Lab 6_2.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="description" content="Data Visualization"/>
    <meta name="keywords" content="HTML, CSS, D3"/>
    <meta name="author" content="Dai Vy"/>
    <title>Task 6.2 D3 Interactivity - Sort</title>

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

    </style>
</head>
<body>

    <button id="add">Add</button>
    <button id="remove">Remove</button>
    <button id="sort">Sort</button>
    <h1>LAB 6.2 D3 Interactivity - Sort</h1>

    <script>
        var w = 500;
        var h = 100;
        var maxValue = 25;
        var dataset = [24, 10, 29, 19, 8, 15, 20, 12, 9, 6, 21, 28];
        var duration = 2000;

        // Use scaleBand() to create an ordinal scaleable x-axis based on the range of the
data set.
        var xScale = d3.scaleBand()
                .domain(d3.range(dataset.length))
                .rangeRound([0, w])
                .paddingInner(0.05);

        // Use scaleLinear() to create a linear scaleable y-axis based on the range of the
data set.
        var yScale = d3.scaleLinear()
                .domain([0, d3.max(dataset)])
                .range([0, h]);

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

        // Draw initial bars
        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", "steelblue")
            // Mouseover effect
            .on("mouseover", function(event, d) {
                d3.select(this)
                  .transition()
                  .duration(200)
                  .attr("fill", "orange");
                // SVG tooltip
                svg.append("text")
                  .attr("class", "svg-tooltip")
                  .attr("x", +d3.select(this).attr("x") + xScale.bandwidth()/2)
                  .attr("y", +d3.select(this).attr("y") - 8)
                  .attr("text-anchor", "middle")
                  .attr("font-size", "14px")
                  .attr("font-weight", "bold")
                  .attr("fill", "black")
                  .text(d);
            })
            .on("mouseout", function(event, d) {
                d3.select(this)
                  .transition()
                  .duration(200)
                  .attr("fill", "steelblue");
                svg.selectAll(".svg-tooltip").remove();
            });

        // Draw initial labels
        svg.selectAll("text")
            .data(dataset)
            .enter()
            .append("text")
            .text(function(d) {
                return d; // This displays the number on each bar
            })
            .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")
            .attr("font-size", "12px");

        // Add bar function
        function addBar() {
            var newValue = Math.floor(Math.random() * maxValue);
            dataset.push(newValue);
            xScale.domain(d3.range(dataset.length));
            yScale.domain([0, d3.max(dataset)]);
            var delayStep = duration / dataset.length;

            // Bars
            var bars = svg.selectAll("rect")
                .data(dataset, function(d, i) { return i; });

            // ENTER new bar
            var barsEnter = bars.enter()
                .append("rect")
                .attr("x", w) // start off right edge
                .attr("y", function(d) { return h - yScale(d); })
                .attr("width", xScale.bandwidth())
                .attr("height", function(d) { return yScale(d); })
                .attr("fill", "steelblue")
                // Mouseover effect for new bars
                .on("mouseover", function(event, d) {
                    d3.select(this)
                      .transition()
                      .duration(200)
                      .attr("fill", "orange");
                    svg.append("text")
                      .attr("class", "svg-tooltip")
                      .attr("x", +d3.select(this).attr("x") + xScale.bandwidth()/2)
                      .attr("y", +d3.select(this).attr("y") - 8)
                      .attr("text-anchor", "middle")
                      .attr("font-size", "14px")
                      .attr("font-weight", "bold")
                      .attr("fill", "black")
                      .text(d);
                })
                .on("mouseout", function(event, d) {
                    d3.select(this)
                      .transition()
                      .duration(200)
                      .attr("fill", "steelblue");
                    svg.selectAll(".svg-tooltip").remove();
                });
```

