TASK

1. Write a note on array methods in javascript with example. **Note:You should find all the methods and try to implement them.**

Explanation:

- Push():The push() method adds a new element to an array (at the end)
  The push() method returns the new array length:

Eg:
let fruits = ['apple', 'banana']; let newLength = fruits.push('orange'); // fruits is now ['apple', 'banana', 'orange'] // newLength is 3

- Pop():Removes the last element from an array and returns that element.
  The pop() method removes the last element from an array.

```
Eg: let fruits = ['apple', 'banana', 'orange'];
    let lastFruit = fruits.pop();
  // fruits is now ['apple', 'banana']
 // lastFruit is 'orange'
```

- Shift() : The shift() method removes the first array element and "shifts" all other elements to a lower index. The shift() method returns the value that was "shifted out"

```
Eg:
   let fruits = ['apple', 'banana', 'orange'];
    let firstFruit = fruits.shift();
   // fruits is now ['banana', 'orange']
    // firstFruit is 'apple'
```

- Unshift(): The unshift() method adds a new element to an array (at the beginning), and "unshifts" older elements:

Eg:

```
let fruits = ['banana', 'orange'];
let newLength = fruits.unshift('apple');
// fruits is now ['apple', 'banana', 'orange']
// newLength is 3
```

- Concat()

The concat() method creates a new array by merging (concatenating) existing arrays.
The concat() method does not change the existing arrays. It always returns a new array.The concat() method can take any number of array arguments.
Eg:

```
let fruits = ['apple'];
let moreFruits = fruits.concat(['banana', 'orange']);

// moreFruits is ['apple', 'banana', 'orange'].
```

- Slice()

  The slice() method slices out a piece of an array into a new array. The slice() method creates a new array.The slice() method does not remove any elements from the source array.

  Example:

  ```
    let fruits = ['apple', 'banana', 'orange', 'peach'];
  let citrus = fruits.slice(2);
  // citrus is ['orange', 'peach'], fruits remains unchanged
  ```

- Splice()

  The splice() method adds new items to an array.

  Changes the contents of an array by removing or replacing existing elements and/or adding new elements.

  Example

  ```
  let fruits = ['apple', 'banana', 'orange'];
  fruits.splice(1, 1, 'grape', 'melon');
  // fruits is now ['apple', 'grape', 'melon', 'orange']
  ```

- forEach()

  Executes a provided function once for each array element.

  Example

  ```
  let fruits = ['apple', 'banana', 'orange'];
   fruits.forEach(function(fruit) { console.log(fruit); });
  // Logs: 'apple', 'banana', 'orange'
  ```

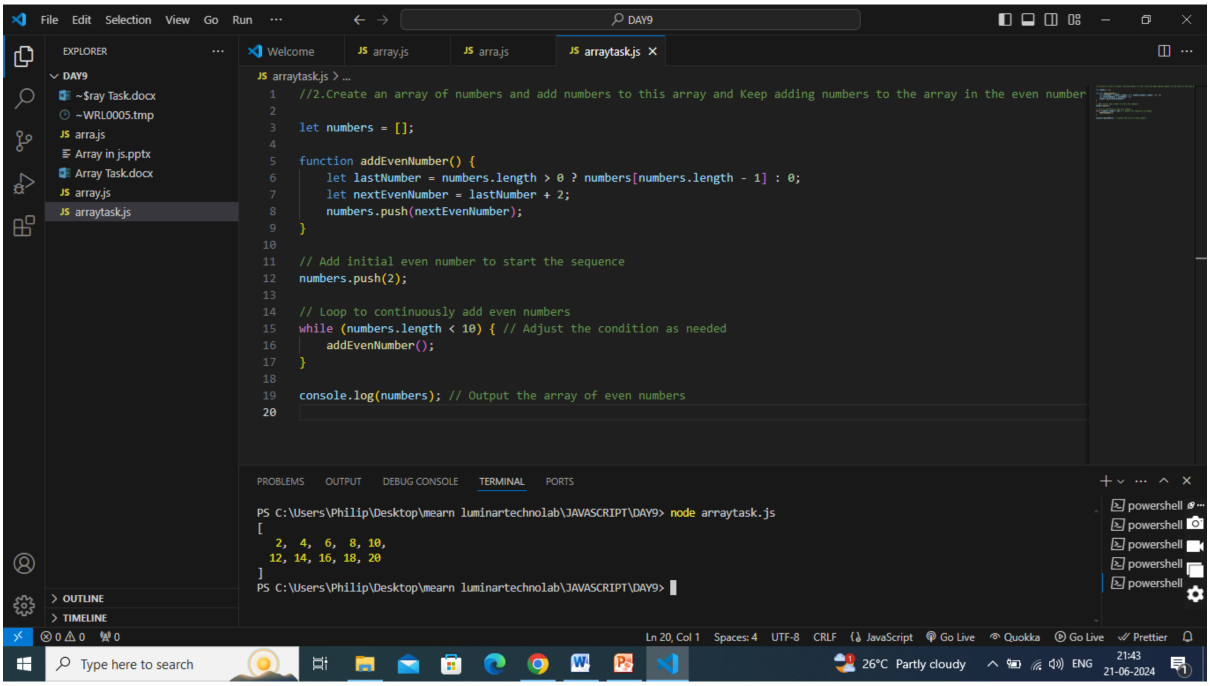
- filter()

