**Automation Testing with playwright and JavaScript**

**\*What is software?**

**-**Software is basically a set of instructions or commands that tells a computer what to do?

\***Types of software-**

1)**System Software**-Drivers, operating systems etc.

2)**Programming Software**-Compilers, debuggers etc.

3)**Application Software**-Web applications, mobile applications etc.

**Software Testing** is a part of software development Process.

Software testing is an activity to detect and identify the defects in the software.

**Objective-**to release a quality product to the client.

**Quality** –

1)Bug free

2)Delivered on time

3)Within client budget

4)Maintainable

5)Meets requirements and expectations.

**Project vs Product-**

If software application is developed for specific customers based on requirement is called **project.**

If software application is developed for multiple customers based on market requirements, then it called **product.**

**What is Manual Testing?**

* Manual testing is conducted to discover bugs in the developed software application.
* The tester checks all the essential features of the application.
* The tester executes test cases and generates test reports without any help from the automation tools.
* There is chance of any human errors.

## What is Automation Testing?

[Automation testing](https://www.geeksforgeeks.org/automation-testing-software-testing/)is a [type of testing](https://www.geeksforgeeks.org/types-software-testing/)in which we take the help of tools (automation) to perform the testing. It is faster than manual testing because it is done with some automation tools. There is no chance of any human errors.

* It relies entirely on pre-scripted test which runs automatically to compare actual results with expected results.
* Automation testing helps the tester determine whether the application performs as expected or not.
* It allows the execution of repetitive tasks and regression tests.
* Automation requires manual effort to create initial testing scripts.

**When to Perform Automation Testing?**

* **When need to run repetitive tasks:** Automated tests are the best option in scenarios where there is a requirement to run repetitive tests. For example, in the case of regression tests must be executed periodically to make sure that the newly added code does not disrupt the existing functionality of the software.
* **When human resources are scarce:** Automated tests are viable and the best option to get tests executed within deadlines when there are only a limited number of dedicated testers.

### **Benefits of Automation Testing-**

* **Finds more bugs:**Automation testing helps to find more bugs and defects in the software.
* **Reduce time for regression tests:**Automated tests are suitable for regression tests as the tests can be executed in a repetitive manner periodically.
* **The process can be recorded:**This is one of the benefits of using automation tests as these tests can be recorded and thus allows to reuse of the tests.
* **No fatigue:**As automation, tests are executed using software tools so there is no fatigue or tiring factor as in manual testing.
* **Increased test coverage:**Automation tests help to increase the test coverage as using the tool for testing helps to make sure that not even the smallest unit is left for testing.

### **Limitations of Automation Testing**

* **High cost:** Automation tests have a high cost of implementation as tools are required for testing, thus adding the cost to the project budget.
* **Test maintenance is costly:** In automation tests, test maintenance is costly.
* **Not false proof:** Automation tests also have some limitations and mistakes in automated tests can lead to errors and omissions.
* **Trained employees required:** For conducting automated tests, trained employees with knowledge of programming languages and testing knowledge are required.

**Playwright –**

**1)Browsers support –**Chromium,WebKit(Safari) and FireFox

(Headed/Headless).

2**)Application support**- Web apps, mobile apps, API.

3)**Languages support-** JavaScript,TypeScript,Java,python and .Net(C#).

4)**OS support**-Windows,MacOs,Linux.

**Playwright Important Points to know –**

1)Free and Open source.

2)Multi -browser,Multi-language.

3)Easy setup and configuration.

4)Parallel Testing Possible.

5)Auto wait.

6)Inbuilt code generator.

7)Built-in Assertions.

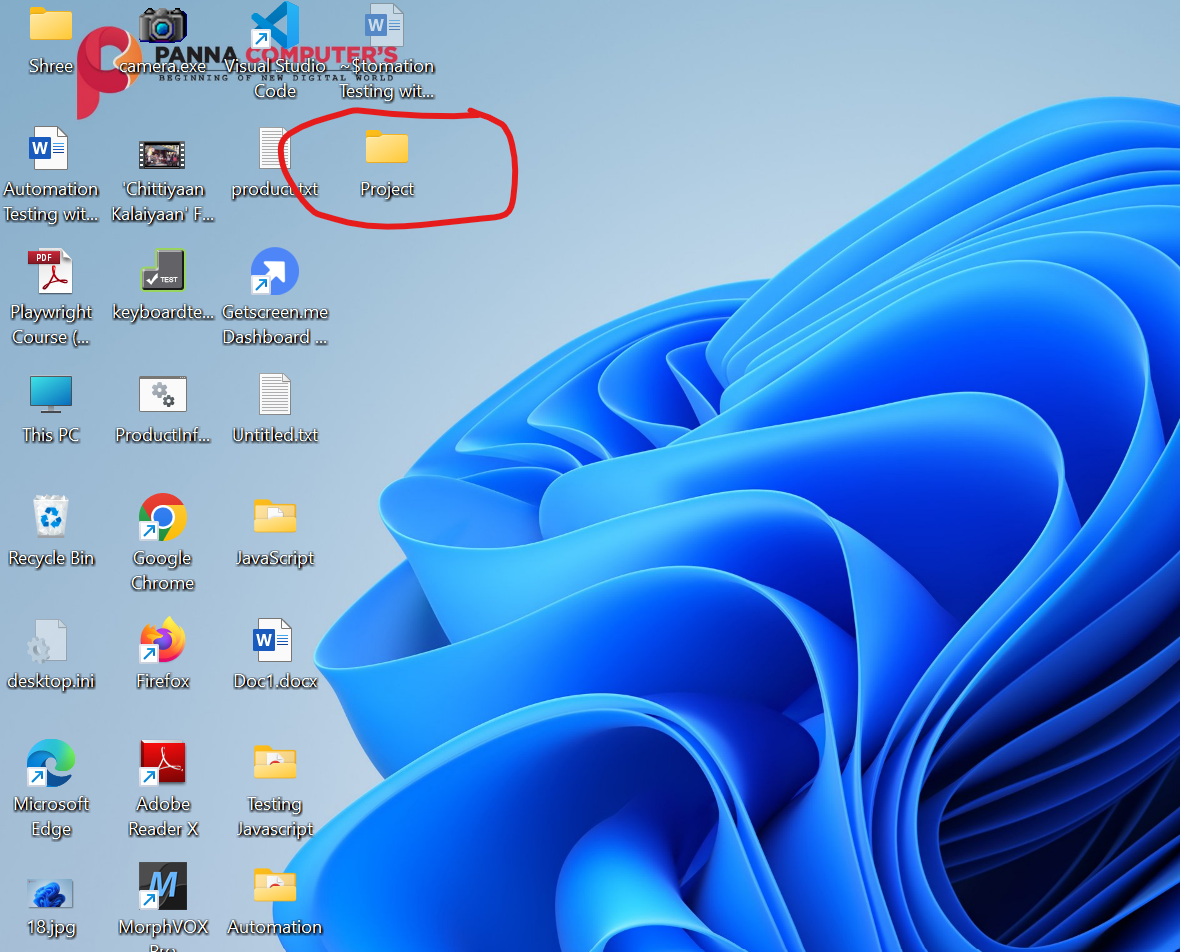
8)Multi tab and multi widows support

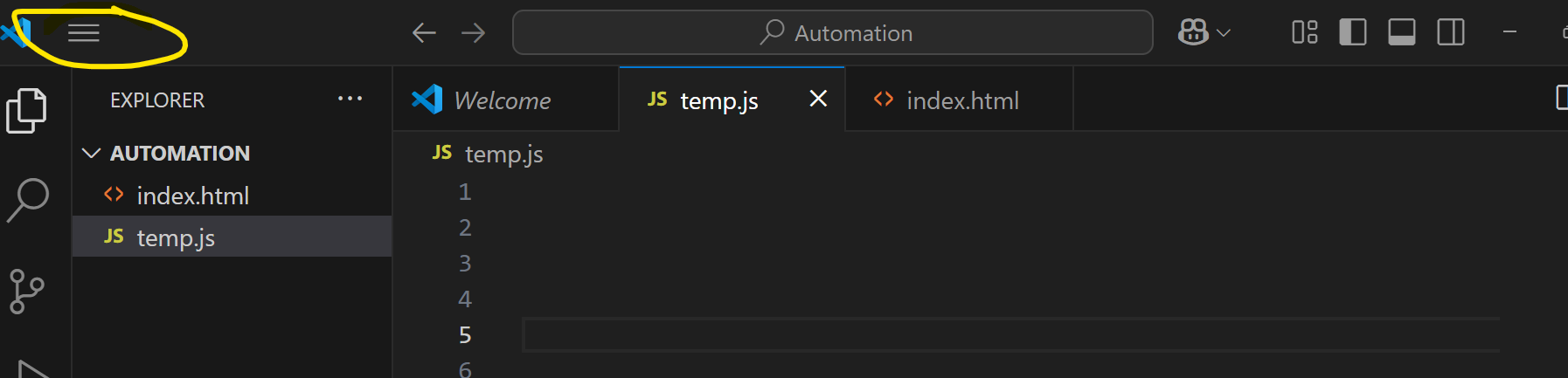
9)Built-in reporters, custom reports

10)Test parameter

11)It is faster

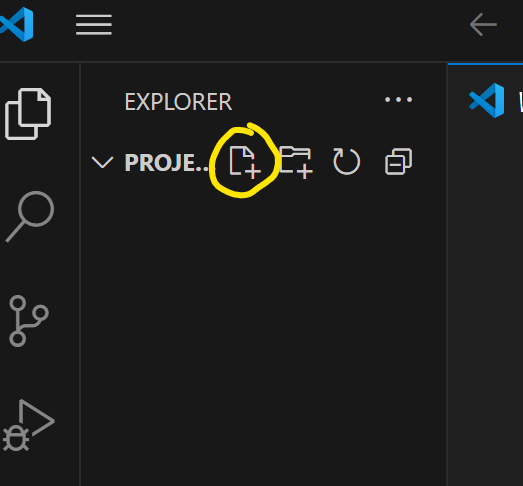
|  |  |  |
| --- | --- | --- |
| **Features** | **Playwright** | **Selenium** |
| Browser support | Chromium ,Web Kit and Firefox etc | Chrome, Firefox, Edge etc |
| Auto wait | Yes | No |
| API Testing | Yes | No |
| Built-in Report | Yes | Depend on testNG |
| Code generator | Yes | NO |
| Built in Assertions | Yes | No |
| Debugging | Built-in debuggers | We require to integrate with Browser developer tools |
| Parallel testing | Yes | Will be depends on TestNG |
| Performance | High Performance | Good performance |
| Community support | Growing community | Large community |

****

**Open VS code and **

**File Open Folder Select Folder**

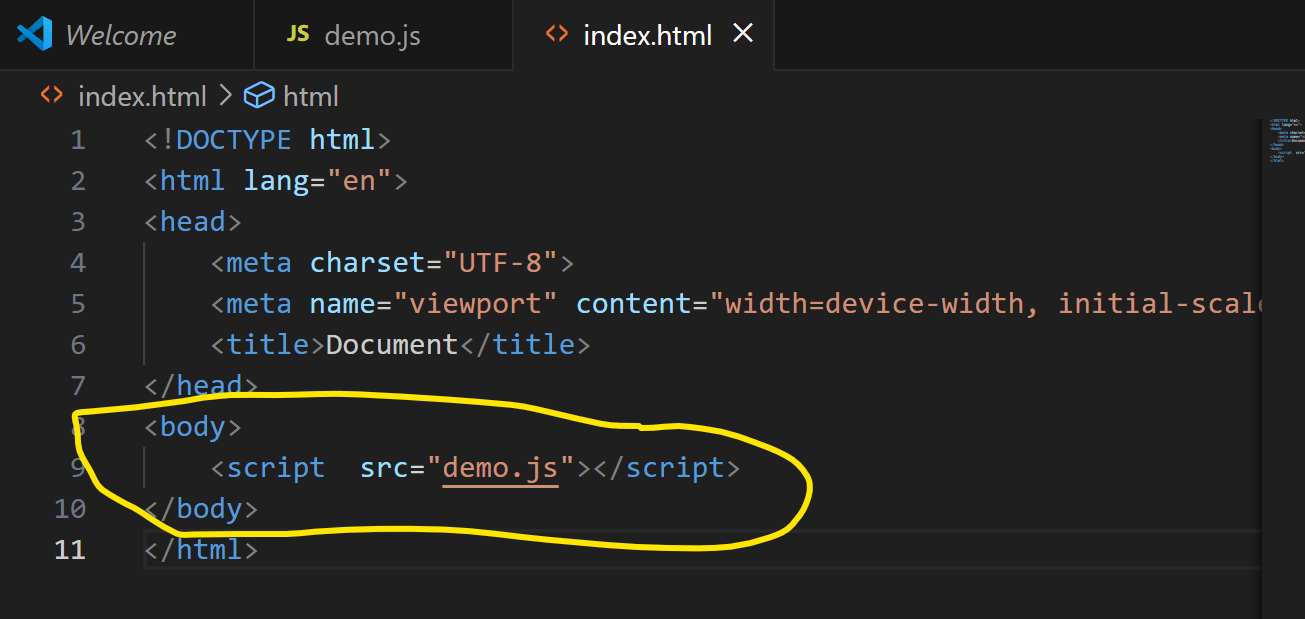
**1)Create file with any\_name.js extension.**

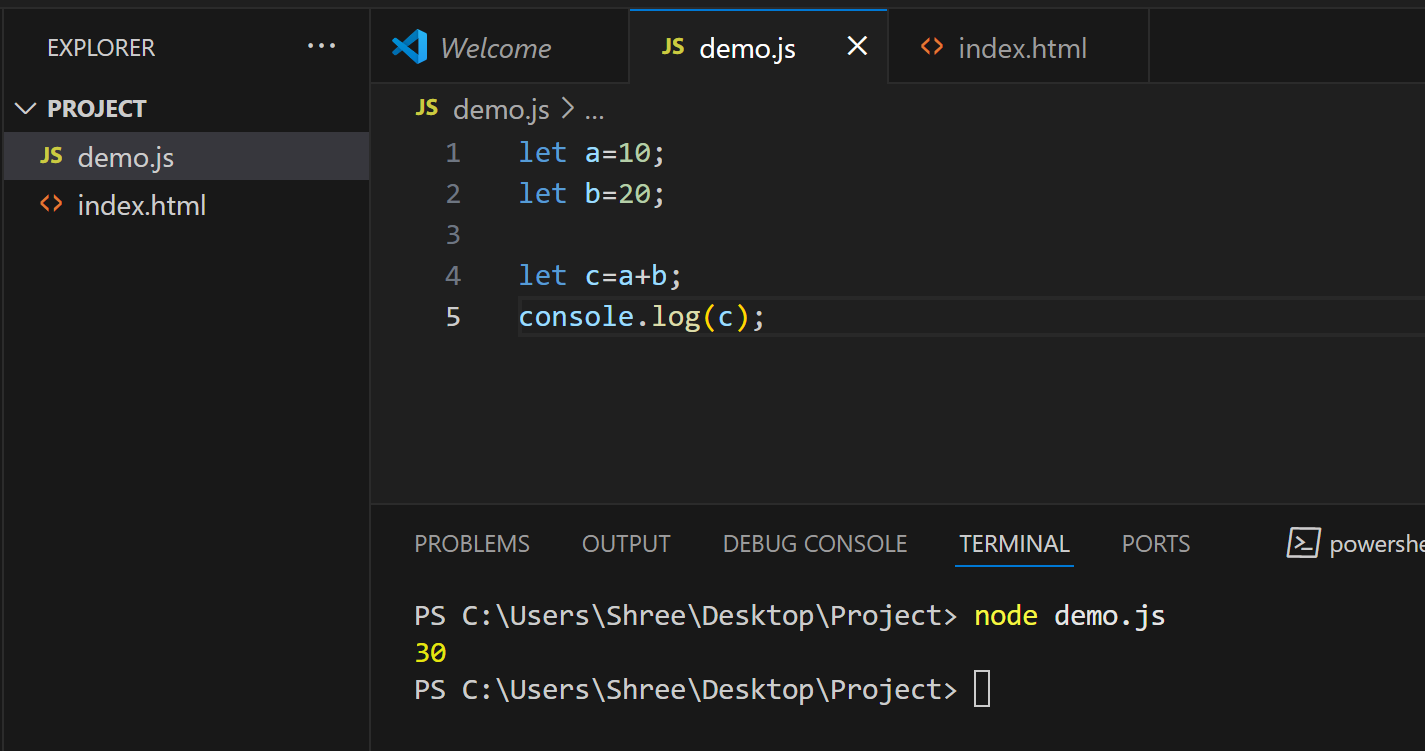
****

**2)To connect with browser create .html file.**

****

**Press** (shift+!) **Mark to create html code.**

****

****

**Press**(ctrl+l)**to clear the terminal.**

(::line\_number) provides you to go at that line number.

