F21DV - Data Visualization and Analytics

CW -Lab 2

Ankitha Susan Cherian[H00392039]

Demonstrated to: Benjamin Kenwright

Demonstration on: 28th February 2022

Due date: 25th February 2022

School of Mathematical and Computer Sciences

Heriot-Watt University

Introduction

The purpose of Lab2 coursework is to different CSS effects like pulse actions using @keyframes, hovering
and other CSS styling. Part2 of the coursework dealt with different events mainly the mouse events (mouseout and mouseover) and binding of the events to the DOM elements. We had exercises to capture colour and size changes during mouse events. We used d3. pointer to get the X and Y coordinates based on mouse position and to display value of charts or graphs as texts. Part 3 tasks were based on various transition concepts like easing effects, setting duration and delay time for transitions to take place. Transactions based on Changing data with a click event or mouse event was also implemented. Using Interpolate functions like d3. interpolate(a,b), d3.interpolateNumber(a,b),d3,interpolateDate(a,b) etc, we could find the intermediate values for a given set of number, array, colour, date etc. Lastly, we used d3. force layout to translate the nodes to specific position pulling to the center and the collision function to prevent nodes from overlapping.

Part 1. CSS Effects/Animations

Exercise 1: Using the keyframes animation concept from the example above, write a simple D3 program that draws a 'line-graph' For each of the points on the graph, draw a small 'svg' circle. Set an animated keyframe style on each graph point, so when the mouse cursor moves over the point, it 'pulses'

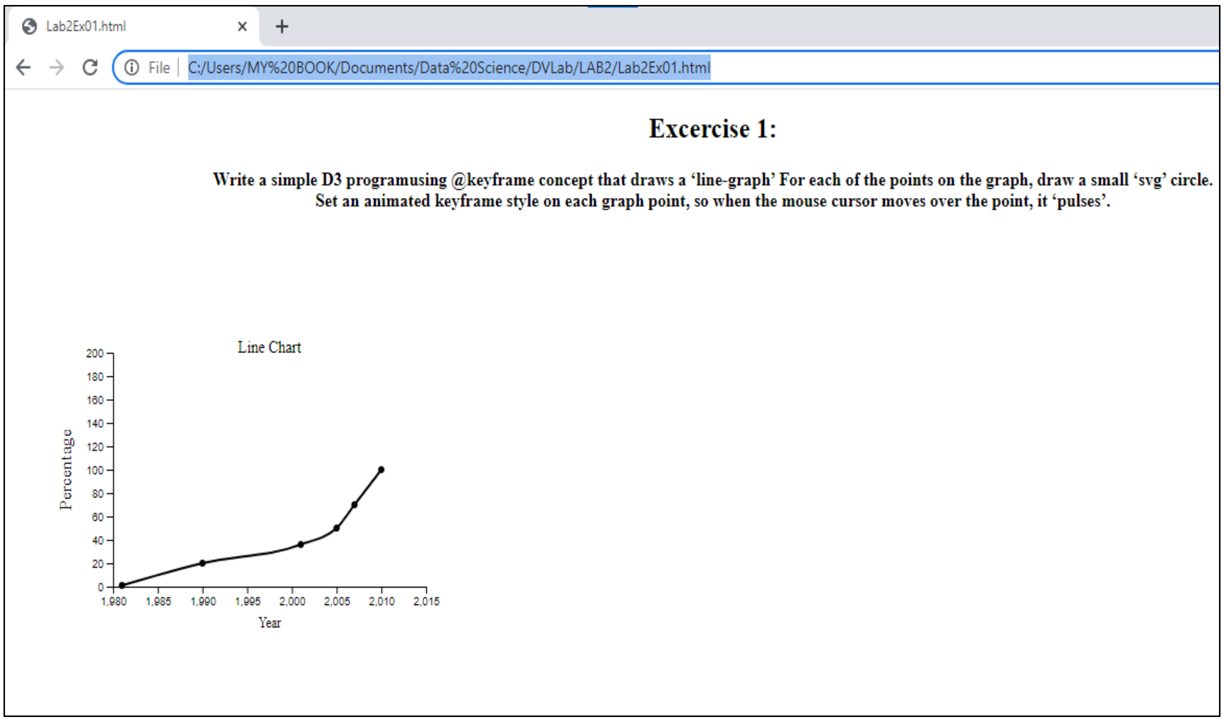
Code: Lab1Ex01.

Snippet:

```
<script>
        //creating new dataset for and y axis
      var dataset1 = [
        [1981, 1],
        [1990, 20],
        [2001, 36],
        [2005, 50],
        [2007, 70],
        [2010, 100],
      ];
       //setting margin for the axes
      var svg = d3.select("svg"),
        margin = 200,
        width = svg.attr("width") - margin,
        height = svg.attr("height") - margin;

        //setting scale for x-axis and y-axis
      var xScale = d3.scaleLinear().domain([1980, 2015]).range([0, width])
        yScale = d3.scaleLinear().domain([0, 200]).range([height, 0]);

          //appending 'g' to add multiple svg elements
      var g = svg
        .append("g")
        .attr("transform", "translate(" + 100 + "," + 100 + ")");

         //appending title for line graph
      svg
        .append("text")
        .attr("x", width / 2 + 100)
        .attr("y", 100)
        .attr("text-anchor", "middle")
        .style("font-size", "14px")
        .text("Line Chart");

        //appending title for x-axis
      svg
        .append("text")
        .attr("x", width / 2 + 100)
        .attr("y", height - 15 + 150)
        .attr("text-anchor", "middle")
        .style("font-size", 12)
        .text("Year");

         //appending title for y-axis
      svg
```

```
        .append("text")
        .attr("text-anchor", "middle")
        .attr("transform", "translate(60," + height + ")rotate(-90)")
        .text("Percentage");

        //calling the left and bottom axis
      g.append("g")
        .attr("transform", "translate(0," + height + ")")
        .call(d3.axisBottom(xScale));

      g.append("g").call(d3.axisLeft(yScale));

      //appending dots to the line graph based on values from dataset
      svg
        .append("g")
        .selectAll("dot")
        .data(dataset1)
        .enter()
        .append("circle")
        .attr("cx", function (d) {
          return xScale(d[0]);
        })
        .attr("cy", function (d) {
          return yScale(d[1]);
        })
        .attr("r", 3)
        .attr("transform", "translate(" + 100 + "," + 100 + ")")
        .attr("class","pulse-circle"); // calling pulse-circle class for pulse
action of dot

        //appending line to the svg based on given scale of x-axis and y-axis
      var line = d3
        .line()
        .x(function (d) {
          return xScale(d[0]);
        })
        .y(function (d) {
          return yScale(d[1]);
        })
        .curve(d3.curveMonotoneX);

      svg
        .append("path")
        .datum(dataset1) //to bind data to multiple dom element
        .attr("class", "line")
        .attr("transform", "translate(" + 100 + "," + 100 + ")")
        .attr("d", line)
        .style("fill", "none")
        .style("stroke", "black")
        .style("stroke-width", "2");
    </script>
```

Partial Output:


Lab2Ex01.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex01.html

Excercise 1:

Write a simple D3 programusing @keyframe concept that draws a 'line-graph' For each of the points on the graph, draw a small 'svg' circle. Set an animated keyframe style on each graph point, so when the mouse cursor moves over the point, it 'pulses'.

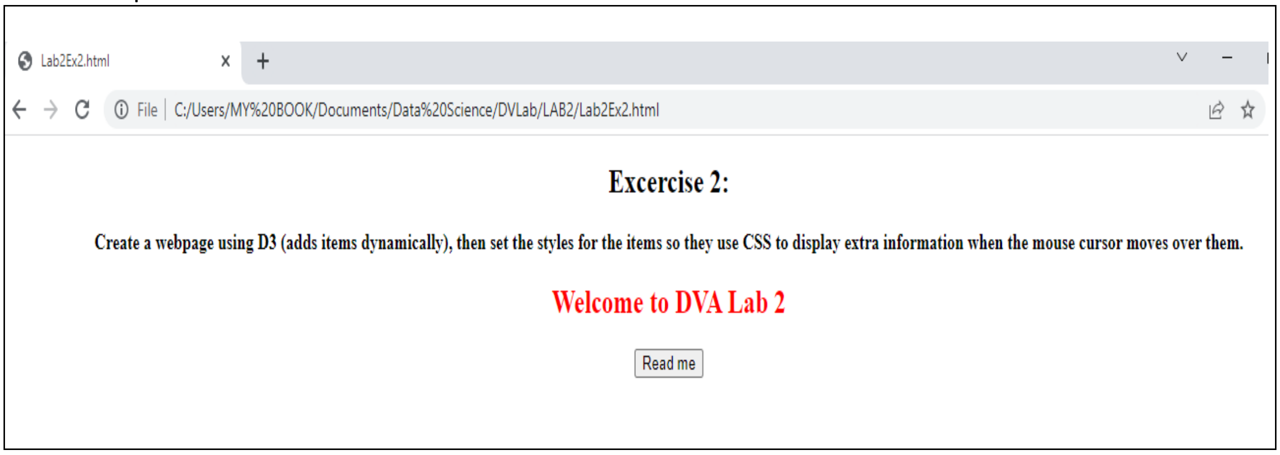
Exercise 2:

Create a webpage using D3 (adds items dynamically), then set the styles for the items so they use CSS to display extra information when the mouse cursor moves over them.

Snippet:
DOM elements were dynamically added to implement hovering action in CSS styling

```
<script>
        //adding html items dynamically
        document.getElementById("btn").
            addEventListener("click", function () {
                document.getElementById("innerdiv").
                    innerHTML += "<span>CSS </span>"

            });
        document.ready(function () {
            (".newspan").hover(function (e) {

                var description = ' <span>' + "Hello World" + '</span>';
                document.getElementById('newspan').innerHTML = description;

            }, function (e) {
                document.getElementById('newspan').innerHTML = '';
            });
        });

    </script>
```

Partial Output:



Lab2Ex2.html

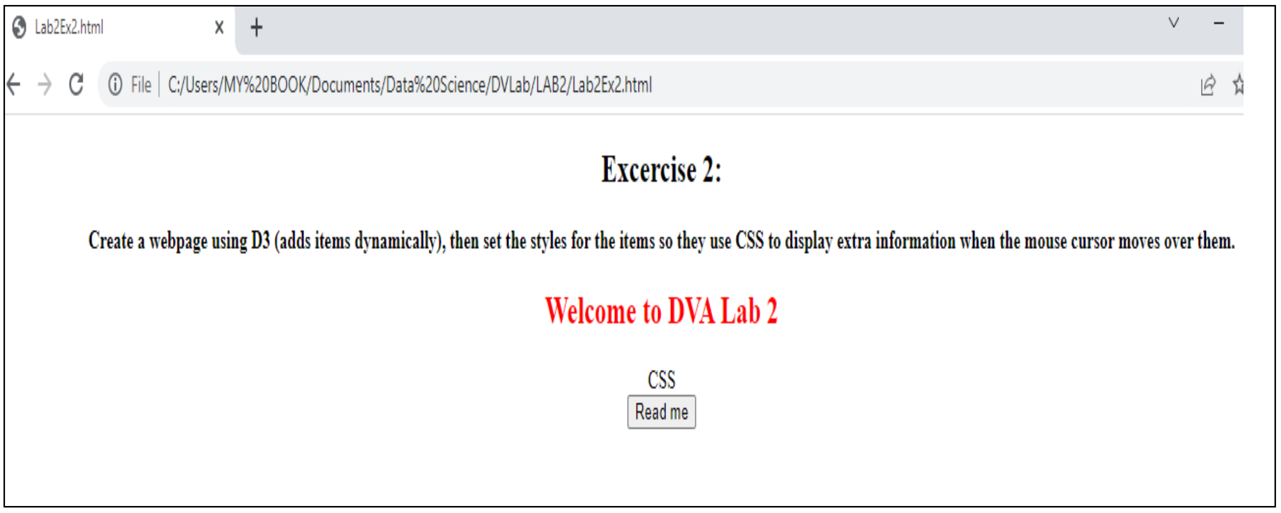
File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex2.html

Excercise 2:

Create a webpage using D3 (adds items dynamically), then set the styles for the items so they use CSS to display extra information when the mouse cursor moves over them.

Welcome to DVA Lab 2

Read me


Lab2Ex2.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex2.html

Excercise 2:

Create a webpage using D3 (adds items dynamically), then set the styles for the items so they use CSS to display extra information when the mouse cursor moves over them.

Welcome to DVA Lab 2

CSS

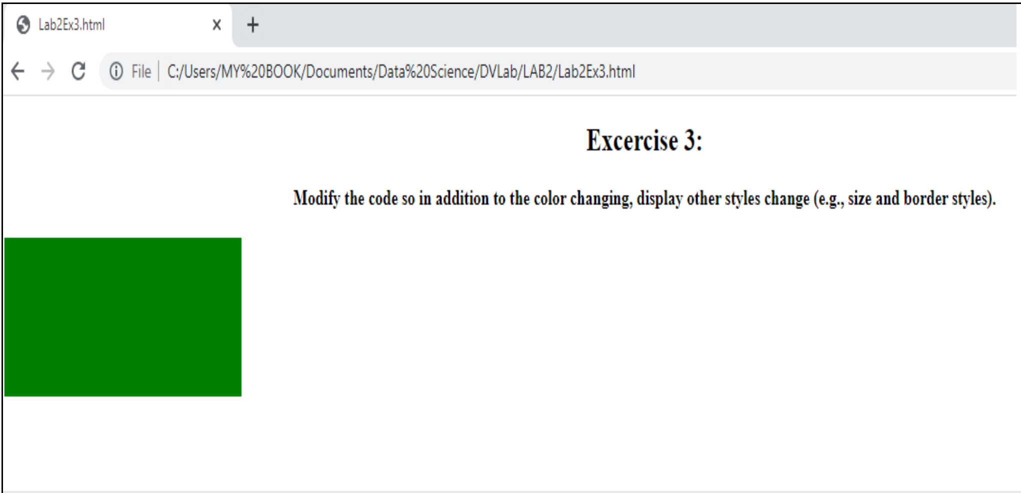
Part 2. Events

Exercise 3: Display div element in addition to the color changing with other styles change (e.g., size and border styles)

Snippet:
Different colour ,size and border styles were added to 3 div elements

```
 <script>
       //adding div element dynamically to the body
      d3.select('body')
      .append('div')
      .style('width', '250px')
      .style('height', '120px')
      .style('background-color', 'green');

       //selecting all the div element
      d3.selectAll("div")

      //during mouseover event, the current div changes color to blue with a
double border
      .on("mouseover", function(event){
        d3.select(this)
        .style("background-color", "steelblue")
        .style('border-style','double');
         console.log(event); // Get current event info
         console.log(d3.pointer(event));  //Get x & y co-ordinates
      })

       //during mouseout event, the current div changes color to red with a
dashed border
      .on("mouseout", function(){
          d3.select(this)
          .style("background-color", "red")
          .style('width', '500px')
          .style('height', '200px')
          .style('border-style','dashed')
          });
   </script>
```

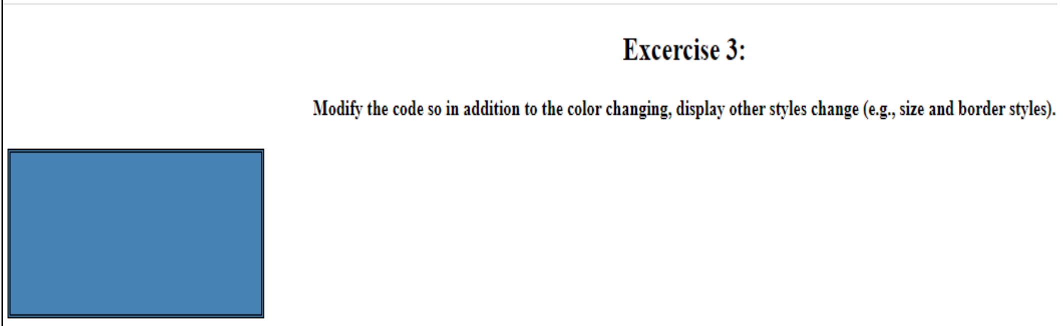
Output:


Lab2Ex3.html

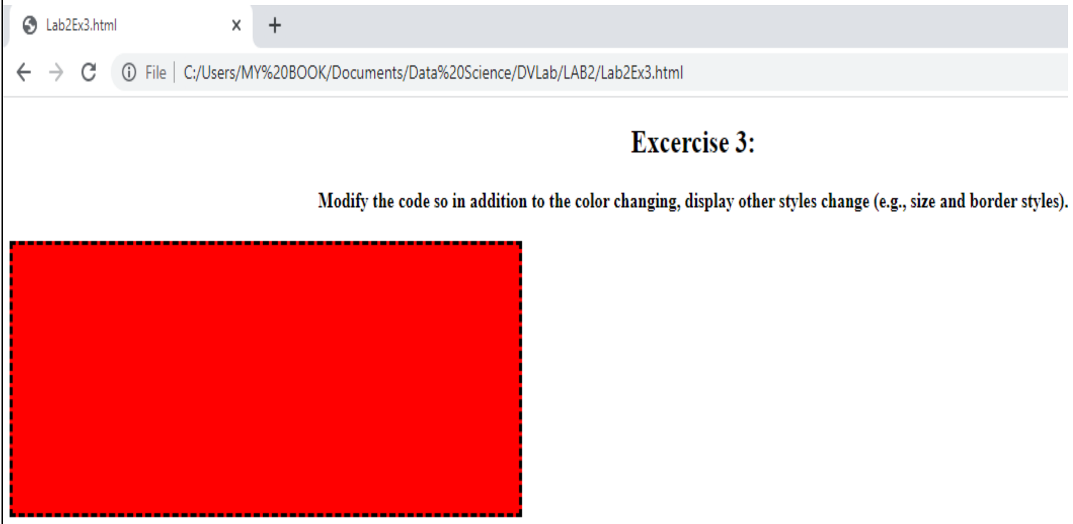
File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex3.html

Excercise 3:

Modify the code so in addition to the color changing, display other styles change (e.g., size and border styles).


Excercise 3:

Modify the code so in addition to the color changing, display other styles change (e.g., size and border styles).


Lab2Ex3.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex3.html

Excercise 3:

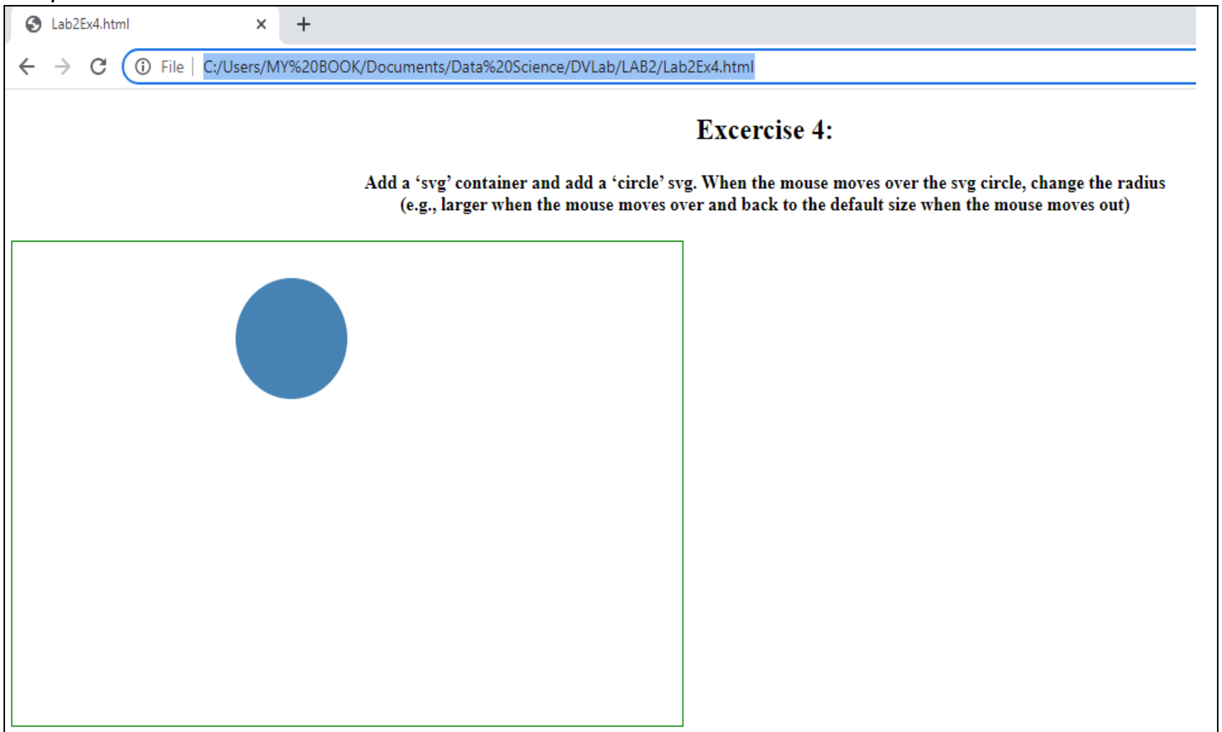
Modify the code so in addition to the color changing, display other styles change (e.g., size and border styles).

Exercise 4:
Use an ‘svg’ container and add a ‘circle’ svg. When the mouse moves over the svg circle, change the radius (e.g., larger when the mouse moves over and back to the default size when the mouse moves out).