  Creates a new array with all elements that pass the test implemented by the provided function.

  Example:

  ```
  let numbers = [1, 2, 3, 4, 5];
  let evens = numbers.filter(function(num) {
     return num % 2 === 0;
  });
  // evens is [2, 4], numbers remains unchanged
  ```

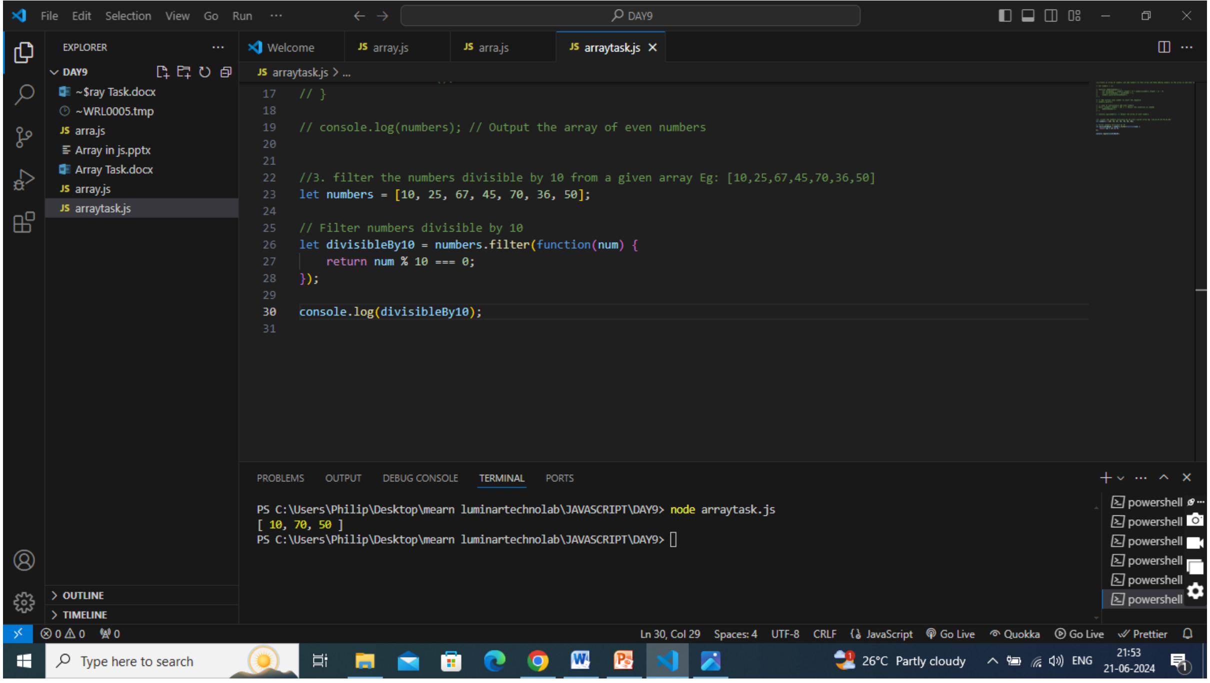
2. Create an array of numbers and add numbers to this array and Keep adding numbers to the array in the even number format.



```javascript
//2.Create an array of numbers and add numbers to this array and Keep adding numbers to the array in the even number

let numbers = [];

function addEvenNumber() {
    let lastNumber = numbers.length > 0 ? numbers[numbers.length - 1] : 0;
    let nextEvenNumber = lastNumber + 2;
    numbers.push(nextEvenNumber);
}

// Add initial even number to start the sequence
numbers.push(2);

// Loop to continuously add even numbers
while (numbers.length < 10) { // Adjust the condition as needed
    addEvenNumber();
}

console.log(numbers); // Output the array of even numbers
```

```
PS C:\Users\Philip\Desktop\mearn luminartechnolab\JAVASCRIPT\DAY9> node arraytask.js
[
   2,  4,  6,  8, 10,
  12, 14, 16, 18, 20
]
PS C:\Users\Philip\Desktop\mearn luminartechnolab\JAVASCRIPT\DAY9>
```

