JAVASCRIPT

JavaScript is the world most popular programming language.

Java script is used for

* Front end web development
* Mobile app
* Desktop app development
* Server-side programming etc.

It is no 1 programming language.

It is case sensitive.

-Installation node js first

First javascript is run only on browser .nodejs create runtime environment

or javascript run on compiler .

* output

In js to print any thing ----console.log is use

Console.log(“ “);

Run in terminal node file name-----eg node demo.js

* variable

is use to store data in box .box is name as variable.

String means word or sentence written in double quotes “ “

Var indentifier

Varables naming rules

No number in starting ,but it is allowed

Underscore is use

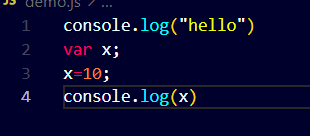
$ sign can use

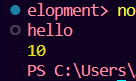
Keyword cannot be use

Var demo;----declared

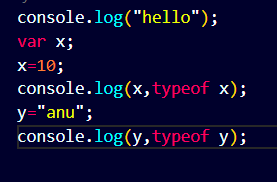
Var demo=1;----assign

We can declared variable of same name more times.





To check datatype---typeof----is used





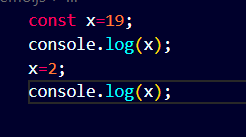
Let—is same as var

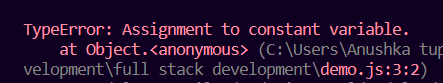
Difference between var and let is

In var we can redeclared variable but in let we cannot redeclared variable

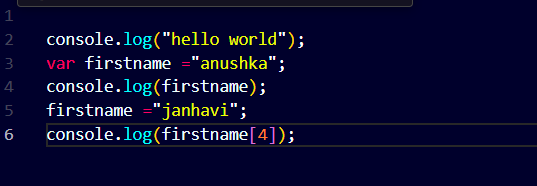
It show error

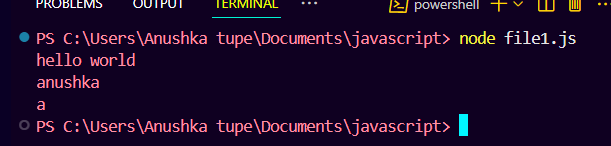
Constant-----we canot change value



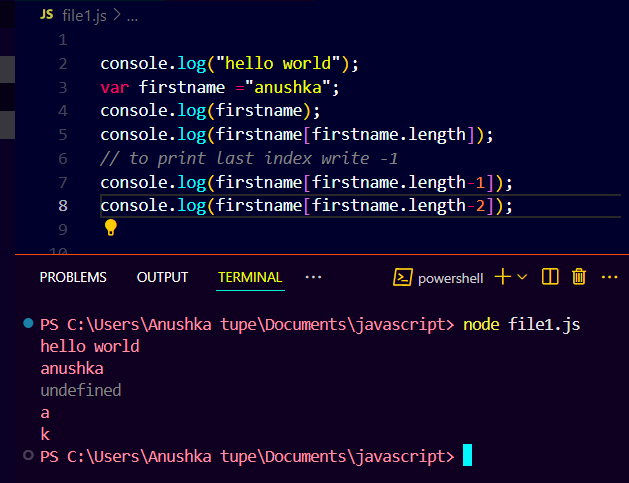


Indexing



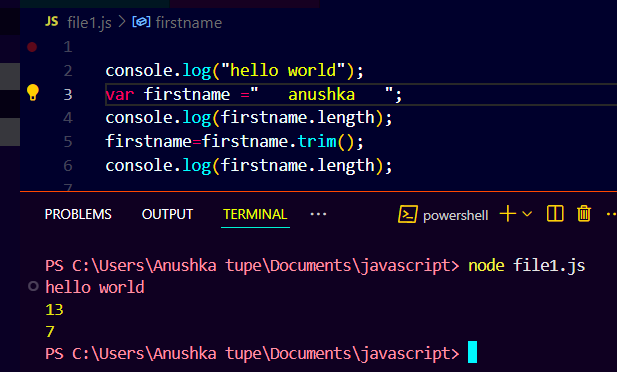


To print the length of string use .length



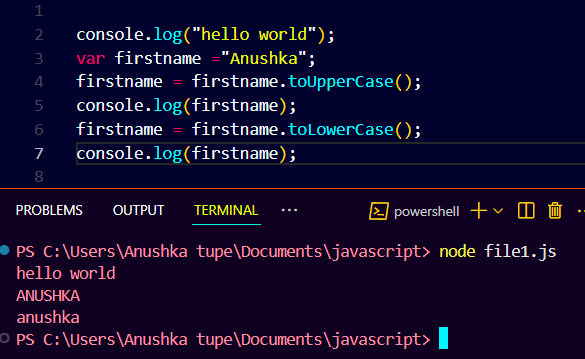
Trim

Trim is use to remove spaces from string



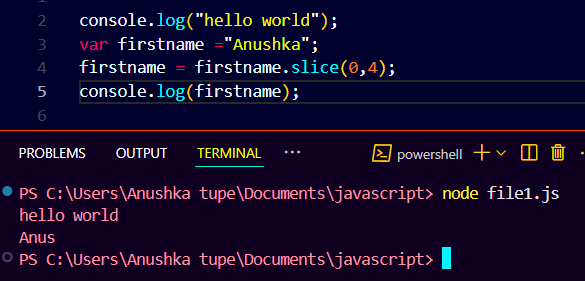
toUpperCase() toLowerCase()

convert into uppercase and lowercase



Slice()

The slice() method in JavaScript is used to extract a portion of an array or a string without modifying the original.

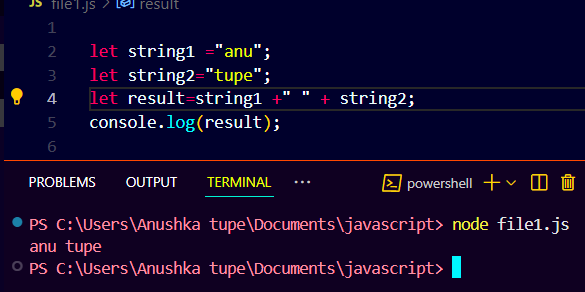


Convert number into string and string into number



String concatenation

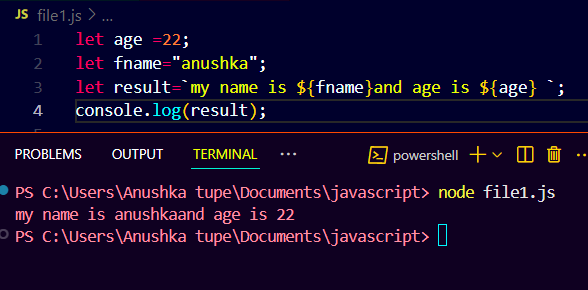
String concatenation means joining two or more strings together to form a single string



Template string

A template string (also called a template literal) is a way to create strings that allows embedding variables and expressions directly within the string — usually using special syntax.

