App.java

```
import javax.swing.*;
import javax.swing.border.EmptyBorder;
import java.awt.*;
import java.awt.Dimension;
import java.awt.event.*;
public class App {
    final static int FRAME_WIDTH = 1500;
    final static int FRAME_HEIGHT = 800;
    static GraphPanel panel;
    static JTextField meanField, sdField, rawScoreField;
    static JButton calButton;
    static JTextField resultField;
    public static void main(String[] args) throws Exception {
        JFrame frame = new JFrame();
        frame.setSize(FRAME_WIDTH, FRAME_HEIGHT);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setLocationRelativeTo(null);
        panel = new GraphPanel();
        JLabel label = new JLabel("Activity 2",SwingConstants.CENTER);
        label.setForeground(Color.blue);
        label.setFont(new Font(Font.DIALOG, Font.BOLD, 60));
        JPanel p = new JPanel();
        p.add(label);
        p.setBackground(Color.orange);
        JPanel sidePanel = setSidePanel();
        calButton.addActionListener(new ActionListener(){
            public void actionPerformed(ActionEvent e){
                try{

                    panel.computeZScore(Double.parseDouble(meanField.getText()),
                    Double.parseDouble(sdField.getText()),Double.parseDouble(raw
                    ScoreField.getText()));
                }catch(Exception exception){
                    JOptionPane.showMessageDialog(frame, "Wrong Input", "error",
                   JOptionPane.ERROR_MESSAGE);
                }
            }
        });
        frame.add(sidePanel, BorderLayout.EAST);
        frame.add(p, BorderLayout.NORTH);
```

```
        frame.add(panel, BorderLayout.CENTER);
        frame.setVisible(true);
    }
    static JPanel setSidePanel(){
        JPanel panel = new JPanel();
        panel.setBackground(Color.ORANGE);
        panel.setLayout(new BoxLayout(panel, BoxLayout.Y_AXIS));
        panel.setBorder(new EmptyBorder(new Insets(50, 50, 150, 50)));
        meanField = new JTextField(10);
        sdField = new JTextField(10);
        rawScoreField = new JTextField(10);

        JLabel label = new JLabel("Enter mean:");
        panel.add(label);
        panel.add(meanField);
        panel.add(Box.createRigidArea(new Dimension(30,100)));

        label = new JLabel("Enter standard deviation:");
        panel.add(label);
        panel.add(sdField);
        panel.add(Box.createRigidArea(new Dimension(30,100)));

        label = new JLabel("Enter raw score:");
        panel.add(label);
        panel.add(rawScoreField);
        panel.add(Box.createRigidArea(new Dimension(30,100)));

        calButton = new JButton("Calculate");
        panel.add(calButton);

        return panel;
    }
}
```

GraphPanel.java

```
import javax.swing.*;
import javax.swing.plaf.ColorUIResource;
import org.apache.commons.math3.distribution.*;
import org.apache.commons.math3.stat.descriptive.rank.Percentile;
import java.awt.Color;
import java.awt.*;
import java.awt.image.*;

public class GraphPanel extends Canvas {

    final static int MARGIN = 25;
    int windowWidth, windowHeight;
    int xStart;
    int vLines = 8;
    int hLines = 4;
    Graphics graphics;
    Double zScore;
    void GraphPanel(){
    }
    public void paint(Graphics g){
        graphics = g;
        xStart = MARGIN;
        windowWidth = getSize().width  - (MARGIN*2);
        windowHeight = getSize().height - MARGIN;
        BufferedImage pixel = new BufferedImage(3,3, BufferedImage.TYPE_INT_RGB);
        pixel.setRGB(0,0, Color.BLACK.getRGB());
       setBackground(Color.white);
        drawLines(g, pixel);
        if(zScore != null){
            paintZScore(zScore, g);
        }
    }

    void drawLines(Graphics g, BufferedImage pixel){ // used to draw the lines in
the graph
        //horizontal border
        g.drawLine(xStart,0,windowWidth+MARGIN,0);
