5.1.html

<!DOCTYPE html>

<html lang="en">

    <head>

        <meta name="author" content="Minh Nguyen">

        <meta name="keywords" content="HTML, CSS">

        <meta name="description" content="5.1">

        <title>Task 5.1 D3 Updating the Data</title>

        <link rel="stylesheet" href="5.1.css">

    </head>

    <body>

        <h1>Update Data</h1>

        <button type="button" id="btn">Update</button>

        <p id="chart"></p>

        <footer style="color:grey">COS30045 Data Visualisation<br>Minh Nguyen</footer>

    </body>

    <script src="https://d3js.org/d3.v7.min.js"></script>

    <script src="5.1.js"></script>

</html>

5.1.css

body{

    margin: auto;

    width: 95%;

    padding: 1px;

}

footer{

    padding-left: 1px;

    margin: auto;

}

h1{

    color: black;

}

button{

    background-color: green;

    border: none;

    box-shadow: gold 1px 0px 0px;

    border-radius: 15px;

    height: 30px;

    width: 100px;

    transition-duration: 0.4s;

    cursor: pointer;

}

button:hover{

    background-color: lightblue;

}

5.1.js

function init() {

    //Max value for data

    var maxValue = 25;

    // width and height

    var w = 500;

    var h = 250;

    var barPadding = 1;

    var dataset= [4,22,18,5,9,7,16,21,9,13,23,9,24,17,8,25,19,4,22,14,6,21];

    // Scale method

    var xScale = d3.scaleBand()

                    .domain(d3.range(dataset.length))

                    .rangeRound([0, w])

                    .paddingInner(0.05);

    var yScale = d3.scaleLinear()

                    .domain([d3.max(dataset,function(d){

                        return 30;

                    }),

                    d3.min(dataset,function(d){

                        return 0;

                    })])

                    .range([0, h]);

    //Create SVG element

    var svg = d3.select("#chart")

            .append("svg")

            .attr("width", w+50)

            .attr("height", h+50);

    //On click, update with new data

    d3.select("button")

        .on("click", function() {

    //Random values for dataset

    var numValues = dataset.length;

    dataset = [];

    for (var i = 0; i < numValues; i++) {

        var newNumber = Math.floor(Math.random()\* maxValue);

        dataset.push(newNumber);

    }

    //Update all rects

    svg.selectAll("rect")

        .data(dataset)

        .attr("x", function(d, i){

            return xScale(i)+20;

        })

        .attr("y", function(d){

            return yScale(d);

        })

        .attr("width", xScale.bandwidth())

        .attr("height", function(d) {

            return h-yScale(d);})

        .attr("fill", function(d){

            return "rgb(0, 0, " + (d \* 10) + ")";

        });

        console.log(dataset);

    //Update all texts

    svg.selectAll("text")

    .data(dataset)

    .text(function(d) {

        return d;

    })

    .attr("text-anchor", "middle")

    .attr("x", function(d, i) {

        return xScale(i)+20+ xScale.bandwidth()/2;;

    })

    .attr("y", function(d) {

        return yScale(d)+14;

    })

    .attr("font-family", "sans-serif")

    .attr("font-size", "11px")

    .attr("fill", "white");

    })

    //Create bars

    svg.selectAll("rect")

        .data(dataset)

        .enter()

        .append("rect")

        .attr("x", function(d, i){

            return xScale(i)+20;

        })

        .attr("y", function(d){

            return yScale(d);

        })

        .attr("width", xScale.bandwidth())

        .attr("height", function(d) {

            return h-yScale(d)})

        .attr("fill", function(d){

                return "rgb(0, 0, " + (d \* 10) + ")";

            });

    //Create labels

    svg.selectAll("text")

            .data(dataset)

            .enter()

            .append("text")

            .text(function(d) {

                return d;

            })

            .attr("text-anchor", "middle")

            .attr("x", function(d, i) {

                return xScale(i) + 20 + xScale.bandwidth() / 2;

            })

            .attr("y", function(d) {

                return yScale(d) + 14;

            })

            .attr("font-family", "sans-serif")

            .attr("font-size", "10px")

            .attr("fill", "white");

    //Width

    svg.append("g")

        .attr("transform","translate(0, "+(h - barPadding+10) +")")

        .call(xAxis);

    //Height

    svg.append("g")