Template literals are enclosed by backticks ( ` ` ) and use ${} to embed expressions.

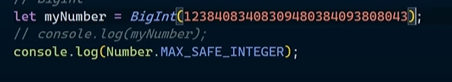


**Why type of null is an object ?**

This is a well-known quirk (bug) in JavaScript. It dates back to the first version of the language. null was meant to represent the absence of any object value, but due to how types were stored in the early implementation, typeof null ended up being "object" — and it has stayed that way for backward compatibility.

Big int

BigInt is a special numeric type in JavaScript that allows you to **represent integers larger**



* operator

|

arithmetic operator

- sub two operands

\* Multiple



\*\* power star--------eg-5\*\*2---it will give u--25

/ divide

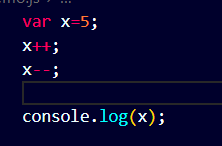
% modulo---remainder

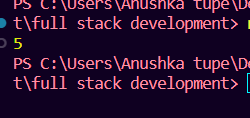
++ increment-------+1

- - decrement-------- -1

One operand







Assignment operator

=

+=

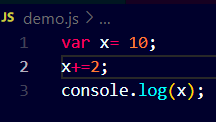
-=

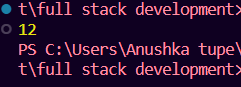
\*=

/=

%=

\*\*=





Comparison operator

>

<

>=

<=

==

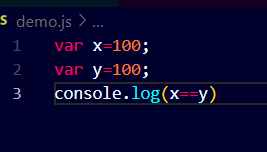
!=

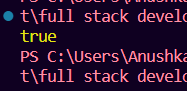
In this output is always true or false

= assign

== equal too

=== equal and typeoff



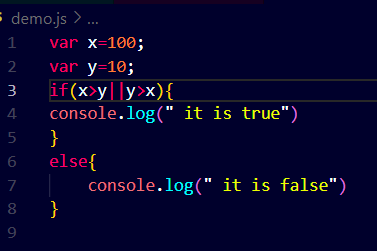


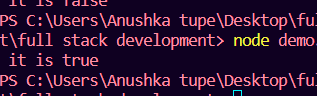
Logical operator

&& AND------BOTH TRUE ------ANS----TRUE

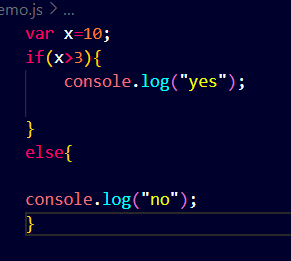
|| OR-------ONE TRUE-------ANS-----TRUE

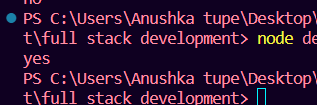
! NOT-----TRUE-FALSE-----FALSE-TRUE



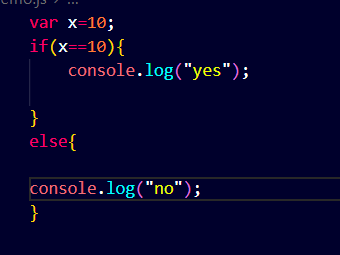


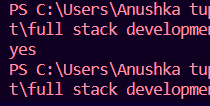
If,else if and else

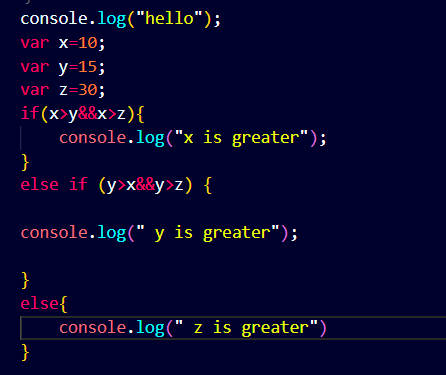


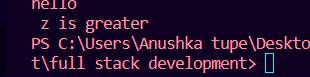


------------------------------------------------------------------------------------------------------









Loops

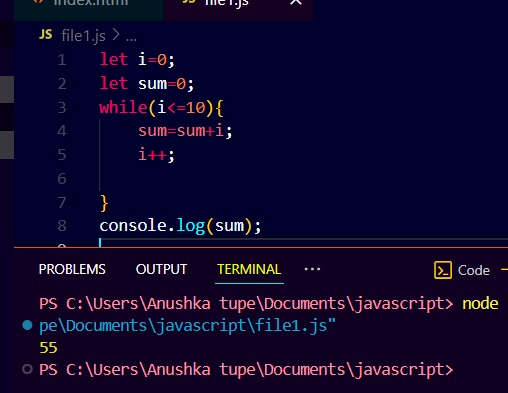
Loops are used to repeat a block of code multiple times.

While loop

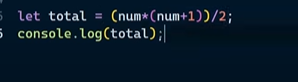
it is use to repeat a set of code number of times



Sum of 1 to 10 number

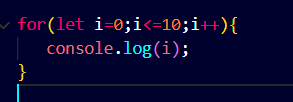


Without loop



For loop

In JavaScript, a for loop is used to run a block of code a specific number of times.



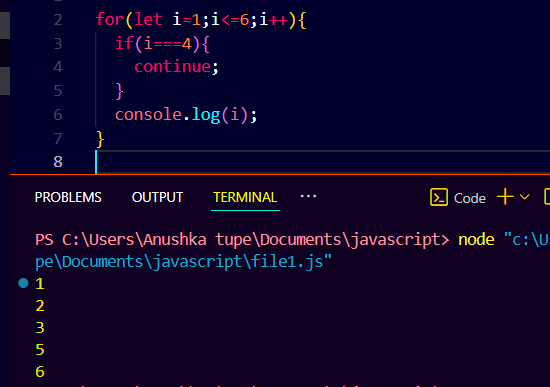
Break

In JavaScript, the break statement is used to exit a loop or a switch statement prematurely—that is, before it would normally finish on its own.