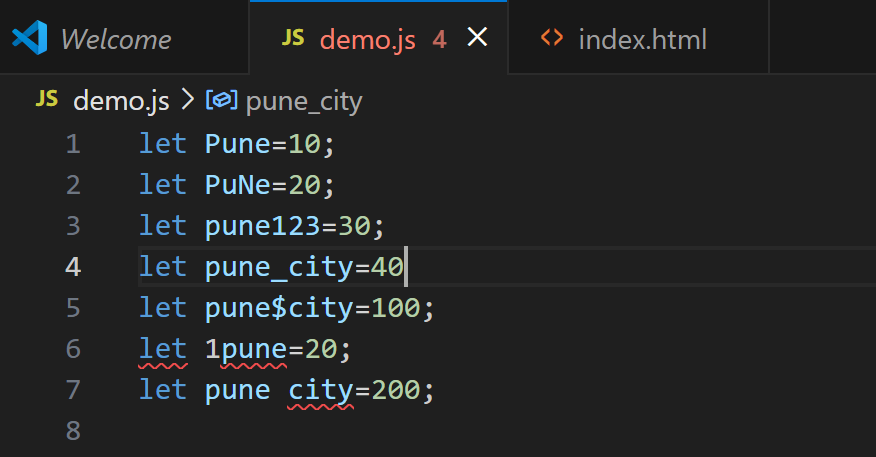
**Javascript variable rules-**

1)Variable names are case sensitive “a” and “A” is different.

2)Only letters,digits,underscore(\_) and $ is allowed.(no space allowed).

3)Only a letter,underscore(\_)or $should be 1st character.

4)Reserved words cannot be variable name.

****

**Javascript Keywords-**

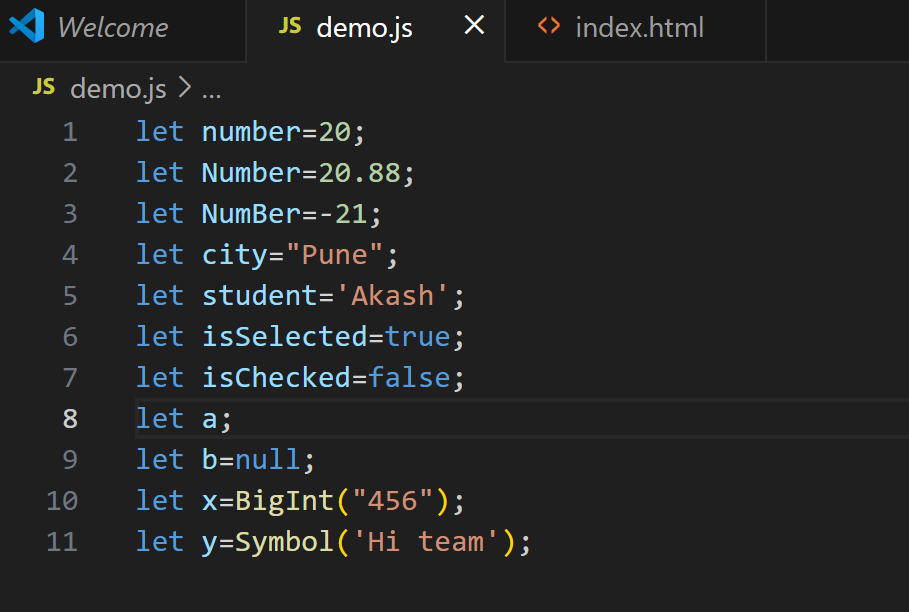
**var**=Variable can be redeclared and updated a global scope.

**let** =Variable cannot be redeclared but can be updated , block scope.

**const**= Variable cannot be redeclared or updated ,block scope.

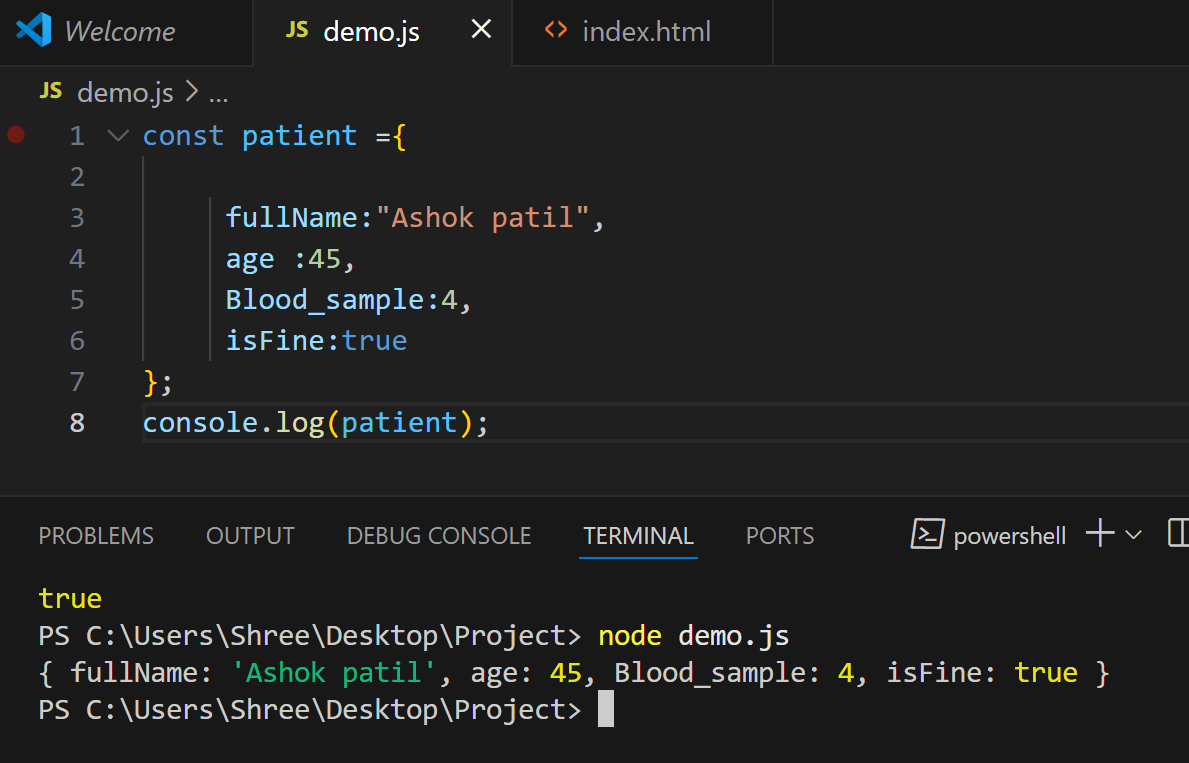
**Data types in JS-Primitive**

Number,String,Boolean,Undefined,Null,BigInt,Symbol.

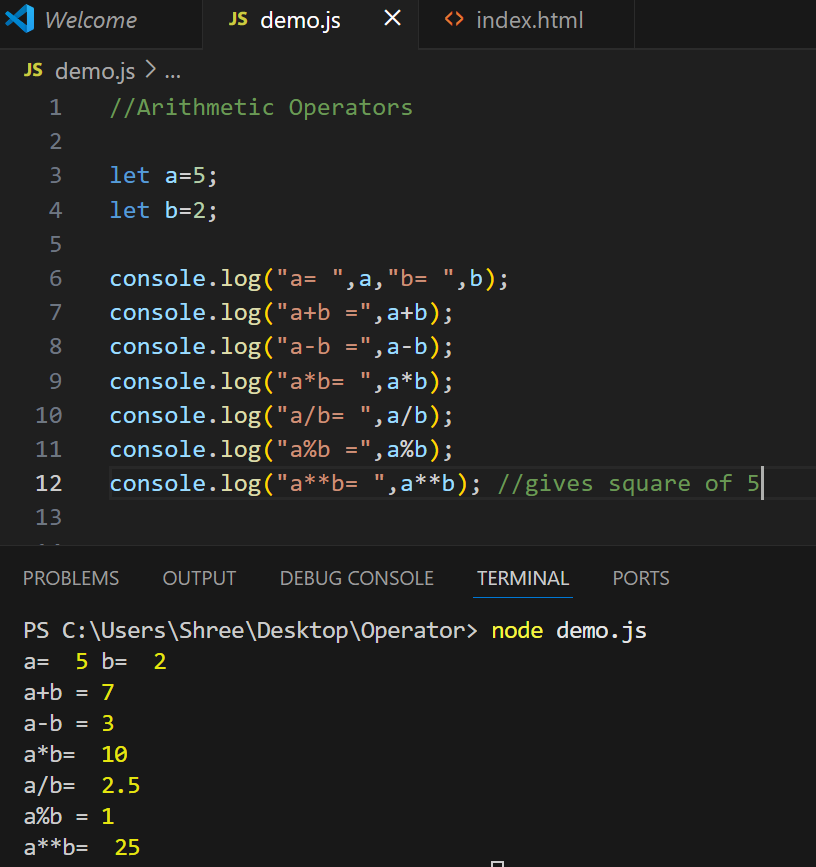
****

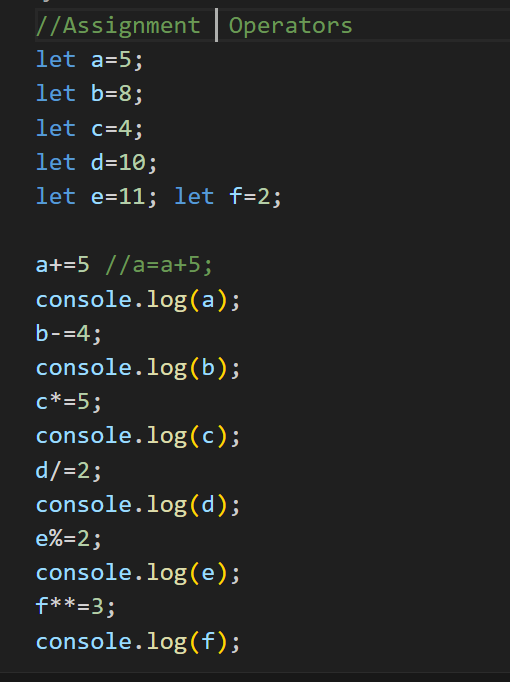
**Non-primitive-** objects,Arrays,Functions

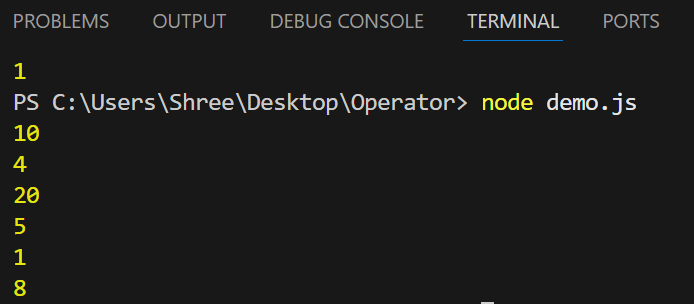
**Objects -**it is collectionof values.