        .attr("transform","translate("+(barPadding+10) +")")

        .call(yAxis);

}

window.onload = init;

5.2.html

<!DOCTYPE html>

<html lang="en">

    <head>

        <meta name="author" content="Minh Nguyen">

        <meta name="keywords" content="HTML, CSS">

        <meta name="description" content="5.2">

        <title>Task 5.2 D3 Transitions</title>

        <link rel="stylesheet" href="5.2.css">

    </head>

    <body>

        <h1>Updates and Transitions</h1>

        <button type="button" id="Update">Update</button>

        <button type="button" id="Tran1">Transition 1</button>

        <button type="button" id="Tran2">Transition 2</button>

        <p id="chart"> </p>

        <footer style="color:grey">COS30045 Data Visualisation<br>Minh Nguyen</footer>

    </body>

    <script src="https://d3js.org/d3.v7.min.js"></script>

    <script src="5.2.js"></script>

</html>

5.2.css

body{

    margin: auto;

    width: 95%;

    padding: 1px;

}

footer{

    padding-left: 1px;

    margin: auto;

}

h1{

    color: black;

}

#Update{

    background-color: lightblue;

    border: none;

    border-radius: 15px;

    height: 30px;

    width: 100px;

    transition-duration: 0.4s;

    cursor: pointer;

}

#Update:hover{

    background-color: green;

}

#Tran1{

    background-color: gold;

    border: none;

    border-radius: 15px;

    height: 30px;

    width: 100px;

    transition-duration: 0.4s;

    cursor: pointer;

}

#Tran1:hover{

    background-color: yellowgreen;

}

#Tran2{

    background-color: darkorange;

    border: none;

    border-radius: 15px;

    height: 30px;

    width: 100px;

    transition-duration: 0.4s;

    cursor: pointer;

}

#Tran2:hover{

    background-color: khaki;

}

5.2.js

function init() {

    var w = 600;

    var h = 200;

    var maxValue = 30;

    var dataset = [14, 5, 26, 23, 9, 20, 12, 17, 10, 4, 9, 20, 29, 30, 12, 4, 8, 15, 28, 27, 25, 3, 11, 13, 5];

    var xScale = d3.scaleBand()

                   .domain(d3.range(dataset.length))

                   .rangeRound([0, w])

                   .paddingInner(0.05);

    var yScale = d3.scaleLinear()

                   .domain([0, d3.max(dataset)])

                   .range([0, h]);

    var svg = d3.select("#chart")

                .append("svg")

                .attr("width", w)   //total length

                .attr("height", h); //total height

    //Update

    d3.select("#Update")

    .on("click", function() {

        //alert("Hey, the button works!")

        var numValues = dataset.length;

        dataset = [];

        for (var i = 0; i < numValues; i++) {

            var newNumber = Math.floor(Math.random() \* maxValue);

            dataset.push(newNumber);

        }

        svg.selectAll("rect")

           .data(dataset)

           .transition() //calling a transition

           .delay(function (d, i) {

               return i / dataset.length \* 1000;

           })

           .duration(function(d, i) {

               return i\* 100;

           })

           .ease(d3.easeCircleIn) //easing function

           .attr("x", function(d, i) {

           return xScale(i);

           })

           .attr("y", function(d) {

               return h - yScale(d);

           })

           // .attr("width", xScale.bandwidth())

           .attr("height", function(d) {

               return yScale(d);

           })

           .attr("fill", function(d) {

               return "rgb(" + (d \* 10) + ", 0, 0)";

           });

           svg.selectAll("text")

           .data(dataset)

           .text(function(d) {

               return d;

           })

           .attr("x", function(d, i) {

               return xScale(i) + xScale.bandwidth() / 2;

           })

           .attr("y", function(d) {

               return h - yScale(d) + 14;

           })

           .attr("fill", "white")

           .attr("text-anchor", "middle");

    });

    //Transition 1

    d3.select("#Tran1")

    .on("click", function() {

        var numValues = dataset.length;

        dataset = [];

        for (var i = 0; i < numValues; i++) {

            var newNumber = Math.floor(Math.random() \* maxValue);

            dataset.push(newNumber);

        }

        svg.selectAll("rect")

           .data(dataset)

           .transition()

           .duration(500)

           .ease(d3.easeCircleOut)

           .attr("x", function(d, i) {

            return xScale(i);

