ANALYSIS OF ALGORITHM
IA-2

MERGESORT VISUALISER

NAME & ROLL NO.

SHREYA MENON :- 16010123324
SHREYANS TATIYA :- 16010123325
SIDDHANT RAUT :- 16010123331

Date of Submission : 19-04-2025
Batch : E-2

Introduction

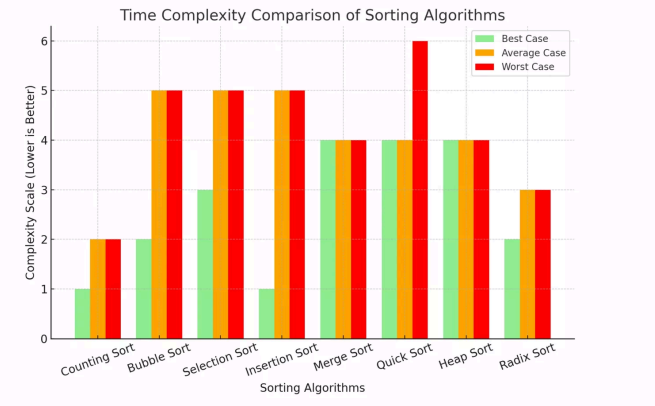
What is merge sort?

Merge Sort is a classic divide-and-conquer sorting algorithm that divides an array into halves, recursively sorts each half, and then merges the sorted halves together. It's known for its **O(n log n)** time complexity and **stable sorting behavior**.

Analysis

Merge sort is a stable, efficient algorithm with O(n log n) time complexity in all cases (average/worst), solving the recurrence $T(n) = 2T(n/2) + n$. Its worst-case comparisons range between $n \log n - n + 1$ and $n \log n + n$, with the best case using ~50% fewer steps. On average, it saves $\alpha \cdot n$ comparisons ($\alpha \approx 0.2645$) versus the worst case.

Compared to quicksort, merge sort uses 39% fewer comparisons in its worst case than quicksort's average, while matching quicksort's best-case move complexity (O(n log n)). Though not in-place (requiring O(n) extra space, reducible to n/2), it excels with sequential data access, making it popular in languages like Lisp.



MergeSort Visualizer

[Visualizer Link]
[Video Demo]

Merge Sort Visualizer is an **interactive web tool** that shows how the **Merge Sort** algorithm works step-by-step using animations. Instead of just printing numbers in a console, it visually represents an array as a series of bars where the height of each bar corresponds to the value of the element.

As the algorithm sorts the array, the bars change positions, colors, and heights, letting users see how the array is being divided, compared, and merged back together in a sorted order.

Why use a visualizer?

- **Educational**: It helps learners understand how the merge sort algorithm works behind the scenes by breaking down each step.
- **Visual Learning**: Recursion and the merging process become easier to understand when represented visually.
- **Debugging**: A visualizer allows users to observe the algorithm's behavior in real-time, making it easier to analyze and improve performance.

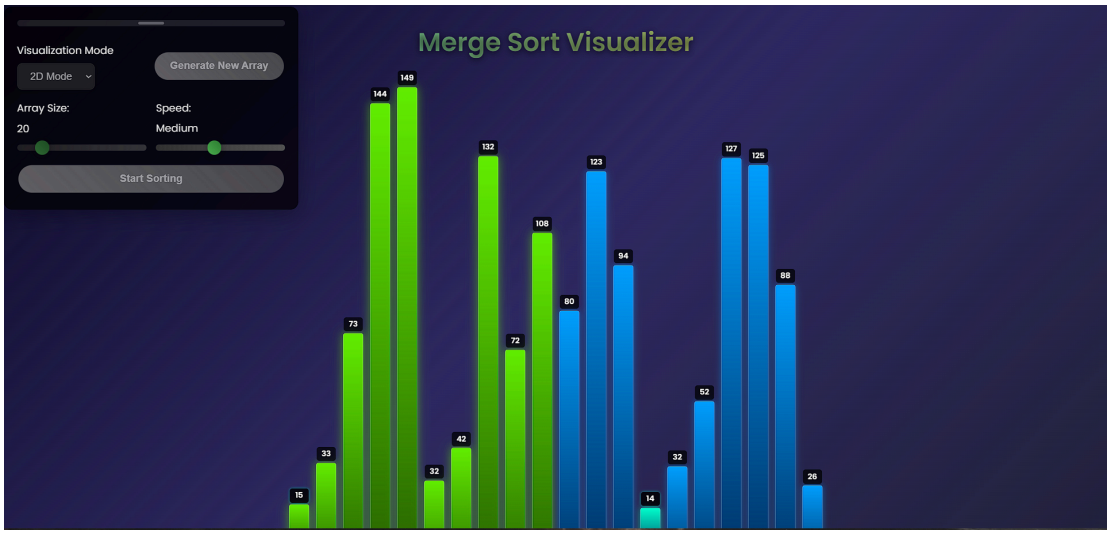
How does it work?