Snippet:
Circle radius was changed to grow in mouseover and shrink in mouseout event

```
    // radius grows than the original size in mouseover event
    d3.selectAll("circle")
     .on("mouseover", function(event){
      d3.select(this)
        .style("fill", "red") //changing color to red
        .attr("cx", 250)
        .attr("cy", 250)
        .attr("r", 80) // the circle grows as radius in increased

        })
    //radius shrinks to its original size in mouseout event
    .on("mouseout", function(){
      d3.select(this)
        .style("fill", "steelblue")
        .attr("cx", 250)
        .attr("cy", 80)
        .attr("r", 50)
       });
```

Output:

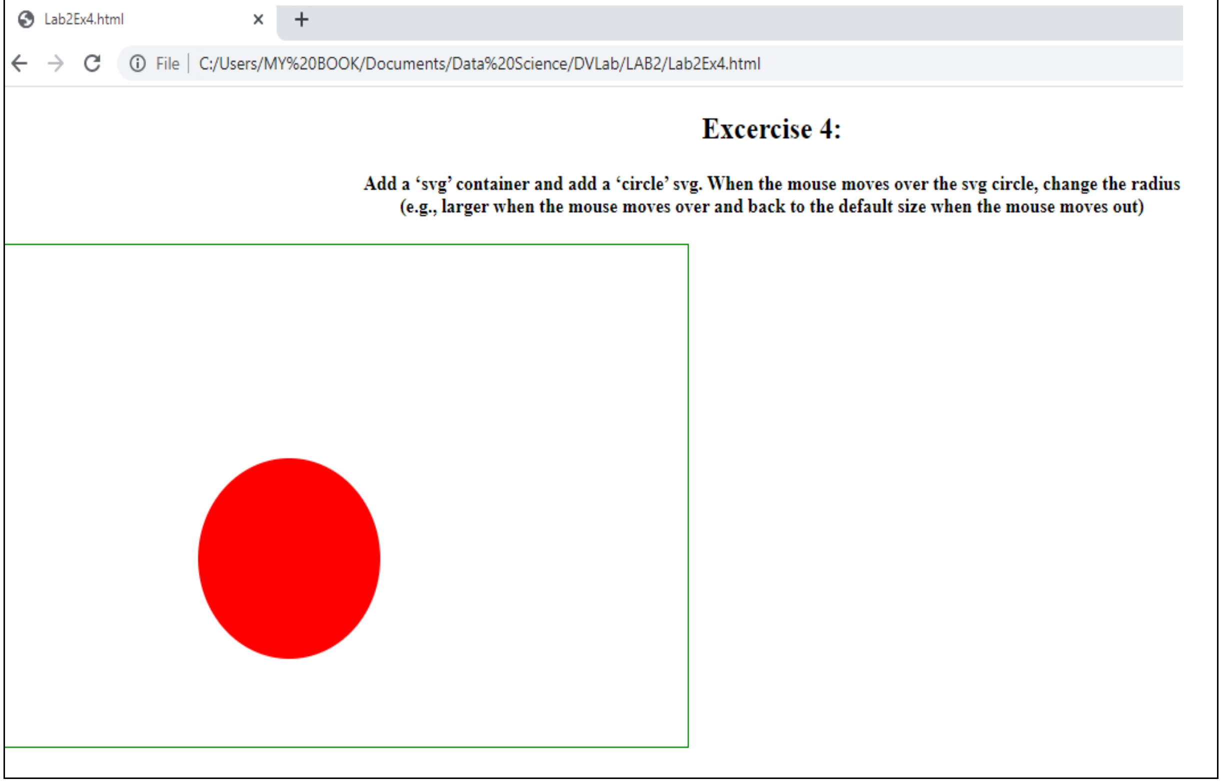


Lab2Ex4.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex4.html

Excercise 4:

Add a 'svg' container and add a 'circle' svg. When the mouse moves over the svg circle, change the radius (e.g., larger when the mouse moves over and back to the default size when the mouse moves out)

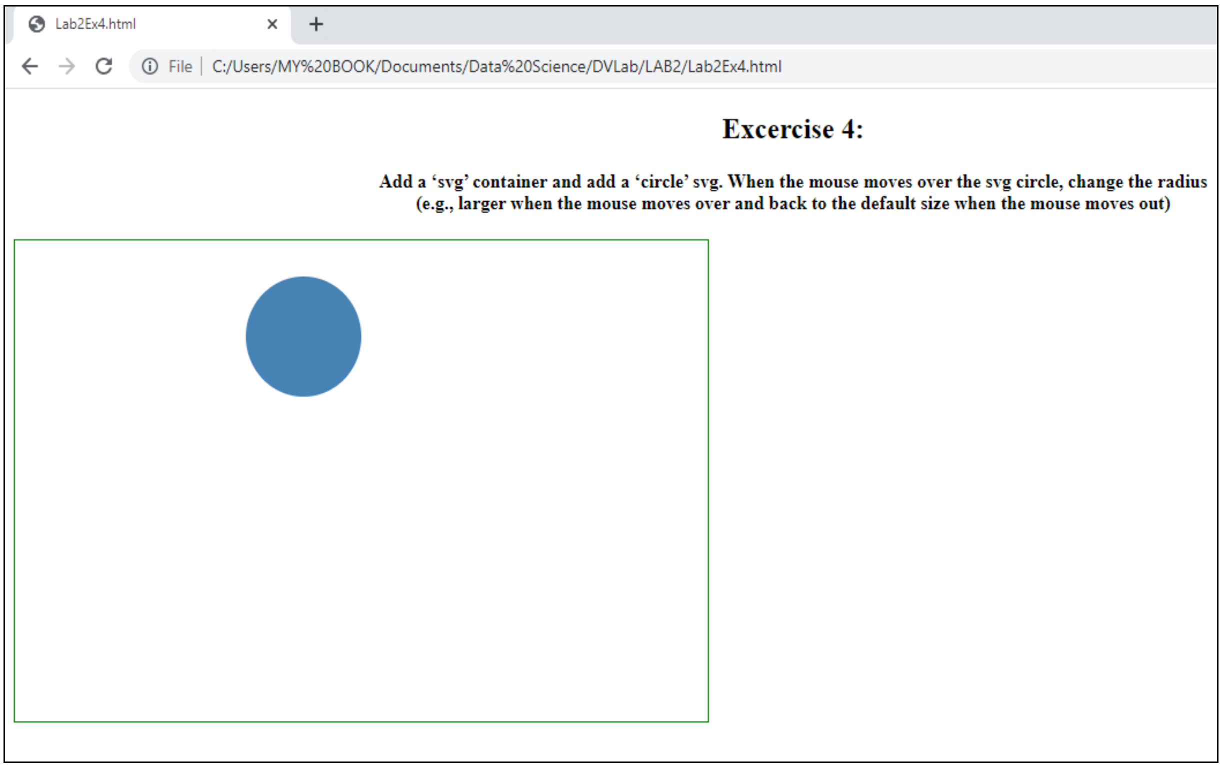


Lab2Ex4.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex4.html

Excercise 4:

Add a 'svg' container and add a 'circle' svg. When the mouse moves over the svg circle, change the radius (e.g., larger when the mouse moves over and back to the default size when the mouse moves out)


Lab2Ex4.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex4.html

Excercise 4:

Add a ‘svg’ container and add a ‘circle’ svg. When the mouse moves over the svg circle, change the radius (e.g., larger when the mouse moves over and back to the default size when the mouse moves out)

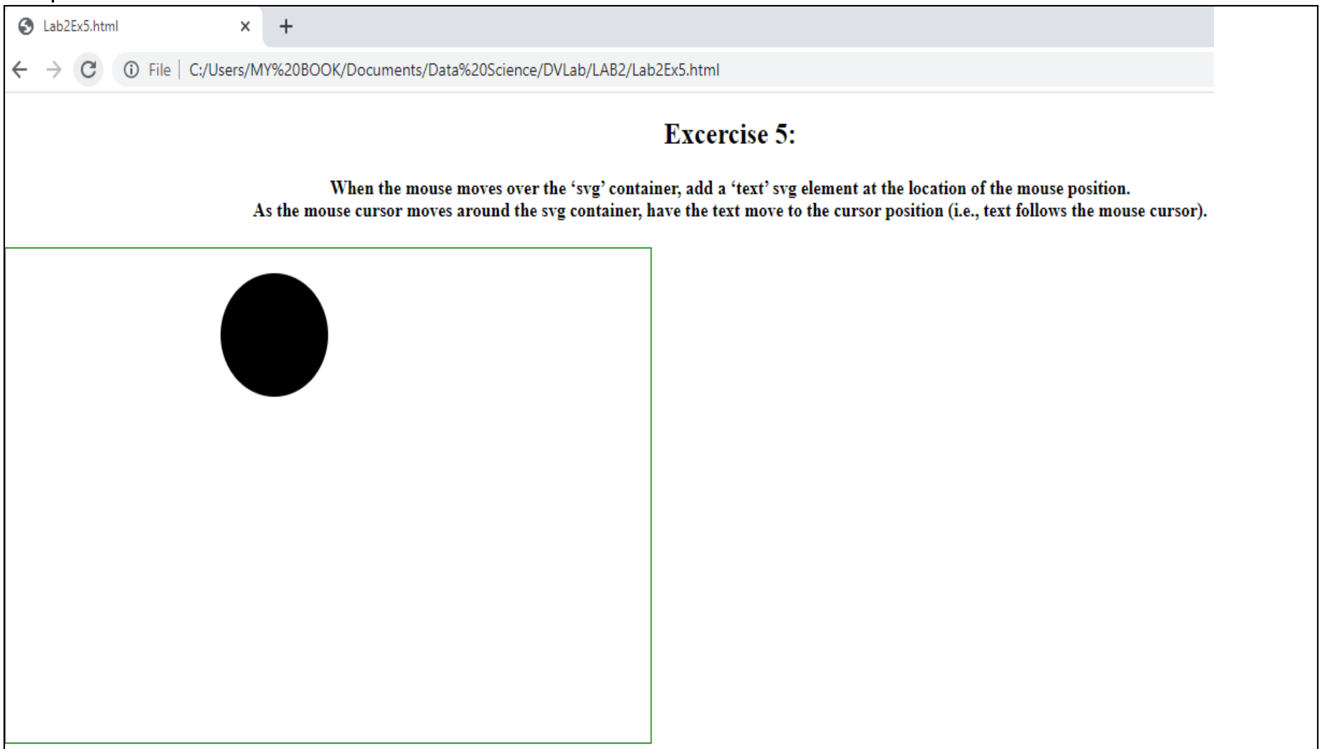
Exercise 5:

When the mouse moves over the ‘svg’ container, add a ‘text’ svg element at the location of the mouse position. As the mouse cursor moves around the svg container, have the text move to the cursor position (i.e., text follows the mouse cursor)

Snippet:

```
<script>
       //Appending svg container to the html body
       var svg = d3
         .select("body")
         .append("svg")
         .attr("width", 600)
         .attr("height", 400)
         .style("border", "1px solid green");

       //Appending circle
       svg.append("circle")
           .attr("cx", 250)
           .attr("cy", 70)
           .attr("r", 50);
           d3.selectAll("circle")

         //getting mouse positions for x and y cordinates using d3.pointer
         .on("mouseover", function (event) {
           var coordinates = d3.pointer(event);
           var x_c = coordinates[0];
           var y_c = coordinates[1];

           //styling the circle at mouseover event
           d3.select(this)
             .style("fill", "steelblue")
             .attr("cx", 250)
             .attr("cy", 90)
             .attr("r", 70);

           //appending text  to svg during mouse event
           svg.append("text").attr("x", x_c).attr("y", y_c).text("Hello");
         })
         //removing the text at mouseout event
         .on("mouseout", function () {
           //d3.select("text").select(".text").remove;
           d3.selectAll("text").remove();
         });
```

Output:

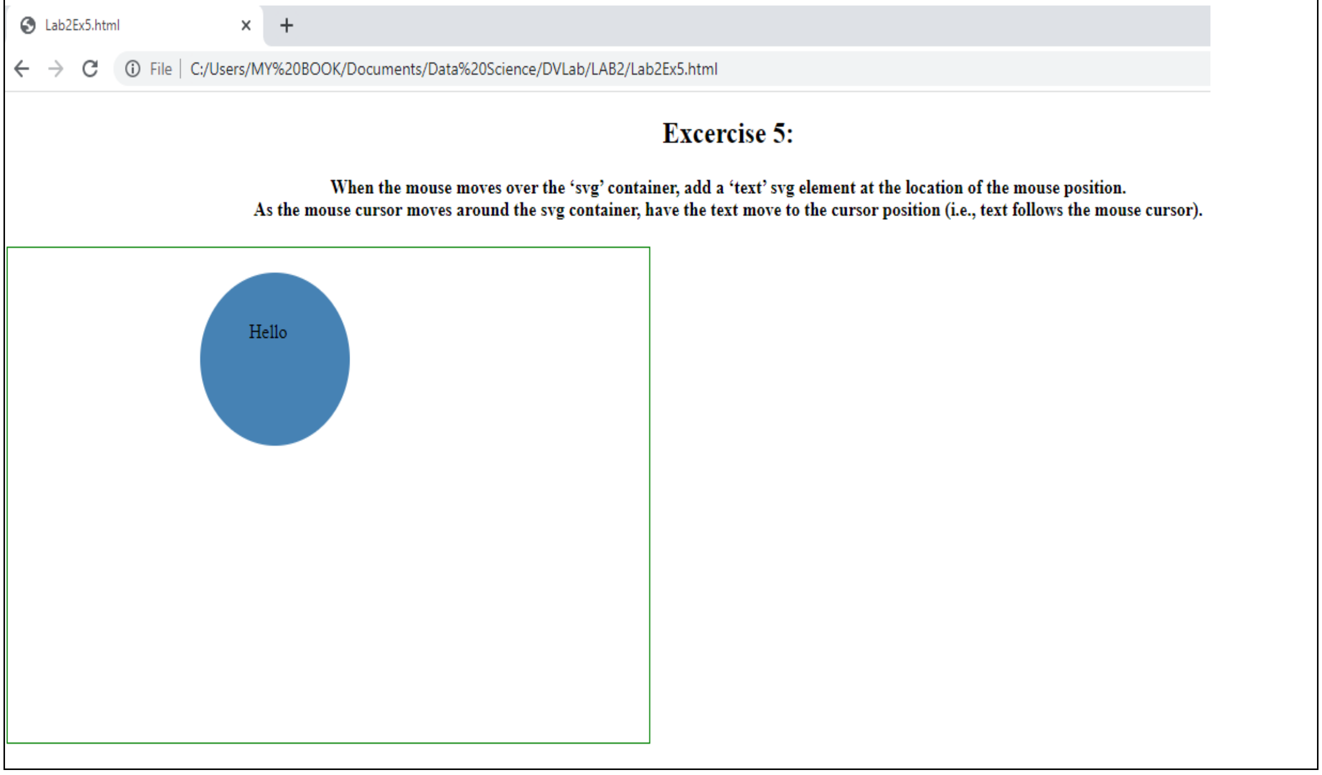


Lab2Ex5.html

C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex5.html

Excercise 5:

When the mouse moves over the ‘svg’ container, add a ‘text’ svg element at the location of the mouse position.
As the mouse cursor moves around the svg container, have the text move to the cursor position (i.e., text follows the mouse cursor).



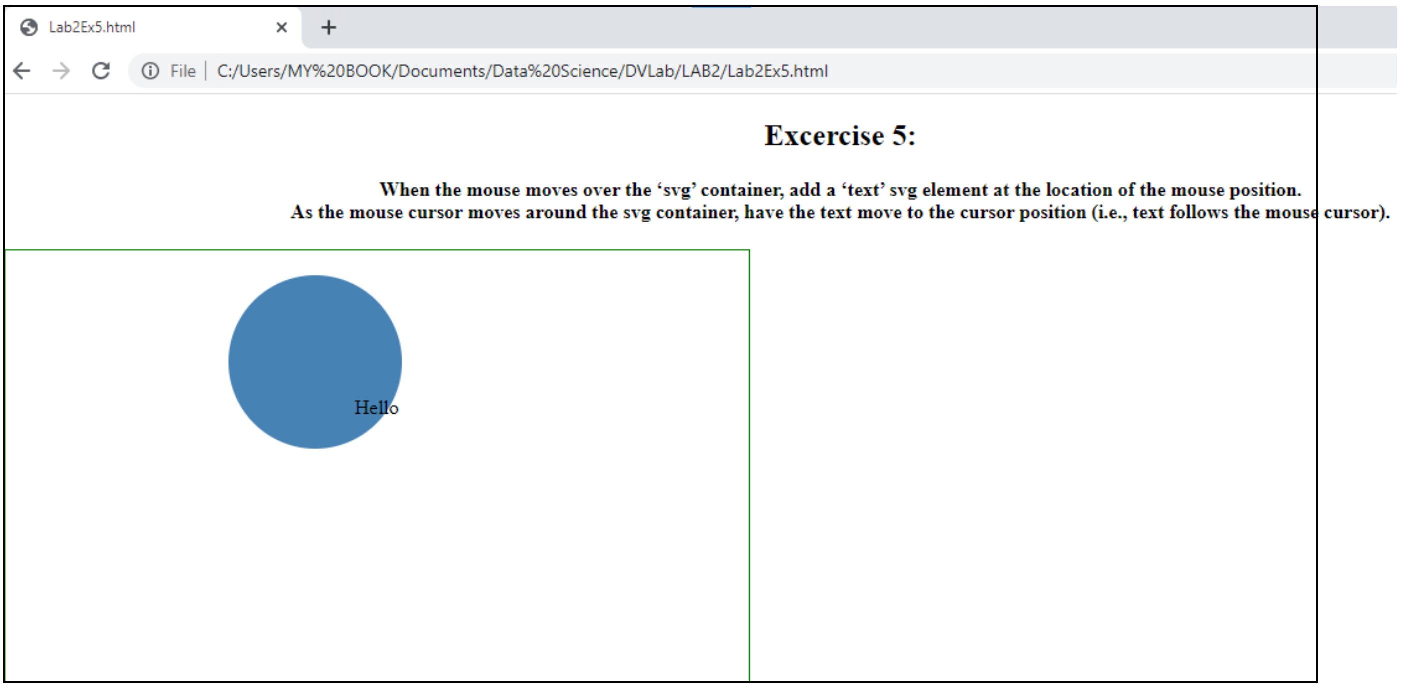
Lab2Ex5.html

C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex5.html

Excercise 5:

When the mouse moves over the ‘svg’ container, add a ‘text’ svg element at the location of the mouse position.
As the mouse cursor moves around the svg container, have the text move to the cursor position (i.e., text follows the mouse cursor).

Hello



Lab2Ex5.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex5.html

Excercise 5:

When the mouse moves over the 'svg' container, add a 'text' svg element at the location of the mouse position.
As the mouse cursor moves around the svg container, have the text move to the cursor position (i.e., text follows the mouse cursor).

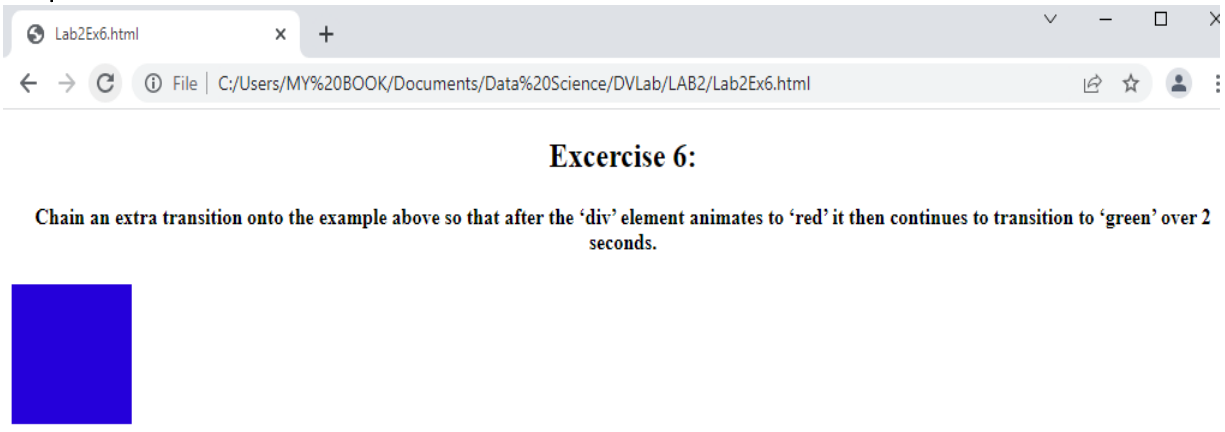
Part 3. D3 Transitions

Exercise 6: Chain an extra transition onto the example above so that after the 'div' element animates to 'red' it then continues to transition to 'green' over 2 seconds.

