EECS 6893: Big Data Analytics
HW3
Yutao Zhou UNI: yz4359
Part I
Problem 1
1.1.1
In SVG Coordinate origin is the top left of the screen and the x value increase as we move right and the y value increase as we move down the screen. In Mathematical/Graph Coordinate Space the origin is the center, the x value increases as we move right, and the y value increases as we move up.
1.1.2
Enter and exit are two functions in d3.js. Enter is used to add data into DOM element when data have more entry than DOM element. For example, when DOM have element ‘1’,’2’ and our data have element ‘1’,’2’,’3’, we could use enter to add element ‘3’ into DOM element. Exit is used to remove DOM element when data is shorter than DOM element. For example, when DOM have element ‘1’,’2’,’3’ and our data have element ‘1’,’2’, we could use exit to remove element ‘3’ from DOM element.
1.1.3
Transform is an attribute in SVG that contains a bunch of transformation that we could apply to an element. In other words, transform apply the same transformation to all of the points of an element. Translate is a simpecific transformation in the Transform attribute. Translate is used to move an element by x and y, where x is distance move horizontally and y is distance move verticallys.
1.1.4
The return value of this anonymous function is a = [5, 6, 7, 8, 9]
1.1.5

The output of the 2 code snippets will be different. In the snippet II the enter and append will add data into the body with a space between each of them(the entire data list will be pint out where there will be a space between each elements). Where in snippet I the first element will been used to replace the body(only the first element will been print out).

Problem 2

```
<!DOCTYPE html>
<meta charset="utf-8">
<html>
  <head>
    <script src="https://d3js.org/d3.v4.js"></script>
    <link rel="stylesheet" href="1.2.5.css">
  </head>

  <body>

  <script>

    d3.csv("https://raw.githubusercontent.com/vega/vega/main/docs/data/seattle-weather.csv", function(data) {
        time = (data.map(function(d) { return d3.timeParse("%Y-%m-%d")(d.date)}))

        //#### Start of 1.2.1 histogram with bin 10 ####

        var marginHis10 = {top: 20, right: 50, bottom: 50, left: 60},
          widthHis10 = 460 - marginHis10.left - marginHis10.right,
          heightHis10 = 400 - marginHis10.top - marginHis10.bottom;

        var svg = d3.select("body")
          .append("svg")
            .attr("width", widthHis10 + marginHis10.left + marginHis10.right)
            .attr("height", heightHis10 + marginHis10.top + marginHis10.bottom)
          .append("g")
            .attr("transform",
                  "translate(" + marginHis10.left + "," + marginHis10.top + ")");

        svg.append("text")
            .attr("x", widthHis10 / 2)
            .attr("y", -8)
            .attr("text-anchor", "middle")
            .style("font-size", "16px")
            .style("font-weight", "bold")
            .text("Histogram With Bin Of 10");

        svg.append("circle").attr("cx",widthHis10 - 110).attr("cy",10).attr("r", 6).style("fill", "#69b3a2")
        svg.append("text").attr("x", widthHis10 - 100).attr("y", 10).text("Wind Speed").style("font-size", "15px").attr("
          alignment-baseline","middle")

        var x = d3.scaleLinear()
          .domain([0,10])
          .range([0, widthHis10]);
        svg.append("g")
          .attr("transform", "translate(0," + heightHis10 + ")")
          .call(d3.axisBottom(x));
        svg.append("text")
          .attr("transform",
                "translate(" + (widthHis10/2) + " ," +
                               (heightHis10 + marginHis10.top + 20) + ")")
          .style("text-anchor", "middle")
          .text("Wind Speed");

        var histogram = d3.histogram()
          .value(function(d) { return d.wind; })
          .domain(x.domain())
```

```
    .thresholds(x.ticks(10));

  var bins = histogram(data);

  var y = d3.scaleLinear()
    .domain([0, d3.max(bins, function(d) { return d.length; })])
    .range([heightHis10, 0])
  svg.append("g")
    .text("Number of days")
    .call(d3.axisLeft(y));
  svg.append("text")
    .attr("transform", "rotate(-90)")
    .attr("y", 0 - marginHis10.left)
    .attr("x",0 - (heightHis10 / 2))
    .attr("dy", "1em")
    .style("text-anchor", "middle")
    .text("Number of days");

  svg.selectAll("rect")
    .data(bins)
    .enter()
    .append("rect")
      .attr("x", 1)
      .attr("transform", function(d) { return "translate(" + x(d.x0) + "," + y(d.length) + ")"; })
      .attr("width", function(d) { return x(d.x1) - x(d.x0) - 1 ; })
      .attr("height", function(d) { return heightHis10 - y(d.length); })
      .style("fill", "#69b3a2")

  //#### Start of 1.2.2 histogram with bin 25 ####

  var marginHis25 = {top: 20, right: 50, bottom: 50, left: 60},
      widthHis25 = 460 - marginHis25.left - marginHis25.right,
      heightHis25 = 400 - marginHis25.top - marginHis25.bottom;

  var svg = d3.select("body")
    .append("svg")
      .attr("width", widthHis25 + marginHis25.left + marginHis25.right)
      .attr("height", heightHis25 + marginHis25.top + marginHis25.bottom)
    .append("g")
      .attr("transform",
            "translate(" + marginHis25.left + "," + marginHis25.top + ")");

  svg.append("text")
      .attr("x", widthHis25 / 2)
      .attr("y", -8)
      .attr("text-anchor", "middle")
      .style("font-size", "16px")
      .style("font-weight", "bold")
      .text("Histogram With Bin Of 25");

  svg.append("circle").attr("cx",widthHis10 - 110).attr("cy",10).attr("r", 6).style("fill", "#69b3a2")
  svg.append("text").attr("x", widthHis10 - 100).attr("y", 10).text("Wind Speed").style("font-size", "15px").attr("
    alignment-baseline","middle")

  var x = d3.scaleLinear()
    .domain([0,10])
    .range([0, widthHis25]);
```

```
        svg.append("g")
          .attr("transform", "translate(0," + heightHis25 + ")")
          .call(d3.axisBottom(x));
        svg.append("text")
          .attr("transform",
                "translate(" + (widthHis25/2) + " ," +
                               (heightHis25 + marginHis25.top + 20) + ")")
          .style("text-anchor", "middle")
          .text("Wind Speed");

        var histogram = d3.histogram()
          .value(function(d) { return d.wind; })
          .domain(x.domain())
            .thresholds(x.ticks(25));

        var bins = histogram(data);

        var y = d3.scaleLinear()
          .domain([0, d3.max(bins, function(d) { return d.length; })])
          .range([heightHis25, 0])
        svg.append("g")
          .text("Number of days")
          .call(d3.axisLeft(y));
        svg.append("text")
          .attr("transform", "rotate(-90)")
          .attr("y", 0 - marginHis25.left)
          .attr("x",0 - (heightHis25 / 2))
          .attr("dy", "1em")
          .style("text-anchor", "middle")
          .text("Number of days");

        svg.selectAll("rect")
          .data(bins)
          .enter()
          .append("rect")
            .attr("x", 1)
            .attr("transform", function(d) { return "translate(" + x(d.x0) + "," + y(d.length) + ")"; })
            .attr("width", function(d) { return x(d.x1) - x(d.x0) -1 ; })
            .attr("height", function(d) { return heightHis25 - y(d.length); })
            .style("fill", "#69b3a2")

        //#### Start of 1.2.3 pie chart ####

        var widthPie = 400,
          heightPie = 400,
          radius = (Math.min(widthPie, heightPie)-50) / 3;

        var color = d3.scaleOrdinal(d3.schemeCategory10);

        var arc = d3.arc()
          .outerRadius(radius - 10)
          .innerRadius(0);

        var pie = d3.pie()
          .sort(null)
          .value(function(d) {
            return d.value;
```

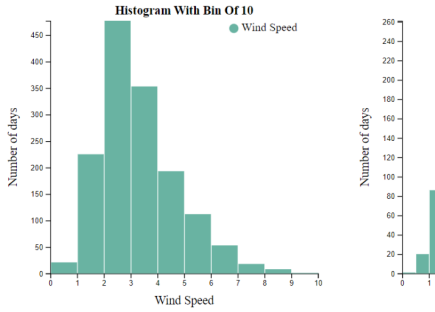
```
    });

var svg = d3.select("body")
    .append("svg")
    .attr("width", widthPie)
    .attr("height", heightPie)
      .append("g")
      .attr("transform", "translate(" + widthPie / 2 + "," + heightPie / 2 + ")");

svg.append("text")
    .attr("x", 0)
    .attr("y", -widthPie / 2 + 30)
    .attr("text-anchor", "middle")
    .style("font-size", "16px")
    .style("font-weight", "bold")
    .text("Pie Chart of Weather in Seattle");

//reduce by key; key : weather; value: count
var dataPie = d3.nest()
  .key(function(d) {
    return d.weather;
  })
  .rollup(function(d) {
    return d3.sum(d, function(g) {
      return 1;
    });
  }).entries(data);

//total count for percentage calculation
var total = d3.sum(dataPie, function(d) {
        return d.value;
      });

//percentage calculation
dataPie.forEach(function(d) {
          d.percentage = (d.value  / total * 100).toFixed(2);
  });

var counter = 0
var c = ["rgb(31, 119, 180)","rgb(255, 127, 14)","rgb(44, 160, 44)","rgb(214, 39, 40)","rgb(148, 103, 189)"]
var name = ["drizzle", "rain", "sun", "snow", "fog"]
dataPie.forEach(function(d) {
  svg.append("circle").attr("cx",widthPie / 2 - 110).attr("cy",-widthPie / 2 + 50 +counter * 15).attr("r", 6).style("fill", c[
    counter])
  svg.append("text").attr("x", widthPie / 2 - 100).attr("y", -widthPie / 2 + 50 +counter * 15).text(name[counter]).style("
    font-size", "15px").attr("alignment-baseline","middle")
  counter = counter + 1
});

var g = svg.selectAll(".arc")
  .data( pie(dataPie) )
  .enter()
    .append("g")
    .attr("class", "arc");

g.append("path")
  .attr("d", arc)
```