The Merge Sort Visualizer begins by generating a random array, which is displayed as a series of vertical bars where each bar's height represents the value of the corresponding element. When the user initiates the sorting process, the algorithm starts dividing the array into smaller subarrays using a recursive approach. This division is shown through animations where bars shift or change color, helping users visually track the process. Once the smallest units are reached, the merging phase begins, where subarrays are combined back together in sorted order. During this phase, the visualizer updates the bar heights and values in real-time to reflect

the changes. Users have control over the array size and the speed of the animation through sliders, and the entire sorting process is initiated by clicking the "Start Sorting" button.

When the user first loads the page, the generateArray() function is called, which creates an array of random integers and calls renderBars() to display them as vertical bars. Each bar's height is proportional to its value, and the bars are styled with colors and transitions using CSS to enhance the visual feedback.The sorting process begins when the "Start Sorting" button is clicked.

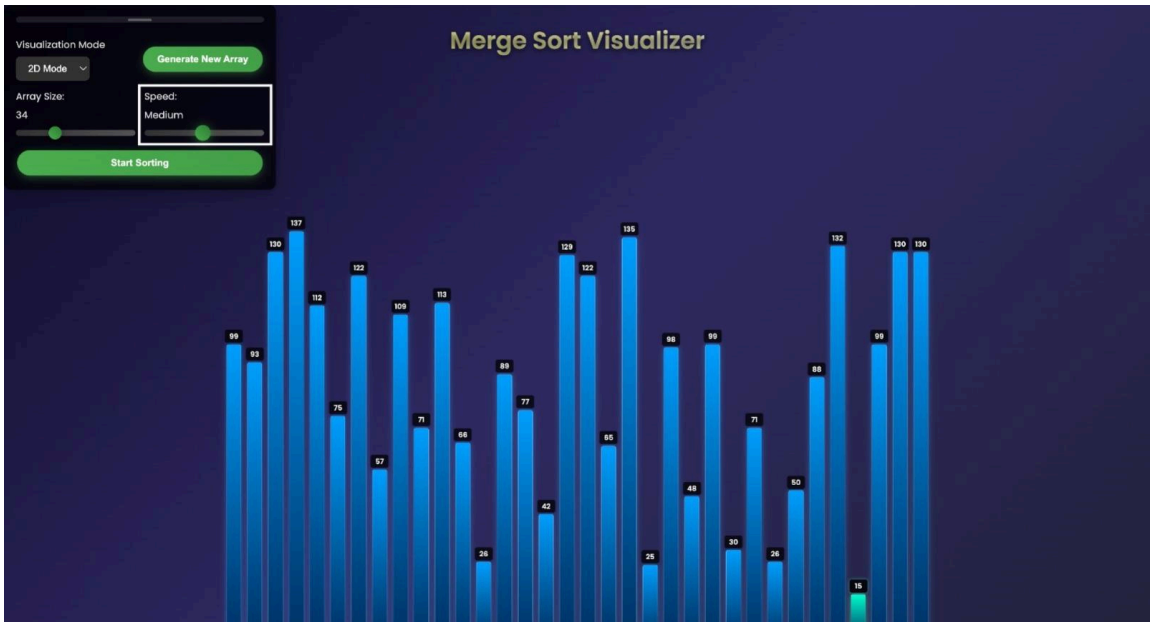
This triggers startMergeSort(), which begins the recursive divide-and-conquer approach of merge sort by calling mergeSort(). This function keeps dividing the array into halves until each portion contains a single element. Once that's done, it starts merging them back together in sorted order.