Snippet:

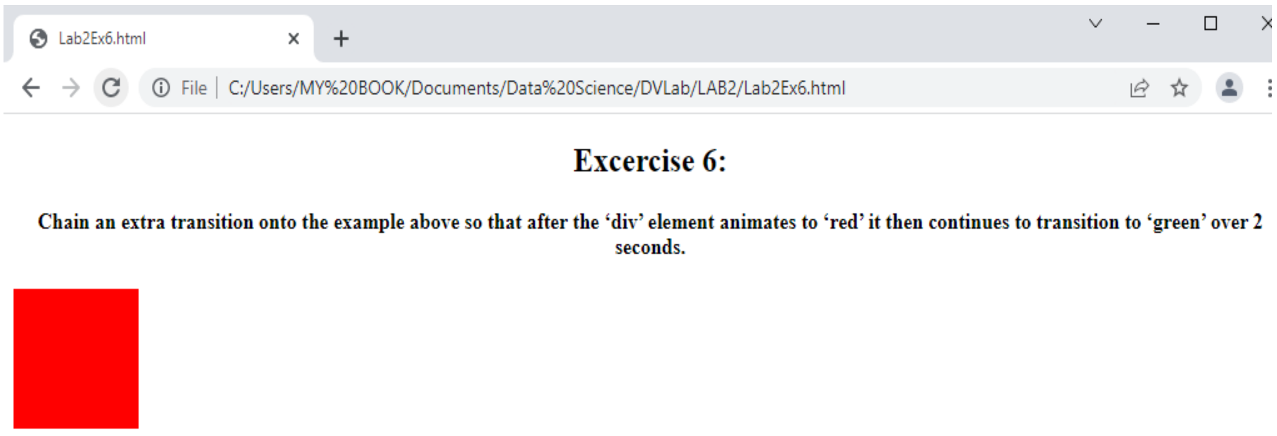
```
      <script>//selecting the body and appending div
        d3.select('body')
        .append("div")
        .style('width', '100px')
        .style('height', '100px')
        .style('background-color', 'blue')
        .transition() //setting transition duration to 1000ms
        .duration(1000)
        .style("background-color", "red") //transistion to red after 1 sec
        .transition()
        .duration(2000)
        .style("background-color", "green")//transistion to green after 2 sec
        .transition()
        .duration(3000)
        .style("background-color", "black")//transistion to black after 3 sec
        .transition()
        .duration(4000)
        .style("background-color", "pink");//transistion to pink after 4 sec
      </script>
```

Output:


Lab2Ex6.html
File
C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex6.html

Excercise 6:

Chain an extra transition onto the example above so that after the 'div' element animates to 'red' it then continues to transition to 'green' over 2 seconds.

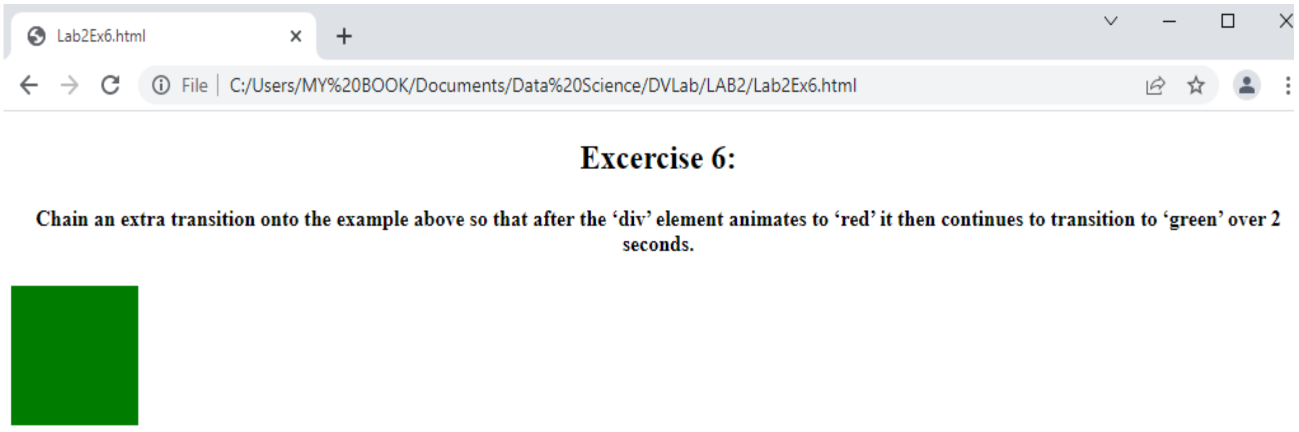


Lab2Ex6.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex6.html

Excercise 6:

Chain an extra transition onto the example above so that after the 'div' element animates to 'red' it then continues to transition to 'green' over 2 seconds.



Lab2Ex6.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex6.html

Excercise 6:

Chain an extra transition onto the example above so that after the 'div' element animates to 'red' it then continues to transition to 'green' over 2 seconds.

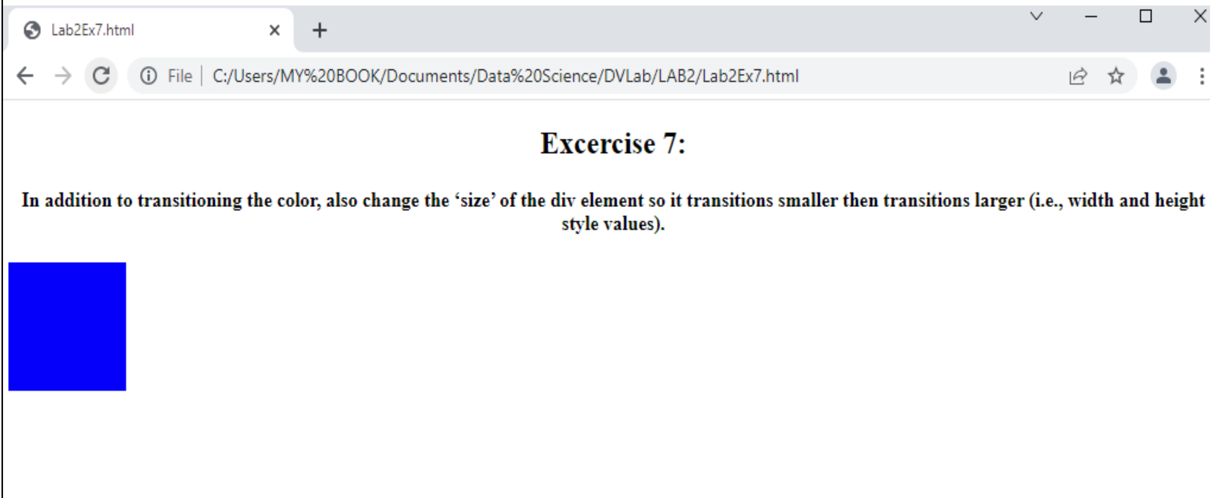
Exercise 7:

In addition to transitioning the color, also change the ‘size’ of the div element so it transitions smaller then transitions larger (i.e., width and height style values).

Snippet:

Output:

N

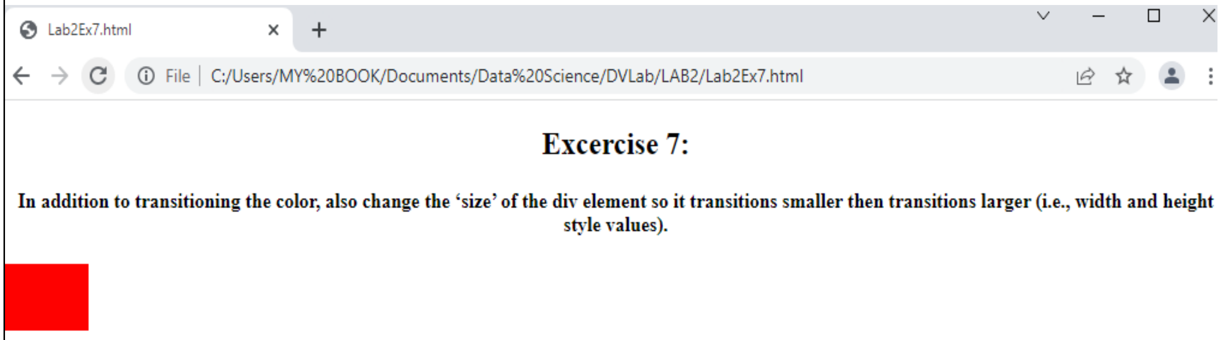


Lab2Ex7.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex7.html

Excercise 7:

In addition to transitioning the color, also change the ‘size’ of the div element so it transitions smaller then transitions larger (i.e., width and height style values).

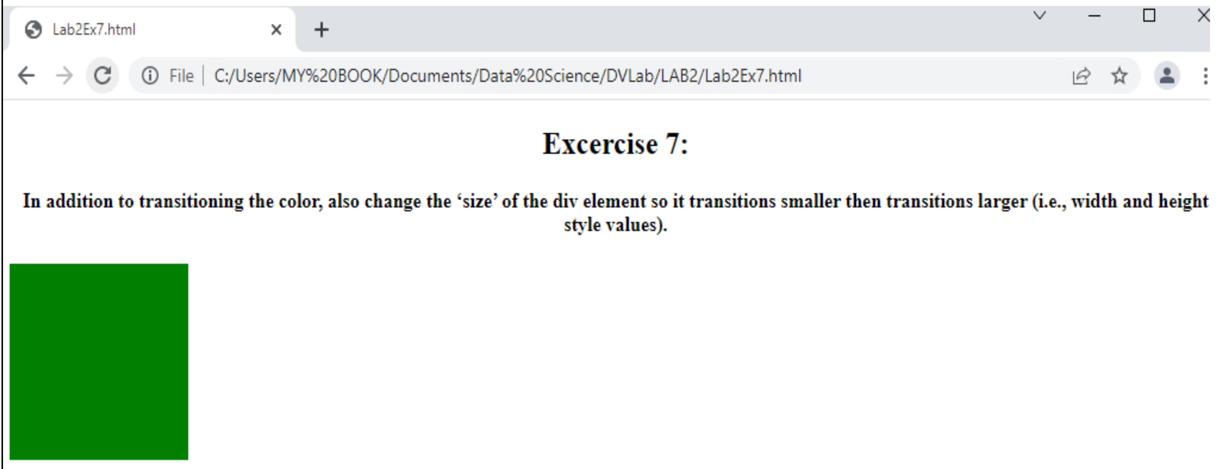


Lab2Ex7.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex7.html

Excercise 7:

In addition to transitioning the color, also change the ‘size’ of the div element so it transitions smaller then transitions larger (i.e., width and height style values).



Lab2Ex7.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex7.html

Excercise 7:

In addition to transitioning the color, also change the ‘size’ of the div element so it transitions smaller then transitions larger (i.e., width and height style values).

Part 4. Data Binding

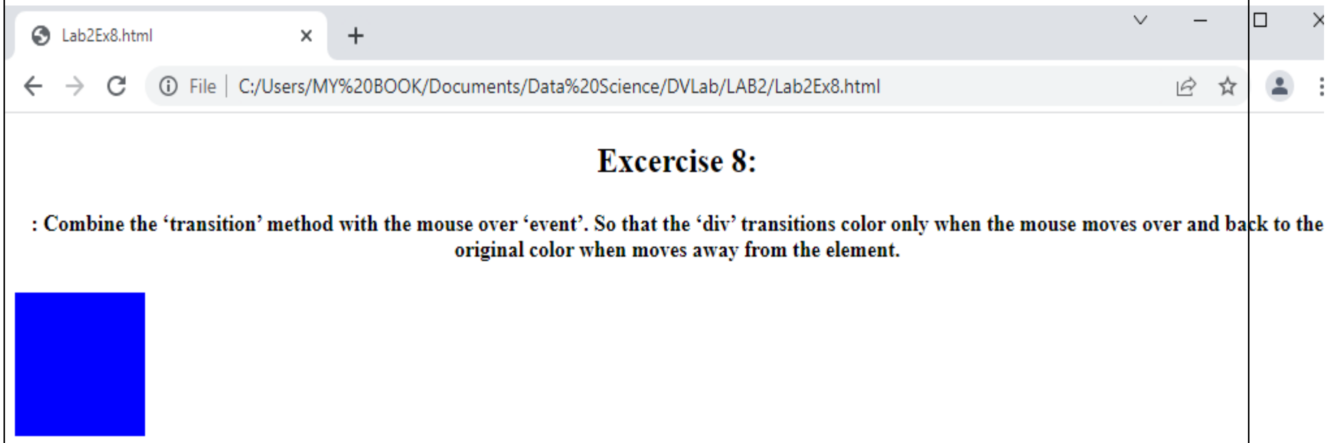
Exercise 8:Combining the ‘transition’ method with the mouse over ‘event’. So that the ‘div’ transitions color only when the mouse moves over and back to the original color when moves away from the element.

Snippet:

```
<script>
        d3.select('body')
        .append("div")
        .style('width', '100px')
        .style('height', '100px')
        .style('background-color', 'blue')
        d3.selectAll("div")
        //smooth transition of color, border-style, width and height
        //of div on onmouseover and mouseout events
         .on("mouseover", function(event){//
            d3.select(this)
            .transition()
            .duration(1000)
            .style('width', '150px')
            .style('height', '150px')
            .style('border-style','inset')
            .style("background-color", "red") })
        //changing color, border-style, width and height during mouse out event
        .on("mouseout", function(){
            d3.select(this)
            .transition()
            .duration(2000)
            .style('width', '250px')
            .style('height', '90px')
            .style('border-style','dashed')
            .style("background-color", "green")
            });
      </script>
```

Output:

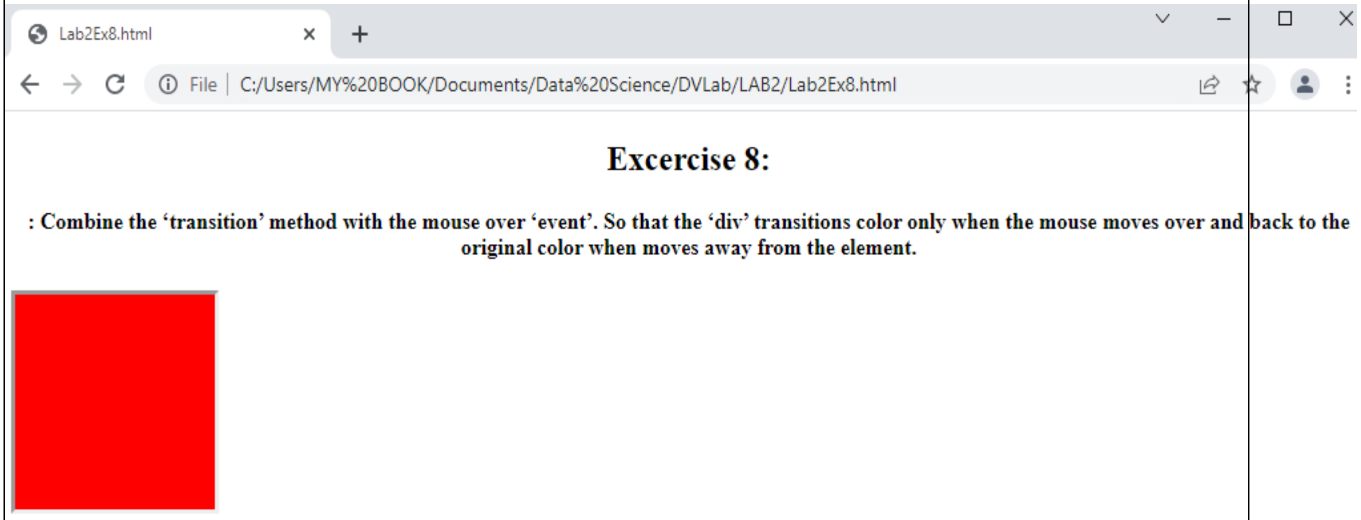
N


Lab2Ex8.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex8.html

Excercise 8:

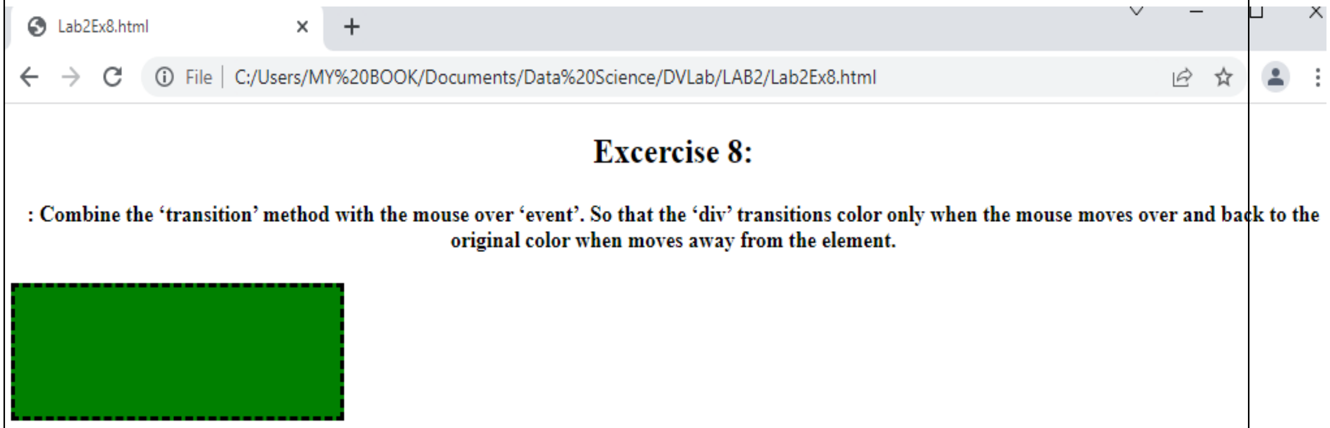
: Combine the ‘transition’ method with the mouse over ‘event’. So that the ‘div’ transitions color only when the mouse moves over and back to the original color when moves away from the element.


Lab2Ex8.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex8.html

Excercise 8:

: Combine the ‘transition’ method with the mouse over ‘event’. So that the ‘div’ transitions color only when the mouse moves over and back to the original color when moves away from the element.


Lab2Ex8.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex8.html

Excercise 8:

: Combine the ‘transition’ method with the mouse over ‘event’. So that the ‘div’ transitions color only when the mouse moves over and back to the original color when moves away from the element.

Exercise 9:

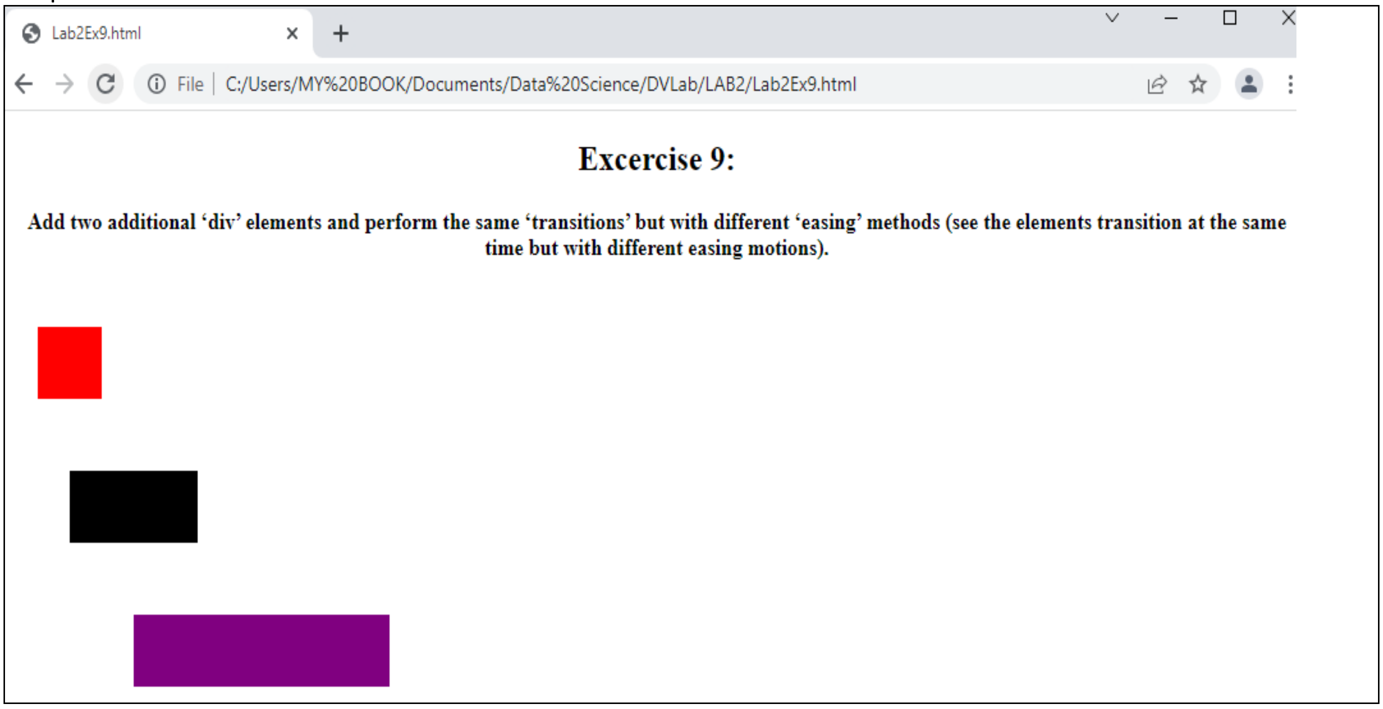
Add two additional 'div' elements and perform the same 'transitions' but with different 'easing' methods (see the elements transition at the same time but with different easing motions).