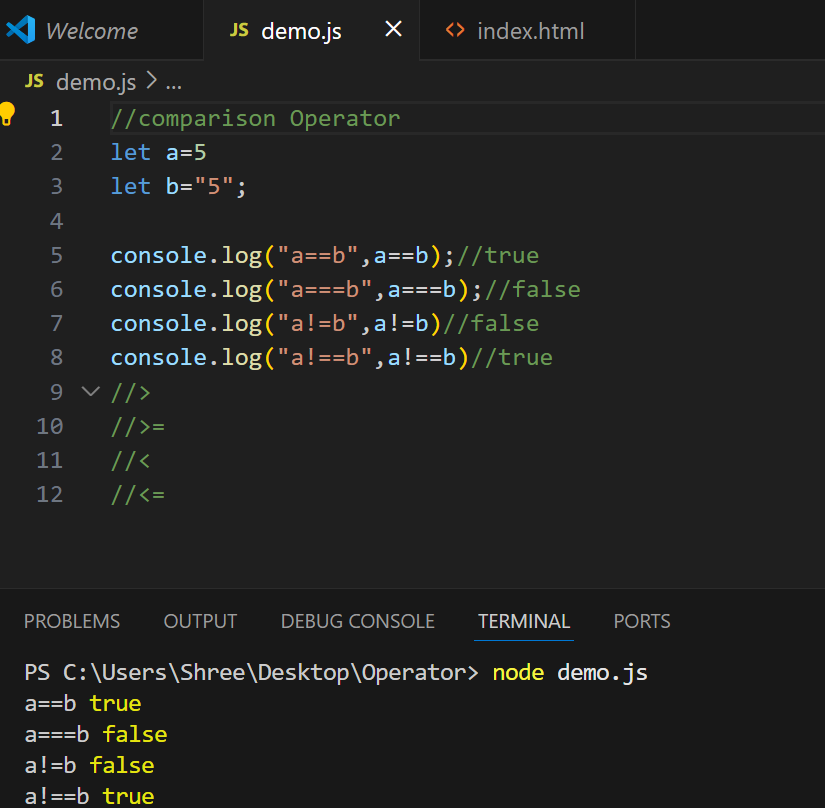
****

**Operators and conditional statements**

****

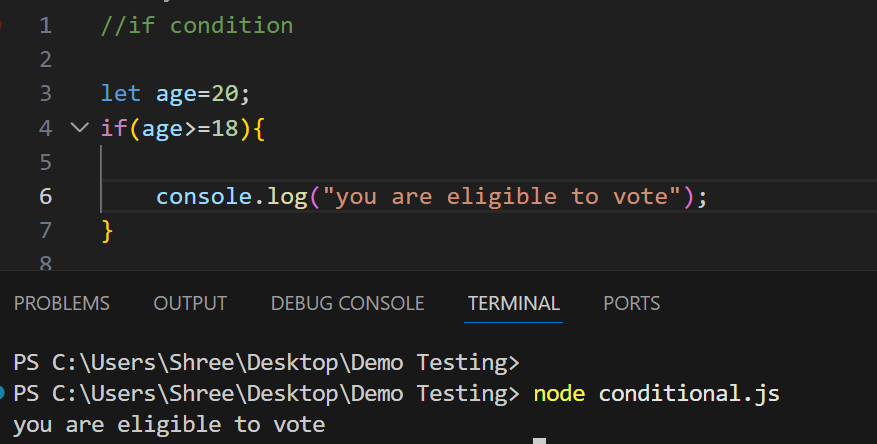
****

****

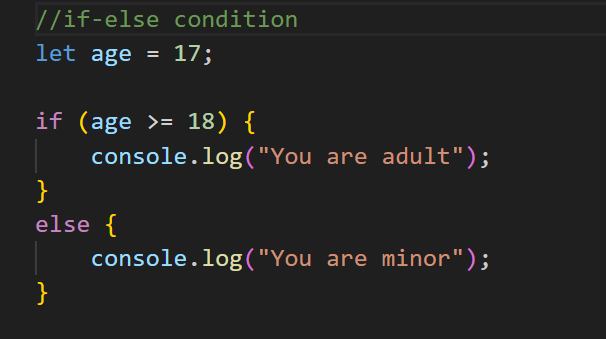
****

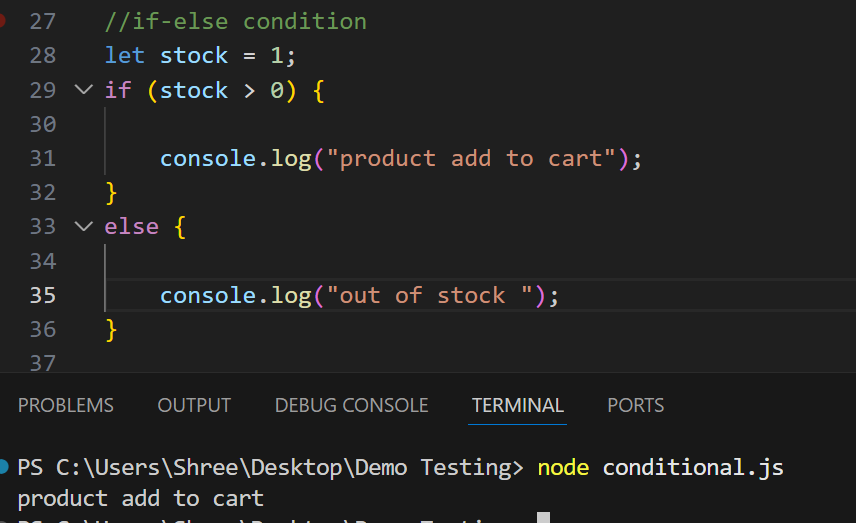
**Conditional statements –**

**If condition**

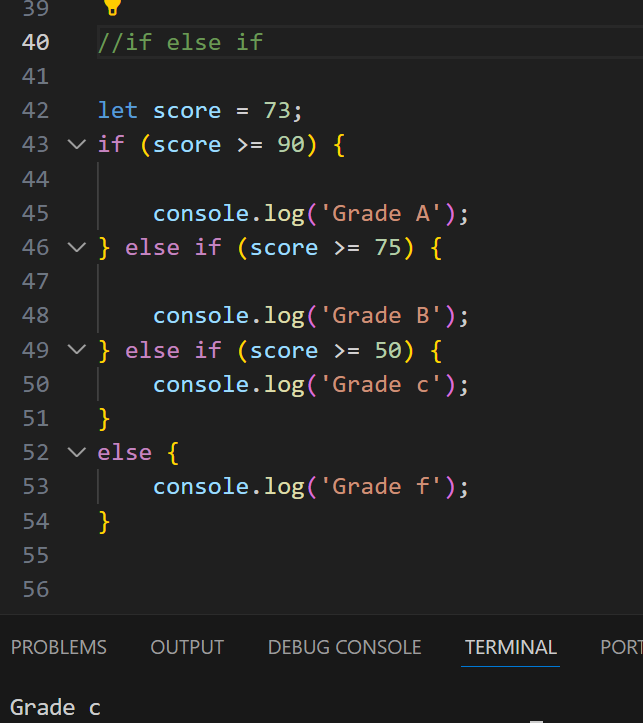
****

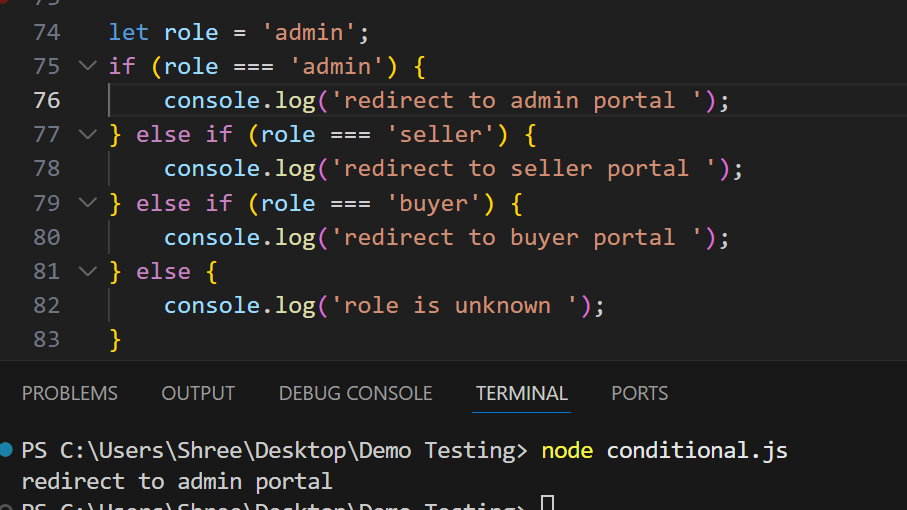
**If else condition –**

****

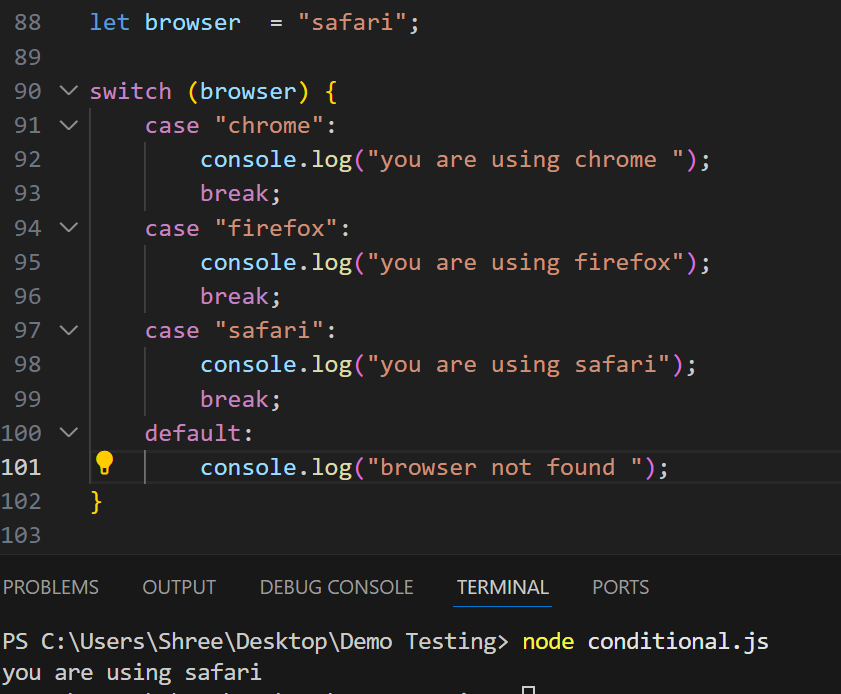
****

**If else if**

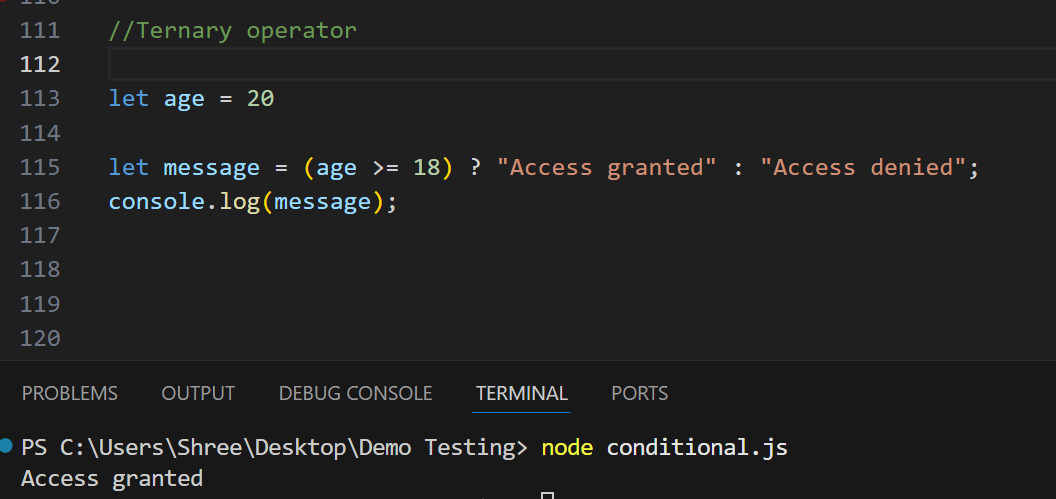
****

****

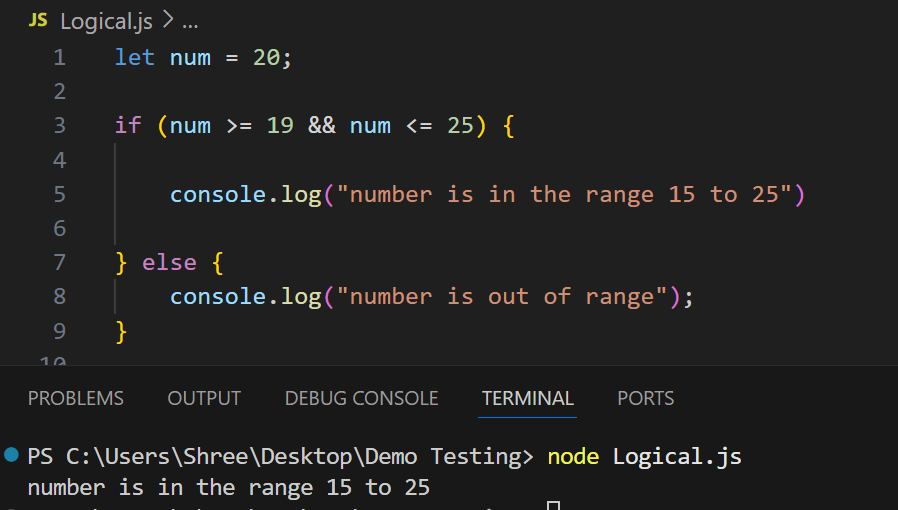
**Switch condition**

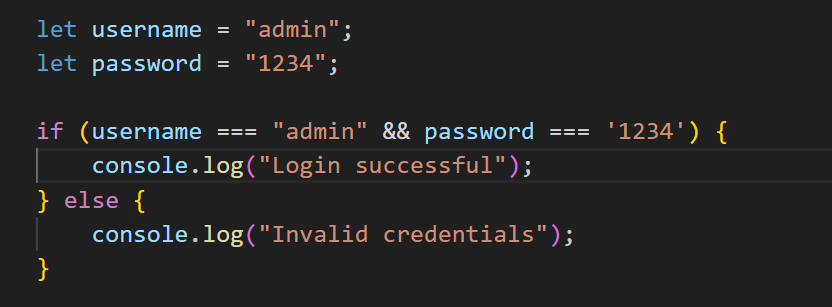
****

**Ternary operator**

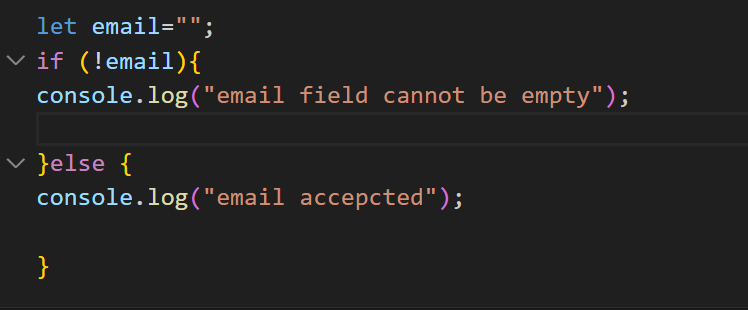
****

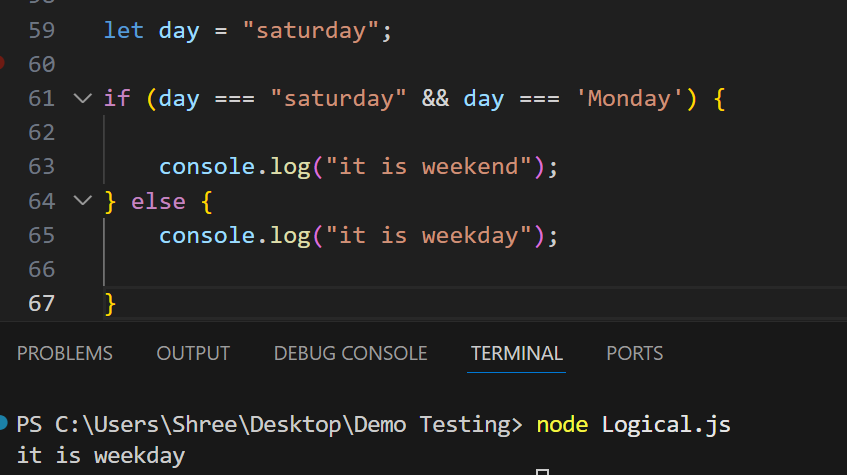
**Logical and operator –**both condition required to be pass

