# JavaScript Basic & DOM

Q1) What is JavaScript?

Ans: JavaScript is a high-level, interpreted programming language primarily used for client-side web development. It allows developers to add dynamic and interactive elements to websites. JavaScript code is executed by web browsers, making it an essential component for building modern web applications.

-> Client-Side Scripting: JavaScript is mainly used for client-side scripting, meaning it runs on the user's browser rather than on the server. This allows for dynamic content, interactivity, and a smoother user experience.

-> Object-Oriented: JavaScript is an object-oriented language, and it supports both prototype-based and class-based object-oriented programming.

-> Event-Driven: JavaScript is often used to respond to user actions, such as clicks, form submissions, and keyboard input. This makes it well-suited for building interactive user interfaces.

-> Cross-platform: JavaScript is supported by all major web browsers, making it a cross-platform language. This enables developers to create applications that work consistently across different browsers and operating systems.

Q2) What is the use of isNaN function?Ans: The isNaN function in JavaScript is used to determine whether a value is NaN (Not-a-Number) or not. NaN is a special value in JavaScript that represents the result of an operation that cannot produce a meaningful numeric result.

-> The isNaN function takes a single argument and returns true if the argument is NaN, and false if the argument is a valid number or can be converted to one. It is important to note that the isNaN function attempts to convert the argument to a number before checking if it is NaN. If the argument is a non-numeric value or cannot be converted to a number, isNaN will return true.

// Examples of using isNaN function

console.log(isNaN(123)); // false, 123 is a number

console.log(isNaN('Hello')); // true, 'Hello' cannot be converted to a number

console.log(isNaN('123')); // false, '123' can be converted to the number 123

console.log(isNaN(true)); // false, true is equivalent to 1 (number)

console.log(isNaN(undefined)); // true, undefined cannot be converted to a number

console.log(isNaN(null)); // false, null is equivalent to 0 (number)

console.log(isNaN(NaN)); // true, NaN is, by definition, NaN

Q3) What is negative Infinity?

Ans: In JavaScript, negative infinity is a special numeric value that represents the concept of negative infinity. It is often denoted as -Infinity. Negative infinity is the result of certain mathematical operations that approach negative infinity or when a numeric value is smaller than the smallest representable number in JavaScript.

Q4) Which company developed JavaScript?

Ans: JavaScript was developed by Netscape Communications Corporation, a technology company that played a significant role in the early development of the web. The initial development of JavaScript was led by Brendan Eich in 1995. It was originally named "Mocha" and later "LiveScript" before finally being named JavaScript. The language was designed to add interactivity and dynamic features to web pages in Netscape Navigator, one of the earliest web browsers.

-> It's important to note that while Netscape initiated the development of JavaScript, the language quickly gained popularity, and its standardization was taken over by the Ecma International organization. The standardized version is officially known as ECMAScript. JavaScript is now supported by all major web browsers, and its development and evolution involve contributions from various organizations and the open-source community.

Q5) What are undeclared and undefined variables?

Ans: Undeclared and undefined variables are two different concepts in JavaScript, but they both relate to variables in different ways.

-> Undeclared variables:

An undeclared variable is a variable that has been used in code without being formally declared using the var, let, or const keyword.

In JavaScript, you should always declare variables before using them to avoid potential issues and ensure cleaner and more maintainable code.

-> Undefined Variables:

An undefined variable is a variable that has been declared, but it has not been assigned a value. In JavaScript, when you declare a variable without assigning a value to it, the variable is automatically initialized with the value undefined.

You might also encounter undefined variables when trying to access properties of an object that do not exist.

Q6) Write the code for adding new elements dynamically?

-> We can add elements dynamically with the help of “createdElement” function.

-> in index.html:

<body>

<p>This is hardcoded Paragraph. </p>

</body>

-> In script.js :

let para = document.createElement(“p”);

para.textContent = “This is para that dynamically added by JavaScript”;

document.body.appendChild(para);

-> This dynamically adds the paragraph from JavaScript.

Q7) What is === operator?

Ans: “===” is operator that compares the two value in JavaScript.

-> It not compares the value only but, compares the data types and values

-> E.g

console.log(12 == ‘12’) //true

console.log(12 === ‘12’) //false

Q8) How can the style/class of an element be changed?

Ans: JavaScript provides us build in function like “.style” & “.classList.add()” OR “.classList.remove()” to add classes or remove classes or add styles.

-> E.g

-> To add style

document.getElementByClassName(“.name”).style.color = “red”

This upper code will turn color of that element which have class of “name” to red.

-> To add class or remove class

document.getElementByClassName(“.name”).classList.add(“open”)

document.getElementByClassName(“.name”).classList.remove (“close”)

This upper line will add class of “open” and remove class of “close” in “name” class element respectively.

Q9) What are all the looping structures in JavaScript?

Ans: JavaScript supports several looping structures that allow you to execute a block of code repeatedly. Here are the main looping structures in JavaScript:

-> for loop:

The for loop is a traditional loop that has three optional parts: initialization, condition, and iteration.

It is commonly used when the number of iterations is known in advance.

for (let i = 0; i < 5; i++) {

console.log(i);

}

-> while loop:

The while loop continues to execute a block of code while a specified condition is true.

It is suitable when the number of iterations is not known in advance.

let i = 0;

while (i < 5) {

console.log(i);

i++;

}

-> do while loop:

The do-while loop is similar to the while loop, but it guarantees that the block of code is executed at least once before checking the condition.

let i = 0;

do {

console.log(i);

i++;

} while (i < 5);

-> for in loop:

The for...in loop is used to iterate over the properties of an object.

It is suitable for iterating over the enumerable properties of an object.

const person = { name: 'John', age: 30, gender: 'male' };

for (let key in person) {

console.log(key + ': ' + person[key]);

}

-> for of loop:

The for...of loop is introduced in ECMAScript 2015 (ES6) and is used to iterate over iterable objects (arrays, strings, etc.).

It provides a more concise syntax compared to the traditional for loop.

const numbers = [1, 2, 3, 4, 5];

for (let num of numbers) {

console.log(num);

}

Q10) How can you convert the string of any base to an integer in JavaScript?

Ans: In JavaScript, you can use the parseInt function to convert a string representing a number in a specified base to an integer. The parseInt function takes two arguments: the string to be converted and the base of the numeral system. The base is an integer between 2 and 36.

Q11) What is the function of the delete operator?

Ans: The delete operator in JavaScript is used to delete a property from an object or an element at a specified index from an array. The syntax for the delete operator is as follows:

delete object.property; // Deletes a property from an object

delete array[index]; // Removes an element at the specified index from an array

Q12) What are all the types of Pop up boxes available in JavaScript?