Snippet:

```
div1.style('transform', 'scale(1.0)')
             .transition()
             .ease(d3.easeLinear)// adding easeLinear transition to div1
             .duration(1000)
             .style("background-color", "red")
             .style('transform', 'scale(0.5)')

div2.style('transform', 'scale(1.0)')
             .transition()
             .ease( d3.easeCubicInOut )// adding easeCubic transition to div2
             .duration(1000)
             .style("background-color", "black")
             .style('transform', 'scale(0.5)')

div3.style('transform', 'scale(1.0)')
             .transition()
             .ease( d3.easeElasticIn ) // adding easeElastic transition to div3
             .duration(1000)
             .style("background-color", "purple")
             .style('transform', 'scale(0.5)')
```

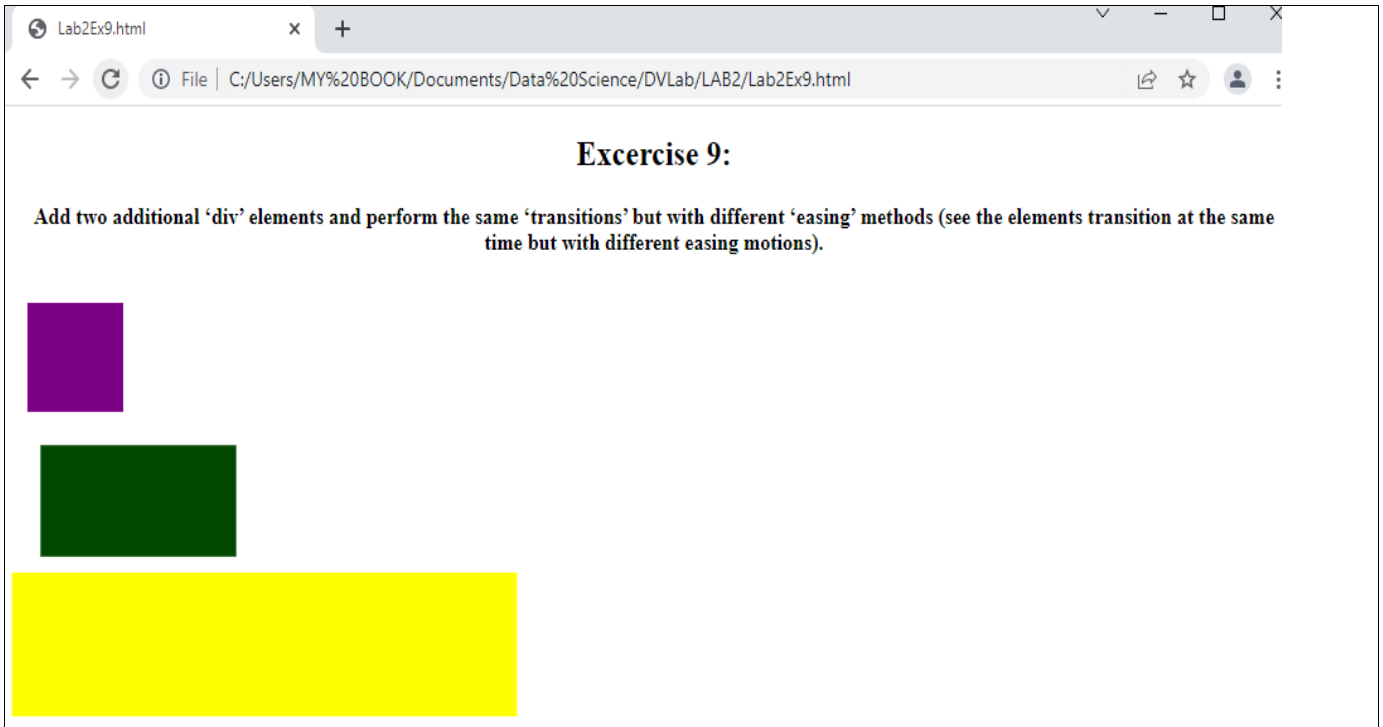
```
       }
```

Output:


Lab2Ex9.html
File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex9.html

Excercise 9:

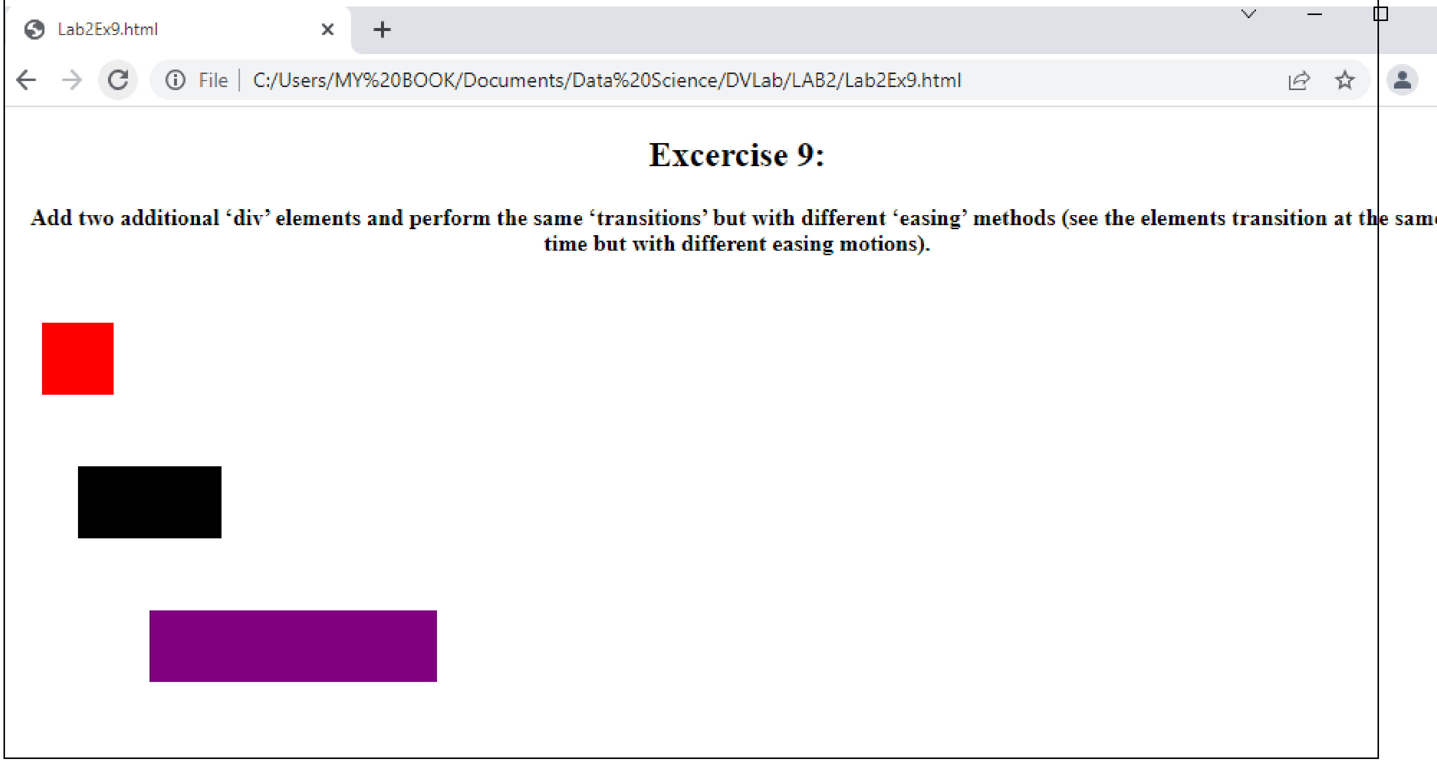
Add two additional 'div' elements and perform the same 'transitions' but with different 'easing' methods (see the elements transition at the same time but with different easing motions).


Lab2Ex9.html

C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex9.html

Excercise 9:

Add two additional 'div' elements and perform the same 'transitions' but with different 'easing' methods (see the elements transition at the same time but with different easing motions).


Lab2Ex9.html

C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex9.html

Excercise 9:

Add two additional 'div' elements and perform the same 'transitions' but with different 'easing' methods (see the elements transition at the same time but with different easing motions).

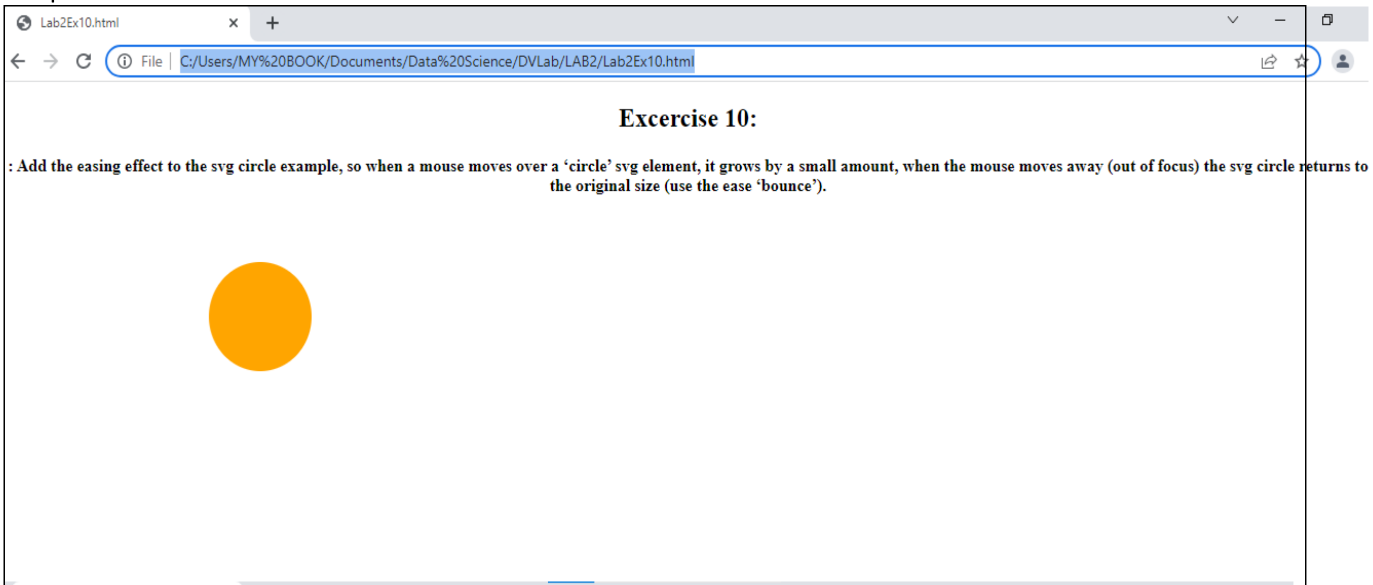
Exercise 10:

Add the easing effect to the svg circle example, so when a mouse moves over a ‘circle’ svg element, it grows by a small amount, when the mouse moves away (out of focus) the svg circle returns to the original size (use the ease ‘bounce’).

Snippet:

```
//Append circle
        svg.append("circle")
         .attr("fill","orange")
         .attr("cx", 250)
         .attr("cy", 90)
         .attr("r", 50)
         d3.selectAll("circle")

        .on("mouseover", function(event){
            d3.select(this)
            .transition()
            .ease( d3.easeBounceIn)//performing easeBounceIn transition on 
mouseover event
            .duration(1000)
            .attr("fill","yellow")
            .attr("cx", 250)
            .attr("cy", 90)
            .attr("r", 75)})// transition into bigger circle on mouseover event

        .on("mouseout", function(){
            d3.select(this)
            .transition()
            .ease( d3.easeBounceOut) //performing easeBounceOut transition on 
mouseout event
            .duration(2000)
            .attr("fill","orange")
            .attr("cx", 250)
            .attr("cy", 90)
            .attr("r", 50)}) // transition into original size on mouseout event
         ;
      </script>
```

Output:

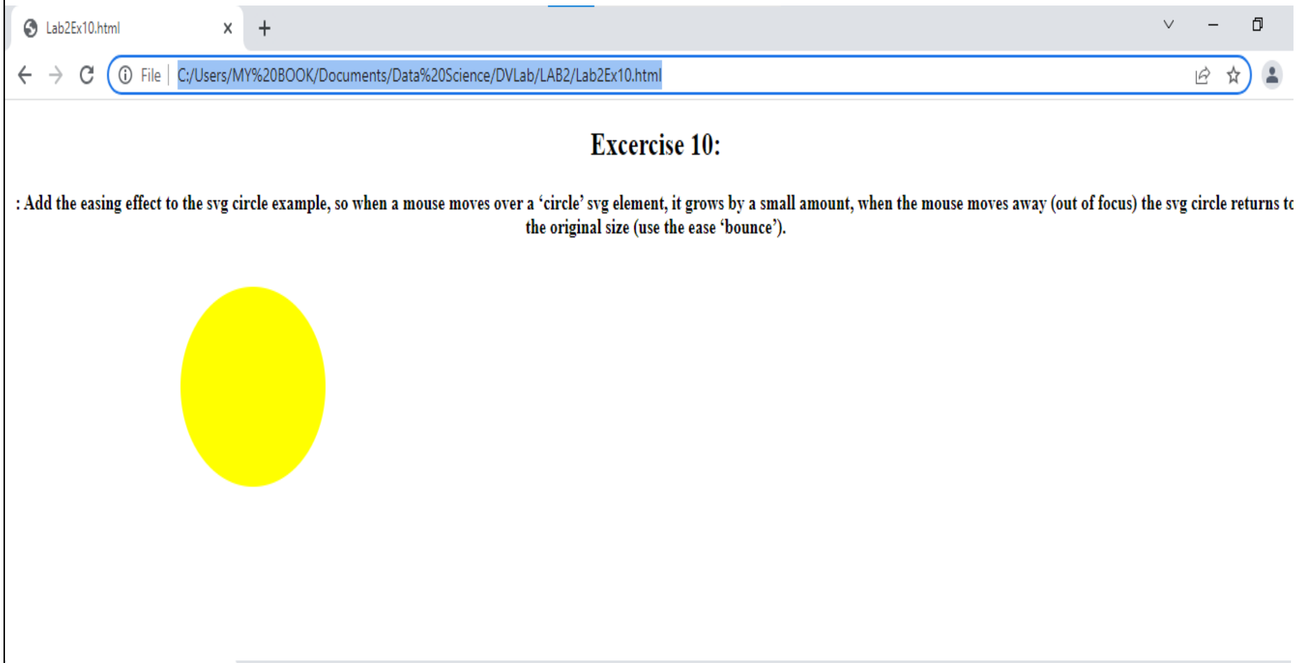


Lab2Ex10.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex10.html

Excercise 10:

: Add the easing effect to the svg circle example, so when a mouse moves over a ‘circle’ svg element, it grows by a small amount, when the mouse moves away (out of focus) the svg circle returns to the original size (use the ease ‘bounce’).

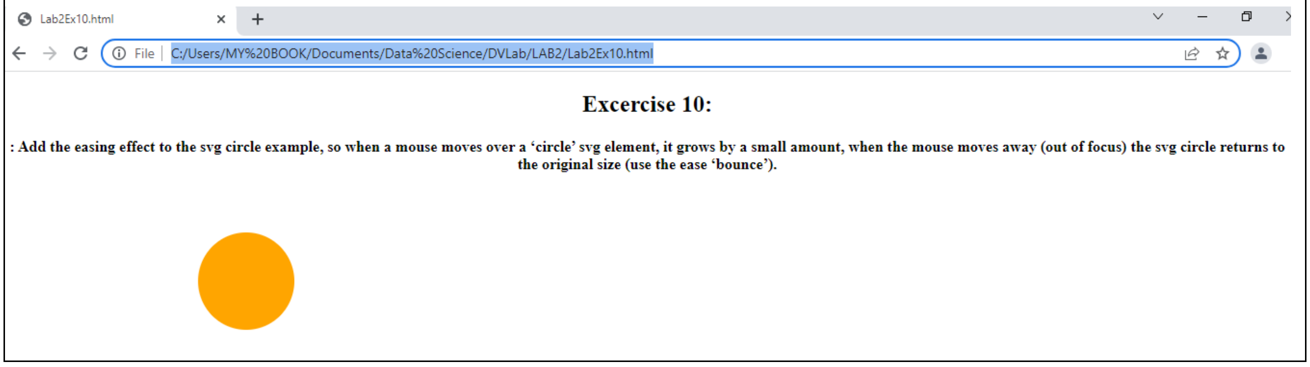


Lab2Ex10.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex10.html

Excercise 10:

: Add the easing effect to the svg circle example, so when a mouse moves over a ‘circle’ svg element, it grows by a small amount, when the mouse moves away (out of focus) the svg circle returns tc the original size (use the ease ‘bounce’).



Lab2Ex10.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex10.html

Excercise 10:

: Add the easing effect to the svg circle example, so when a mouse moves over a ‘circle’ svg element, it grows by a small amount, when the mouse moves away (out of focus) the svg circle returns to the original size (use the ease ‘bounce’).

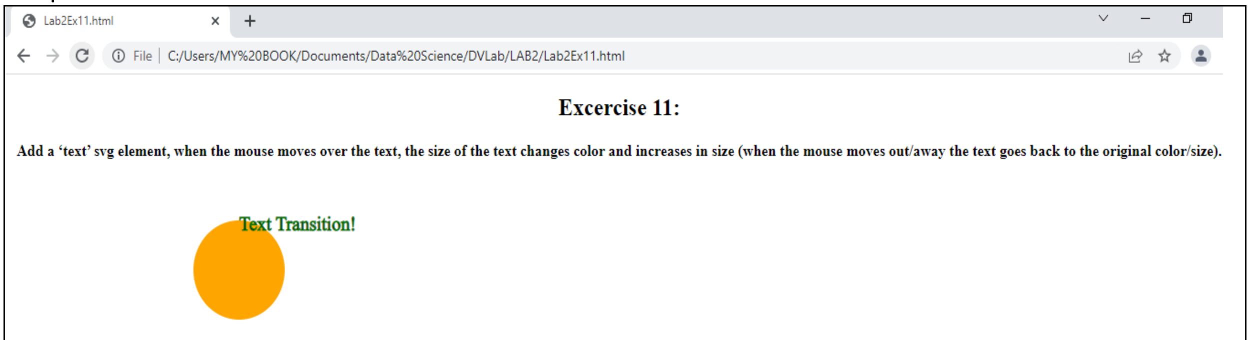
Exercise 11:

Add a ‘text’ svg element, when the mouse moves over the text, the size of the text changes color and increases in size (when the mouse moves out/away the text goes back to the original color/size)

Snippet:

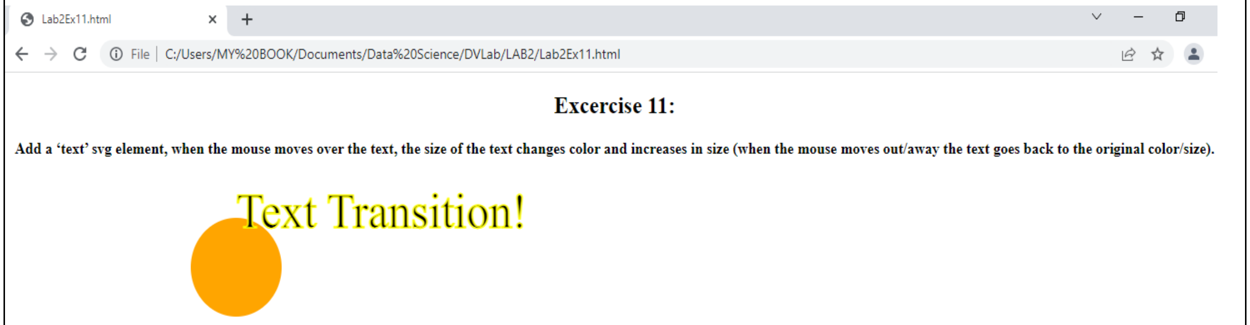
```
      //Create and append text element into group
     svg.append("text")
     .attr("x", 250)
     .attr("y", 50)
     .attr("stroke", "green")
     .text("Text Transition!")
     .attr("font-size","20px")

      .on("mouseover", function(event){
        d3.selectAll("text")
        .transition()
        .ease( d3.easeElasticIn)//performing easeElasticIn transition on text
        .duration(1000)
        .attr("x", 250)
        .attr("y", 50)
        .attr("stroke", "yellow")
        .attr("font-size","50px")})

      .on("mouseout", function(){
        d3.select(this)
        .transition()
        .ease( d3.easeBackInOut) //performing easeBackInOut transition on text
        .duration(2000)
        .attr("x", 150)
        .attr("y", 50)
        .attr("stroke", "purple")
        .attr("font-size","20px")})
     ;
</script>
```

Output:


Lab2Ex11.html
File
C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex11.html

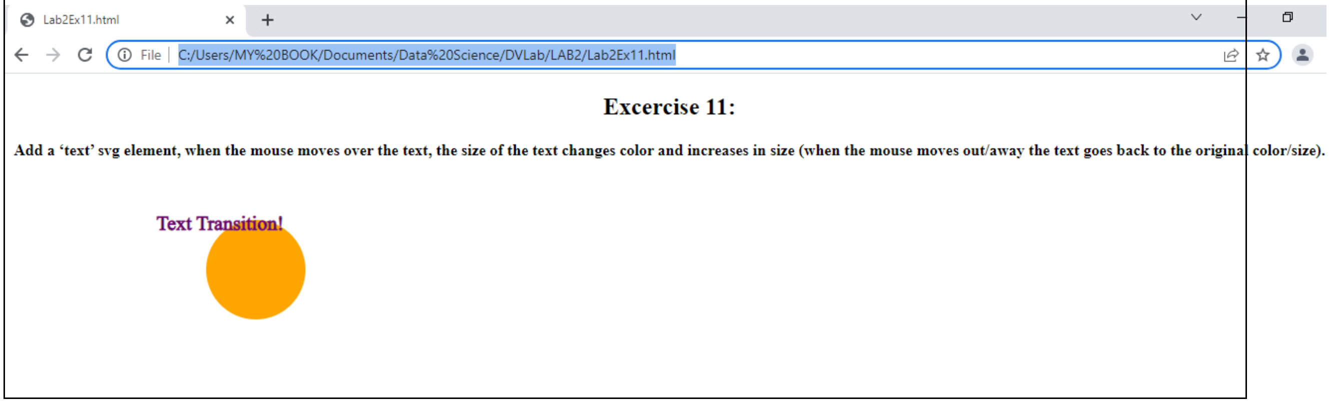
Excercise 11:

Add a 'text' svg element, when the mouse moves over the text, the size of the text changes color and increases in size (when the mouse moves out/away the text goes back to the original color/size).


