F21DV Data Visualisation and Analytics – Lab 2
Josh Yang jy84 091514042

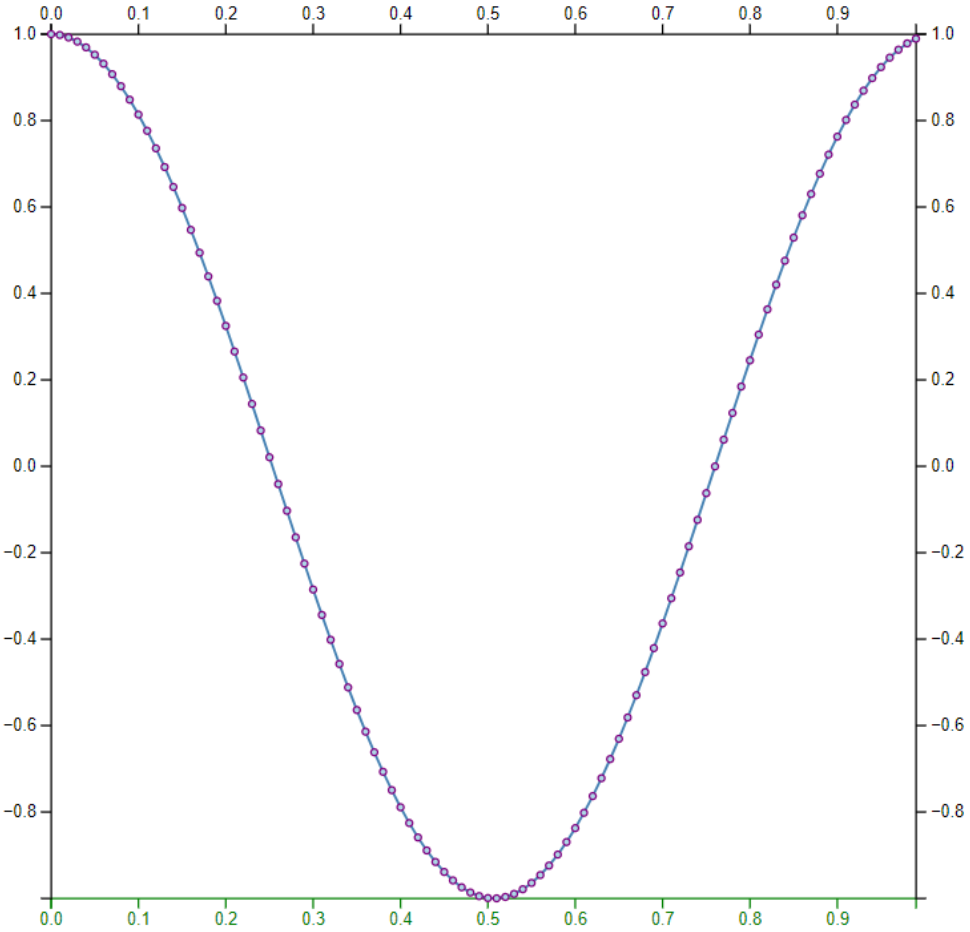
**Demonstrated on 25/02/2022 to Benjamin Kenwright.**

Introduction

The following report will provide screenshots of the answers and, when relevant, context to the code developed. Every exercise is self-contained and can be run on a browser. When requested, answers can be found in the console log, and I will highlight when this occurs. To test my code is working I used the Visual Studio extension Live Server.

I have committed all my answers to a public GitHub repository which can be accessed here: https://github.com/JoshYang1/F21DV-Data-Visualisation-and-Analytics

Exercise 1



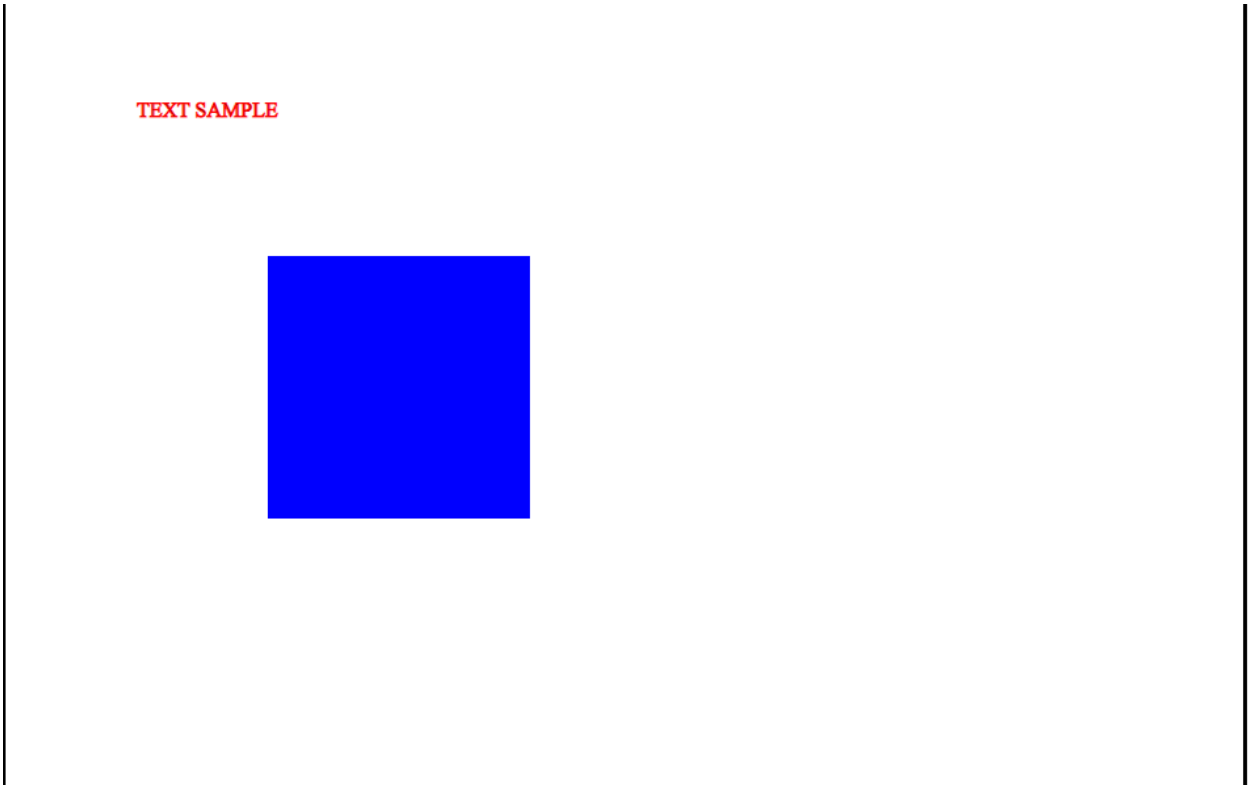

```
        // Add circles to each cosine data point
        svg.append("g")
            .selectAll("dot")
            .data(data)
            .enter()
```

```
            .append("circle")
            .attr("r", 2)
            .attr("cx", function(d){return x(d.x)})
            .attr("cy", function(d){return y(d.y)})
```

```
circle:hover {
    -webkit-animation-name: pulsar;
    -webkit-animation-duration: 0.5s;
    -webkit-animation-iteration-count: infinite;
    -webkit-animation-direction: alternate;
    animation-name: pulsar;
    animation-duration: 0.5s;
    animation-iteration-count: infinite;
    animation-direction: alternate;
    -webkit-transform-origin: 50% 50%;
    transform-origin: 50% 50%;
}
```

CSS will define the pulse effect for all circles.

Exercise 2




```
    text {
```

```
        stroke: red;
        visibility: hidden;
    }

    #container:hover text {
        visibility: visible;
    }
```

Hovering over the svg container will display the extra information.

