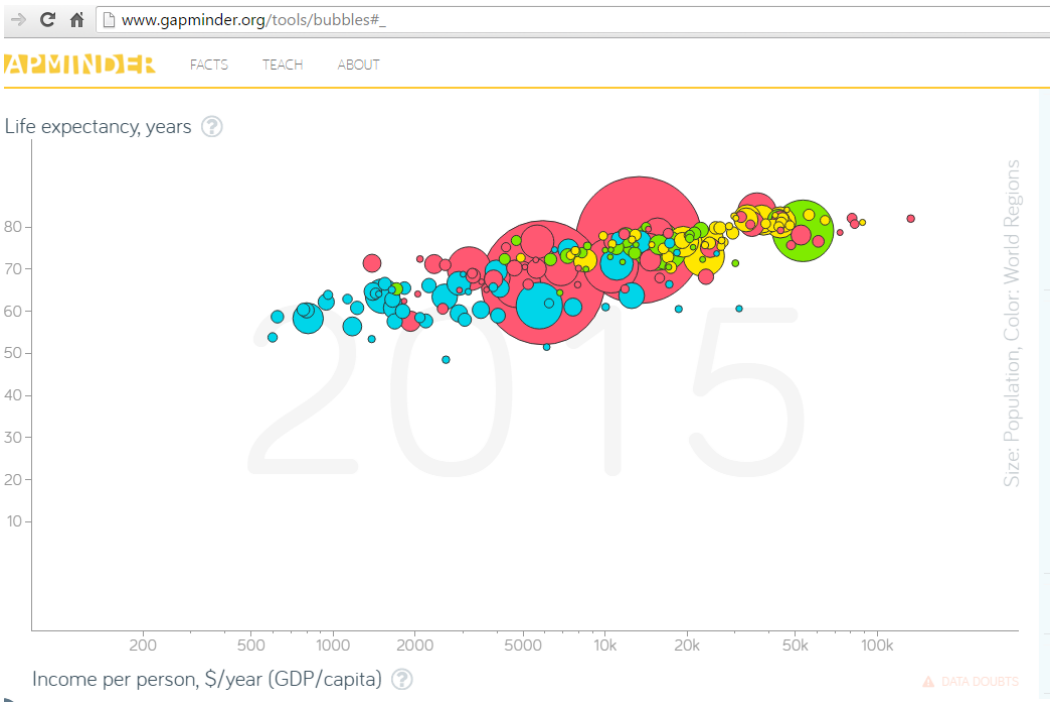
Hanjiao Qiu
CSCIE63 Big Data Analytics
Assignment 09

Assignment 09 Solution

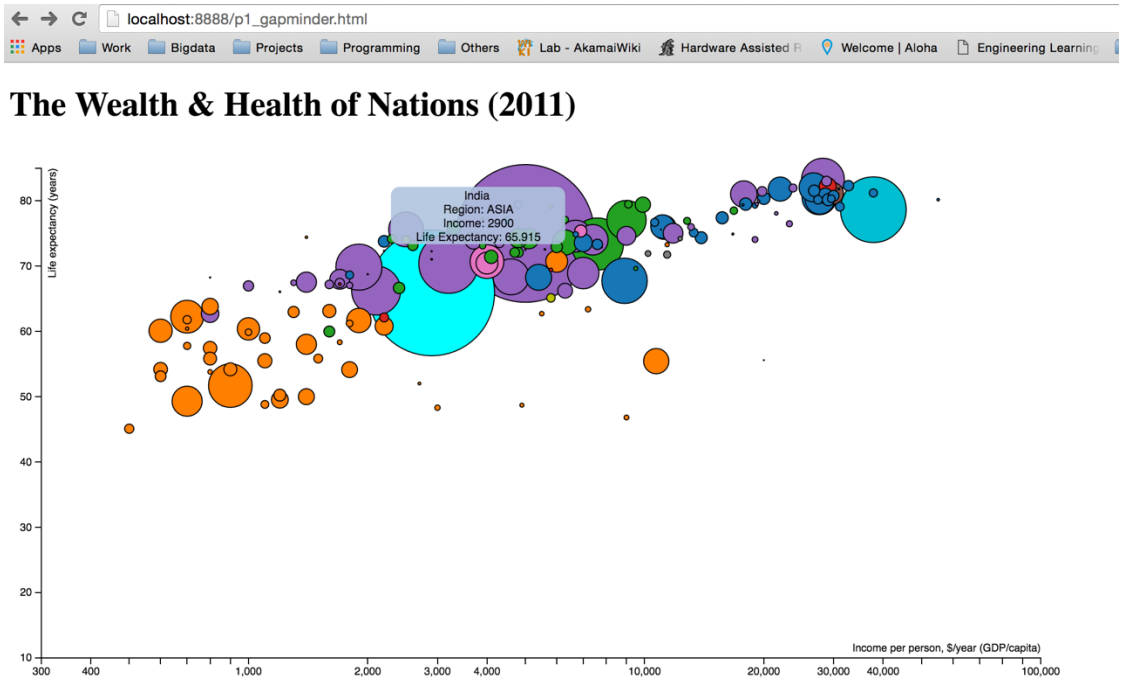
Please, describe every step of your work and present all intermediate and final results in a Word document. Please, copy past text version of all essential command and snippets of results into the Word document with explanations of the purpose of those commands. We cannot retype text that is in JPG images. Please, always submit a separate copy of the original, working scripts and/or class files you used. Sometimes we need to run your code and retyping is too costly. Please include in your MS Word document only relevant portions of the console output or output files. Sometime either console output or the result file is too long and including it into the MS Word document makes that document too hard to read. PLEASE DO NOT EMBED files into your MS Word document. For issues and comments visit the class Discussion Board. You are not obliged to use Java or Eclipse. You are welcome to use any language and any IDE of your choice.