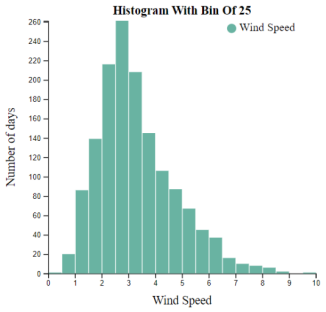
```
            .style("fill", function(d) {
              return color(d.data.key);
            });

          g.append("text")
            .attr("transform", function(d) {
              var _d = arc.centroid(d);
              _d[0] *= 2.7;
              _d[1] *= 2.7;
              return "translate(" + _d + ")"

            })
            .attr("dy", ".35em")
            .style("text-anchor", "middle")
            .text(function(d) {
              return d.data.key;
            });

          g.append("text")
            .attr("transform", function(d) {
              var _d = arc.centroid(d);
              _d[0] *= 2.7;
              _d[1] *= 2.7;
              return "translate(" + _d + ")"

            })
            .attr("dy", "2.0em")
            .style("text-anchor", "middle")
            .text(function(d) {
              return d.data.percentage + '%';
            });

          //#### Start of 1.2.4 line graph ####

          var marginLine = {top: 20, right: 50, bottom: 50, left: 60},
              widthLine = 600 - marginLine.left - marginLine.right,
              heightLine = 337.5 - marginLine.top - marginLine.bottom;

          var svg = d3.select("body")
            .append("svg")
              .attr("width", widthLine + marginLine.left + marginLine.right)
              .attr("height", heightLine + marginLine.top + marginLine.bottom)
            .append("g")
              .attr("transform",
                    "translate(" + marginLine.left + "," + marginLine.top + ")");

          svg.append("text")
              .attr("x", widthLine / 2)
              .attr("y", -8)
              .attr("text-anchor", "middle")
              .style("font-size", "16px")
              .style("font-weight", "bold")
              .text("Line Graph of Precipitation vs Date");

          svg.append("line").attr("x1", widthHis10 - 110).attr("y1", 10).attr("x2", widthHis10 - 80).attr("y2", 10).style("stroke", "
            steelblue").style("stroke-width", 3)
          svg.append("text").attr("x", widthHis10 - 70).attr("y", 10).text("Precipitation").style("font-size", "15px").attr("
```

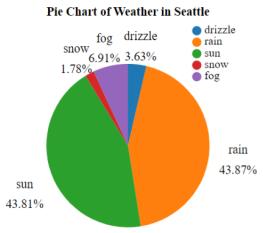
```
            alignment-baseline","middle")

          var x = d3.scaleTime()
            .domain(d3.extent(data, function(d) { return d3.timeParse("%Y-%m-%d")(d.date); }))
            .range([ 0, widthLine ]);
          svg.append("g")
            .attr("transform", "translate(0," + heightLine + ")")
            .call(d3.axisBottom(x));
          svg.append("text")
            .attr("transform",
                  "translate(" + (widthLine/2) + " ," +
                                 (heightLine + marginLine.top + 20) + ")")
            .style("text-anchor", "middle")
            .text("Date");

          var y = d3.scaleLinear()
            .domain([0, d3.max(data, function(d) { return +d.precipitation; })])
            .range([ heightLine, 0 ]);
          svg.append("g")
            .call(d3.axisLeft(y));
          svg.append("text")
            .attr("transform", "rotate(-90)")
            .attr("y", 0 - marginLine.left)
            .attr("x",0 - (heightLine / 2))
            .attr("dy", "1em")
            .style("text-anchor", "middle")
            .text("Precipitation");

          svg.append("path")
            .datum(data)
            .attr("fill", "none")
            .attr("stroke", "steelblue")
            .attr("stroke-width", 1.5)
            .attr("d", d3.line()
              .x(function(d) { return x(d3.timeParse("%Y-%m-%d")(d.date)) })
              .y(function(d) { return y(d.precipitation) })
              );

        });

    </script>

    <p>
      As we can see, the wind speed in Seattle is 1-4 most of the time. The distribution is mainly binomial. About 44% of the time is sunny
      and another 44% of the time is raining. So, most of the time Seattle is wither sunny or rainy. Most of the precipitation happens in
      spring and summer. Adding up all of the observations, we could conclude that Seattle is a city that has the characteristics of both
      the Mediterranean zone and oceanic zone.
    </p>
    </body>

  </html>
```

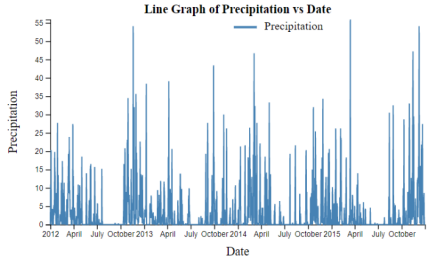
```css
p {
    font-family: Arial, Helvetica, Sans Serif;
    font-size: xx-large;
    color: LightSkyBlue;
    text-indent: 3%;
  }
```

As we can see, the wind speed in Seattle is 1-4 most of the time. The distribution is mainly binomial. About 44% of the time is sunny and another 44% of the time is raining. So, most of the time Seattle is wither sunny or rainy. Most of the precipitation happens in spring and summer. Adding up all of the observations, we could conclude that Seattle is a city that has the characteristics of both the Mediterranean zone and oceanic zone.