Exercise 3

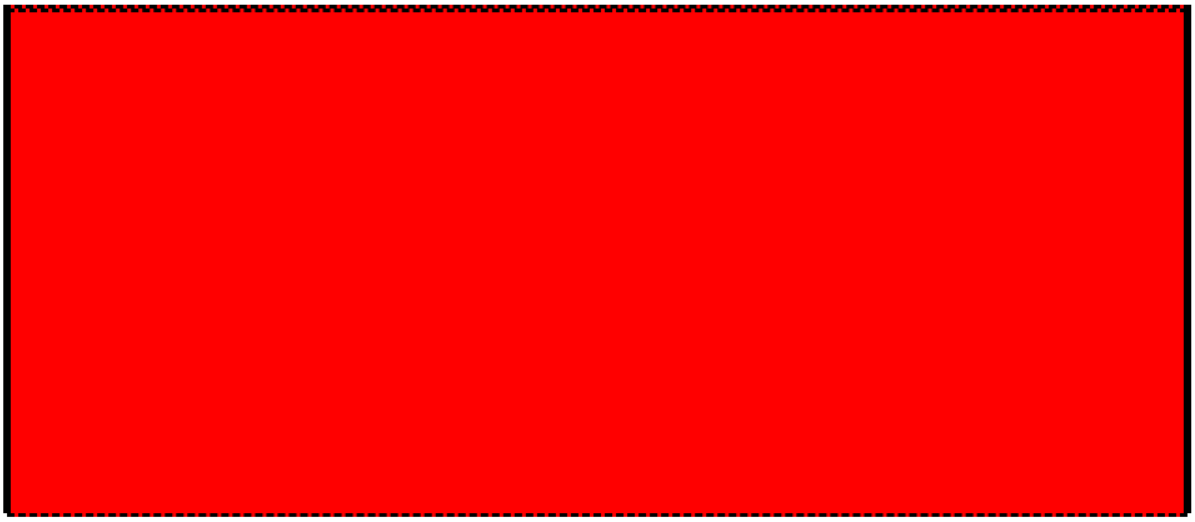
```
    // Mousing over the div element will change the style
    // https://developer.mozilla.org/en-US/docs/Web/CSS/border-style
    d3.selectAll("div")
        .on("mouseover", function(event){
            d3.select(this)
                .style("background-color", "orange")
                .style("height", "200px")
                .style("border-style", "dotted");
        })
        .on("mouseout", function(){
            d3.select(this)
                .style("background-color", "red")
                .style("height", "400px")
                .style("border-style", "dashed solid");
        })
```

Initialised:

F21DV Data Visualisation and Analytics – Lab 2
Josh Yang jy84 091514042

Mouse In:

Mouse Out:

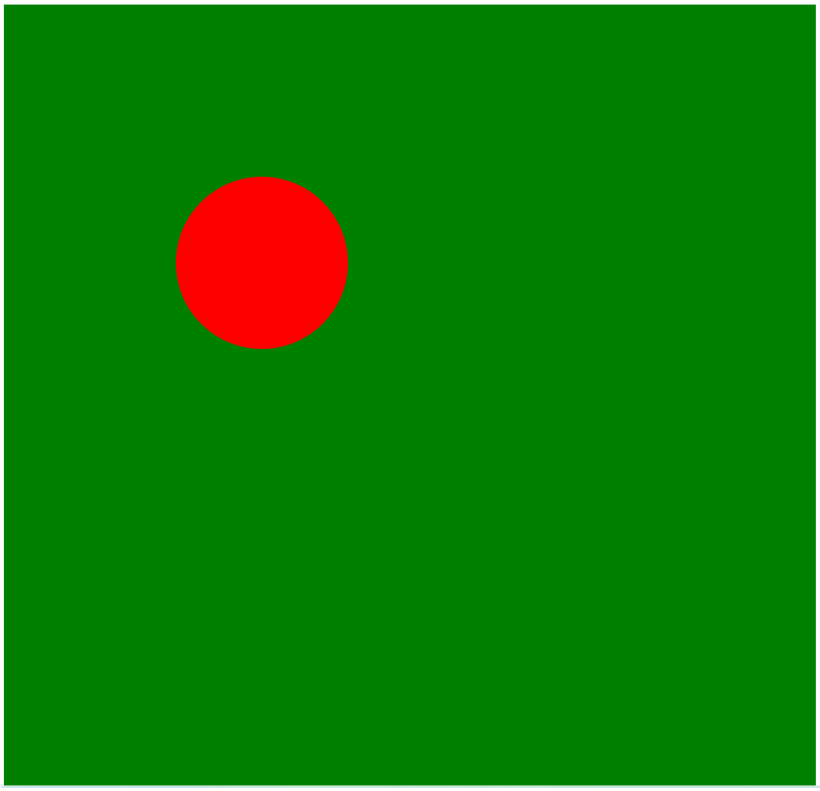


The div element's border and colour style changes when mousing in and out.

F21DV Data Visualisation and Analytics – Lab 2
Josh Yang jy84 091514042

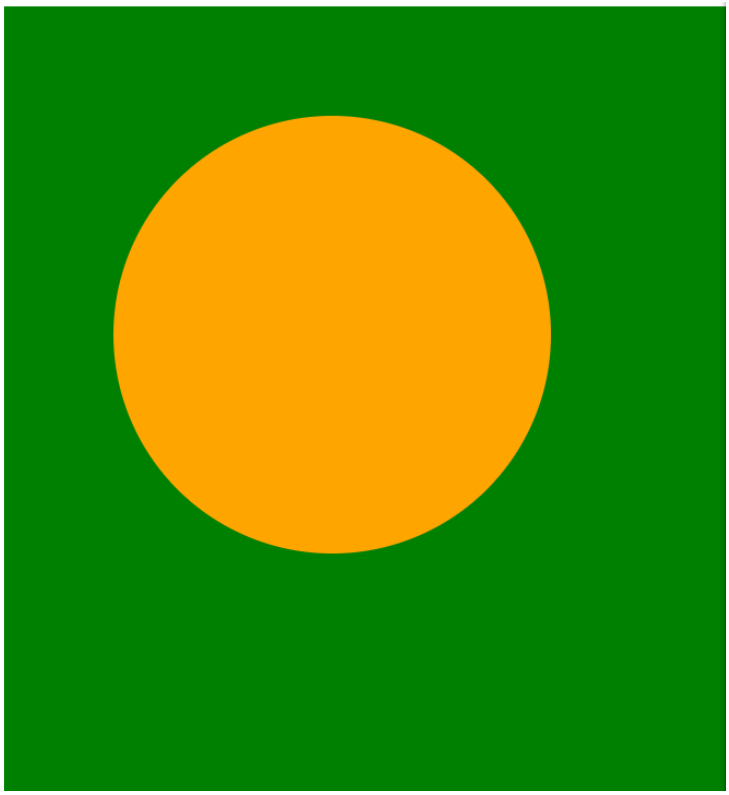
Exercise 4

Initialised:

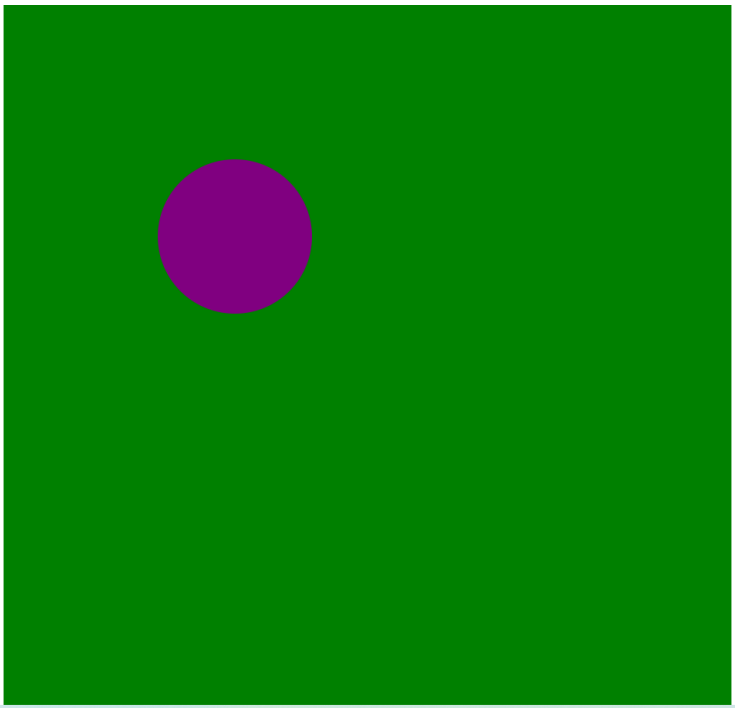


F21DV Data Visualisation and Analytics – Lab 2
Josh Yang jy84 091514042

Mouse In:



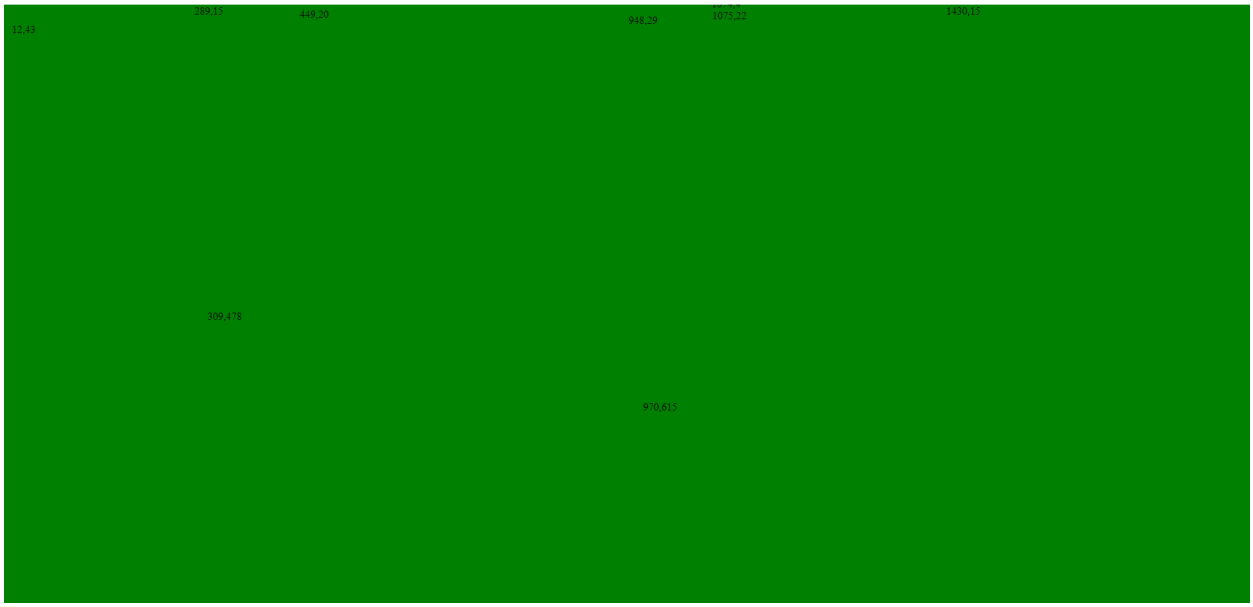
Mouse Out:



```
    // Appending a circle to the svg that changes size and colour when mouse
travels over
    svg.append("circle")
        .attr("cx", 300)
        .attr("cy", 300)
        .attr("r", 100)
        .attr("fill", "red")
        .on("mouseover", function(event){
            d3.select(this)
                .style("fill", "orange")
                .attr("r", 200)
        })
        .on("mouseout", function(){
            d3.select(this)
                .style("fill", "purple")
                .attr("r", 100)
        })
```

Radius and colour of the circle element in the svg container changes on the activity of the mouse.

Exercise 5




```
    // Setting the the dimensions of the svg to fit the container
    svg.style("height", ySize)
                .style("width", xSize)
                .style('background-color', 'green')
                .on("mouseover", handleMouseOver);
```

```
// Create Event Handler for mouse over
function handleMouseOver(d, i) {

    // Specify where to put label of text
    svg.append("text")
        .attr("x", d.x)
        .attr("y", d.y)
        .html(function() {
            return [d.x, d.y];  // Value of the text
        });
    }
```

The coordinates of the mouse will be displayed whenever the svg has been moused over.

Exercise 6

```
// Append div element to the Page
d3.select("#container")
    .append("div")
    .style('width', '100px')
    .style('height', '100px')
    .style('background-color', 'blue')
    .transition()
    // 2 second transition
    .duration(2000)
    .style("background-color", "red")
    .transition()
    .duration(2000)
    .style("background-color","green");
```

The div element will initialise blue then after 2 seconds transition to red and then after another 2 seconds transitions to green.

Exercise 7

```
// Append div element to the Page
d3.select("#container")
    .append("div")
    .style('width', '100px')
    .style('height', '100px')
```

```
        .style('background-color', 'blue')
        .transition()
        // 2 second transition
        .duration(2000)
        .style('width', '50px')
        .style('height', '50px')
        .style("background-color", "red")
        .transition()
        .duration(2000)
        .style('width', '200px')
        .style('height', '200px')
        .style("background-color","green");
```

In addition to the colour, the size now reduces in size after 2 seconds and then grows larger after another 2.

Exercise 8

```
    // Append div element to the page with mouseover interactivity which effectss
the style
    d3.select("#container")
        .append("div")
        .style('width', '100px')
        .style('height', '100px')
        .style('background-color', 'blue')
        .on("mouseover", function(){
            d3.select(this)
                .transition()
                // 2 second transition
                .duration(2000)
                .style('width', '50px')
                .style('height', '50px')
                .style("background-color", "red");
        })
        .on("mouseout", function(){
            d3.select(this)
                .transition()
                .duration(2000)
                .style('width', '100px')
                .style('height', '100px')
                .style('background-color', 'blue');
        })
```

F21DV Data Visualisation and Analytics – Lab 2
Josh Yang jy84 091514042

When mousing over, the size and colour of the div element transitions and then mousing out returns it to the original.

Exercise 9

```
    // Function created which takes an easement style as a parameter
    // Creates a div element and appends it to the container
    function easeExamples(easement) {

        d3.select('#container')
        .append("div")
        .style('width', '100px')
        .style('height', '100px')
        .style('background-color', 'blue')
        .style('transform', 'scale(1.0)')
        .transition()
        .ease(easement )
        // 8 second transition
        .duration(8000)
        .style("background-color", "red")
        .style('transform', 'scale(0.5)');

    };

    // Example of transition ease bounce
    easeExamples(d3.easeBounce);

    // Example of transition ease elastic
    easeExamples(d3.easeElastic);

    // Example of transition ease exponential
    easeExamples(d3.easeExp);
```

Created a function to display 3 different easing types.

Exercise 10

```
    // Appending a circle to the svg that changes size and colour when mouse
travels over
    svg.append("circle")
        .attr("cx", 300)
        .attr("cy", 300)
        .attr("r", 100)
        .attr("fill", "red")
```

```
        .on("mouseover", function(event){
            d3.select(this)
                .transition()
                .ease(d3.easeBounce)
                .duration(5000)
                .style("fill", "blue")
                .attr("r", 200);
        })
        .on("mouseout", function(){
            d3.select(this)
                .transition()
                .ease(d3.easeBounce)
                .duration(5000)
                .attr("r", 100);
        })
```

The circle element will grow when the mouse hovers over and will return to it original size when the mouse moves outside.

Exercise 11

```
    // Appending a circle to the svg that changes size and colour when mouse
travels over
    svg.append("text")
        .attr("x", 200)
        .attr("y", 200)
        .html("Hello, mouseover me!")
        .style("font-size", "40px")
        .style("fill","red")
        .on("mouseover", function(event){
            d3.select(this)
                .transition()
                .ease(d3.easeBounce)
                .duration(5000)
                .style("fill", "blue")
                .style("font-size", "60px")
        })
        .on("mouseout", function(){
            d3.select(this)
                .transition()
                .ease(d3.easeBounce)
                .duration(5000)
                .style("font-size", "40px")
                .style("fill","red");
```

```
        })
```

Size and colour of the text element changes on mouse over and then mousing out.

Exercise 12

```
    // Create function to avoid duplicate code
    // Parameters include x and y position and the total delay
    function barDelay(x, y, delay) {
        svg.append("rect")
            .attr("fill", "blue")
            .attr("x", x)
            .attr("y", y)
            .attr("height", 20)
            .attr("width", 10)
            .transition()
            .ease(d3.easeLinear)
            .duration(2000)
            .delay(delay)
            .attr("height",100)
    };

    // Calling the function to create bars on the svg element
    var bar1 = barDelay(100, 20, 0);
    var bar2 = barDelay(120, 20, 2000);
    var bar3 = barDelay(140, 20, 4000);
```

Created a function which takes the x and y coordinates and the delay in milliseconds to avoid duplicate code. This will display 3 bars in the svg.

