.

Lesson 02 Demo 03

Balancing an Array

Objective: To determine whether an even-length array can be split into two halves with equal sums and unique elements in each half, reinforcing practical skills in algorithm development and array evaluation

Tools required: Visual Studio Code and JavaScript

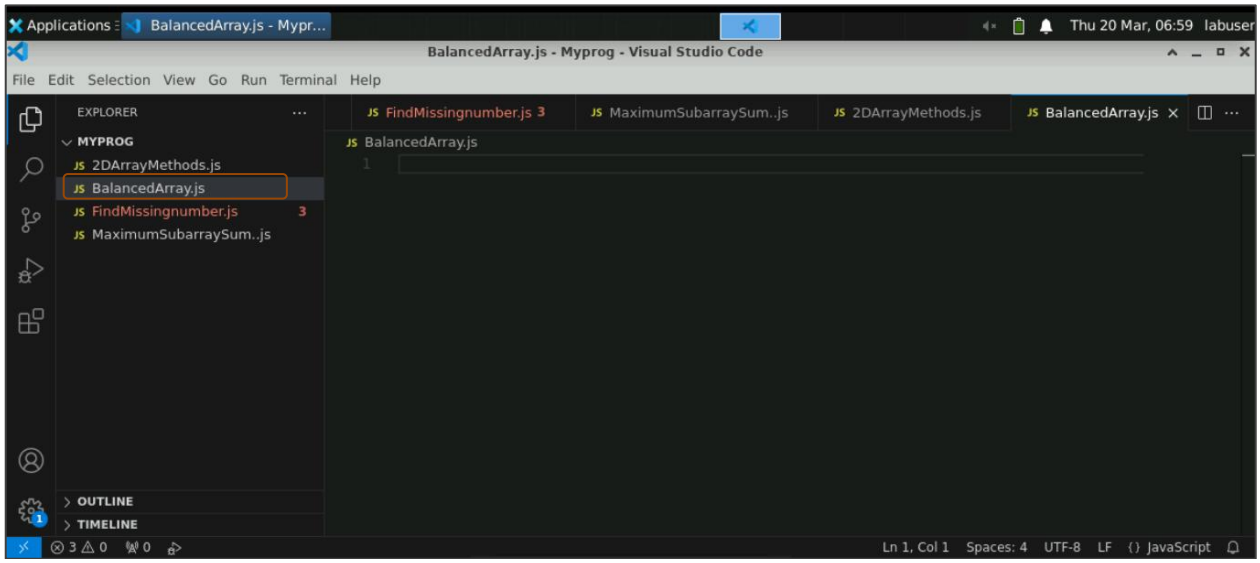
Prerequisites: Completion of Lesson 02 Demo 01

Steps to be followed:

1. Create an algorithm and run it

Step 1: Create an algorithm and run it

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

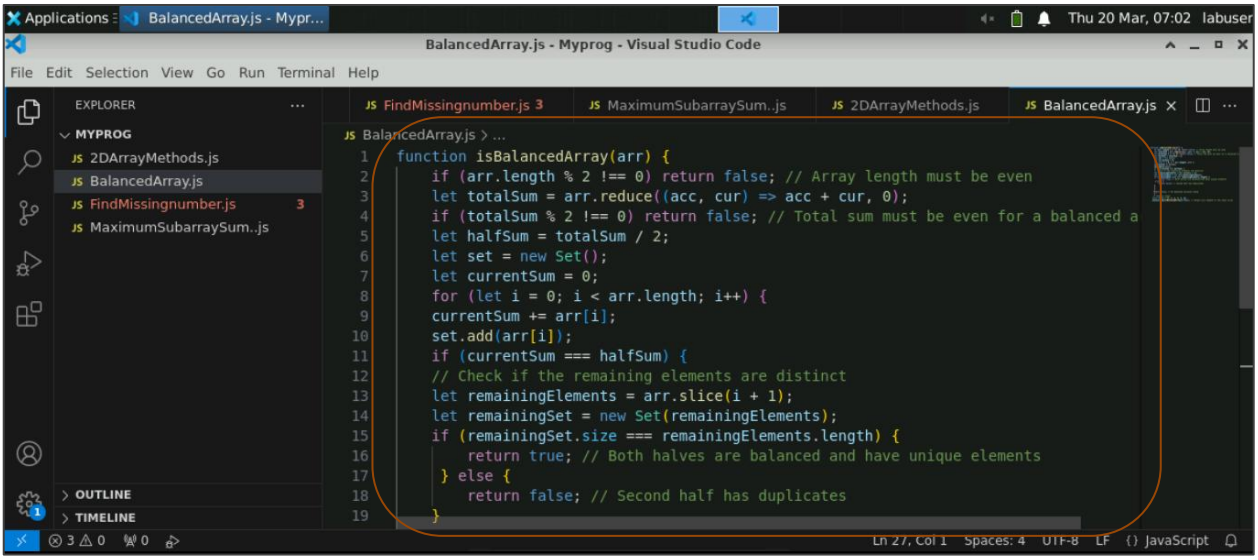


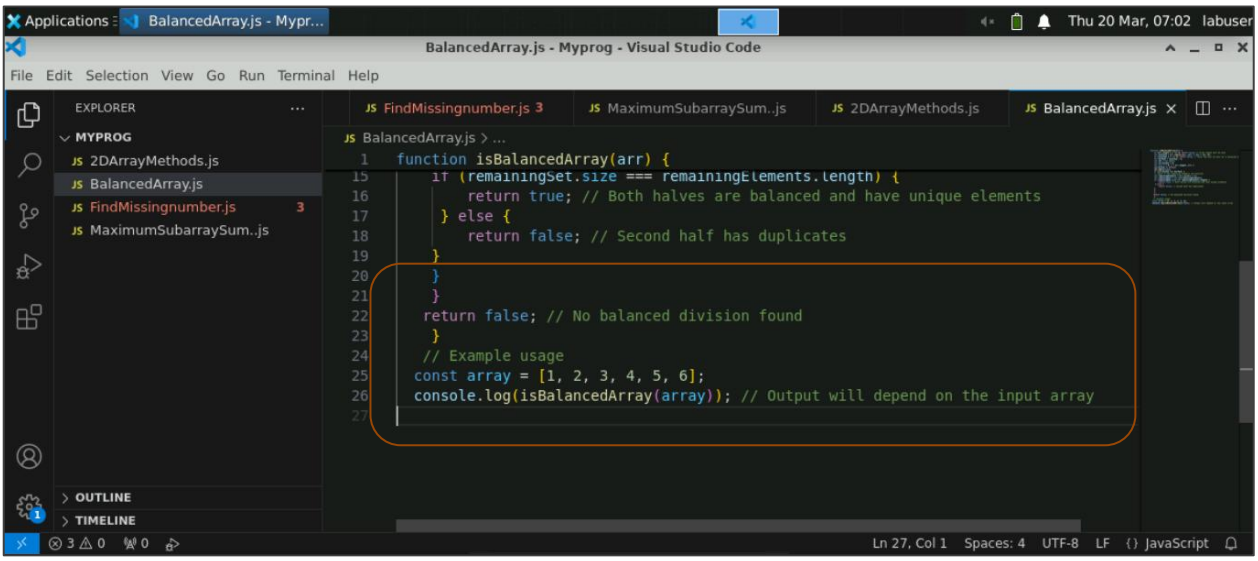
.

1.2 Add the following code to the **BalancedArray.js** file:

```
function isBalancedArray(arr) {
  if (arr.length % 2 !== 0) return false; // Array length must be even
  let totalSum = arr.reduce((acc, cur) => acc + cur, 0);
  if (totalSum % 2 !== 0) return false; // Total sum must be even for a balanced array
  let halfSum = totalSum / 2;
  let set = new Set();
  let currentSum = 0;
  for (let i = 0; i < arr.length; i++) {
  currentSum += arr[i];
  set.add(arr[i]);
  if (currentSum === halfSum) {
  // Check if the remaining elements are distinct
  let remainingElements = arr.slice(i + 1);
  let remainingSet = new Set(remainingElements);
  if (remainingSet.size === remainingElements.length) {
     return true; // Both halves are balanced and have unique elements
   } else {
     return false; // Second half has duplicates
  }
  }
  }
 return false; // No balanced division found
  }
 // Example usage
const array = [1, 2, 3, 4, 5, 6];
console.log(isBalancedArray(array)); // Output will depend on the input array
```

.


Applications
BalancedArray.js - Mypr...
Thu 20 Mar, 07:02 labuser
BalancedArray.js - Myprog - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
MYPROG
2DArrayMethods.js
BalancedArray.js
FindMissingnumber.js 3
MaximumSubarraySum..js
OUTLINE
TIMELINE
FindMissingnumber.js 3
MaximumSubarraySum..js
2DArrayMethods.js
BalancedArray.js
BalancedArray.js > ...
function isBalancedArray(arr) {
if (arr.length % 2 !== 0) return false; // Array length must be even
let totalSum = arr.reduce((acc, cur) => acc + cur, 0);
if (totalSum % 2 !== 0) return false; // Total sum must be even for a balanced a
let halfSum = totalSum / 2;
let set = new Set();
let currentSum = 0;
for (let i = 0; i < arr.length; i++) {
currentSum += arr[i];
set.add(arr[i]);
if (currentSum === halfSum) {
// Check if the remaining elements are distinct
let remainingElements = arr.slice(i + 1);
let remainingSet = new Set(remainingElements);
if (remainingSet.size === remainingElements.length) {
return true; // Both halves are balanced and have unique elements
} else {
return false; // Second half has duplicates
}
Ln 27, Col 1
Spaces: 4
UTF-8
LF
JavaScript