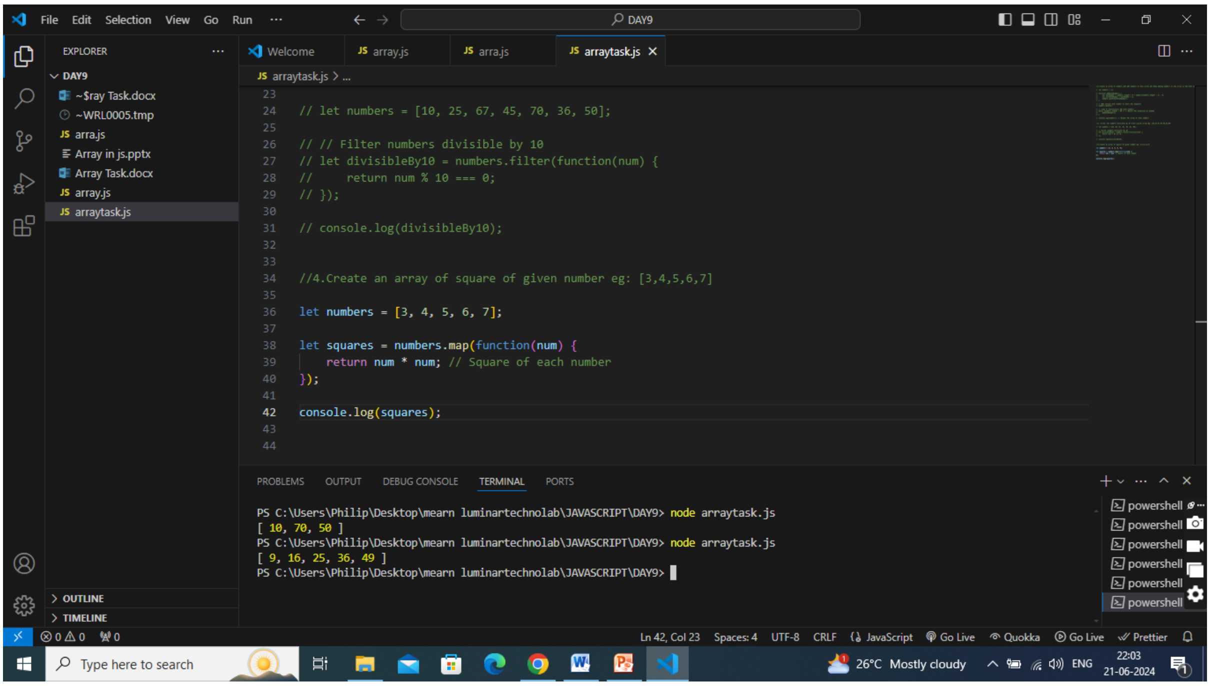
3. filter the numbers divisible by 10 from a given array Eg: [10,25,67,45,70,36,50]



```javascript
// }

// console.log(numbers); // Output the array of even numbers


//3. filter the numbers divisible by 10 from a given array Eg: [10,25,67,45,70,36,50]
let numbers = [10, 25, 67, 45, 70, 36, 50];

// Filter numbers divisible by 10
let divisibleBy10 = numbers.filter(function(num) {
    return num % 10 === 0;
});

console.log(divisibleBy10);
```

```
PS C:\Users\Philip\Desktop\mearn luminartechnolab\JAVASCRIPT\DAY9> node arraytask.js
[ 10, 70, 50 ]
PS C:\Users\Philip\Desktop\mearn luminartechnolab\JAVASCRIPT\DAY9>
```

