Visualization Mini Project 1

1. Introduction:

   In this assignment, we learn to

   (1) Use JavaScript to import and process a large amount of data

   (2) Use the d3 package to present collected data via bar chart and pie chart.

   (3) Use the d3 package for elegant visual effect and animation for better data visualization.

   (4) Build an user friendly interface by using basic HTML elements and CSS syntax.

2. Requirement:

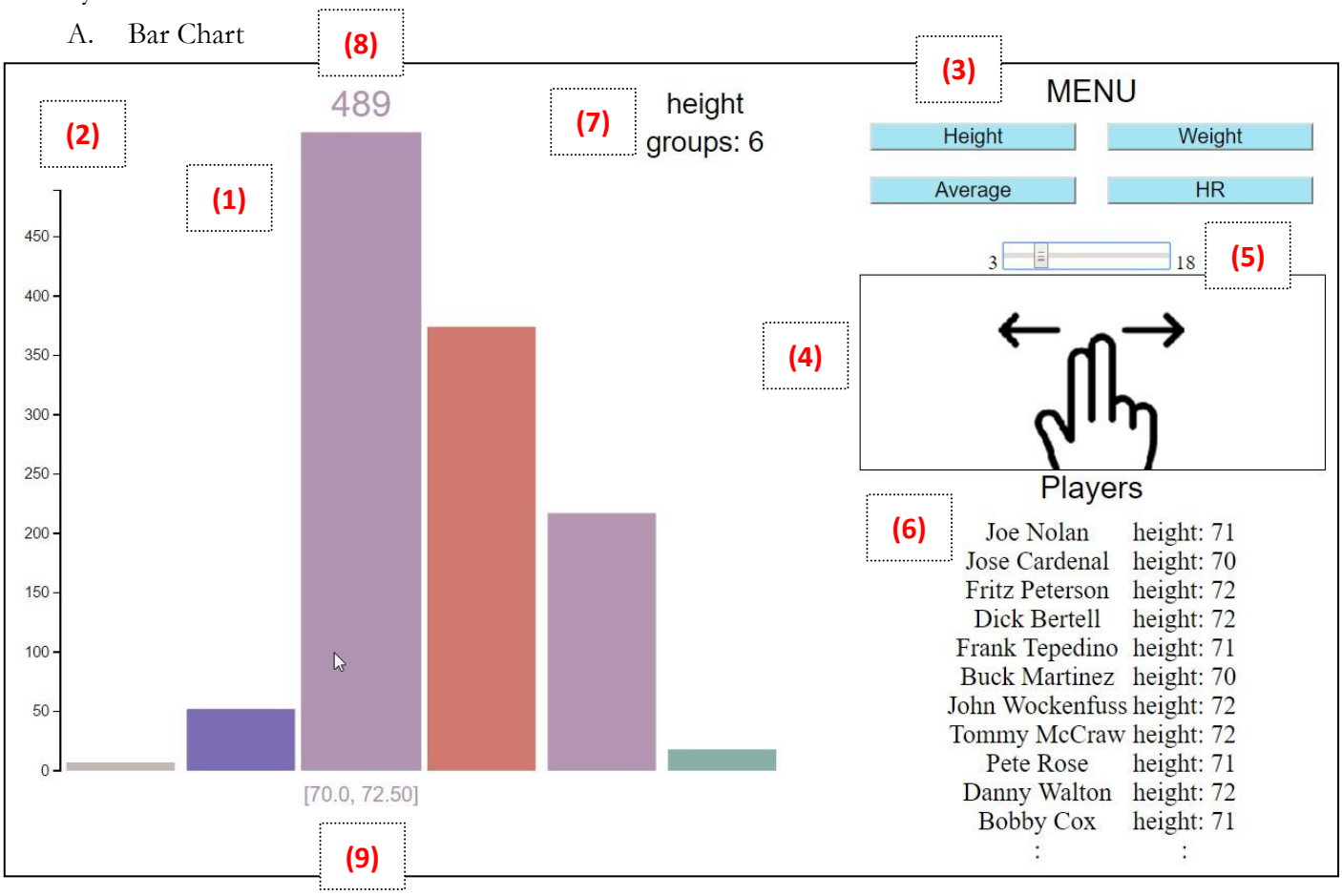
Your D3-based visual interface should be able to (all 10 pts):