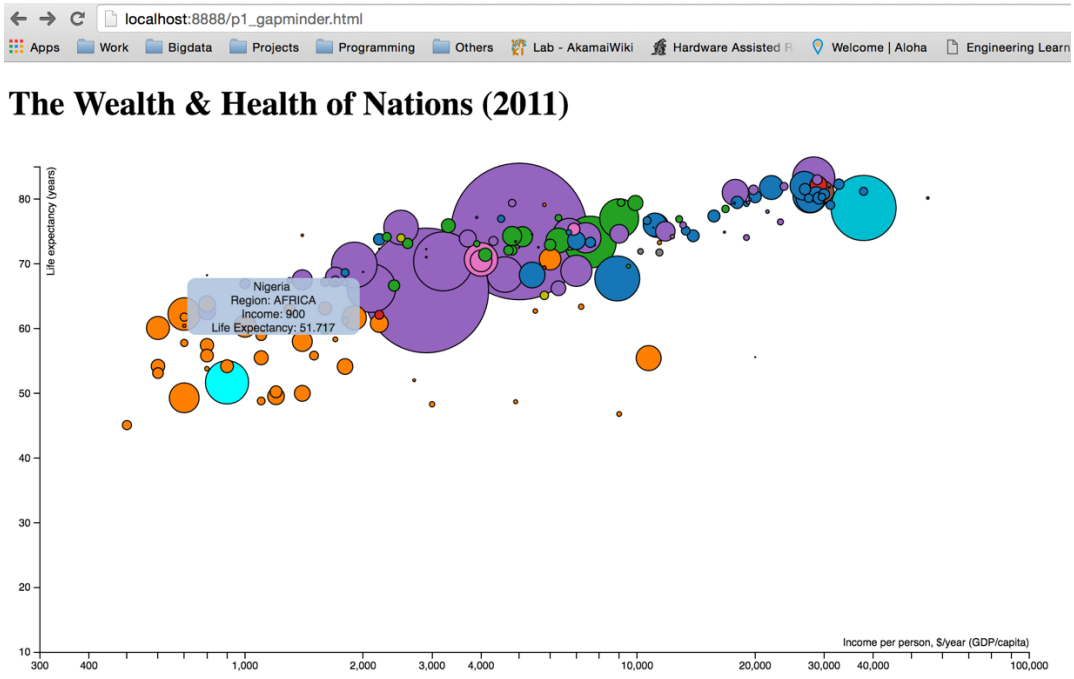
Problem 1) Public site GapMinder.org presents many excellent visualizations of data about the World. One such presentation (http://www.gapminder.org/tools/bubbles#_) displays average life expectancy in year as a function of average income per person in countries of the world. Countries are represented as circles in different colors depending on their continent, e.g. countries in Europe are yellow, countries in Asia, red, etc. Every country is presented by a circle of area proportional to its population. Radius of the circle is therefore proportional to the square root of population. As the cursor hovers over each country, its name appears over its circle. Data presented in this graph can be found in various Excel files provided by the same site (http://www.gapminder.org/data/). We extracted some files which we believe contain data used in the graph bellow:



Please recreate above graph using D3 or any similar technology of your choice**.** You do not need all data present in provided Excel files. Select most recent data for every country. If you know what you are doing, keep your data in files on your OS or in a database of your choice. Otherwise, copy relevant data directly into your "HTML/JavaScript" code. Please note that the horizontal axis is logarithmic.

