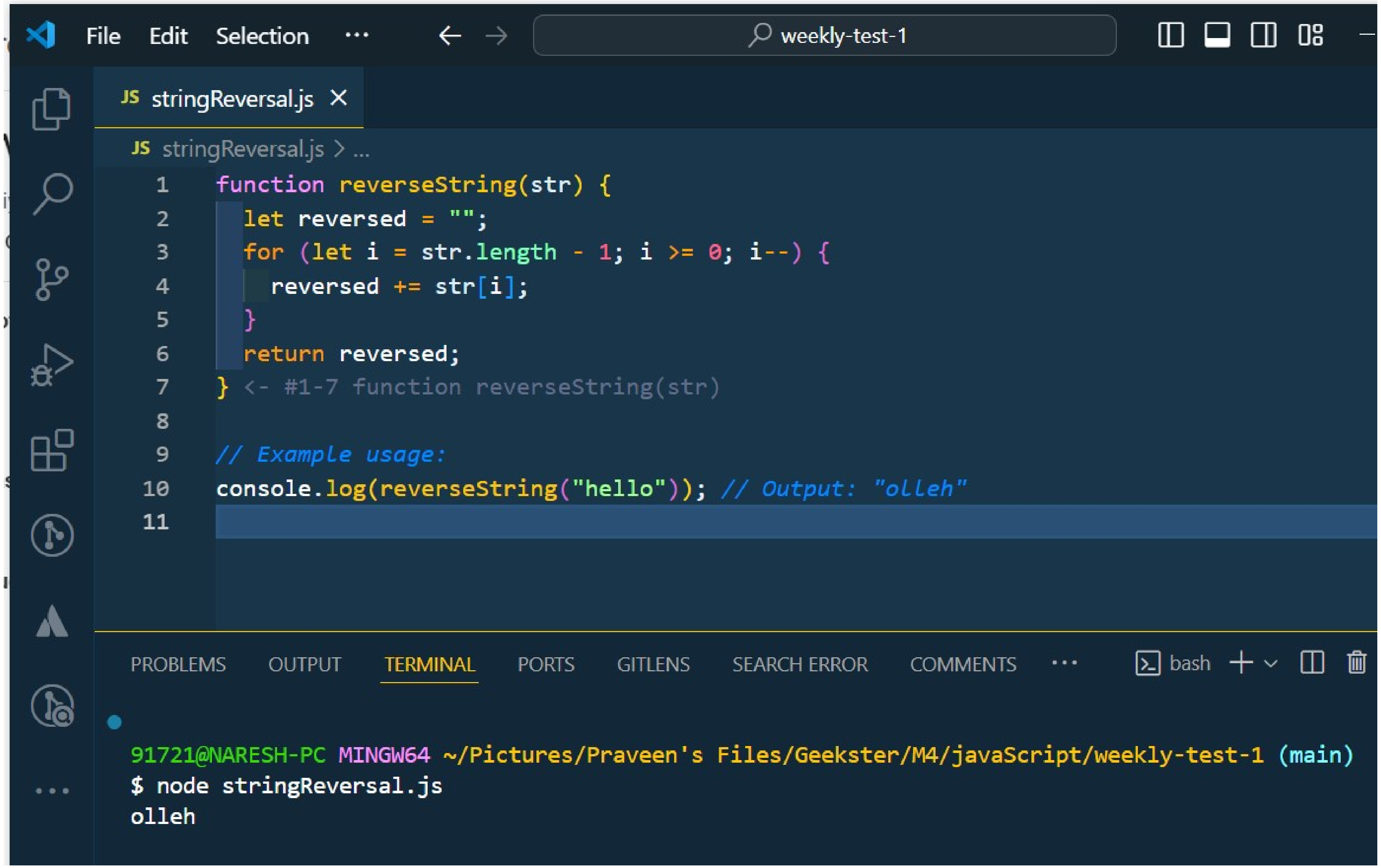
Note : "All the solutions to the questions are provided, and we also have the output of these questions in the terminal."

