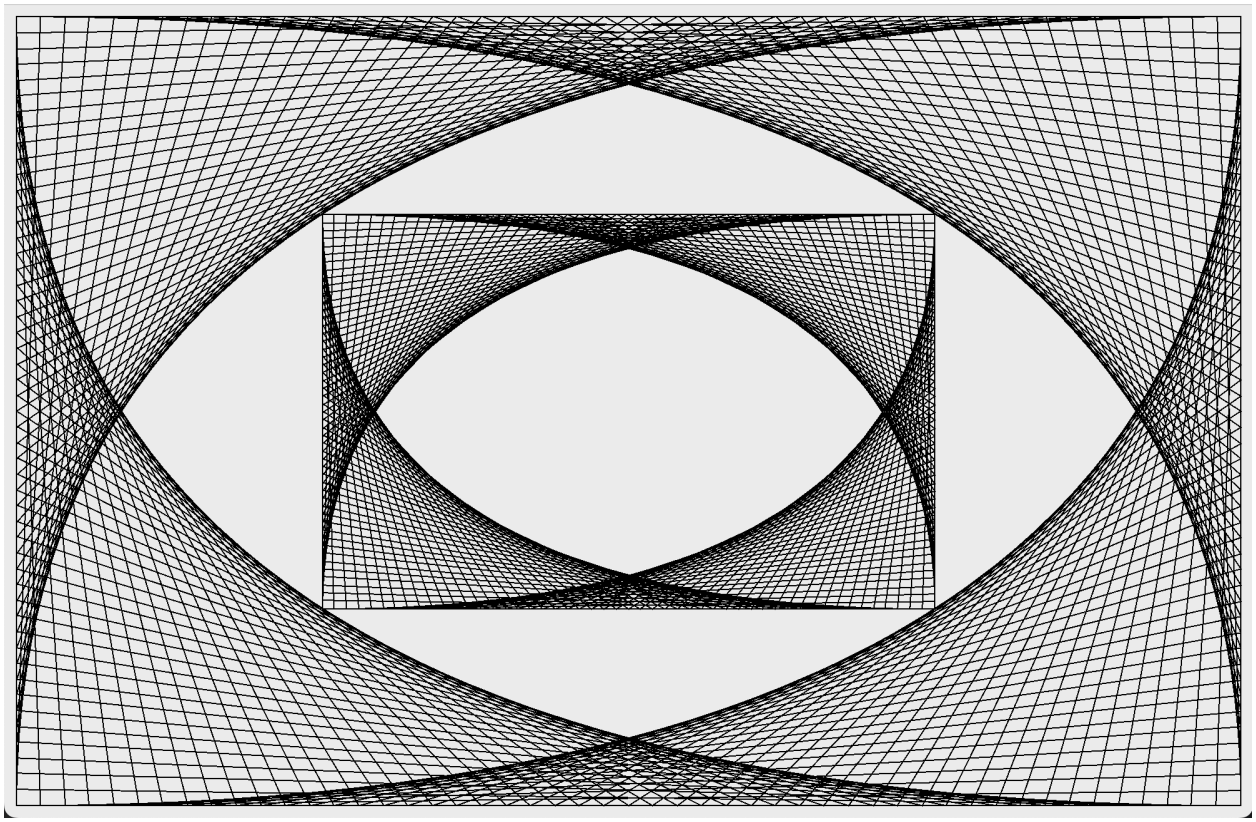

```
import java.awt.Dimension;
import java.awt.Graphics;
import javax.swing.JPanel;
import javax.swing.JFrame;

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) {

                // Draw the initial rectangle
                g.drawRect(10, 10, width, height);

                 int n = 50;
                // Draw bottom-left corner
                 for (int i = 0; i <= n; i++)
                        g.drawLine(10 + (i * width) / n, 10 + height, 10 + width, 10 + height - (i *
height) / n);

                // Draw bottom-right corner
                 for (int i = 0; i <= n; i++)
                        g.drawLine(10, 10 + (i * height) / n, 10 + (i * width) / n, 10 + height);

                // Draw top-right corner
                 for (int i = 0; i <= n; i++)
                        g.drawLine(10 + width - (i * width) / n, 10, 10 + width, 10 + height - (i *
height) / n);

                // Draw top-left corner
                 for (int i = 0; i <= n; i++)
                        g.drawLine(10 + (i * width) / n, 10, 10, 10 + height - (i * height) / n);



                 // Little Rectangle Section
```

```
                int newWidth = (int) (width * 0.5);
                int newHeight = (int) (height * 0.5);
                int newX = 10 + 245;
                int newY = 10 + 158;

                // Small rectangle
                g.drawRect(newX, newY, newWidth, newHeight);

                // Bottom-left corner
                for (int i = 0; i <= n; i++)
                    g.drawLine(newX + (i * newWidth) / n, newY + newHeight, newX + newWidth,
newY + newHeight - (i * newHeight) / n);

                // Bottom-right corner
                for (int i = 0; i <= n; i++)
                    g.drawLine(newX, newY + (i * newHeight) / n, newX + (i * newWidth) / n, newY +
newHeight);

                // Top-right corner
                for (int i = 0; i <= n; i++)
                    g.drawLine(newX + newWidth - (i * newWidth) / n, newY, newX + newWidth,
newY + newHeight - (i * newHeight) / n);

                // Top-left corner
                for (int i = 0; i <= n; i++)
                    g.drawLine(newX + (i * newWidth) / n, newY, newX, newY + newHeight - (i *
newHeight) / n);
                    }


        /**
         * 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);
        }

}
```