***Module (JAVASCRIPT BASIC & DOM)***

Q. What is JavaScript. How to use it?

Ans. The primary application of JavaScript, a dynamic, high-level programming language, is the creation of interactive and dynamic web pages. Along with HTML (Hypertext Markup Language) and CSS (Cascading Style Sheets), it is a crucial part of web development. With JavaScript, developers may customize features, work with the Document Object Model (DOM), and react to user input on webpages.

* browser: Using <script> elements, JavaScript code can be included directly into HTML documents. Either put the script at the end of the <body> section or in the <head> section.
* A common application of JavaScript is to improve HTML and CSS interactions. For instance, executing asynchronous requests, modifying the DOM, or reacting to user events.
* External JavaScript File: Use the <script> tag in your HTML file to include an external JavaScript file that you create and save with the '.js' extension.

Q. **How many types of Variables in JavaScript?**

* Based on how they are declared, variables in JavaScript can be classified as either var, let, or const. Each has unique qualities and applications.
* var = Variables in JavaScript were first declared using the var keyword. It lacks certain let and const capabilities and has function scoping rather than block scoping. Variables designated with var are elevated to the highest level of their scope.
* let: ECMAScript 6 (ES6) introduced let, which offers block-scoping. Declaring variables that are exclusive to the block, statement, or expression in which they are utilized is made possible by this. Let-declared variables aren't pushed to the edge of their capabilities.
* Const: Const is used to declare constants and was also introduced in ES6. Once a const variable is assigned, its value cannot be changed. Const is block-scoped, just like let.

Q. Define a Data Types in js?

**Ans**. In JavaScript, there are several types of variables based on the kind of data they can store. The fundamental data types in JavaScript are divided into two categories: primitive data types and object reference types.

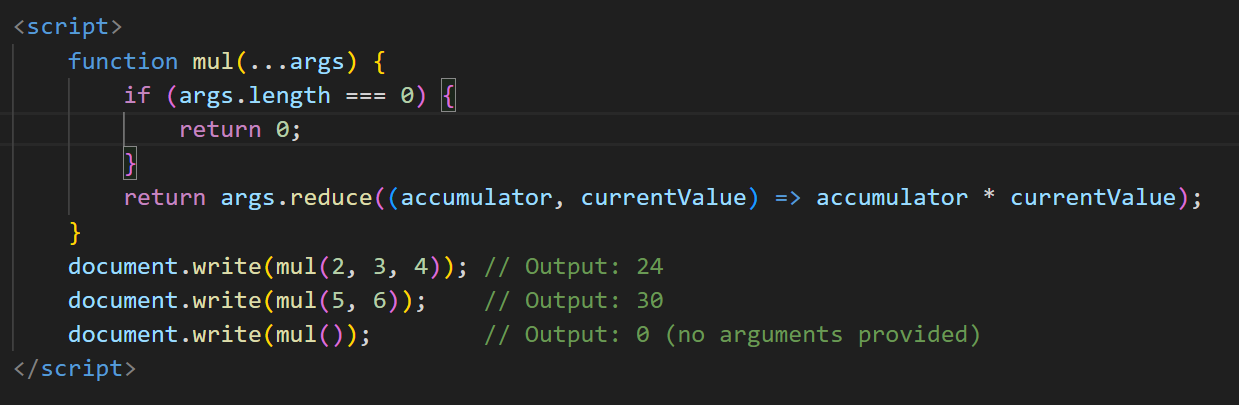
1. Primitive Data Types:

* Number: Shows values in numbers. It contains both floating-point and integer numbers.
* String: indicates character sequences (text).
* Boolean: indicates true or false logical values.
* Defines a variable that has not been initialized.
* Null: indicates the deliberate removal of all object values.
* Symbol (ES6 and later): Indicates a distinct identity.

2. Object Reference Types:

* Object: A grouping of properties and methods, or key-value pairs.
* An array is a list of values in sequence.
* A reusable code block with a defined purpose is called a function.
* Date: Indicates a precise moment in time.
* Additional Object Reference Types: RegExp, which is used for regular expressions, and other browser-specific objects are among the additional built-in object types in JavaScript.

Q. Write a mul Function Which will Work Properly When invoked With Following Syntax.



* This mul function uses the rest parameter (...args) to accept any number of arguments. It then uses the reduce method to multiply all the values together. If no arguments are provided, it returns 0.

Q. What the deference between undefined and undeclare in JavaScript?

* Undefined Variable: If you declare a variable but don't assign a value to it, or if you try to access the value of a variable that has not been assigned any value, its value will be undefined.

let x; // variable declared but not assigned a value

console.log(x); // Output: undefined

* Undeclared Variable: If you try to access the value of a variable that has not been declared using var, let, or const, you'll encounter a ReferenceError. This is referred to as an "undeclared variable."

console.log(y); // y is not defined

Q. Using console.log() print out the following statement: The quote 'There is no exercise better for the heart than reaching down and lifting people up.' by John Holmes teaches us to help one another. Using console.log() print out the following quote by Mother Teresa:

**Ans**. The quote by John Holmes

* console.log("The quote 'There is no exercise better for the heart than reaching down and lifting people up.' by John Holmes teaches us to help one another.");
* The quote by Mother Teresa
* console.log("Peace begins with a smile. - Mother Teresa");

Q. Check if typeof '10' is exactly equal to 10. If not make it exactly equal?

* In JavaScript, the typeof operator returns a string indicating the type of the operand. If you use typeof '10', it will return the string "string". To check if it's exactly equal to the number 10, you can convert the string to a number and then perform the comparison. Here's how you can do it:
* In this case, the string "10" is changed to the number 10 by using parseInt(stringValue, 10). The next if statement determines whether the outcome is a number and whether it is precisely equal to 10. It updates the value to 10 if it is not.