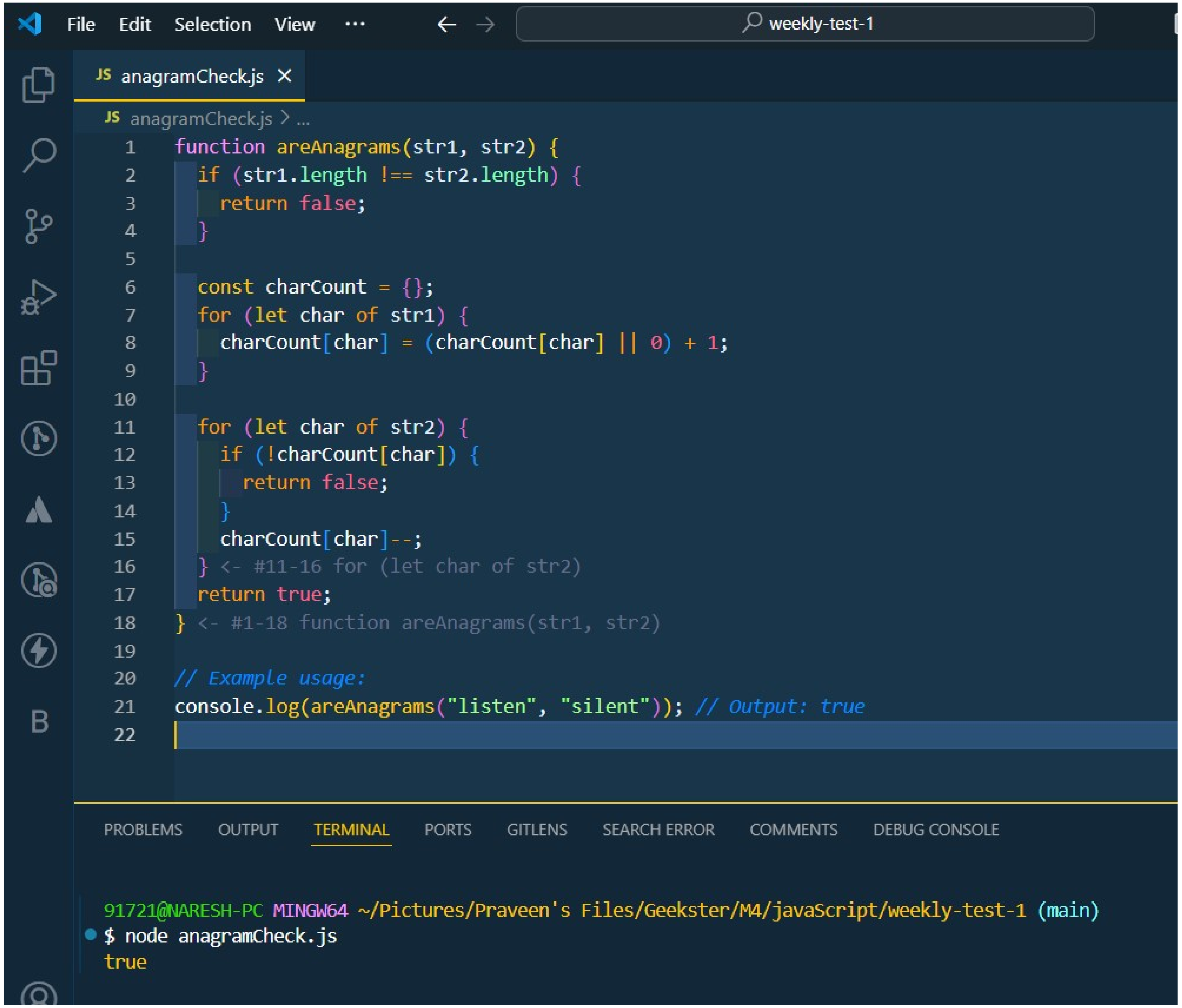
1.) String Reversal :



```javascript
function reverseString(str) {
    let reversed = "";
    for (let i = str.length - 1; i >= 0; i--) {
        reversed += str[i];
    }
    return reversed;
} <- #1-7 function reverseString(str)

// Example usage:
console.log(reverseString("hello")); // Output: "olleh"

```

```
91721@NARESH-PC MINGW64 ~/Pictures/Praveen's Files/Geekster/M4/javaScript/weekly-test-1 (main)
$ node stringReversal.js
olleh
```

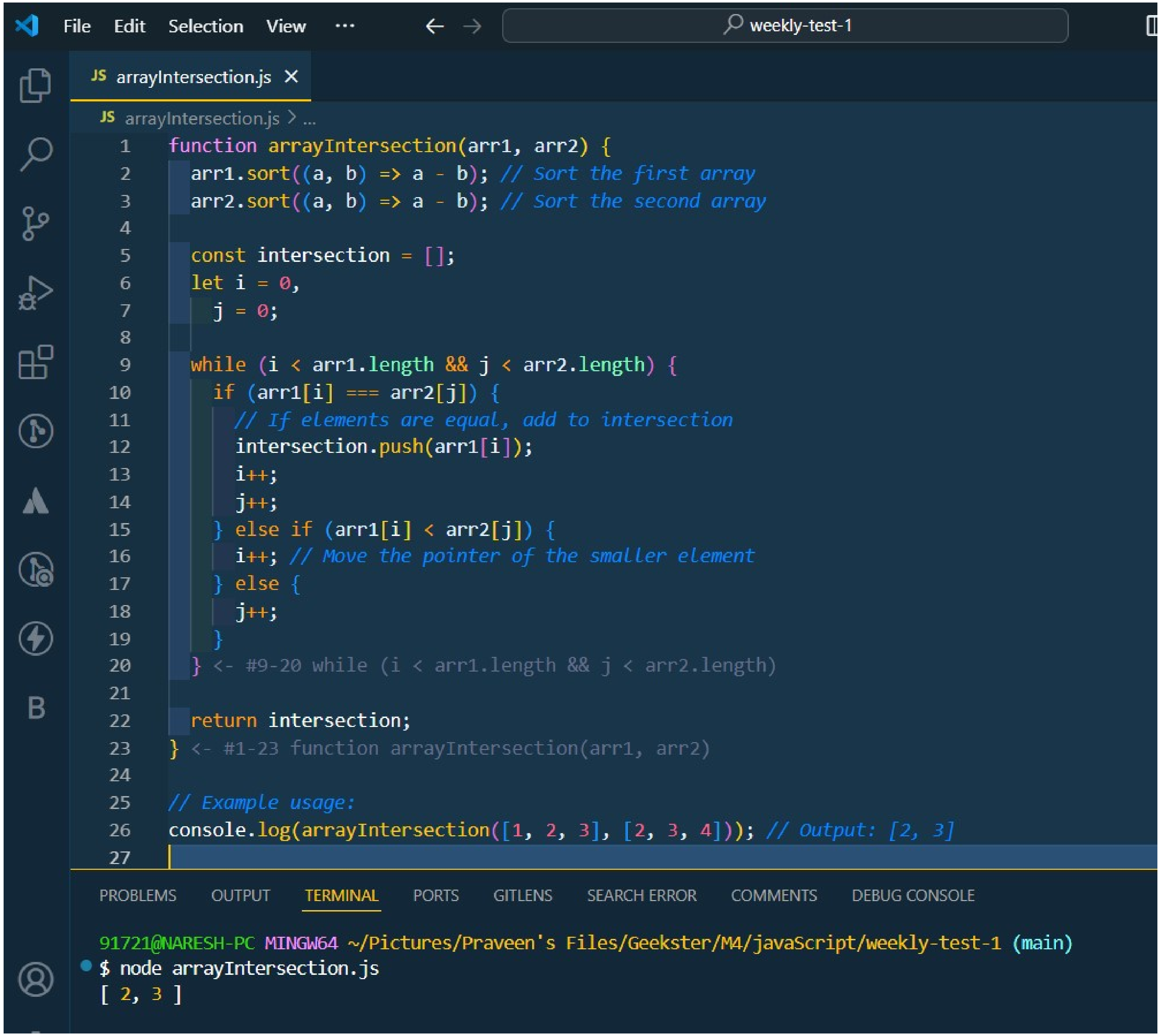
2.) Anagram Check :



```javascript
function areAnagrams(str1, str2) {
  if (str1.length !== str2.length) {
    return false;
  }

  const charCount = {};
  for (let char of str1) {
    charCount[char] = (charCount[char] || 0) + 1;
  }

  for (let char of str2) {