           })

           .attr("y", function(d) {

            return h - yScale(d);

           })

           // .attr("width", xScale.bandwidth())

           .attr("height", function(d) {

               return yScale(d);

           })

           .attr("fill", function(d) {

               return "rgb(0, 0, " + (d \* 10) + ")";

           });

           svg.selectAll("text")

           .data(dataset)

           .text(function(d) {

               return d;

           })

           .attr("x", function(d, i) {

               return xScale(i) + xScale.bandwidth() / 2;

           })

           .attr("y", function(d) {

               return h - yScale(d) + 14;

           })

           .attr("fill", "white")

           .attr("text-anchor", "middle");

    });

    //Transition 2

    d3.select("#Tran2")

    .on("click", function() {

        var numValues = dataset.length;

        dataset = [];

        for (var i = 0; i < numValues; i++) {

            var newNumber = Math.floor(Math.random() \* maxValue);

            dataset.push(newNumber);

        }

        svg.selectAll("rect")

           .data(dataset)

           .transition()

           .delay(1000)

           .duration(2000)

           .ease(d3.easeCircleInOut)

           .attr("x", function(d, i) {

           return xScale(i);

           })

           .attr("y", function(d) {

               return h - yScale(d);

           })

           // .attr("width", xScale.bandwidth())

           .attr("height", function(d) {

               return yScale(d);

           })

           .attr("fill", function(d) {

               return "rgb(0, 0, " + (d \* 10) + ")";

           });

        svg.selectAll("text")

           .data(dataset)

           .text(function(d) {

               return d;

           })

           .attr("x", function(d, i) {

               return xScale(i) + xScale.bandwidth() / 2;

           })

           .attr("y", function(d) {

               return h - yScale(d) + 14;

           })

           .attr("fill", "white")

           .attr("text-anchor", "middle");

    });

    svg.selectAll("rect")

        .data(dataset)

        .enter()

        .append("rect")

        .attr("x", function(d, i) {

            return xScale(i);

        })

        .attr("y", function(d) {

            return h - yScale(d);

        })

        .attr("width", xScale.bandwidth())

        .attr("height", function(d) {

            return yScale(d);

        })

        .attr("fill", function(d) {

            return "rgb(0, 0, " + (d \* 10) + ")";

        });

    svg.selectAll("text")

       .data(dataset)

       .enter()

       .append("text")

       .text(function(d) {

           return d;

       })

       .attr("x", function(d, i) {

           return xScale(i) + xScale.bandwidth() / 2;

       })

       .attr("y", function(d) {

           return h - yScale(d) + 14;

       })

       .attr("fill", "white")

       .attr("text-anchor", "middle");

}

window.onload = init;

5.3.html

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8"/>

    <meta name="description" content="Data Visualisation"/>

    <meta name="keyword" content="HTML, CSS, D3"/>

    <meta name="author" content="Minh Nguyen"/>

    <title>Task 5.3</title>

    <link rel="stylesheet" href="5.3.css">

</head>

<body>

    <h1>Adding and Removing Values</h1>

    <button type="button" id="Add">Add</button>

    <button type="button" id="Remove">Remove</button>

    <p id="chart"> </p>

    <p id="chart"></p>

    <br>

    <bf></bf>

    <footer style="color:grey">COS30045 Data Visualisation<br>Minh Nguyen</footer>

    <script src="https://d3js.org/d3.v7.min.js"></script>

    <script src="5.3.js"></script>

</body>

</html>

5.3.css

body{

    margin: auto;

    width: 95%;

    padding: 1px;

}

footer{

    padding-left: 1px;

    margin: auto;

}

h1{

    color: black;

}

#Add{

    background-color: lightblue;

    border: none;

    border-radius: 15px;

    height: 30px;

    width: 100px;

    transition-duration: 0.4s;

    cursor: pointer;

}

#Add:hover{

    background-color: green;

}

#Remove{

    background-color: gold;

    border: none;

    border-radius: 15px;

    height: 30px;

    width: 100px;

    transition-duration: 0.4s;

    cursor: pointer;

}

#Remove:hover{

    background-color: khaki;

}

5.3.js

function init() {

    var w = 600;

    var h = 200;

    var maxValue = 25;

    var dataset = [14, 5, 26, 23, 9, 20, 12, 17, 10, 4, 9, 20, 29, 30, 12, 4, 8, 15, 28, 27, 25, 3, 11, 13, 5];

    var xScale = d3.scaleBand()