****

****

**Logical ! operator**

****

****

**OR operator-**if one condition passes out of two then if code executed**.**

****

**Loop Statements**

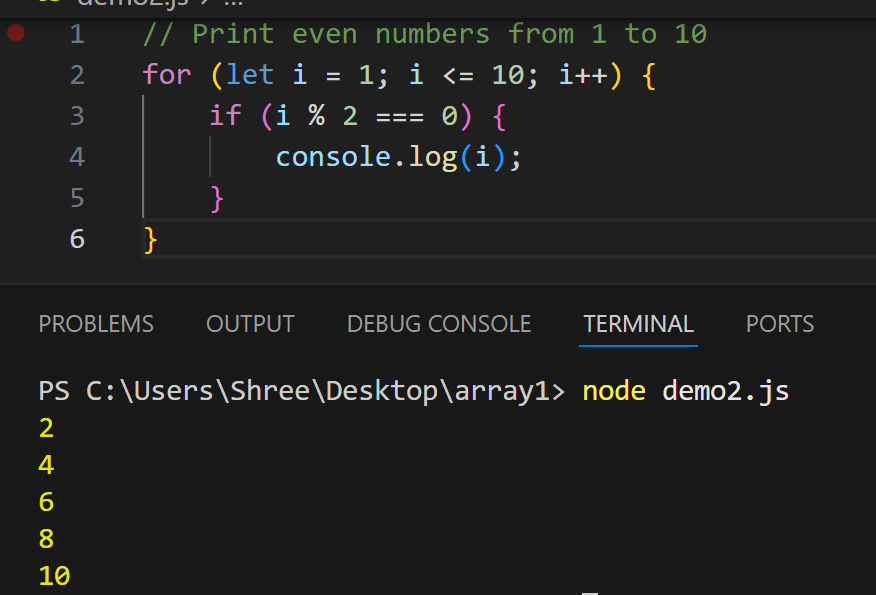
**For LOOP----->**

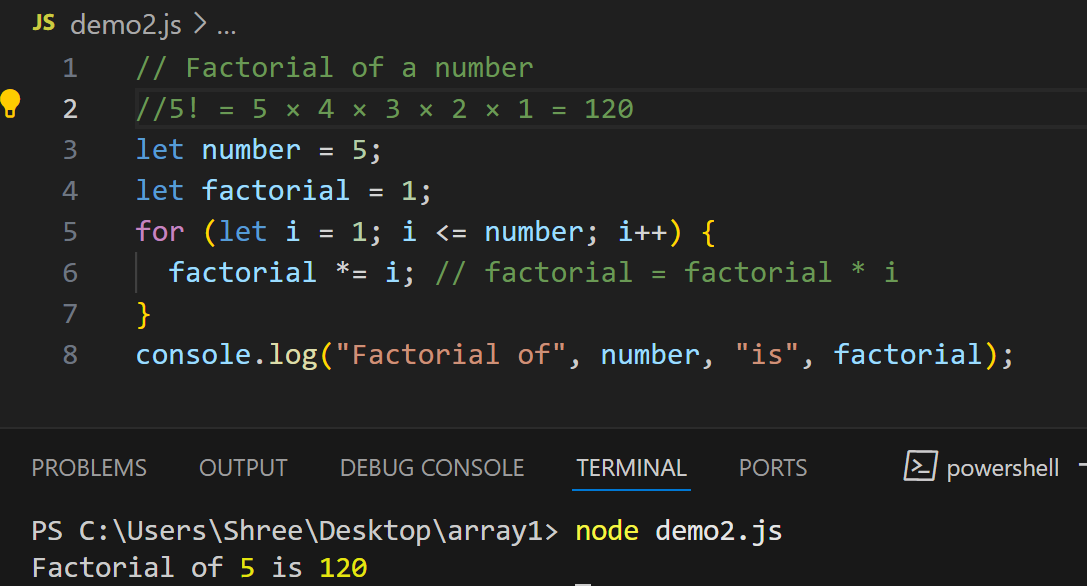
**for (initialization; condition; increment) {**

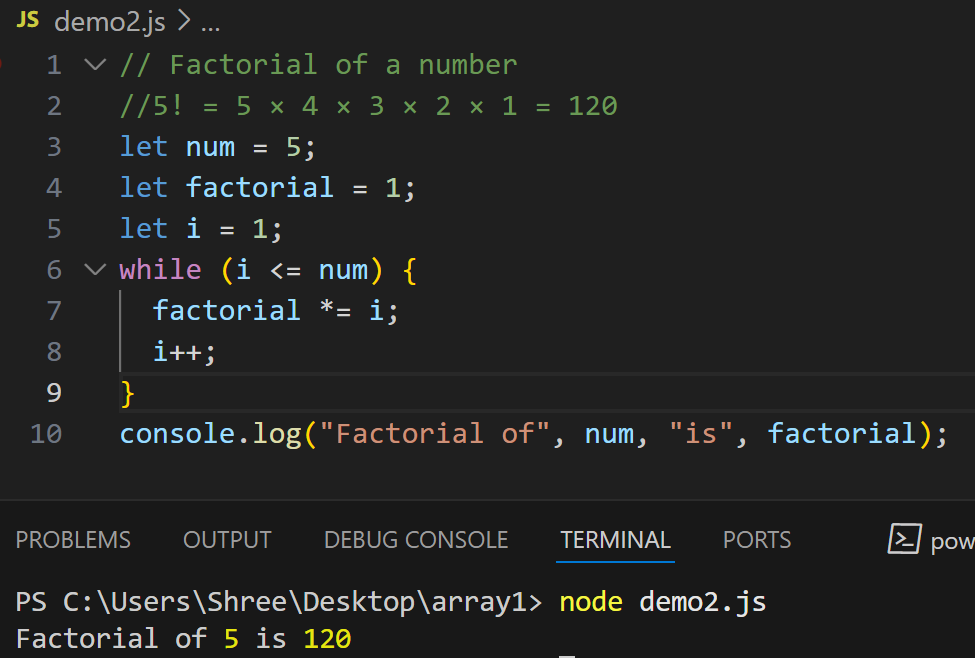
**// code to be repeated**

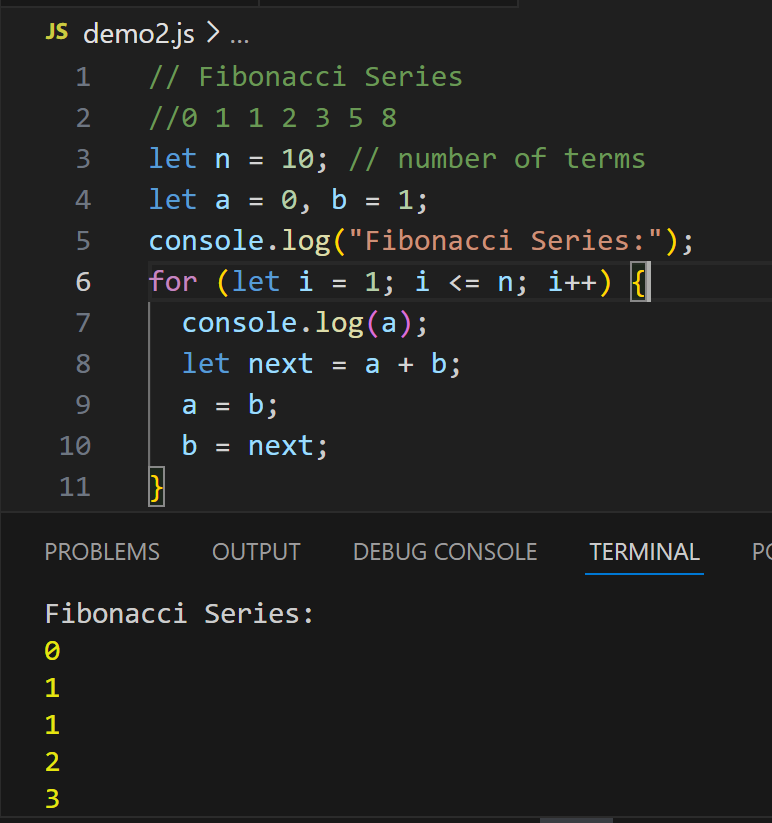
**}**

1. **Initialization** – Set a starting value (usually a counter like let i = 0)
2. **Condition** – The loop runs **as long as this condition is true**
3. **Increment/Decrement** – Updates the counter (i++, i--, etc.)







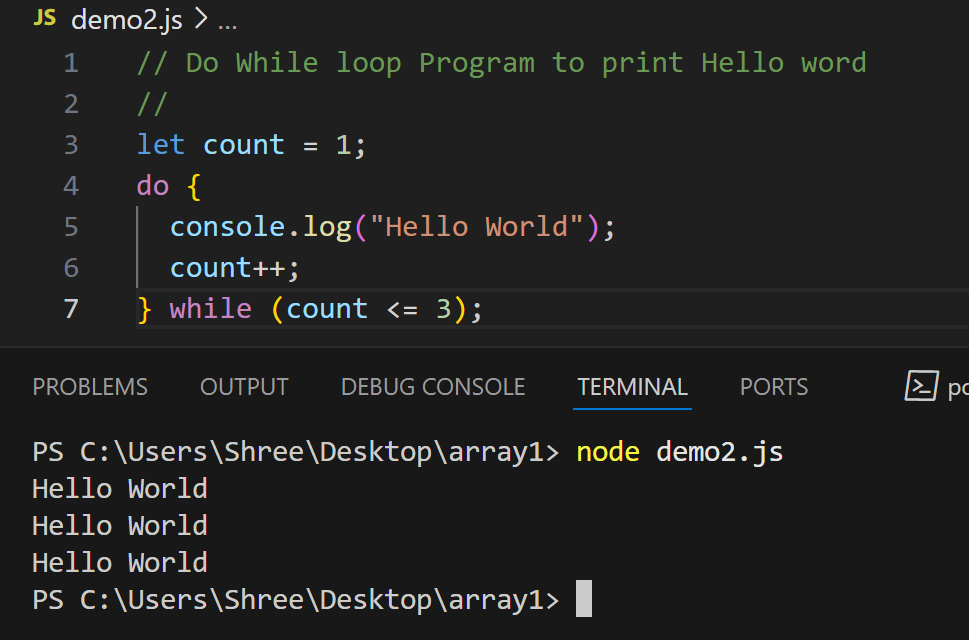


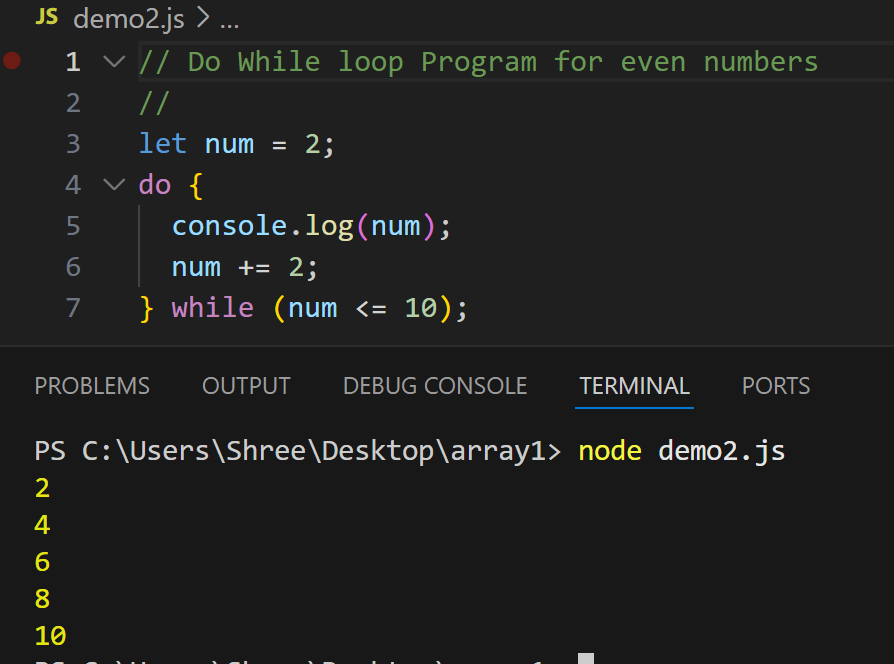
**Do\_\_while**

**do {**

**// code block**

**} while (condition);**

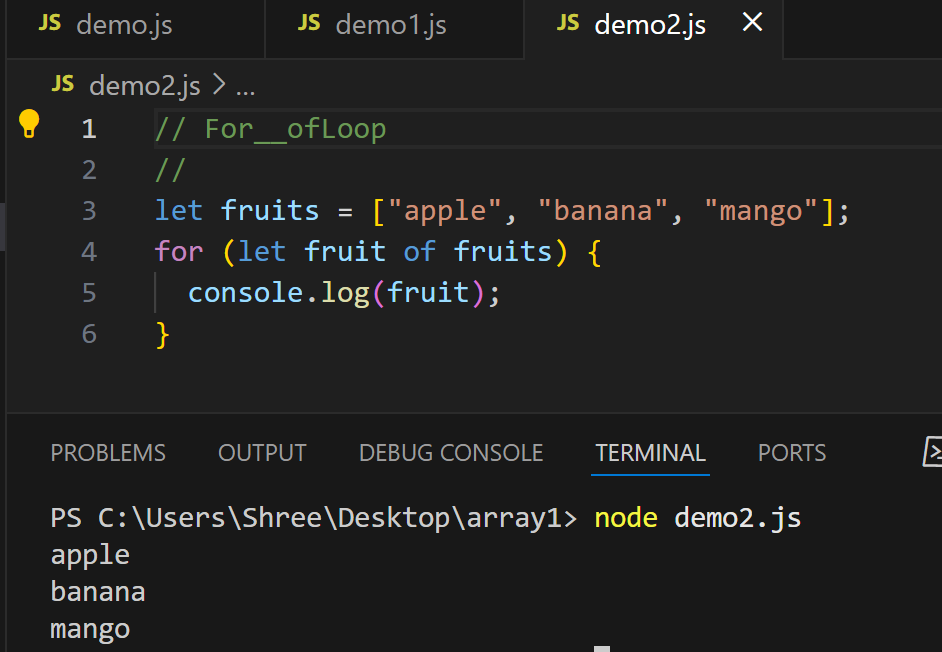
****

****

**for...of loop**

for (let item of iterable) { // use item

}

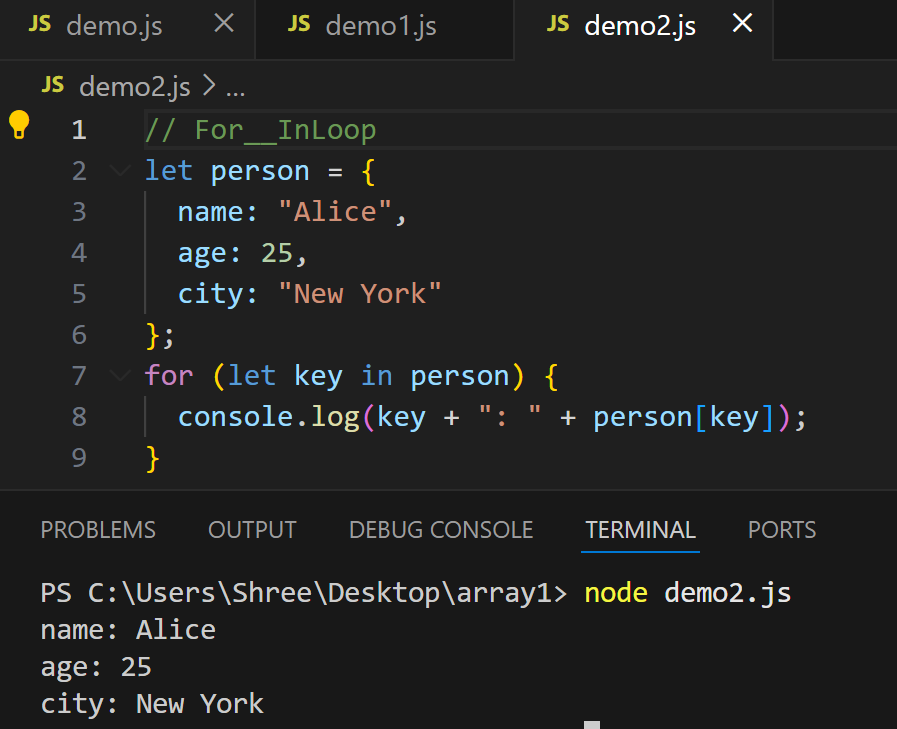


**for...in loop**

for (let key in object) {

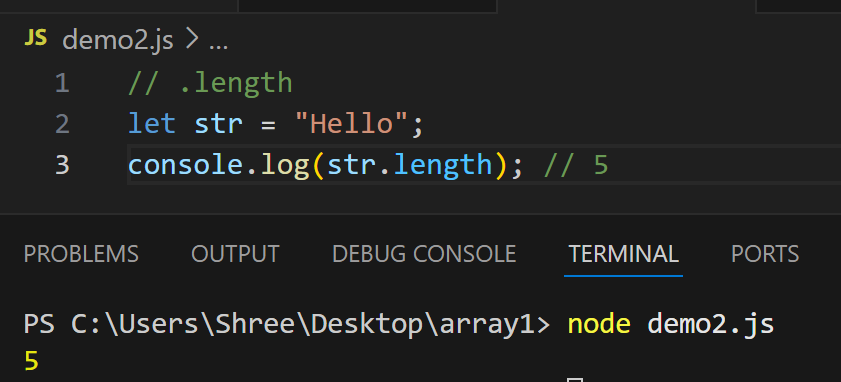
// use key and object[key]