Q. Write a JavaScript Program to find the area of a triangle?

**Ans**. Certainly! The formula to find the area of a triangle is given by:

Area=12×base×heightArea=21​×base×height

<script>

        function calculateTriangleArea(base, height) {

            if (base > 0 && height > 0) {

                let area = 0.5 \* base \* height;

                return area;

            } else {

                return "Base and height must be positive numbers.";

            }

        }

        let baseLength = 8;

        let triangleHeight = 5;

        let result = calculateTriangleArea(baseLength, triangleHeight);

        if (typeof result === 'number') {

            document.write(`The area of the triangle ${baseLength

            } and height ${triangleHeight} is: ${result}`);

        } else {

            document.write(result);

        }

    </script>

Q. Write a JavaScript program to calculate days left until next Christmas?

<script>

        function daysUntilChristmas() {

            let currentDate = new Date();

            let currentYear = currentDate.getFullYear();

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

            if (currentDate > christmasDate) {

                christmasDate.setFullYear(currentYear + 1);

            }

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

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

            return daysLeft;

        }

        let daysUntil = daysUntilChristmas();

        document.write(`There are ${daysUntil} days left until next Christmas.`);

    </script>

Q. What is Condition Statement?

**Ans**. In programming, a conditional statement is a construct that determines whether to execute a code block differently depending on whether a certain condition evaluates to true or false. It gives a software the ability to decide what to do and how to do it based on the circumstances. In the majority of programming languages, conditional statements come in the following fundamental forms:

* If = The most fundamental kind of conditional statement is the if statement. It analyzes a given condition and then runs a block of code if the condition is true.
* if-else Statement: This phrase expands on the if statement. If the condition is true, one block of code is executed; otherwise, another block.
* Switch Statement: When a variable can have more than one possible value, the switch statement is used. It offers a mechanism to compare a variable with several values and run several code blocks in response.

Syn = let age = 18;

if (age >= 18) {

document.write("You are an adult.");

} else {

document.write ("You are a minor.");

}

Syn = let day = "Monday";

switch (day) {

case "Monday":

document.write ("It's the start of the week.");

break;

case "Friday":

document.write ("It's almost the weekend!");

break;

default:

document.write It's a regular day.");

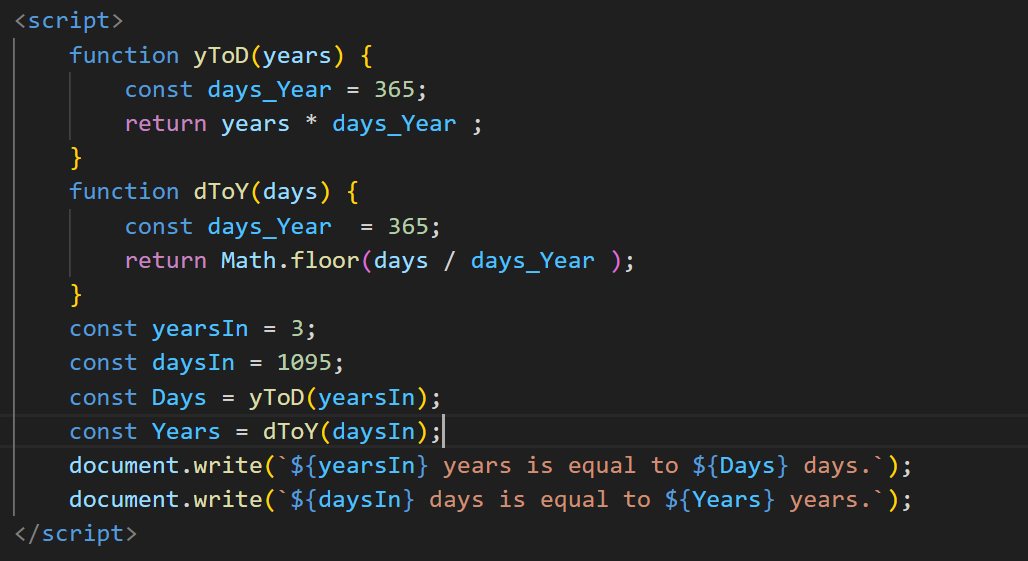
}

Q. Find circumference of Rectangle formula: C = 4 \* a ?

**Ans**. It appears that the formula you have given might include an error. Usually, the total of the four sides is used to determine a rectangle's circumference, or perimeter. The circumference (C) of a rectangle with sides of length a and b can be found using the following formula:

C=2a+2b

Q. WAP to convert years into days and days into years?



Q. Convert temperature Fahrenheit to Celsius? (Conditional logic Question)

<script>

        function convertFToC(fahrenheit) {

            if (typeof fahrenheit === 'number' && !isNaN(fahrenheit)) {

                let celsius = (5 / 9) \* (fahrenheit - 32);

                return celsius;

            } else {

                return "Please provide a valid temperature in Fahrenheit.";

            }

        }

        let fahrenheitTemperature = 32;

        let celsiusTemperature = convertFToC(fahrenheitTemperature);

        document.write(`${fahrenheitTemperature} degrees Fahrenheit is

         ${celsiusTemperature} degrees Celsius.`);

    </script>

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

 <script>

        function getFileExtension(filename) {

            let dotIndex = filename.lastIndexOf('.');

            if (dotIndex !== -1 && dotIndex > 0) {

                let extension = filename.substring(dotIndex + 1);

                return extension;

            } else {

                return "Invalid filename. No file extension found.";

            }

        }

        let filename1 = "document.txt";

        document.write(`The extension of ${filename1} is: ${getFileExtension(filename1)}`);

    </script>

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

* The expression (5 > 3 && 2 < 4) is a logical AND operation in JavaScript. It evaluates to true because both conditions on either side of the && operator are true.
* 5 > 3: This condition is true because 5 is greater than 3.
* 2 < 4: This condition is also true because 2 is less than 4.
* The logical AND (&&) operator requires both conditions to be true for the entire expression to be true. Therefore, (5 > 3 && 2 < 4) evaluates to true.

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

**Ans**. The expression (true && 1 && "hello") is a logical AND operation in JavaScript. It evaluates to the last truthy value in the expression.

In JavaScript, the logical AND (&&) operator short-circuits. It evaluates the expressions from left to right and stops as soon as it encounters a falsy value. If all values are truthy, it returns the last truthy value.

true: This is truthy.

1: This is also truthy.

"hello": This is the last value and is truthy.

Therefore, the entire expression evaluates to the last truthy value, which is "hello". So, the result of (true && 1 && "hello") is the string "hello".

Q. What is the result of the expression true && false || false && true?

* The JavaScript logical AND (&&) and OR (||) operators are used in the statement true && false || false && true. Compared to the logical OR operator, the logical AND operator is more precedent.
* true && false: This is not true since both operands must be true for the logical AND to hold.
* false && true: For the same reason, this is likewise untrue.
* Now, for false || false, if at least one operand is true, the logical OR operator returns true. Since both operands in this instance are false, the outcome is also false.
* As a result, false is the outcome of the full expression true && false || false && true.

Q. What is a Loop and Switch Case in JavaScript define that?

* for Loop:

The for loop is used to iterate over a sequence of values, executing a block of code for each iteration. It has three parts: initialization, condition, and increment/decrement.

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

// Code to be executed during each iteration

document.write (i);

}

* while Loop:

The while loop executes a block of code as long as a specified condition is true.

let i = 0;

while (i < 5) {

// Code to be executed during each iteration

document.write (i);

i++;

}

* Switch Case in JavaScript:

The switch statement in JavaScript is a control structure used to evaluate an expression against multiple possible case values. It provides an alternative to a series of if-else statements.

let day = "Monday";

switch (day) {

case "Monday":

document.write ("It's the start of the week.");

break;

case "Friday":

document.write ("It's almost the weekend!");

break;

default:

document.write ("It's a regular day.");

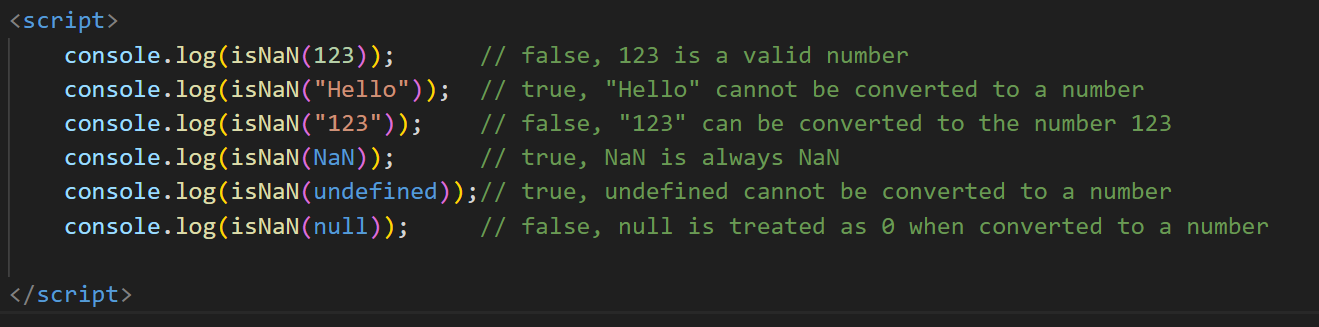
}

In this example, the value of day is compared against different cases. If a match is found, the corresponding block of code is executed. The break statement is used to exit the switch statement after a match.

Q. What is the use of is Nan function?

Ans. To find out if a value is NaN (Not-a-Number) or not, use JavaScript's isNaN function. A non-numeric string being attempted to be converted to a number, or dividing zero by zero, are examples of mathematical operations that result in NaN, a peculiar value.

