

Lesson 04 Demo 08

Implementing the Radix Sort Algorithm

Objective: To sort numeric data using the radix sort algorithm in JavaScript for processing digit-based datasets such as zip codes or invoice numbers

Tools required: Visual Studio Code and Node.js

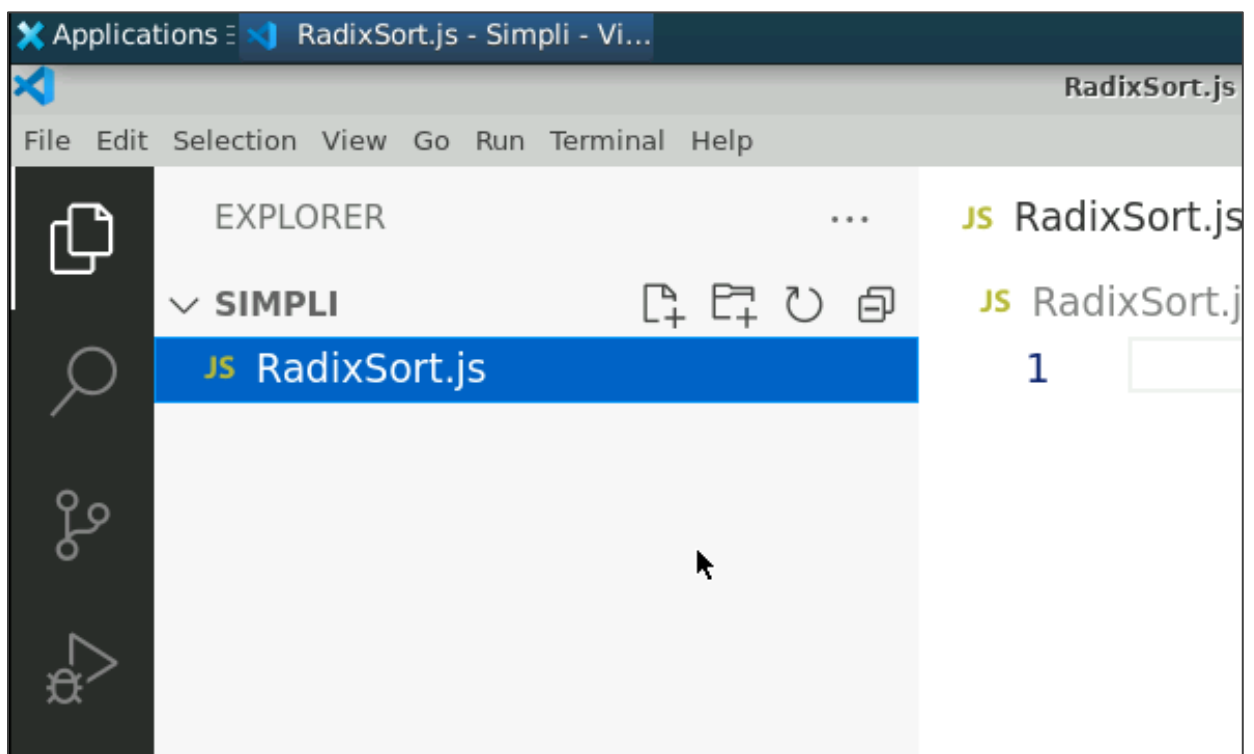
Prerequisites: A basic understanding of arrays and loops in JavaScript

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 **RadixSort.js**



1.2 Add the following code to file:

```
function radixSort(arr) {  
  // Function to get the maximum value in the array  
  const getMax = (arr) => {  
    let max = 0;  
    for (const num of arr) {  
      if (num > max) max = num;  
    }  
    return max;  
  };  
  
  // Function to perform counting sort on the array based on a specific digit (exp)  
  const numberCountSort = (arr, exp) => {  
    const output = new Array(arr.length);  
    const count = new Array(10).fill(0);  
  
    // Count occurrences of digits  
    for (let i = 0; i < arr.length; i++) {  
      count[Math.floor(arr[i] / exp) % 10]++;  
    }  
  
    // Change count[i] so it contains the actual position of this digit in output[]  
    for (let i = 1; i < 10; i++) {  
      count[i] += count[i - 1];  
    }  
  
    // Build the output array  
    for (let i = arr.length - 1; i >= 0; i--) {  
      output[count[Math.floor(arr[i] / exp) % 10] - 1] = arr[i];  
      count[Math.floor(arr[i] / exp) % 10]--;  
    }  
  
    // Copy the output array to arr[], so that arr[] now contains sorted numbers  
    for (let i = 0; i < arr.length; i++) {  
      arr[i] = output[i];  
    }  
  };  
  
  // Get the maximum value in the array  
  const max = getMax(arr);
```

```

// Measure the execution time
console.time('radixSort');

// Do counting sort for every digit (exp)
for (let exp = 1; Math.floor(max / exp) > 0; exp *= 10) {
  numberCountSort(arr, exp);
}

// Measure and log the execution time
console.timeEnd('radixSort');
}

// Example usage
const array = [170, 45, 75, 90, 802, 24, 2, 66];

// Call radixSort function
radixSort(array);

// Log the sorted array
console.log(array);