Ans: JavaScript provides three types of pop-up boxes or dialog boxes that are commonly used for interacting with users. These are:

-> Alert Box (alert):

The alert function is used to display an alert box with a specified message and an "OK" button.

It is often used to provide information to the user or to display a warning.

-> Prompt Box (prompt)

The prompt function displays a dialog box with a message, an input field for the user to enter data, and "OK" and "Cancel" buttons.

It returns the text entered by the user or null if the user clicks "Cancel."

-> Confirm Box (confirm):

The confirm function displays a dialog box with a message and "OK" and "Cancel" buttons.

It returns a Boolean value (true if the user clicks "OK" and false if the user clicks "Cancel").

Q13) What is the use of Void (0)?

Ans: The use of void(0) in JavaScript is a common pattern that is often used to prevent the browser from navigating to a new page when clicking on a hyperlink. It's a workaround to ensure that the browser stays on the current page while executing a JavaScript function.

Here's a typical scenario where void(0) might be used:

<a href="javascript:void(0);" onclick="myFunction()">Click me</a>

In this example:

The href attribute is set to "javascript:void(0);". This effectively creates a "no-operation" JavaScript statement.

The onclick attribute is set to a JavaScript function (myFunction() in this case) that will be executed when the link is clicked.

Q14) How can a page be forced to load another page in JavaScript?

Ans: In JavaScript, you can use the window.location object to manipulate the current page's URL and force the browser to load another page. There are several ways to achieve this:

-> You can set the window.location.href property to the URL of the page you want to load. This will trigger a page navigation.

window.location.href = "https://www.example.com";

-> The window.location.replace() method is another way to force a page to load another page. It works similarly to setting window.location.href, but it replaces the current page in the browser's history, making it not possible to navigate back to the original page using the browser's back button.

window.location.replace("https://www.example.com");

Q15) What are the disadvantages of using innerHTML in JavaScript?

Ans: While the innerHTML property in JavaScript is a powerful and convenient way to manipulate the content of HTML elements, it comes with some potential disadvantages and considerations.

Here are some of the drawbacks associated with using innerHTML:

Injecting user-generated content directly into the innerHTML property can pose a security risk. If the content is not properly sanitized or validated, it may expose the application to Cross-Site Scripting (XSS) attacks.

Manipulating the innerHTML property involves parsing and rendering the HTML content, which can have performance implications, especially when dealing with large amounts of content. In some cases, using other DOM manipulation methods, such as creating and appending individual elements, may be more efficient.

## Advance JavaScript for Front-End Introduction and Code Quality

1) Write a program to Show an alert

=> alert(“This is an alert”)

2) What will be the result for these expressions?

-> 5 > 4 // true

-> "apple" > "pineapple" // false

-> "2" > "12" //true

-> undefined == null // true

-> undefined === null //false

-> null == "\n0\n" //false

-> null === +"\n0\n" //false

3) if ("0") { alert( 'Hello'); } -> Yes

4) What is the code below going to output? alert( null || 2 || undefined ); //2

## Data Types and Objects

=> Write the code, one line for each action:

-> Create an empty object user : const user = {}

-> Add the property name with the value John : user.name = “John”

-> Add the property surname with the value Smith. : user.surname = “Smith”

-> Change the value of the name to Pete : user.name = “Pete”

-> Remove the property name from the object : delete user.name

=> Is array copied?

let fruits = ["Apples", "Pear", "Orange"]; // push a new value into the "copy"

let shoppingCart = fruits; shoppingCart.push("Banana");

what's in fruits? // ["Apples", "Pear", "Orange", "Banana"]

alert( fruits.length ); // 4

=> Map to names

let john = { name: "John", age: 25 };

let pete = { name: "Pete", age: 30 };

let mary = { name: "Mary", age: 28 };

let users = [ john, pete, mary ];

let names = users.map(user => user.name)

alert( names ); // John, Pete, Mary

=> Map to objects

let john = { name: "John", surname: "Smith", id: 1 };

let pete = { name: "Pete", surname: "Hunt", id: 2 };

let mary = { name: "Mary", surname: "Key", id: 3 };

let users = [ john, pete, mary ];

let usersMapped = users.map((user) => {

return { fullName: user.name, id : user.id };

});

\*/ alert( usersMapped[0].id ) // 1 alert( usersMapped[0].fullName ) // John Smith

=> Sum the properties

There is a salaries object with arbitrary number of salaries. Write the function sumSalaries(salaries) that returns the sum of all salaries using Object.values and the for..of loop.If salaries is empty, then the result must be 0.

let salaries = { "John": 100, "Pete": 300, "Mary": 250 };

function sumSalaries(salaries) {

let sum = 0;

for (let salary of Object.values(salaries)) { sum += salary; }

return sum;

}

alert( sumSalaries(salaries) ); // 650

=> Turn the object into JSON and back Turn the user into JSON and then read it back into another variable.

let user = { name: "John Smith", age: 35};

//Object onto JSON-> JSON.parse(user)

//JSON into Object -> JSON.stringify(user)

## New Request

1) What is JSON

-> JSON is stands for JavaScript Object Notation, is a lightweight data interchange format. It is a text-based, human-readable format for representing structured data and is commonly used for transmitting data between a server and a web application, as well as for configuration files.

JSON is language-independent, which means it can be used with programming languages other than JavaScript. It is easy for both humans to read and write and for machines to parse and generate. JSON data is represented as key-value pairs, where the keys are strings and the values can be strings, numbers, objects, arrays, Booleans, or null.

Here is example of JSON File:

{

"name": "John Doe",

"age": 30,

"isStudent": false,

"hobbies": ["reading", "coding", "traveling"],

"address": {

"city": "Exampleville",

"country": "Exampleland"

}}

## Module 6) JavaScript basic & DOM

Q2) How many types of Variables in JavaScript?

Ans: There are three types of Variables in JavaScript

-> var, let, const

Q3) Define a Data Types in js?

Ans: In JavaScript, a data type defines the type of value that a variable can hold. JavaScript has several built-in data types, which can be broadly categorized into two main groups: primitive data types and non-primitive data types.

-> Primitive Data Types:

-> Number: Represents numeric values, including integers and floating-point numbers.

-> String: Represents sequences of characters enclosed in single or double quotes.

-> Boolean: Represents a logical value indicating either true or false.

-> Undefined: Represents an uninitialized variable or a function that does not return a value.

-> Null: Represents the intentional absence of any object value.

-> Symbol: Introduced in ECMAScript 6 (ES6), represents a unique identifier.

-> Non-primitive Data Types:

-> Object: Represents a collection of key-value pairs, where each key is a string (or symbol) and each value can be any data type, including other objects.

-> Array: Represents an ordered list of values, and each value can be of any data type.

-> Function: Represents a reusable block of code that can be executed by invoking the function.

-> Date: Represents a date and time.