                   .domain(d3.range(dataset.length))

                   .rangeRound([0, w])

                   .paddingInner(0.05);

    var yScale = d3.scaleLinear()

                   .domain([0, d3.max(dataset)])

                   .range([0, h]);

    var svg = d3.select("#chart")

                .append("svg")

                .attr("width", w)

                .attr("height", h);

    //Add Data

    d3.select("#Add")

    .on("click", function() {

        var newNumber = Math.floor(Math.random() \* maxValue);

        dataset.push(newNumber);

        var bars = svg.selectAll("rect")

        .data(dataset);

        var labels = svg.selectAll("text")

        .data(dataset);

        xScale.domain(d3.range(dataset.length));

        bars.enter()

            .append("rect") //creates the new rect

            .attr("x", w)

            .attr("y", function(d) {

                return h - yScale(d);

            })

            .merge(bars) //integrating it with other bars

            .transition()

            .duration(500)

            .attr("x", function(d, i) {

                return xScale(i);

            })

            .attr("y", function(d) {

                return h - yScale(d);

            })

            .attr("width", xScale.bandwidth())

            .attr("height", function(d) {

                return yScale(d);

            })

            .attr("fill", function(d) {

                return "rgb(0, 0, " + (d \* 10) + ")";

            });

        labels.enter() //creates the new text when a bar is entered

              .append("text")

              .merge(labels)

              .transition()

              .duration(500)

              .text(function(d) {

                  return d;

              })

              .attr("x", function(d, i) {

                  return xScale(i) + xScale.bandwidth()/2;

              })

              .attr("y", function(d) {

                  return h - yScale(d) + 14;

              })

              .attr("fill", "white")

              .attr("text-anchor", "middle");

    });

    //Remove

    d3.select("#Remove")

    .on("click", function() {

        //dataset.shift(); //removes first element of the array

        dataset.pop(); //removes last element of the array

        var bars = svg.selectAll("rect").data(dataset);

        var labels = svg.selectAll("text").data(dataset);

        xScale.domain(d3.range(dataset.length));

        bars.exit()

            .transition()

            .duration(500)

            .attr("x", w)

            .remove("x", w)

        bars.transition()

            .delay(500)

            .attr("x", function(d, i) {

            return xScale(i);

            })

            .attr("y", function(d) {

            return h - yScale(d);

            })

            .attr("width", xScale.bandwidth())

            .attr("height", function(d) {

                return yScale(d);

            })

            .attr("fill", function(d) {

                return "rgb(0, 0, " + (d \* 10) + ")";

            });

        labels.exit()

              .transition()

              .duration(500)

              .attr("x", w)

              .remove()

        labels.transition()

              .delay(500)

              .text(function(d) {

                  return d;

              })

              .attr("x", function(d, i) {

                  return xScale(i) + xScale.bandwidth()/2;

              })

              .attr("y", function(d) {

                  return h - yScale(d) + 14;

              })

              .attr("text-anchor", "middle")

              .attr("fill", "white");

    });

    svg.selectAll("rect")

        .data(dataset)

        .enter()

        .append("rect")

        .attr("x", function(d, i) {

            return xScale(i);

        })

        .attr("y", function(d) {

            return h - yScale(d);

        })

        .attr("width", xScale.bandwidth())

        .attr("height", function(d) {

            return yScale(d);

        })

        .attr("fill", function(d) {

            return "rgb(0, 0, " + (d \* 10) + ")";

        });

    svg.selectAll("text")

        .data(dataset)

        .enter()

        .append("text")

        .text(function(d) {

            return d;

        })

        .attr("x", function(d, i) {

            return xScale(i) + xScale.bandwidth() / 2;

        })

        .attr("y", function(d) {

            return h - yScale(d) + 14;

        })

        .attr("fill", "white")

        .attr("text-anchor", "middle");

}

window.onload = init;