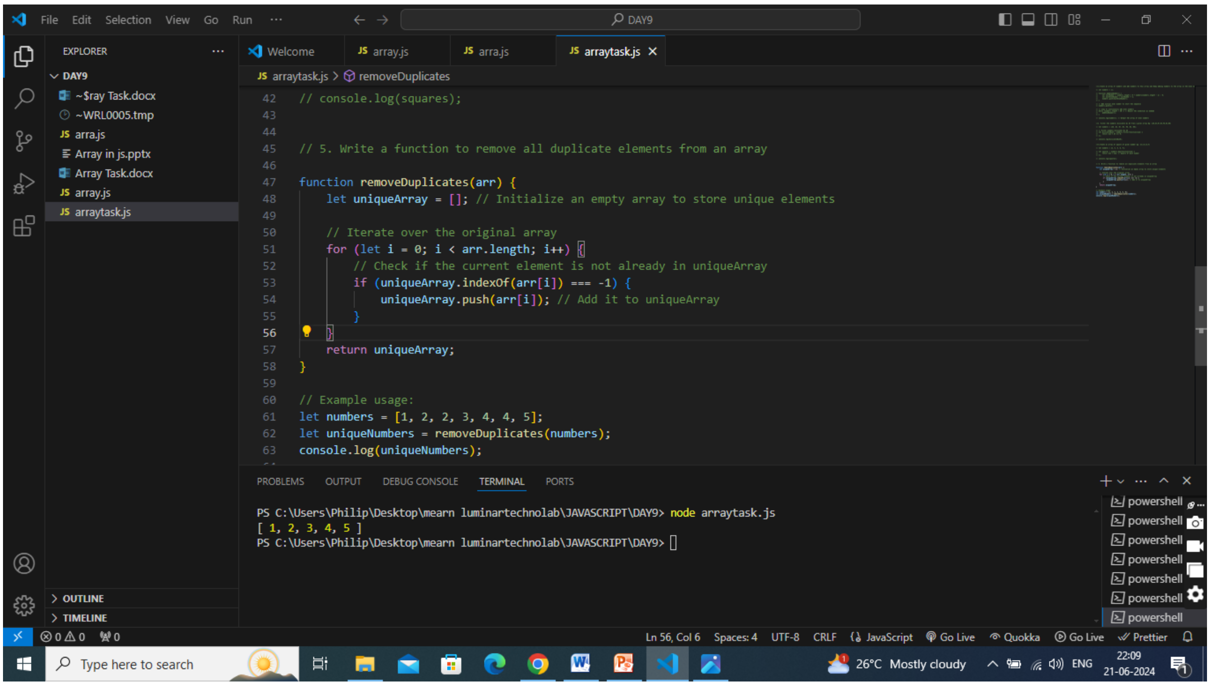
4. Create an array of square of given number eg: [3,4,5,6,7]



```
// let numbers = [10, 25, 67, 45, 70, 36, 50];

// // Filter numbers divisible by 10
// let divisibleBy10 = numbers.filter(function(num) {
//     return num % 10 === 0;
// });

// console.log(divisibleBy10);


//4.Create an array of square of given number eg: [3,4,5,6,7]

let numbers = [3, 4, 5, 6, 7];

let squares = numbers.map(function(num) {
    return num * num; // Square of each number
});

console.log(squares);
```

```
PS C:\Users\Philip\Desktop\mearn luminartechnolab\JAVASCRIPT\DAY9> node arraytask.js
[ 10, 70, 50 ]
PS C:\Users\Philip\Desktop\mearn luminartechnolab\JAVASCRIPT\DAY9> node arraytask.js
[ 9, 16, 25, 36, 49 ]
PS C:\Users\Philip\Desktop\mearn luminartechnolab\JAVASCRIPT\DAY9>
```