Q4) What the deference between undefined and undeclare in JavaScript?

-> In JavaScript, "undefined" and "undeclared" refer to different concepts related to variables.

Undefined:

-> If a variable is declared but not assigned a value, its default value is undefined. If you try to access a variable that has been declared but not assigned a value, or if you try to access a property that doesn't exist in an object, the result is undefined. When a function is called but doesn't explicitly return a value, it implicitly returns undefined.

Undeclared:

-> An undeclared variable is one that has not been declared using var, let, or const. If you try to access an undeclared variable, it will result in a Reference Error. It's important to note that starting from ECMAScript 5 (ES5), using a variable without declaring it is considered bad practice, and it will throw an error in strict mode. In non-strict mode, accessing an undeclared variable creates a global variable (implicitly declares it). In modern JavaScript, it's recommended to always declare variables before using them to avoid unexpected behaviours and to catch errors early in the development process.

Q7) Check if typeof '10' is exactly equal to 10. If not make it exactly equal?Ans: if ‘10’==10 returns true, but if ‘10’===10 returns false.

Q8) Write a JavaScript Program to find the area of a triangle?

Ans: function areaTriangle(b, h){

return ½ \* (b \* h)

}

areaTriangle(10, 10) //50

Q9) Write a JavaScript program to calculate days left until next Christmas?

Ans : function daysUntilChristmas() {

const currentDate = new Date();

const currentYear = currentDate.getFullYear();

const christmasDate = new Date(currentYear, 11, 25);

if (currentDate > christmasDate) {

christmasDate.setFullYear(currentYear + 1);

}

const timeDifference = christmasDate.getTime() - currentDate.getTime();

const daysLeft = Math.ceil(timeDifference / (1000 \* 60 \* 60 \* 24));

return daysLeft;

}

const daysLeftUntilChristmas = daysUntilChristmas();

console.log(`There are ${daysLeftUntilChristmas} days left until Christmas!`);

Q10) What is Condition Statement?

Ans: let temperature = 25;

if (temperature > 30) {

console.log("It's a hot day!");

} else if (temperature >= 20) {

console.log("It's a nice day.");

} else {

console.log("It's a bit chilly.");

}

Q11) WAP to convert years into days and days into years?

Ans: function yearsToDays(years) {

const days = years \* 365.25;

return days;

}

function daysToYears(days) {

const years = days / 365.25;

return years;

}

const inputYears = 5;

const convertedDays = yearsToDays(inputYears);

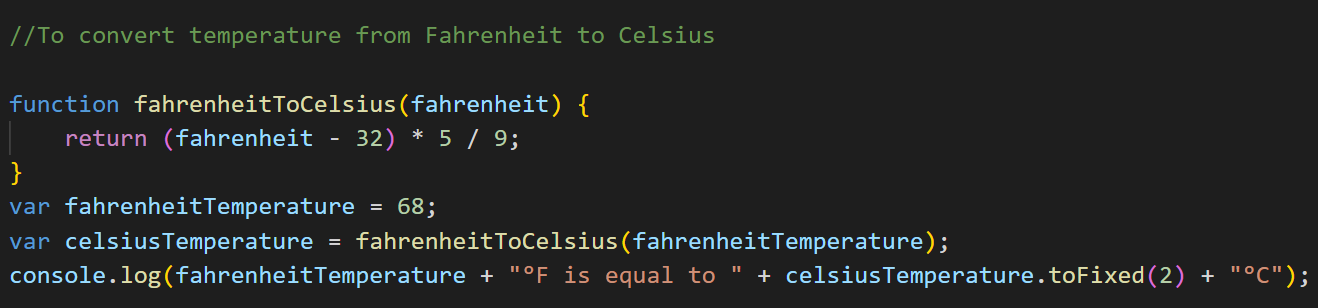
console.log(`${inputYears} years is approximately ${convertedDays} days.`);

const inputDays = 1825; // Example: 5 years worth of days

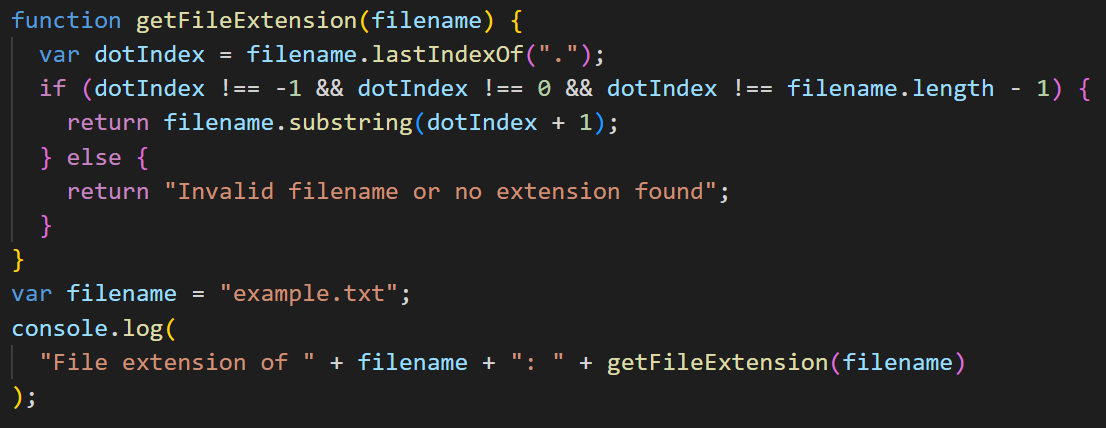
const convertedYears = daysToYears(inputDays);

console.log(`${inputDays} days is approximately ${convertedYears.toFixed(2)} years.`);

Q12) Convert temperature Fahrenheit to Celsius?

Ans: 

Q13) Write a JavaScript exercise to get the extension of a filename.?

Ans: 

Q14) What is the result of the expression (5 > 3 && 2 < 4)?

Ans: true

Q15) What is the result of the expression (true && 1 && "hello")?

Ans: “hello”

Q16) What is the result of the expression true && false || false && true?

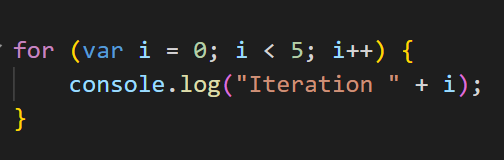
Ans: false

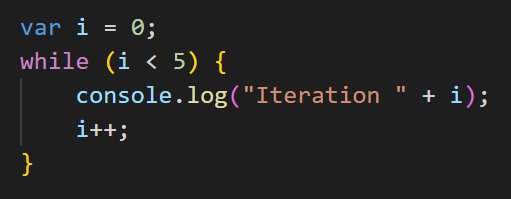
Q17) What is a Loop and Switch Case in JavaScript define that ?

