```java
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

      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
    * a perpendicular side. The lines must be drawn in such a way that both the starting
    * one side and the ending points on the other side are equi-distant along the sides.  The
    * 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);
      //g.drawLine(989, 639, 10, 639-7);
      int x = 1003;
      int y = 648;
      for (int i=0; i<70; i++) {
         x = x-14;
         y = y-9;
         g.drawLine(x, 639, 10, y);
      }
      x=-4;
      y=648;
      for (int i=0; i<70; i++) {
         x = x+14;
         y = y-9;
         g.drawLine(x, 639, 989, y);
      }
      x=-4;
      y=648;
      for (int i=0; i<70; i++) {
         x = x+14;
```

```
59          y = y-9;
60          g.drawLine(10, y, x, 10);
61       }
62       x=1003;
63       y=648;
64       for (int i=0; i<70; i++) {
65          x = x-14;
66          y = y-9;
67          g.drawLine(989, y, x, 10);
68       }
69
70       int smallWidth = width / 2;
71       int smallHeight = height / 2;
72       int offsetX = 10 + width / 4;
73       int offsetY = 10 + height / 4;
74       g.drawRect(offsetX, offsetY, smallWidth, smallHeight);
75
76       x = 759;
77       y = 492;
78       for (int i=0; i<35; i++) {
79          x = x-14;
80          y = y-9;
81          g.drawLine(x, 483, 255, y);
82       }
83       x=241;
84       y=492;
85       for (int i=0; i<35; i++) {
86          x = x+14;
87          y = y-9;
88          g.drawLine(255, y, x, 166);
89       }
90       x=241;
91       y=175;
92       for (int i=0; i<35; i++) {
93          x = x+14;
94          y = y+9;
95          g.drawLine(x,166,745,y);
96       }
97       x=241;
98       y=492;
99       for (int i=0; i<35; i++) {
100         x = x+14;
101         y = y-9;
102         g.drawLine(x,483,745,y);
103      }
104
105   }
106
107   /**
108    * Overrides JPanel's paintComponent method to perform custom drawing.
109    *
110    * @param g the Graphics object used for drawing shapes, text, and images
111    */
112   @Override
113   protected void paintComponent(Graphics g) {
114      super.paintComponent(g); // Clears the panel before drawing
115      drawLineArt(g);
116   }
```

```
117
118 }
119
```