Exercise 13

```
    function barDelay(x, y, delay) {
        svg.append("rect")
            .attr("fill", "blue")
            .attr("x", x)
            .attr("y", y)
            .attr("height", 20)
            .attr("width", 10)
            // Grow transition
            .transition()
            .ease(d3.easeLinear)
            .duration(2000)
            .delay(delay)
```

```
            .attr("height",100)
            // Shrink transition
            .transition()
            .ease(d3.easeLinear)
            .duration(2000)
            .delay(delay)
            .attr("height",20)
    };
```

Added transition effect to the bars so that they will return to the original size after 2 seconds.

Exercise 14

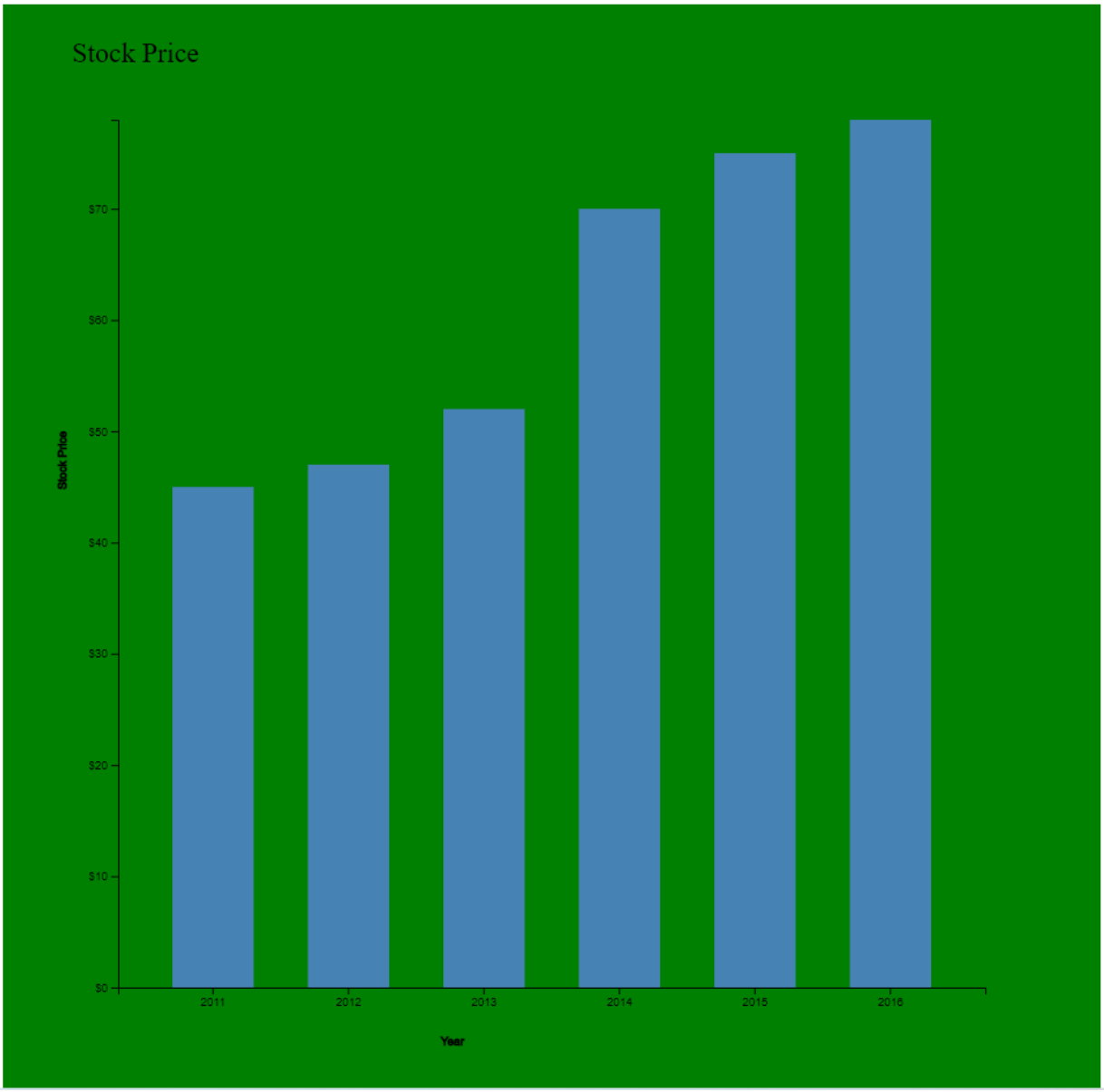
```
    // Create function to avoid duplicate code
    // Parameters include x and y position and the total delay
    function barDelay(x, y, delay) {
        svg.append("rect")
            .style("fill", "blue")
            .attr("x", x)
            .attr("y", y)
            .attr("height", 20)
            .attr("width", 10)
            // Grow transition
            .transition()
            .ease(d3.easeLinear)
            .duration(2000)
            .delay(delay)
            .style("fill", "red")
            .attr("height",100)
            // Shrink transition
            .transition()
            .ease(d3.easeLinear)
            .duration(2000)
            .delay(delay)
            .style("fill", "blue")
            .attr("height",20)

    };
```

Colour of the bar now changes along with the transitions.

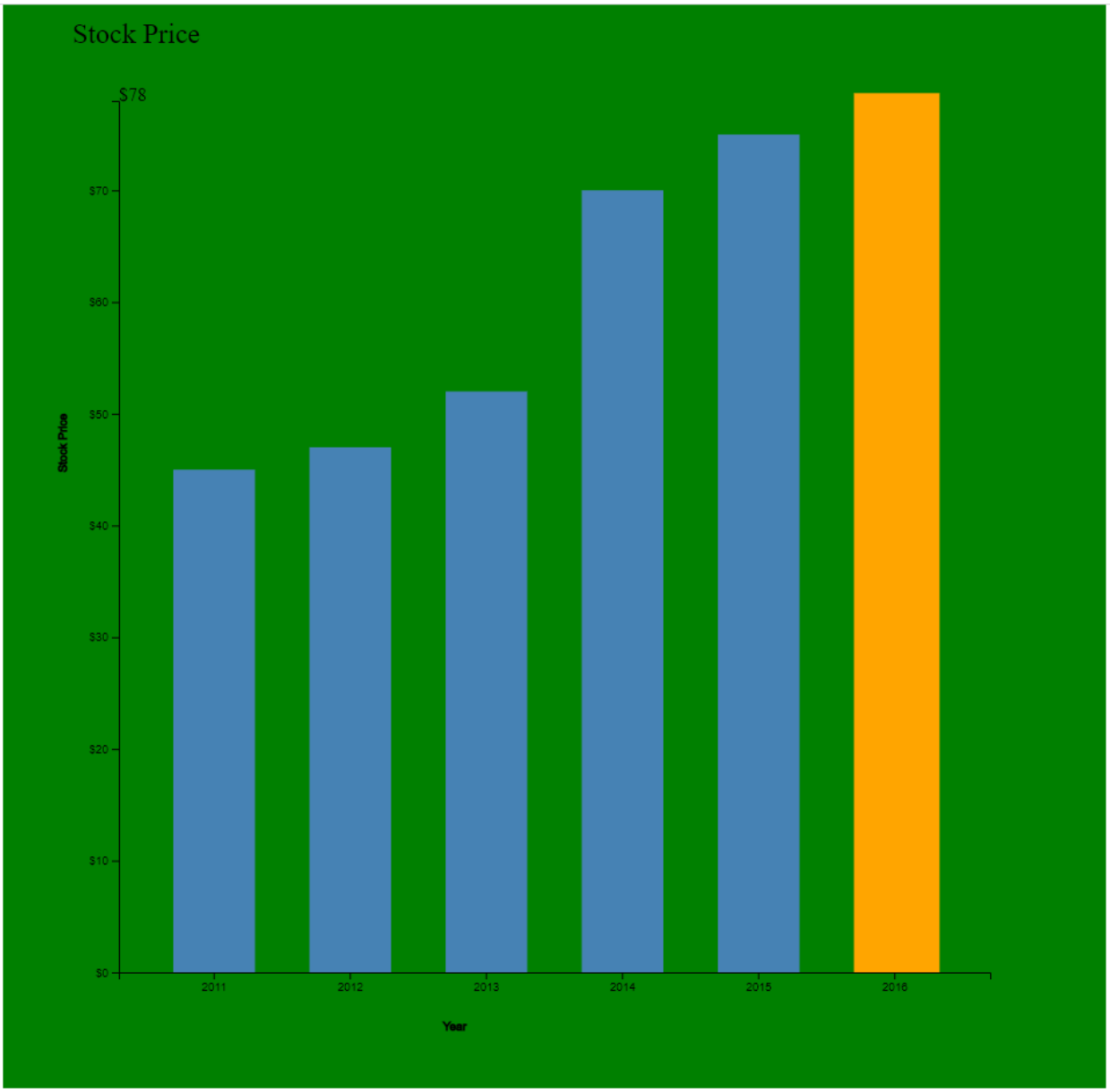
F21DV Data Visualisation and Analytics – Lab 2
Josh Yang jy84 091514042