Solution:





1. Get the data from “countries of the world.xls”. Select the column “country”, “region”, “population” and “GDP” and only reserve them in the form. Add another column “lifeExpectancy” from “indicator life_expectancy_at_birth_1800-2050 .xlsx”. This is all the data we need.

2. Create the html file and the file directory.

To start the HTTP server, run the command line in the same directory as the html file.

```
hqiu@bos-mp9cx>> python -m SimpleHTTPServer 8888
Serving HTTP on 0.0.0.0 port 8888 ...
127.0.0.1 - - [07/Apr/2016 16:29:05] "GET /p1_gapminder.html HTTP/1.1" 200 -
```

3. Source code.

Define the CSS style.

```
<!DOCTYPE html>
<html lang="en">
<title>The Wealth & Health of Nations (2011)</title>
<head>
    <meta charset="utf-8">
    <script type="text/javascript" src="../d3/d3.js"></script>
```

```
    <style type="text/css">
        #chart {
            margin-left: -40px;
            height: 506px;
        }

        text {
            font: 10px sans-serif;
        }

        .dot {
            stroke: #000;
        }

        .axis path, .axis line {
            fill: none;
            stroke: #000;
            shape-rendering: crispEdges;
        }

        div.tooltip {
            position: absolute;
            text-align: center;
            width: 160px;
            height: 50px;
            padding: 2px;
            font: 11px sans-serif;
            background: lightsteelblue;
            border: 0px;
            border-radius: 8px;
            pointer-events: none;
        }

        .legend {
            padding: 5px;
            font: 10px sans-serif;
            box-shadow: 2px 2px 1px #888;
        }
    </style>
</head>
```

Write the main javascript.

Define the methods to get each column from the dataset.

```
<h1>The Wealth & Health of Nations (2011)</h1>

<p id="chart"></p>

<body>
<script type="text/javascript">
    // Various accessors that specify the five dimensions of data to visualize.
    function x(d) { return d.income; }
    function y(d) { return d.lifeExpectancy; }
    function radius(d) { return d.population; }
    function color(d) { return d.region; }
    function key(d) { return d.country; }
```

Define the x & y axes and the radius scale. The horizontal axis is logarithmic. Radius of the circle is proportional to the square root of population.

```
// Chart dimensions.
var margin = {top: 19.5, right: 19.5, bottom: 19.5, left: 70},
        width = 960 - margin.right,
        height = 500 - margin.top - margin.bottom;

// Various scales. These domains make assumptions of data, naturally.
var xScale = d3.scale.log().domain([300, 1e5]).range([0, width]),
        yScale = d3.scale.linear().domain([10, 85]).range([height, 0]),
        radiusScale = d3.scale.sqrt().domain([0, 5e8]).range([0, 40]),
        colorScale = d3.scale.category10();

// The x & y axes.
var xAxis = d3.svg.axis().orient("bottom").scale(xScale).ticks(12,
d3.format(",d")),
        yAxis = d3.svg.axis().scale(yScale).orient("left");

// Create the SVG container and set the origin.
var svg = d3.select("#chart").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 + ")");

// Add the x-axis.
svg.append("g")
        .attr("class", "x axis")
        .attr("transform", "translate(0," + height + ")")
        .call(xAxis);

// Add the y-axis.
svg.append("g")
        .attr("class", "y axis")
        .call(yAxis);

// Add an x-axis label.
svg.append("text")
        .attr("class", "x label")
        .attr("text-anchor", "end")
        .attr("x", width)
        .attr("y", height - 6)
        .text("Income per person, $/year (GDP/capita)");

// Add a y-axis label.
svg.append("text")
        .attr("class", "y label")
        .attr("text-anchor", "end")
        .attr("y", 6)
        .attr("dy", ".75em")
        .attr("transform", "rotate(-90)")
        .text("Life expectancy (years)");
```

Define the tooltip.

```
// Define the div for the tooltip
var div = d3.select("body").append("div")
        .attr("class", "tooltip")
        .style("opacity", 0);
```

```
// Load the data.
d3.csv("gapminder.csv", function(nations) {

    // Add a dot per nation and set the colors.
    var dot = svg.append("g")
            .attr("class", "dots")
            .selectAll("circle")
            .data(interpolateData())
            .enter()
            .append("g");
```