Applications
BalancedArray.js - Mypr...
Thu 20 Mar, 07:02 labuser
BalancedArray.js - Myprog - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
MYPROG
2DArrayMethods.js
BalancedArray.js
FindMissingnumber.js 3
MaximumSubarraySum..js
OUTLINE
TIMELINE
FindMissingnumber.js 3
MaximumSubarraySum..js
2DArrayMethods.js
BalancedArray.js
BalancedArray.js > ...
function isBalancedArray(arr) {
return true; // Both halves are balanced and have unique elements
} else {
return false; // Second half has duplicates
}
}
}
return false; // No balanced division found
}
// Example usage
const array = [1, 2, 3, 4, 5, 6];
console.log(isBalancedArray(array)); // Output will depend on the input array
Ln 27, Col 1
Spaces: 4
UTF-8
LF
JavaScript

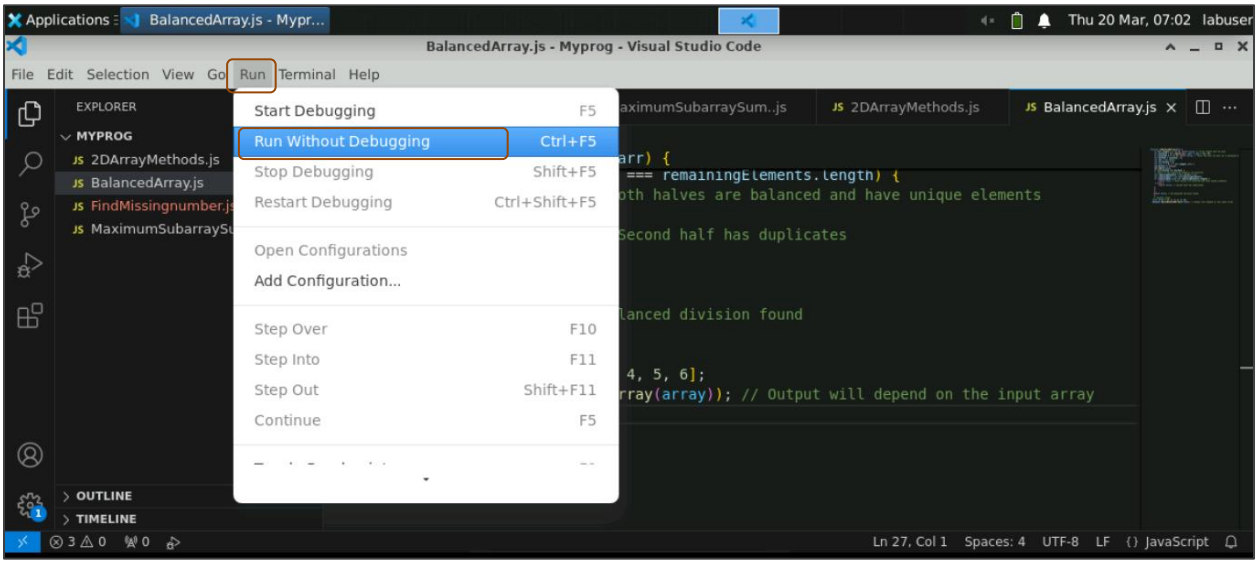
.

Note:

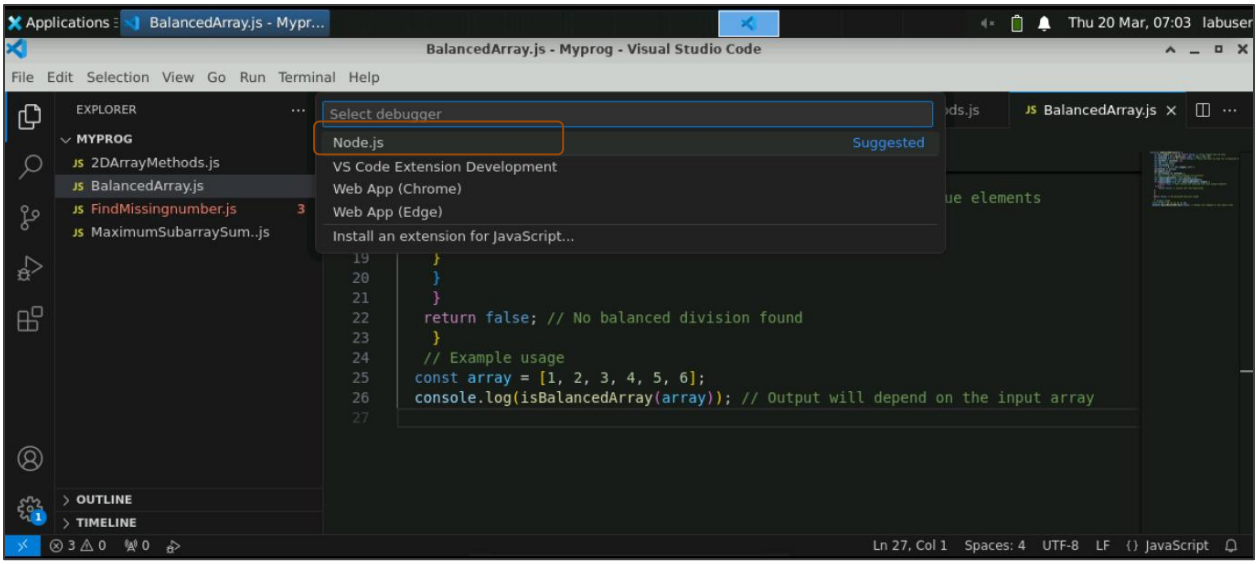
1. First, we check if the total sum of the array is even and the length of the array is even.
2. We calculate the half sum, which is what each half should sum up to for the array to be balanced.
3. As we iterate through the array, we keep adding elements to a set and summing up their values.