* With a single parameter, the isNaN function returns true if the argument is NaN and false otherwise. Additionally, it attempts to transform the argument to a number before determining whether it is NaN, which is known as type coercion. It is crucial to comprehend how isNaN functions because this can result in unexpected behavior.



Q. What is the difference between && and || in JavaScript?

Ans. “&& “ If both operands are true, the "&&" operator returns "true," else it returns "false."

The operator is short-circuiting. The second operand is not examined if the first operand is false since the overall result will still be false regardless of the second operand.

true && true; // true

true && false; // false

false && true; // false

false && false; // false

|| = The operators "||" yield "true" when at least one operand is "true," and "false" if both operands are false.

It is a short-circuiting operator, just like &&. The second operand is not assessed if the first operand is true since the overall result will always be true, regardless of the second operand.

true || true; // true

true || false; // true

false || true; // true

false || false; // false

Q. What is the use of Void (0)?

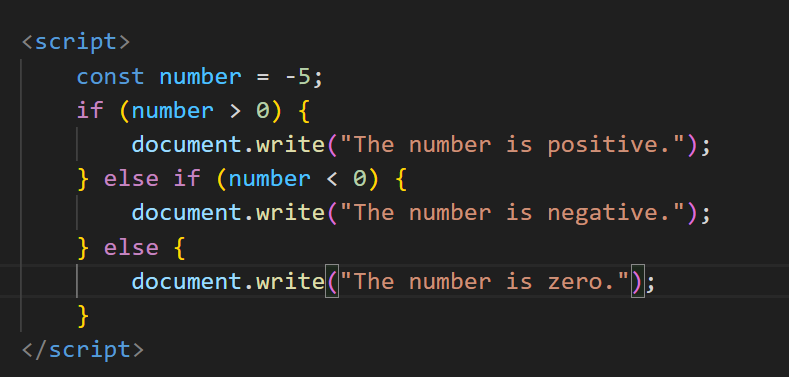
* In JavaScript, void(0) is often used to create a "void" or "undefined" value and is commonly seen in the context of anchor (<a>) elements in HTML. It is used to prevent the default action of a hyperlink while still allowing the execution of JavaScript code when the link is clicked.
* <a href="javascript:void(0);" onclick="myFunction()">Click me</a>
* The value of the href attribute is javascript:void(0);.
* There is a JavaScript function (myFunction in this case) assigned to the onclick attribute.
* Using this technique, you can make a clickable element (such as a button or link) that, when clicked, opens on an other page. The void(0) expression prevents the browser from going to a new page by returning undefined and acting as a placeholder for a URL in the href property.
* It's important to note that javascript:void(0) is no longer widely used in modern web development. Instead, developers frequently handle activities without relying on the href attribute by using event listeners in JavaScript or by using # as a placeholder in the href attribute.

Q. How can a page be forced to load another page in JavaScript?

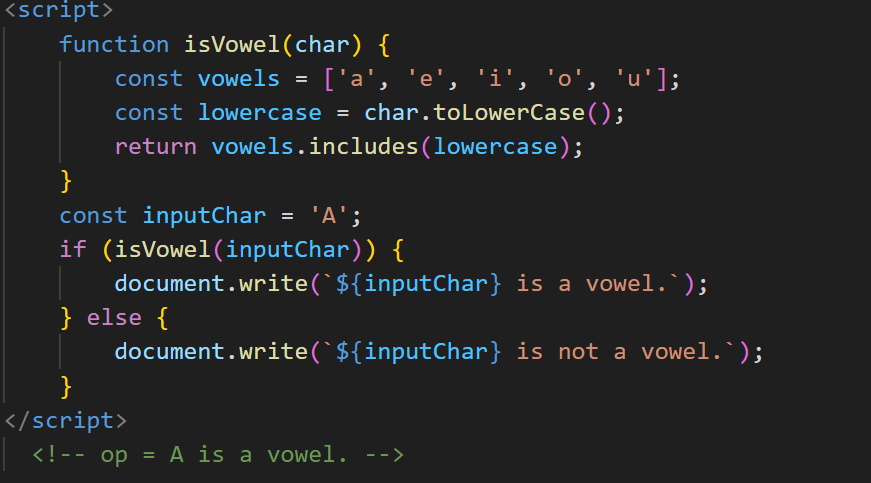
* In JavaScript, you can use the window.location object to navigate to another page. The window.location object represents the current URL of the browser. You can change its href property to a new URL to force the page to load another page.
* window.location.href = "https://www.example.com";
* This line of code sets the href property of window.location to a new URL, which effectively forces the browser to load the specified page.

Q. Check Number Is Positive or Negative in JavaScript?

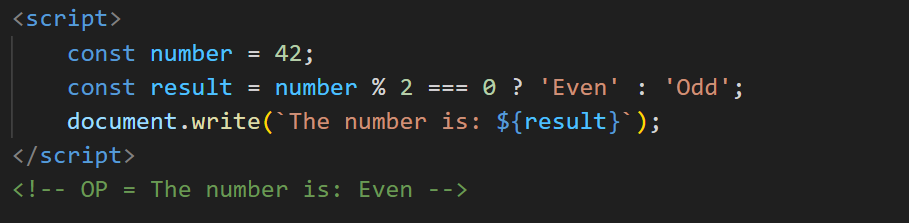
Ans . A basic conditional statement (an if statement) in JavaScript can be used to determine if a given integer is positive or negative. This function serves as an example of how to check a number's sign.



