**Assignment 2: Bar and Line Charts**
Due: 02-07-2017, 11:59pm (midnight)

In this assignment, you will be using Processing to draw a bar chart and a line chart. This is your first assignment with Processing and you will be learning the basics of Processing such as, handling mouse events, basic intersection detection, and keeping track of the state of the visualization. You will be required to display a visualization based on input data, and there are a few concepts that you need to explore: (1) reading and parsing data; (2) mouse hovering for highlighting visual elements; and (3) animated transition.

**Basic Requirements:**

1. **Dataset**
   You will be given a simple comma delimited file (CSV) called "data.csv". This file will have the following properties:
   a. It has two columns, the first column contains categorical (ordinal) data, and the second column contains quantitative data.
   b. The first row has labels for each column.
   c. There are around 10 rows of data.

2. **Bar and Line Charts**
   In Processing, do the following:
   a. Parse the CSV file and read in the data
   b. The canvas should display the data in the CSV file as a bar chart as default. The bar chart should display each data item as a bar.
   c. The axes need to be labeled using the labels from the first row of the CSV file. The X-axis should display the categorical data and the Y-axis should display the quantitative data from the CSV file.
   d. Implement mouse hovering. This means that when the user moves the mouse over a bar, the bar should be highlighted. Furthermore, additional text information should be displayed (like a tooltip). The additional information should be the value of the data item from the CSV file and should be displayed as "(Apple, 12)", showing both x and y values of the data item. For testing intersection, test the geometry of the mouse position with the rectangles.
   e. Create a line chart with the same data and functionality as the bar chart:
      i. The line chart should display each data item as a circle.
      ii. Lines should connect the circles.
      iii. The axes should be labeled using the labels from the first row of the CSV file.
      iv. Hovering over a circle should display the tooltip.
   f. Combine the two graphs by creating a button at the upper-right-corner of the canvas. The button should transition between two states:
      i. State 1: The canvas should display the data in the CSV file as a bar chart
      ii. State 2: The canvas should display the data in the CSV file as a line chart

3. **Animated Transition**
   When the user clicks the button, the visualization will "transition" to the other one. The gradual transition between visualizations helps a user to tell that these two visualizations

depict the same dataset. We expect these animations to connect the visual elements between the source and target visualizations.

**Tips:**

1. Read in the data and then parse it into the format you want.
2. Most of you already know Java. Remember that Processing is built upon Java and you can use all the readers that you already know.
3. However, Processing offers some simpler methods like createInput(), createReader(), loadStrings(), loadTable() (Note that loadTable() does not work when you convert your application to JavaScript. ).
4. Below is an example of loadStrings(). You are welcome to use other advanced data structures and methods.

```
String path = "data.csv";                          // define the path

void loadStrings(){
    String[] lines = loadStrings(path);            // load the file as each line is a string
    String[] firstLine = split(lines[0], ",");     // the first line is the title
    xName = firstLine[0];                          // the name of the x-axis
    yName = firstLine[1];                          // the name of the y-axis
    names = new String [lines.length - 1];         // the rest is the record
                                                   // for the dimension has the categorical data
    values = new int[lines.length - 1];            // for the dimension has the numeric data

    for(int i = 1; i < lines.length; i++){         // traverse each line
        String[] row = split(lines[i], ",");       // split the line using ","
        names[i - 1] = row[0];
        values [i - 1] = (int) parseFloat(row[1]); // the second one should be the numeric data
                                                   // parseFloat is to convert the string to the float
                                                   // Or you can use Integer.parseInt. It is a method in
                                                   Java to convert string to integers.
    }
}
print(xName + "," + yName);                        //check
print(names);
print(values);
```

**Notes:**

1. The system should be resizable and you may not assume a fixed dimension for your canvas. In other words, if the canvas size becomes larger, your visualization should also become larger. If the canvas size becomes smaller, your visualization should not be "cut off".
    a. To add resizing to your Processing code, add the following line in your setup function:

```
surface.setResizable(true);
```

   b. Note that you need to remove this line is you wish to post the program online. This line of code is not compatible with Processing.js and will prevent your code from running in a browser.
   c. Pay attention to the size of the spacing (e.g., spacing between the bars and circles). The spacing should not be a static number (e.g., hard coded to 10 pixels), but should be dynamically determined based on the canvas size and the number of data elements.
   d. If done right, I should be able to load another data file of any size into your application.
2. Do not assume a fixed range for neither the X- nor Y-axis. For instance, do not assume that the values in the Y-axis are always between 0 and 100. Determine the lower and upper bound based on what you read from the data and scale the Y-axis accordingly. Similarly, do not assume that the number of data items in the X-axis is always below a certain number. Your visualization should adapt to the data that it is given and display the data in the most appropriate manner. In other words, you can't assume there is a hard-coded range of the dataset.
3. However, you may assume that the data will not be too big.
4. Your code should be modular. I recommend have a class for the bar chart and one for the line. Feel free to add as many classes as you see fit.

**Extra Credit Opportunity:**

1. Add a third chart of your choice.
2. Implement a 3-way animated transition. You should have three buttons (ideally on the right side of the screen), named Line Chart, Bar Chart and Chart_Of_Your_Choice. Each button corresponds to one of the three visualizations. When the user clicks a button, the canvas will transition to the corresponding visualization

**Submission:**

Submit your work via **Blackboard** by Tuesday, February 7 2016, 11:59pm (midnight). Use the naming convention: "FirstnameLastname_a2.zip" (e.g., AlvittaOttley_a2.zip).

**Need Help?**

If you have questions about this assignment:

1. First, check Piazza to see if others have had a similar problem.
2. If not, post your question on Piazza.
3. If the question is sensitive, please email the instructors privately.
4. Note that you are NOT allowed to post code or solutions on Piazza – I will monitor the forum and delete any inappropriate posts. Fishing for programming solutions could result in a penalty toward your grade.