1. pick a variable and bin it into a fixed range (equi-width) of your choice
2. create a bar chart of the variable you picked in 1.
3. using a menu, allow users to select a new variable and update chart
4. only on mouse-over display the value of the bar on top of the bar
5. on mouse-over make the bar wider and higher to focus on it
6. on mouse-click transform the bar chart into a pie chart (and back)
7. mouse moves left (right) should decrease (increase) bin width/size

An additional 10 pts for elegant implementation/function

3. Layout

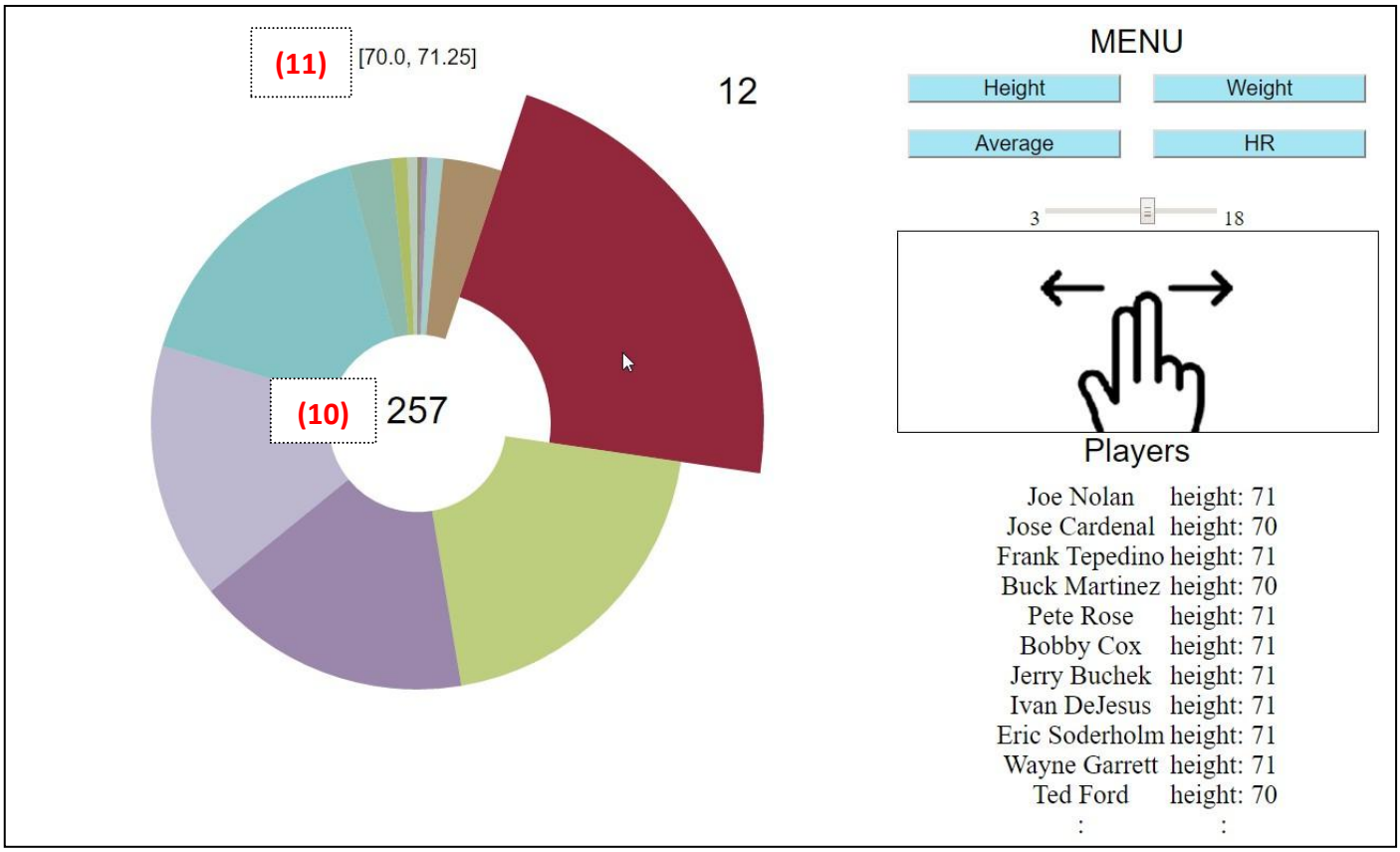
   A. Bar Chart



Components:

(1) Bar Chart
(2) Y-axis
(3) Statistic type: Allow user to choose a new variable and update chart
(4) Slider: Control the number of bins
(5) Scroll bar: Control the number of bins
(6) Display detail data
(7) Display current variable type and number of bins
(8) Display the range of the selected item
(9) Display the number of data in a given range of the selected item

B. Pie Chart



Components:

(10) Display the number of data in a given range of the selected item
(11) Display the range of the selected item

4. Code Implementation

A. Data Source:

https://docs.google.com/document/d/1w7KhqotVi5eoKE3I_AZHbsxdr-NmcWsLTIiZrpxWx4w/pub

-> Baseball data

B. Import CSV data:

Pre-process the CSV file into a JavaScript file with a parameter.

```
var csvDataSet = [
    {name: "Tom Brown" , hand: "R" , height:    73  , weight:   170 , avg:  0    , HR:   0   },
    {name: "Denny Lemaster" , hand: "R" , height:   73  , weight:   182 , avg:  0.13    , HR:   4   },
    {name: "Joe Nolan" , hand: "L" , height:    71  , weight:   175 , avg:  0.263   , HR:   27  },
    {name: "Denny Doyle" , hand: "L" , height:  69  , weight:   175 , avg:  0.25    , HR:   16  },
```

Include it at the beginning of <body> element.

```
    <!-- Pre-process CSV data into a variable in JavaScript file -->
    <script src="baseball_data.js"></script>
```

Note: I tried to use d3.csv to extract data from the CSV file, as below. However, when I use this function, all of my onClick event associated with <button> and are out of function. So, I use another way to import CSV data.

C. Create a SVG element

```
        // Build a svg element
        var svg_w = 600, svg_h = 600;
        var svg = d3.select("body").append("svg")
        .attr("class", "svg")
        .attr("width", svg_w)
        .attr("height", svg_h);
```

D. Separate the input data into different bins based on the number of bins and the chosen variable

```
// Seperate data into different bins
function buildBins(mode, dataset, bin_number, field){

    var index = 0, binDataSet = [], binDataCount = [], scale, lowerBound = [], upperBound = [],
        binWidth = 0;
    for (var i = 0; i < bin_number; i++) {
        binDataSet[i] = [];
        binDataCount[i] = 0;
        lowerBound[i] = 0;
        upperBound[i] = 0;
    };

    switch(field){

        case "height":
            binWidth = Math.floor((max.height - min.height) / bin_number * 100) / 100;
            lowerBound[0] = min.height;
            upperBound[bin_number-1] = max.height;
            for (var i = 1; i < bin_number; i++) {
                lowerBound[i] = lowerBound[i-1] + binWidth;
                upperBound[i-1] = lowerBound[i];
            };
            for (var i = 0; i < dataset.length; i++) {
                index = Math.floor((dataset[i]["height"] - min.height) / binWidth);
                if(index == bin_number) index--;
                binDataSet[index].push([dataset[i]["height"], dataset[i]["name"]]);
                binDataCount[index]++;
            }
            break;
```

Find the range of a specific bar, and count the number of data lies in the range.

Collect all values in a range corresponding to a specific variable.

Return the data as a new object.

```
246             return {
247                 set: binDataSet,
248                 cnt: binDataCount,
249                 chart_mode: mode,
250                 bin_num: bin_number,
251                 field: field,
252                 color: color,
253                 scale: scale,
254                 lowerBound: lowerBound,
255                 upperBound: upperBound
256             };
```

E. Build a bar chart

```
260         // Build a bar chart
261         function buildBarChart(dataset){
262 
263             // Create bars
264             var bar_w = (svg_w - 40) / dataset.bin_num;
265             var bar_padding = bar_w * 0.1;
266             rects = svg.append("g")
267                 .selectAll("rect")
268                 .data(dataset.cnt)
269                 .enter()
270                 .append("rect")
271                 .attr("id", "bar_chart")
272                 .attr("fill", function(d, i){
273                     return "rgba(" + dataset.color[i][0] + ", " + dataset.color[i][1] + ", "
274                         + dataset.color[i][2] + ", 0.6)";
275                 })
276                 .attr("x", function(d, i){
277                     return 40 + (i * bar_w) + (bar_padding / 2);
278                 })
279                 .attr("y", function(d, i){
280                     return 0.9 * svg_h - dataset.scale(d);
281                 })
282                 .attr("height", function(d, i){
283                     return dataset.scale(d);
284                 })
285                 .attr("width", bar_w - bar_padding);
```