6.1.html

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8"/>

    <meta name="description" content="Data Visualisation"/>

    <meta name="keyword" content="HTML, CSS, D3"/>

    <meta name="author" content="Minh Nguyen"/>

    <title>Task 6.1 D3 Mouse Over</title>

    <link rel="stylesheet" href="6.1.css">

</head>

<body>

    <h1>Bar Chart with Mouse Over</h1>

    <p><button id = "Add">Add</button> <button id="Remove">Remove</button> </p>

    <p id="chart"></p>

    <br>

    <bf></bf>

    <footer style="color:grey">COS30045 Data Visualisation<br>Minh Nguyen</footer>

    <script src="https://d3js.org/d3.v7.min.js"></script>

    <script src="6.1.js"></script>

</body>

</html>

6.1.css

#Add {

    border-radius: 10px;

    color: black;

    border-color: lightblue;

    background-color: lightblue;

}

#Remove {

    border-radius: 10px;

    color: black;

    border-color: orange;

    background-color:orange;

}

#Add:hover, #Remove:hover {

    background-color: white;

}

6.1.js

function init(){

    var w = 500;

    var h = 100;

    var MaxValue = 25;

    var dataset = [14, 5, 26, 23, 9, 20, 25, 29, 15];

    //ordinal data

    var xScale = d3.scaleBand()

                    .domain(d3.range(dataset.length))

                    .rangeRound([0,w])

                    .paddingInner(0.05);

    //quantitive data

    var yScale = d3.scaleLinear()

                    .domain([0, d3.max(dataset)])

                    .range([h, 0]);

    var svg = d3.select("#chart")

                .append("svg")

                .attr("width", w)

                .attr("height", h);

    svg.selectAll("rect")

        .data(dataset)

        .enter()

        .append("rect")

        .attr("x", function(d, i){

            return xScale(i); //change the amounts of data relative to the dataset

        })

        .attr("y", function(d){

            return yScale(d);

        })

        .attr("width", xScale.bandwidth()) //change the space relative to the dataset

        .attr("height", function(d){

            return h - yScale(d);

        })

        .attr("fill", "blue") // add color to bars

        .on("mouseover", function (event, d) {

            var xPosition = parseFloat(d3.select(this).attr("x")) +xScale.bandwidth() / 2;

            var yPosition = parseFloat(d3.select(this).attr("y")) +15

            svg.append("text")

            .attr("id","tooltip")

            .attr("x", xPosition)

            .attr("y", yPosition)

            .text(d)

            .attr("font-family", "sans-serif")

            .attr("font-size", "12px")

            .attr("fill", "black")

            .attr("text-anchor", "middle")

            .style("font-weight", "bold");

            d3.select(this).attr("fill", "lightblue");

        })

        //Create the tooltip text element and set attributes

        .on("mouseout", function(){

            d3.select("#tooltip").remove()

            d3.select(this)

            .attr("fill", "blue");

        });

    //Add Button

    d3.select("#Add")

        .on("click", function(){

            Add()

        });

    //Remove Button

    d3.select("#Remove")

        .on("click", function(){

            Remove()

        });

        //Update Button Functions

    function Add() {

        var NewNumber = Math.floor(Math.random()\* MaxValue);

        dataset.push(NewNumber);

        xScale.domain(d3.range(dataset.length));

        var bars = svg.selectAll("rect")

                        .data(dataset)

                        bars.enter()

                        .append("rect")

                        .attr("x", w)

                        .attr("y", function(d) {

                            return h - yScale(d);

                        })

                        .merge(bars)

                        .transition()

                        .duration(250)

                        .attr("x", function(d, i){

                                return xScale(i); //change the amounts of data relative to the dataset

                        })

                        .attr("y", function(d){

                                return yScale(d);

                        })

                        .attr("width", xScale.bandwidth()) //change the space relative to the dataset

                        .attr("height", function(d){

                                return h - yScale(d);

                        })

                        .attr("fill", "blue") // add color to bars

                        svg.selectAll("rect")

                        .data(dataset)

                        .on("mouseover", function (event, d) {

                            bars.attr("title", "This value is: " + d);

                            var xPosition = parseFloat(d3.select(this).attr("x")) +xScale.bandwidth() / 2;

                            var yPosition = parseFloat(d3.select(this).attr("y")) +15;

                            svg.append("text")

                            .attr("id","tooltip")

                            .attr("x", xPosition)

                            .attr("y", yPosition)

                            .text(d)

                            .attr("font-family", "sans-serif")

                            .attr("font-size", "12px")

                            .attr("fill", "black")

                            .attr("text-anchor", "middle")

                            .style("font-weight", "bold");