Exercise 15

Initialised:



Hover over bar:

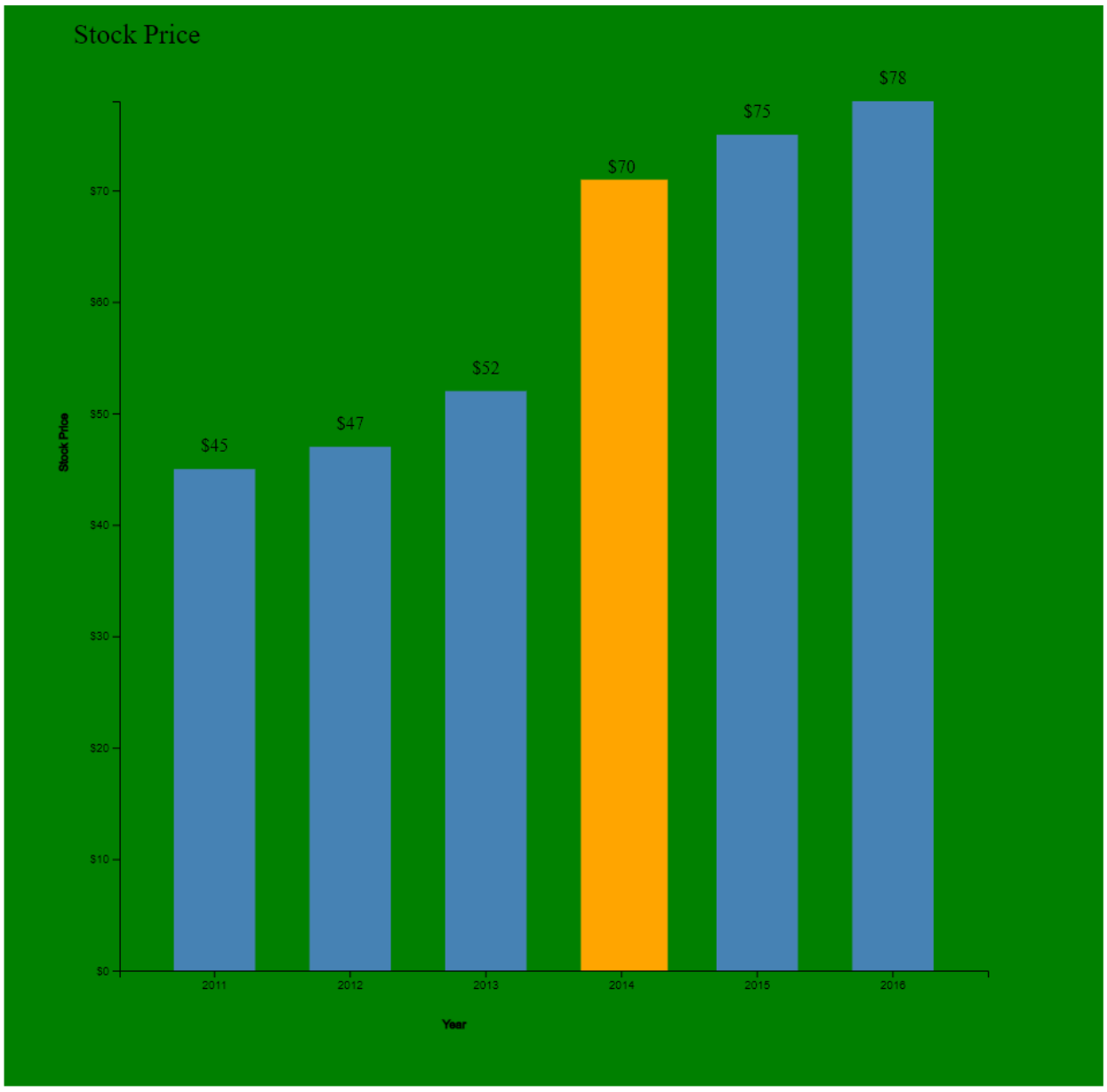



Bar changes colour and grows when mousing over. Also, text is displayed in the top left which provides the value of the bar.

Exercise 16

```
            // https://stackoverflow.com/questions/67480486/d3-js-add-text-above-
bar-chart-not-show
            // Appending a text element
            g.selectAll(".bar-title")
```

```
            .data(data)
            .enter()
            .append("text")
            .classed('bar-title', true)
            .attr('text-anchor', 'middle')
            .attr("x", d => x(d.year) + x.bandwidth()/2)
            .attr("y", d => y(d.value) - 15)
            .text(d => `$${d.value}`);
            };
```