F. Add d3 mouse events: Click, MouseOver, MouseOut

```
301             // Add mouse events
302 
303             // Event 1: Click
304             rects.on("click", function(d, i){
305                 svg.selectAll("#bar_label").remove();
306                 svg.selectAll("#bar_chart").remove();
307                 buildChart(1, csvDataSet, dataset.bin_num, dataset.field);
308             });
```

Click event:

Remove all elements in SVG

Plot a pie chart

MouseOver event:

Add the value and range for the chosen bar

Increase the bar width and change the opacity

Add text in div elements to display detailed data

MouseOut event:

Remove all the changes in MouseOver event.

```
310             // Event 2: Mouse over
311             rects.on("mouseover", function(d, i){
312 
313                 d3.select(this)
314                     .style("fill", "rgba(" + dataset.color[i][0] + ", " + dataset.color[i][1] + ", "
315                         + dataset.color[i][2] + ", 1)")
316                     .attr("x", 40 + i * bar_w)
317                     .attr("width", bar_w)
318                     .attr("y", 0.9 * svg_h - (dataset.scale(d) * 1.1))
319                     .attr("height", dataset.scale(d) * 1.1);
320 
321                 // Add value above a bar
322                 svg.select("g")
323                     .append("text")
324                     .text(d)
325                     .attr("id","bar_label")
326                     .attr("x", 40 + ((i+0.5) * bar_w))
327                     .attr("y", 0.88 * svg_h - (dataset.scale(d) * 1.1))
328                     .style("text-anchor", "middle")
329                     .attr("font-family", "sans-serif")
330                     .attr("font-size", "28px")
331                     .style("fill", "rgba(" + dataset.color[i][0] + ", " + dataset.color[i][1]
332                         + ", " + dataset.color[i][2] + ", 1)");
```

```
334                  // Add range under a bar
335                  svg.select("g")
336                      .append("text")
337                      .text("[" + Math.floor(dataset.lowerBound[i]) + "."
338                          + (Math.floor(dataset.lowerBound[i] * 100) % 100) + ", "
339                          + Math.floor(dataset.upperBound[i]) + "."
340                          + (Math.floor(dataset.upperBound[i] * 100) % 100) + "]")
341                      .attr("id","bar_range")
342                      .attr("x", 40 + (i + 0.5) * bar_w)
343                      .attr("y", 0.94 * svg_h)
344                      .style("text-anchor", "middle")
345                      .attr("font-family", "sans-serif")
346                      .attr("font-size", "16px")
347                      .style("fill", "rgba(" + dataset.color[i][0] + ", " + dataset.color[i][1]
348                          + ", " + dataset.color[i][2] + ", 1)");
349 
350                  // Update player's statistic
351                  for (var j = 0; j < d && j < 11; j++) {
352                      d3.select("#player_value")
353                      .append("div")
354                      .attr("id", "#player_value_item")
355                      .append("text")
356                      .text(binData.field + ": " + binData.set[i][j][0]);
```

```
378             /** Event 3: Mouse out **/
379             rects.on("mouseout", function(d, i) {
380                 d3.select(this)
381                     .style("fill", "rgba(" + dataset.color[i][0] + ", " + dataset.color[i][1] + ", "
382                         + dataset.color[i][2] + ", 0.6)")
383                     .attr("x", 40 + i * bar_w + (bar_padding / 2))
384                     .attr("width", bar_w - bar_padding)
385                     .attr("y", 0.9 * svg_h - dataset.scale(d))
386                     .attr("height", dataset.scale(d));
387 
388                 // Remove text label
389                 d3.selectAll("#bar_label").remove();
390                 d3.selectAll("#bar_range").remove();
391                 d3.selectAll(".player").text(" ");
392             });
```