For each dot, draw the circle and fill in the color. Add the “mouseover” and “mouseout” actions. Pop up the label for the country when “mouseover”.

```
dot.append("circle")
        .attr("class", "dot")
        .style("fill", function (d) {
            return colorScale(color(d));
        }).call(position).sort(order)
    .on("mouseover", function(d) {
        // d3.select(this).style("opacity", 0.5);
        d3.select(this).style("fill", "#00ffff");

        div.transition()
                .duration(200)
                .style("left", d3.select(this).attr("cx") + "px")
                .style("top", d3.select(this).attr("cy") + "px")
                .style("opacity", .9);
        div.html(d.country + "<br/>" + "Region: " + d.region + "<br/>" +
                "Income: " + d.income + "<br/>" + "Life Expectancy: " +
d.lifeExpectancy )
    })
    .on("mouseout", function(d) {
        // d3.select(this).style("opacity", 1);
        d3.select(this).style("fill", colorScale(color(d)));

        div.transition()
                .duration(500)
                .style("opacity", 0);
    });
```

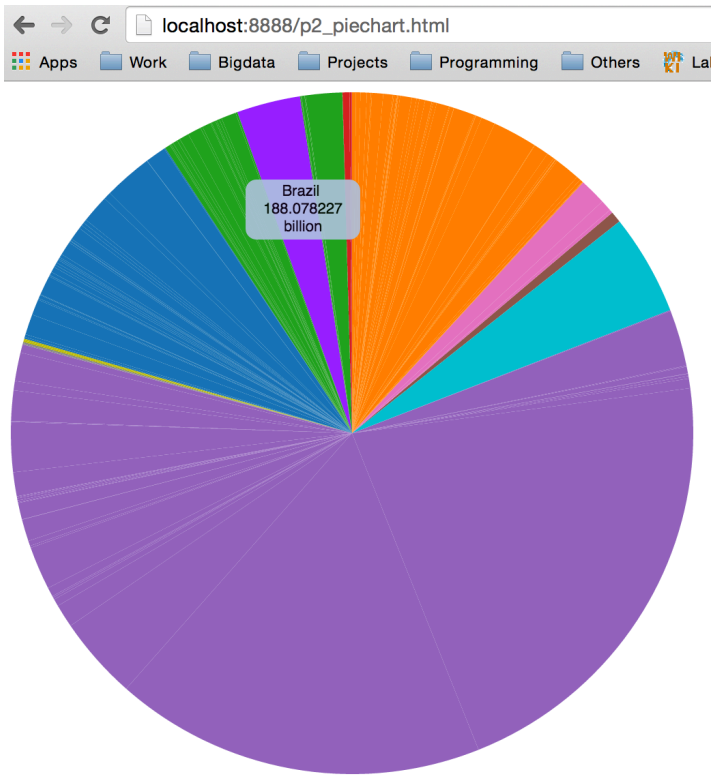
```
    function position(dot) {
        dot.attr("cx", function (d) {
                    return xScale(x(d));
                })
                .attr("cy", function (d) {
                    return yScale(y(d));
                })
                .attr("r", function (d) {
                    return radiusScale(radius(d));
                });
    }

    // Defines a sort order so that the smallest dots are drawn on top.
    function order(a, b) {
        return radius(b) - radius(a);
    }

    // Interpolates the dataset for the given (fractional) year.
    function interpolateData() {
```