Lab2Ex11.html
File
C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex11.html

Excercise 11:

Add a 'text' svg element, when the mouse moves over the text, the size of the text changes color and increases in size (when the mouse moves out/away the text goes back to the original color/size).


Lab2Ex11.html
File
C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex11.html

Excercise 11:

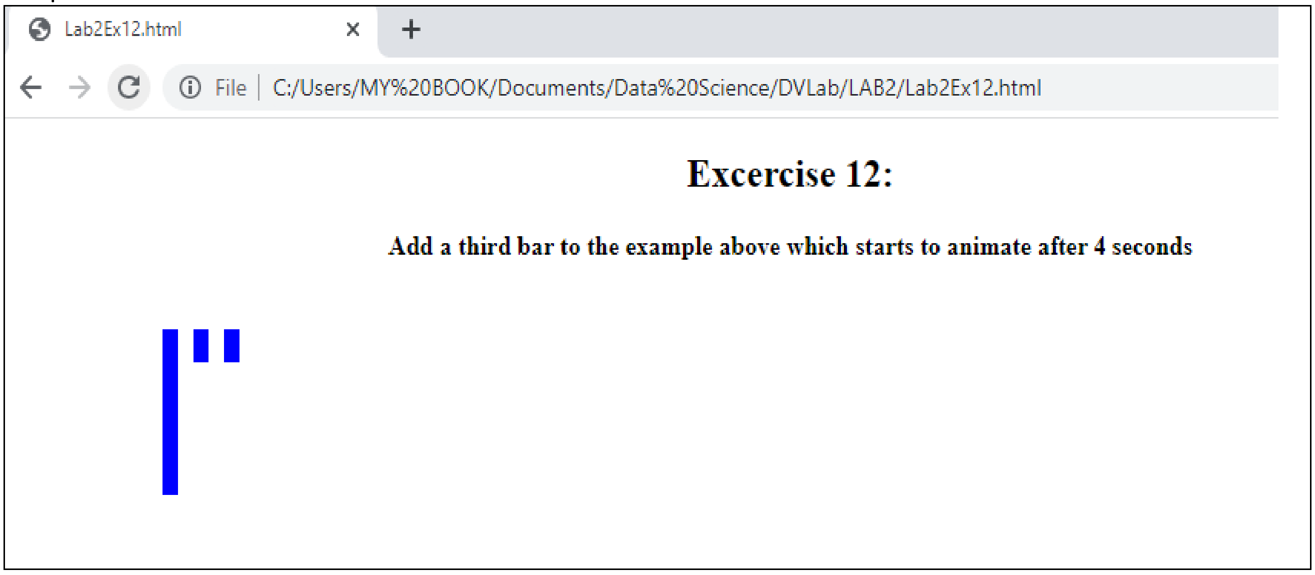
Add a 'text' svg element, when the mouse moves over the text, the size of the text changes color and increases in size (when the mouse moves out/away the text goes back to the original color/size).

Exercise 12:
Add a third bar to the example above which starts to animate after 4 seconds

Snippet:

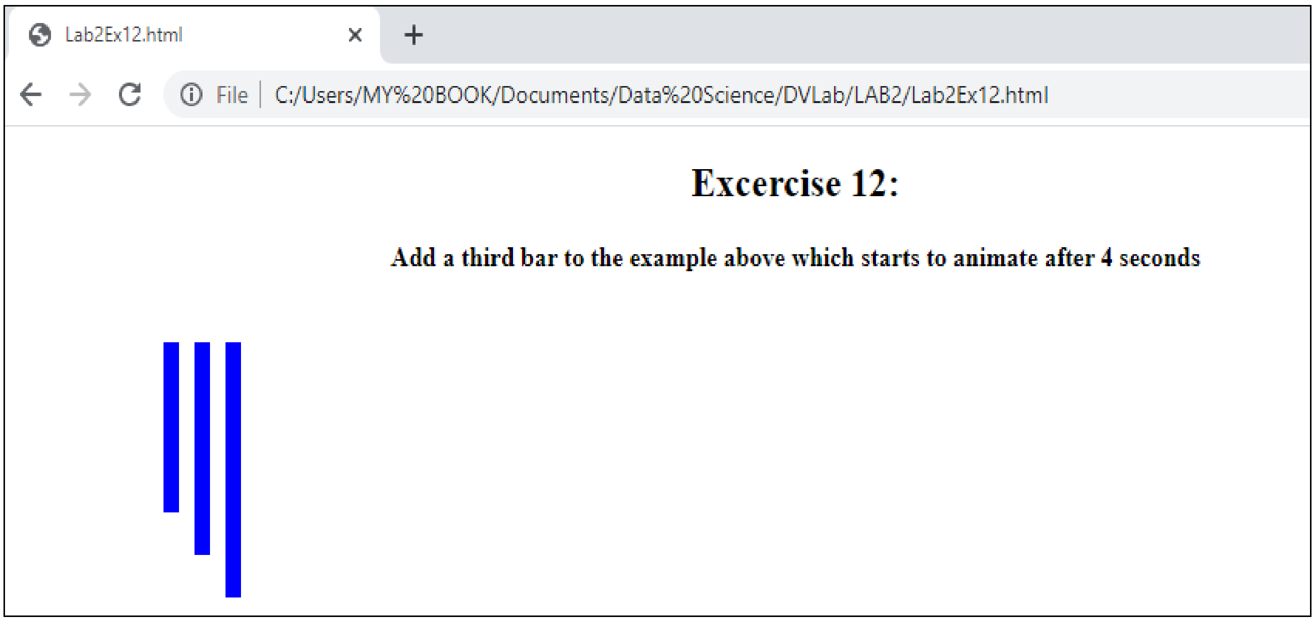
```
//function to perform transition of 3 bars
        function update() {
               bar1.transition()
               .ease(d3.easeLinear) //performing easeLinear transition on bar1
               .duration(1000)
               .attr("height",100)

               bar2.transition()
               .ease(d3.easeLinear)//performing easeLinear transition on bar2
               .duration(2000)
               .delay(2000) //transition after delay of 2 s
               .attr("height",125)

               bar3.transition()
               .ease(d3.easeLinear) //performing easeLinear transition on bar3
               .duration(3000)
               .delay(6000) //transition after delay of 6 s
               .attr("height",150)

                   }
```

Output:


Lab2Ex12.html
File
C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex12.html

Excercise 12:

Add a third bar to the example above which starts to animate after 4 seconds



Lab2Ex12.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex12.html

Excercise 12:

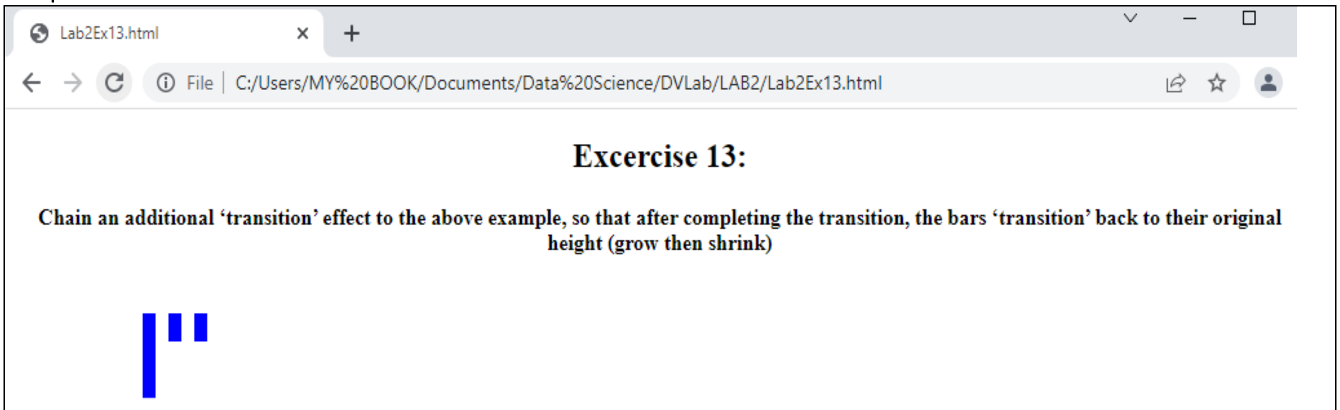
Add a third bar to the example above which starts to animate after 4 seconds

Exercise 13:
Chain an additional 'transition' effect to the above example, so that after completing the transition, the bars 'transition' back to their original height (grow then shrink).

Snippet:

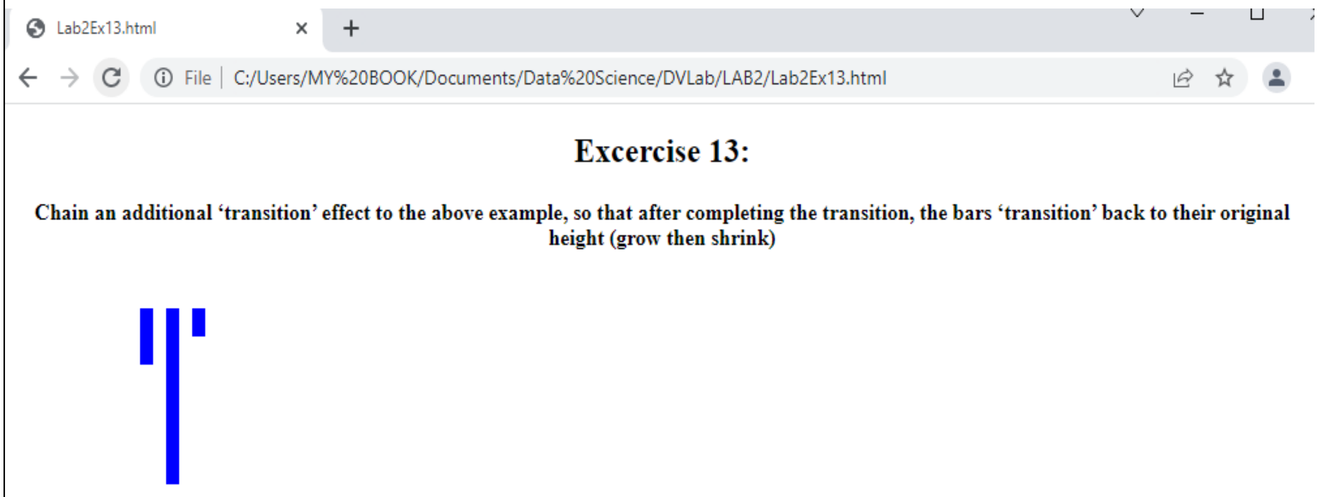
```
         function update() {
                bar1.transition()
                .ease(d3.easeLinear) //performing easeLinear transition on bar1
                .duration(2000)
                .attr("height",100)
                bar1.transition()
                .duration(2000)
                .delay(1000)
                .attr("height",40) //transitioning back to original height of
bar1
                bar2.transition()
                .ease(d3.easeLinear) //performing easeLinear transition on bar1
                .duration(2000)
                .delay(2000)
                .attr("height",125)
                bar2.transition()
                .duration(2000)
                .delay(4000)
                .attr("height",40) //transitioning back to original height of
bar1
                bar3.transition()
                .ease(d3.easeLinear)//performing easeLinear transition on bar2
                .duration(3000)
                .delay(6000)
                .attr("height",150)
                bar3.transition()
                .duration(2000)
                .delay(8000)
                .attr("height",40)//transitioning back to original height of bar3
                    }
```

Output:


Lab2Ex13.html
File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex13.html

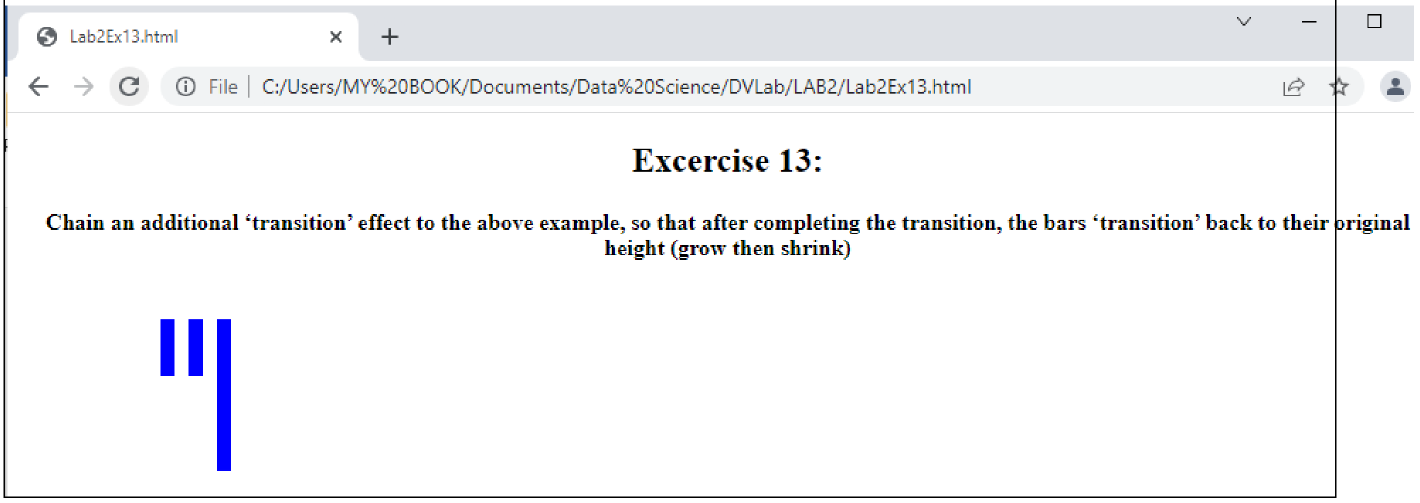
Excercise 13:

Chain an additional 'transition' effect to the above example, so that after completing the transition, the bars 'transition' back to their original height (grow then shrink)


Lab2Ex13.html
File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex13.html

Excercise 13:

Chain an additional 'transition' effect to the above example, so that after completing the transition, the bars 'transition' back to their original height (grow then shrink)


Lab2Ex13.html
File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex13.html

Excercise 13:

Chain an additional 'transition' effect to the above example, so that after completing the transition, the bars 'transition' back to their original height (grow then shrink)

Part 7. Bar Chart

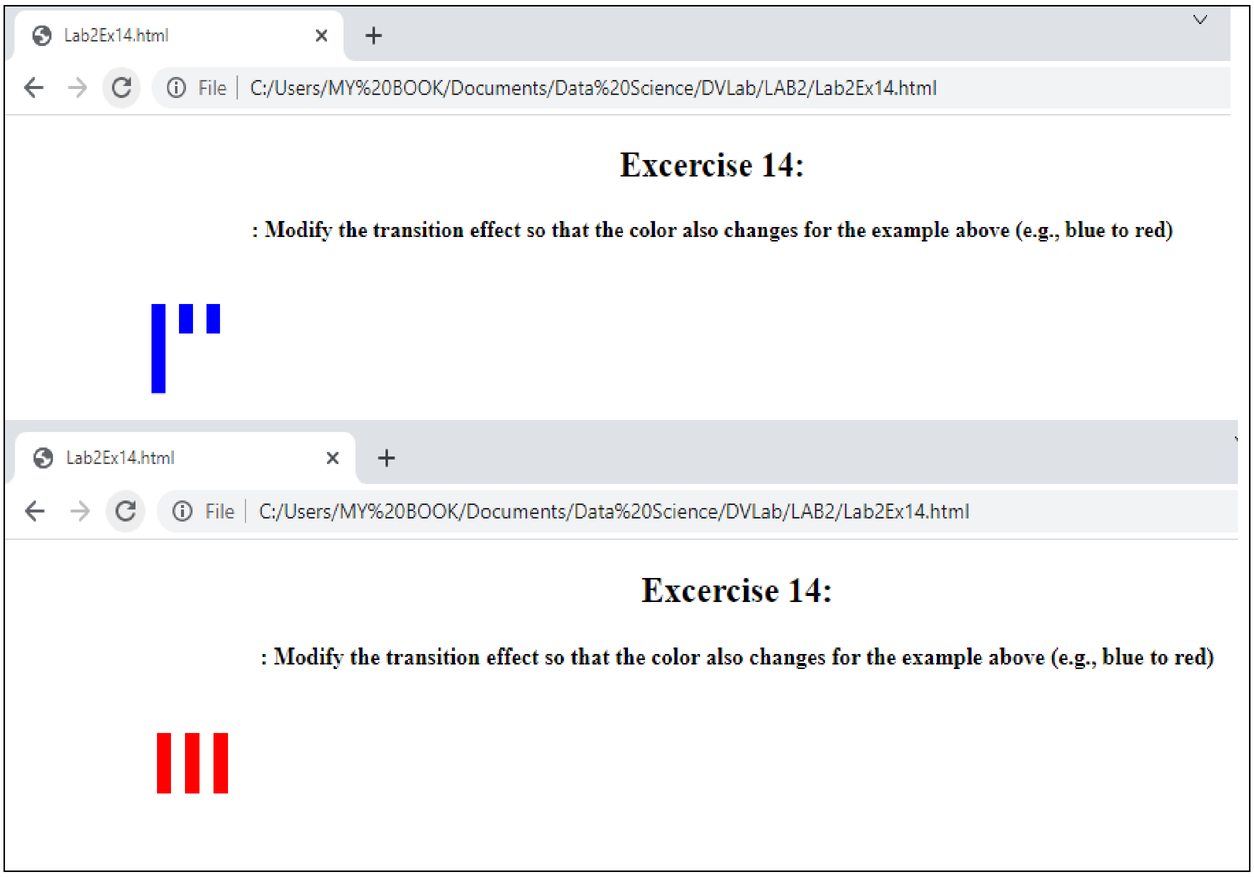
Exercise 14:

Modify the transition effect so that the color also changes for the example above (e.g., blue to red

Snippet:

```
function update() {
                bar1.transition()
                .ease(d3.easeLinear) //performing easeLinear transition on
bar1
                .duration(2000)
                .attr("height",100)
                bar1.transition()
                .duration(2000)
                .delay(1000)
                .attr("fill", "red") //changing color of bar1 to red from blue
                .attr("height",40)
                bar2.transition()
                .ease(d3.easeLinear)//performing easeLinear transition on bar2
                .duration(2000)
                .delay(2000)
                .attr("height",125)
                bar2.transition()
                .duration(2000)
                .delay(4000)
                .attr("fill", "red") //changing color of bar2 to red from blue
                .attr("height",40)
                bar3.transition()
                .ease(d3.easeLinear)//performing easeLinear transition on bar3
                .duration(3000)
                .delay(6000)
                .attr("height",150)
                bar3.transition()
                .duration(2000)
                .delay(8000)
                .attr("fill", "red")//changing color of bar3 to red from blue
                .attr("height",40)
                    }
```

Output:



Lab2Ex14.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex14.html

Excercise 14:

: Modify the transition effect so that the color also changes for the example above (e.g., blue to red)

Lab2Ex14.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex14.html

Excercise 14:

: Modify the transition effect so that the color also changes for the example above (e.g., blue to red)