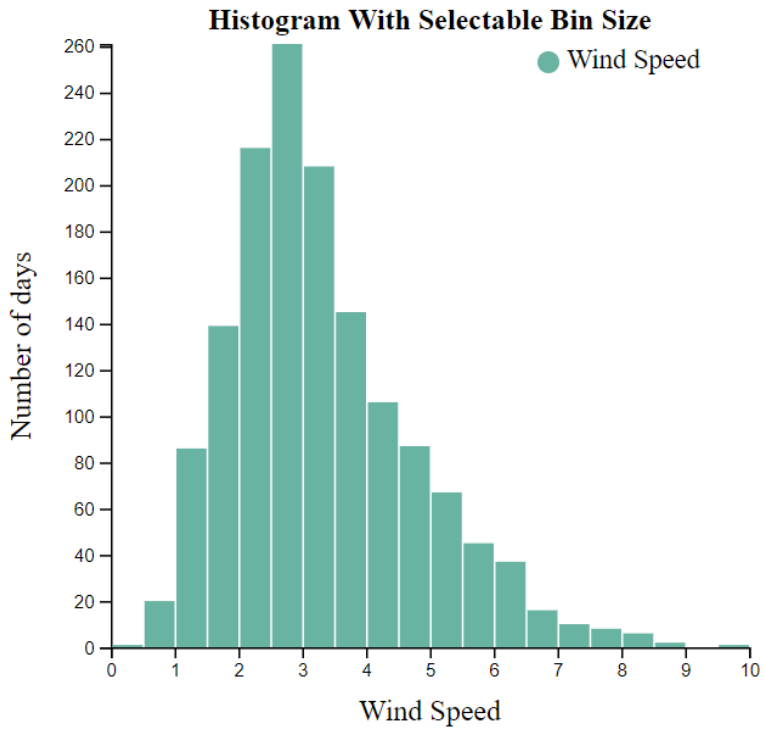


Problem 3

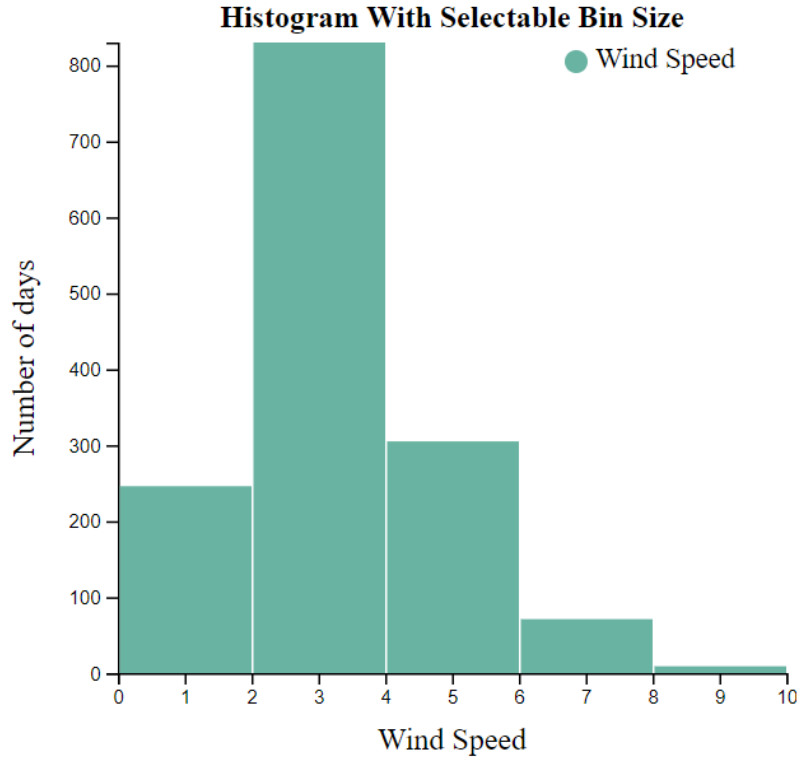
1.3.1

```
<!DOCTYPE html>
<meta charset="utf-8">
<script src="https://d3js.org/d3.v4.js"></script>
<div id="bin_select_histogram"></div>

<p>
  <label>number of bins</label>
  <input type="number" min="5" max="25" step="5" value="15" id="bin_num">
</p>

<script>

var margin = {top: 20, right: 50, bottom: 50, left: 60},
    width = 460 - margin.left - margin.right,
    height = 400 - margin.top - margin.bottom;

var svg = d3.select("#bin_select_histogram")
  .append("svg")
    .attr("width", width + margin.left + margin.right)
    .attr("height", height + margin.top + margin.bottom)
  .append("g")
    .attr("transform",
          "translate(" + margin.left + "," + margin.top + ")");

    svg.append("text")
      .attr("x", width / 2)
      .attr("y", -8)
      .attr("text-anchor", "middle")
      .style("font-size", "16px")
      .style("font-weight", "bold")
      .text("Histogram With Selectable Bin Size");

    svg.append("circle").attr("cx",width - 110).attr("cy",10).attr("r", 6).style("fill", "#69b3a2")
    svg.append("text").attr("x", width - 100).attr("y", 10).text("Wind Speed").style("font-size", "15px").attr("alignment-baseline","
      middle")

d3.csv("https://raw.githubusercontent.com/vega/vega/main/docs/data/seattle-weather.csv", function(data) {
    var x = d3.scaleLinear()
      .domain([0, 10])
      .range([0, width]);
    svg.append("g")
      .attr("transform", "translate(0," + height + ")")
      .call(d3.axisBottom(x));
    svg.append("text")
      .attr("transform",
            "translate(" + (width/2) + " ," +
                           (height + margin.top + 20) + ")")
      .style("text-anchor", "middle")
      .text("Wind Speed");

    var y = d3.scaleLinear()
      .range([height, 0]);
    var yAxis = svg.append("g")
    svg.append("text")
      .attr("transform", "rotate(-90)")
      .attr("y", 0 - margin.left)
      .attr("x",0 - (height / 2))
```

```
        .attr("dy", "1em")
        .style("text-anchor", "middle")
        .text("Number of days");

    function update(bin_num) {

        var histogram = d3.histogram()
            .value(function(d) { return d.wind; })
            .domain(x.domain())
            .thresholds(x.ticks(bin_num));

        var bins = histogram(data);

        y.domain([0, d3.max(bins, function(d) { return d.length; })]);
        yAxis
            .transition()
            .duration(1000)
            .call(d3.axisLeft(y));

        var u = svg.selectAll("rect")
            .data(bins)
        u
          .enter()
          .append("rect")
          .merge(u)
          .transition()
          .duration(1000)
            .attr("x", 1)
            .attr("transform", function(d) { return "translate(" + x(d.x0) + "," + y(d.length) + ")"; })
            .attr("width", function(d) { return x(d.x1) - x(d.x0) -1 ; })
            .attr("height", function(d) { return height - y(d.length); })
            .style("fill", "#69b3a2")

        u
          .exit()
          .remove()

        }

    update(15)

    d3.select("#bin_num").on("input", function() {
      update(+this.value);

    });

});

</script>
```