        g.drawLine(xStart,windowHeight, windowWidth+MARGIN, windowHeight);
        //vertical border
        g.drawLine(xStart, 0, xStart, windowHeight);
        //g.drawLine(windowWidth+MARGIN, 0, windowWidth+MARGIN, windowHeight);
        g.setColor(Color.blue);
        g.setFont(new Font("Dialog",Font.BOLD,15));
```

```
        for(int y = 0; y < hLines; y++){ //horizontal lines
             g.drawLine(xStart, y * (windowHeight/hLines), windowWidth + MARGIN,
             y * (windowHeight/hLines));
            g.drawString(String.format("0.%d",4-y), 10, y *
            (windowHeight/hLines) + 10);
        }
             g.drawString("0", 10, windowHeight + 20);

        for(int x = 0; x < vLines+1; x++){//vertical lines
             g.drawLine(( x*(windowWidth)/vLines )+ MARGIN, 0,(
             x*(windowWidth)/vLines )+ MARGIN, windowHeight);
            g.drawString(String.format("%d",x-4), ( x*(windowWidth)/vLines )+
            MARGIN, windowHeight + 20);
        }
        drawBellCurve(g, pixel);
    }


    void drawBellCurve(Graphics g, BufferedImage pixel){  // used to draw the
bell curve

        for(double x = 0; x < windowWidth; x++){
            double y = computeBellCurve(x/((windowWidth)/vLines)-4);

             g.drawImage(pixel,(int) x + MARGIN ,windowHeight-
             (int)Math.ceil(y*10*(windowHeight/4)), null);
        }
    }

    double computeBellCurve(double x){ // used to calculate/compute the value of
y with respect to x in the bell curve
        double y = Math.exp(-(Math.pow(x, 2)/2)) / Math.sqrt(2*Math.PI);
        return y;
    }

    public void computeZScore(double mean, double sd, double rawScore){

        double z = (rawScore-mean)/sd;

         zScore = Double.parseDouble(String.format("%.2f",z)); // format the
      zscore to have only two decimal places so it will match the z table.
        repaint();

    }
```

```
    void paintZScore(double zScore,Graphics g){     // for shading the area of
the distribution
        double xValue = ((windowWidth/hLines) * (zScore/2 + 2));//trial and error
        NormalDistribution nd = new NormalDistribution();
        double percentile;

        if(zScore < 0){
            percentile = (nd.cumulativeProbability(zScore) * 100)  + 50;

        }else{
            percentile = (nd.cumulativeProbability(zScore) * 100) -50;
        }

        Color color = new Color(252,186,3,200);
        g.setColor(color);


        if(zScore > 0){

            for(double x = (windowWidth/2); x < xValue; x++){
                   double height = computeBellCurve(x/(windowWidth/vLines)-
                   4)*(10*(windowHeight/hLines)) ;
                g.drawLine((int)x + MARGIN ,windowHeight-(int)Math.ceil(height),
                (int)x +MARGIN ,   windowHeight)
            }
        }else{
            for(double x = (windowWidth/2); x > xValue; x--){
                double height = computeBellCurve(x/(windowWidth/vLines)-
                4)*(10*(windowHeight/hLines)) ;
                g.drawLine((int)x + MARGIN ,windowHeight-(int)Math.ceil(height),
                (int)x +MARGIN ,   windowHeight);
            }
        }

        if(zScore < 0){ // if z-score is negative
            percentile = 100 - percentile;
        }
        g.setColor(Color.black);
        g.setFont(new Font("Dialog", Font.BOLD,20));

        if(Math.abs(zScore) > 3.9){
             g.drawString(String.format("Percentage: Approx 50 ") + "%", 6 *
             (windowWidth/vLines) + MARGIN + 10, 2 * (windowHeight/hLines));
        }else{
```

```
            g.drawString(String.format("Percentage: %.2f",percentile) + "%", 6 *
            (windowWidth/vLines) + MARGIN + 10, 2 * (windowHeight/hLines));
        }
            g.drawString(String.format("ZScore: %.2f" ,zScore), 6 *
            (windowWidth/vLines)+ MARGIN + 10, 2 * (windowHeight/hLines) - 25);

    }
}
```