The merge() function is where the real visualization magic happens. During merging, each part of the array that's being processed is visually highlighted; bars move down and change color to indicate they are being compared or merged. Once elements are merged, their heights and labels are updated to reflect the new, sorted values. The animation delay is controlled by the sleep() function, which pauses execution between each operation to make the process visible to the user.

Features

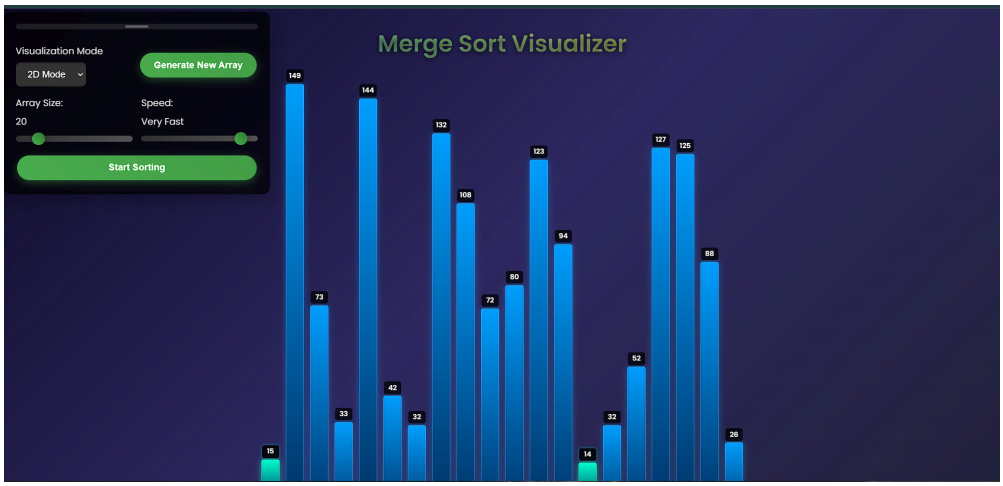
Dynamic Speed Adjustment



The speed of the animation can be adjusted using the slider, which inversely affects the delay — a higher value results in faster animations. Similarly, the size slider allows users to increase or decrease the number of bars displayed, enabling them to test the algorithm with arrays of varying lengths.

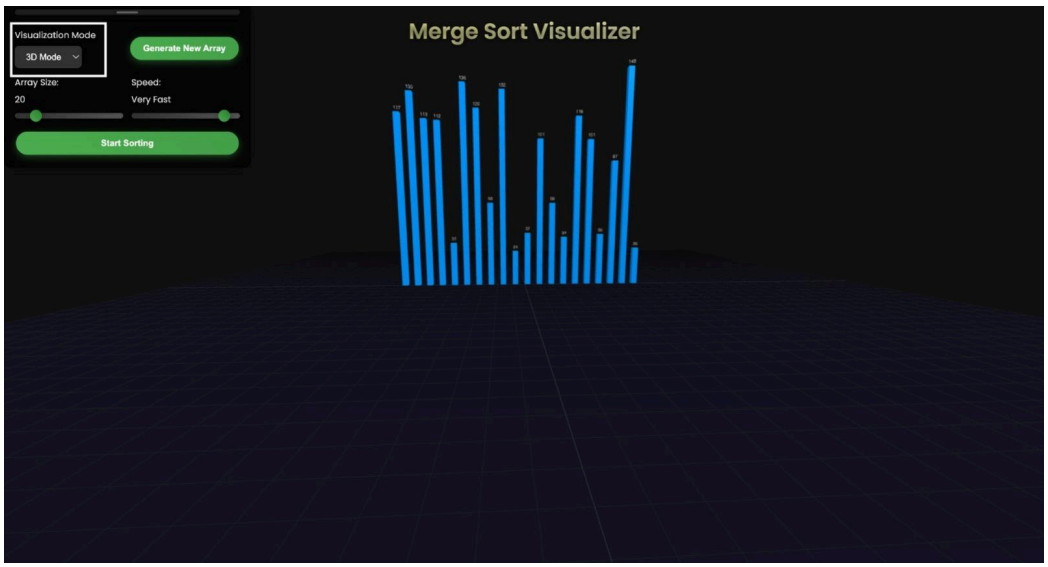
In addition to showcasing how merge sort works, the visualizer includes interactive **visualization modes** to enhance the user experience. The visualization mode can toggle between **2D** and **3D** modes, providing different perspectives on how the array elements are being sorted.