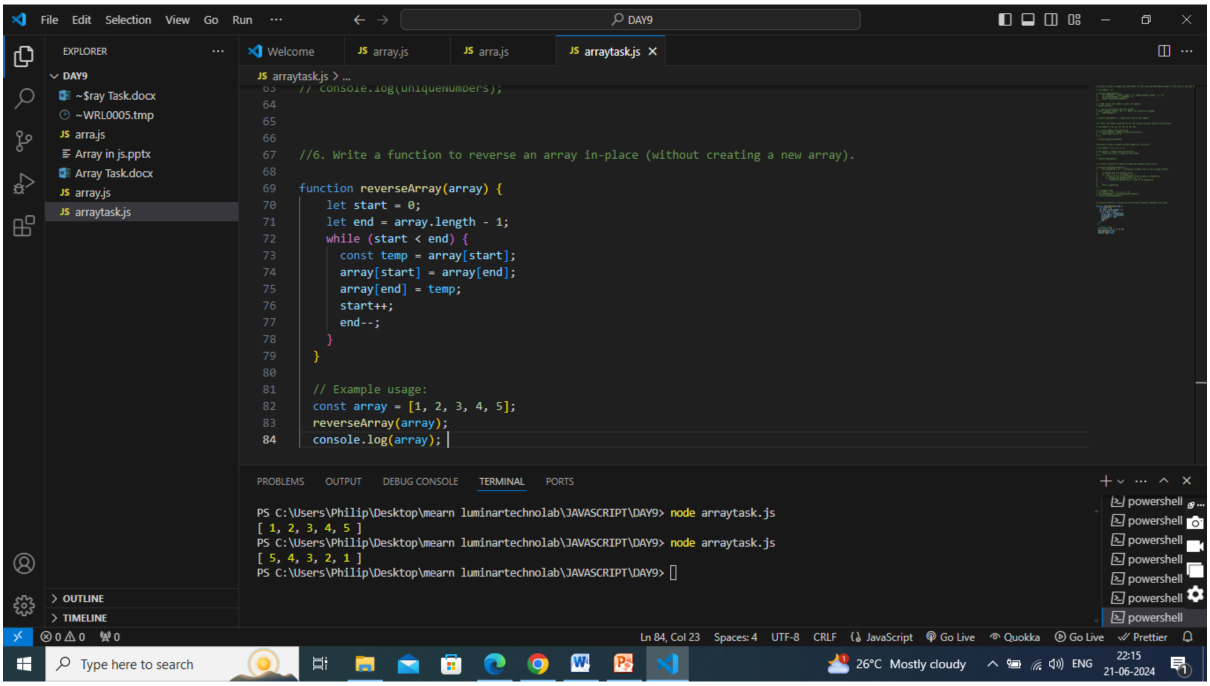
5. Write a function to remove all duplicate elements from an array



```
// console.log(squares);


// 5. Write a function to remove all duplicate elements from an array

function removeDuplicates(arr) {
    let uniqueArray = []; // Initialize an empty array to store unique elements

    // Iterate over the original array
    for (let i = 0; i < arr.length; i++) {
        // Check if the current element is not already in uniqueArray
        if (uniqueArray.indexOf(arr[i]) === -1) {
            uniqueArray.push(arr[i]); // Add it to uniqueArray
        }
    }
    return uniqueArray;
}

// Example usage:
let numbers = [1, 2, 2, 3, 4, 4, 5];
let uniqueNumbers = removeDuplicates(numbers);
console.log(uniqueNumbers);
```

```
PS C:\Users\Philip\Desktop\mearn luminartechnolab\JAVASCRIPT\DAY9> node arraytask.js
[ 1, 2, 3, 4, 5 ]
PS C:\Users\Philip\Desktop\mearn luminartechnolab\JAVASCRIPT\DAY9>
```

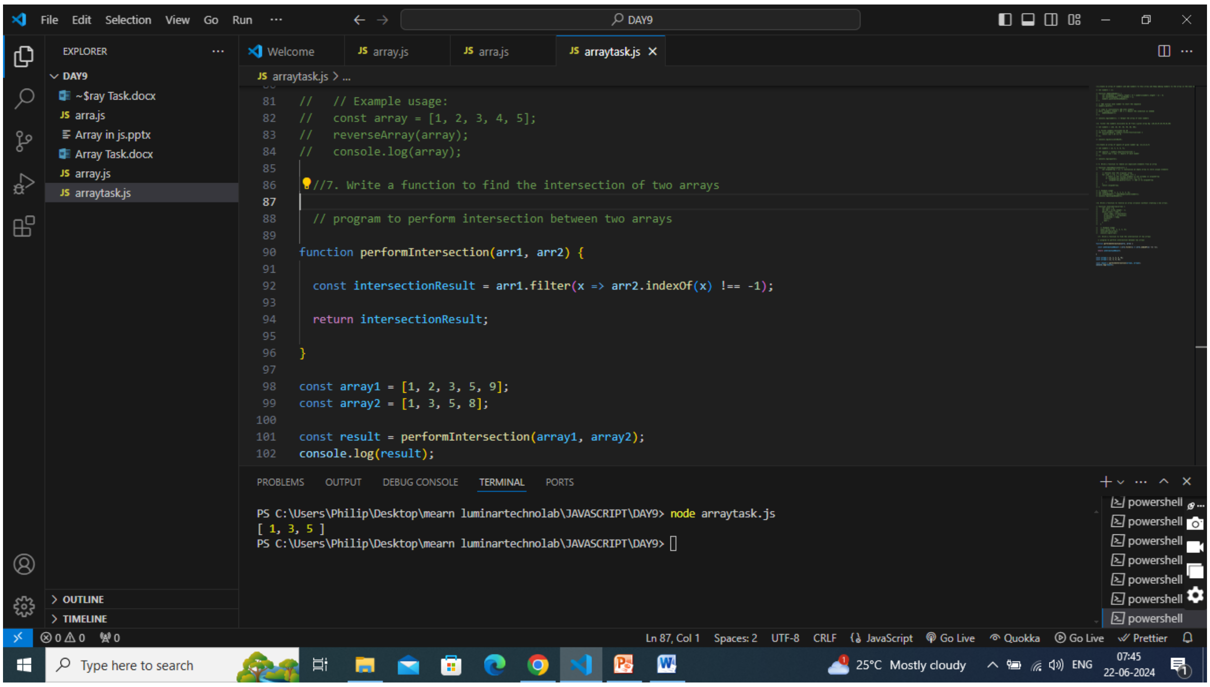
6. Write a function to reverse an array in-place (without creating a new array).



```
  // console.log(uniqueNumbers);



//6. Write a function to reverse an array in-place (without creating a new array).

function reverseArray(array) {
    let start = 0;
    let end = array.length - 1;
    while (start < end) {
      const temp = array[start];
      array[start] = array[end];
      array[end] = temp;
      start++;
      end--;
    }
  }

  // Example usage:
  const array = [1, 2, 3, 4, 5];
  reverseArray(array);
  console.log(array);
```

```
PS C:\Users\Philip\Desktop\mearn luminartechnolab\JAVASCRIPT\DAY9> node arraytask.js
[ 1, 2, 3, 4, 5 ]
PS C:\Users\Philip\Desktop\mearn luminartechnolab\JAVASCRIPT\DAY9> node arraytask.js
[ 5, 4, 3, 2, 1 ]
PS C:\Users\Philip\Desktop\mearn luminartechnolab\JAVASCRIPT\DAY9> 
```