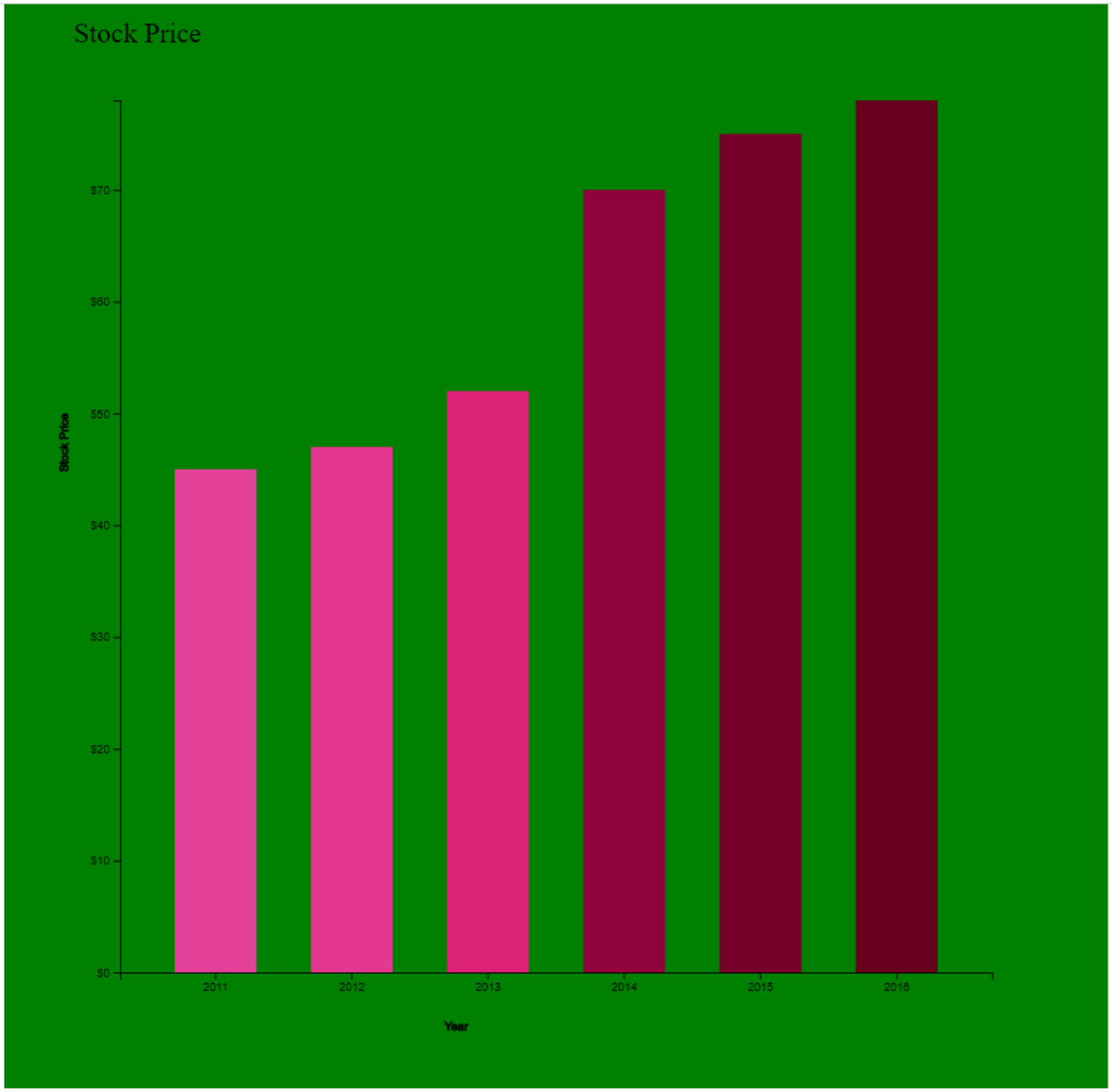

Mousing over the bar will display the value of the data on top of the bars.

Exercise 17

```
        // Colour scheme
        var myColor = d3.scaleSequential()
                        // Domain set from 0 to max value in the dataset
                        .domain([0,d3.max(data, function(d) {
                            return d.value;
                            })])
                        .interpolator(d3.interpolatePuRd);
```

```
.attr("fill",  d => myColor(d.value))
```

F21DV Data Visualisation and Analytics – Lab 2
Josh Yang jy84 091514042




The bars' colour is now scaled based on the sequential colour scheme.

F21DV Data Visualisation and Analytics – Lab 2
Josh Yang jy84 091514042

Exercise 18


Added a third dataset and relevant button

Exercise 19

```
    <button onclick="update(data1,colours[0])">Variable 1</button>
    <button onclick="update(data2, colours[1])">Variable 2</button>
    <button onclick="update(data3, colours[2])">Variable 3</button>
```

When the button is clicked the data and colour is provided to the function which is then applied to the bars.

Exercise 20

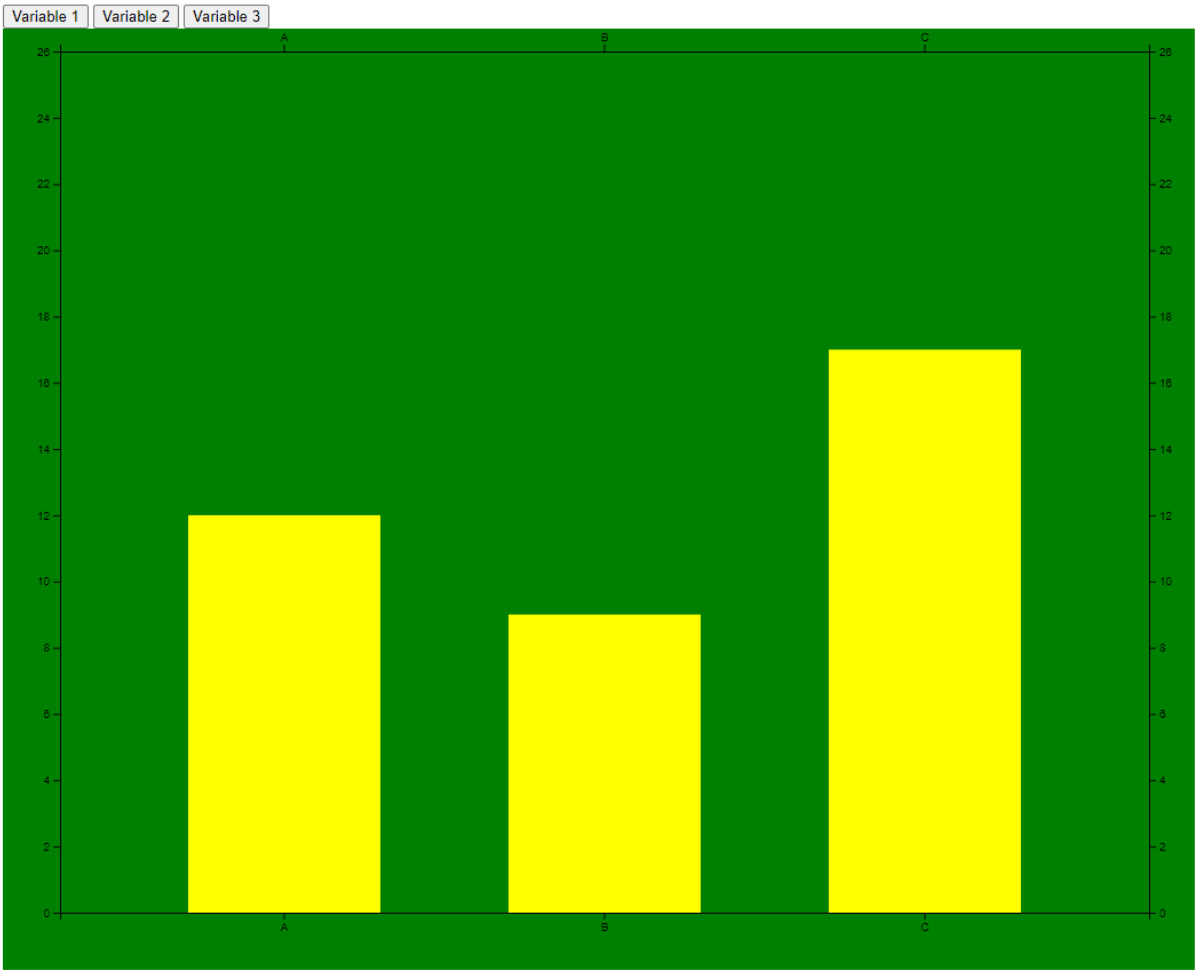
```
            // https://stackoverflow.com/questions/34179006/d3-text-on-mouseover
            // Value of the bar will be displayed when the mouse hovers over
            .on("mouseover", function(d, i){
                d3.select(this.parentNode)
                    .append("text")
```

```
                .attr('text-anchor', 'middle')
                .classed('bar-title', true)
                .attr("x", d => x(i.group) + x.bandwidth()/2)
                .attr("y", d => y(i.value) - 15)
                .text(d => `${i.value}`)
        })
        .on("mouseout", function(){
            // Remove the text label
            d3.selectAll('.bar-title')
                .remove()
        })
```

The i parameter will give access to the data element.