Ans: In JavaScript, a loop is a programming construct that allows you to repeatedly execute a block of code as long as a specified condition is true. There are several types of loops in JavaScript, including: for loop, while loop, do while loop

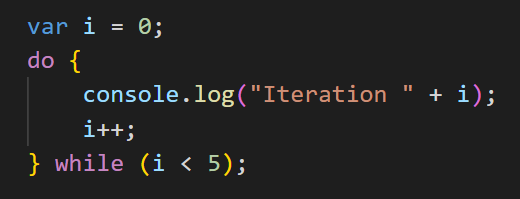
A switch case is a control flow statement used to evaluate an expression against multiple possible case values. It provides a way to execute different code blocks based on the value of an expression.

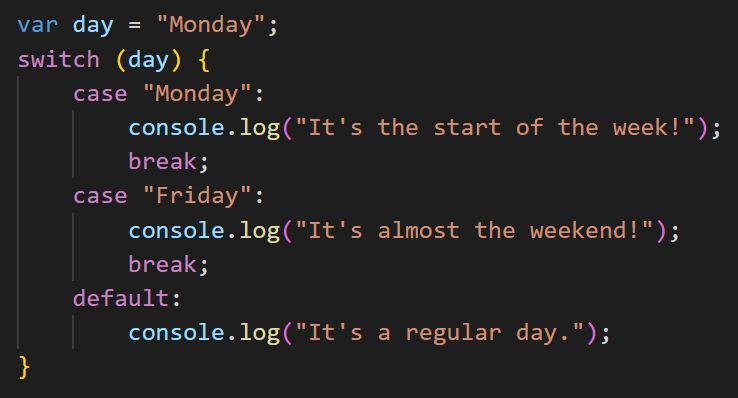
Loops:

For Loop :



While Loop :

Do While :



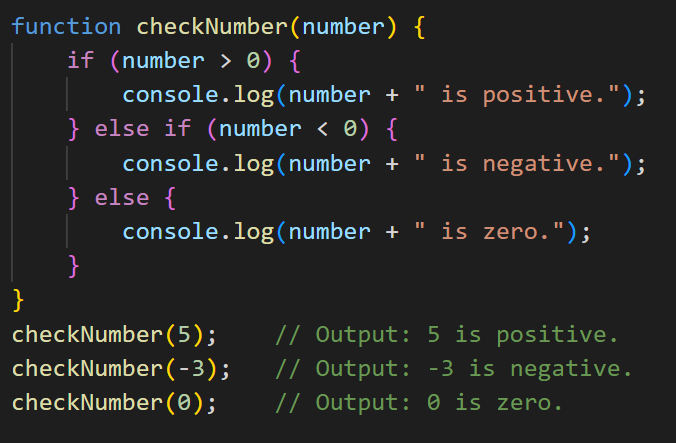
Switch :

Q18) What is the difference between && and || in JavaScript?

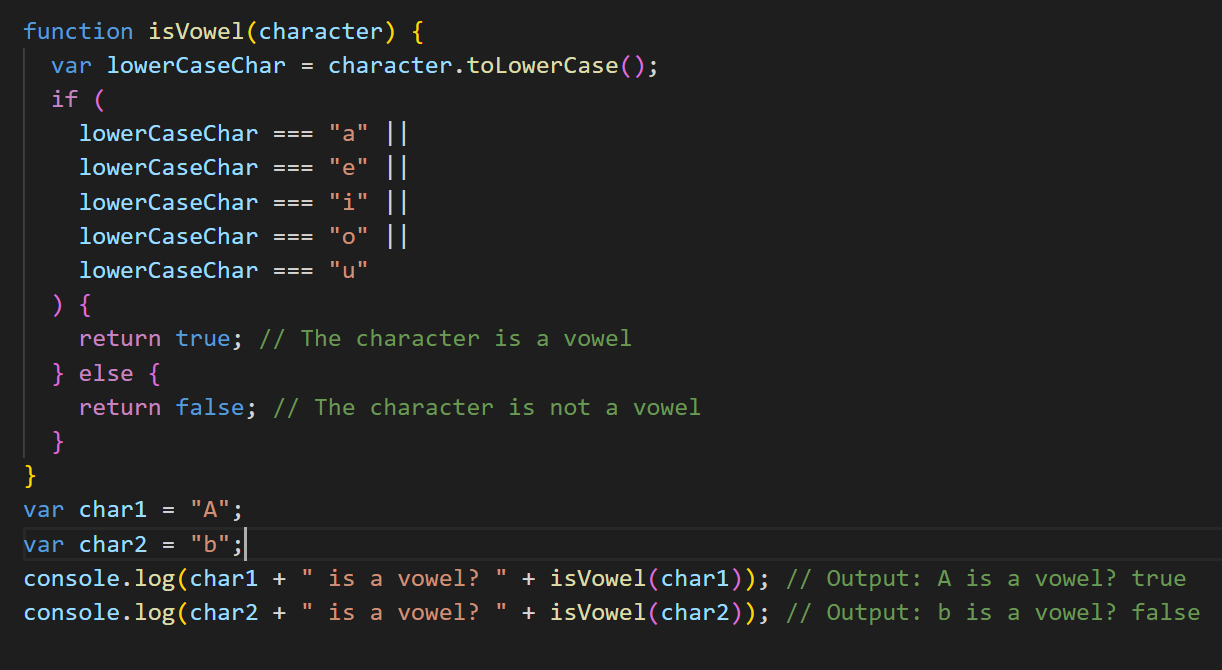
Ans: && operator is used in JavaScript to

Q19) Check Number Is Positive or Negative in JavaScript?

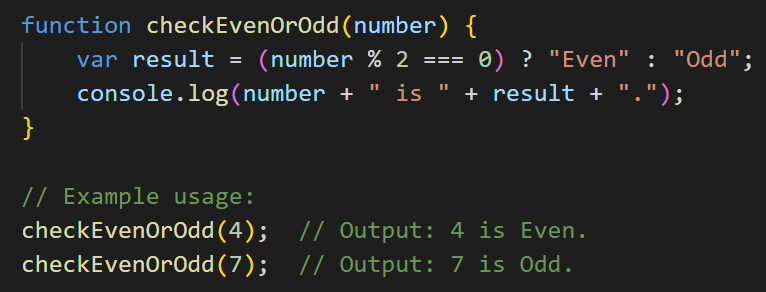
Ans:



Q20) Find the Character Is Vowel or Not ?