2D Mode

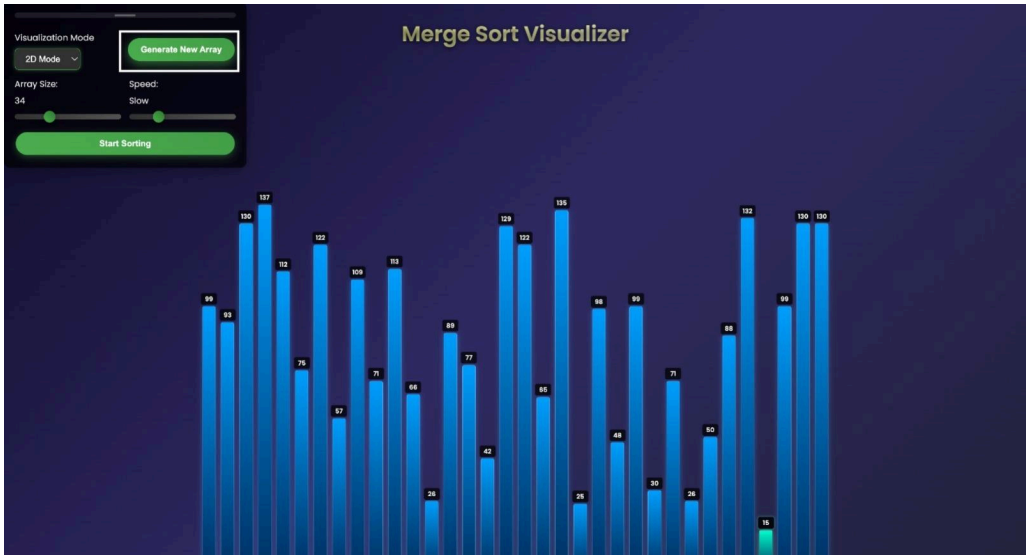


In **2D mode**, the array elements are represented as vertical bars displayed in a flat, 2D space. Each bar's height represents the value of the corresponding array element. The process of dividing and merging is clearly visible as the bars move, change color, and adjust in height as the algorithm runs. This is the default mode for a simple and straightforward view of the sorting process.

3D Mode



Switching to **3D mode** adds depth to the bars, creating a more immersive, three-dimensional view of the array. The bars are now represented as 3D objects, with the height corresponding to the value, and the user can observe how the sorting process works in a more dynamic and spatial environment. This mode adds a layer of visual complexity, allowing users to see the algorithm at work from different angles.