    if (!charCount[char]) {
      return false;
    }
    charCount[char]--;
  } <- #11-16 for (let char of str2)
  return true;
} <- #1-18 function areAnagrams(str1, str2)

// Example usage:
console.log(areAnagrams("listen", "silent")); // Output: true

```

```
91721@NARESH-PC MINGW64 ~/Pictures/Praveen's Files/Geekster/M4/javaScript/weekly-test-1 (main)
$ node anagramCheck.js
true
```

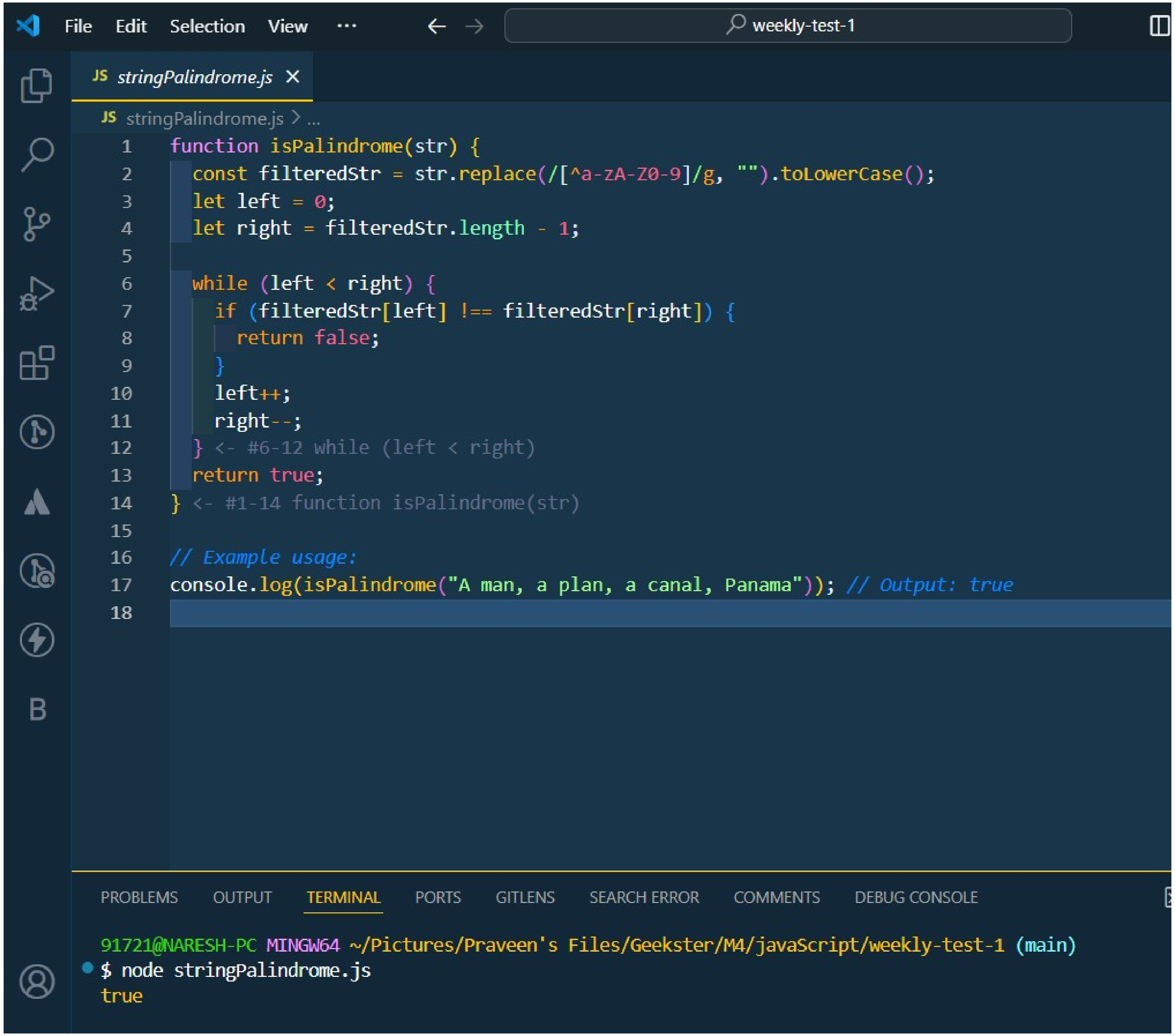
3.) Array Intersection :



```javascript
function arrayIntersection(arr1, arr2) {
  arr1.sort((a, b) => a - b); // Sort the first array
  arr2.sort((a, b) => a - b); // Sort the second array

  const intersection = [];
  let i = 0,
    j = 0;

  while (i < arr1.length && j < arr2.length) {
    if (arr1[i] === arr2[j]) {
      // If elements are equal, add to intersection