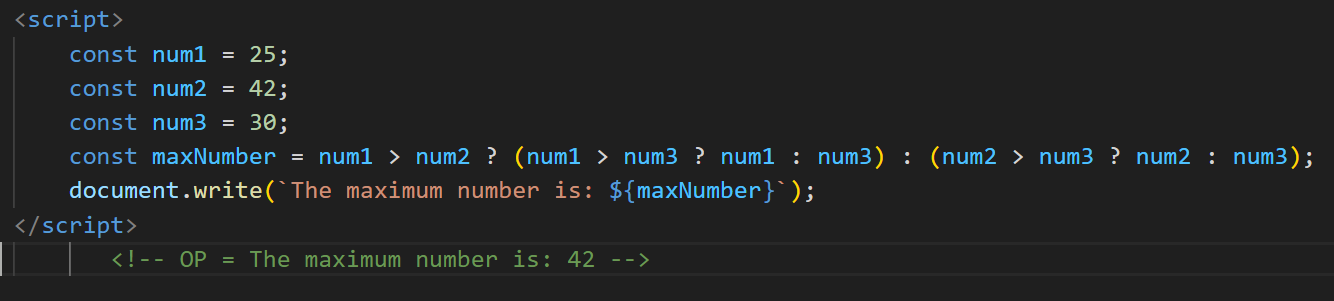
Q. Find the Character Is Vowel or Not ?



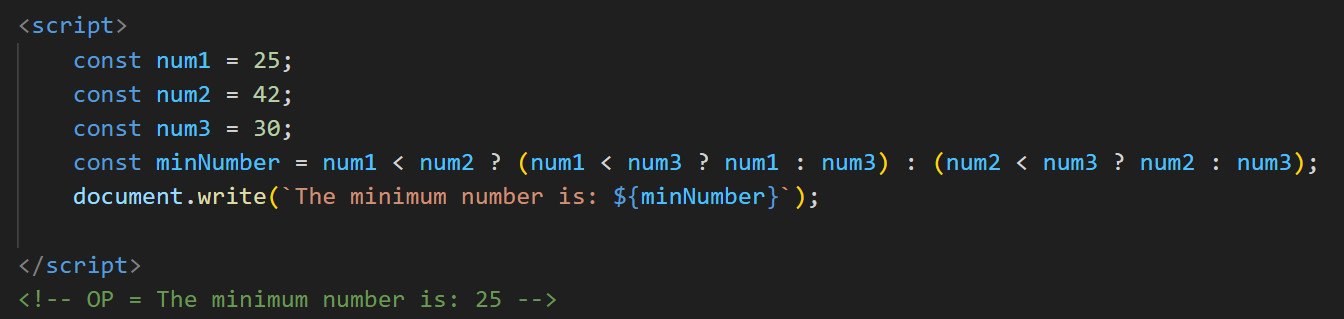
Q. Write to find number is even or odd using ternary operator in JS?



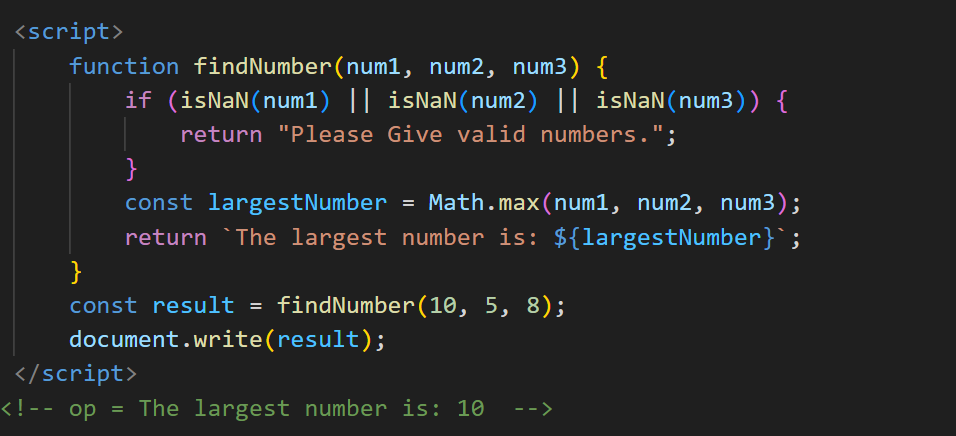
Q. Write find maximum number among 3 numbers using ternary operator in JS?



Q. Write to find minimum number among 3 numbers using ternary operator in JS?



Q. Write to find the largest of three numbers in JS?



Q. Write to show

i. Monday to Sunday using switch case in JS?

ii. Vowel or Consonant using switch case in JS?