                            d3.select(this).attr("fill", "lightblue");

                        })

                        .on("mouseout", function(){

                            d3.select("#tooltip").remove()

                            d3.select(this)

                            .attr("fill", "blue");

                        });

    }

    //Remove Button Functions

    function Remove() {

        dataset.pop();

        var bars = svg.selectAll("rect")

                        .data(dataset)

                        bars.exit()

                        .transition()

                        .duration(250)

                        .attr("x", w)

                        .remove();

    }

}

window.onload = init;

6.2.html

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8"/>

    <meta name="description" content="Data Visualisation"/>

    <meta name="keyword" content="HTML, CSS, D3"/>

    <meta name="author" content="Minh Nguyen"/>

    <title>Task 6.2</title>

    <link rel="stylesheet" href="6.2.css">

</head>

<body>

    <h1>Bar Chart with Mouse Over and Sort</h1>

    <p><button id = "Add">Add</button> <button id="Remove">Remove</button> <button id="Sort">Sort</button></p>

    <p id="chart"></p>

    <br>

    <bf></bf>

    <footer style="color:grey">COS30045 Data Visualisation<br>Minh Nguyen</footer>

    <script src="https://d3js.org/d3.v7.min.js"></script>

    <script src="6.2.js"></script>

</body>

</html>

6.2.css

#Add {

    border-radius: 10px;

    color: black;

    border-color: lightblue;

    background-color: lightblue;

}

#Remove {

    border-radius: 10px;

    color: black;

    border-color: orange;

    background-color: orange;

}

#Sort {

    border-radius: 10px;

    color: black;

    border-color: yellowgreen;

    background-color: yellowgreen;

}

#Add:hover, #Remove:hover, #Sort:hover {

    background-color: white;

}

6.2.js

function init(){

    var w = 500;

    var h = 100;

    var MaxValue = 25;

    var dataset = [14, 5, 26, 23, 9, 20, 25, 29, 15];

    var sortStatus = "unsorted";

    //ordinal data

    var xScale = d3.scaleBand()

                    .domain(d3.range(dataset.length))

                    .rangeRound([0,w])

                    .paddingInner(0.05);

    //quantitive data

    var yScale = d3.scaleLinear()

                    .domain([0, d3.max(dataset)])

                    .range([h, 0]);

    var svg = d3.select("#chart")

                .append("svg")

                .attr("width", w)

                .attr("height", h);

    svg.selectAll("rect")

        .data(dataset)

        .enter()

        .append("rect")

        .attr("x", function(d, i){

            return xScale(i); //change the amounts of data relative to the dataset

        })

        .attr("y", function(d){

            return yScale(d); //change width of the bar relative to the amount of data in dataset

        })

        .attr("width", xScale.bandwidth()) //change the space relative to the dataset

        .attr("height", function(d){

            return h - yScale(d); //set the bar height relatively

        })

        .attr("fill", "blue") // add color to bars

        .on("mouseover", function (event, d) {

            //get x and y positions of the bars

            var xPosition = parseFloat(d3.select(this).attr("x")) +xScale.bandwidth() / 2;

            var yPosition = parseFloat(d3.select(this).attr("y")) +15

      //Update the tooltip position and value

            svg.append("text")

            .attr("id","tooltip")

            .attr("x", xPosition)

            .attr("y", yPosition)

            .text(d)

            .attr("font-family", "sans-serif")

            .attr("font-size", "12px")

            .attr("fill", "black")

            .attr("text-anchor", "middle")

            .style("font-weight", "bold");

            d3.select(this).attr("fill", "lightblue");

        })

        //mouseout effect for cursor

        .on("mouseout", function(){

            d3.select("#tooltip").remove()

            d3.select(this)

            .attr("fill", "blue");

        });

    //add button functionality

    d3.select("#Add")

        .on("click", function(){

            Add()

        });

    //remove button functionality

    d3.select("#Remove")

        .on("click", function(){

            Remove()

        });

    //sort button functionality

    d3.select("#Sort")

        .on("click", function(){

            Sort();

        });

    //add function

    function Add() {

        var NewNumber = Math.floor(Math.random()\* MaxValue); // Generate random number

        dataset.push(NewNumber); // add random number to dataset

        xScale.domain(d3.range(dataset.length));

        var bars = svg.selectAll("rect")

                        .data(dataset)

                        bars.enter()

