```java
// ChartBars.java: This program draws bar charts based on data sets.

import java.awt.*;
import java.awt.event.*;
import java.io.*;
import java.math.RoundingMode;
import java.util.*;
import java.text.DecimalFormat;
import javax.management.RuntimeErrorException;
import javax.swing.*;

public class ChartBars {

    private static JFrame mainFrame;
    private JPanel controlPanel;
    private JPanel canvasPanel;
    private CvBars cvb;

    public ChartBars() {
        // Get and Store Data
        java.util.List<java.util.Map.Entry<String, Integer>> dataEntries = new
        ArrayList<java.util.Map.Entry<String, Integer>>();
        String dataTitle = getDataSet(dataEntries);

        // Display GUI
        prepareGUI(dataEntries, dataTitle);
    }

    public static void main(String[] args) {

        new ChartBars();
    }

    private String getDataSet(java.util.List<java.util.Map.Entry<String, Integer>> dataEntries) {
        String csvFile = "";
        BufferedReader br = null;
        Scanner sc = null;
        String line = "";
        String csvSplitBy = ",";

        try {
            // Read In File Name From Console
            sc = new Scanner(System.in);
            System.out.print("CSV File Name: ");
            while (csvFile.equals(""))
                csvFile = sc.nextLine();
            sc.close();
            // Open and Read File Data
            br = new BufferedReader(new FileReader(csvFile));
            while ((line = br.readLine()) != null) {
                String[] content = line.split(csvSplitBy);
                if (content.length != 2)
                    throw new Error("Too Much Content.");
                if ((Character.isLetter(content[0].charAt(0))) && (!
                Character.isLetter(content[1].charAt(0))))
                    dataEntries.add(new java.util.AbstractMap.SimpleEntry<>(content[0],
                    Integer.valueOf(content[1])));
                else if ((!Character.isLetter(content[0].charAt(0))) &&
                (Character.isLetter(content[1].charAt(0))))
                    dataEntries.add(new java.util.AbstractMap.SimpleEntry<>(content[1],
                    Integer.valueOf(content[0])));
                else
                    throw new Error("Bad Content.");
            }
            String[] Title = csvFile.split(".csv");
            return Title[0];
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        } catch (IOException e) {
```

```
            e.printStackTrace();
        } finally {
            if (br != null) {
                try {
                    br.close();
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        }
        return "";
    }

    private void prepareGUI(java.util.List<java.util.Map.Entry<String, Integer>> dataEntries,
    String dataTitle) {
        mainFrame = new JFrame("Bar Chart: " + dataTitle);
        mainFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        mainFrame.setSize(new Dimension(1000, 600));
        mainFrame.setResizable(false);

        controlPanel = new JPanel();
        controlPanel.setSize(new Dimension(1100, 600));
        controlPanel.setLayout(new BoxLayout(controlPanel, BoxLayout.Y_AXIS));

        canvasPanel = new JPanel();
        canvasPanel.setSize(new Dimension(1100, 600));
        canvasPanel.setLayout(new FlowLayout());
        cvb = new CvBars(dataEntries);
        cvb.setSize(new Dimension(511, 511));
        canvasPanel.add(cvb);

        controlPanel.add(canvasPanel);
        mainFrame.add(controlPanel);
        mainFrame.setVisible(true);
    }
}

class CvBars extends Canvas {
    private static final long serialVersionUID = 1L;

    java.util.List<java.util.Map.Entry<String, Integer>> dataEntries = new
    ArrayList<java.util.Map.Entry<String, Integer>>();
    int maxX, maxY, gMaxX, gMaxY, barWidth, barGap, spacing;
    Color[] colors;

    public CvBars(java.util.List<java.util.Map.Entry<String, Integer>> dataEntries) {
        setBackground(Color.lightGray);

        this.dataEntries = dataEntries;

        try {
            if (dataEntries.size() > 15)
                throw new Error("Too Many Entries.");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    private void initgr() {
        Dimension d = getSize();
        maxX = d.width - 61;
        maxY = d.height - 61;
        barGap = 5;
        spacing = 50;
        gMaxX = maxX - 50;
        gMaxY = maxY - spacing;
        barWidth = (int) (gMaxX / dataEntries.size()) - barGap;
        colors = new Color[dataEntries.size()];
        setColors();
    }
```

```
    private void setColors() {
        int jump = 255 / dataEntries.size();
        for (int i = jump; i < 255; i += jump)
            colors[(i / jump) - 1] = new Color(i, 50, 100);
    }

    private void drawString(Graphics g, String text, int x, int y) {
        for (String line : text.split(" "))
            g.drawString(line, x, y += g.getFontMetrics().getHeight());
    }

    public void paint(Graphics g) {
        initgr();

        // Draw the x-Axis and y-Axis
        g.drawLine(spacing, spacing, spacing, maxY);
        g.drawLine(spacing, maxY, maxX, maxY);

        // Draw the Bars
        int max = 0;
        for (java.util.Map.Entry dataEntry : dataEntries)
            if (Integer.valueOf(dataEntry.getValue().toString()) > max)
                max = Integer.valueOf(dataEntry.getValue().toString());

        int start = barGap;
        int count = 0;
        for (java.util.Map.Entry dataEntry : dataEntries) {
            double percentage = (double) Integer.valueOf(dataEntry.getValue().toString()) / max;
            Color c = colors[count++];
            g.setColor(c);
            g.fillRect(spacing + start, gMaxY - (int) (gMaxY * percentage) + spacing, barWidth,
                    (int) (gMaxY * percentage));
            g.setColor(Color.black);
            g.setFont(g.getFont().deriveFont(7.0f));
            g.drawString(dataEntry.getValue().toString(), spacing + start,
                    gMaxY - (int) (gMaxY * percentage) + spacing - barGap);
            drawString(g, dataEntry.getKey().toString(), spacing + start, maxY);
            start += barWidth + barGap;
        }

        // Draw the y-Axis Ticks
        double step = max;
        double inc = (double) max / 10;
        DecimalFormat df = new DecimalFormat("#.#");
        df.setRoundingMode(RoundingMode.CEILING);
        for (int i = 50; i <= gMaxY + 50; i += (int) (gMaxY / 10)) {
            g.drawLine(45, i, 50, i);
            g.drawString(String.valueOf(df.format(step)), barGap, i);
            step -= inc;
        }
    }
}
```