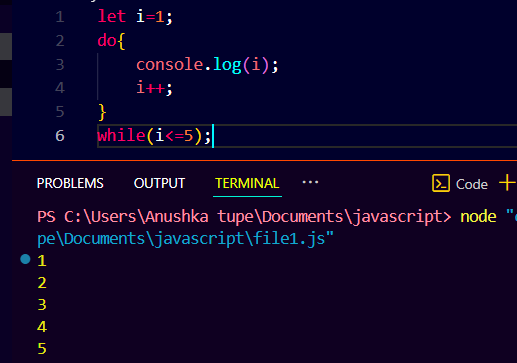
Continue

In JavaScript, the continue statement is used to skip the current iteration of a loop and jump to the next one.



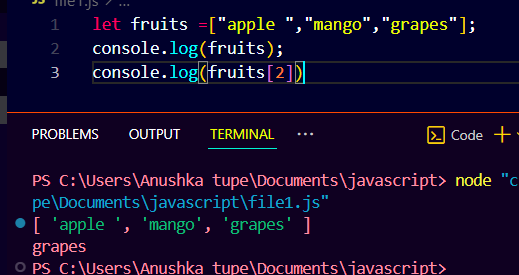
Do while

In JavaScript, the do...while loop is a control flow statement that executes a block of code at least once, and then continues executing it as long as a specified condition is true.



Array

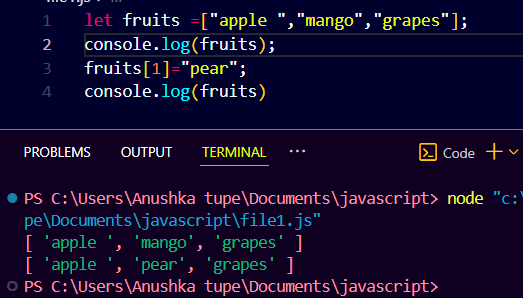
array is a data structure used in JavaScript to store multiple values in a single variable. Each value in an array has a numbered position, called an index, which starts from 0.



Arrays are mutable

 can change their contents after creation.

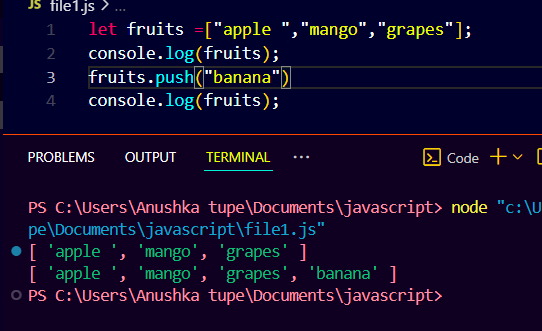
 You can add, remove, or modify elements without changing the original reference.



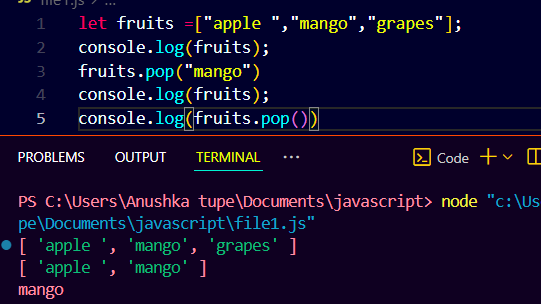
Array push pop

push(): Adds one or more elements to the end of an array.

pop(): Removes the last element from an array and returns it.

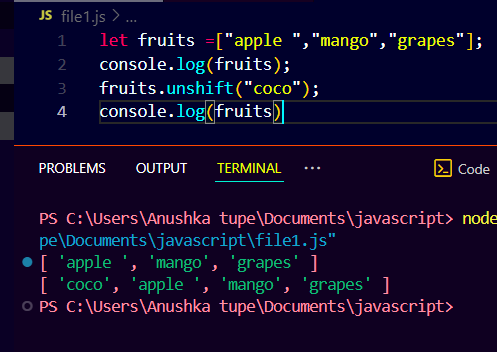


Also give the pop element



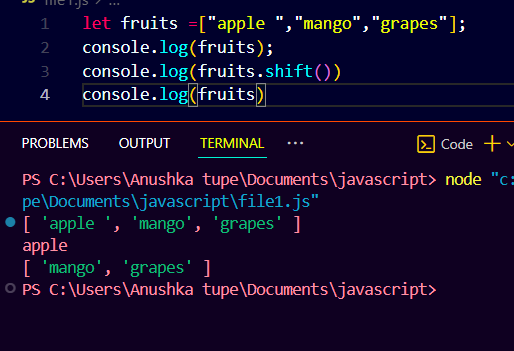
Unshift

unshift() adds one or more elements to the beginning of an array.



Shift

shift() removes the first element from an array.



Primitive vs references data types

Primitive data types are the most basic types of data in JavaScript. These are immutable values, meaning they cannot be altered after creation.

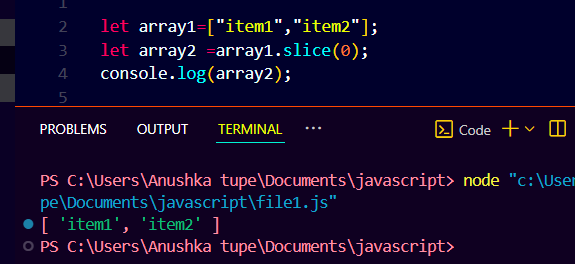
Reference data types, on the other hand, are more complex and include objects, arrays, and functions. They are mutable, meaning their properties or elements can be changed.



How to clone array

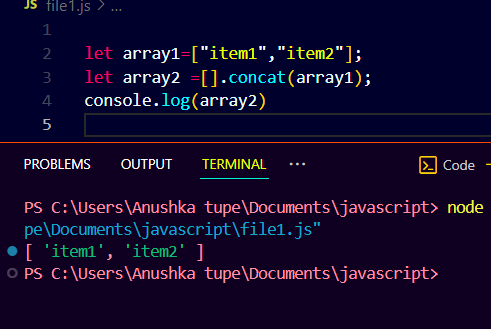
Slice method is use

The slice() method is used to extract a portion of an array or string without changing the original data.