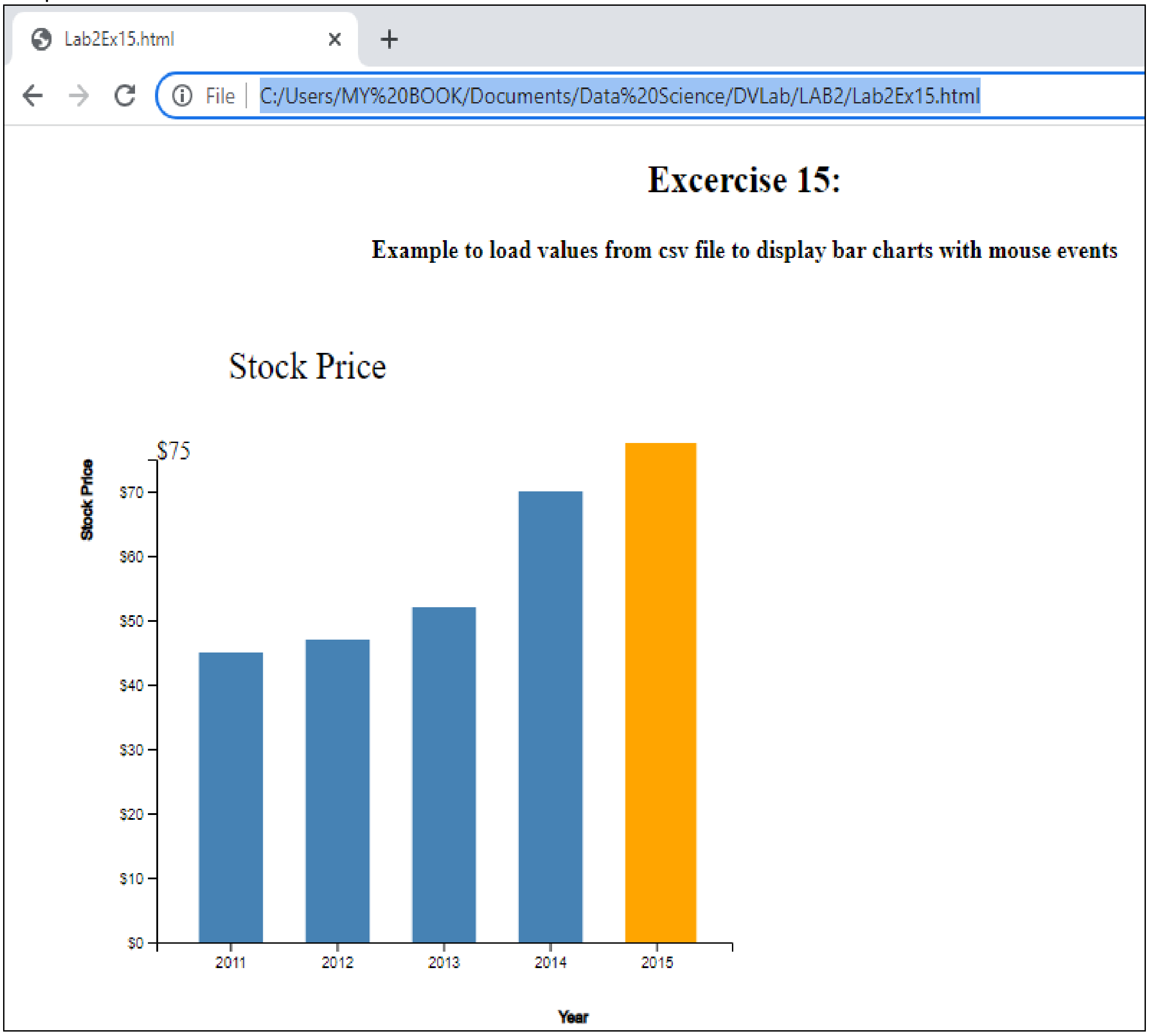
Part 4. Animated Chart

Exercise 15:

Load values from csv file to display values in bar charts based on mouse events

CSV file used: `"https://raw.githubusercontent.com/ankithacherian/DVLab2/main/data.csv";`

Output:



Lab2Ex15.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex15.html

Excercise 15:

Example to load values from csv file to display bar charts with mouse events

Exercise 16:

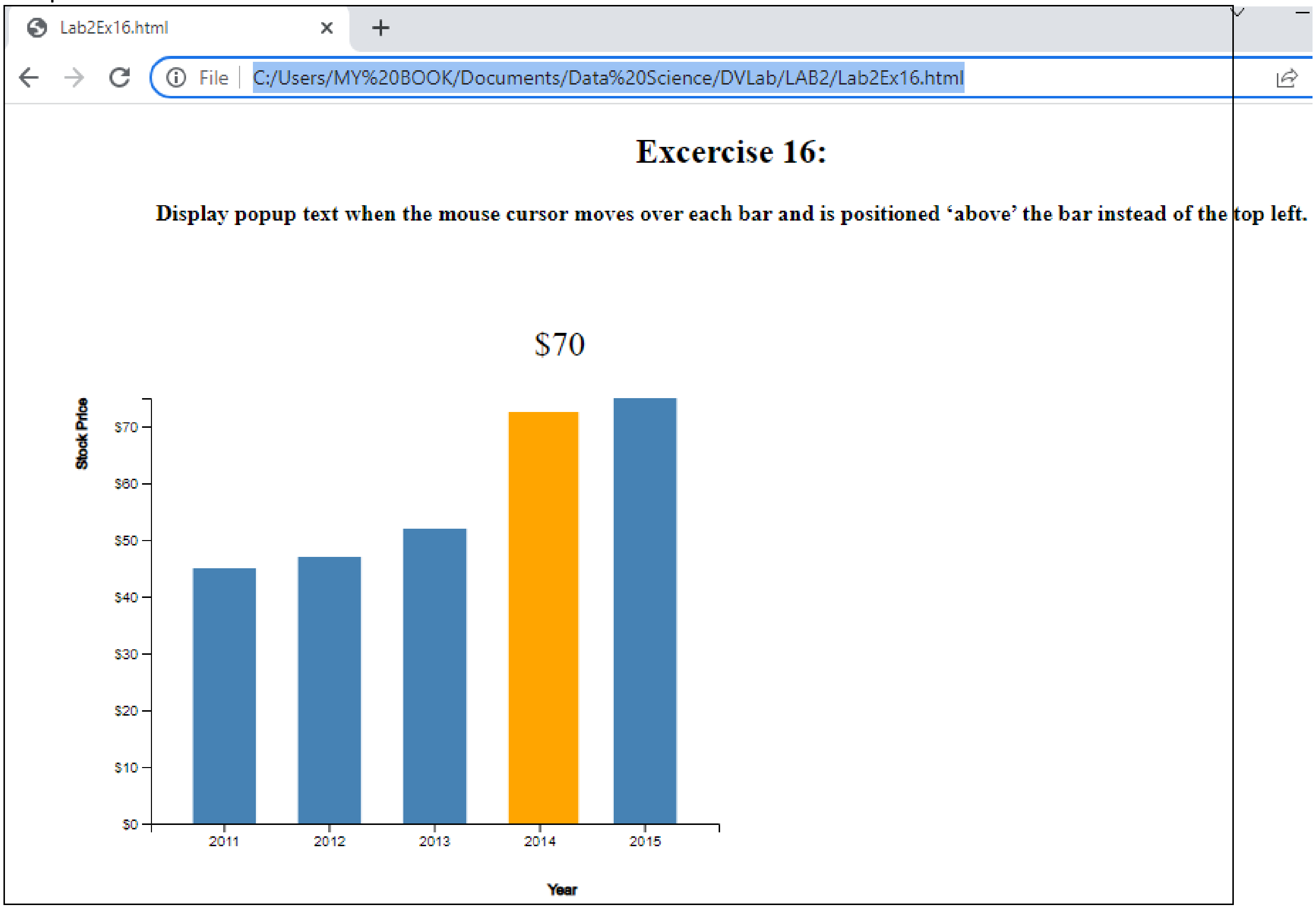
Modify the example so the popup text that is displayed when the mouse cursor moves over each bar is positioned ‘above’ the bar instead of the top left.

CSV file: `"https://raw.githubusercontent.com/ankithacherian/DVLab2/main/data.csv";`

Snippet:

```
 //mouseover event handler function
        function onMouseOver(d, i) {
          var pos = d3.pointer(this);
          d3.select(this).attr("class", "highlight");
          d3.select(this)
            .transition() // adds animation
            .duration(400)
            .attr("width", x.bandwidth() + 5)
            .attr("y", function (d) {
              return y(d.value) - 10;
            })
            .attr("height", function (d) {
              return height - y(d.value) + 10;
            });

          d3.select("text")
            //adding x and y coordinates to display text value on top of
horixontal bar
            .attr("y", function (d) {
              return y(i.value) + 50
            })
            .attr("x", function (d) {
              return height- y(i.value) -10 ;
            })
            .style("pointer-events", "none") //disabling other pointer events
            .text(function (d) {
              return "$" + i.value;
            }); // Value of the text
        }

        //mouseout event handler function
        function onMouseOut(d, i) {
          // use the text label class to remove label on mouseout
          d3.select(this).attr("class", "bar");
          d3.select(this)
            .transition() // adds animation
            .duration(400)
            .attr("width", x.bandwidth())
            .attr("y", function (d) {
              return y(i.value);
            })
            .attr("height", function (d) {
              return height - y(i.value);
            });
          d3.selectAll(".val").remove();
        }
```

Output:


Lab2Ex16.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex16.html

Excercise 16:

Display popup text when the mouse cursor moves over each bar and is positioned ‘above’ the bar instead of the top left.

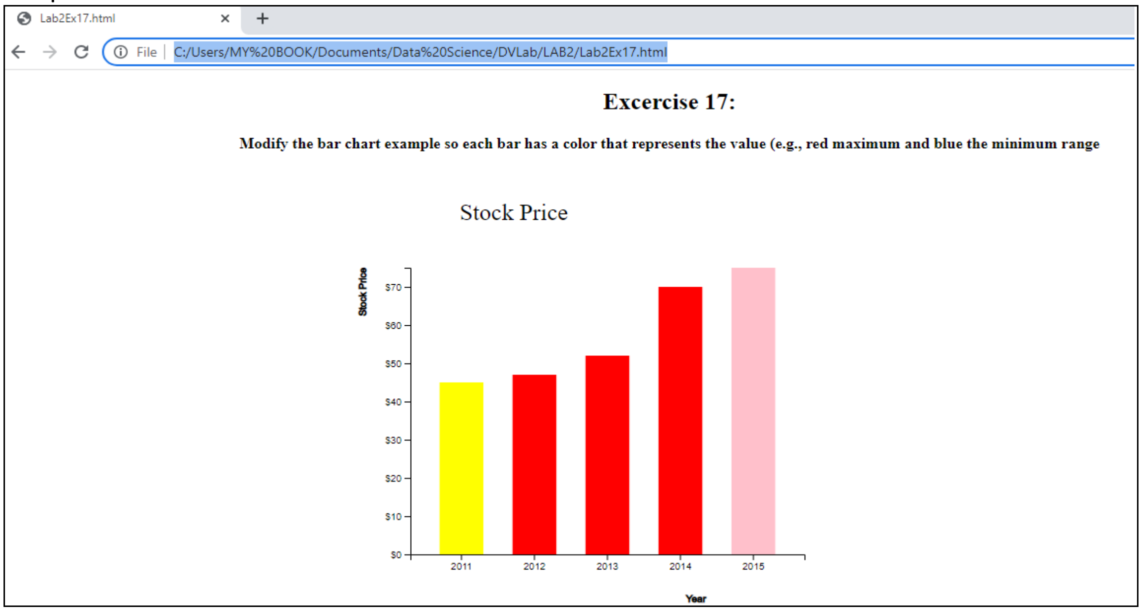
Exercise 17:

Modify the example above so the bars are green if below 100 and red if above 500.

Snippet:

```
g.selectAll(".bar")
            .data(data)
            .enter().append("rect")
            .attr("class", "bar")

            //using d3.max() and d3.min() to display bars in different colors
            .attr("fill", function (d) {
               var dmax = d3.max(data, function (d) { return +d.value; })
               var dmin = d3.min(data, function (d) { return +d.value; })
               if (d.value == dmax) {
                  return 'pink';

               } else if (d.value == dmin) {
                  return 'yellow';

               } else {
                  return 'red'
               }
            })
            .on("mouseover", onMouseOver)
            .on("mouseout", onMouseOut)
```

Output:



Lab2Ex17.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex17.html

Excercise 17:

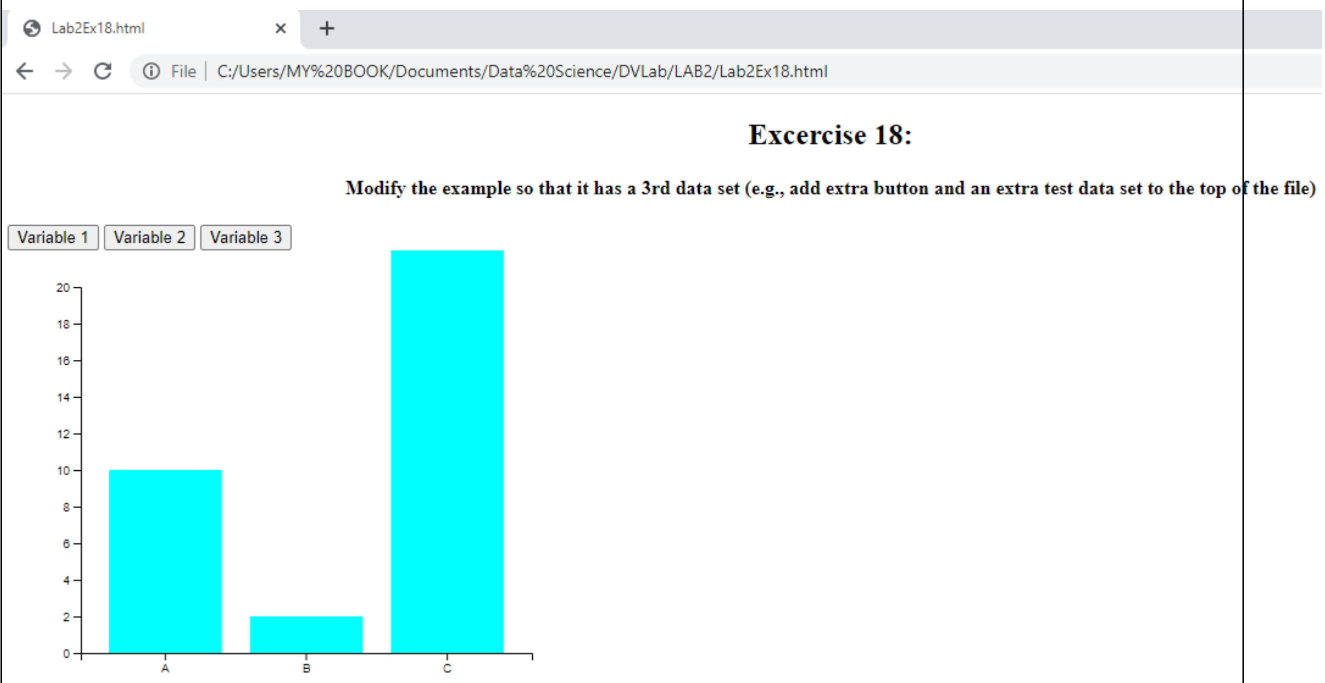
Modify the bar chart example so each bar has a color that represents the value (e.g., red maximum and blue the minimum range

Part 5. Changing Data and Transitioning

Exercise 18:

Modify the example so that it has a 3rd data set (e.g., add extra button and an extra test data set to the top of the file)

Output:

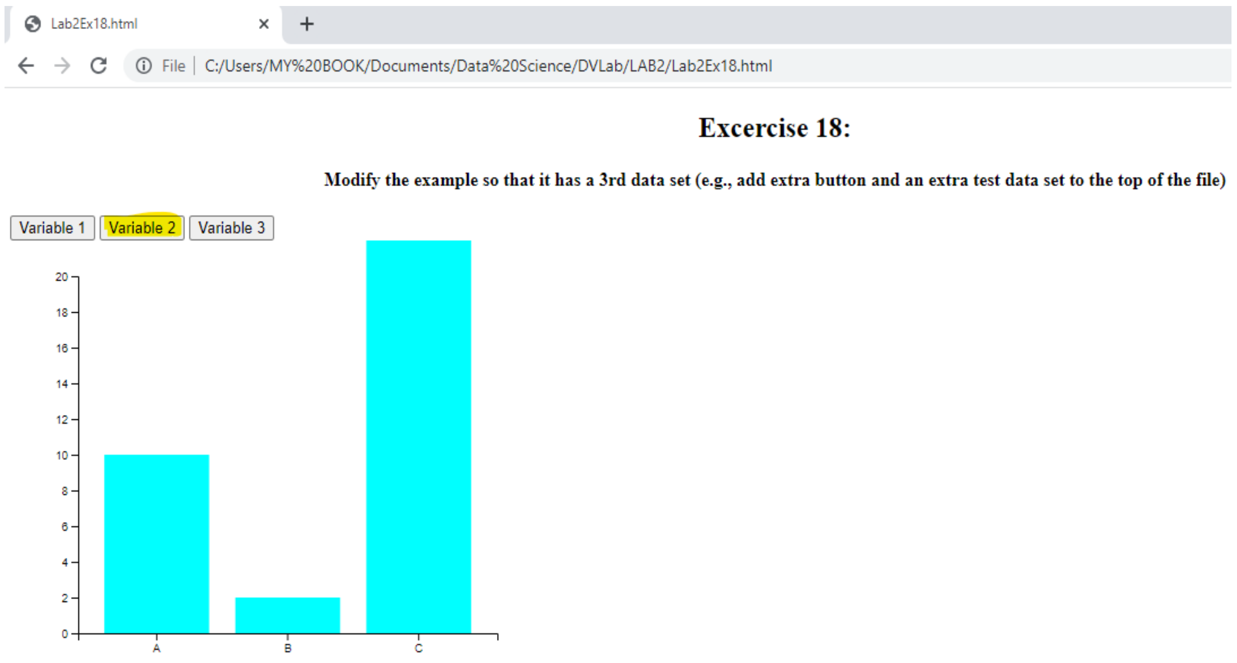


Lab2Ex18.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex18.html

Excercise 18:

Modify the example so that it has a 3rd data set (e.g., add extra button and an extra test data set to the top of the file)

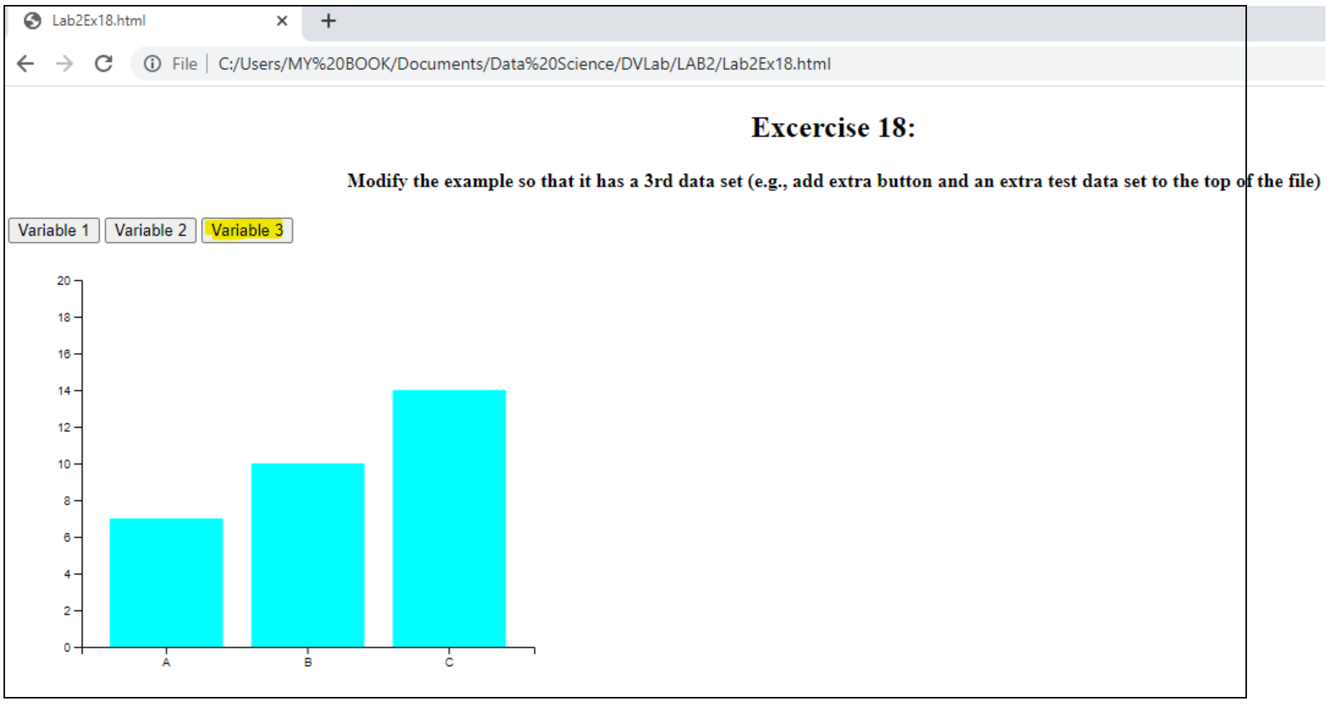


Lab2Ex18.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex18.html

Excercise 18:

Modify the example so that it has a 3rd data set (e.g., add extra button and an extra test data set to the top of the file)


Lab2Ex18.html

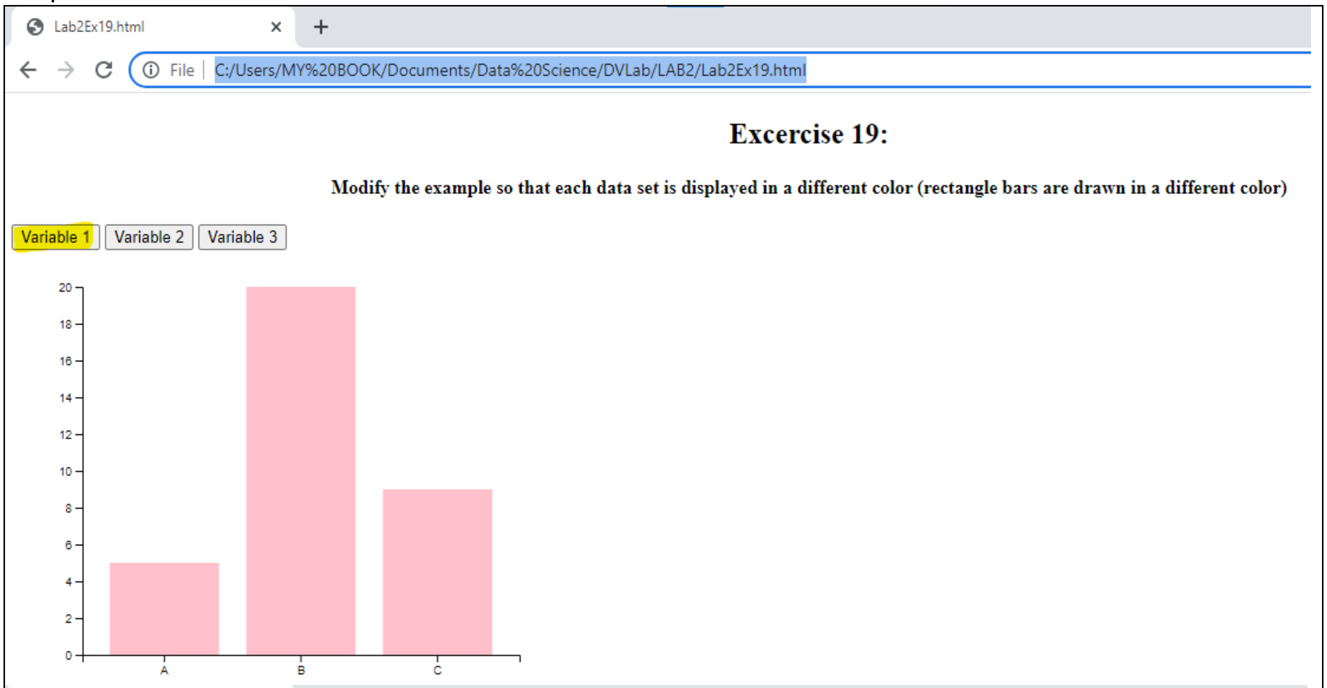
C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex18.html

Excercise 18:

Modify the example so that it has a 3rd data set (e.g., add extra button and an extra test data set to the top of the file)

Exercise 19:
Modify the example so that each data set is displayed in a different colour (rectangle bars are drawn in a different color.

