```java
//******************************************************************************
//
// File:    PlotHandler.java
// Package: ---
// Unit:    Class PlotHandler
//
//******************************************************************************

import java.io.File;
import java.io.IOException;
import java.text.DecimalFormat;
import edu.rit.numeric.ListXYSeries;
import edu.rit.numeric.plot.Plot;
import edu.rit.numeric.plot.Strokes;
import edu.rit.util.AList;

/**
 * Class PlotHandler is the delegate for dealing with visualizing the data
 * generated by the "number crunching" program, SimulationStation.
 * Its purpose is to be instantiated in SimulationStation with the data to plot,
 * where the write() method should then be called.
 *
 * Running this program and specifying in the command line arguments the plot
 * files previously generated will open a graphical representation of these
 * plots for each file.
 *
 * @author Jimi Ford
 * @version 4-4-2015
 *
 */
public class PlotHandler {

    // private data members
    private final String averagePowerFile;
    private final String probabilityFile;
    private final AList<SimulationResult> results;

    /**
     * Construct a new plot handler that plots average distances for a fixed
     * vertex count v, while varying the edge probability p
     *
     * @param   plotFilePrefix prefix to be used in the name of
     *          the plot file
     * @param   results collection of results of the finished set of
     *          simulations.
     */
    public PlotHandler(String plotFilePrefix,
            AList<SimulationResult> results) {
        averagePowerFile = plotFilePrefix + "-average-power.dwg";
        probabilityFile = plotFilePrefix + "-probability-connected.dwg";
        this.results = results;
    }

    /**
     * Save the plot information into a file to visualize by running
     * the main method of this class
     *
     * @throws IOException if it can't write to the file specified
     */
    public void write() throws IOException {
        ListXYSeries averagePowerSeries = new ListXYSeries();
        ListXYSeries probabilitySeries = new ListXYSeries();
        SimulationResult result = null;
        for(int i = 0; i < this.results.size(); i++) {
            result = results.get(i);
            if(!Double.isNaN(result.averagePower))
                averagePowerSeries.add(result.v, result.averagePower);
            if(!Double.isNaN(result.percentConnected))
                probabilitySeries.add(result.v, result.percentConnected);
        }

        Plot powerPlot = new Plot()
            .plotTitle ("Average Power vs. Number of Nodes")
            .xAxisTitle ("Number of Nodes <I>V</I>")
            .xAxisTickFormat(new DecimalFormat("0"))
            .yAxisTitle ("Average Power Needed")
            .leftMargin(84)
            .yAxisTitleOffset(60)
            .yAxisTickFormat (new DecimalFormat ("0.0E0"))
            .seriesDots(null)
            .seriesStroke (Strokes.solid(2))
            .xySeries (averagePowerSeries);
        Plot.write(powerPlot, new File(averagePowerFile));
        Plot probabilityPlot = new Plot()
        .plotTitle ("Percent Connected vs. Number of Nodes")
        .xAxisTitle ("Number of Nodes <I>V</I>")
        .xAxisTickFormat(new DecimalFormat("0"))
        .yAxisTitle ("Percent Connected")
        .yAxisTickFormat (new DecimalFormat ("0.0"))
        .seriesDots(null)
        .seriesStroke (Strokes.solid(2))
        .xySeries (probabilitySeries);
        Plot.write(probabilityPlot, new File(probabilityFile));
    }

    /**
     * Open a GUI for each plot in order to visualize the results of a
     * previously run set of simulations.
     *
     * @param args each plot file generated that you wish to visualize
     */
    public static void main(String args[]) {
        if(args.length < 1) {
            System.err.println("Must specify at least 1 plot file.");
            usage();
        }

        for(int i = 0; i < args.length; i++) {
            try {
                Plot plot = Plot.read(args[i]);
                plot.getFrame().setVisible(true);
            } catch (ClassNotFoundException e) {
                System.err.println("Could not deserialize " + args[i]);
            } catch (IOException e) {
                System.err.println("Could not open " + args[i]);
            } catch (IllegalArgumentException e) {
                System.err.println("Error in file " + args[i]);
            }
        }
    }

    /**
     * Print the usage message for this program and gracefully exit.
     */
    private static void usage() {
        System.err.println("usage: java PlotHandler <plot-file-1> "+
                "(<plot-file-2> <plot-file-3>... etc.)");
        System.exit(1);
    }
}
```