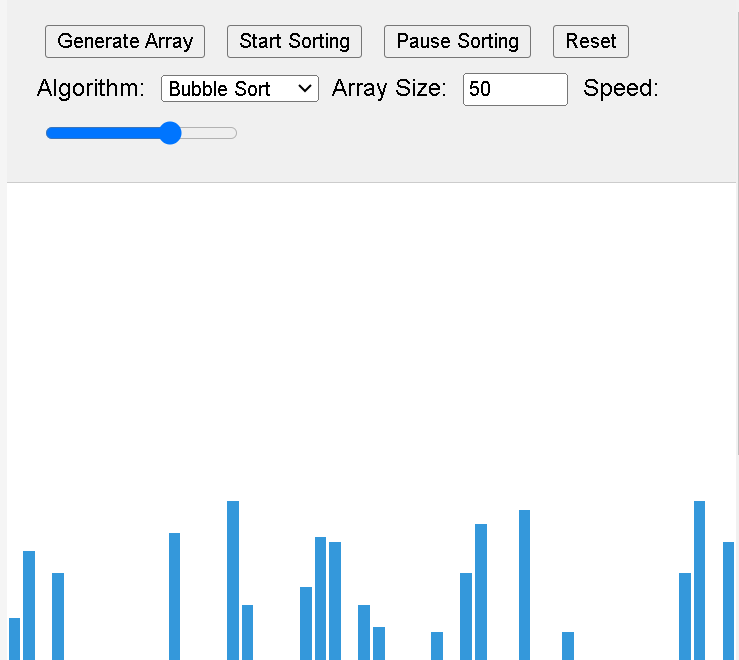
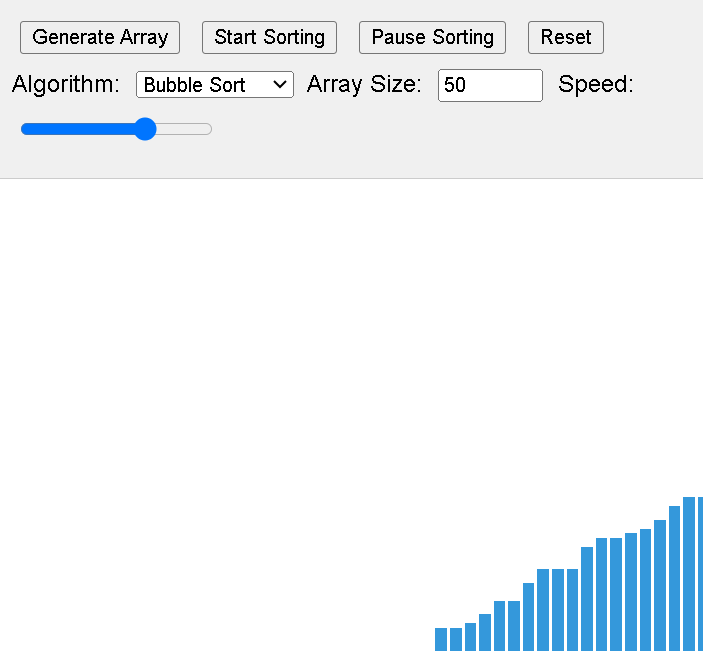
Sorting Visualizer



<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Sorting Visualizer</title>

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

</head>

<body>

<div id="controls">

<button id="generateArray">Generate Array</button>

<button id="startSorting">Start Sorting</button>

<button id="pauseSorting">Pause Sorting</button>

<button id="reset">Reset</button>

<label for="algorithm">Algorithm:</label>

<select id="algorithm">

<option value="bubble">Bubble Sort</option>

<option value="insertion">Insertion Sort</option>

<option value="selection">Selection Sort</option>

<option value="merge">Merge Sort</option>

<option value="quick">Quick Sort</option>

<option value="heap">Heap Sort</option>

</select>

/\*style.css\*/

body {

font-family: Arial, sans-serif;

display: flex;

flex-direction: column;

align-items: center;

margin: 0;

padding: 0;

background-color: #f0f0f0;

}

#controls {

margin: 20px;

}

button, select, input {

margin: 5px;

}

#arrayContainer {

display: flex;

align-items: flex-end;

height: 500px;

width: 100%;

max-width: 800px;

background-color: #fff;

border: 1px solid #ccc;

}

.bar {

width: 10px;

margin: 0 1px;

background-color: #3498db;

transition: height 0.3s ease;

}

<label for="size">Array Size:</label>

<input type="number" id="size" min="5" max="100" value="50">

<label for="speed">Speed:</label>

<input type="range" id="speed" min="1" max="10" value="5">

</div>

<div id="arrayContainer"></div>

<script src="script.js"></script>

</body>

</html>

/\*script.js\*/

let array = [];

let intervalId;

let isSorting = false;

const delay = (ms) => new Promise(resolve => setTimeout(resolve, ms));

const generateArray = (size) => {

array = Array.from({ length: size }, () => Math.floor(Math.random() \* 100) + 1);

renderArray();

};

const renderArray = () => {

const container = document.getElementById('arrayContainer');

container.innerHTML = '';

array.forEach(value => {

const bar = document.createElement('div');

bar.className = 'bar';

bar.style.height = `${value \* 3}px`; // Scale the bar height

container.appendChild(bar);

});

};

const swap = (i, j) => {

[array[i], array[j]] = [array[j], array[i]];

};

const bubbleSort = async () => {

for (let i = 0; i < array.length - 1; i++) {

for (let j = 0; j < array.length - i - 1; j++) {

if (array[j] > array[j + 1]) {

swap(j, j + 1);

renderArray();

await delay(100 / document.getElementById('speed').value);

}

}

}

};

const insertionSort = async () => {

for (let i = 1; i < array.length; i++) {

let key = array[i];

let j = i - 1;

while (j >= 0 && array[j] > key) {

array[j + 1] = array[j];

j--;

renderArray();

await delay(100 / document.getElementById('speed').value);

}

array[j + 1] = key;

renderArray();

}

};

// Add more sorting algorithms here...

const sort = async () => {

if (isSorting) return;

isSorting = true;

const selectedAlgorithm = document.getElementById('algorithm').value;

switch (selectedAlgorithm) {

case 'bubble':

await bubbleSort();

break;

case 'insertion':

await insertionSort();

break;

// Add cases for other algorithms

}

isSorting = false;

};

const startSorting = () => {

if (!isSorting) {

sort();

}

};

const pauseSorting = () => {

if (intervalId) {

clearInterval(intervalId);

isSorting = false;

}

};

const reset = () => {

if (isSorting) pauseSorting();

generateArray(parseInt(document.getElementById('size').value));

};

document.getElementById('generateArray').addEventListener('click', () => {

generateArray(parseInt(document.getElementById('size').value));

});

document.getElementById('startSorting').addEventListener('click', startSorting);

document.getElementById('pauseSorting').addEventListener('click', pauseSorting);

document.getElementById('reset').addEventListener('click', reset);

window.onload = () => {

generateArray(parseInt(document.getElementById('size').value));

};