number of bins 15



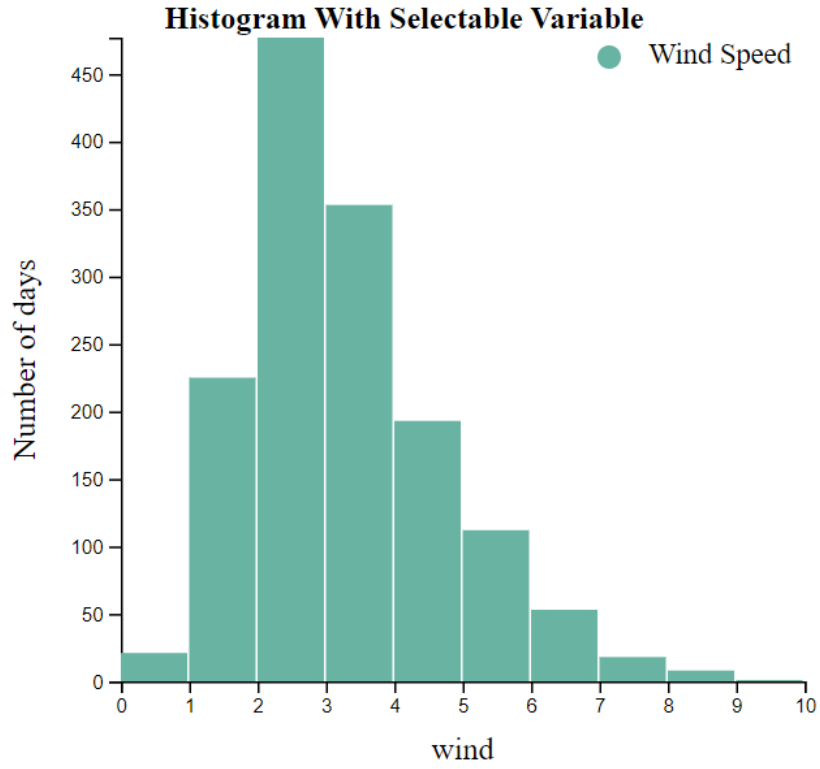
number of bins 5

1.3.2

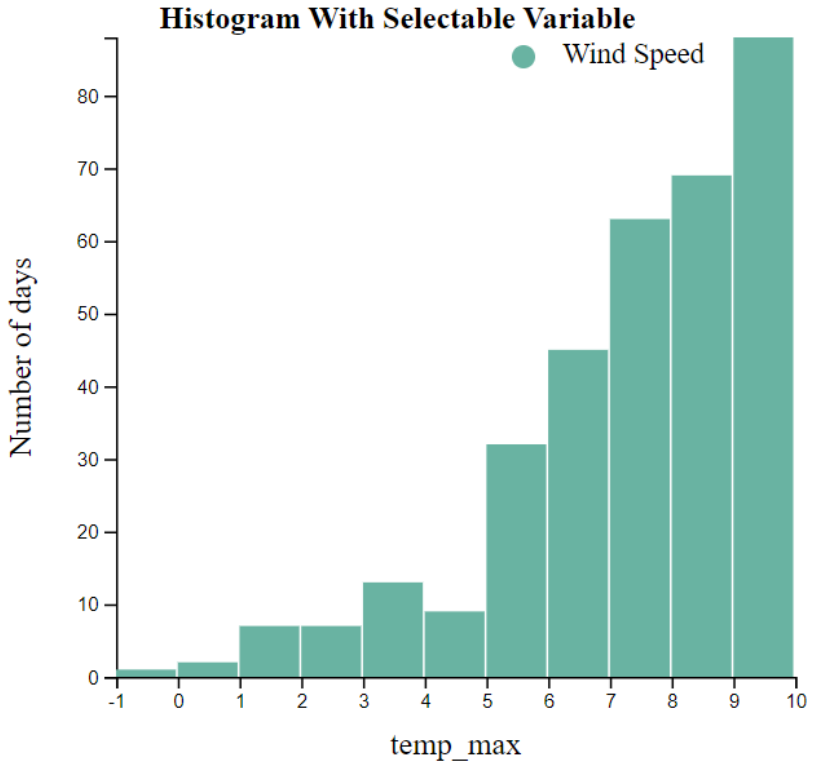
```
<!DOCTYPE html>
<meta charset="utf-8">
<script src="https://d3js.org/d3.v4.js"></script>
<div id="data_select_histogram"></div>

<body>
  <label>Data </label>
  <select id="option">
      <option value="wind" selected>wind</option>
      <option value="precipitation">precipitation</option>
      <option value="temp_max">temp max</option>
      <option value="temp_min">temp min</option>
  </select>

  <script>
  var margin = {top: 20, right: 50, bottom: 50, left: 60},
      width = 460 - margin.left - margin.right,
      height = 400 - margin.top - margin.bottom;

  var svg = d3.select("#data_select_histogram")
    .append("svg")
      .attr("width", width + margin.left + margin.right)
      .attr("height", height + margin.top + margin.bottom);

      svg.append("text")
        .attr("x", width / 2 + 30)
        .attr("y", 15)
        .attr("text-anchor", "middle")
        .style("font-size", "16px")
        .style("font-weight", "bold")
        .text("Histogram With Selectable Variable");

      svg.append("circle").attr("cx",width - 80).attr("cy",30).attr("r", 6).style("fill", "#69b3a2")
      svg.append("text").attr("x", width - 60).attr("y", 30).text("Wind Speed").style("font-size", "15px").attr("alignment-baseline","
        middle")

  var container = svg
    .append("g")
      .attr("transform",
            "translate(" + margin.left + "," + margin.top + ")");

  function histogram(option) {
    d3.csv("https://raw.githubusercontent.com/vega/vega/main/docs/data/seattle-weather.csv", function(data) {

      // pick data by option
      const xFeature = (d) => d[option];
      const yFeature = (d) => d.length;

      // set up xticks and yticks
      var x = d3.scaleLinear()
        .domain(d3.extent(data, xFeature))
        .range([0, width])
        .nice();

      // draw bar x axis and x title
      container.append("g")
        .attr("transform",
```

```
                "translate(" + 0 + " ," + height + ")")
            .call(d3.axisBottom(x));

        container.append("text")
            .attr("transform",
                  "translate(" + (width/2) + " ," +
                                 (height + margin.top + 20) + ")")
            .style("text-anchor", "middle")
            .text(option);

        var histogram = d3.histogram()
            .value(xFeature)
            .domain(x.domain())
            .thresholds(10);

        var bins = histogram(data);

        var y = d3.scaleLinear()
          .domain([0, d3.max(bins, yFeature)])
          .range([height, 0])

        // draw bar y axis and y title
        container.append("g")
          .call(d3.axisLeft(y));

        container.append("text")
          .attr("transform", "rotate(-90)")
          .attr("y", 0 - margin.left)
          .attr("x", 0 - (height / 2))
          .attr("dy", "1em")
          .style("text-anchor", "middle")
          .text('Number of days');

        //draw bars
        bars = container
        .append("g").selectAll("rect")
          .data(bins)
          .enter()
          .append("rect")
          .attr("height", function(d) { return height - y(yFeature(d)); })
          .attr('width', function(d) { return x(d.x1) - x(d.x0) -1 ; })
          .attr("x", (d) => x(d.x0) )
          .attr("y", (d) => y(yFeature(d)) )
          .style("fill", "#69b3a2")

    });

    }

    d3.select('#option')
      .on('change', function (e) {
        // clear the container for replotting
        container.selectAll('*').remove();
        histogram(this.value);
      });

    histogram('wind')
```

```
114
115       </script>
116     </body>
117     </html>
```



Data wind



Data temp max

Part II

2.1

```html
<!DOCTYPE html>
<meta charset="utf-8">
<html>
  <head>
    <script src="https://d3js.org/d3.v4.js"></script>
  </head>

  <body>
    <script type = "text/javascript" src="2.1.js"></script>
  </body>
</html>
```

```javascript
function displayTable(data, displayRows){
      if (displayRows==null) {
          var slicedData = data;
        }
      else {var slicedData = data.slice(0, displayRows);}

      var table = d3.select('body').append('table')
      var thead = table.append('thead')
      var tbody = table.append('tbody')

      thead.selectAll('th')
        .data(d3.map(slicedData[0]).keys())
        .enter()
        .append('th')
        .text(function (d) { return d });

      var rows = tbody.selectAll('tr')
        .data(slicedData)
        .enter()
        .append('tr');

      var cells = rows.selectAll('td')
        .data(function(row) {return d3.map(row).values(); })
        .enter()
        .append('td')
        .text(function (d) { return d })

    }

    d3.csv('https://raw.githubusercontent.com/plotly/datasets/master/auto-mpg.csv',function (data) {
      //display first 5 rows
      displayTable(data, 5);

      //groupby model year and count
      var countByModelYear = d3.nest()
        .key(function(d) { return d['model-year'];})
        .rollup(function(dd) {
        return d3.sum(dd, function(g) {return 1; });
        })
        .entries(data)
        .map(function(data) {
          return {
            'model': data.key,
            'count': data.value,
```

```
45              }
46            });
47
48          displayTable(countByModelYear, null);
49
50          //groupby model year and sum by cylinders
51          var countByModelYearCylinder = d3.nest()
52              .key(function(d) { return d['model-year'];})
53              .rollup(function(dd) {
54                return d3.sum(dd, function(d) {return d.cylinders; });
55              })
56              .entries(data)
57              .map(function(data) {
58                return {
59                  'number of cylinders': data.key,
60                  'total count': data.value,
61                }
62              });
63
64          displayTable(countByModelYearCylinder, null);
65
66          //groupby acceleration and count
67          var countByAccelerationCount = d3.nest()
68              .key(function(d) { return d.acceleration;})
69              .rollup(function(dd) {
70                return d3.sum(dd, function(d) {return 1; });
71              })
72              .entries(data)
73              .map(function(data) {
74                return {
75                  'acceleration': data.key,
76                  'count': data.value,
77                }
78              });
79          countByAccelerationCount.sort(function(x, y){
80                 return d3.ascending(x.acceleration, y.acceleration);
81              })
82
83          displayTable(countByAccelerationCount, null);
84
85
86        })
```

mpg	cylinders	displacement	horsepower	weight	acceleration	model-year
18	8	307	130	3504	12	70
15	8	350	165	3693	11.5	70
18	8	318	150	3436	11	70
16	8	304	150	3433	12	70
17	8	302	140	3449	10.5	70

model	count
70	29
71	28
72	28
73	40
74	27
75	30
76	34
77	28
78	36

79	29
80	29
81	29
82	31

number of cylinders	total count
70	196
71	156
72	163
73	255
74	142
75	168
76	192
77	153
78	193
79	169
80	120
81	134
82	130

acceleration	count
10	4
10.5	1
11	7
11.1	1
11.2	1
11.3	1
11.4	2
11.5	7
11.6	1
12	10
12.1	1
12.2	2
12.5	8
12.6	2
12.8	3

12.9	2	14.9	7	17.1	1
13	12	15	14	17.2	2
13.2	6	15.1	2	17.3	5
13.4	2	15.2	3	17.4	2
13.5	15	15.3	3	17.5	4
13.6	2	15.4	4	17.6	4
13.7	2	15.5	21	17.7	3
13.8	2	15.6	1	17.8	2
13.9	2	15.7	4	17.9	1
14	16	15.8	7	18	8
14.1	1	15.9	2	18.1	1
14.2	3	16	16	18.2	5
14.3	2	16.1	1	18.3	1
14.4	5	16.2	4	18.5	5
14.5	23	16.4	9	18.6	4
14.7	5	16.5	13	18.7	2
14.8	3	16.6	3	18.8	1
		16.7	3	19	12
		16.8	2	19.2	3
		16.9	4	19.4	3
		17	14	19.5	6

19.6	2
19.9	1
20.1	2
20.4	1
20.5	3
20.7	1
21	5
21.5	1
21.7	1
21.8	1
21.9	1
22.1	1
22.2	2
23.5	1
23.7	1
24.6	1
24.8	1
8	1
8.5	2
9	1
9.5	2

2.2

```html
<!DOCTYPE html>
<meta charset="utf-8">
<html>
  <head>
    <script src="https://d3js.org/d3.v4.js"></script>
  </head>

  <body>
    <script type = "text/javascript" src="2.2.js"></script>
  </body>
</html>
```

```javascript
function displayTable(data, displayRows){
      if (displayRows==null) {
          var slicedData = data;
        }
      else {var slicedData = data.slice(0, displayRows);}

      var table = d3.select('body').append('table')
      var thead = table.append('thead')
      var tbody = table.append('tbody')

      thead.selectAll('th')
        .data(d3.map(slicedData[0]).keys())
        .enter()
        .append('th')
        .text(function (d) { return d });

      var rows = tbody.selectAll('tr')
        .data(slicedData)
        .enter()
        .append('tr');

      var cells = rows.selectAll('td')
        .data(function(row) {return d3.map(row).values(); })
        .enter()
        .append('td')
        .text(function (d) { return d })

    }

    d3.csv('https://raw.githubusercontent.com/plotly/datasets/master/auto-mpg.csv',function (data) {

      //#### Start of 2.2.1 pie chart ####
      //groupby model year and count
      var pieData = d3.nest()
        .key(function(d) { return d['model-year'];})
        .rollup(function(dd) {
        return d3.sum(dd, function(g) {return 1; });
        })
        .entries(data)

      var countByModelYear = pieData
        .map(function(d) {
          return {
            'Model Year': d.key,
```