      intersection.push(arr1[i]);
      i++;
      j++;
    } else if (arr1[i] < arr2[j]) {
      i++; // Move the pointer of the smaller element
    } else {
      j++;
    }
  } <- #9-20 while (i < arr1.length && j < arr2.length)

  return intersection;
} <- #1-23 function arrayIntersection(arr1, arr2)

// Example usage:
console.log(arrayIntersection([1, 2, 3], [2, 3, 4])); // Output: [2, 3]

```

```
91721@NARESH-PC MINGW64 ~/Pictures/Praveen's Files/Geekster/M4/javaScript/weekly-test-1 (main)
$ node arrayIntersection.js
[ 2, 3 ]
```

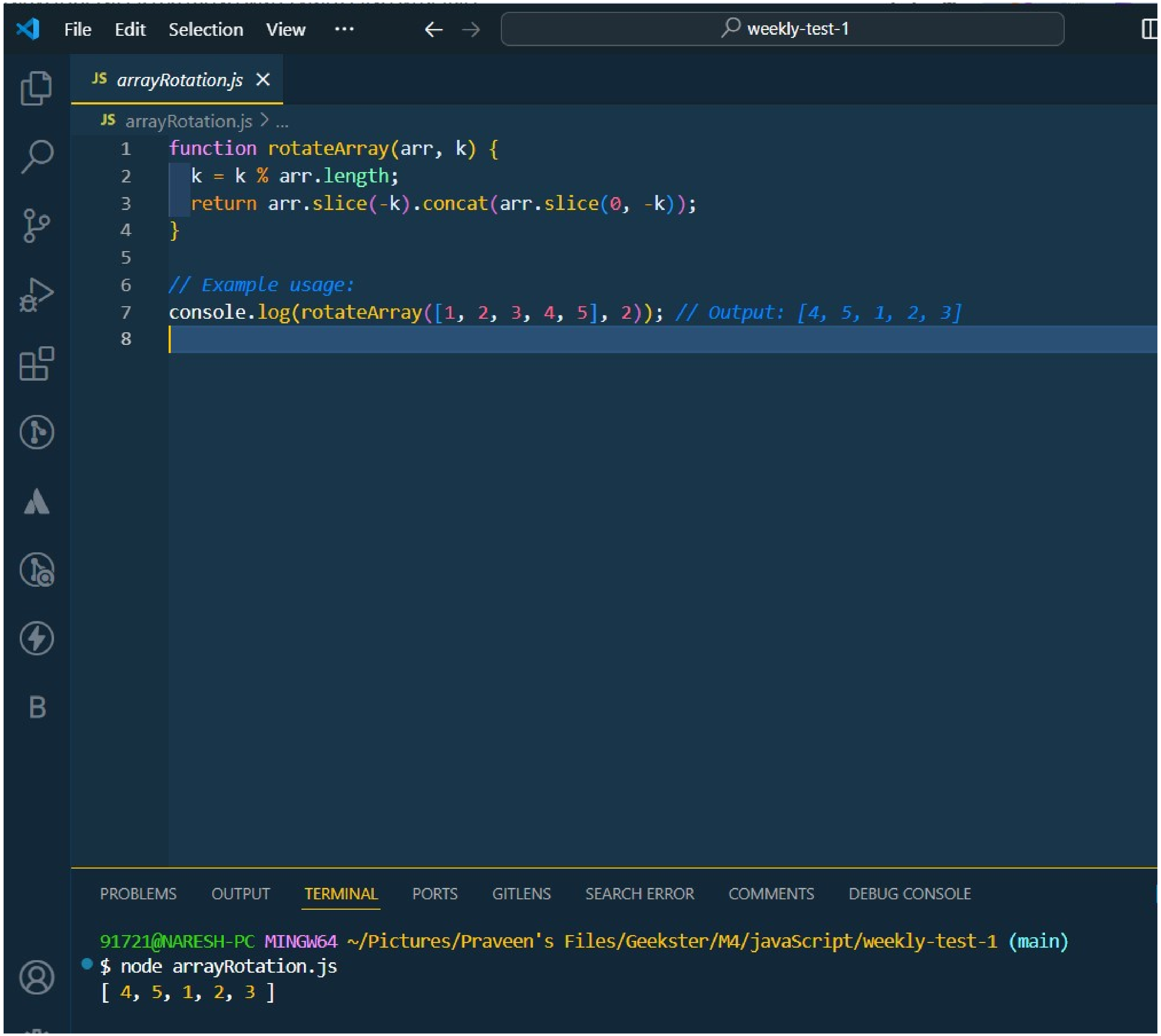
4.) String Palindrome :


File Edit Selection View ···
weekly-test-1
stringPalindrome.js
stringPalindrome.js > ...
PROBLEMS OUTPUT TERMINAL PORTS GITLENS SEARCH ERROR COMMENTS DEBUG CONSOLE

```javascript
