**Report: Bar Chart Implementation Using p5.js**

**Introduction:**

In this project, I explored how to implement various types of bar charts using p5.js, a popular JavaScript library for creative coding. The goal was to create dynamic, configurable bar charts that are visually engaging and easy to interpret. The project involved creating horizontal bar charts, stacked bar charts, and percentage-based stacked bar charts, all of which are highly customizable in terms of appearance and data representation.

**Approach:**

The process started with defining the core structure for the bar charts. I used p5.js's flexible drawing capabilities, such as rect(), fill(), and text() to create the visual elements. The approach was modular, meaning I built a class-based system to ensure the charts were reusable and customizable. This made the implementation more scalable, allowing me to adjust parameters such as color, size, and the number of bars dynamically.

I also used map() to scale data values and fit them into a visual representation within the canvas dimensions. The core challenge was ensuring that the data was represented proportionally in the chart, which I addressed by scaling the heights and widths of bars based on their data values.

**Key Highlights and Findings:**

1. **Dynamic Chart Configuration:** One interesting feature was the ability to modify the chart's appearance in real-time. By adding sliders and input fields, I allowed users to dynamically adjust the chart’s properties, such as the width of bars, the spacing between them, and even the colors used for each segment. This dynamic configurability provided users with an interactive experience and demonstrated the flexibility of p5.js.
2. **Responsive Scaling Using map():** To ensure the bars’ heights and widths corresponded accurately to their data, I utilized p5.js’s map() function. This function proved useful in scaling numerical data to fit within a designated range, ensuring that the charts were not only visually appealing but also accurately represented the data.
3. **Stacked Bar Charts:** The implementation of stacked bar charts was particularly challenging. The concept required breaking each bar into segments, with each segment representing a part of the whole. By iterating over the data for each bar, I was able to position these segments correctly, stacking them from the bottom up. This was particularly interesting as it involved understanding how to divide a whole value into smaller, proportional segments.
4. **Percentage-Based Stacked Bar Charts:** Implementing a 100% stacked bar chart, where each bar represents a percentage of a whole, was a unique challenge. It required transforming raw data into percentage values before visualizing them. This approach allowed the chart to focus on proportional relationships, providing a clear and immediate understanding of data composition.
5. **Color Coding for Clarity:** Another useful aspect was the use of color coding for different segments of the bar charts. By assigning different colors to various data categories within each bar, I made the charts more readable and visually distinct. This was essential in stacked bar charts, where distinguishing between multiple data points within a single bar is crucial.

**Conclusion:**

This project helped me better understand how to visualize data using p5.js and JavaScript. The experience highlighted the importance of scalability and flexibility when designing data visualizations. I learned to leverage p5.js's powerful drawing tools and functions to create interactive, informative, and aesthetically pleasing bar charts. Overall, this project allowed me to experiment with different data visualization techniques, which will be useful for future projects

Git hub repository

<https://github.com/Morgueniadt/creative-coding-labs>