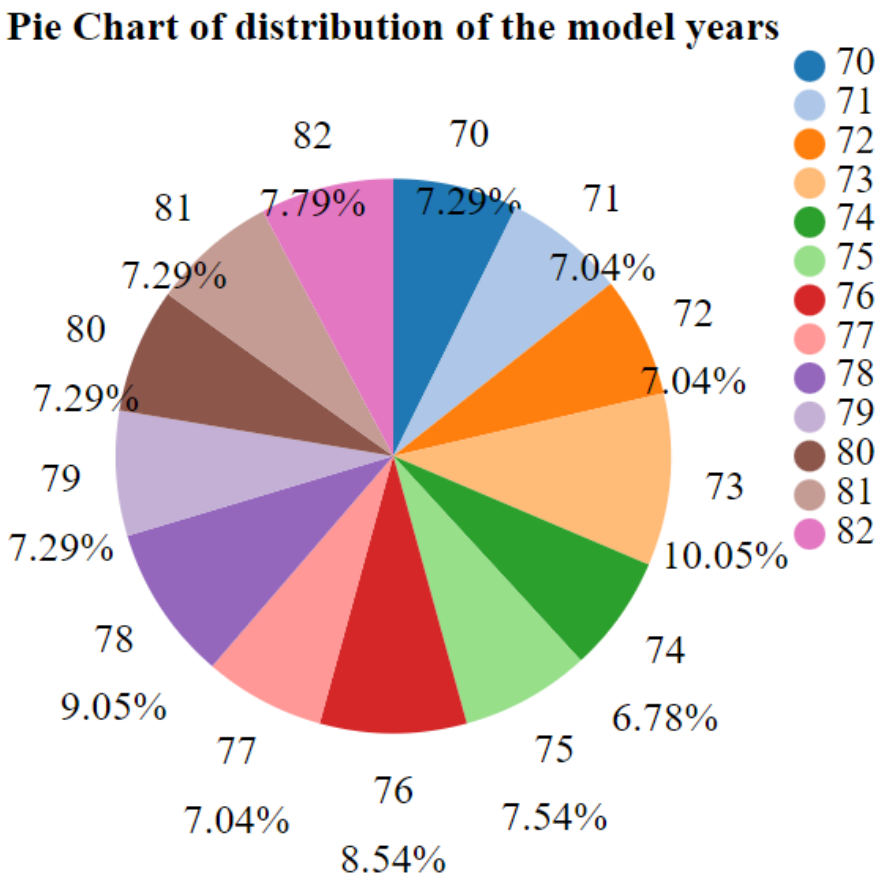
```
            'Count': d.value,
          }
        });

    var width = 400,
    height = 410,
    radius = (Math.min(width, height)-50) / 3;

    var color = d3.scaleOrdinal(d3.schemeCategory20);

    var arc = d3.arc()
      .outerRadius(radius - 10)
      .innerRadius(0);

    var pie = d3.pie()
      .sort(null)
      .value(function(d) {
        return d.value;
      });

    var svg = d3.select("body")
        .append("svg")
        .attr("width", width)
        .attr("height", height)
          .append("g")
          .attr("transform", "translate(" + width / 2 + "," + height / 2 + ")");

    svg.append("text")
        .attr("x", 0)
        .attr("y", -width / 2 + 40)
        .attr("text-anchor", "middle")
        .style("font-size", "16px")
        .style("font-weight", "bold")
        .text("Pie Chart of distribution of the model years");

    var counter = 0
    var c = ["rgb(31, 119, 180)","rgb(174, 199, 232)","rgb(255, 127, 14)","rgb(255, 187, 120)","rgb(44, 160, 44)",
      "rgb(152, 223, 138)","rgb(214, 39, 40)","rgb(255, 152, 150)","rgb(148, 103, 189)","rgb(197, 176, 213)",
      "rgb(140, 86, 75)","rgb(196, 156, 148)","rgb(227, 119, 194)"]
    var name = ["70", "71", "72", "73", "74", "75","76", "77", "78", "79", "80", "81","82"]
    pieData.forEach(function(d) {
      svg.append("circle").attr("cx",width / 2 - 40).attr("cy",-width / 2 + 50 +counter * 15).attr("r", 6).style("fill", c[counter])
      svg.append("text").attr("x", width / 2 - 30).attr("y", -width / 2 + 50 +counter * 15).text(name[counter])
          .style("font-size", "15px").attr("alignment-baseline","middle")
      counter = counter + 1
    });

    //total count for percentage calculation
    var total = d3.sum(pieData, function(d) {
            return d.value;
          });

    //percentage calculation
    pieData.forEach(function(d) {
              d.percentage = (d.value  / total * 100).toFixed(2);
      });
```

```
101
102         var g = svg.selectAll(".arc")
103           .data( pie(pieData) )
104           .enter()
105             .append("g")
106             .attr("class", "arc");
107
108         g.append("path")
109           .attr("d", arc)
110           .style("fill", function(d) {
111             return color(d.data.key);
112           });
113
114         g.append("text")
115           .attr("transform", function(d) {
116             var _d = arc.centroid(d);
117             _d[0] *= 2.4;
118             _d[1] *= 2.4;
119             return "translate(" + _d + ")"
120
121           })
122           .attr("dy", ".35em")
123           .style("text-anchor", "middle")
124           .text(function(d) {
125             return d.data.key;
126           });
127
128         g.append("text")
129           .attr("transform", function(d) {
130             var _d = arc.centroid(d);
131             _d[0] *= 2.4;
132             _d[1] *= 2.4;
133             return "translate(" + _d + ")"
134
135           })
136           .attr("dy", "2.0em")
137           .style("text-anchor", "middle")
138           .text(function(d) {
139             return d.data.percentage + '%';
140           });
141
142         displayTable(countByModelYear, null);
143
144         //#### Start of 2.2.2 line chart ####
145         //groupby model year and sum by cylinders
146         var lineGraph = d3.nest()
147             .key(function(d) { return d['model-year'];})
148             .rollup(function(dd) {
149               return d3.sum(dd, function(d) {return d.cylinders; });
150             })
151             .entries(data)
152
153         var countByModelYearCylinder = lineGraph
154             .map(function(data) {
155               return {
156                 'Model Year': data.key,
```