```
147
148             // MERGE and transition all bars
149             bars.merge(barsEnter)
150                 .transition()
151                 .duration(duration)
152                 .delay(function(d, i) { return i * delayStep; })
153                 .attr("x", function(d, i) { return xScale(i); })
154                 .attr("y", function(d) { return h - yScale(d); })
155                 .attr("width", xScale.bandwidth())
156                 .attr("height", function(d) { return yScale(d); });
157
158             // Labels
159             var labels = svg.selectAll("text")
160                 .data(dataset, function(d, i) { return i; });
161
162             var labelsEnter = labels.enter()
163                 .append("text")
164                 .text(function(d) { return d; })
165                 .attr("x", w + xScale.bandwidth() / 2)
166                 .attr("y", function(d) { return h - yScale(d) + 14; })
167                 .attr("text-anchor", "middle")
168                 .attr("fill", "white")
169                 .attr("font-size", "12px");
170
171             labels.merge(labelsEnter)
172                 .transition()
173                 .duration(duration)
174                 .delay(function(d, i) { return i * delayStep; })
175                 .text(function(d) { return d; })
176                 .attr("x", function(d, i) { return xScale(i) + xScale.bandwidth() / 2; })
177                 .attr("y", function(d) { return h - yScale(d) + 14; });
178         }
179
180         // Remove bar function
181         function removeBar() {
182             if (dataset.length === 0) return;
183             dataset.shift();
184             xScale.domain(d3.range(dataset.length));
185             yScale.domain([0, d3.max(dataset)]);
186             var delayStep = duration / (dataset.length || 1);
187
188             // Bars
189             var bars = svg.selectAll("rect")
190                 .data(dataset, function(d, i) { return i; });
191
192             // EXIT first bar
193             bars.exit()
194                 .transition()
195                 .duration(duration)
196                 .attr("x", w)
```

```
            .remove();

        // UPDATE remaining bars
        bars.transition()
            .duration(duration)
            .delay(function(d, i) { return i * delayStep; })
            .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); });

        // Re-apply mouseover/mouseout to updated bars
        bars.on("mouseover", function(event, d) {
                d3.select(this)
                  .transition()
                  .duration(200)
                  .attr("fill", "orange");
                svg.append("text")
                  .attr("class", "svg-tooltip")
                  .attr("x", +d3.select(this).attr("x") + xScale.bandwidth()/2)
                  .attr("y", +d3.select(this).attr("y") - 8)
                  .attr("text-anchor", "middle")
                  .attr("font-size", "14px")
                  .attr("font-weight", "bold")
                  .attr("fill", "black")
                  .text(d);
            })
            .on("mouseout", function(event, d) {
                d3.select(this)
                  .transition()
                  .duration(200)
                  .attr("fill", "steelblue");
                svg.selectAll(".svg-tooltip").remove();
            });

        // Labels
        var labels = svg.selectAll("text")
            .data(dataset, function(d, i) { return i; });

        labels.exit()
            .transition()
            .duration(duration)
            .attr("x", w)
            .remove();

        labels.transition()
            .duration(duration)
            .delay(function(d, i) { return i * delayStep; })
            .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; });
        }

        // Sort function
        var sortAscending = true;
        function sortBars() {
            // Create an array of objects with value and original index
            var indexedData = dataset.map(function(d, i) { return {value: d, index: i}; });
            // Sort the array by value
            indexedData.sort(function(a, b) {
                return sortAscending ? d3.ascending(a.value, b.value) :
d3.descending(a.value, b.value);
            });
            // Update xScale domain to new order (array of original indices in sorted order)
            xScale.domain(indexedData.map(function(d) { return d.index; }));
            // Transition bars to new x positions
            svg.selectAll("rect")
                .transition()
                .duration(duration)
                .attr("x", function(d, i) {
                    // Find the new x position for this bar's index
                    var sortedIndex = indexedData.findIndex(function(obj) { return obj.index
=== i; });
                    return xScale(i = indexedData[sortedIndex].index);
                });
            // Transition labels to new x positions
            svg.selectAll("text")
                .transition()
                .duration(duration)
                .attr("x", function(d, i) {
                    var sortedIndex = indexedData.findIndex(function(obj) { return obj.index
=== i; });
                    return xScale(i = indexedData[sortedIndex].index) + xScale.bandwidth() /
2;
                });
            // Toggle sort order for next click
            sortAscending = !sortAscending;
        }
        d3.select("#sort").on("click", sortBars);

        // Button event listeners
        d3.select("#add").on("click", addBar);
        d3.select("#remove").on("click", removeBar);
    </script>

    <br>

    <footer style="color: grey">COS30045 Data Visualization<br>
        Dai Vy
    </footer>
</body>
```

```
294 </html>
```