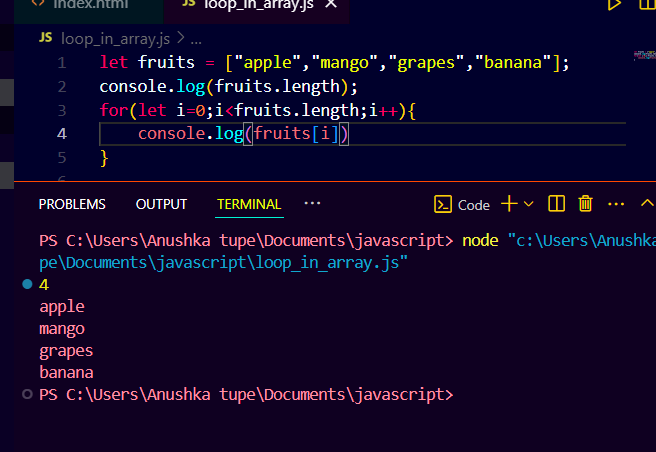


By concat ()

The concat() method is used to combine two or more arrays or strings into a new one. It does not change the original values.



For loop in array



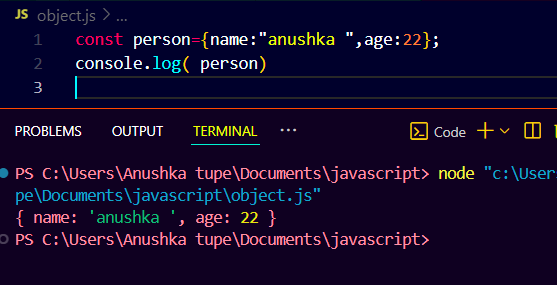
While loop in array

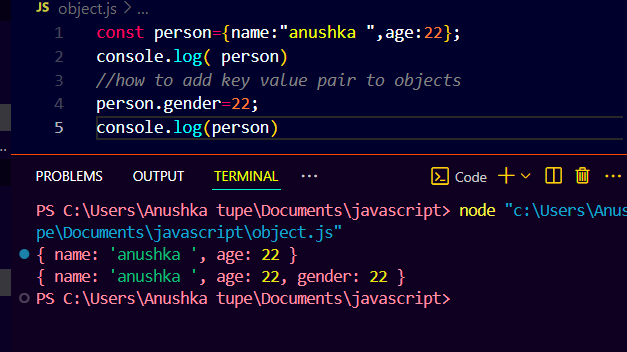


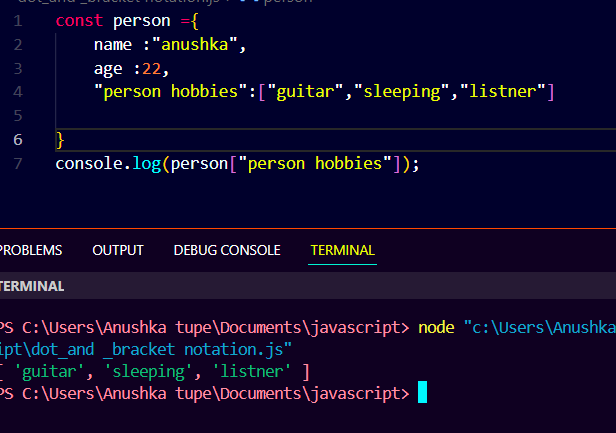


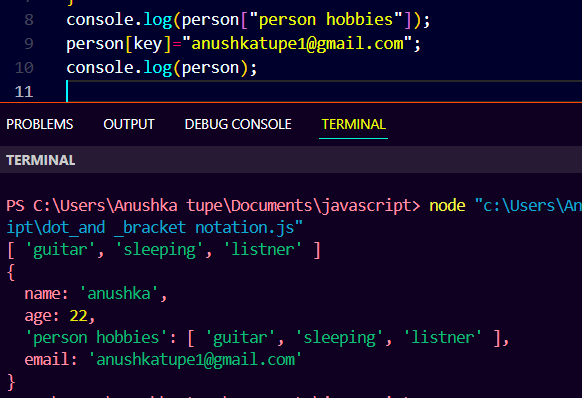
Object

An object in JavaScript is a collection of key-value pairs used to store and manage data.