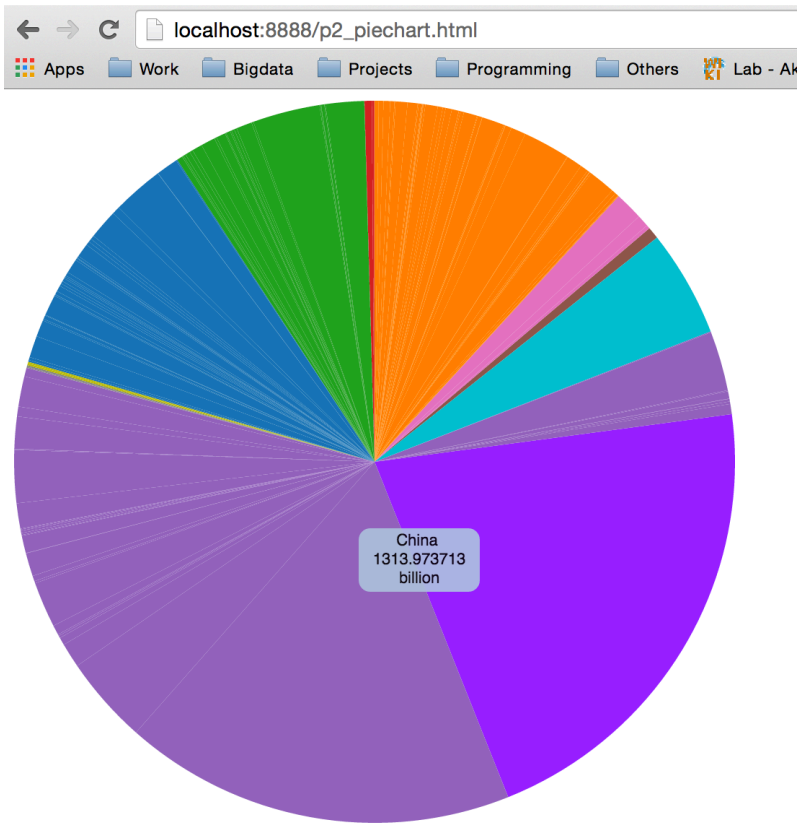
```
        return nations.map(function (d) {
            console.log(d);
            return {
                country: d.country,
                region: d.region,
                population: d.population,
                income: d.income,
                lifeExpectancy: d.lifeExpectancy
            };
        });
    }

});
</script>
</body>
```

Problem 2) Display the graph of population per country as a pie chart. Color countries by the continents and group all countries by the continents. Leave no separation between different countries, however, when you hover over a country change it color to light purple and display its name and population in millions.

For every problem, please provide a working copy of your HTML file including the JavScript (D3) portion.

Solution:





1. Define the CSS style.

```
<!DOCTYPE html>
<head>
    <meta http-equiv="Content-type" content="text/html; charset=utf-8">
    <title>Population Pie Chart</title>
    <script type="text/javascript" src="../d3/d3.js"></script>
    <script src="http://d3js.org/d3.v3.min.js"></script>
    <style type="text/css">
        .arc text {
            font: 10px sans-serif;
          text-anchor: middle;
        }

        div.tooltip {
            position: absolute;
            text-align: center;
            width: 80px;
            height: 40px;
            padding: 2px;
            font: 11px sans-serif;
            background: lightsteelblue;
            border: 0px;
            border-radius: 8px;
            pointer-events: none;
        }
    </style>
</head>
<body>
```

2. Define the pie chart. The size of the slice represents the population of the country.

```
<script type="text/javascript">

    var width = 960,
            height = 500,
            radius = Math.min(width, height) / 2;

    var color = d3.scale.category10();

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

    var labelArc = d3.svg.arc()
            .outerRadius(radius - 40)
            .innerRadius(radius - 40);

    var pie = d3.layout.pie()
            .sort(order)
            .value(function(d) { return d.population; });

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

    function order(a, b) {
        return d3.ascending(a.region, b.region);
    }
```

3. Read data from the csv file and draw the pie chart. Add the “mouseover” and “mouseout” actions using tooltip.

```
// Define the div for the tooltip
var div = d3.select("body").append("div")
        .attr("class", "tooltip")
        .style("opacity", 0);
d3.csv("gapminder.csv", type, function(error, data) {
    if (error) throw error;

    var arcs = svg.selectAll("arc")
            .data(pie(data))
            .enter()
            .append("g")
            .attr("class", "arc")
            .attr("transform", "translate(" + radius + ", " + radius + ")");

    arcs.append("path")
         .style("fill", function(d) {
             return color(d.data.region);
         })
         .attr("d", arc)
         .on("mouseover", function(d) {
             // d3.select(this).style("opacity", .5);
             d3.select(this).style("fill", "#9933ff")
             div.transition()
                     .duration(200)
                     .style("left", d3.mouse(this)[0] + 200 + "px")
```

```
                    .style("top", d3.mouse(this)[1] + 200 + "px")
                    .style("opacity", .9);
                div.html(d.data.country + "<br/>"  + d.data.population/1000000 + "
billion")
            })
            .on("mouseout", function(d) {
                // d3.select(this).style("opacity", 1);
                d3.select(this).style("fill", color(d.data.region))
                div.transition()
                    .duration(500)
                    .style("opacity", 0);
            });
});
```

```
    function type(d) {
        d.population = +d.population;
        return d;
    }

</script>
```