```

```

function radixSort(arr) {
  // Function to get the maximum value in the array
  const getMax = (arr) => {
    let max = 0;
    for (const num of arr) {
      if (num > max) max = num;
    }
    return max;
  };

  // Function to perform counting sort on the array based on a specific digit (exp)
  const numberCountSort = (arr, exp) => {
    const output = new Array(arr.length);
    const count = new Array(10).fill(0);

    // Count occurrences of digits
    for (let i = 0; i < arr.length; i++) {
      count[Math.floor(arr[i] / exp) % 10]++;
    }
  };
}

```

```

    // Change count[i] so it contains the actual position of this digit in output[]
    for (let i = 1; i < 10; i++) {
        count[i] += count[i - 1];
    }

    // Build the output array
    for (let i = arr.length - 1; i >= 0; i--) {
        output[count[Math.floor(arr[i] / exp) % 10] - 1] = arr[i];
        count[Math.floor(arr[i] / exp) % 10]--;
    }

    // Copy the output array to arr[], so that arr[] now contains sorted numbers
    for (let i = 0; i < arr.length; i++) {
        arr[i] = output[i];
    }
};

```

```

// Get the maximum value in the array
const max = getMax(arr);

// Measure the execution time
console.time('radixSort');

// Do counting sort for every digit (exp)
for (let exp = 1; Math.floor(max / exp) > 0; exp *= 10) {
    numberCountSort(arr, exp);
}

// Measure and log the execution time
console.timeEnd('radixSort');
}

// Example usage
const array = [170, 45, 75, 90, 802, 24, 2, 66];

// Call radixSort function
radixSort(array);

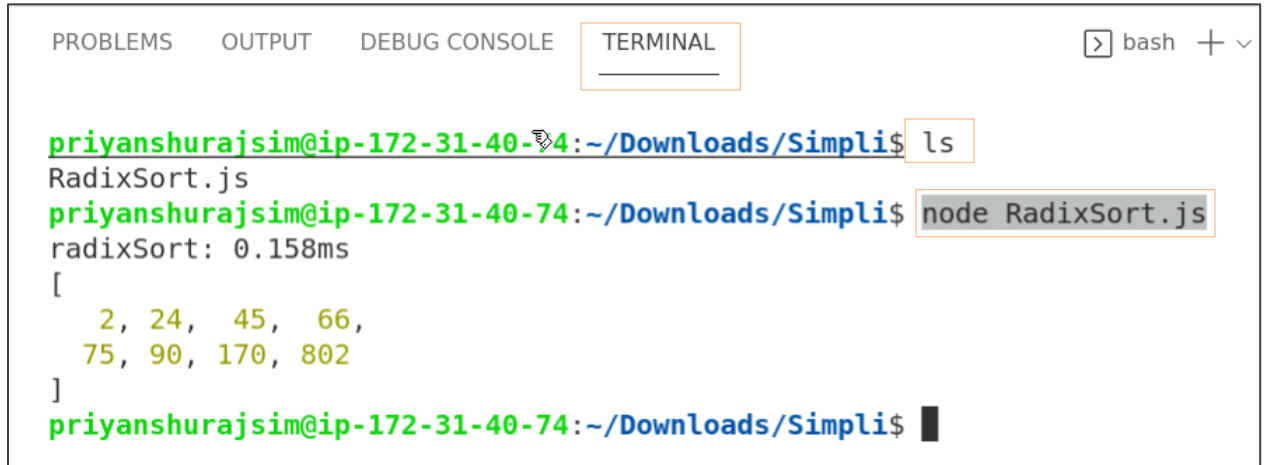
// Log the sorted array
console.log(array);

```

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

ls

node RadixSort.js



The screenshot shows a terminal window with tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL. The TERMINAL tab is active. The prompt is `priyanshurajsim@ip-172-31-40-74:~/Downloads/Simpli$`. The user enters `ls`, and the output is `RadixSort.js`. Then, the user enters `node RadixSort.js`, and the output is `radixSort: 0.158ms` followed by a JSON array: `[2, 24, 45, 66, 75, 90, 170, 802]`. The prompt returns to `priyanshurajsim@ip-172-31-40-74:~/Downloads/Simpli$`.

```
priyanshurajsim@ip-172-31-40-74:~/Downloads/Simpli$ ls
RadixSort.js
priyanshurajsim@ip-172-31-40-74:~/Downloads/Simpli$ node RadixSort.js
radixSort: 0.158ms
[
  2, 24, 45, 66,
  75, 90, 170, 802
]
priyanshurajsim@ip-172-31-40-74:~/Downloads/Simpli$
```

By following these steps, you have successfully implemented and executed the radix sort algorithm in JavaScript, efficiently sorting an array while analyzing its time complexity of $O(d(n + b))$ and space complexity of $O(n + b)$.