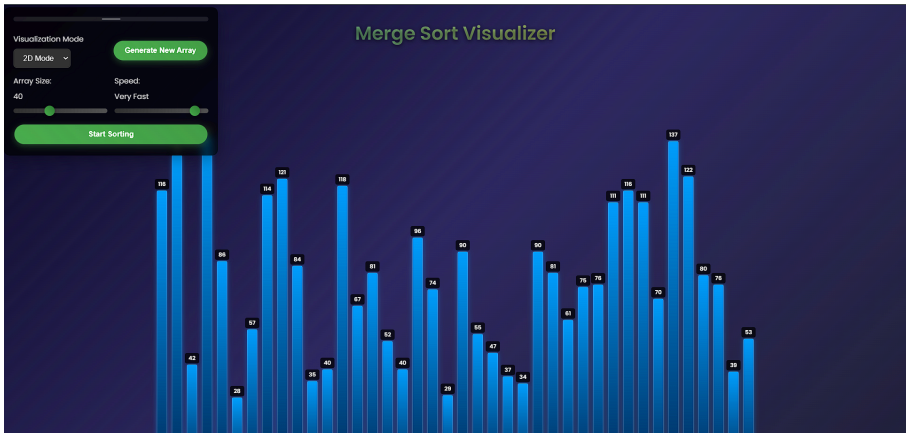


Additionally, the "**Generate New Array**" functionality remains integral to the experience. Every time the Array Size slider is adjusted, it triggers the generateArray() function, which creates a new random array and re-renders the bars accordingly. This gives users the option to test the algorithm with various datasets and switch between 2D and 3D views to compare how the sorting process appears in each mode.

How to Use the Merge Sort Visualizer

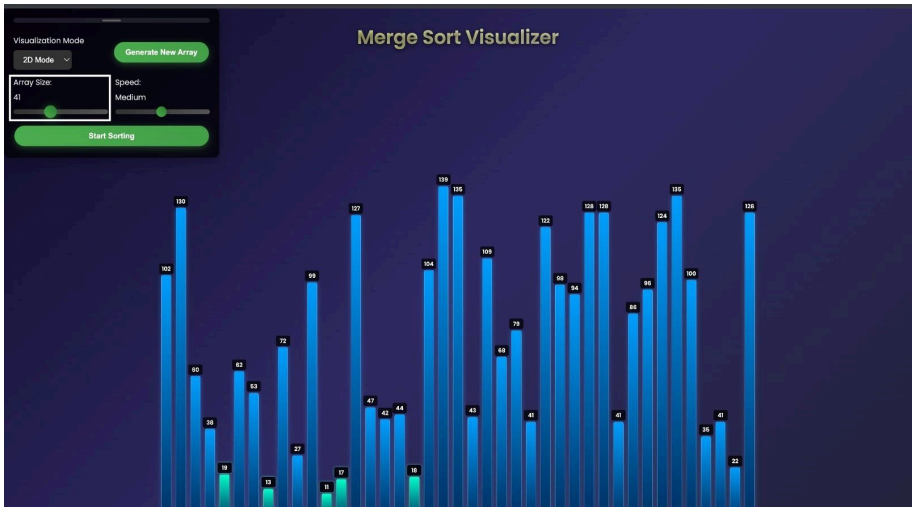
1. Open the Visualizer

Launch the project in a browser. You'll see a header, two sliders, a button, and a set of vertical bars representing the array.