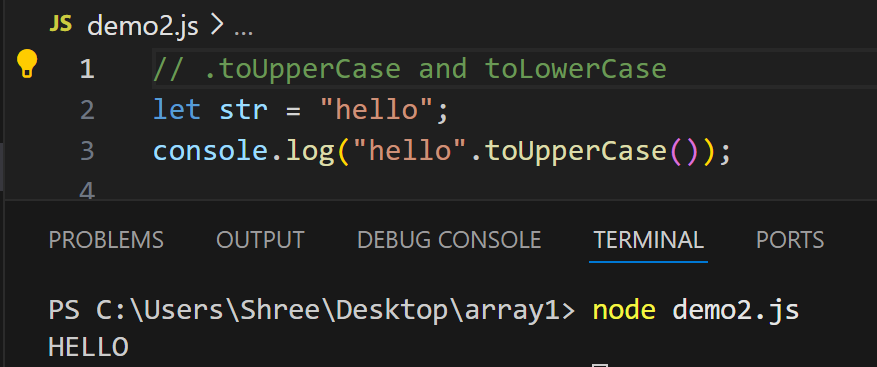
}



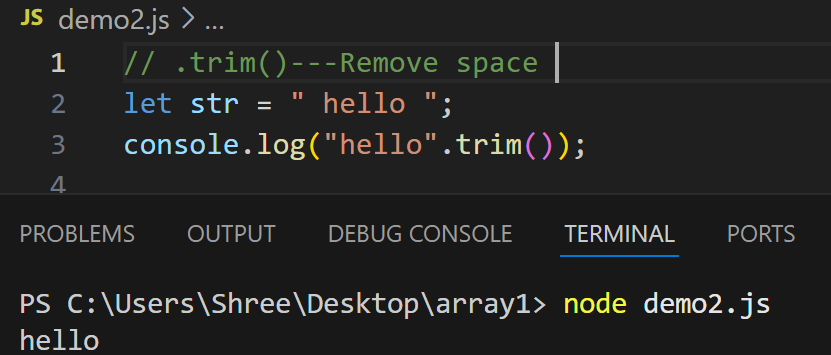
**Common JavaScript String Methods**

**.length**

****

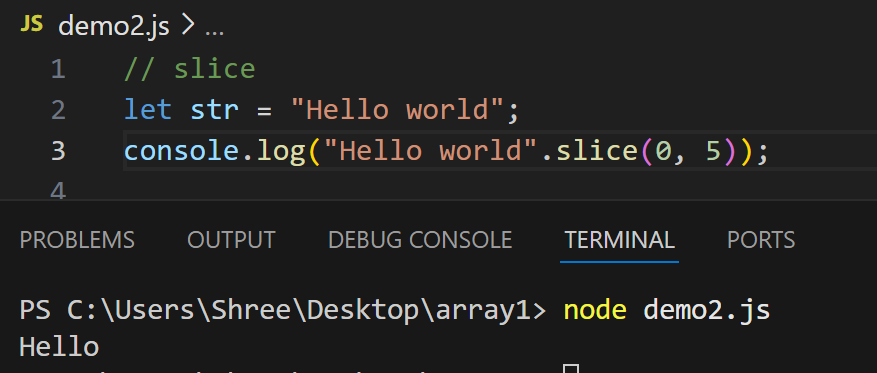
****

**.trim()**

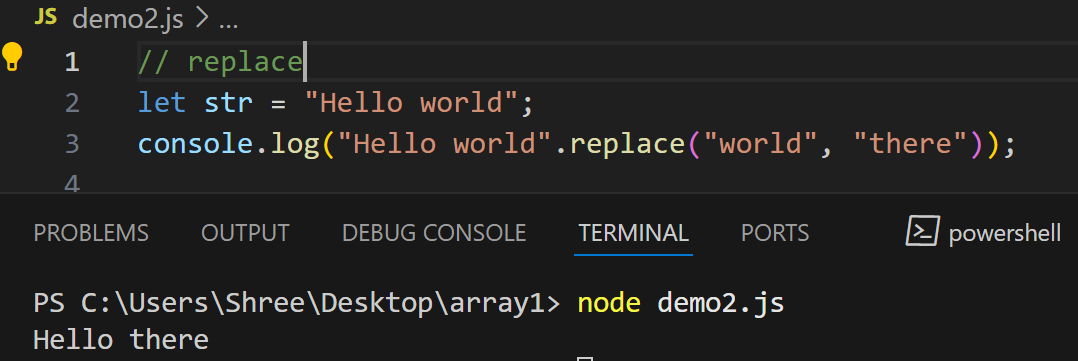
****

**.slice(start, end)**

Extracts a part of the string from start index up to (but not including) end index.

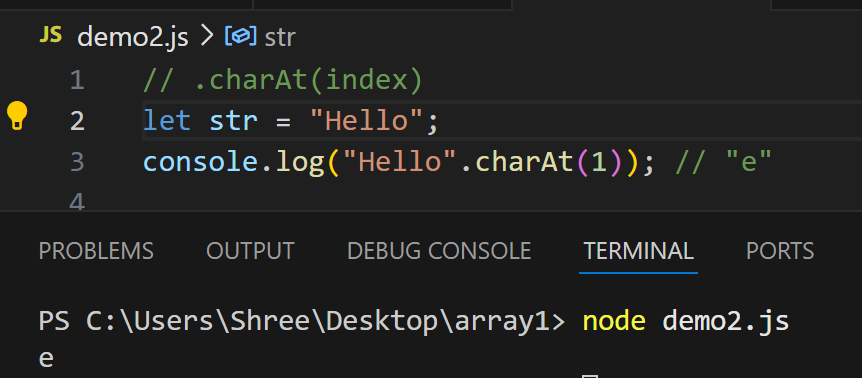
****

.replace(oldText, newText)

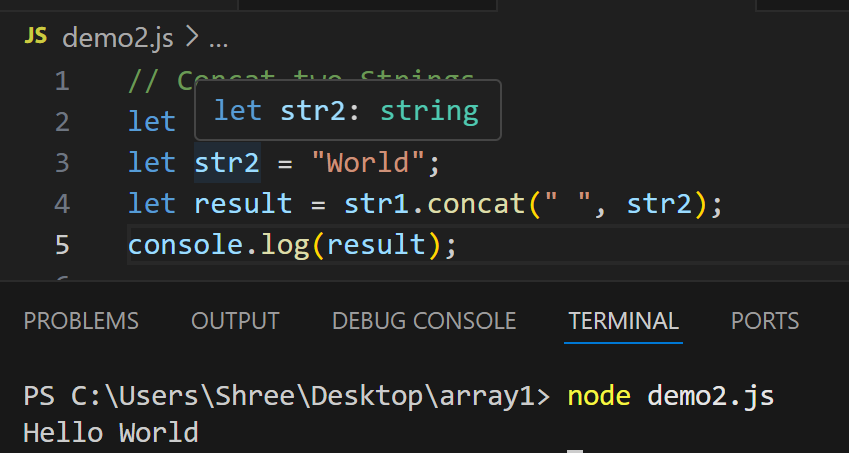
****

**.charAt(Index)**

Returns the character at the specified index.

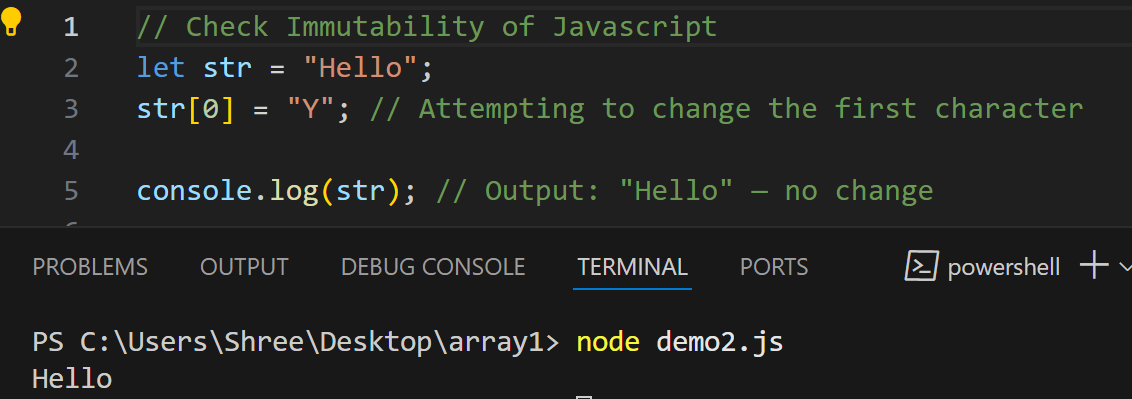
****

**.concat()----Join Two Strings**

****

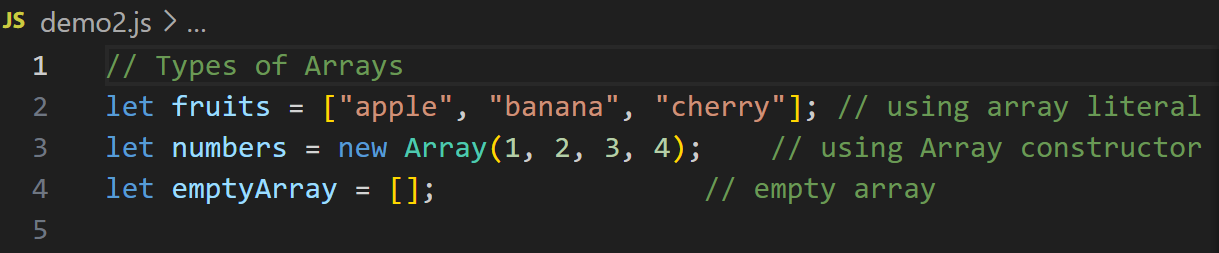
**Strings are immutable in JavaScript.**

immutability means that once a string is created, it cannot be changed. Any operation that appears to modify a string actually creates a new string instead.



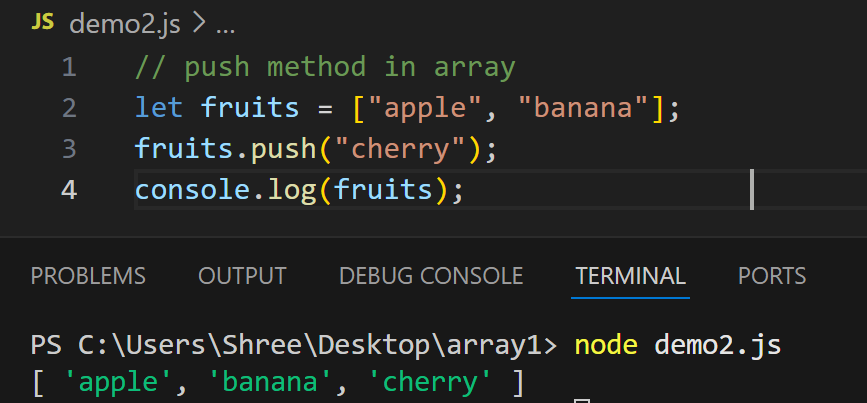
**Arrays**

Arrays in JavaScript are a type of object used to store multiple values in a single variable. They are dynamic, meaning their size can change, and they can hold different types of elements (numbers, strings, objects, etc.) in the same array.



| **Method** | **Description** |
| --- | --- |
| push() | Adds item to the end |
| pop() | Removes last item |
| unshift() | Adds item to the beginning |
| shift() | Removes first item |
| splice() | Adds/removes elements at specific index |
| slice() | Returns a shallow copy |
| indexOf() | Finds index of an element |
| includes() | Checks if element exists |

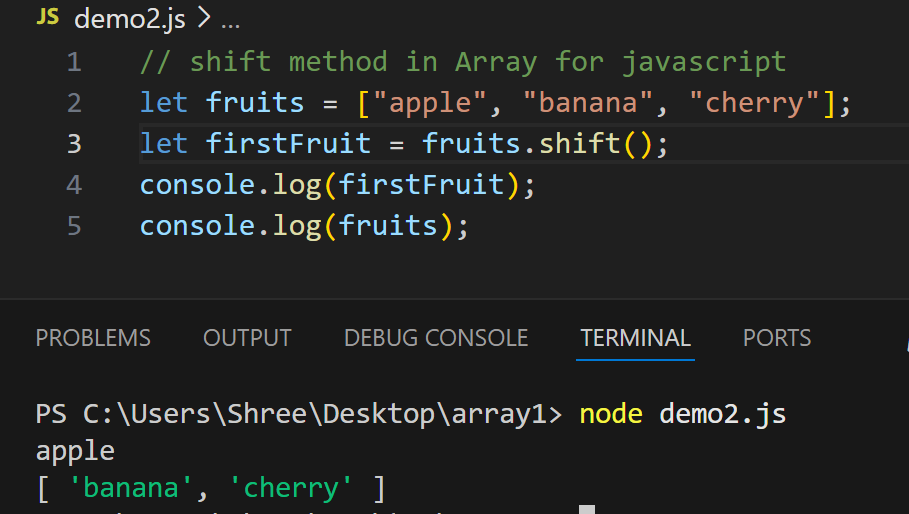
**push()-**This method in JavaScript is used to **add one or more elements to the end of an array**. It modifies the original array and returns the **new length** of the array.