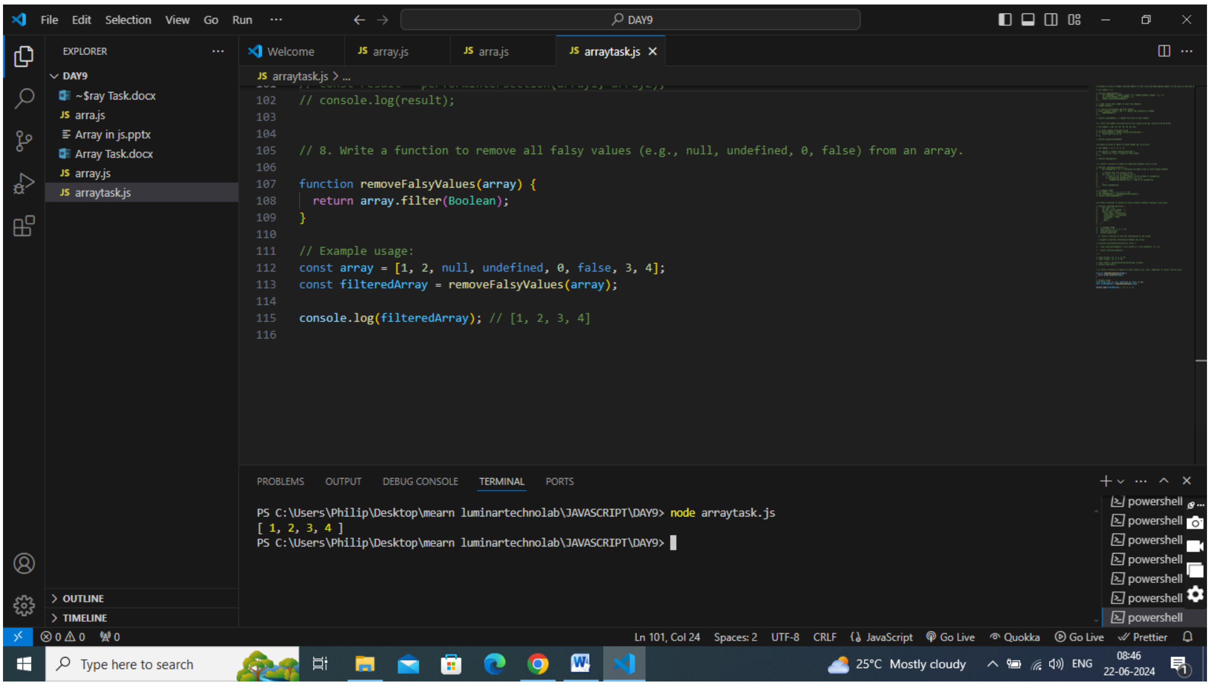
7. Write a function to find the intersection of two arrays



```
//   // Example usage:
//   const array = [1, 2, 3, 4, 5];
//   reverseArray(array);
//   console.log(array);

  //7. Write a function to find the intersection of two arrays

  // program to perform intersection between two arrays

function performIntersection(arr1, arr2) {

  const intersectionResult = arr1.filter(x => arr2.indexOf(x) !== -1);

  return intersectionResult;

}

const array1 = [1, 2, 3, 5, 9];
const array2 = [1, 3, 5, 8];

const result = performIntersection(array1, array2);
console.log(result);
```

```
PS C:\Users\Philip\Desktop\mearn luminartechnolab\JAVASCRIPT\DAY9> node arraytask.js
[ 1, 3, 5 ]
PS C:\Users\Philip\Desktop\mearn luminartechnolab\JAVASCRIPT\DAY9> 
```