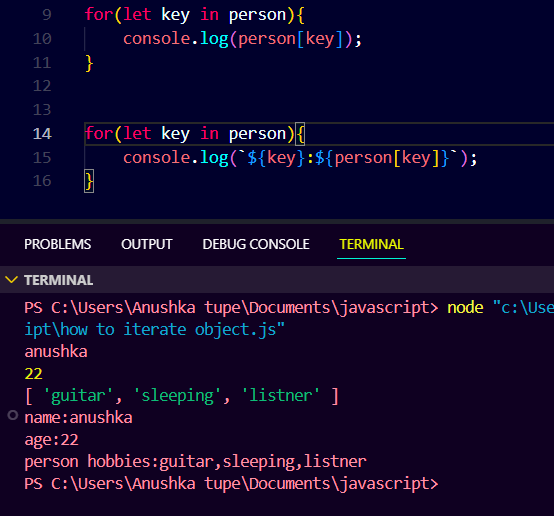






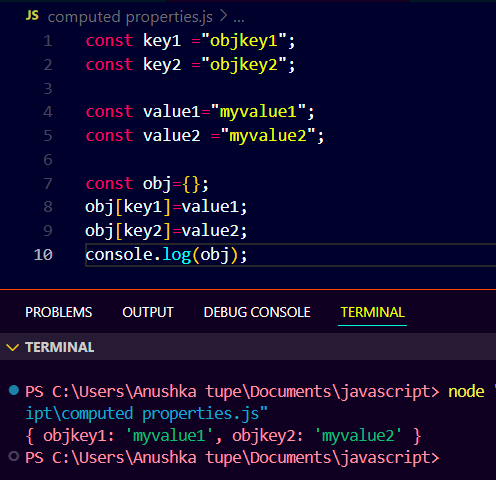


How to iterate object



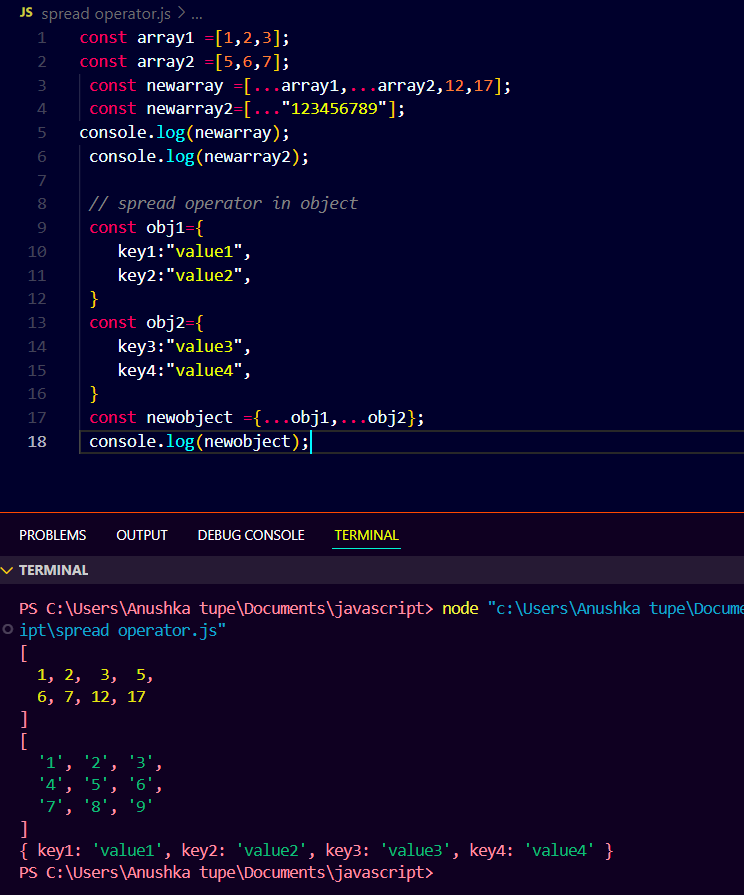
Computed value

means a value that is calculated based on an expression or dynamically gnerated at runtime.



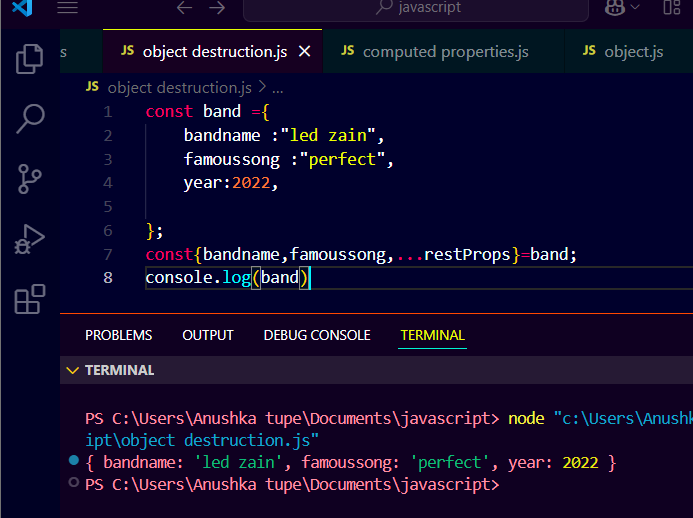
Spread operator

The **spread operator** (...) is used to **expand** elements of an array or object.



Object destructuring

**Object destructuring** means **pulling out values from an object** and storing them in variables easily.

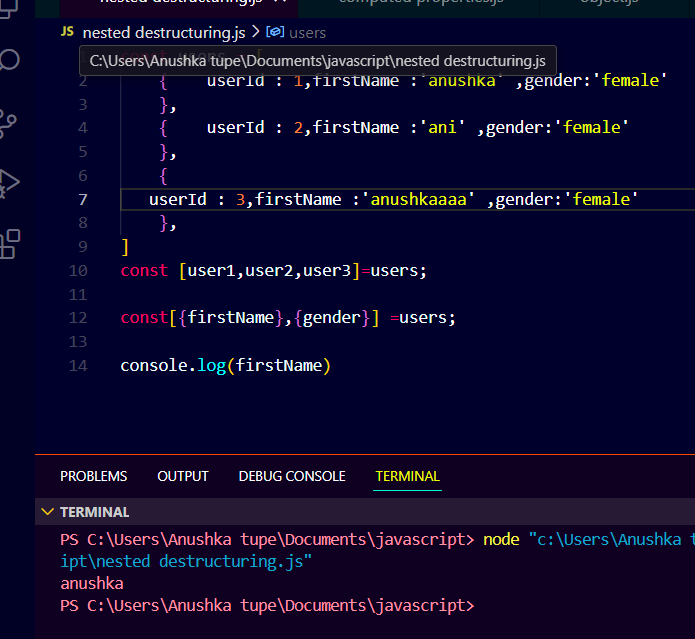


Object inside array

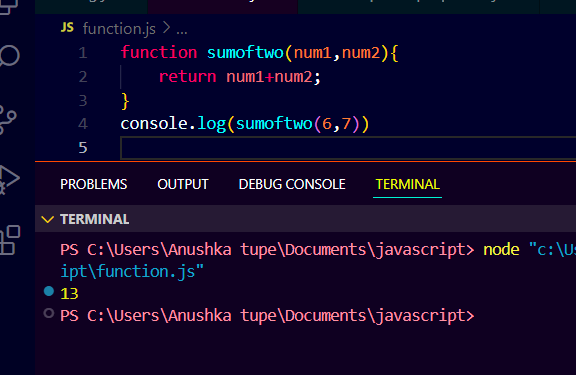


Nested destructer

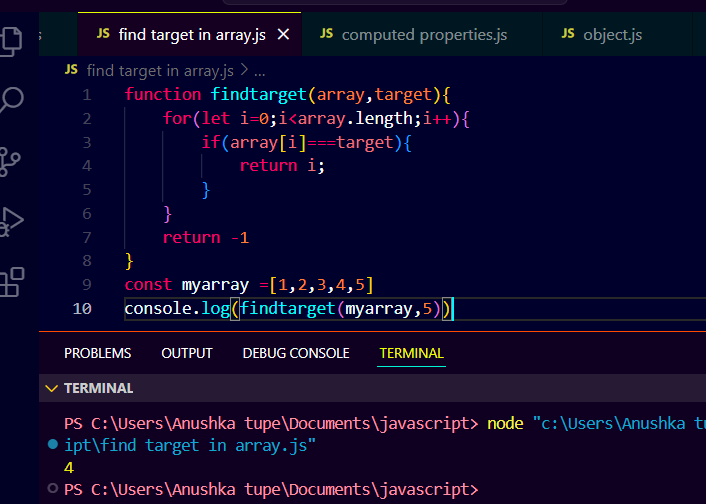
**Nested destructuring** means extracting values **from objects inside other objects** (or arrays inside objects), directly.