F21DV Data Visualisation and Analytics – Lab 2
Josh Yang jy84 091514042

Exercise 21

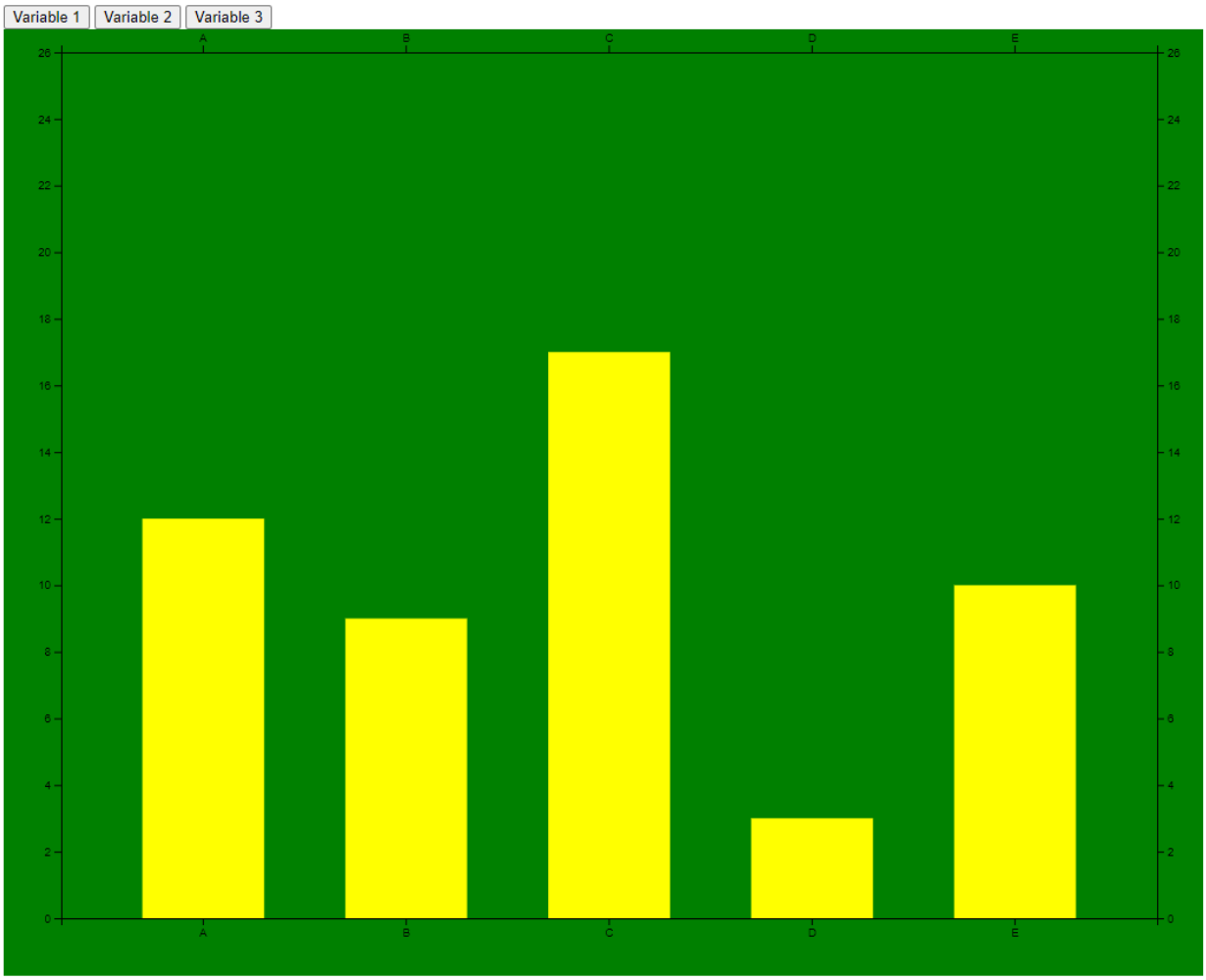



```
// Appending top axis
svg.append("g")
    .attr("transform", "translate(" + margin.bottom + ","+ margin.top + ")")
    .call(d3.axisTop(x));

// Appending right axis
svg.append("g")
    .attr("class", "myYaxis")
    .attr("transform", "translate(" + (ySize + 250) + ","+ margin.left +")")
    .call(d3.axisRight(y));
```

Top and right axis has been added to the graph.

F21DV Data Visualisation and Analytics – Lab 2
Josh Yang jy84 091514042

Exercise 22

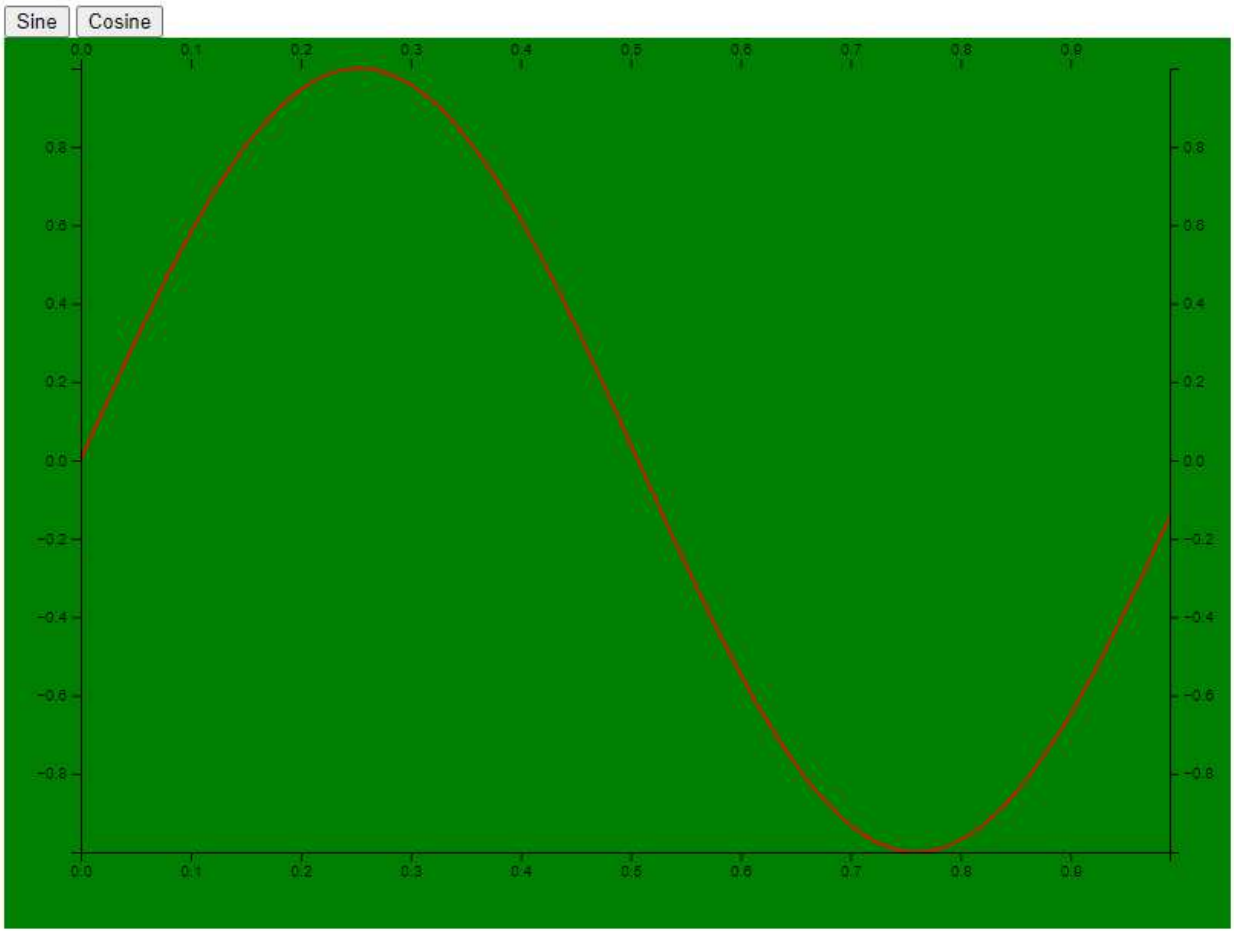



```
    // A function that create / update the plot for a given dataset
    function update(data, colour) {

        // Removing all the elements from the svg
        // https://stackoverflow.com/questions/14422198/how-do-i-remove-all-
children-elements-from-a-node-and-then-apply-them-again-with
        svg.selectAll("*").remove();

        // Setting scales
        var x = d3.scaleBand()
        .range([0, xSize])
        .domain(data.map(function(d) {
            return d.group
        }))
        .padding(0.4);
```

F21DV Data Visualisation and Analytics – Lab 2
Josh Yang jy84 091514042

Must include the scale function for the x axis within the update function as the scale is based on the number of groups.

Exercise 23




When the button is clicked the data will be passed to the ‘graph’ function and the new line will be displayed.

