

Lesson 04 Demo 08

Implementing Radix Sort Algorithm

Objective: To demonstrate the radix sort algorithm and explain its time and space complexity using JavaScript

Tools required: Visual Studio Code and Node.js

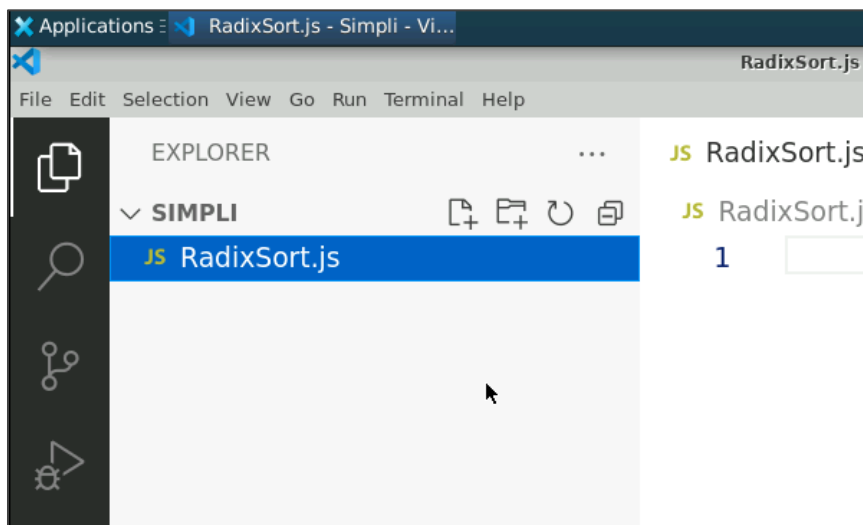
Prerequisites: Basic understanding of arrays and loops in JavaScript

Steps to be followed:

1. Create and execute the JS file

Step 1: Create and execute the JS file

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



1.2 Write the code given below in the **RadixSort.js** 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);
```

```
1  function radixSort(arr) {
2      // Function to get the maximum value in the array
3      const getMax = (arr) => {
4          let max = 0;
5          for (const num of arr) {
6              if (num > max) max = num;
7          }
8          return max;
9      };
10
11     // Function to perform counting sort on the array based on a specific digit (exp)
12     const numberCountSort = (arr, exp) => {
13         const output = new Array(arr.length);
14         const count = new Array(10).fill(0);
15
16         // Count occurrences of digits
17         for (let i = 0; i < arr.length; i++) {
18             count[Math.floor(arr[i] / exp) % 10]++;
19         }
20     };
21 }
```

```

// 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];
}
};

```

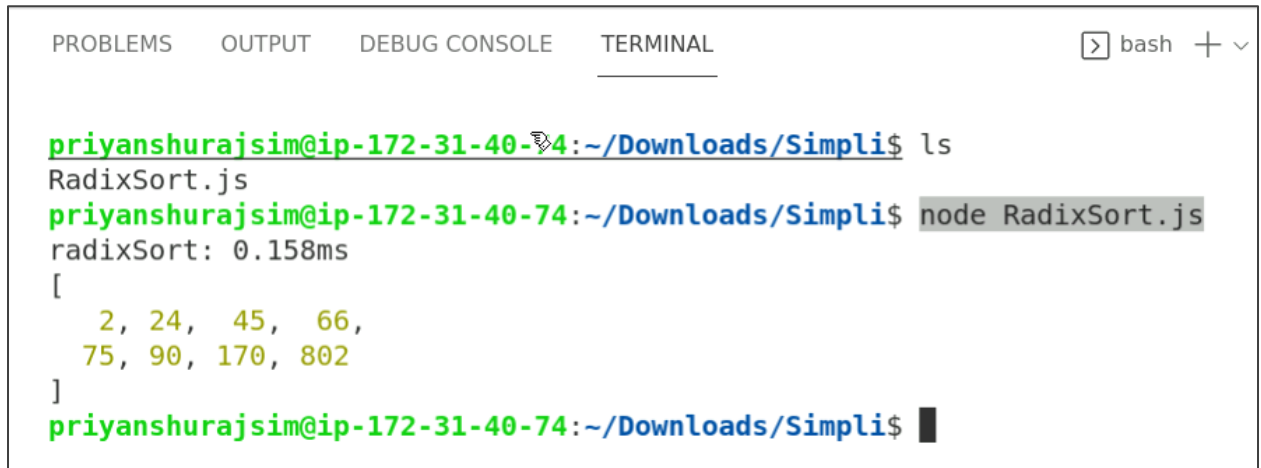
```

38 // Get the maximum value in the array
39 const max = getMax(arr);
40
41 // Measure the execution time
42 console.time('radixSort');
43
44 // Do counting sort for every digit (exp)
45 for (let exp = 1; Math.floor(max / exp) > 0; exp *= 10) {
46     numberCountSort(arr, exp);
47 }
48
49 // Measure and log the execution time
50 console.timeEnd('radixSort');
51 }
52
53 // Example usage
54 const array = [170, 45, 75, 90, 802, 24, 2, 66];
55
56 // Call radixSort function
57 radixSort(array);
58
59 // Log the sorted array
60 console.log(array);

```

1.3 Save the file and execute it in the terminal using the following command:

node RadixSort.js



The screenshot shows a terminal window with tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL. The terminal is running a bash shell. The user has listed the files in the current directory, showing 'RadixSort.js'. Then, they executed 'node RadixSort.js', which output the execution time 'radixSort: 0.158ms' and a sorted array of numbers: [2, 24, 45, 66, 75, 90, 170, 802].

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  bash + v

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$
```

In this example, we used the radix sort algorithm in JavaScript to arrange the items in an array. It has a time complexity of $O(d(n + b))$ and a space complexity of $O(n + b)$.

By following these steps, you have successfully implemented and executed the radix sort algorithm in JavaScript, including measuring its execution time.