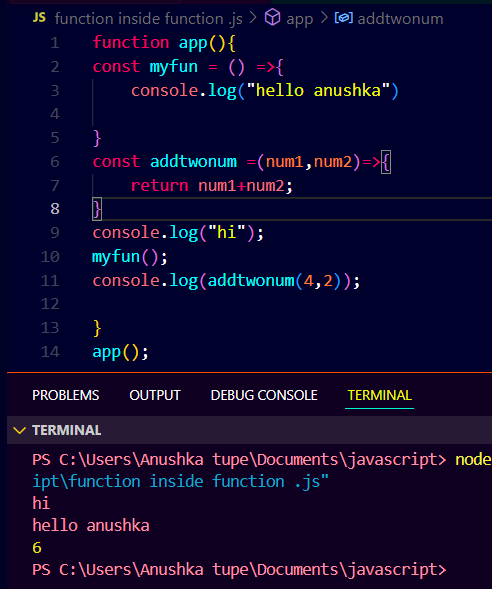
Function



Find the target in array

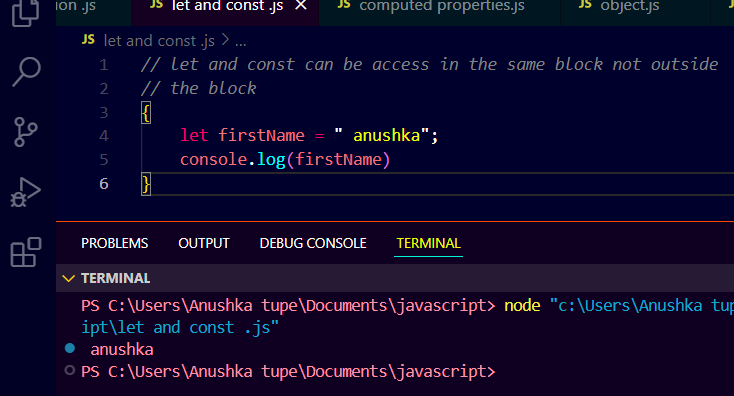


Function inside function



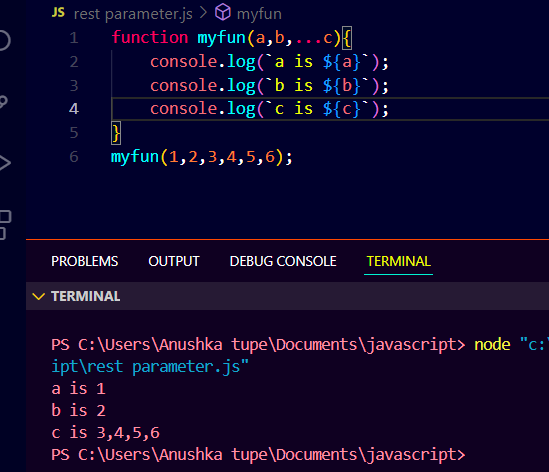
Let and const block

In JavaScript, both let and const are **block-scoped** — they only work **inside the block {} where they are declared**.



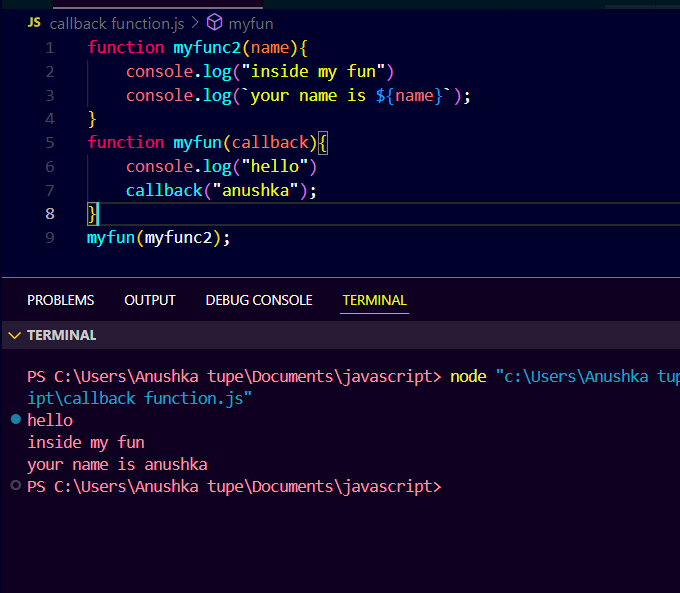
Rest parameter

The **rest parameter** (...) allows a function to accept **any number of arguments** as an **array**.



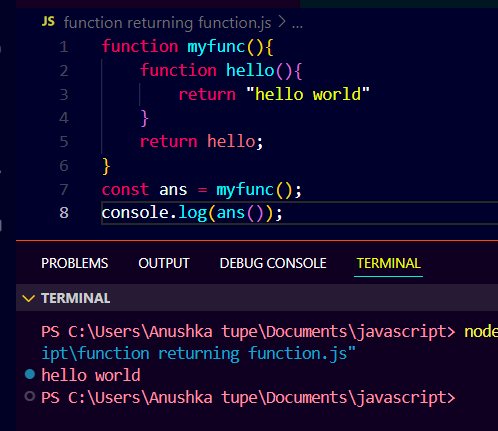
Call function

The **call()** method is used to **call a function with a specific this value** and arguments.



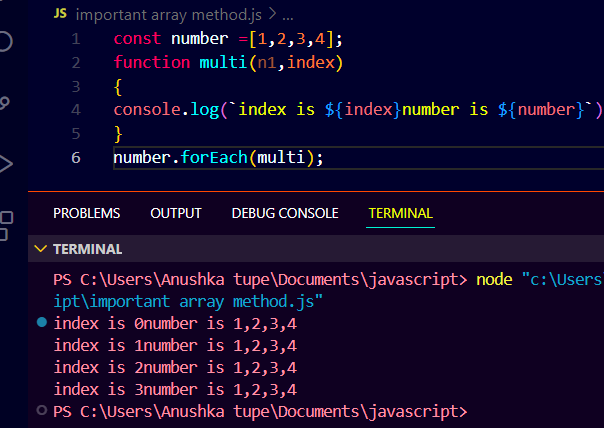
Function returning function

In JavaScript, a **function can return another function**. This is called a **higher-order function**.