Ans.  <script>

        const dayNumber = 4;

        switch (dayNumber) {

            case 1:

                document.write("Monday");

                break;

            case 2:

                document.write("Tuesday");

                break;

            case 3:

                document.write("Wednesday");

                break;

            case 4:

                document.write("Thursday");

                break;

            case 5:

                document.write("Friday");

                break;

            case 6:

                document.write("Saturday");

                break;

            case 7:

                document.write("Sunday");

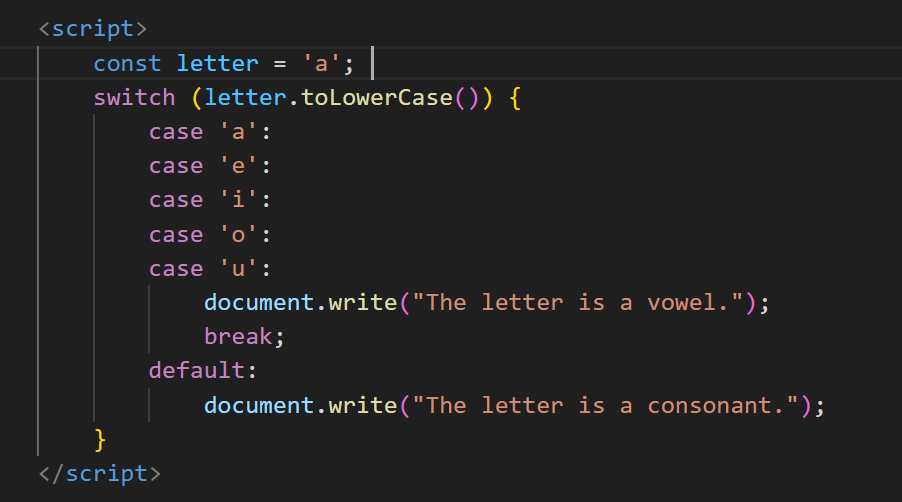
                break;

            default:

                document.write("Invalid day number");

        }

    </script>



**(Conditional looping logic Question)**

Q. What are the looping structures in JavaScript? Any one Example?

Ans. Looping structures in JavaScript are used to repeatedly execute a block of code until a certain condition is met. Here are the main types of looping structures in JavaScript:

1. for Loop:

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

console.log(i);

}

1. while Loop:

let i = 0;

while (i < 5) {

console.log(i);

i++;

}

1. do...while Loop:

let i = 0;

do {

console.log(i);

i++;

} while (i < 5);

1. for...of Loop:

const fruits = ['apple', 'banana', 'orange'];

for (let fruit of fruits) {

console.log(fruit);

}

**Q.** Write a print 972 to 897 using for loop in JS?

**Ans**. To print integers from 972 to 897 in JavaScript using a "for" loop, create a loop that begins at 972, decreases by 1 for each iteration, and keeps going until it reaches the final result (897).

**This loop initializes i to 972, decrements i by 1 after each iteration, and prints the value of i for each iteration. It then performs the loop body for as long as i is higher than or equal to 897. The results will show the numbers 897 through 972.**

 <script>

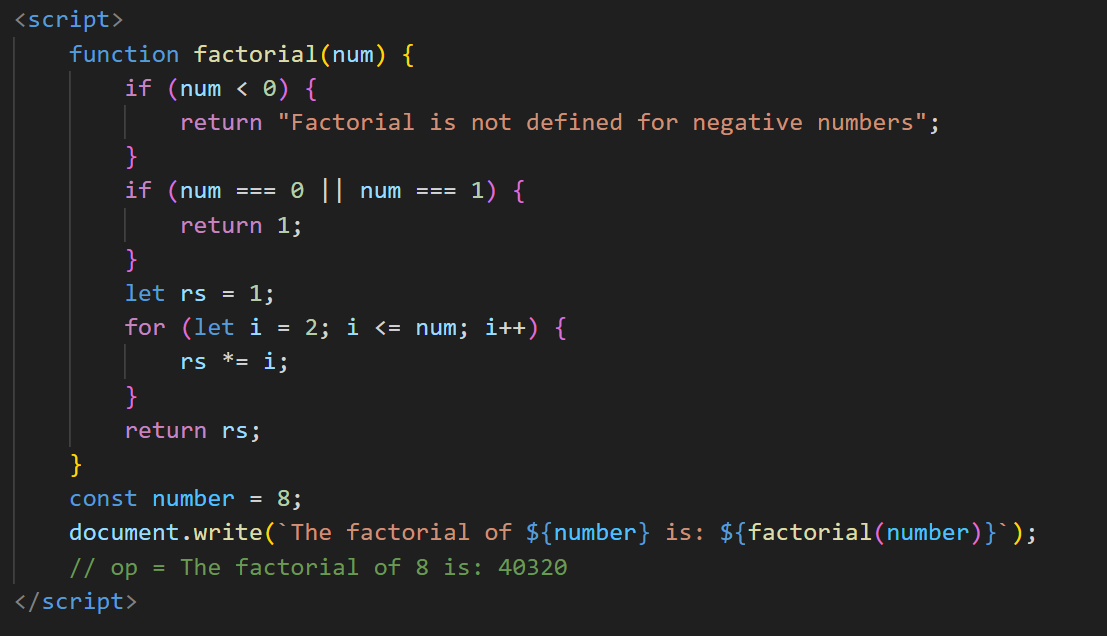
        for (var i = 972; i >= 897; i--) {

            document.write(i + "<br>");