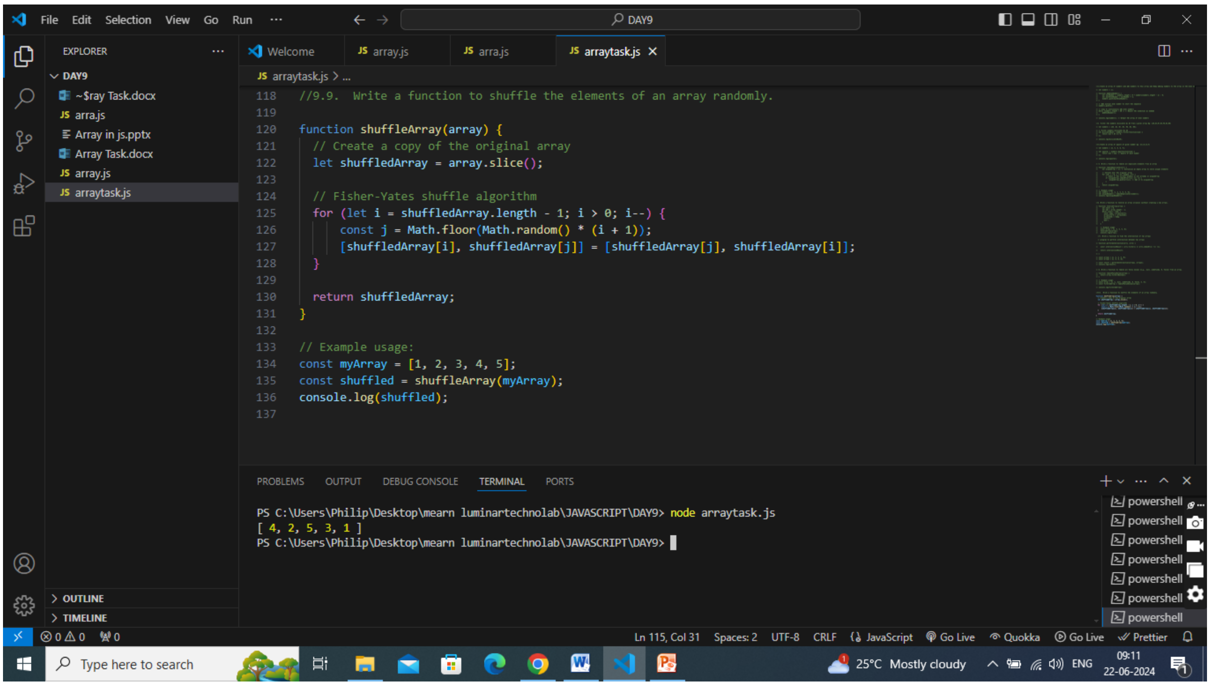
8. Write a function to remove all falsy values (e.g., null, undefined, 0, false) from an array.



```
// console.log(result);


// 8. Write a function to remove all falsy values (e.g., null, undefined, 0, false) from an array.

function removeFalsyValues(array) {
  return array.filter(Boolean);
}

// Example usage:
const array = [1, 2, null, undefined, 0, false, 3, 4];
const filteredArray = removeFalsyValues(array);

console.log(filteredArray); // [1, 2, 3, 4]
```

```
PS C:\Users\Philip\Desktop\mearn luminartechnolab\JAVASCRIPT\DAY9> node arraytask.js
[ 1, 2, 3, 4 ]
PS C:\Users\Philip\Desktop\mearn luminartechnolab\JAVASCRIPT\DAY9> 
```