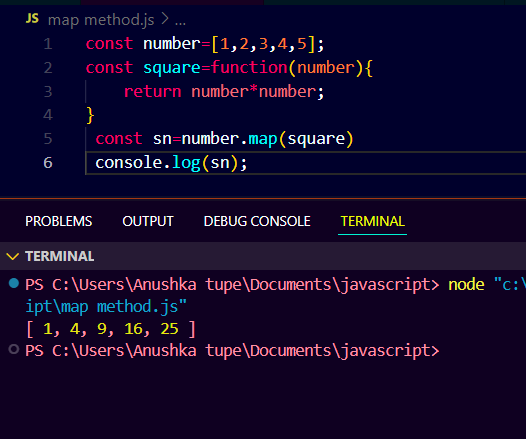
For each

The **forEach()** method is used to **loop through arrays** and run a function on **each item**.



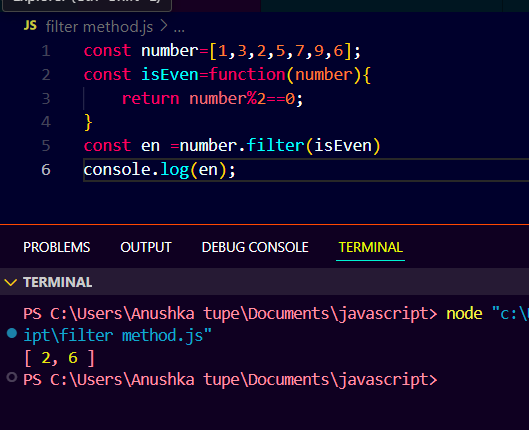
Map method

The **map()** method is used to **create a new array** by applying a function to each item of the original array



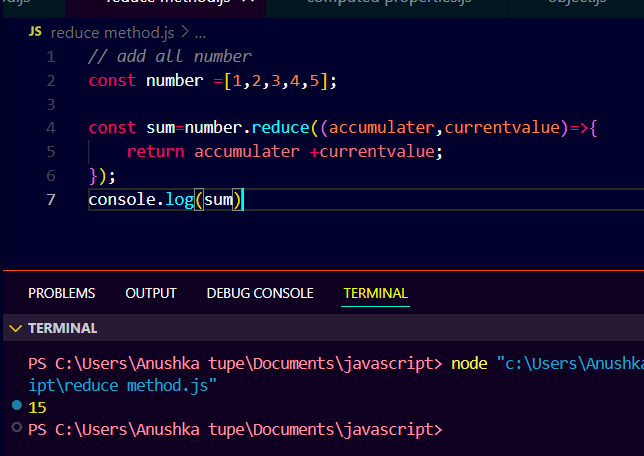
Filter method

The **filter()** method creates a **new array** containing all elements that **pass a specific test** (i.e., meet a condition).

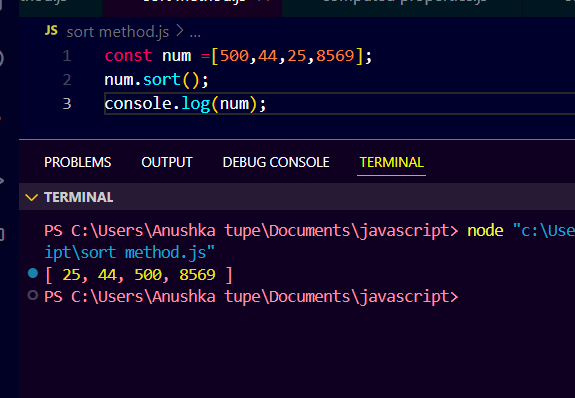


Reduce method

The **reduce()** method is used to **accumulate** or **reduce** an array into a single value, by applying a function to each element.

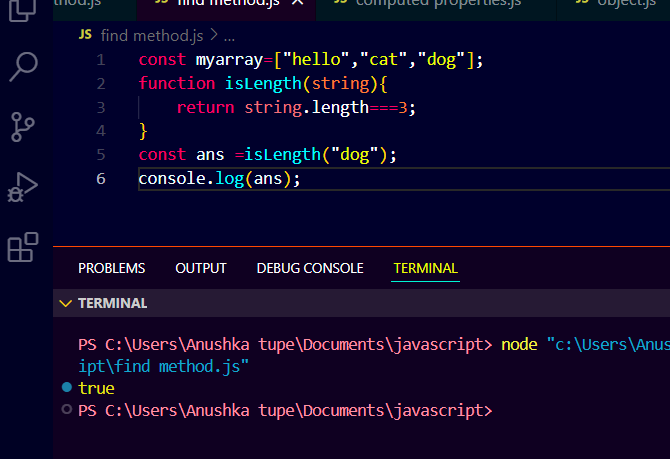


Sort method



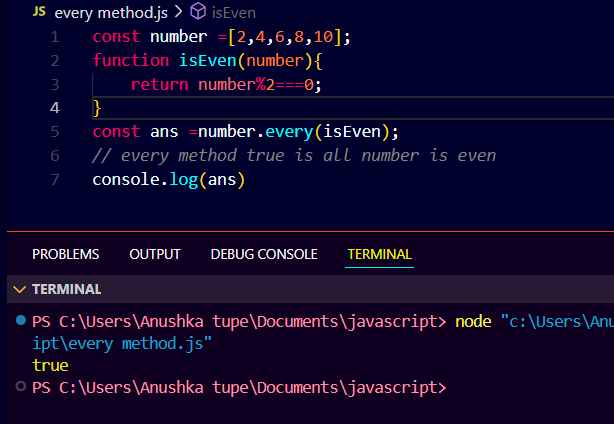
Find method

The **find()** method is used to **find the first element** in an array that satisfies a **given condition**. It returns the **value** of the first matching element or undefined if no element matches.



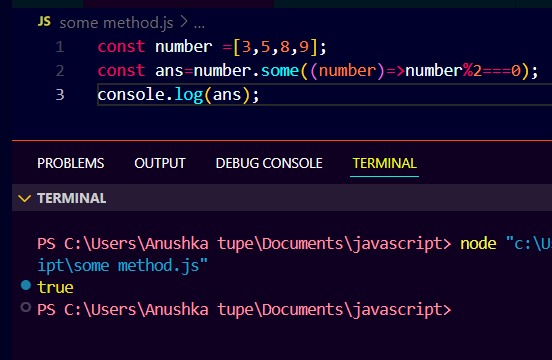
Every method

The **every()** method checks if **every element** in an array **meets a specified condition**. It returns true if **all elements** satisfy the condition; otherwise, it returns false.