If, at any point, the current sum equals the half sum, we check the remaining elements in the array. If these remaining elements are all unique (checked by comparing the size of a set constructed from these elements with the length of these elements), then the array can split into two balanced halves. Otherwise, it cannot.

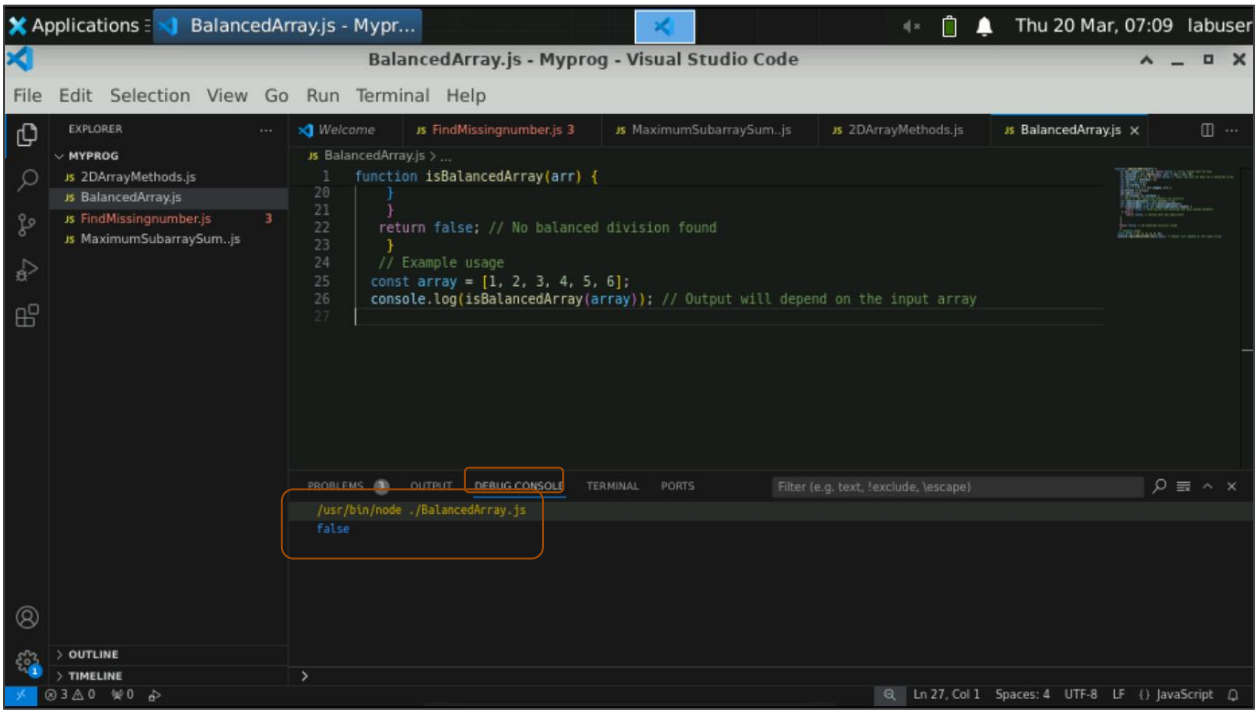
1.3 Click **Run** and then **Run Without Debugging**. Select **Node.js** to check the output in the DEBUG CONSOLE.



.



1.4 View the output in the **DEBUG CONSOLE** as shown below:



By following these steps, you have successfully checked if a given even-length array is balanced, meaning it can be divided into two halves with equal sums and distinct elements. This reinforces practical skills in algorithm development and array evaluation.