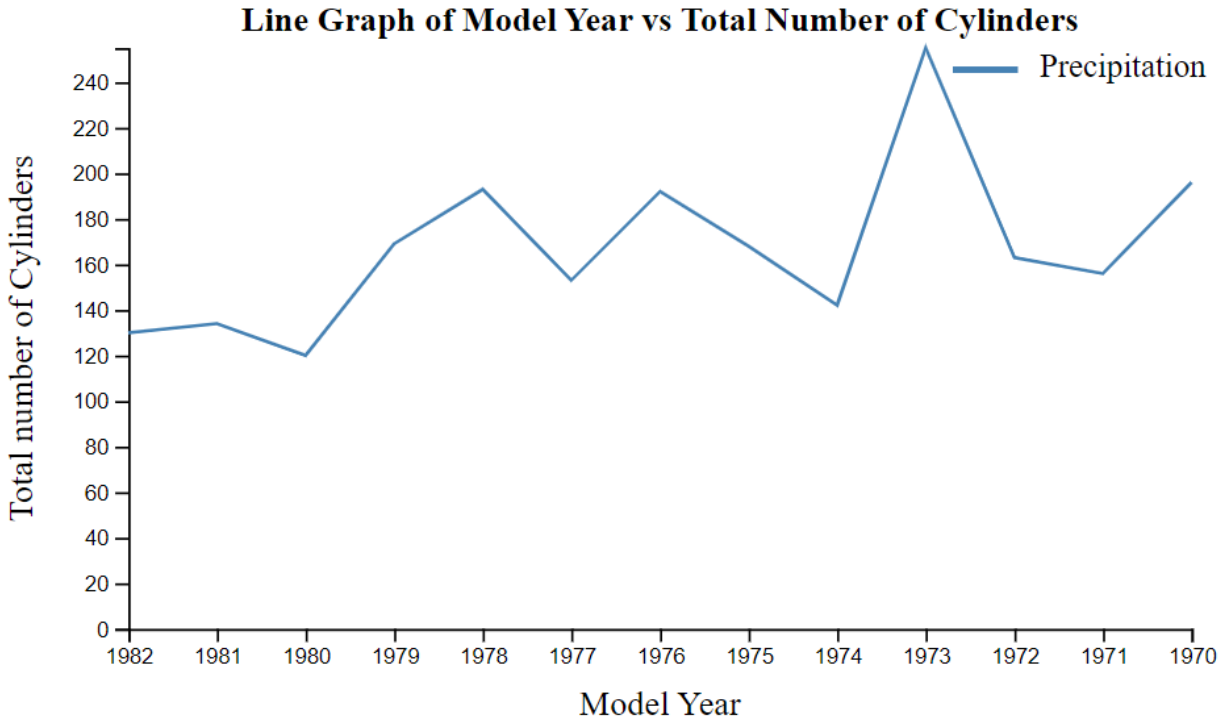
```
            'Number of Cylinders': data.value,
          }
        });

    var margin = {top: 20, right: 50, bottom: 50, left: 60},
        width = 600 - margin.left - margin.right,
        height = 337.5 - margin.top - margin.bottom;

    var svg = d3.select("body")
      .append("svg")
        .attr("width", width + margin.left + margin.right)
        .attr("height", height + margin.top + margin.bottom)
      .append("g")
        .attr("transform",
              "translate(" + margin.left + "," + margin.top + ")");

        svg.append("text")
          .attr("x", width / 2)
          .attr("y", -8)
          .attr("text-anchor", "middle")
          .style("font-size", "16px")
          .style("font-weight", "bold")
          .text("Line Graph of Model Year vs Total Number of Cylinders");

        svg.append("line").attr("x1", width - 110).attr("y1", 10).attr("x2", width - 80).attr("y2", 10).style("stroke", "steelblue")
            .style("stroke-width", 3)
        svg.append("text").attr("x", width - 70).attr("y", 10).text("Precipitation").style("font-size", "15px")
            .attr("alignment-baseline","middle")

    var parseTime = d3.timeParse("%y");
    lineGraph.forEach(function(d) {d.key = parseTime(d.key)})

    lineGraph.sort(function(x, y){
          return d3.descending(x.key, y.key);
        })

    var x = d3.scaleTime()
      .domain(d3.extent(lineGraph, function(d) { return d.key; }))
      .range([width, 0]);
    svg.append("g")
      .attr("transform", "translate(0," + height + ")")
      .call(d3.axisBottom(x));
    svg.append("text")
      .attr("transform",
            "translate(" + (width/2) + " ," +
                           (height + margin.top + 20) + ")")
      .style("text-anchor", "middle")
      .text("Model Year");

    var y = d3.scaleLinear()
      .domain([0, d3.max(lineGraph, function(d) { return +d.value; })])
      .range([ height, 0 ]);
    svg.append("g")
      .call(d3.axisLeft(y));
    svg.append("text")
      .attr("transform", "rotate(-90)")
```

```
213             .attr("y", 0 - margin.left)
214             .attr("x",0 - (height / 2))
215             .attr("dy", "1em")
216             .style("text-anchor", "middle")
217             .text("Total number of Cylinders");
218
219           svg.append("path")
220             .datum(lineGraph)
221             .attr("fill", "none")
222             .attr("stroke", "steelblue")
223             .attr("stroke-width", 1.5)
224             .attr("d", d3.line()
225               .x(function(d) { return x(d.key) })
226               .y(function(d) { return y(d.value) })
227               )
228
229           displayTable(countByModelYearCylinder, null);
230
231           //#### Start of 2.2.3 histogram ####
232           //groupby acceleration and count
233           var histogramData = d3.nest()
234               .key(function(d) { return d.acceleration;})
235               .rollup(function(dd) {
236                 return d3.sum(dd, function(d) {return 1; });
237               })
238               .entries(data)
239
240           var countByAccelerationCount = histogramData
241               .map(function(data) {
242                 return {
243                   'Acceleration': data.key,
244                   'Count': data.value,
245                 }
246               });
247
248           countByAccelerationCount.sort(function(x, y){
249                   return d3.ascending(x.acceleration, y.acceleration);
250               })
251
252           var margin = {top: 20, right: 50, bottom: 50, left: 60},
253               width = 600 - margin.left - margin.right,
254               height = 337.5 - margin.top - margin.bottom;
255
256           var svg = d3.select("body")
257             .append("svg")
258               .attr("width", width + margin.left + margin.right)
259               .attr("height", height + margin.top + margin.bottom)
260             .append("g")
261               .attr("transform",
262                     "translate(" + margin.left + "," + margin.top + ")");
263
264               svg.append("text")
265                 .attr("x", width / 2)
266                 .attr("y", -8)
267                 .attr("text-anchor", "middle")
268                 .style("font-size", "16px")
```

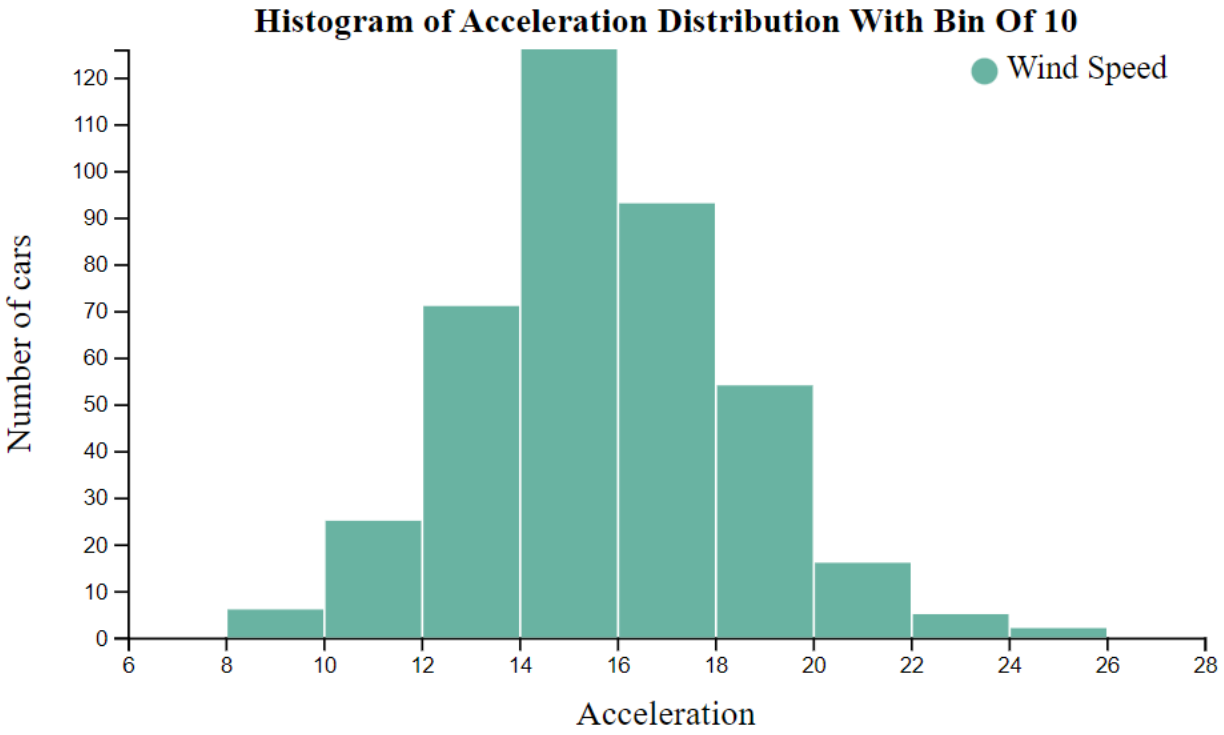
```
            .style("font-weight", "bold")
            .text("Histogram of Acceleration Distribution With Bin Of 10");

          svg.append("circle").attr("cx",width - 100).attr("cy",10).attr("r", 6).style("fill", "#69b3a2")
          svg.append("text").attr("x", width - 90).attr("y", 10).text("Wind Speed").style("font-size", "15px")
              .attr("alignment-baseline","middle")

          var x = d3.scaleLinear()
            .domain([6,28])
            .range([0, width]);
          svg.append("g")
            .attr("transform", "translate(0," + height + ")")
            .call(d3.axisBottom(x));
          svg.append("text")
            .attr("transform",
                  "translate(" + (width/2) + " ," +
                                 (height + margin.top + 20) + ")")
            .style("text-anchor", "middle")
            .text("Acceleration");

          var histogram = d3.histogram()
            .value(function(d) { return d.acceleration; })
            .domain(x.domain())
            .thresholds(x.ticks(10));

          var bins = histogram(data);
          console.log(bins)
          var y = d3.scaleLinear()
            .domain([0, d3.max(bins, function(d) { return d.length; })])
            .range([height, 0])
          svg.append("g")
            .text("Number of days")
            .call(d3.axisLeft(y));
          svg.append("text")
            .attr("transform", "rotate(-90)")
            .attr("y", 0 - margin.left)
            .attr("x",0 - (height / 2))
            .attr("dy", "1em")
            .style("text-anchor", "middle")
            .text("Number of cars");

          svg.selectAll("rect")
            .data(bins)
            .enter()
            .append("rect")
              .attr("x", 1)
              .attr("transform", function(d) { return "translate(" + x(d.x0) + "," + y(d.length) + ")"; })
              .attr("width", function(d) { return x(d.x1) - x(d.x0) -1 ; })
              .attr("height", function(d) { return height - y(d.length); })
              .style("fill", "#69b3a2")

        displayTable(countByAccelerationCount, null);

      })
```



Model Year	Count
70	29
71	28
72	28
73	40
74	27
75	30
76	34
77	28
78	36
79	29
80	29
81	29
82	31



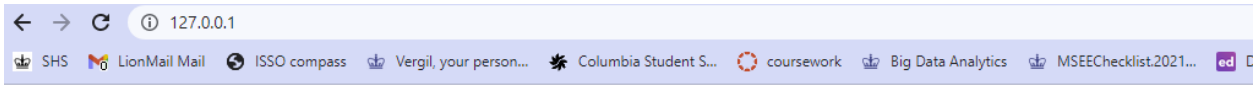
Model Year	Number of Cylinders
70	196
71	156
72	163
73	255
74	142
75	168
76	192
77	153
78	193
79	169
80	120
81	134
82	130