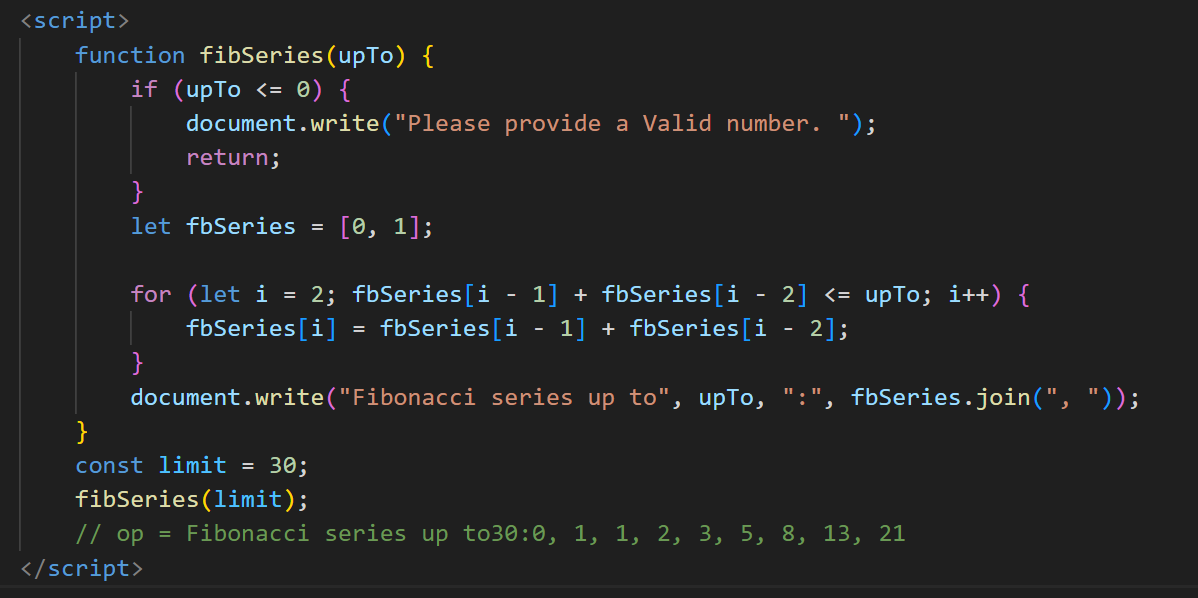
        }

    </script>

Q. Write to print factorial of given number?

****

Q. Write to print Fibonacci series up to given numbers?

****

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

**Ans**.     <script>

        const originalNumber = 64728;

        const reversedNumber = Number(originalNumber.toString().split('').reverse().join(''));

        document.write(reversedNumber);

    </script>

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

**Ans**.  <script>

        function summation(number) {

            let sum = 0;

            const numberString = Math.abs(number).toString();

            for (let i = 0; i < numberString.length; i++) {

                sum += parseInt(numberString[i]);

            }

            return number < 0 ? -sum : sum;

        }

        const Number = 1523;

        const result = summation(inputNumber);

        document.write(`The summation of the digits of ${Number} is: ${result}`);

    </script>

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

 <script>

        function sumOfFirstAndLastDigit(number) {

            const numberString = Math.abs(number).toString();

            const firstDigit = parseInt(numberString[0]);

            const lastDigit = parseInt(numberString[numberString.length - 1]);

            const sum = firstDigit + lastDigit;

            return number < 0 ? -sum : sum;

        }

        const inputNumber = 1234;

        const result = sumOfFirstAndLastDigit(inputNumber);

        document.write(`The summation of the first and last digits of ${inputNumber} is: ${result}`);

    </script>

Q. 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

 <script>

        for (let i = 1; i <= 5; i++) {

            let row = '<br>';

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

                if (j === 0) {

                    row += i + '';

                } else {

                    row += Math.pow(i, j) + '';

                }

            }

            console.log(row.trim());

        }

    </script>

1

1 0

1 0 1

1 0 1 0

1 0 1 0 1 (Use pattern in console.log in JS? )

 <script>

        for (let i = 1; i <= 5; i++) {

            let row = '<br>';

            for (let j = 1; j <= i; j++) {

                row += (j % 2 === 1) ? '1 ' : '0 ';

            }

            console.log(row.trim());

        }

    </script>

A

B C

D E F

G H I J

K L M N O

 <script>

        let currentChar = 'A';

        for (let i = 1; i <= 5; i++) {

            let row = '<br>';

            for (let j = 1; j <= i; j++) {

                row += currentChar + ' ';

                currentChar = String.fromCharCode(currentChar.charCodeAt(0) + 1);

            }

            console.log(row.trim());

        }

    </script>

1

2 3

4 5 6

7 8 9 10

11 12 13 14 15

 <script>

        let currentNumber = 1;

        for (var i = 1; i <= 5; i++) {

            for (var j = 1; j <= i; j++) {

                console.log(" " + currentNumber);

                currentNumber++;

           }

            console.log ("</br>");

        }

    </script>

\*

\* \*

\* \* \*

\* \* \* \*

\* \* \* \* \*

   <script>

        for (var i = 1; i <= 5; i++) {

            for (var j = 1; j <= i; j++) {

                console.log("\*");