Exercise 24

```
let intr = d3.interpolate( [20, 40, 4], [1, 12, 10])
console.log("Type of returned function is: ", typeof (intr) );
// Type of returned function is:  function
console.log( intr(0.2) )
// 0: 16.2
// 1: 34.4
// 2: 5.2
```

```
    // The d3-interpolate module offers a way of computing intermediate values
between two given values
    // When an array of the same length is provided it will interpolate the
elements at the same index
    // 0.2 is closer to element a in the first array whereas 0.5 would be the
halfway point and closer to 1 would be the second array
```

Exercise 25

```
    let cc = d3.interpolate('red', 'green')
    console.log( cc(0.5) );
    // rgb(128, 64, 0)
      // Saddle Brown

    // The d3-interpolate module offers a way of computing intermediate values
between two given values
    // A color interpolator is usually represented as a color ramp going from 0
to the left (color a), to 1 to the right (color b).
    // https://observablehq.com/@d3/d3-interpolate
```

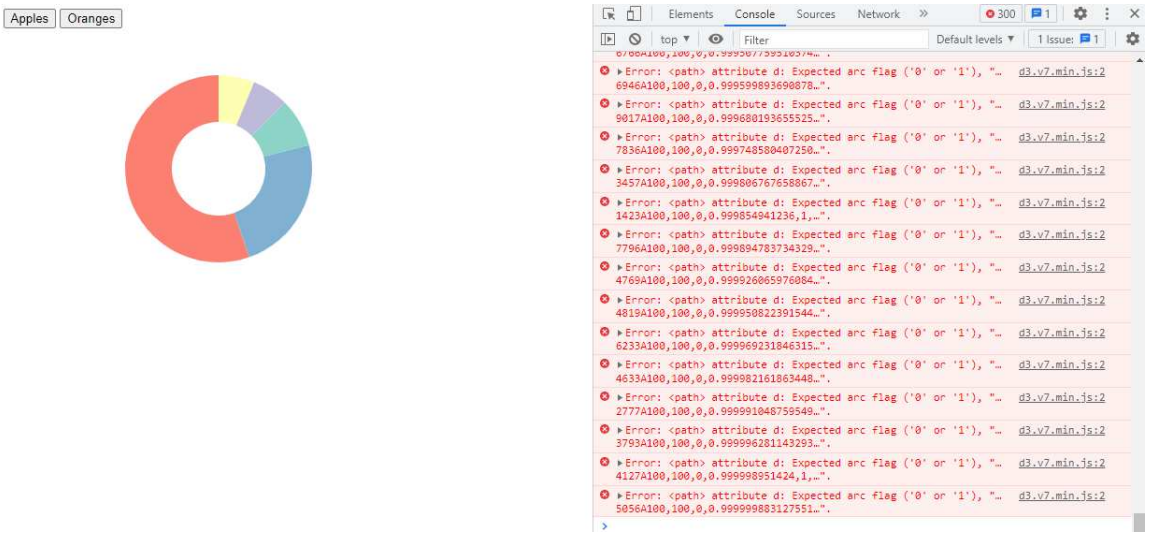
Exercise 26

```
    let date = d3.interpolateDate(new Date("1970-01-01"), new Date("1971-01-01"))

    console.log(date(0.25) );
    // Thu Apr 02 1970 07:00:00 GMT+0100 (Greenwich Mean Time)

    // The d3-interpolate module offers a way of computing intermediate values
between two given values
    // Given two dates start and end, it returns an interpolator that builds an
intermediate Date object between the start and end dates.
    // https://observablehq.com/@d3/d3-interpolatedate
```

Exercise 27

```
    // A function that create / update the plot for a given dataset
    const pieChart = data => {

        var pie = d3.pie()
                .value(function(d) {return d; })
                .sort(function(a, b) {
                    return d3.ascending(a, b);
                    } ) // This make sure that group order remains the same in
the pie chart
```

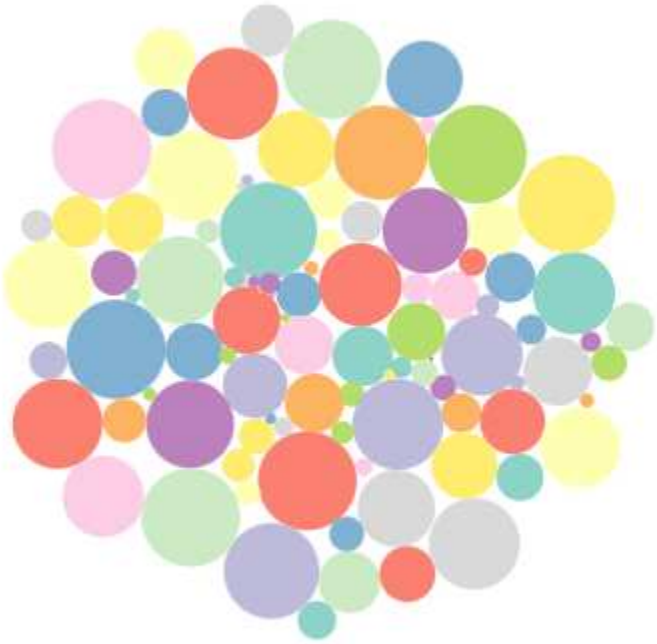
```
        var data_ready = pie(data)

        var update = svg.selectAll("path")
                        .data(data_ready);

        update.enter()
                .append("path")
                .merge(update)
                .transition()
                .duration(5000)
                .attr("d", arc)
                .attr("fill", function(d, i) {
                    return color(i);
                });

        update.exit()
                .remove()
    };
```




The transition between the two datasets works except for the first element in the dataset which causes the bug shown. The data for the red element disappears and then reappears once the transition has finished. I was unable to fix this bug.

F21DV Data Visualisation and Analytics – Lab 2
Josh Yang jy84 091514042

Exercise 28



```
// Colour scheme
// https://www.d3-graph-gallery.com/graph/custom_color.html
var myColor = d3.scaleOrdinal().domain([1,numNodes]).range(d3.schemeSet3);
```

```
                    // based on the index of the data
                    .style('fill', function(d, i) {
                        return myColor(i)
                    })
```

The colour of the of the nodes are provided by the ordinal colour scheme.

F21DV Data Visualisation and Analytics – Lab 2
Josh Yang jy84 091514042

Exercise 29

```
// Reading in csv file
  d3.csv('./Data.csv').then(function(data) {
```

Data is taken from a csv file and creates the necessary nodes.

Exercise 30

```
                    .on("mouseover", function(d, i){
                        d3.select(this.parentNode)
                            .append("text")
                            .attr('text-anchor', 'middle')
                            .classed('bar-title', true)
                            .attr("x", d => i.x)
                            .attr("y", d => i.y)
                            .text(d => `${i.radius}`)
                            .style("font-size", '9px')
                    })
                    .on("mouseout", function(){
                        // Remove the text label
                        d3.selectAll('.bar-title')
                        .remove()
                    });
```

F21DV Data Visualisation and Analytics – Lab 2
Josh Yang jy84 091514042

On mouse over the radius of the node will be displayed and when moving the mouse outside, it will be removed.

Exercise 31

```
                        .on("mouseover", function(d, i){
                            d3.select(this)
                                // based on the index of the data
                                .style('fill', function(d) {
                                    return myColor(d)
                                });
                        });
```

The colour of the node will change when mouse overed

Exercise 32

```
        //https://roshansanthosh.wordpress.com/2016/09/25/forces-in-d3-js-v4/
        var attractForce = d3.forceManyBody().strength(100).distanceMax(400)
                    .distanceMin(60);

        var repelForce = d3.forceManyBody().strength(-1000).distanceMax(50)
                   .distanceMin(10);

        var simulation = d3.forceSimulation(data).alphaDecay(0.1)
                  .force("attractForce",attractForce)
                  .force("repelForce",repelForce)
                  .force("x", d3.forceX(200).strength(0.1))
                 .force("y", d3.forceY(200).strength(0.1))
                 .on('tick', ticked);
```

```
circle:hover {
    -webkit-animation-name: pulsar;
    -webkit-animation-duration: 0.5s;
    -webkit-animation-iteration-count: infinite;
    -webkit-animation-direction: alternate;
    animation-name: pulsar;
    animation-duration: 0.5s;
    animation-iteration-count: infinite;
    animation-direction: alternate;
    -webkit-transform-origin: 50% 50%;
    transform-origin: 50% 50%;
}

@keyframes pulsar {
```

```
    from {
        fill: red;
    }
    to {
        transform: scale(1.2,1.2);
        transform-origin: 50% 50%;
    }
}
```

Tinkered around with different forces to understand the params and how changing them effects the simulation.

Took inspiration from the earlier CSS example to make the nodes pulse when they are hovered over.