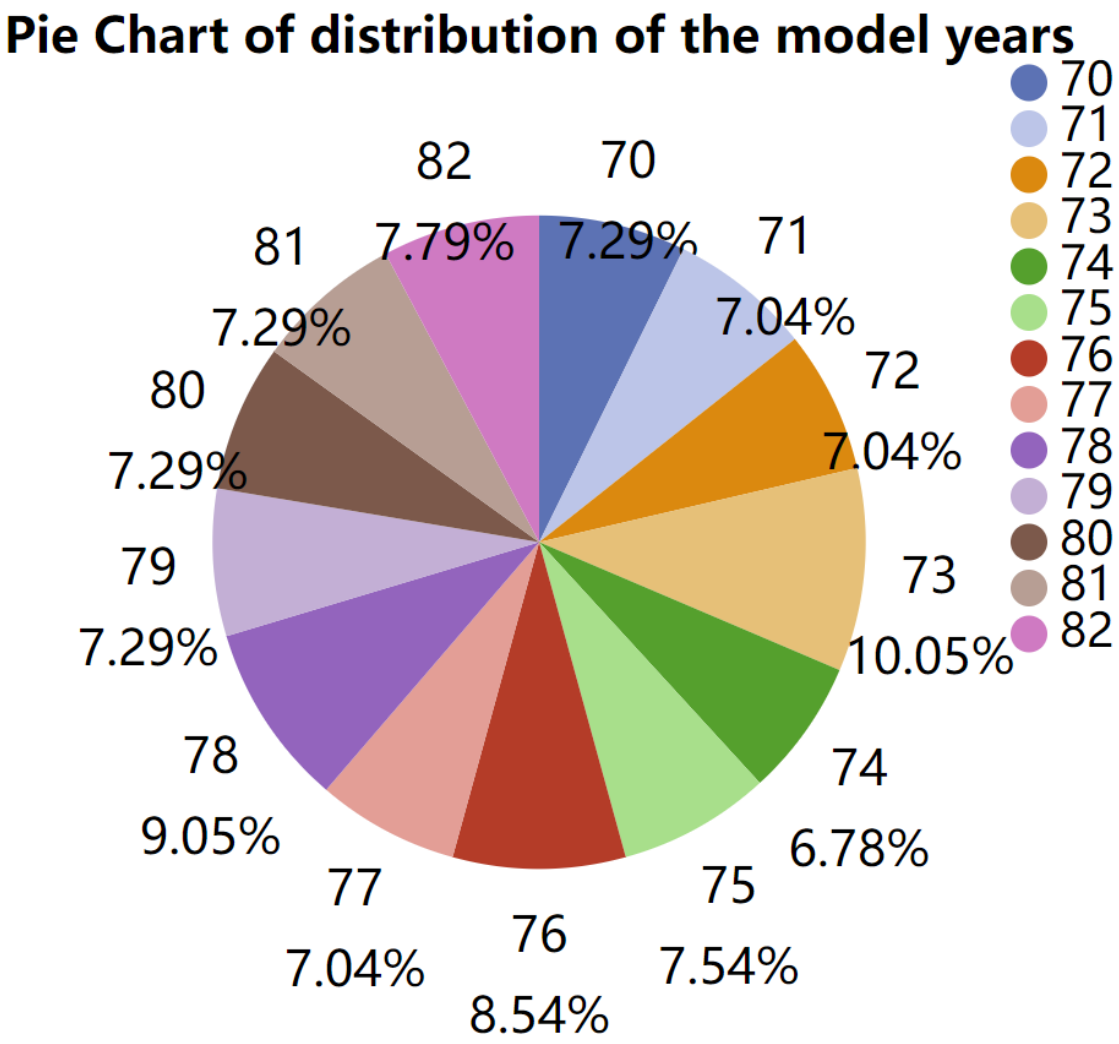


Acceleration	Count
12	10
11.5	7
11	7
10.5	1
10	4
9	1
8.5	2
8	1
9.5	2
15	14
15.5	21
16	16
14.5	23
20.5	3
17.5	4
12.5	8
14	16
13.5	15

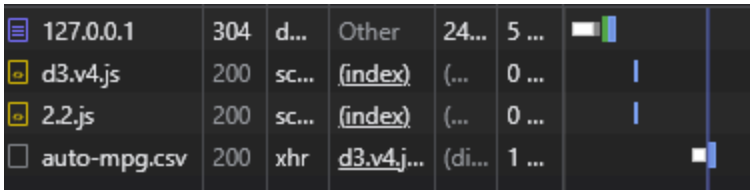
18.5	5
19	12
13	12
19.5	6
18	8
17	14
23.5	1
16.5	13
21	5
16.9	4
14.9	7
17.7	3
15.3	3
13.9	2
12.8	3
15.4	4
17.6	4
22.2	2
22.1	1
14.2	3
17.4	2
16.2	4
17.8	2
12.2	2
16.4	9
13.6	2
15.7	4
13.2	6
21.9	1
16.7	3
12.1	1
14.8	3
18.6	4
16.8	2
13.7	2
11.1	1
11.4	2
18.2	5
15.8	7
15.9	2
14.1	1
21.5	1
14.4	5
19.4	3
19.2	3
17.2	2
18.7	2
15.1	2
13.4	2
11.2	1
14.7	5
16.6	3
17.3	5
15.2	3
14.3	2
20.1	2
24.8	1
11.3	1
12.9	2
18.8	1
18.1	1
17.9	1
21.7	1
23.7	1
19.9	1
21.8	1
13.8	2
12.6	2
16.1	1
20.7	1
18.3	1
20.4	1
19.6	2
17.1	1
15.6	1
24.6	1
11.6	1

Problem 3





Model Year	Count
70	29
71	28



Part III

```
<!DOCTYPE html>
<meta charset="utf-8">
<html>

<head>
    <link rel="stylesheet" href="3.1&3.2.css">
    <script src="http://d3js.org/d3.v3.js"></script>
    <script src="3.1&3.2.js"></script>
</head>

<body>

    <script>

        var width = 800,
            height = 450;

        var color = d3.scale.category20();

        var force = d3.layout.force()
            .charge(-120)
            .linkDistance(30)
            .size([width, height]);

        var svg = d3.select("body").append("svg")
            .attr("width", width)
            .attr("height", height);

            svg.append("text")
              .attr("x", width / 2)
              .attr("y", 20)
              .attr("text-anchor", "middle")
              .style("font-size", "16px")
              .style("font-weight", "bold")
              .text("Network Graph For Character In Les Miserables");

        var counter = 0
        var c = ["rgb(31, 119, 180)","rgb(174, 199, 232)","rgb(152, 223, 138)","rgb(255, 127, 14)","rgb(44, 160, 44)","rgb(255, 187, 120)","rgb
            (214, 39, 40)","rgb(148, 103, 189)","rgb(140, 86, 75)","rgb(255, 152, 150)","rgb(197, 176, 213)"]
        var n = ["Group 1","Group 2","Group 3","Group 4","Group 5","Group 6","Group 7","Group 8","Group 9","Group 10","Group 11"]
        d3.range(11).forEach(function(d) {
          svg.append("circle").attr("cx",100).attr("cy",100 + counter * 15).attr("r", 6).style("fill", c[counter])
          svg.append("text").attr("x", 110).attr("y", 100 + counter * 15).text(n[counter]).style("font-size", "15px").attr("alignment-baseline"
              ,"middle")
          counter = counter + 1
        });

        var fisheye = d3.fisheye.circular()
          .radius(50)
          .distortion(5);

        d3.csv("https://gist.githubusercontent.com/timelyportfolio/5049980/raw/66a239b4aa325c05c7a19bd96bf093632591e809/les_mis.csv", function
            (data) {

            graph = { "nodes": [], "links": [] };

            data.forEach(function (d) {
```