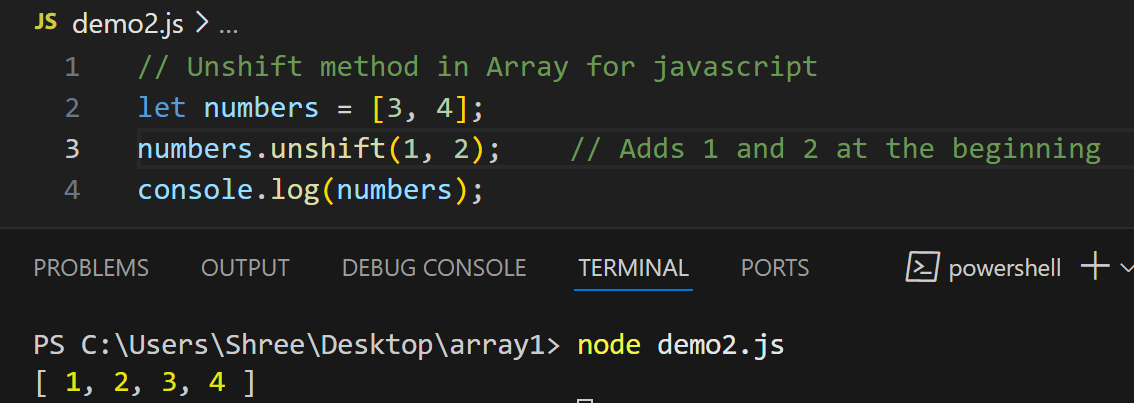
**pop()-**Thismethod in JavaScript is used to remove the last element from an array. It modifies the original array and returns the removed element.

****

**shift()-** This method in JavaScript is used to remove the first element from an array. It modifies the original array and returns the removed element.

****

**Unshift()-**This method in JavaScript is used to add one or more elements to the beginning of an array. It modifies the original array and returns the new length of the array.

****

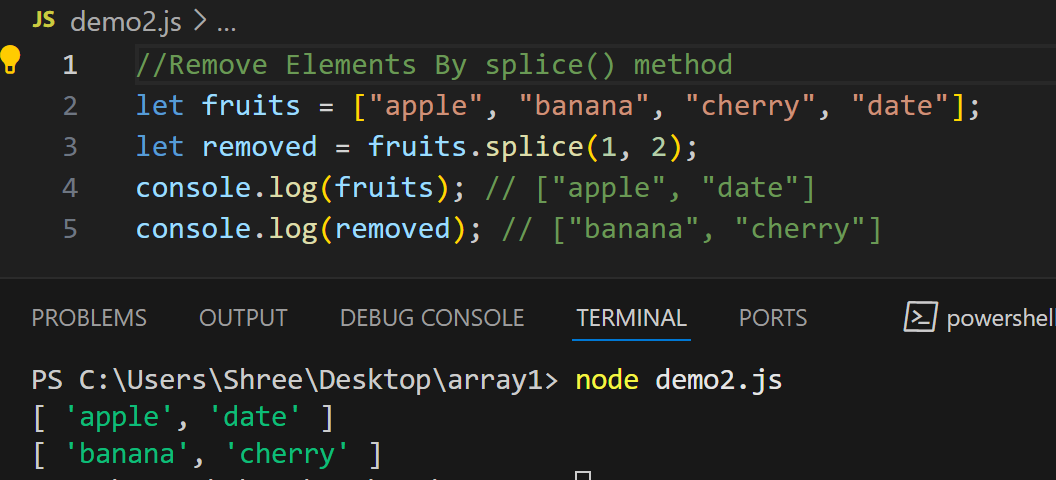
**splice()-** This method in JavaScript is one of the most powerful and flexible array methods. It is used to add, remove, or replace elements in an array by modifying the original array.

**array.splice(startIndex, deleteCount, item1, item2, ...);**

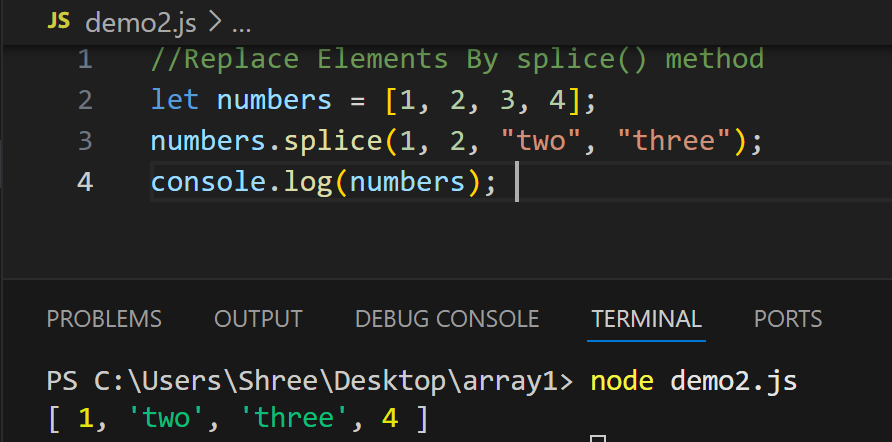
**startIndex →** The position to start changing the array.

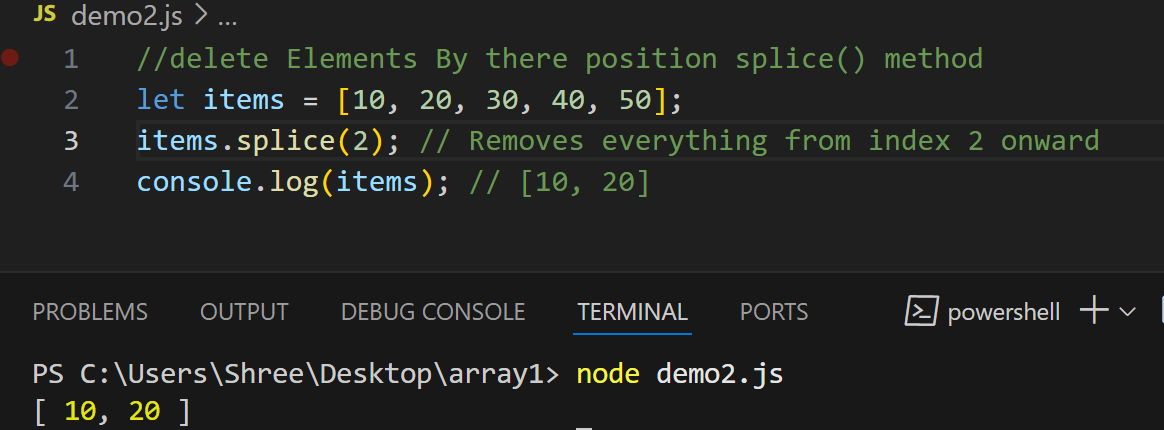
**deleteCount →** Number of elements to remove.

**item1, item2, ... →** Optional items to add in place of removed elements.

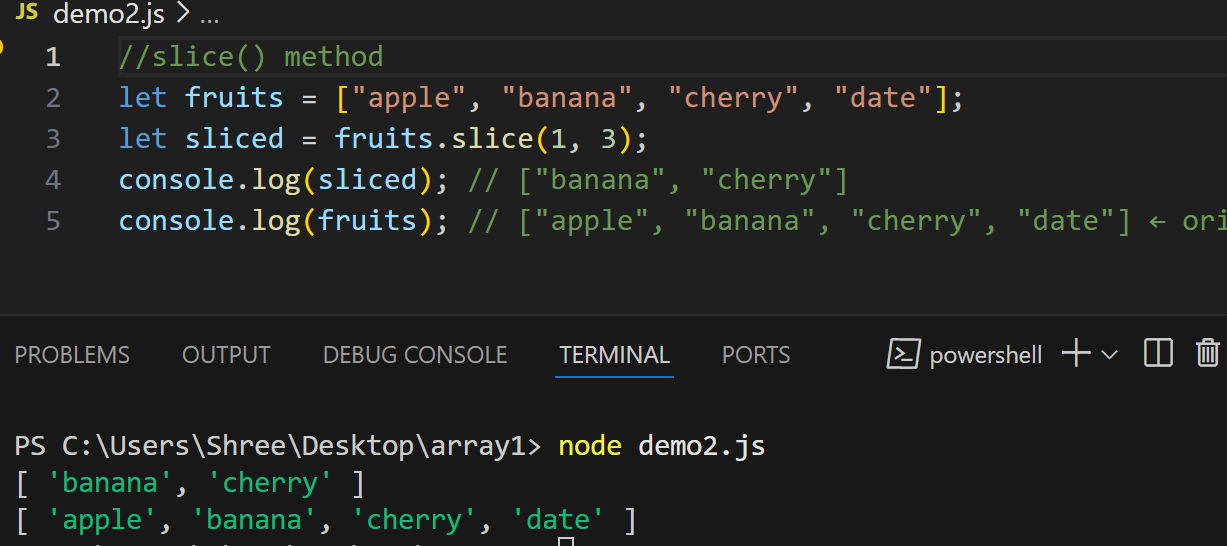
****

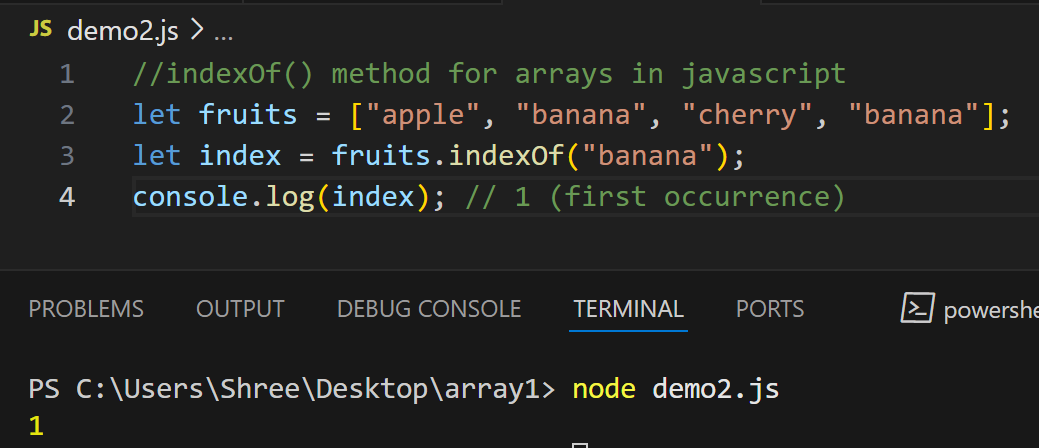
****

****

****

**slice() –** This method in JavaScript is used to create a shallow copy of a portion of an array without modifying the original array.

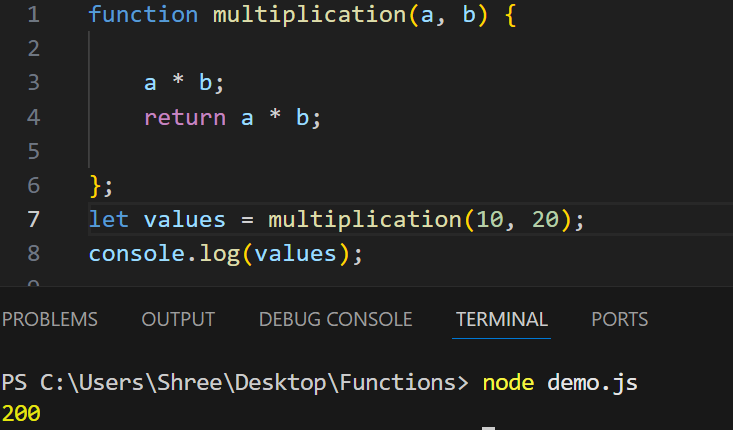
****

****

**includes() -**This method in JavaScript is used to check if an array contains a specific value. It returns a boolean (true or false) and does not modify the array.

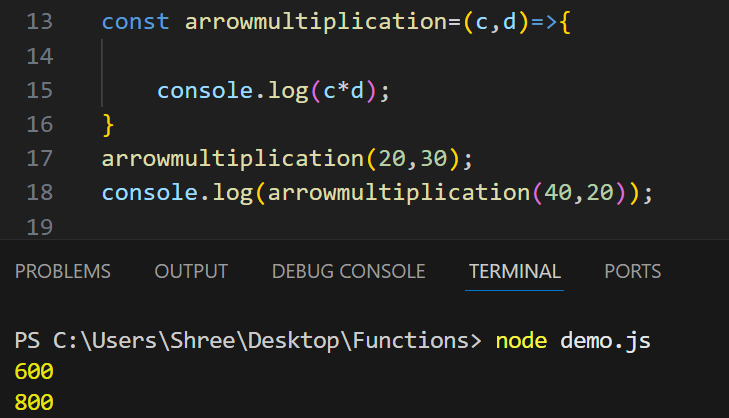


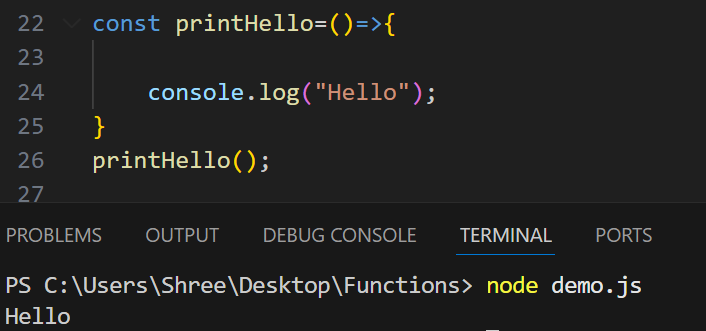
**In JavaScript, functions are reusable blocks of code that perform a specific task.**