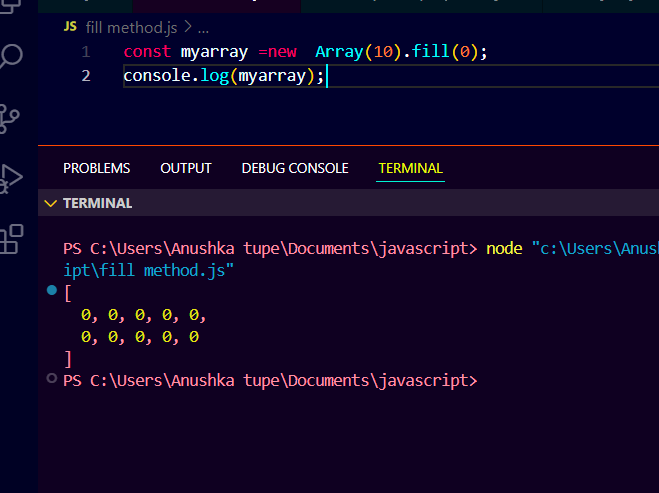
Some method

The **some()** method checks if **at least one element** in an array **meets a specified condition**. It returns true if any element satisfies the condition, and false if none of the elements satisfy it.



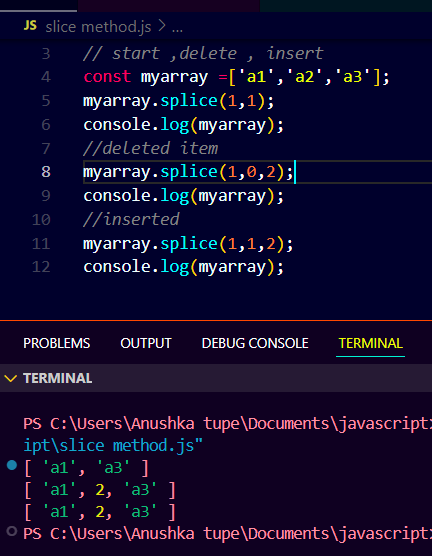
Fill method

The **fill()** method is used to **change** all elements in an array to a specific value, **from a start index to an end index**. It modifies the original array in place.



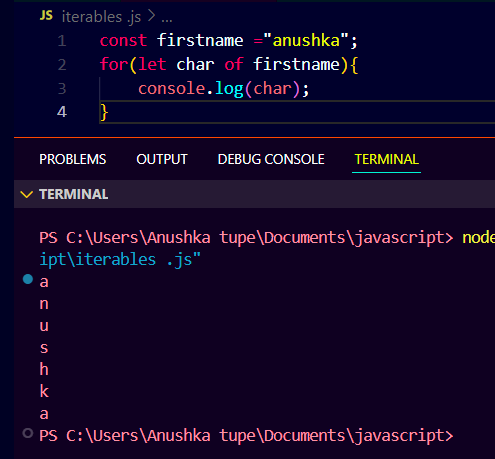
Splice method

The **splice()** method is used to **add, remove, or replace elements** in an array at a specific position. It **modifies the original array**.



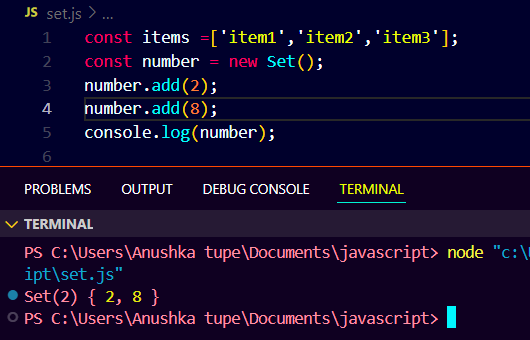
Iterables

In JavaScript, an **iterable** is any object that can be looped over, meaning you can access each element or property one by one. Iterables allow the use of **loops** like for...of or methods like map(), filter(), and reduce() to process their elements.



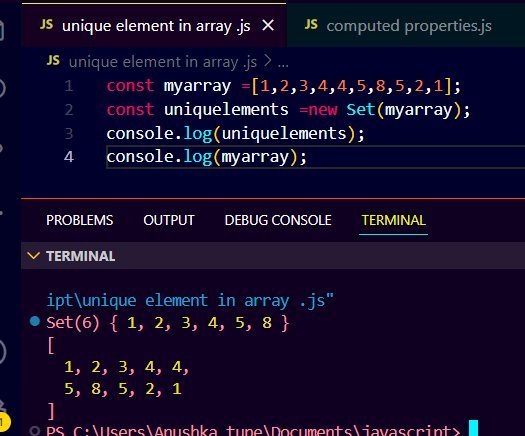
Sets

A **Set** is a built-in JavaScript object that allows you to store **unique values** of any type. Unlike arrays, a Set **does not allow duplicate values**.



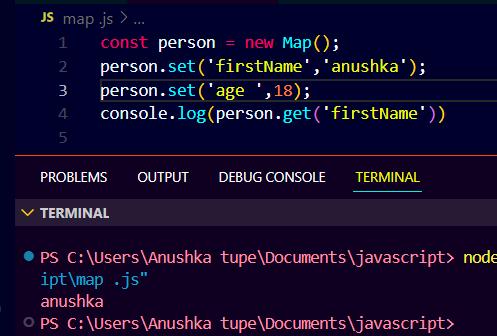
Unique element

To find **unique elements** in an array, you need to remove **duplicates**. In JavaScript, there are a few ways to accomplish this. Here are some common approaches:



Map ()

The **map()** method is a built-in JavaScript array function that **creates a new array** by applying a **function** to each element of an existing array. It **does not modify** the original array, and it **returns a new array** with the results.





Clone using object

In JavaScript, **cloning an object** means creating a **new object** that is a copy of the original object. This new object should have the same properties and values, but it should be a completely separate object (i.e., modifying the clone should not affect the original object).



Optional chaining

**Optional Chaining** is a feature in JavaScript that allows you to safely **access deeply nested properties** of an object without having to check for each level of the object. It helps avoid runtime errors like TypeError: Cannot read property 'X' of undefined when trying to access properties on null or undefined.