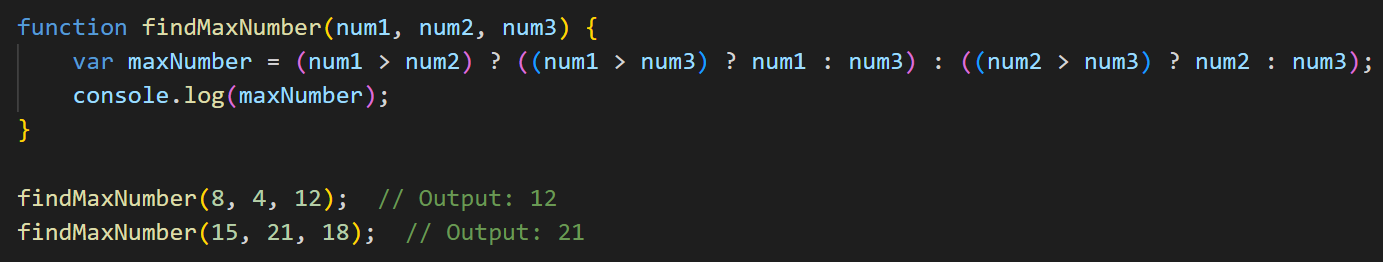
Ans: 

Q21) Write to find number is even or odd using ternary operator in JS?

Ans: 

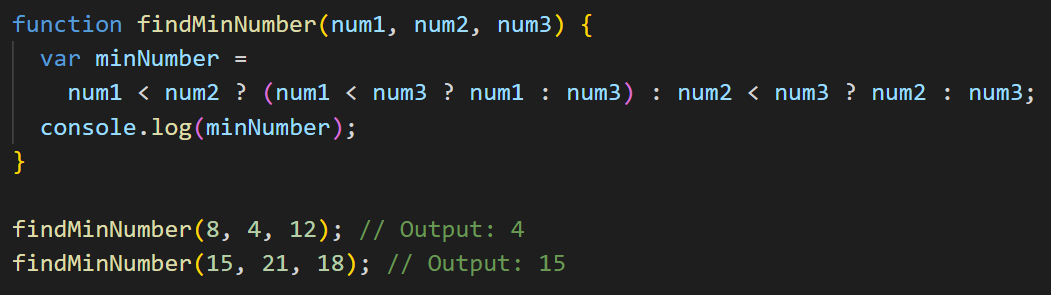
Q22) Write find maximum number among 3 numbers using ternary operator in JS?

Ans:



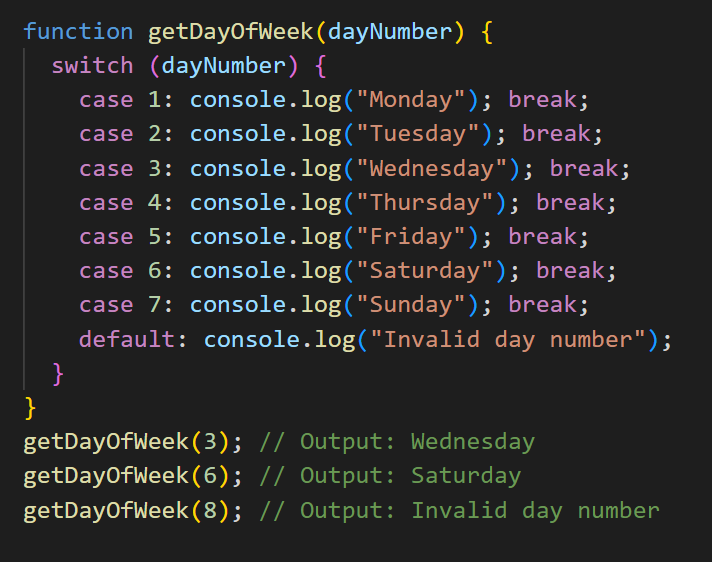
Q23) Write to find minimum number among 3 numbers using ternary operator in JS?

Ans:



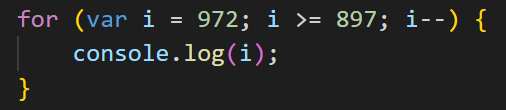
Q24) Write to show i. Monday to Sunday using switch case in JS? ii. Vowel or Consonant using switch case in JS?

Ans:



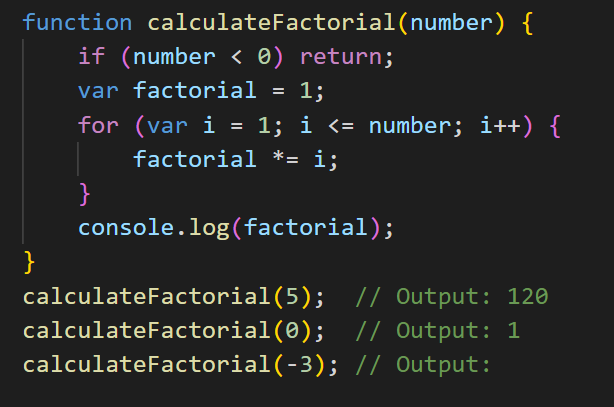
Q25) Write a print 972 to 897 using for loop in JS?

Ans:



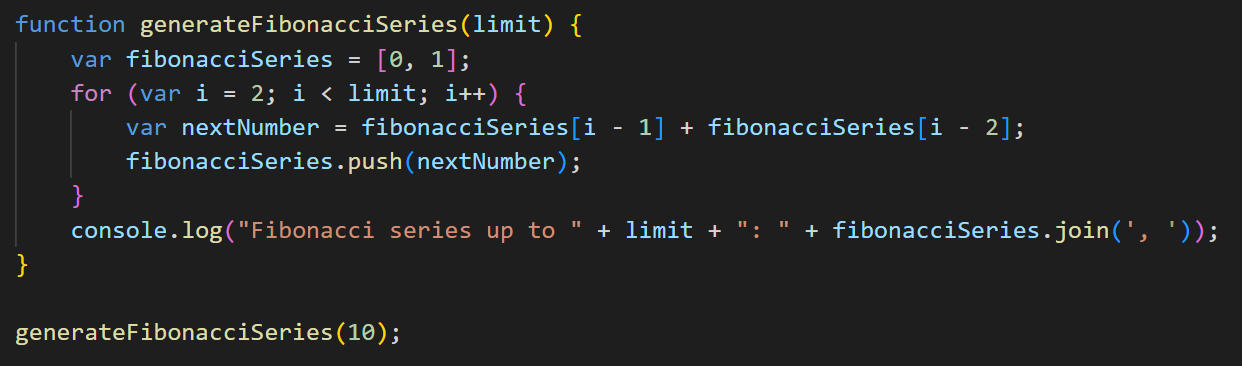
Q26) Write to print factorial of given number?