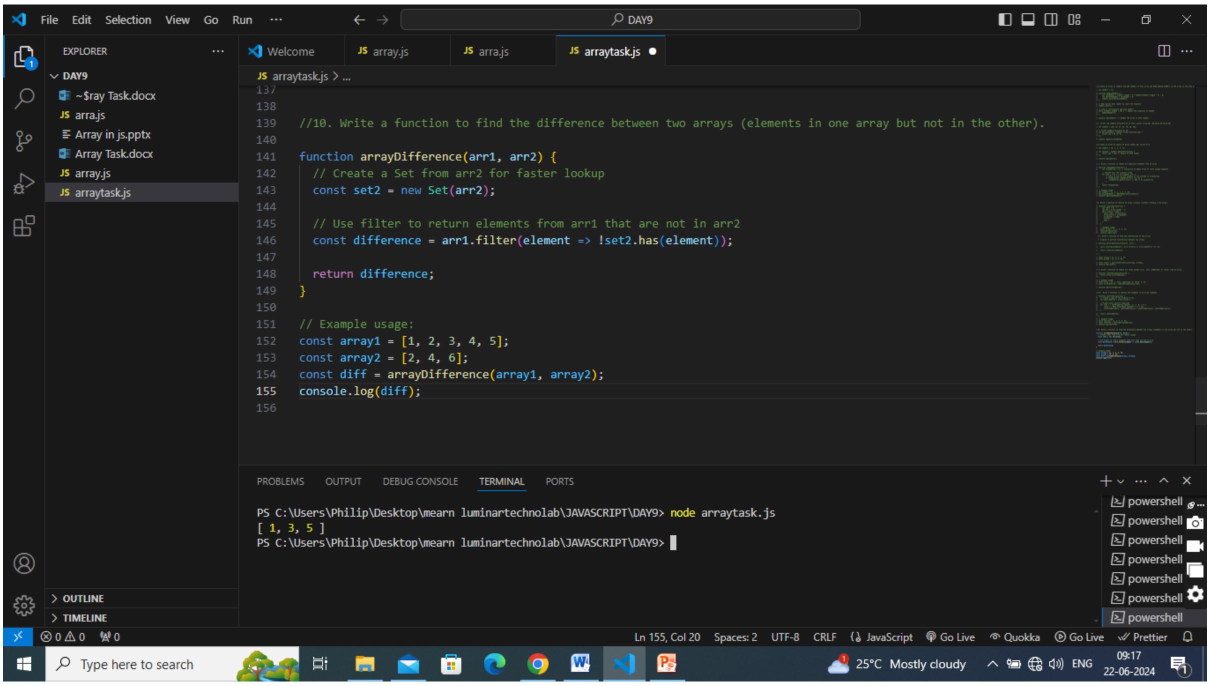
9. Write a function to shuffle the elements of an array randomly.



```
//9.9.  Write a function to shuffle the elements of an array randomly.

function shuffleArray(array) {
  // Create a copy of the original array
  let shuffledArray = array.slice();

  // Fisher-Yates shuffle algorithm
  for (let i = shuffledArray.length - 1; i > 0; i--) {
      const j = Math.floor(Math.random() * (i + 1));
      [shuffledArray[i], shuffledArray[j]] = [shuffledArray[j], shuffledArray[i]];
  }

  return shuffledArray;
}

// Example usage:
const myArray = [1, 2, 3, 4, 5];
const shuffled = shuffleArray(myArray);
console.log(shuffled);
```

```
PS C:\Users\Philip\Desktop\mearn luminartechnolab\JAVASCRIPT\DAY9> node arraytask.js
[ 4, 2, 5, 3, 1 ]
PS C:\Users\Philip\Desktop\mearn luminartechnolab\JAVASCRIPT\DAY9> 
```

10. Write a function to find the difference between two arrays (elements in one array but not in the other).



```javascript

//10. Write a function to find the difference between two arrays (elements in one array but not in the other).

function arrayDifference(arr1, arr2) {
  // Create a Set from arr2 for faster lookup
  const set2 = new Set(arr2);

  // Use filter to return elements from arr1 that are not in arr2
  const difference = arr1.filter(element => !set2.has(element));

  return difference;
}

// Example usage:
const array1 = [1, 2, 3, 4, 5];
const array2 = [2, 4, 6];
const diff = arrayDifference(array1, array2);
console.log(diff);

```

```
PS C:\Users\Philip\Desktop\mearn luminartechnolab\JAVASCRIPT\DAY9> node arraytask.js
[ 1, 3, 5 ]
PS C:\Users\Philip\Desktop\mearn luminartechnolab\JAVASCRIPT\DAY9> 
```