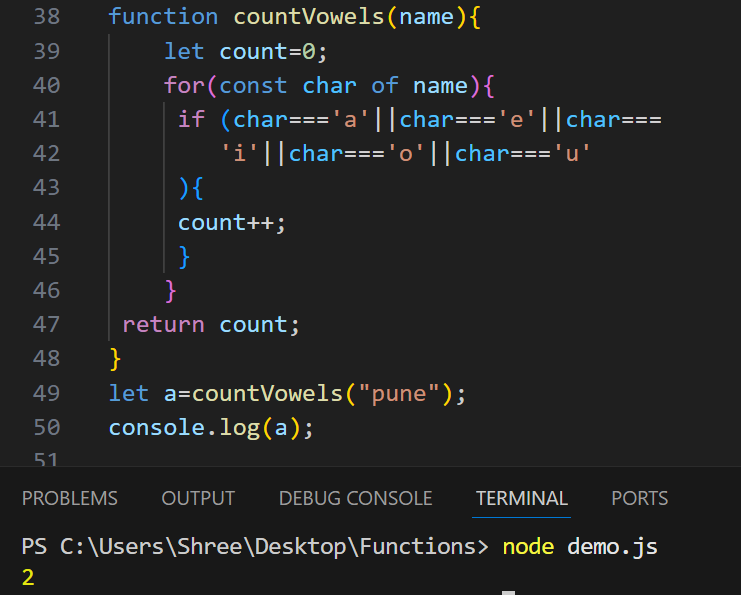
**Arrow Functions in JavaScript**

Arrow functions are a **shorter way** to write function expressions,



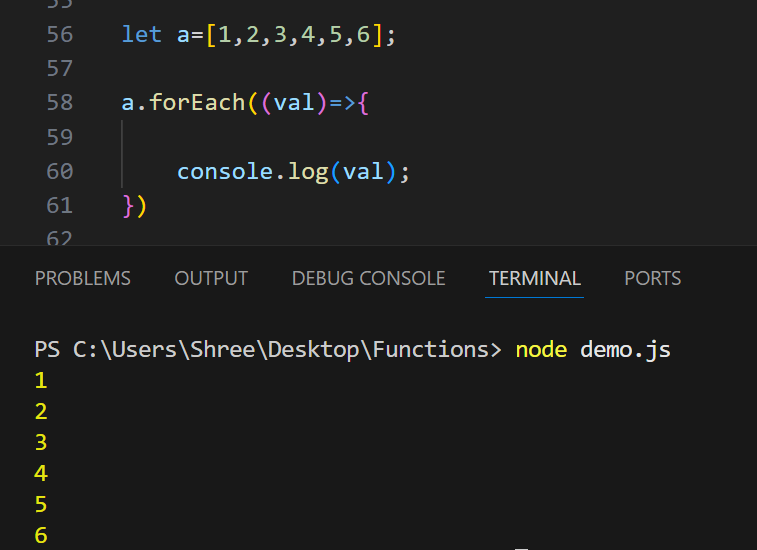


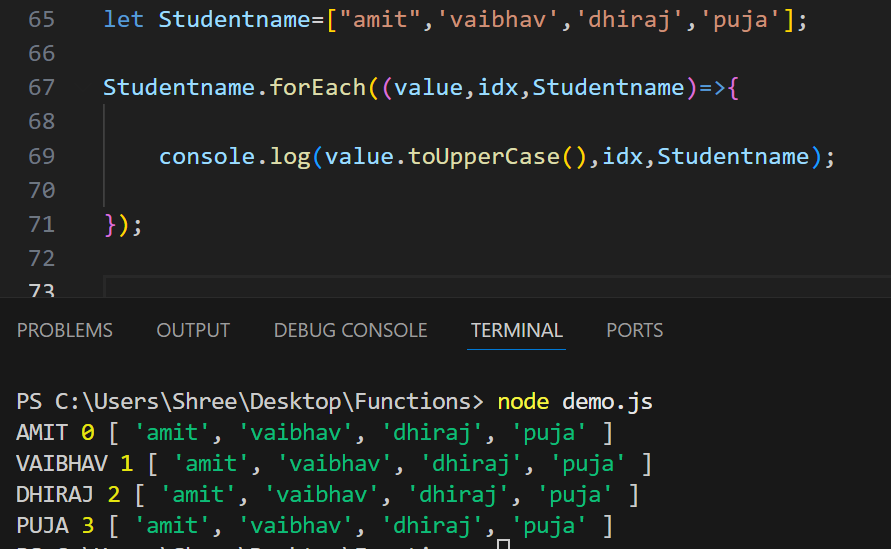




**forEach() in JavaScript**

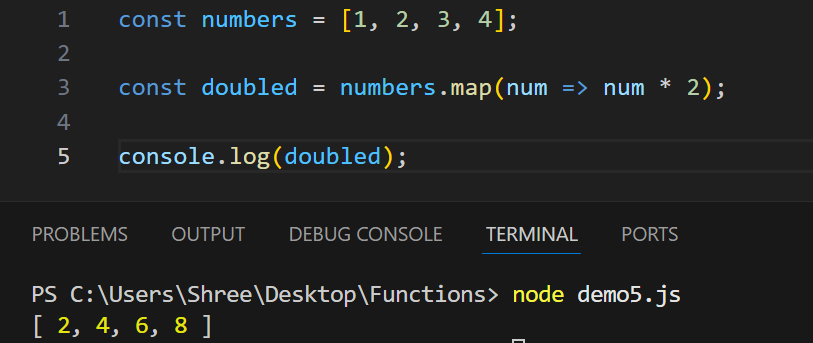
The .forEach() method is used to **iterate over arrays** and execute a **function once for each array element**.

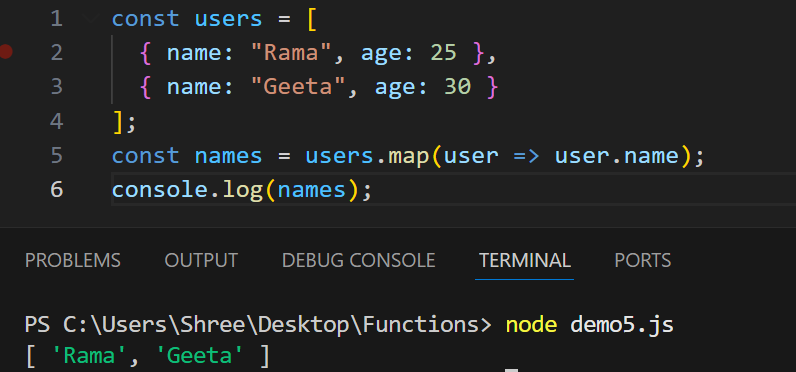




**map() in JavaScript**

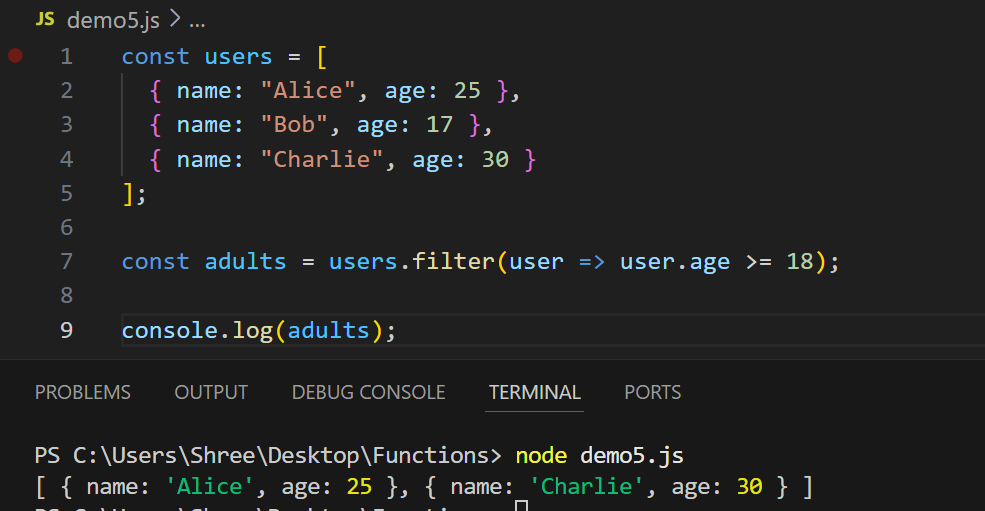
The .map() method creates a **new array** by calling a **function on each element** of the original array.

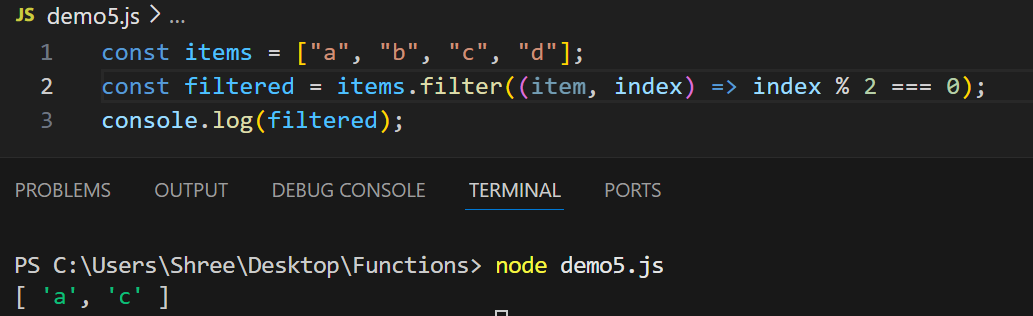


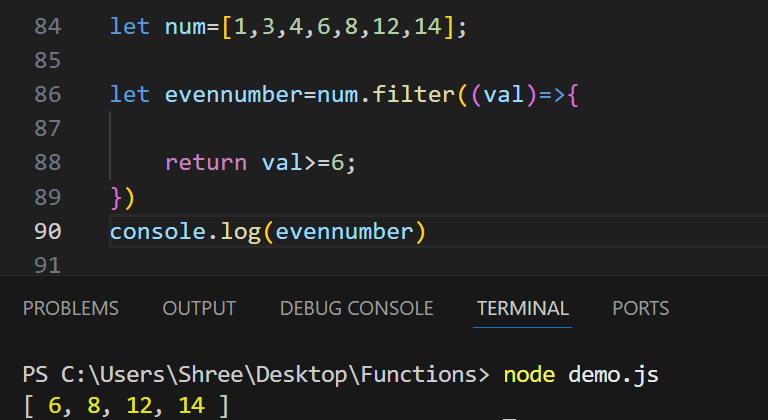


**filter() in JavaScript**

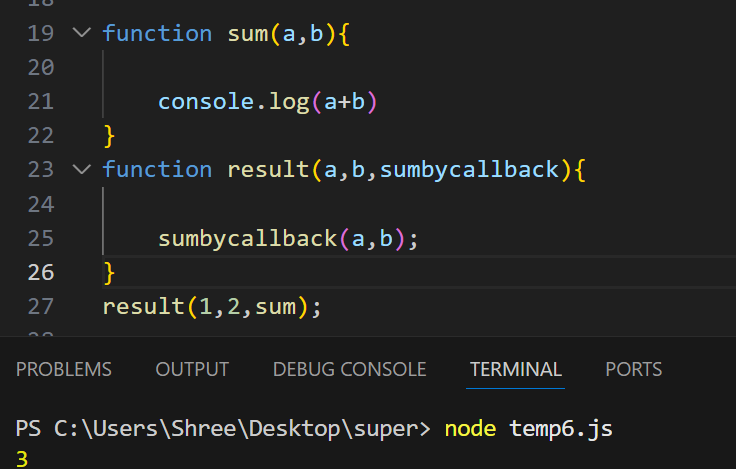
The .filter() method is used to **create a new array** with **only the elements** that pass a **specific test condition**.



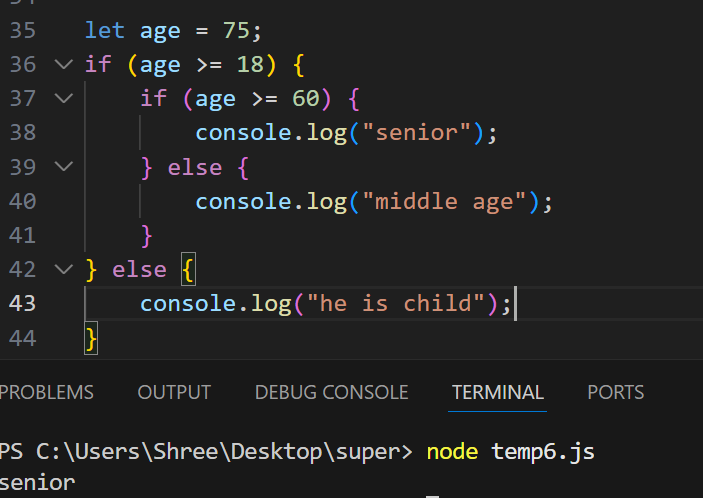




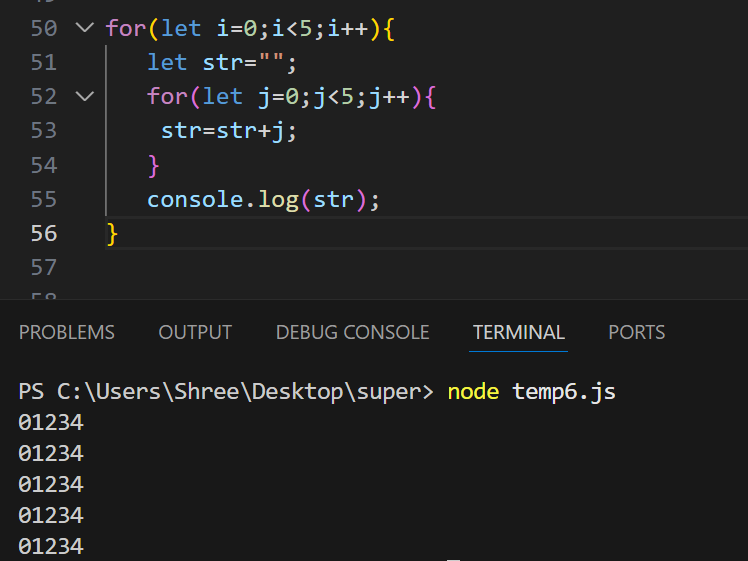
In JavaScript, a **callback** is a **function passed as an argument to another function**, which is then **invoked (called back)** inside that outer function to complete a task or handle a result.



Nested If else-



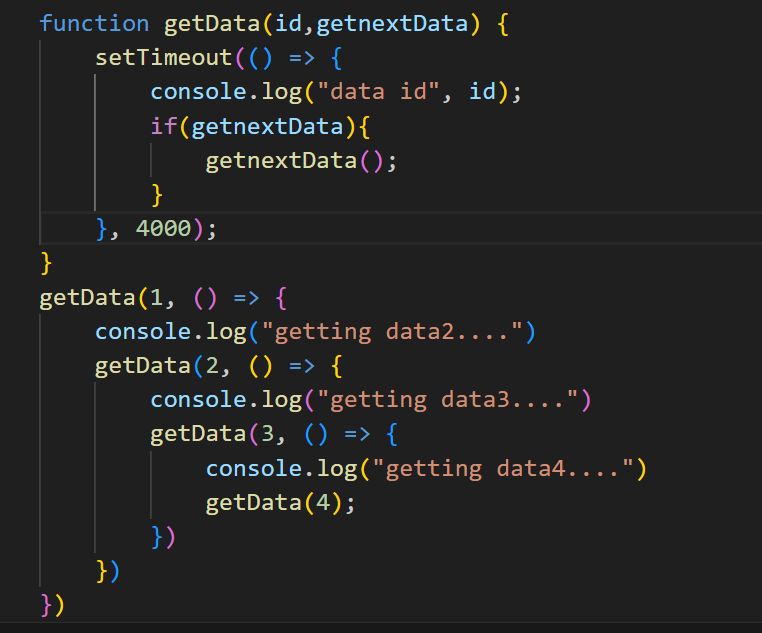
Nested For loop



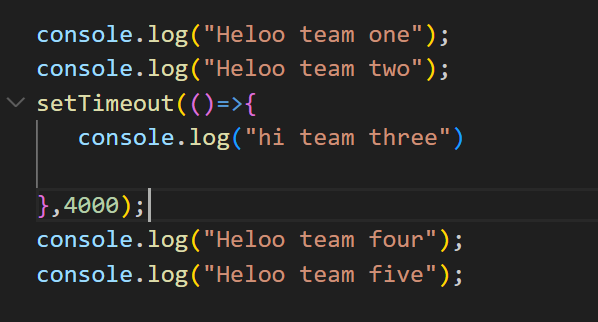
**Callback Hell in JavaScript**

**Callback Hell** (aka "Pyramid of Doom") happens when you have **many nested callbacks**, especially in asynchronous code. It makes code:

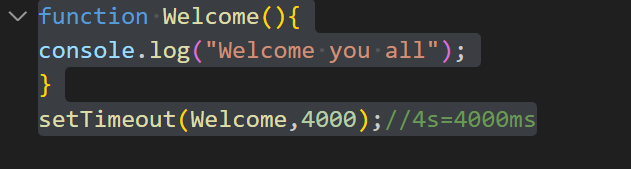
* Hard to read
* Hard to maintain
* Hard to debug



**Asynchronous operation-**



Applying timeout-



A **Promise** is a JavaScript object that represents the **eventual result** (or failure) of an asynchronous operation.