function isPalindrome(str) {
  const filteredStr = str.replace(/[^a-zA-Z0-9]/g, "").toLowerCase();
  let left = 0;
  let right = filteredStr.length - 1;

  while (left < right) {
    if (filteredStr[left] !== filteredStr[right]) {
      return false;
    }
    left++;
    right--;
  } <- #6-12 while (left < right)
  return true;
} <- #1-14 function isPalindrome(str)

// Example usage:
console.log(isPalindrome("A man, a plan, a canal, Panama")); // Output: true

```

```
91721@NARESH-PC MINGW64 ~/Pictures/Praveen's Files/Geekster/M4/javaScript/weekly-test-1 (main)
$ node stringPalindrome.js
true
```

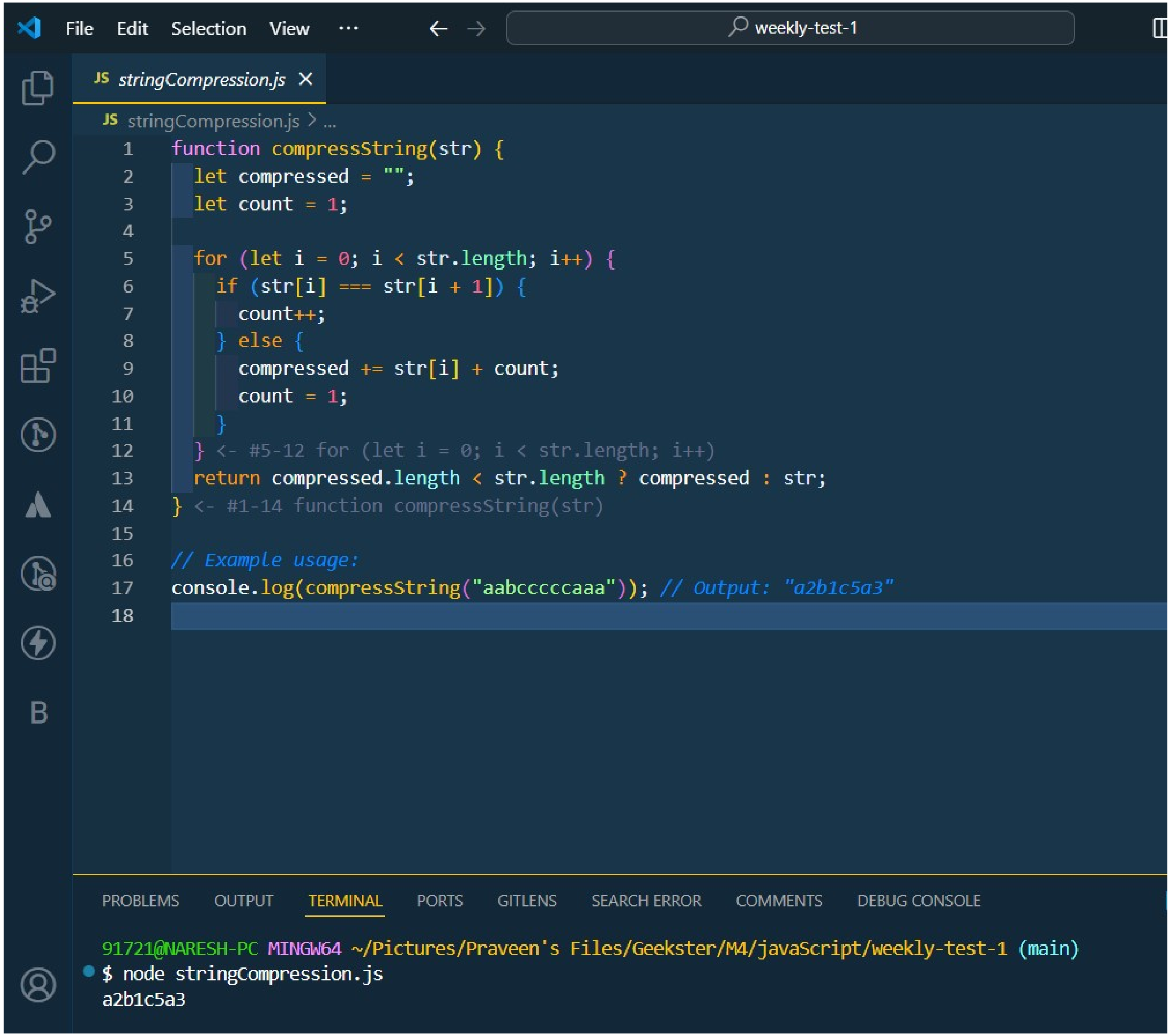
5.) Array Rotation :



```javascript
function rotateArray(arr, k) {
  k = k % arr.length;
  return arr.slice(-k).concat(arr.slice(0, -k));
}

// Example usage:
console.log(rotateArray([1, 2, 3, 4, 5], 2)); // Output: [4, 5, 1, 2, 3]
```

```
91721@NARESH-PC MINGW64 ~/Pictures/Praveen's Files/Geekster/M4/javaScript/weekly-test-1 (main)
$ node arrayRotation.js
[ 4, 5, 1, 2, 3 ]
```

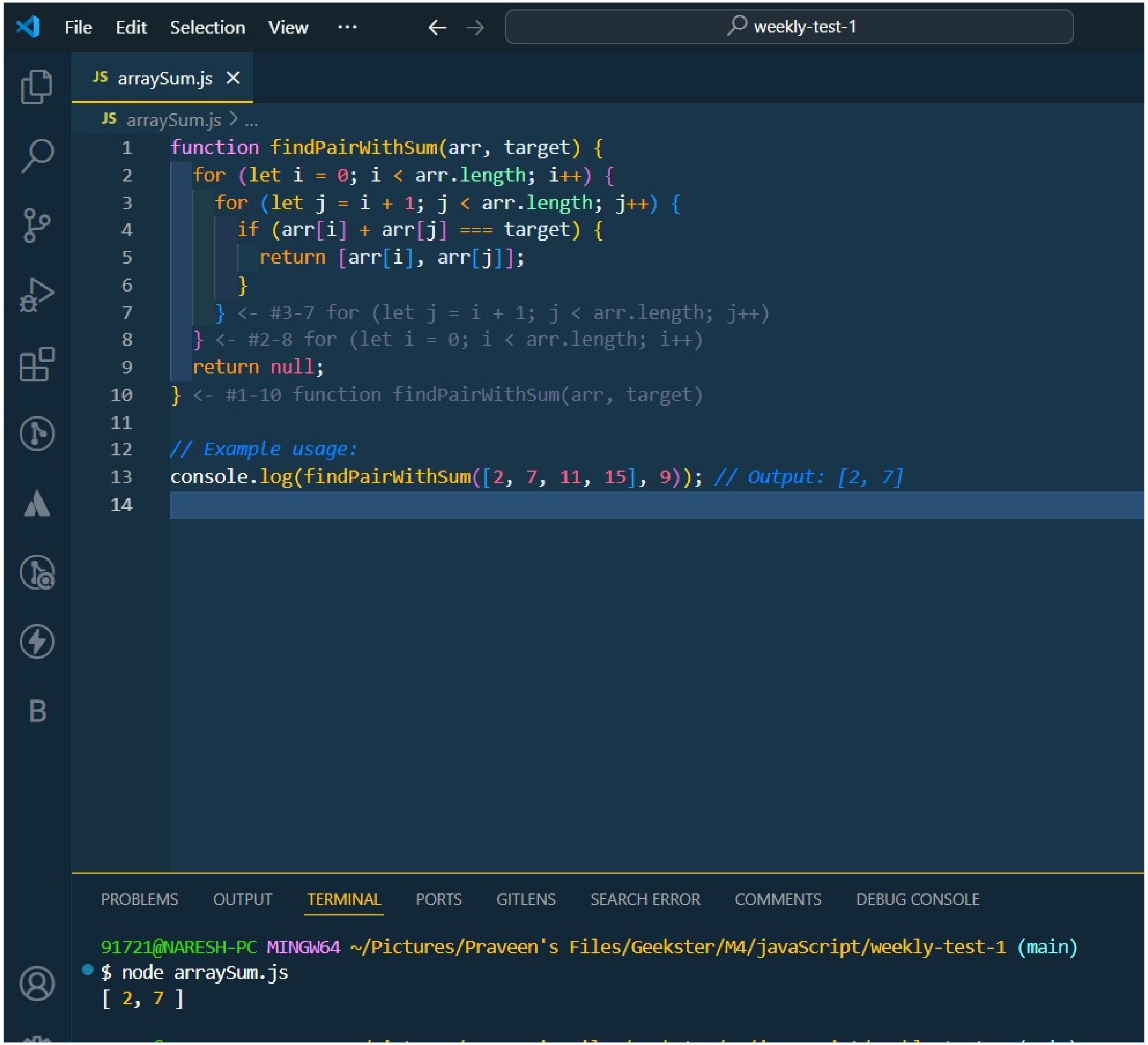
6.) **String Compression :**



```javascript
function compressString(str) {
  let compressed = "";
  let count = 1;

  for (let i = 0; i < str.length; i++) {
    if (str[i] === str[i + 1]) {
      count++;
    } else {
      compressed += str[i] + count;
      count = 1;
    }
  } <- #5-12 for (let i = 0; i < str.length; i++)
  return compressed.length < str.length ? compressed : str;
} <- #1-14 function compressString(str)

// Example usage:
console.log(compressString("aabcccccaaa")); // Output: "a2b1c5a3"
```

```