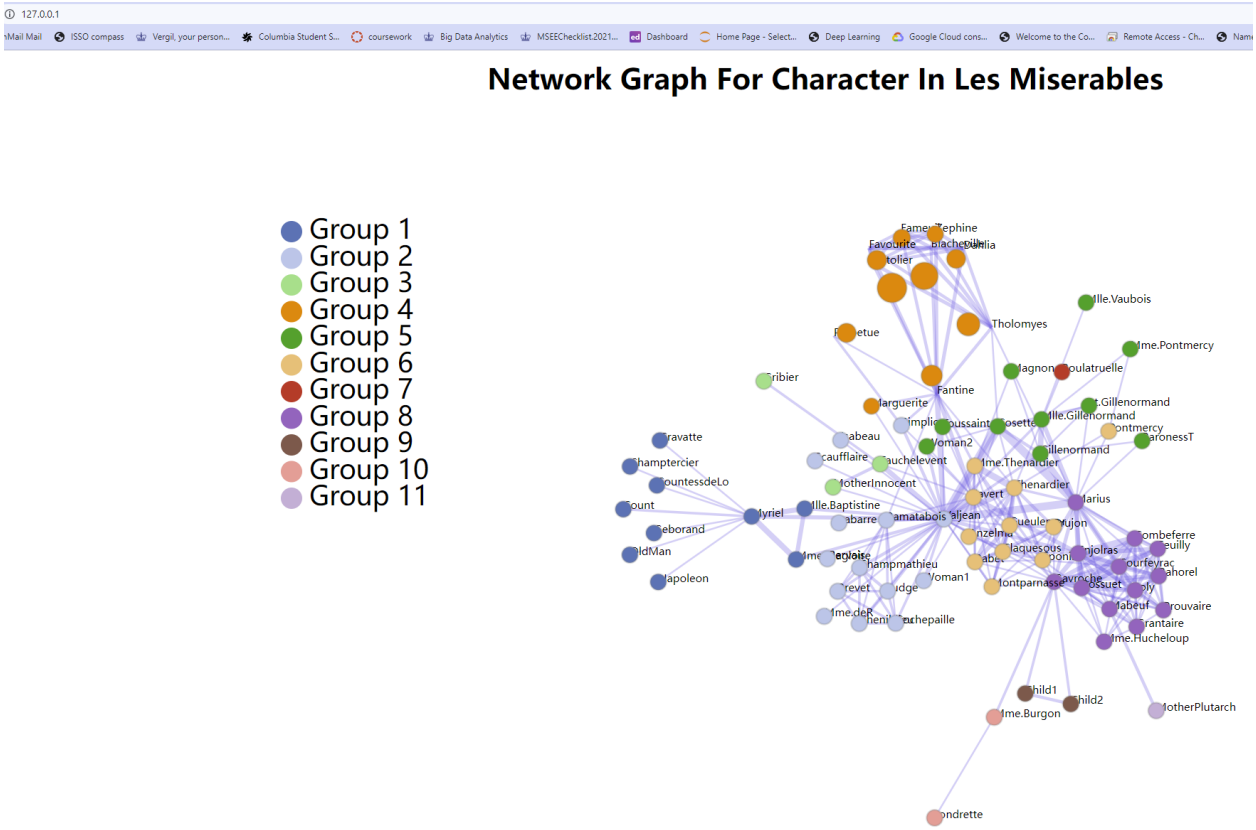
```
            graph.nodes.push({ "character": d.source, "group": +d.groupsource });
            graph.nodes.push({ "character": d.target, "group": +d.grouptarget });

            graph.links.push({ "source": d.source, "target": d.target, "value": +d.value });
        });

        var nodesMap = d3.nest()
                         .key(function (d) { return d.character; })
                         .rollup(function (d) { return { "character": d[0].character, "group": d[0].group }; })
                         .map(graph.nodes);

        graph.nodes = d3.keys(d3.nest()
                         .key(function (d) { return d.character; })
                         .map(graph.nodes));

        graph.links.forEach(function (d, i) {
            graph.links[i].source = graph.nodes.indexOf(graph.links[i].source);
            graph.links[i].target = graph.nodes.indexOf(graph.links[i].target);
        });

        graph.nodes.forEach(function (d,i) { graph.nodes[i]={ "character": nodesMap[d].character, "group": nodesMap[d].group }; })

        force
          .nodes(graph.nodes)
          .links(graph.links)
          .start();

        var link = svg.selectAll(".link")
            .data(graph.links)
            .enter()
            .append("line")
                .attr("class", "link")
                .style("stroke-width", function (d) { return Math.sqrt(d.value); });

        var node = svg.selectAll(".nodes")
            .data(graph.nodes)
            .enter()
            .append("g")
                .attr("class", "nodes")
                .call(force.drag);

        node.append("text")
                .attr("dy", -10)
                .attr("dx",-13)
                .style("font-size", "6px")
                .style("text-align", "center")
                .text(d => d.character);

        node.append("circle")
                .attr("r", d=> 6)
                .attr("id",d=> "circle"+d.id)
                .style("stroke", "grey")
                .style("stroke-opacity",0.3)
                .style("fill", d => color(d.group))

        svg.on("mousemove", function() {
            fisheye.focus(d3.mouse(this));
```

```
112
113                 node.each(function(d) { d.fisheye = fisheye(d); })
114
115                 node.selectAll("circle")
116                   .attr("vx", function(d) { return d.fisheye.x; })
117                   .attr("vy", function(d) { return d.fisheye.y; })
118                   .attr("r", function(d) { return d.fisheye.z * 4.5; });
119
120                 node.selectAll("text")
121                   .attr("dx", function(d) { return d.fisheye.x - d.x; })
122                   .attr("dy", function(d) { return d.fisheye.y - d.y; });
123
124                 link
125                     .attr("x1", function(d) { return d.source.fisheye.x; })
126                     .attr("y1", function(d) { return d.source.fisheye.y; })
127                     .attr("x2", function(d) { return d.target.fisheye.x; })
128                     .attr("y2", function(d) { return d.target.fisheye.y; });
129             });
130
131             force.on("tick", function () {
132                 link
133                     .attr("x1", function (d) { return d.source.x; })
134                     .attr("y1", function (d) { return d.source.y; })
135                     .attr("x2", function (d) { return d.target.x; })
136                     .attr("y2", function (d) { return d.target.y; });
137
138                 node.attr("transform", d => `translate(${d.x},${d.y})`);
139             });
140         });
141
142     </script>
143
144 <p>
145     I see there are multiple clusters or groups that each of them are represented in different colors. People in the same group are generally
        more connected, meaning there are more edges amoung members in the same gorup. Usually, there is one or a few people in one gourp that are
        resonposible for connect with other groups.
146 </p>
147 </body>
148
149 </html>
```

```
(function() {
  d3.fisheye = {
    scale: function(scaleType) {
      return d3_fisheye_scale(scaleType(), 3, 0);
    },
    circular: function() {
      var radius = 200,
          distortion = 2,
          k0,
          k1,
          focus = [0, 0];

      function fisheye(d) {
        var dx = d.x - focus[0],
            dy = d.y - focus[1],
            dd = Math.sqrt(dx * dx + dy * dy);
        if (!dd || dd >= radius) return {x: d.x, y: d.y, z: dd >= radius ? 1 : 10};
        var k = k0 * (1 - Math.exp(-dd * k1)) / dd * .75 + .25;
        return {x: focus[0] + dx * k, y: focus[1] + dy * k, z: Math.min(k, 10)};
      }

      function rescale() {
        k0 = Math.exp(distortion);
        k0 = k0 / (k0 - 1) * radius;
        k1 = distortion / radius;
        return fisheye;
      }

      fisheye.radius = function(_) {
        if (!arguments.length) return radius;
        radius = +_;
        return rescale();
      };

      fisheye.distortion = function(_) {
        if (!arguments.length) return distortion;
        distortion = +_;
        return rescale();
      };

      fisheye.focus = function(_) {
        if (!arguments.length) return focus;
        focus = _;
        return fisheye;
```

```
45        };
46
47        return rescale();
48      }
49    };
50
51    function d3_fisheye_scale(scale, d, a) {
52
53      function fisheye(_) {
54        var x = scale(_),
55            left = x < a,
56            range = d3.extent(scale.range()),
57            min = range[0],
58            max = range[1],
59            m = left ? a - min : max - a;
60        if (m == 0) m = max - min;
61        return (left ? -1 : 1) * m * (d + 1) / (d + (m / Math.abs(x - a))) + a;
62      }
63
64      fisheye.distortion = function(_) {
65        if (!arguments.length) return d;
66        d = +_;
67        return fisheye;
68      };
69
70      fisheye.focus = function(_) {
71        if (!arguments.length) return a;
72        a = +_;
73        return fisheye;
74      };
75
76      fisheye.copy = function() {
77        return d3_fisheye_scale(scale.copy(), d, a);
78      };
79
80      fisheye.nice = scale.nice;
81      fisheye.ticks = scale.ticks;
82      fisheye.tickFormat = scale.tickFormat;
83      return d3.rebind(fisheye, scale, "domain", "range");
84    }
85  })();
```

```
body{
  font: Arial 12px;
  text-align: center;
}

p {
  font-family: Arial, Helvetica, Sans Serif;
  font-size: 20px;
  color: LightSkyBlue;
  text-indent: 3%;
}

.node {
  stroke: black;
  stroke-width: 20px;
}

.link {
  stroke: royalblue;
  stroke-opacity: 0.3;
}
```



I see there are multiple clusters or groups that each of them are represented in different colors. People in the same group are generally more connected, meaning there are more edges amoung members in the same gorup. Usually, there is one or a few people in one gourp that are resonposible for connect with other groups.

Name	St...	Ty..	Initiator	Size	Ti...	Waterfall
127.0.0.1	304	d...	Other	24...	2 ...	
3.1&3.2.js	200	sc...	(index)	(...	0 ...	
3.1&3.2.css	200	st...	(index)	(di...	2 ...	
d3.v3.js	307	sc...	(index)	(di...	2 ...	
d3.v3.js	200	sc...	d3.v3.js	(di...	4 ...	
les_mis.csv	200	xhr	d3.v3.j...	(di...	1 ...	