Ans:



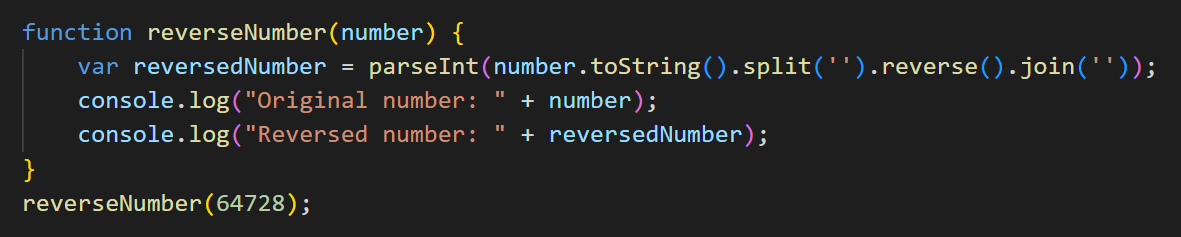
Q27) Write to print Fibonacci series up to given numbers?

Ans:



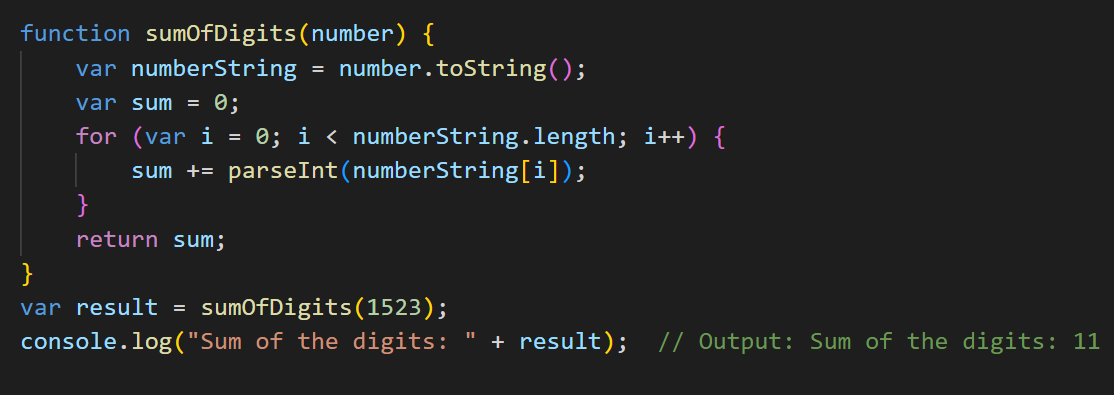
Q28) Write to print number in reverse order e.g.: number = 64728 ---> reverse =82746 in JS?

Ans:



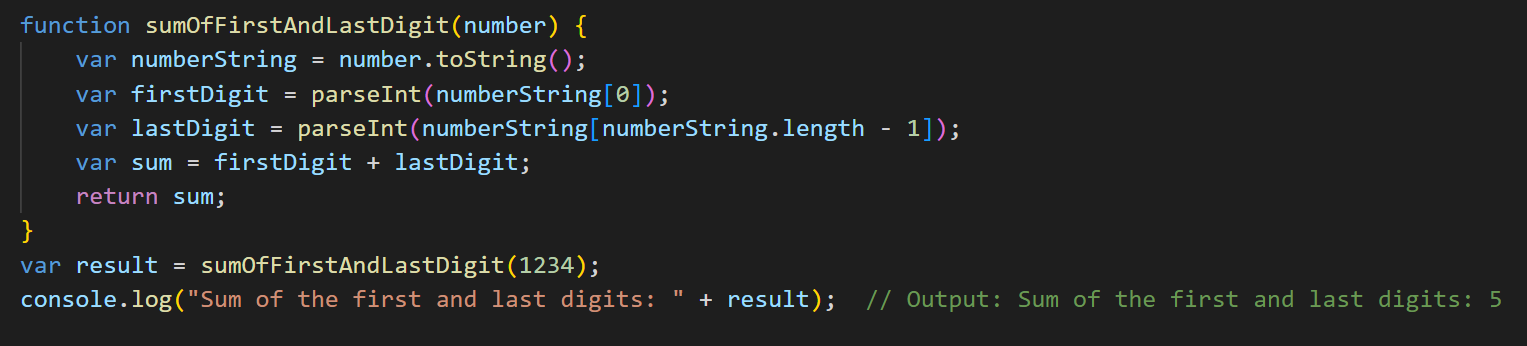
Q29) Write a program make a summation of given number (E.g., 1523 Ans: - 11) in JS?

Ans:



Q30) Write a program you have to make a summation of first and last Digit. (E.g., 1234 Ans: - 5) in JS?

Ans:



Q31) Use console.log() and escape characters to print the following pattern in JS?

1 1 1 1 1

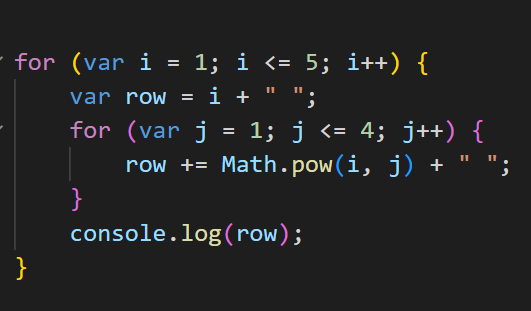
2 1 2 4 8

3 1 3 9 27

4 1 4 16 64

5 1 5 25 125

Ans:



Q32) Use pattern in console.log in JS?

1

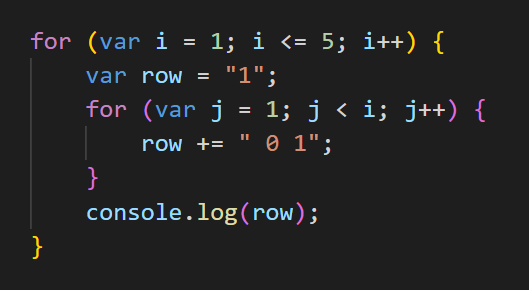
1 0

1 0 1

1 0 1 0

1 0 1 0 1

Ans:



Q33)

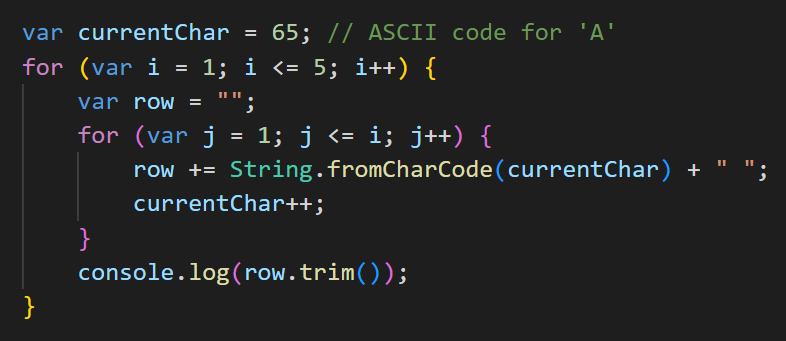
A

B C

D E F

G H I J

K L M N O

Ans: 