91721@NARESH-PC MINGW64 ~/Pictures/Praveen's Files/Geekster/M4/javaScript/weekly-test-1 (main)
$ node stringCompression.js
a2b1c5a3
```

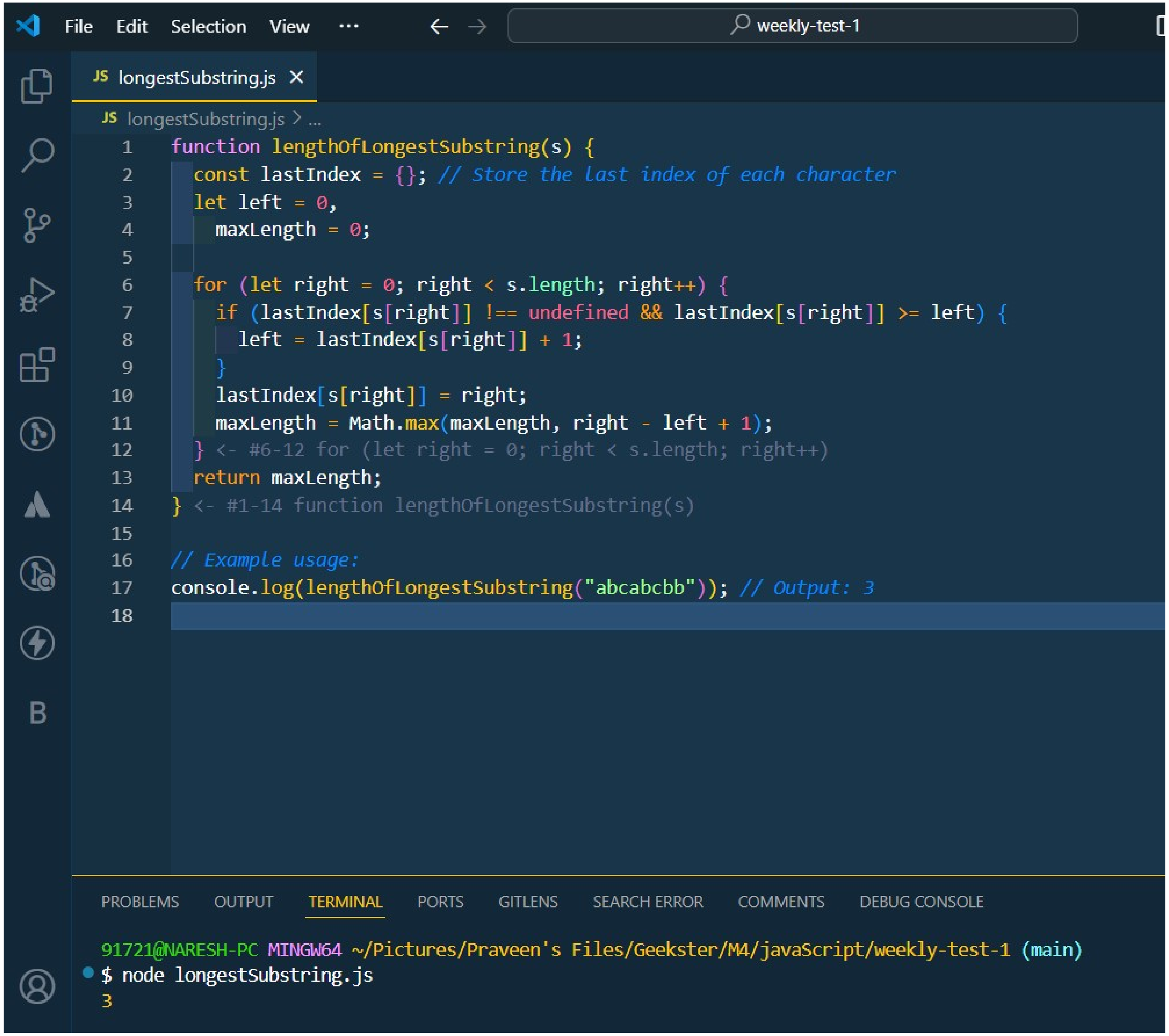
7.) Array Sum :



```javascript
function findPairWithSum(arr, target) {
  for (let i = 0; i < arr.length; i++) {
    for (let j = i + 1; j < arr.length; j++) {
      if (arr[i] + arr[j] === target) {
        return [arr[i], arr[j]];
      }
    } <- #3-7 for (let j = i + 1; j < arr.length; j++)
  } <- #2-8 for (let i = 0; i < arr.length; i++)
  return null;
} <- #1-10 function findPairWithSum(arr, target)

// Example usage:
console.log(findPairWithSum([2, 7, 11, 15], 9)); // Output: [2, 7]

```

```
91721@NARESH-PC MINGW64 ~/Pictures/Praveen's Files/Geekster/M4/javaScript/weekly-test-1 (main)
$ node arraySum.js
[ 2, 7 ]
```

8.) Longest Substring Without Repeating Characters :



```javascript
function lengthOfLongestSubstring(s) {
  const lastIndex = {}; // Store the last index of each character
  let left = 0,
    maxLength = 0;

  for (let right = 0; right < s.length; right++) {
    if (lastIndex[s[right]] !== undefined && lastIndex[s[right]] >= left) {
      left = lastIndex[s[right]] + 1;
    }
    lastIndex[s[right]] = right;
    maxLength = Math.max(maxLength, right - left + 1);
  } <- #6-12 for (let right = 0; right < s.length; right++)
  return maxLength;
} <- #1-14 function lengthOfLongestSubstring(s)

// Example usage:
console.log(lengthOfLongestSubstring("abcabcbb")); // Output: 3
```

```
91721@NARESH-PC MINGW64 ~/Pictures/Praveen's Files/Geekster/M4/javaScript/weekly-test-1 (main)
$ node longestSubstring.js
3
```