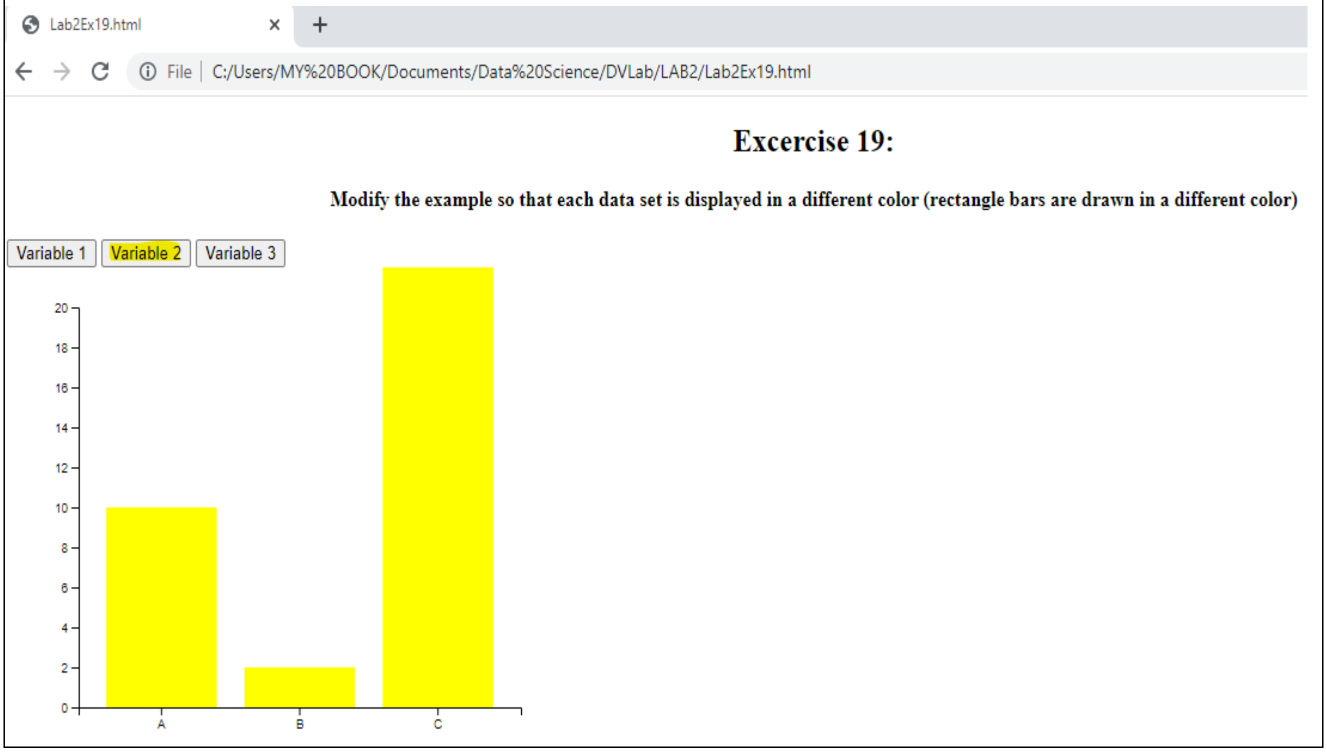
Output:



C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex19.html

Excercise 19:

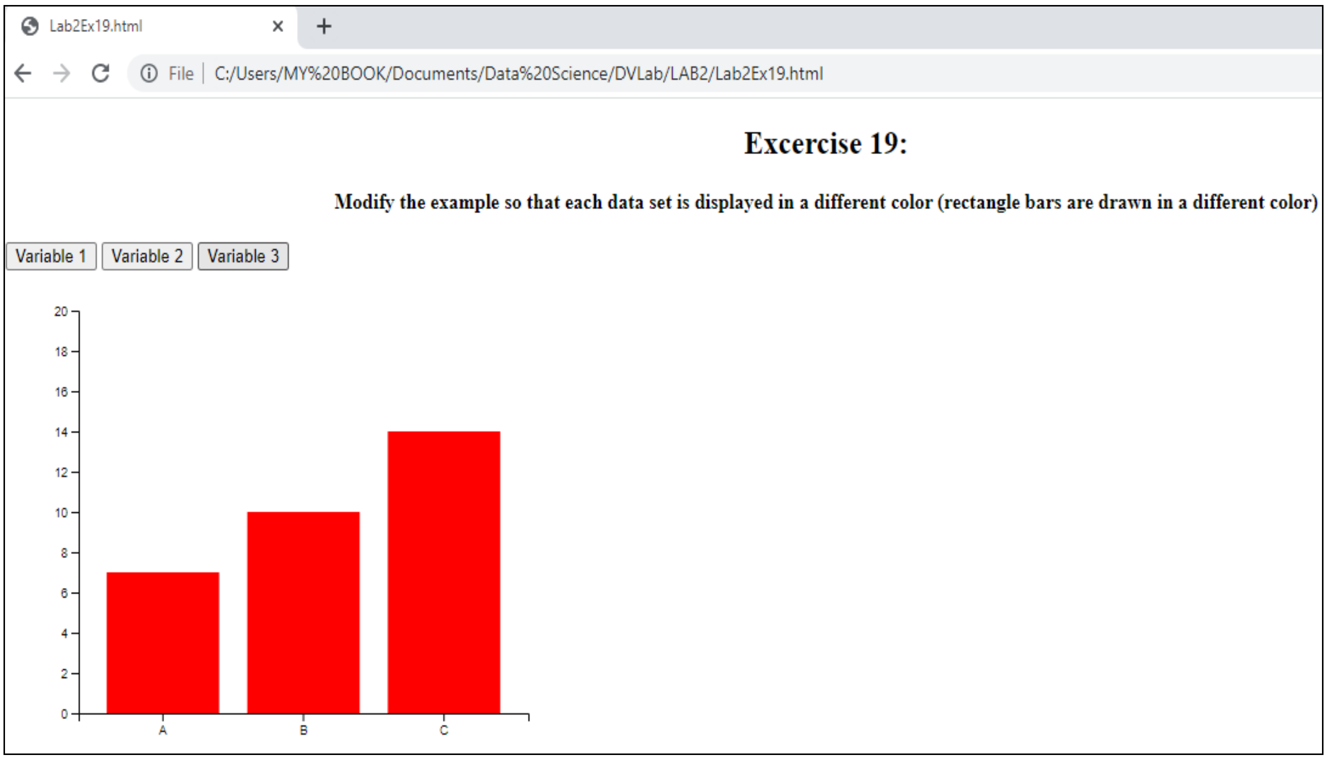
Modify the example so that each data set is displayed in a different color (rectangle bars are drawn in a different color)



C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex19.html

Excercise 19:

Modify the example so that each data set is displayed in a different color (rectangle bars are drawn in a different color)


Lab2Ex19.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex19.html

Excercise 19:

Modify the example so that each data set is displayed in a different color (rectangle bars are drawn in a different color)

Exercise 21:

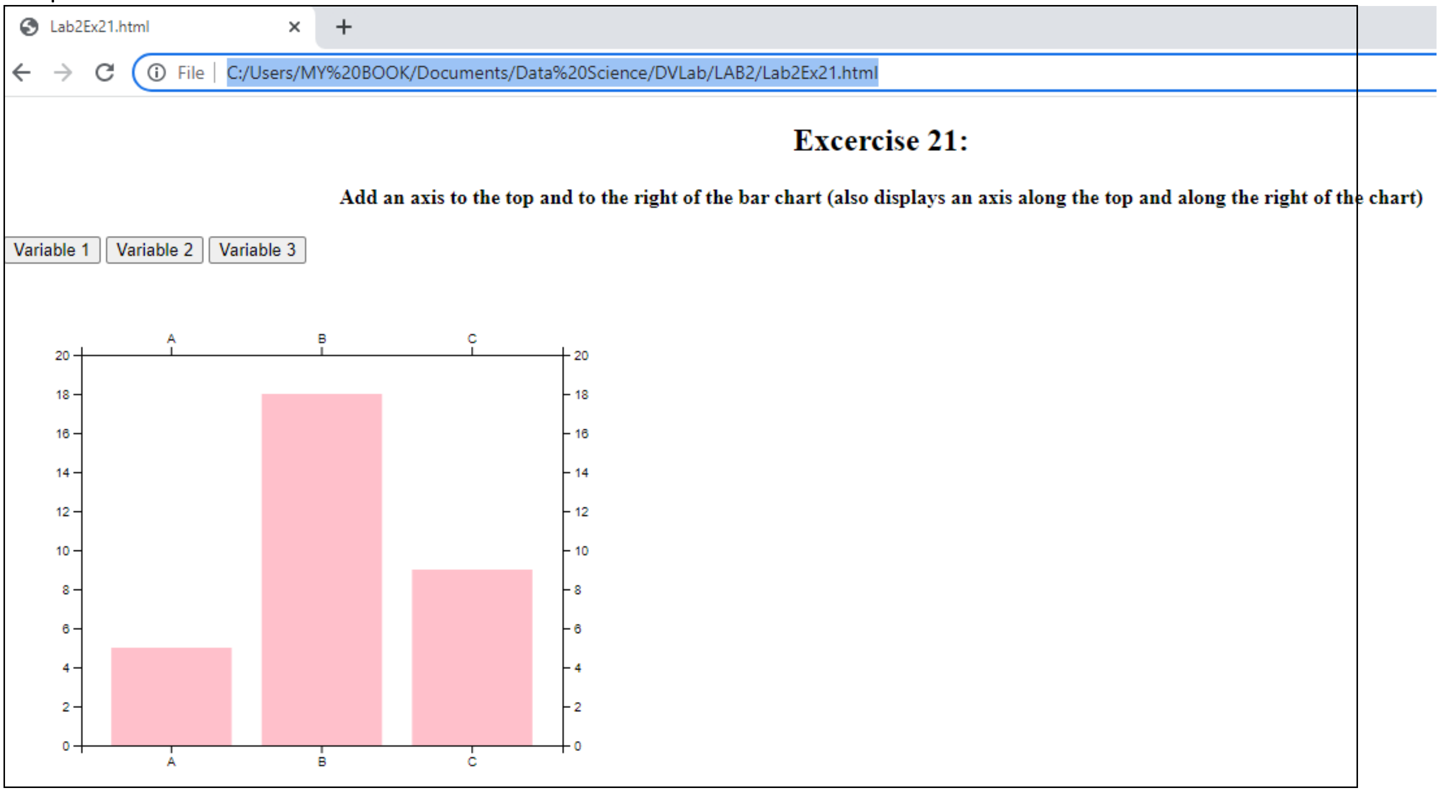
Add an axis to the top and to the right of the bar chart (also displays an axis along the top and along the right of the chart)

Snippet:

```
// append the svg object to the body of the page
       var svg = d3
         .select("body")
         .append("div")
         .append("svg")
         .attr("width", width + margin.left + margin.right)
         .attr("height", height + margin.top + margin.bottom)
         .append("g")
         .attr(
           "transform",
           "translate(" + margin.right + "," + margin.top + ")"
         );
       // X axis
       var x = d3
         .scaleBand()
         .range([0, width])
         .domain(
           data1.map(function (d) {
             return d.group;
           })
         )
         .padding(0.2);

         //Bottom axis
       svg.append("g")
         .attr("transform", "translate(0," + height + ")")
         .call(d3.axisBottom(x));

       // top axis
         svg.append("g")
         .call(d3.axisTop(x));

       // Add Y axis
       var y = d3.scaleLinear()
               .domain([0, 20])
               .range([height, 0]);

         // left y axis
          svg.append("g")
           .attr("class", "myYaxis")
           .call(d3.axisLeft(y));

         // right y axis
         svg.append("g")
         .attr("transform", "translate(370,0)")
         .attr("class", "myYaxis")
```

```
          .call(d3.axisRight(y));
```

Output:



Lab2Ex21.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex21.html

Excercise 21:

Add an axis to the top and to the right of the bar chart (also displays an axis along the top and along the right of the chart)

PART-6: Pie chart

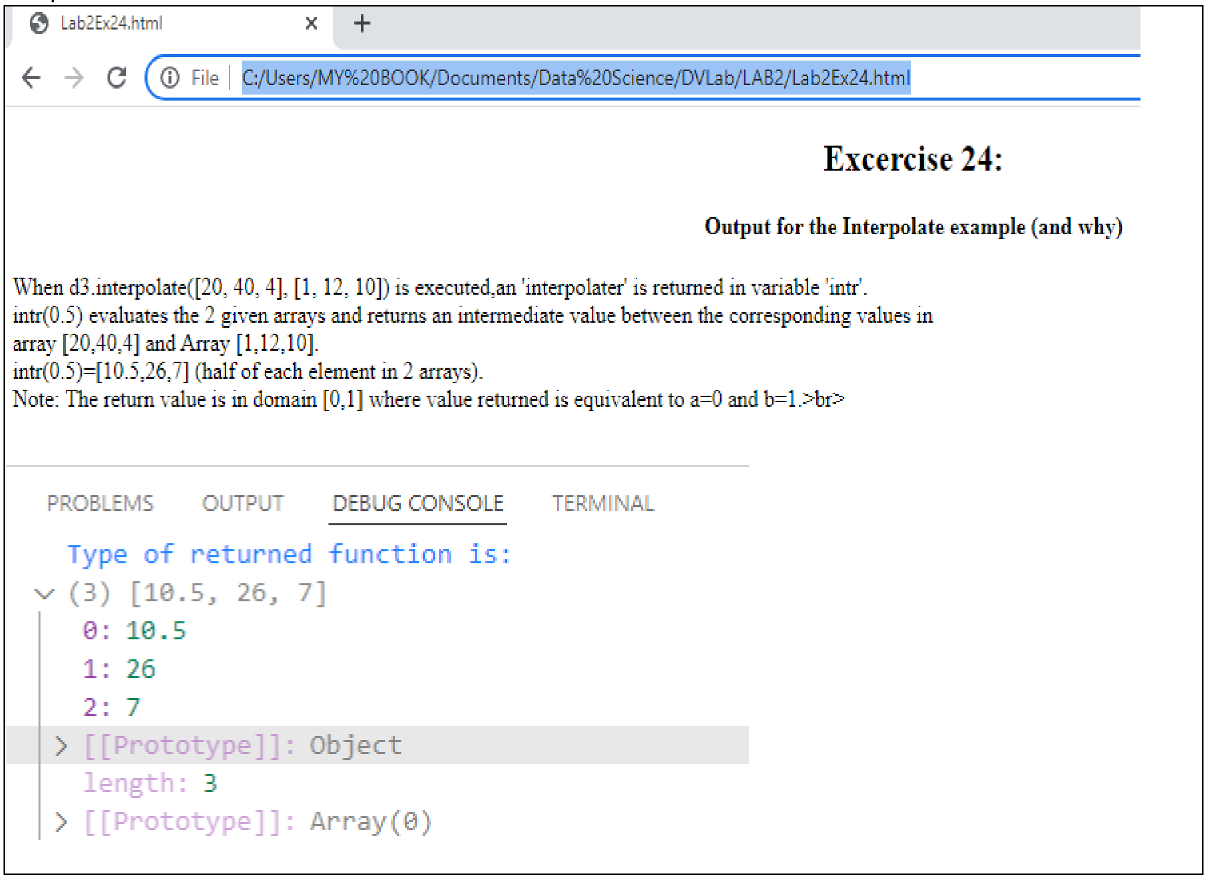
Exercise 24:

Output for the Interpolate example (and why)

Snippet:

```
<p>When d3.interpolate([20, 40, 4], [1, 12, 10]) is executed,an
'interpolater' is returned in variable 'intr'.<br>
           intr(0.5) evaluates the 2 given arrays and returns an
intermediate value between the corresponding values in <br>
           array [20,40,4] and Array [1,12,10].  <br>
          intr(0.5)=[10.5,26,7] (half of each element in 2 arrays).<br>
           Note: The return value is in domain [0,1] where value returned
is equivalent to a=0 and b=1.>br>
           </p>
      <script
        type="text/javascript"
        src="https://d3js.org/d3.v7.min.js"
      ></script>
      <script>
        let intr = d3.interpolate([20, 40, 4], [1, 12, 10]);
        console.log("Type of returned function is:");
        console.log(intr(0.5));
      </script>
```

Output:



Lab2Ex24.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex24.html

Excercise 24:

Output for the Interpolate example (and why)

When d3.interpolate([20, 40, 4], [1, 12, 10]) is executed,an 'interpolater' is returned in variable 'intr'.
intr(0.5) evaluates the 2 given arrays and returns an intermediate value between the corresponding values in
array [20,40,4] and Array [1,12,10].
intr(0.5)=[10.5,26,7] (half of each element in 2 arrays).
Note: The return value is in domain [0,1] where value returned is equivalent to a=0 and b=1.>br>

```
  Type of returned function is:
v (3) [10.5, 26, 7]
    0: 10.5
    1: 26
    2: 7
  > [[Prototype]]: Object
    length: 3
  > [[Prototype]]: Array(0)
```

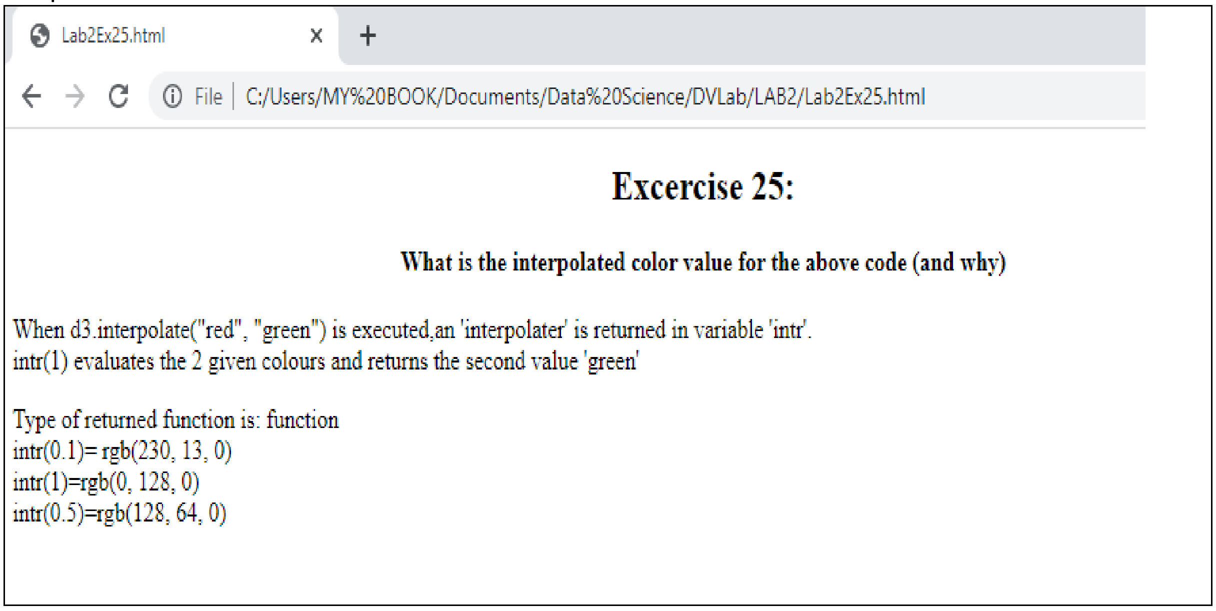
Exercise 25:

What is the interpolated color value for the given code (and why)

Snippet:

```
 <p>When d3.interpolate("red", "green") is executed,an 'interpolater' is
returned in variable 'intr'.<br>
        intr(1) evaluates the 2 given colours and returns the second value
'green' <br>

       </p>
        <p>Type of returned function is:  function <br>

           intr(0.1)= rgb(230, 13, 0) <br>
            intr(1)=rgb(0, 128, 0) <br>
            intr(0.5)=rgb(128, 64, 0) <br>

        </p>

      <script type="text/javascript"
src="https://d3js.org/d3.v7.min.js"></script>
      <script>
        let intr = d3.interpolate("red", "green");
        console.log("Type of returned function is: ", typeof intr);
        console.log(intr(0.1));
        console.log(intr(1));
        console.log(intr(0.5));
      </script>
```

Output:

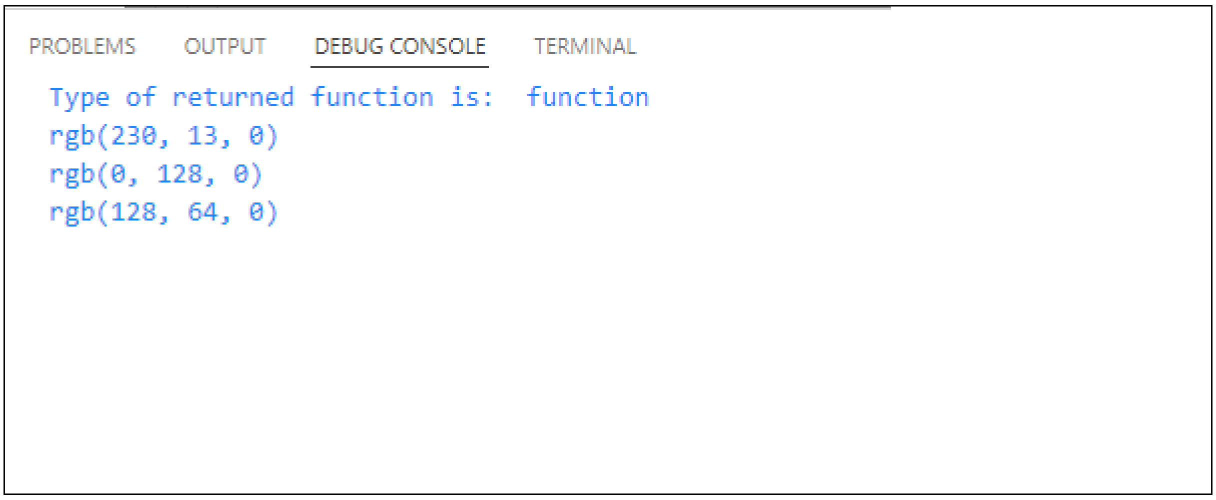

Lab2Ex25.html
File
C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex25.html

Excercise 25:

What is the interpolated color value for the above code (and why)

When d3.interpolate("red", "green") is executed,an 'interpolater' is returned in variable 'intr'.
intr(1) evaluates the 2 given colours and returns the second value 'green'

Type of returned function is: function
intr(0.1)= rgb(230, 13, 0)
intr(1)=rgb(0, 128, 0)
intr(0.5)=rgb(128, 64, 0)



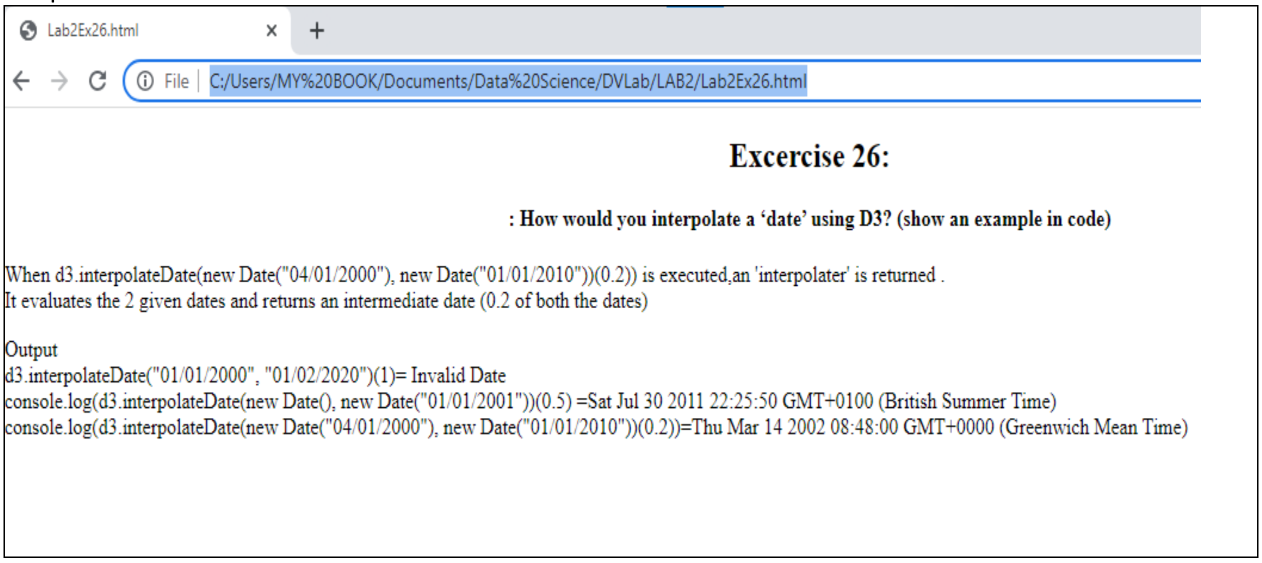
```
Type of returned function is:  function
rgb(230, 13, 0)
rgb(0, 128, 0)
rgb(128, 64, 0)
```

Exercise 26:
How would you interpolate a 'date' using D3? (show an example in code)

Snippet:

```
 <p>When d3.interpolateDate(new Date("04/01/2000"), new Date("01/01/2010"))(0.2))
is executed,an 'interpolater' is returned .<br>
        It evaluates the 2 given dates and returns an intermediate date (0.2 of
both the dates) <br>

       </p>
        <p> Output<br>

          d3.interpolateDate("01/01/2000", "01/02/2020")(1)= Invalid Date <br>
          console.log(d3.interpolateDate(new Date(), new Date("01/01/2001"))(0.5)
=Sat Jul 30 2011 22:25:50 GMT+0100 (British Summer Time)<br>
          console.log(d3.interpolateDate(new Date("04/01/2000"), new
Date("01/01/2010"))(0.2))=Thu Mar 14 2002 08:48:00 GMT+0000 (Greenwich Mean
Time)<br>

        </p>
      <script type="text/javascript" src="https://d3js.org/d3.v7.min.js"></script>
      <script>
        console.log(d3.interpolateDate("01/01/2000", "01/02/2020")(1));
        // Given end date only
        console.log(d3.interpolateDate(new Date(), new Date("01/01/2001"))(0.5)
        );
        // When both start and end date is given
        console.log(d3.interpolateDate(new Date("04/01/2000"), new
Date("01/01/2010"))(0.2));
```

Output:



Lab2Ex26.html

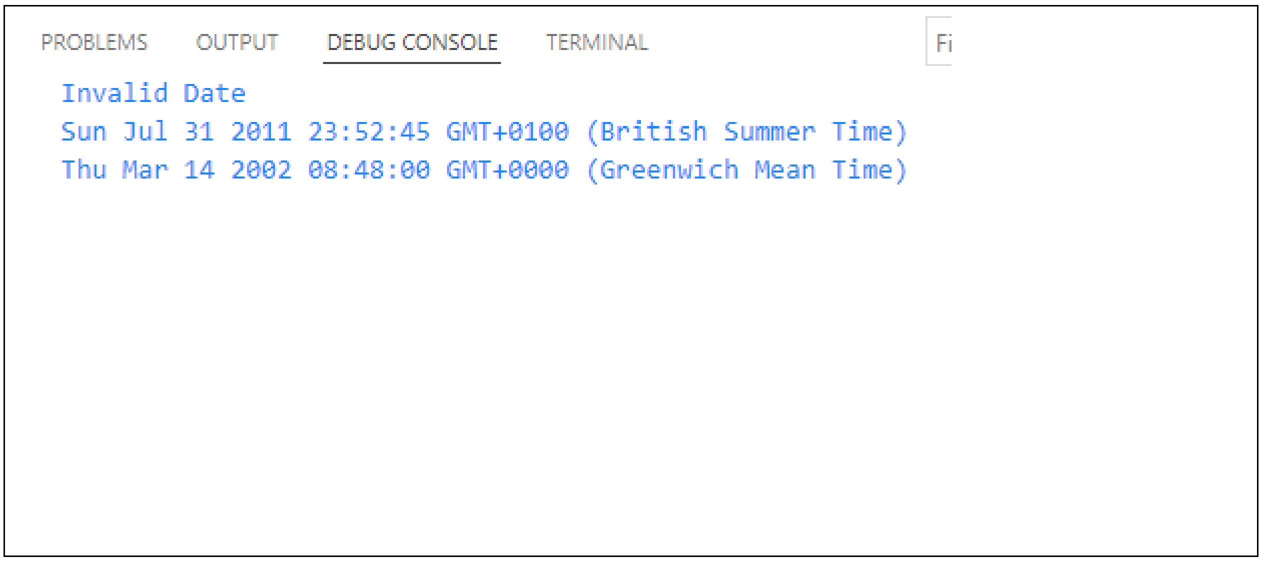
File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex26.html

Excercise 26:

: How would you interpolate a 'date' using D3? (show an example in code)

When d3.interpolateDate(new Date("04/01/2000"), new Date("01/01/2010"))(0.2)) is executed,an 'interpolater' is returned .
It evaluates the 2 given dates and returns an intermediate date (0.2 of both the dates)

Output
d3.interpolateDate("01/01/2000", "01/02/2020")(1)= Invalid Date
console.log(d3.interpolateDate(new Date(), new Date("01/01/2001"))(0.5) =Sat Jul 30 2011 22:25:50 GMT+0100 (British Summer Time)
console.log(d3.interpolateDate(new Date("04/01/2000"), new Date("01/01/2010"))(0.2))=Thu Mar 14 2002 08:48:00 GMT+0000 (Greenwich Mean Time)



```
Invalid Date
Sun Jul 31 2011 23:52:45 GMT+0100 (British Summer Time)
Thu Mar 14 2002 08:48:00 GMT+0000 (Greenwich Mean Time)
```

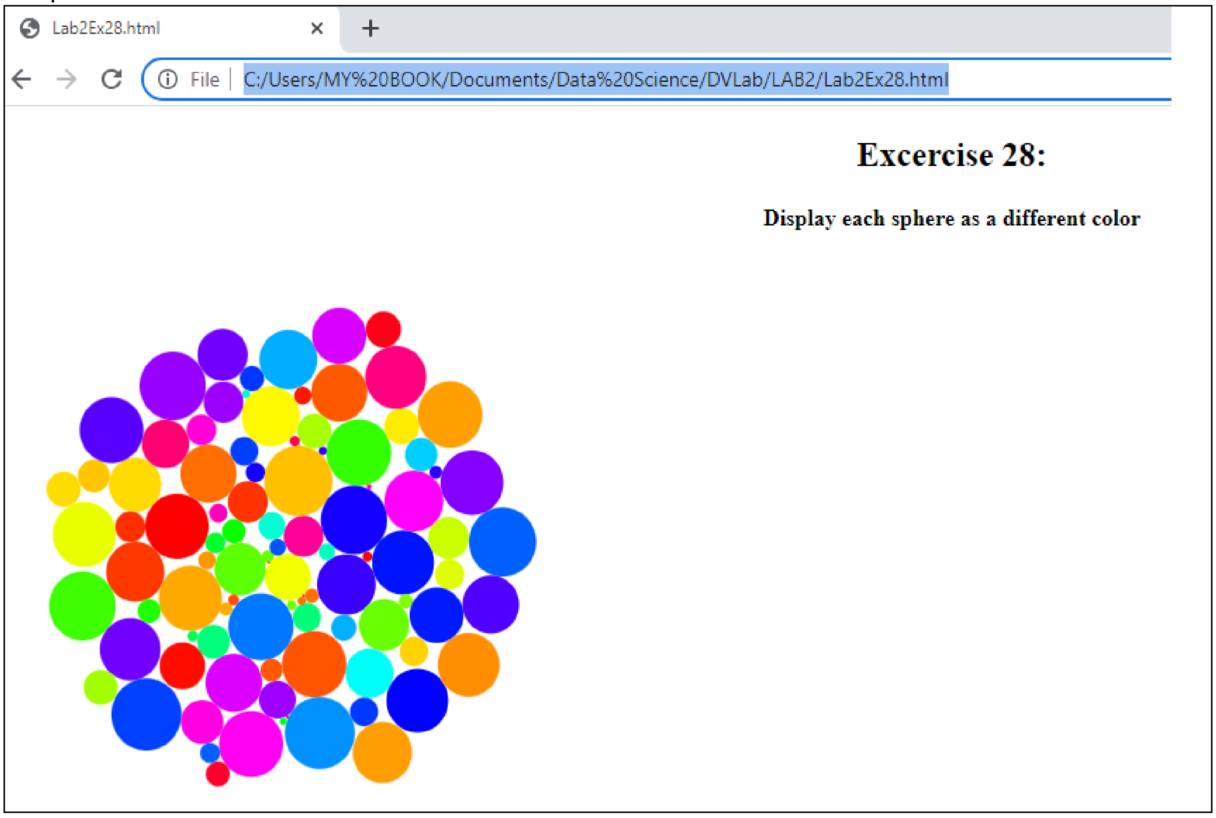
Part 7. D3 Force layout

Exercise 28*:* Display each sphere is as a different colour using d3.force

Snippet:

```
  function ticked() {
          var u = d3
            .select("svg")
            .selectAll("circle")
            .data(nodes)
            .join("circle")
            //using dynamic properties, displaying all circles in different
colors https://d3js.org/#properties
            //var myColor = d3.scaleSequential()
                            //  .interpolator(d3.interpolateInferno)
                             // .domain([1,100])
             //to display random colours
            .attr("fill", function () {
              return "hsl(" + Math.random() * 720 + ",100%,50%)";
            })
            .attr("r", function (d) {
              return d.radius;
            })
            .attr("cx", function (d) {
              return d.x;
            })
            .attr("cy", function (d) {
              return d.y;
            });
        }

      </script>
```

Output:



Lab2Ex28.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex28.html

Excercise 28:

Display each sphere as a different color

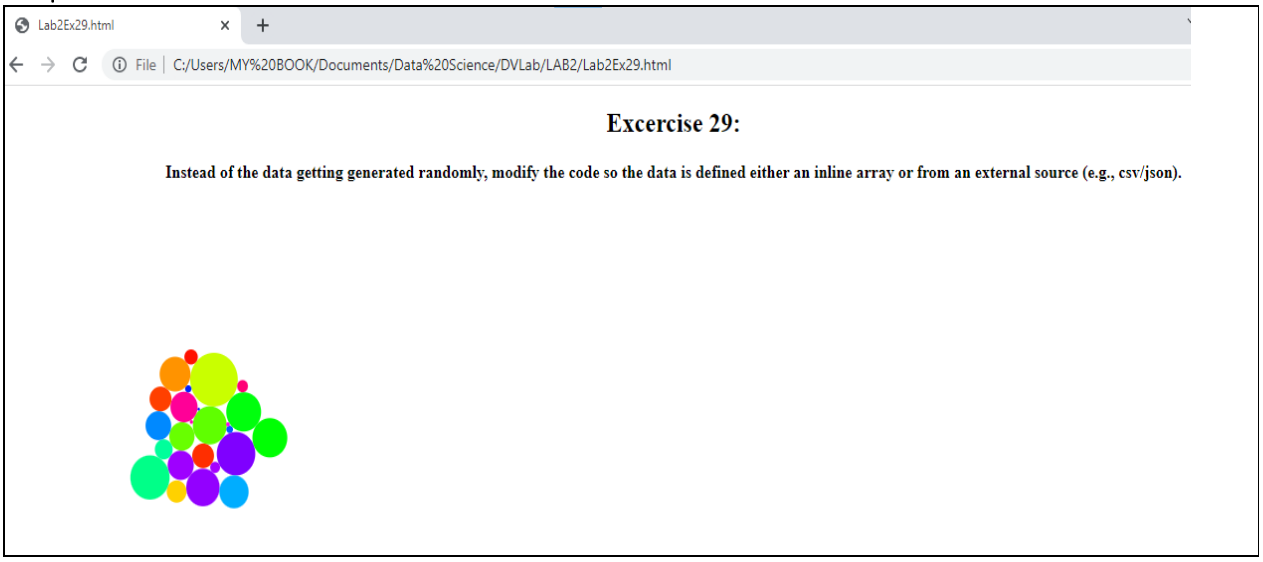
Exercise 29:
Loading data from csv file to display sphere using d3.force

CSV file used: https://raw.githubusercontent.com/ankithacherian/DVLab2/main/radius.csv

Snippet:

```
 var rad = 

"https://raw.githubusercontent.com/ankithacherian/DVLab2/main/radius.csv";
        var width = 400,
          height = 400;
        // setup svg
        d3.select("body")
          .append("svg")
          .attr("width", width)
          .attr("height", height);
        // load data from csv file
        var numNodes = 25;
        var nodes = d3.range(numNodes).map(function (d) {
          return { radius: Math.random() * 25 };
        });
          d3.csv(rad).then(function(d) {
          var nodes = d3.range(numNodes).map(function(d) {
           return {
             radius: d.radius}
```

Output:



Lab2Ex29.html

File | C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex29.html

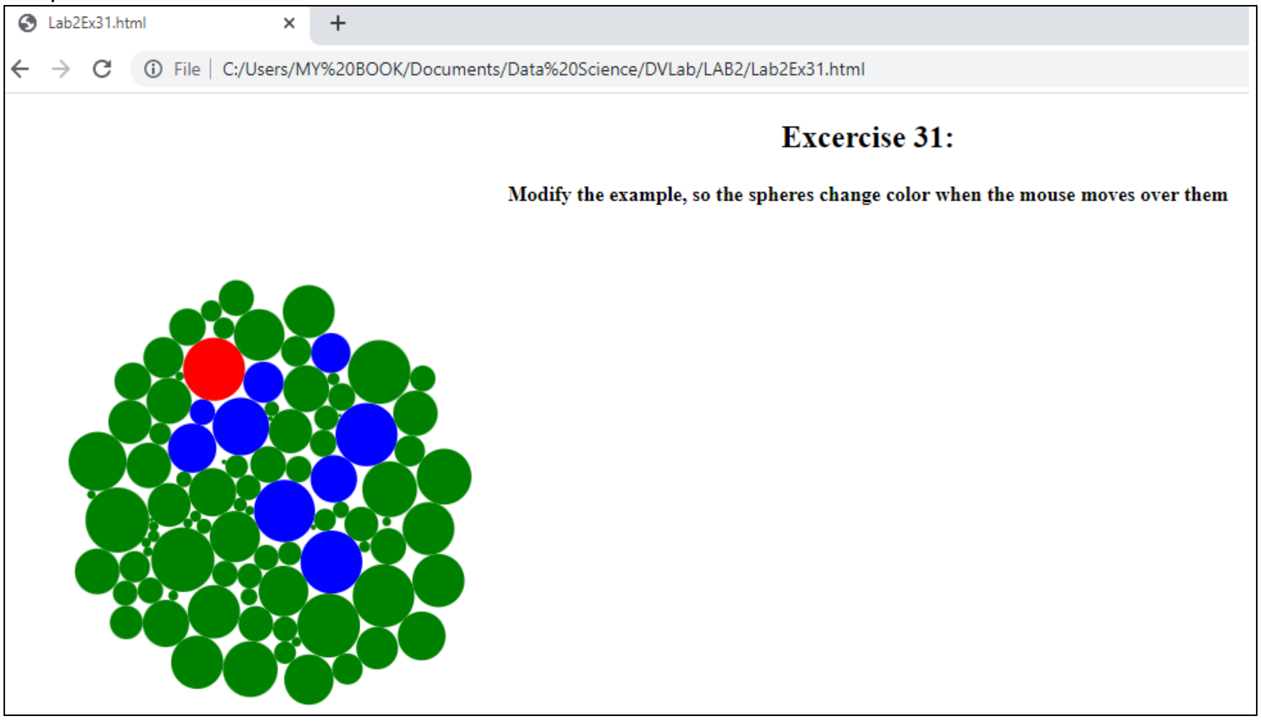
Excercise 29:

Instead of the data getting generated randomly, modify the code so the data is defined either an inline array or from an external source (e.g., csv/json).

Exercise 31:

Display colour change of sphere when the mouse moves over them

Output:


Lab2Ex31.html
File
C:/Users/MY%20BOOK/Documents/Data%20Science/DVLab/LAB2/Lab2Ex31.html

Excercise 31:

Modify the example, so the spheres change color when the mouse moves over them