G. Build pie chart

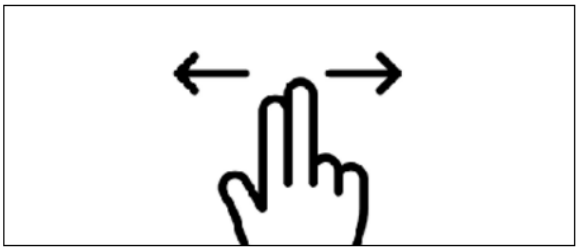
Similar to bar chart, when building a pie chart, we need to set the radius and append new "path" elements.

```
396         // Build pie chart
397         function buildPieChart(dataset){
398 
399             var pie_arc = d3.svg.arc()
400                 .outerRadius(Math.min(svg_w,svg_h)/3)
401                 .innerRadius(Math.min(svg_w,svg_h)/9);
402 
403             var pie = d3.layout.pie().sort(null) ;
404 
405             paths = svg.append("g")
406                 .attr("transform", "translate(" + svg_w/2 + "," + svg_h/2 + ")")
407                 .selectAll("path")
408                 .data(pie(binData.cnt))
409                 .enter()
410                 .append("path")
411                 .attr("id","pie_chart")
412                 .attr("d", pie_arc)
413                 .style("fill", function(d, i) {
414                     return "rgba(" + dataset.color[i][0]  + ", " + dataset.color[i][1] + ", "
415                         + dataset.color[i][2] + ", 0.6)"
416                 });
```

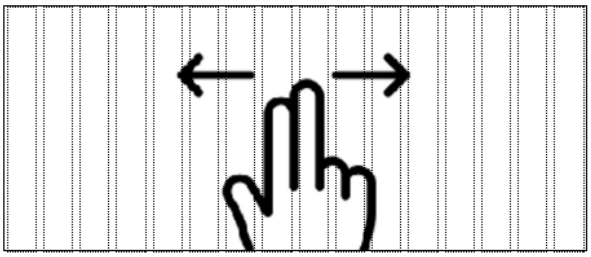
Also, add d3 mouse events: Click, MouseOver, MouseOut for the pie chart.

H. Add sliders to change the number of bins

The slider is actually composed by 15 div elements. By adding mouse event on every element, we can build a new bar chart or pie chart with the different bin number. And, the range of bin numbers is from 3 to 18.



Players



Players

```
546         <div class = slider>
547             <div class = "slider_inside" onmouseover="javascript:mySlider(3)"></div>
548             <div class = "slider_inside" onmouseover="javascript:mySlider(4)" ></div>
549             <div class = "slider_inside" onmouseover="javascript:mySlider(5)" ></div>
550             <div class = "slider_inside" onmouseover="javascript:mySlider(6)" ></div>
551             <div class = "slider_inside" onmouseover="javascript:mySlider(7)" ></div>
552             <div class = "slider_inside" onmouseover="javascript:mySlider(8)" ></div>
553             <div class = "slider_inside" onmouseover="javascript:mySlider(9)" ></div>
554             <div class = "slider_inside" onmouseover="javascript:mySlider(10)" ></div>
555             <div class = "slider_inside" onmouseover="javascript:mySlider(11)" ></div>
556             <div class = "slider_inside" onmouseover="javascript:mySlider(12)" ></div>
557             <div class = "slider_inside" onmouseover="javascript:mySlider(13)" ></div>
558             <div class = "slider_inside" onmouseover="javascript:mySlider(14)" ></div>
559             <div class = "slider_inside" onmouseover="javascript:mySlider(15)" ></div>
560             <div class = "slider_inside" onmouseover="javascript:mySlider(16)" ></div>
561             <div class = "slider_inside" onmouseover="javascript:mySlider(17)" ></div>
562             <div class = "slider_inside" onmouseover="javascript:mySlider(18)" ></div>
563         </div>
```

```
523         function mySlider(i){
524             buildChart(binData.chart_mode, csvDataSet, i, binData.field);
525             bin_disp.text("groups: " + binData.bin_num);
526         }
```

I. A sliding control bar is added for the same purpose of slider.

```
543          3 <input min="3" max="18" value="5" step="1" id="slider" name="range" onchange="mySlider(this.
                 value)" type="range"> 18
544          <div>
```

J. Add four buttons for user to select different variables: Height, Weight, Average, HR

```
535     <div class = "display_block">
536         <p class = "display_title">MENU</p>
537         <div class = "display_button">
538             <button onclick="plot('height')">Height</button>
539             <button onclick="plot('weight')">Weight</button>
540             <button onclick="plot('avg')">Average</button>
541             <button onclick="plot('HR')">HR</button>
```

```
510         /** Plot function for Buttons **/
511         function plot(item){
512             buildChart(binData.chart_mode, csvDataSet, binData.bin_num, item);
513             var_disp.text(binData.field);
514         }
```