Q34)

1

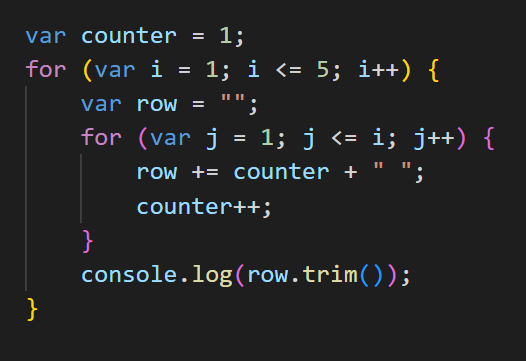
2 3

4 5 6

7 8 9 10

11 12 13 14

Ans:



Q35)

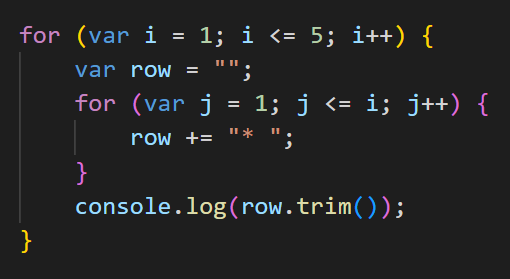
\*

\*\*

\*\*\*

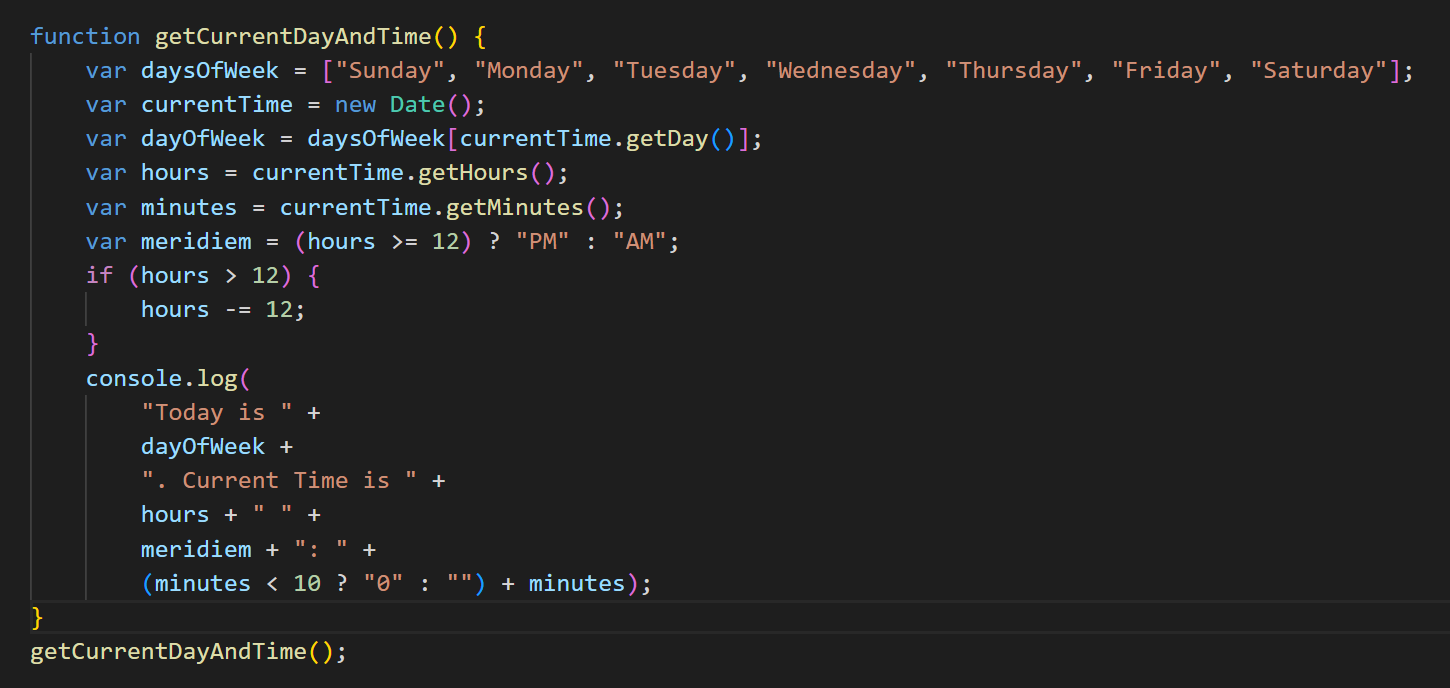
\*\*\*\*

Ans:



Q36) Write a JavaScript Program to display the current day and time in the following format. Sample Output: Today is Friday. Current Time is 12 PM: 12 : 22

Ans:



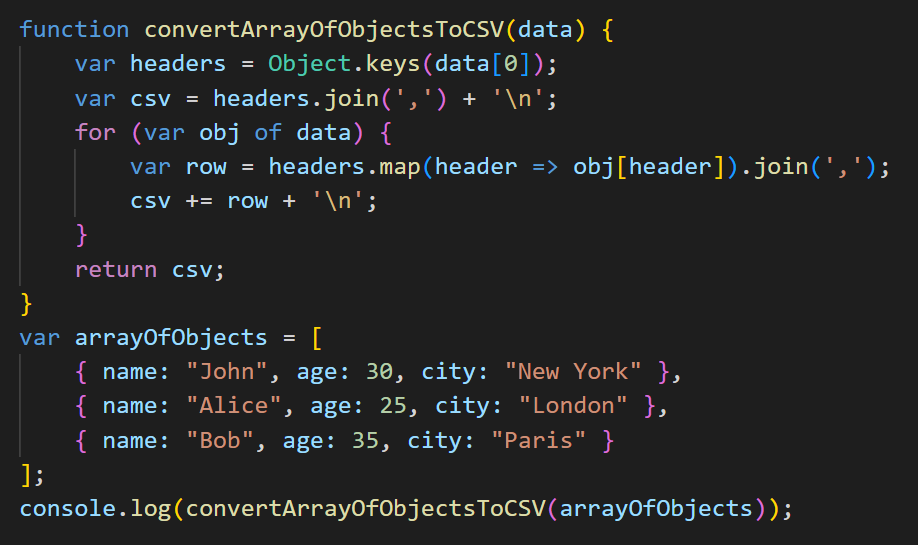
Q37) Write a JavaScript program to get the current date?