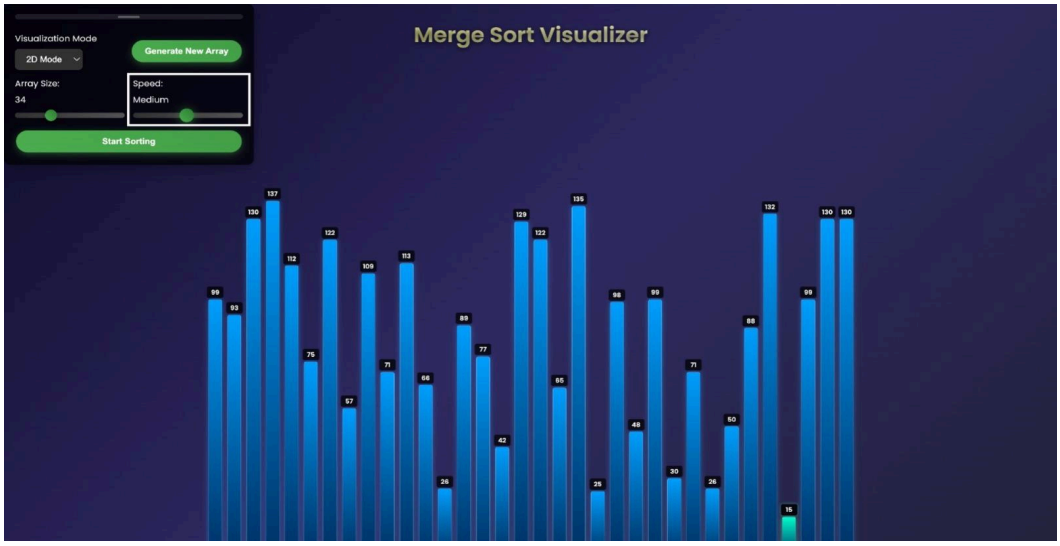
2. Adjust the Array Size

Use the "Array Size" slider to choose how many elements (bars) you want to sort. The bars will automatically regenerate with new random values as you slide.



3. **Set the Animation Speed**

   Move the "Speed" slider to adjust how fast or slow the sorting animation runs. Lower speed values result in slower animations, which are great for learning.



4. **Start the Sorting**

   Click the "Start Sorting" button to begin the merge sort visualization. The bars will animate as the array gets recursively divided and merged back in sorted order.

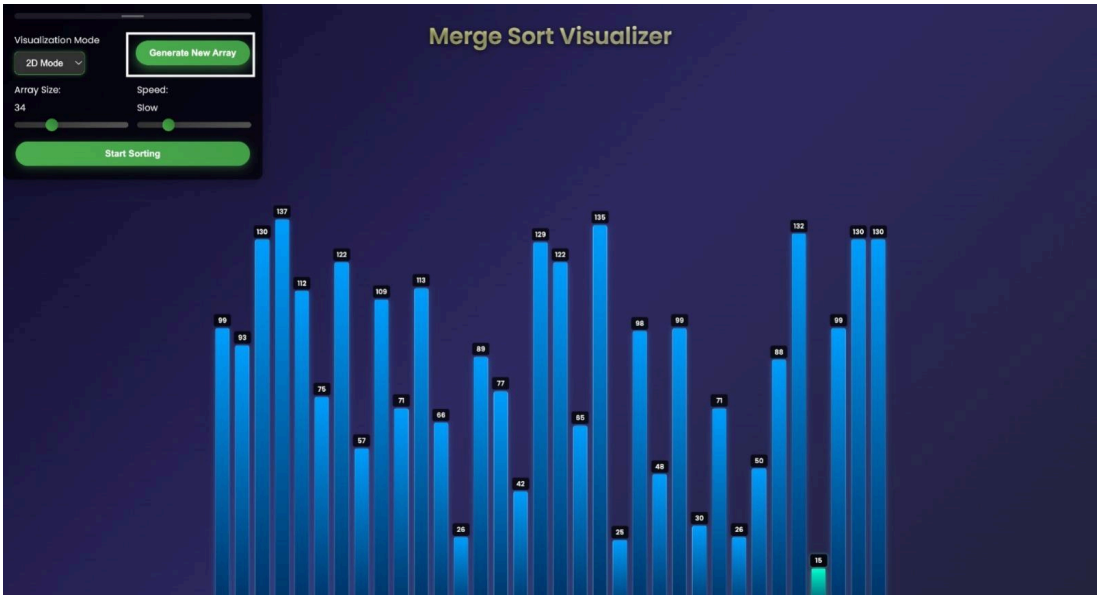
5. **Observe the Process**

   Watch how the algorithm works:

- Bars move or change color to indicate comparison and merging.
- Sorted sections are updated in real-time.

6. **Try again**

   Change the array size or speed again to test different scenarios. You can either refresh the page, adjust the "Array Size" slider to generate a new random array, or click the "Generate New Array" button to create a completely fresh set of data.



Conclusion

The Merge Sort Visualizer serves as a powerful educational tool that transforms a classic algorithm into an interactive and engaging experience. By allowing users to see each step of the sorting process in real-time whether in 2D or 3D this visualizer demystifies recursion and helps users grasp the core concepts of divide-and-conquer sorting.

With adjustable array sizes, animation speeds, and real-time feedback through color-coded bars, users can explore how merge sort behaves under different scenarios. The inclusion of a "Generate New Array" feature ensures that learners can experiment with multiple datasets, making the tool dynamic and versatile.