CSE 564 LAB 1 REPORT

Student Id: 112077242
Name: Kajal Dalvi

- For project 1 using NBA (National Basketball Association) statistics data I have plotted bar chart by binning it into fixed range of default size 5 and default attribute 'Wins by teams'. This project helps in providing detailed analysis of the NBA statistics for various wins, points scored, points attempted range etc.

- Function **loadDataAndPlot()** is created which takes 2 input parameters-attribute name and attribute description. This function is used to plot charts by loading the data of given attribute from CSV file. There is also a dropdown option using which a different attribute can be selected to plot. On selecting different attribute new bar chart is plotted with values of new selected attribute by calling same **loadDataAndPlot()** with updated values.

- After loading CSV file I have calculated bin size by using maximum and minimum of data value and number of bins. Then by using the formula, **[(data value – min value)/bin size]**, I created an array of frequency values which I plot on y axis i.e. creating an array of values which will define data belonging to that particular bin.
  Also, using min value and bin size I created an array of bins which will be shown on x-axis. This helps to identify the range of particular bin.
  For example, 11-20, 21-30, 31-40 points etc.

```
function loadDataAndPlot(val, textVal) {
  d3.selectAll("svg > *").remove(); //clearing the svg before plotting new graph

  var svg = d3.select("svg"),
    margin = 200,
    width = svg.attr("width") - margin,
    height = svg.attr("height") - margin

  var xScale = d3.scaleBand().range([0, width]).padding(0.2);
  var yScale = d3.scaleLinear().range([height, 0]);
```

```
  for (i = 0; i < data.length; i++) {
    var temp = Math.floor((data[i][val] - minValue) / binSize)
    if (temp == numberOfBins) {
      temp--; //to handle edge case when boundary value is present at the range end
    }
    yBinArray[temp]++
  }
  //console.log("data value" + yBinArray)

  xBinArray = []
  var rangeStart = minValue
  for (i = 0; i < numberOfBins; i++) {
    rangeEnd = parseInt(rangeStart) + binSize -1
    xBinArray.push("" + rangeStart + "-" + rangeEnd)
    rangeStart = rangeEnd + 1
  }
  //console.log("x array " + xBinArray)

  xScale.domain(xBinArray);
  yScale.domain([0, d3.max(yBinArray, function(d) {
    return d;
  })]);
```

- Using the d3 functionality of **scaleBand** and **linearScale** I have defined xScale and yScale for given SVG width and height range, respectively. Array created for x-axis range values is set as domain for xScale and 0 to max frequency is set as domain for y range.
  To make actual rectangular bars I created 2D array, **plot2DArray**, with x and y values for each bin. This is set as data to create rectangular bins with xScale and yScale function.

- On **mouse-over** a particular bin designed interface displays the corresponding frequency value of that bin i.e. y-axis value for that bin and the selected bin is highlighted by increasing width and height to focus on it.
  On **mouse-out**, height and width of bin is set to the original value to remove the focus and also frequency value at the top of the bar is removed.

```
function handleMouseOver(d, i) {
  d3.select(this).attr("height", function(d) {
      return height - yScale(d[1]) + 8;
    })
    .attr("width", xScale.bandwidth() + 3)
    .attr("y", function(d) {
      return yScale(d[1]) - 8;
    })
    .attr("fill", "#800000");
  var txt = d3.select(this.nextSibling);
  txt.attr('opacity', '1');
}

function handleMouseOut(d, i) {
  d3.select(this).attr("height", function(d) {
      return height - yScale(d[1]);
    })
    .attr("width", xScale.bandwidth())
    .attr("y", function(d) {
      return yScale(d[1]);
    })
    .attr("fill", "#AA3311");
  var txt = d3.select(this.nextSibling);
  txt.attr('opacity', '0');
}
```

- To support the functionality of converting bar chart into pie chart on '**click**' event and vice versa, I added click listener on SVG. Function **convertToPie()** is called to draw pie chart with the same data.
  By passing the same 2D array **plot2DArray** created for bar chart data I have passed array of frequencies to d3.pie() function to divide pie chart blocks according to data values. Similar '**click**' event is added in pie chart to load bar chart with same data again.

- I have added **slider** which is used to change the number of bins and adjust the data values accordingly. On '**slider.oninput**' value of number of bins is updated and **loadDataAndPlot()** is called to redraw the bar chart according to updated values.

```
var slider = document.getElementById("binRange");
slider.oninput = function() {
  console.log("bin number values is " + this.value)
  numberOfBins = this.value;
  loadDataAndPlot(val, textVal)
}
```

Similar functionality can also be implemented with '**mousemove' event**. However, due to conflict in mouse-over and mousemove I choose an option to go with slider interface.

```
svg.on("mousemove", function() {
  if(state!= "pie") {
  var points = d3.mouse(this)[0]
  if(defaultX == 0)
  {
    defaultX = points
  }

//  console.log(points + " ...." + defaultX)
  if(points - defaultX >=50) {
    //console.log("change in x side by right" + (numberOfBins)+ ", "+defaultX + " points " +points);
    defaultX = points;
    console.log("change in x side by right updted" + (numberOfBins)+ ", "+defaultX + " points " +points);

    if(numberOfBins < 10) {
      numberOfBins++;
    }
    loadDataAndPlot(val, textVal, numberOfBins)
  } else if(defaultX - points >= 50) {
    console.log("change in y side by left"+ (numberOfBins) + ", "+defaultX + " points " +points);
    defaultX = points;
    if(numberOfBins > 2) {
      numberOfBins--;
    }
    loadDataAndPlot(val, textVal)
  }
}});
```

References:

- Code sample template for bar graph given by TA
- http://zeroviscosity.com/d3-js-step-by-step/step-1-a-basic-pie-chart - Pie chart
- https://www.w3schools.com/howto/howto_js_rangeslider.asp - slider
- http://jsfiddle.net/7E2L5/7/ - text on bar chart