**🔹 States of a Promise:**

* **Pending**: The operation is ongoing.
* **Fulfilled**: The operation completed successfully.
* **Rejected**: The operation failed.

**What Do async and await Mean in JavaScript?**

async and await are **keywords** in JavaScript that make it easier to work with **Promises** and **asynchronous code**.

They let you write asynchronous code that **looks and behaves like synchronous code**, making it easier to read, write, and debug.

**🔹 async Keyword**

* Used to **declare a function** that will always return a **Promise**.
* Inside this function, you can use await.



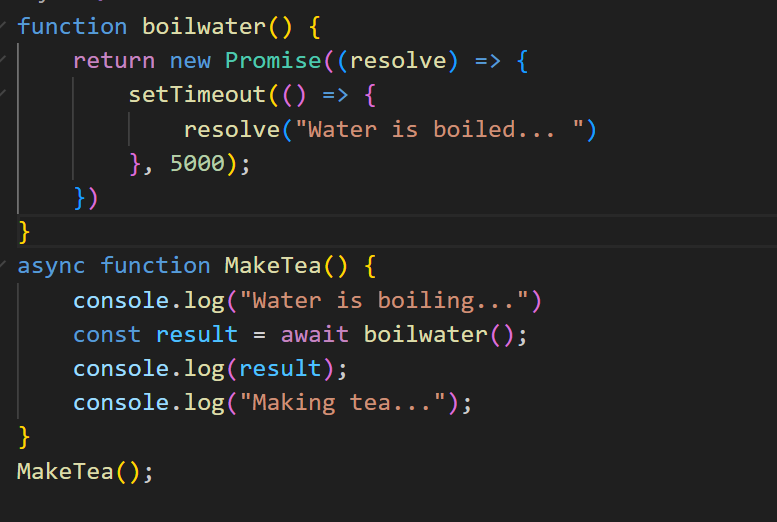
Example 2



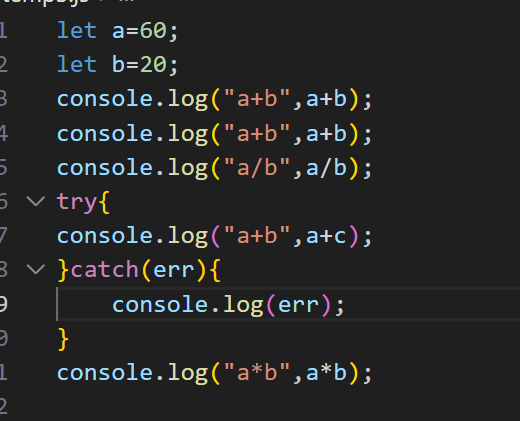
Example 3-



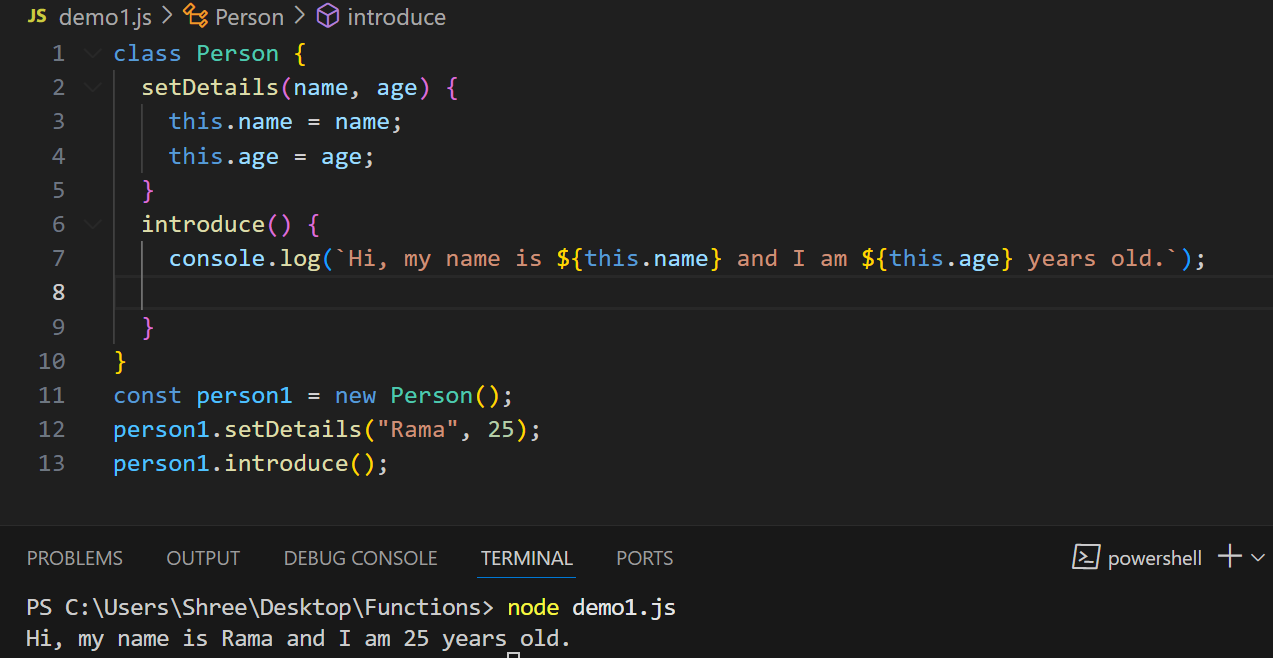
Example 4

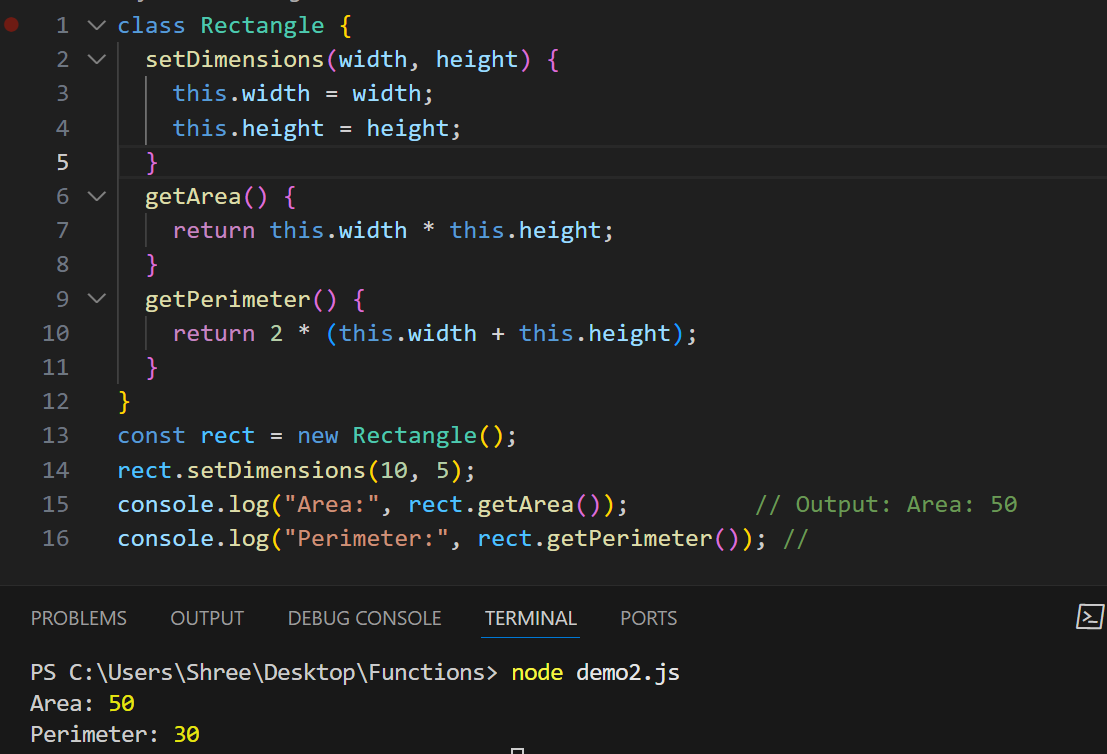


**Applying Try and catch to handle error-**

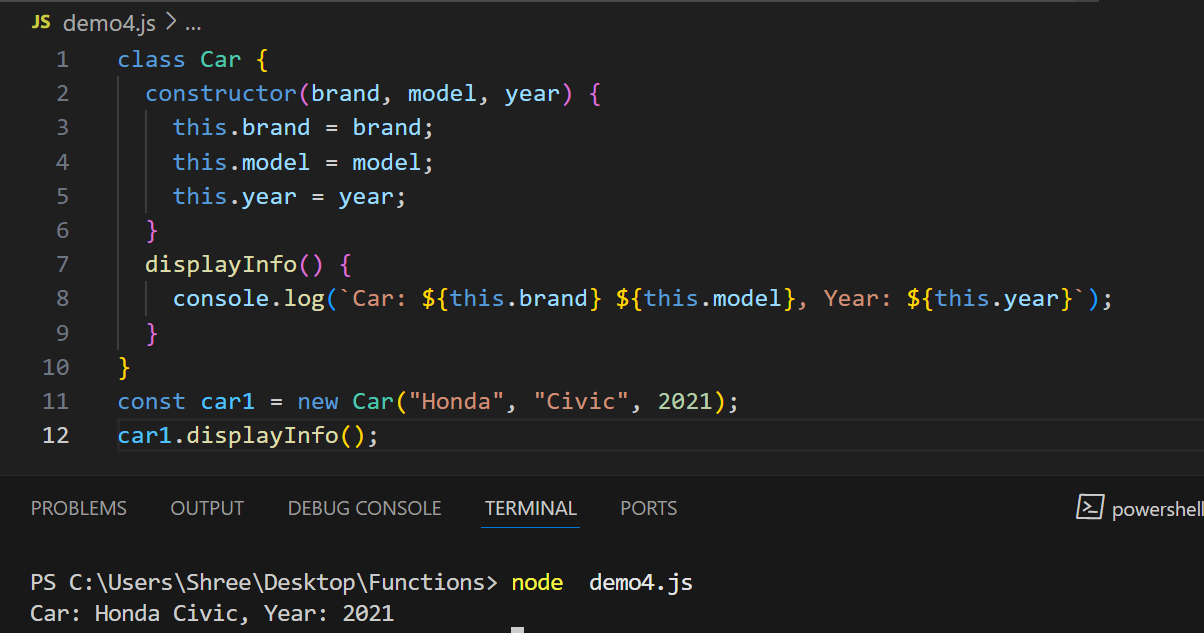
****

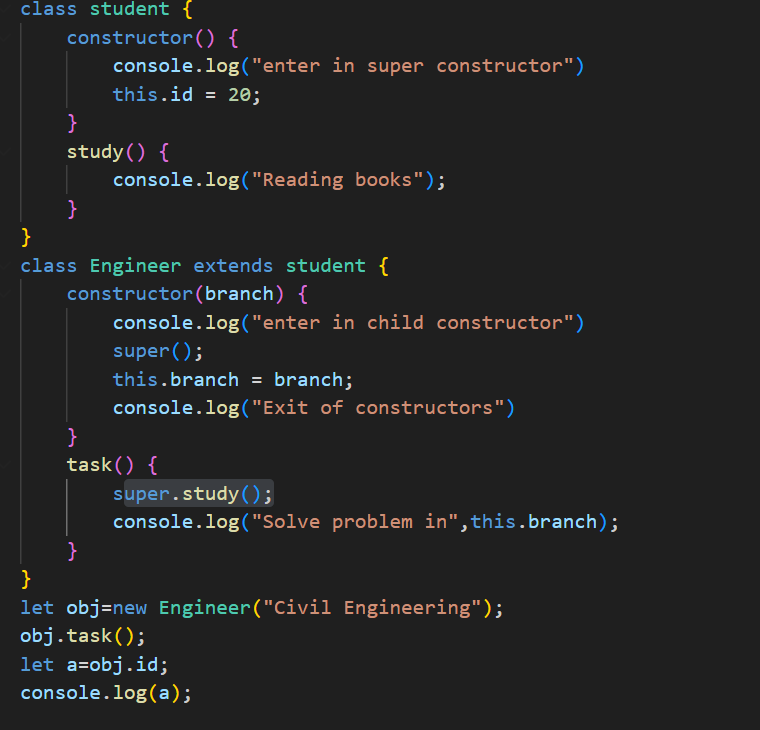
**Classes in java script**

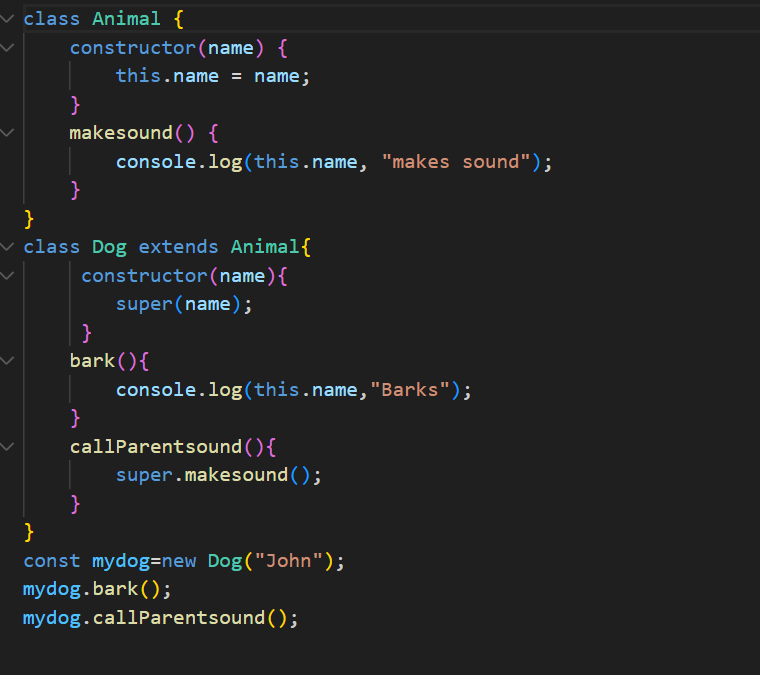
****







****

****

**Thank You**