                        .append("rect")

                        .attr("x", w)

                        .attr("y", function(d) {

                            return h - yScale(d);

                        })

                        .merge(bars)

                        .transition()

                        .duration(500)

                        .attr("x", function(d, i){

                                return xScale(i); //change the amounts of bar relative to the dataset

                        })

                        .attr("y", function(d){

                                return yScale(d); //change width of the bar relative to the amount of data in dataset

                        })

                        .attr("width", xScale.bandwidth()) //change the space relative to the dataset

                        .attr("height", function(d){

                                return h - yScale(d); //set the bar height relatively

                        })

                        .attr("fill", "blue") // add color to bars

                        svg.selectAll("rect")

                        .data(dataset)

                        .on("mouseover", function (event, d) {

                            bars.attr("title", "This value is: " + d);

                            //get x and y positions of the bars

                            var xPosition = parseFloat(d3.select(this).attr("x")) +xScale.bandwidth() / 2;

                            var yPosition = parseFloat(d3.select(this).attr("y")) +15

                            svg.append("text")

                            .attr("id","tooltip")

                            .attr("x", xPosition)

                            .attr("y", yPosition)

                            .text(d)

                            .attr("font-family", "sans-serif")

                            .attr("font-size", "12px")

                            .attr("fill", "black")

                            .attr("text-anchor", "middle")

                            .style("font-weight", "bold");

                            d3.select(this).attr("fill", "lightblue");

                        })

                        //mouseout effect for cursor

                        .on("mouseout", function(){

                            d3.select("#tooltip").remove()

                            d3.select(this)

                            .attr("fill", "blue");

                        });

    }

    //remove function

    function Remove() {

        dataset.pop();

        var bars = svg.selectAll("rect")

                        .data(dataset)

                        bars.exit()

                        .transition()

                        .duration(500)

                        .attr("x", w) //choose the bar at the very end

                        .remove(); //remove rectangle

                    svg.selectAll("rect")

                        .data(dataset)

                        bars.enter()

                        .append("rect")

                        .attr("x", w)

                        .attr("y", function(d) {

                            return h - yScale(d);

                        })

                        .merge(bars)

                        .transition()

                        .duration(500)

                        .attr("x", function(d, i){

                                return xScale(i); //change the amounts of bar relative to the dataset

                        })

                        .attr("y", function(d){

                                return yScale(d); //change width of the bar relative to the amount of data in dataset

                        })

                        .attr("width", xScale.bandwidth()) //change the space relative to the dataset

                        .attr("height", function(d){

                                return h - yScale(d); //set the bar height relatively

                        })

                        .attr("fill", "blue") // add color to bars

                        svg.selectAll("rect")

                        .data(dataset)

                        .on("mouseover", function (event, d) {

                            bars.attr("title", "This value is: " + d);

                            //get x and y positions of the bars

                            var xPosition = parseFloat(d3.select(this).attr("x")) +xScale.bandwidth() / 2;

                            var yPosition = parseFloat(d3.select(this).attr("y")) +15

                            svg.append("text")

                            .attr("id","tooltip")

                            .attr("x", xPosition)

                            .attr("y", yPosition)

                            .text(d)

                            .attr("font-family", "sans-serif")

                            .attr("font-size", "12px")

                            .attr("fill", "black")

                            .attr("text-anchor", "middle")

                            .style("font-weight", "bold");

                            d3.select(this).attr("fill", "lightblue");

                        })

                        //mouseout effect for cursor

                        .on("mouseout", function(){

                            d3.select("#tooltip").remove()

                            d3.select(this)

                            .attr("fill", "blue");

                        });

    }

    //sort function

    function Sort() {

        //check sort status

        if(sortStatus == "ascending"){

            svg.selectAll("rect")

            //sort the bars in descending order

            .sort(function(a, b){

                return d3.descending(a, b);

            })

            .transition()

            .duration(500)

            .attr("x", function(d, i){

                return xScale(i);

            })

            sortStatus = "descending";

        }

        else {

            svg.selectAll("rect")

            .sort(function(a, b){

                //sort the bars in ascending order

                return d3.ascending(a, b);

            })

            .transition()

            .duration(500)

            .attr("x", function(d, i){

                return xScale(i);

            })

            sortStatus = "ascending"; //change sort status

        }

    }

}

window.onload = init;