Lesson 04 Demo 01

Implementing the Bubble Sort Algorithm

Objective: To sort an array using bubble sort in JavaScript, learning basic sorting logic and complexity for handling near-sorted data

Tools required: Visual Studio Code and Node.js

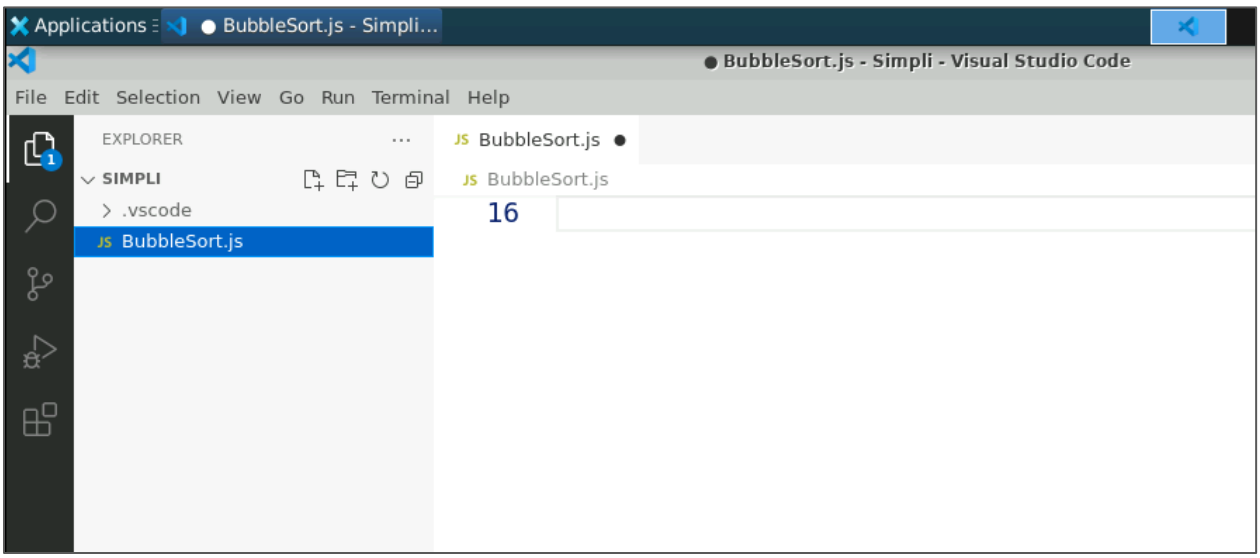
Prerequisites: A basic understanding of arrays and loops in JavaScript

Steps to be followed:

1. Create a JavaScript file and execute it

Step 1: Create a JavaScript file and execute it

1.1 Open the Visual Studio Code editor and create a JavaScript file named **BubbleSort.js**



1.2 Add the following code to the BubbleSort.js file:

```
// Function to perform Bubble Sort on an array
function bubbleSort(array) {
  // Time Complexity: O(n^2) in the worst case
  // Space Complexity: O(1)
  for (let i = 0; i < array.length - 1; i++) {
    for (let j = 0; j < array.length - 1; j++) {
      if (array[j] > array[j + 1]) {
        [array[j], array[j + 1]] = [array[j + 1], array[j]];
      }
    }
  }
  return array;
}

// Example unsorted array
const unsortedArray = [5, 2, 4, 1, 3];

// Measure the execution time using console.time and console.timeEnd
console.time('bubbleSort');
const sortedArray = bubbleSort(unsortedArray);
console.timeEnd('bubbleSort');

console.log('Sorted Array:', sortedArray);
```

BubbleSort.js ●

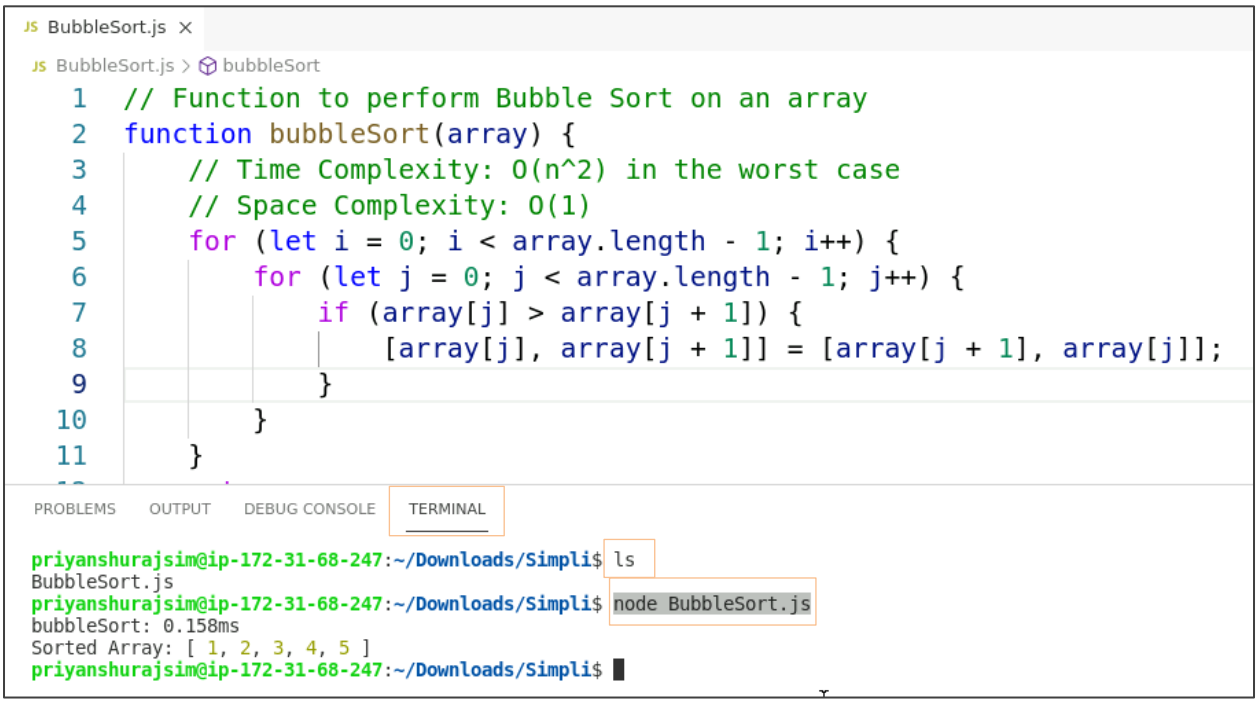
BubbleSort.js > ...

```javascript
// Function to perform Bubble Sort on an array
function bubbleSort(array) {
    // Time Complexity: O(n^2) in the worst case
    // Space Complexity: O(1)
    for (let i = 0; i < array.length - 1; i++) {
        for (let j = 0; j < array.length - 1; j++) {
            if (array[j] > array[j + 1]) {
                [array[j], array[j + 1]] = [array[j + 1], array[j]];
            }
        }
    }
    return array;
}

// Example unsorted array
const unsortedArray = [5, 2, 4, 1, 3];

// Measure the execution time using console.time and console.timeEnd
console.time('bubbleSort');
const sortedArray = bubbleSort(unsortedArray);
console.timeEnd('bubbleSort');

console.log('Sorted Array:', sortedArray);
```

1.3 Press **Ctrl** + **S** to save the file and then execute it in the **TERMINAL** using the following commands:

ls

node BubbleSort.js



By following these steps, you have successfully used bubble sort in JavaScript to sort an array, understood how it works through adjacent swaps, and explored its use in near-sorted data scenarios with $O(n^2)$ time and $O(1)$ space complexity.