```java
import java.awt.Color;
import java.awt.Graphics;
import java.awt.Point;
public class Line
{
    private Point from;
    private Point to;
    private Color color;

    public Line(Point from, Point to, Color color)
    {
        this.from = from;
        this.to = to;
        this.color = color;
    }

    public Line(int fromX, int fromY, int toX, int toY, Color color)
    {
        this.from  = new Point(fromX, fromY);
        this.to    = new Point(toX, toY);
        this.color = color;
    }

    public Point getFrom()
    {
        return from;
    }

    public Point getTo()
    {
        return to;
    }

    public Color getColor()
    {
        return color;
    }

    public void draw (Graphics g)
    {
        g.setColor(color);
        g.drawLine(from.x, from.y, to.x, to.y);
    }

    public void setColor(Color color)
    {
        this.color = color;
    }

    public double getLenght()
    {
          return Math.sqrt(Math.pow(from.x-to.x, 2)+Math.pow(from.y-to.y, 2));
    }

    public String toString()
    {
        return "("+from.x+","+from.y+") -> ("+to.x+","+to.y+")";
    }
}
```

```java
import java.awt.Graphics;
import java.util.ArrayList;
public class Lines
{
    private ArrayList<Line> alLines = new ArrayList<>();

    public void add(Line e)
    {
        alLines.add(e);
    }

    public Line get(int index)
    {
        return alLines.get(index);
    }

    public void sort()
    {
         for (int i=0; i<alLines.size()-1; i++)
         {
              int minPos = i;
              double min = alLines.get(i).getLenght();
              for (int j=i+1; j<alLines.size();j++)
              {
                   if(alLines.get(j).getLenght() < min)
                   {
                        min = alLines.get(j).getLenght();
                        minPos = j;
                   }
              }

              if (i!= minPos)
              {
                   Line temp = alLines.get(i);
                   alLines.set(i, alLines.get(minPos));
                   alLines.set(minPos, temp);
              }
         }
    }

    public void draw (Graphics g)
    {
        for (int i=0; i<alLines.size(); i++)
        {
            alLines.get(i).draw(g);
        }
    }

    public Object[] toArray()
    {
        return alLines.toArray();
    }
}
```

```java
import java.awt.Color;
import java.awt.Graphics;
public class DrawPanel extends javax.swing.JPanel
{
    // NEVER use a *new* here !!!!!!!!
    private Lines lines = null;

    public void setLines(Lines lines)
    {
        this.lines = lines;
    }

    public void paintComponent(Graphics g)
    {
        int w = getWidth();
        int h = getHeight();

        g.setColor(Color.WHITE);
        g.fillRect(0, 0, w, h);

        if (lines != null)
            lines.draw(g);
    }

    public DrawPanel()
    {
        initComponents();
    }
// Skipped: ... initComponents { ... }
// Skipped: ... graphic attributes
}
```

```
import java.awt.Color;
import javax.swing.JColorChooser;
public class MainFrame extends javax.swing.JFrame
{
     // Model

     private Lines lines = new Lines();

     public MainFrame()
     {
          initComponents();
          drawPanel.setLines(lines);

          // TODO: remove test lines - begin
          lines.add(new Line(10, 20, 15, 25, Color.YELLOW));
          lines.add(new Line(20, 20, 45, 25, Color.ORANGE));
          lines.add(new Line(100, 120, 215, 25, Color.RED));
          lines.add(new Line(120, 60, 75, 105, Color.BLUE));
          lines.add(new Line(150, 55, 35, 5, Color.GREEN));
          // TODO: remove test lines - end

          updateView();
     }

     public void updateView()
     {
          linesList.setListData(lines.toArray());
          repaint();
     }
// Skipped: ... initComponents { ... }
    private void addButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_addButtonActionPerformed
         int fromX = Integer.valueOf(fromXTextField.getText());
         int fromY = Integer.valueOf(fromYTextField.getText());
         int toX = Integer.valueOf(toXTextField.getText());
         int toY = Integer.valueOf(toYTextField.getText());

         Color color = new Color((int) (Math.random() * 256), (int) (Math.random() * 256), (int) (Math.random() * 256), (int) (Math.rand
* 256));
         Line line = new Line(fromX, fromY, toX, toY, color);

         lines.add(line);
         updateView();
    }//GEN-LAST:event_addButtonActionPerformed

    private void clearButtonActionPerformed(java.awt.event.ActionEvent evt)//GEN-FIRST:event_clearButtonActionPerformed
    {//GEN-HEADEREND:event_clearButtonActionPerformed
         lines = new Lines();
         drawPanel.setLines(lines);
         updateView();
    }//GEN-LAST:event_clearButtonActionPerformed

    private void colorButtonActionPerformed(java.awt.event.ActionEvent evt)//GEN-FIRST:event_colorButtonActionPerformed
    {//GEN-HEADEREND:event_colorButtonActionPerformed
         int pos = linesList.getSelectedIndex();
         if (pos >= 0)
         {
              Line line = lines.get(pos);

              Color color = JColorChooser.showDialog(this, "Choisir une couleur", line.getColor());
              line.setColor(color);
         }

         repaint();
    }//GEN-LAST:event_colorButtonActionPerformed

     private void sortButtonActionPerformed(java.awt.event.ActionEvent evt)//GEN-FIRST:event_sortButtonActionPerformed
     {//GEN-HEADEREND:event_sortButtonActionPerformed
          lines.sort();
          updateView();
     }//GEN-LAST:event_sortButtonActionPerformed
// Skipped: ... Look & Feel
// Skipped: ... graphic attributes
}
```