Ans:



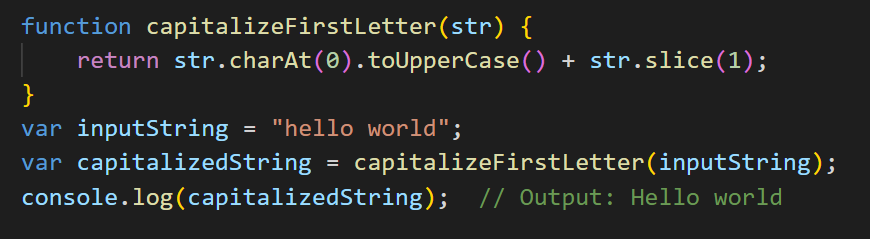
Q38) Write a JavaScript program to convert an array of objects into CSV string?

Ans:



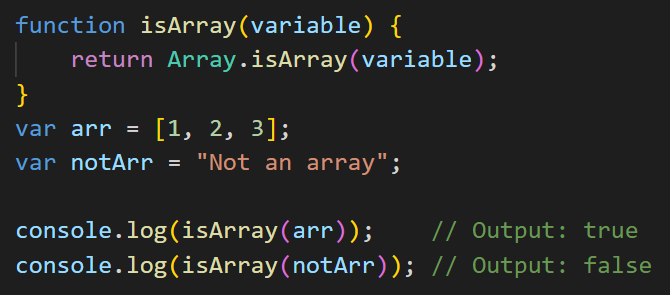
Q39) Write a JavaScript program to capitalize first letter of a string?

Ans:



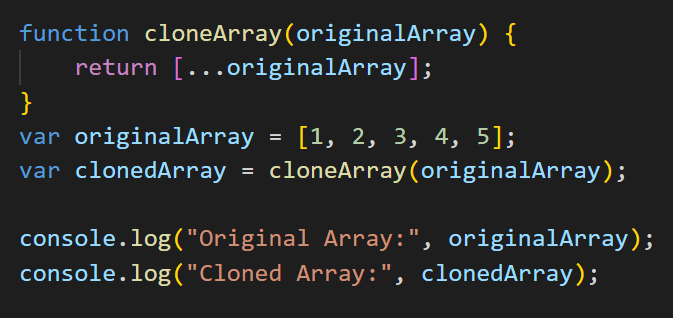
Q40) Write a JavaScript program to determine if a variable is array?

Ans:



Q41) Write a JavaScript program to clone an array?

Ans:



Q42) What is the drawback of declaring methods directly in JavaScript objects?

Ans: While declaring methods directly in JavaScript objects is a common practice, there are a few drawbacks or considerations to keep in mind:

Code Duplication:

If you have multiple instances of an object and each instance has its own method, it can lead to code duplication. In such cases, it might be more efficient to define methods on the prototype to share the same method across all instances.

Memory Consumption:

Each object instance carrying its own methods can consume more memory compared to using prototypes. Prototypes allow methods to be shared among all instances, leading to more memory-efficient code.

Limited Flexibility:

When methods are declared directly within an object, it may become more challenging to modify or extend those methods later. If you decide to add, remove, or modify a method, you would need to do it for each instance of the object.

Difficulty in Inheritance:

If you plan to create a hierarchy of objects and leverage inheritance, defining methods directly might make it harder to set up a clear and effective inheritance structure. Using prototypes can facilitate a more organized and extendable inheritance model.

Maintenance Challenges:

As the project grows, maintaining and updating code with methods directly declared in objects might become challenging. Using more modular and organized structures, such as classes and prototypes, can enhance code maintainability.

Q43) Print the length of the string on the browser console using console.log()?

Ans: let string = “Hello World”

console.log(string.length())

Q44) Change all the string characters to capital letters using toUpperCase() method?

Ans: let string = “This is string”

console.log(string.toUpperCase())

Q45) What is Bom vs Dom in JS?

Ans:

DOM (Document Object Model):

The DOM is a programming interface for web documents.

It represents the structure of a document as a tree of objects, where each object corresponds to a part of the document, such as elements, attributes, and text.

The DOM provides a way for programs to manipulate the structure, style, and content of documents dynamically.

-> It allows JavaScript to interact with and manipulate the content of an HTML or XML document.

The DOM is platform-independent and language-independent, making it accessible from various programming languages.

DOM manipulation is often used for tasks like updating the content of a webpage, handling user interactions, and dynamically changing the structure of a document.

BOM (Browser Object Model):

The BOM is a set of JavaScript objects provided by the browser, representing various components of the browser itself.

Unlike the DOM, the BOM is not standardized, and its features can vary between browsers.

Commonly used BOM objects include window, document, navigator, location, history, screen, etc.

-> BOM provides JavaScript access to browser-related functionalities such as manipulating the browser window, controlling navigation, handling history, and managing the client's screen.

It is not standardized by the W3C, and features may differ between browsers.

Q46) Array vs object deference in JS?

Ans: Array:

Ordering:

Arrays are ordered collections of values, and each value is associated with an index (starting from 0).

The order of elements is significant, and you can access elements using their index.

Length Property:

Arrays have a length property, which automatically updates based on the number of elements in the array.

Methods:

Arrays come with built-in methods for common operations, such as push, pop, shift, unshift, splice, and more.

Use Case:

Arrays are commonly used when the order of elements is essential, and you need to perform operations like sorting, filtering, or iterating over elements.

-> Object:

Key-Value Pairs:

Objects are collections of key-value pairs, where each value is associated with a unique key.

The order of key-value pairs is not guaranteed in all JavaScript engines.

Dynamic Keys

Keys in objects can be dynamic, and you can use variables or expressions as keys.

Use Case:

Objects are used when the data is more appropriately represented as a set of named properties, and the order is not significant.

-> Use Cases:

Array:

Use arrays when the order of elements matters, and you need to perform operations like sorting, filtering, or iterating over elements in a specific order.

Object:

Use objects when you need to represent data with named properties and their values. Objects are especially useful when dealing with entities with distinct attributes.

Q47) Split the string into an array using split() Method?

Ans: let string = “Hello World”

const array = string.split(“”)

console.log(array) //Output : [“H”, “e”, “l”, “l”, “o”, “ ”, “W”, “o”, “r”, “l”, “d”].

Q48) Check if the string contains a word Script using includes() method?

Ans: let string = “This is Javascript Assignment”

Let isIncludes = string.includes(“Assignment”)

console.log(isIncludes) //true

Q49) Change all the string characters to lowercase letters using toLowerCase() Method.

Ans: let string = “THIS IS SOME TEXT”

console.log(string.toLowerCase()) //Output : “this is some text”

Q50) What is Character at index 15 in ’30 Days of JavaScript’ string? Use charAt() method.

Ans: let string = “30 Days of JavaScript’ string”

console.log(string.charAt(15)) //Outputs : “J”

Q51) copy to one string to another string in JS?

Ans: let string1 = “This is text 1”

Let string2 = string1

console.log(string2) // Output : “This is text 1”