            }

            console.log("</br>");

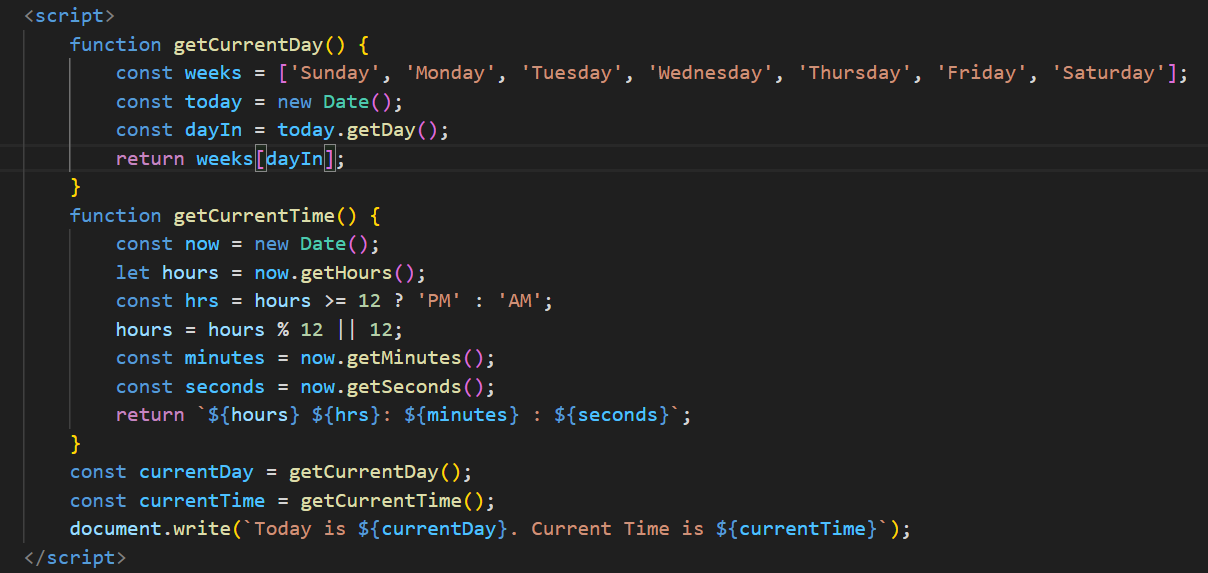
        }

    </script>

***(Array and object Question)***

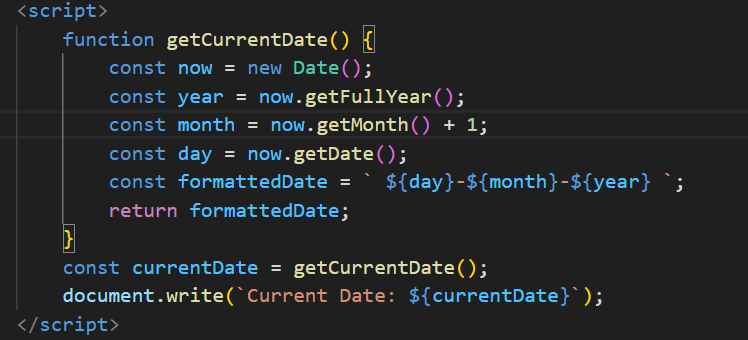
Q. Write a JavaScript Program to display the current day and time in the following format. Sample Output: Today is Friday. Current Time is 12 PM: 12 : 22 2 ?

Ans.



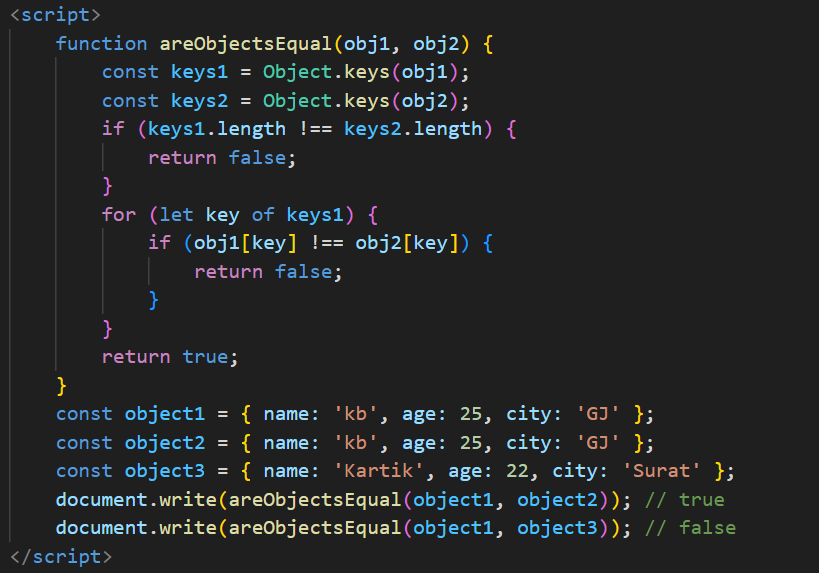
Q. Write a JavaScript program to get the current date?

Ans.



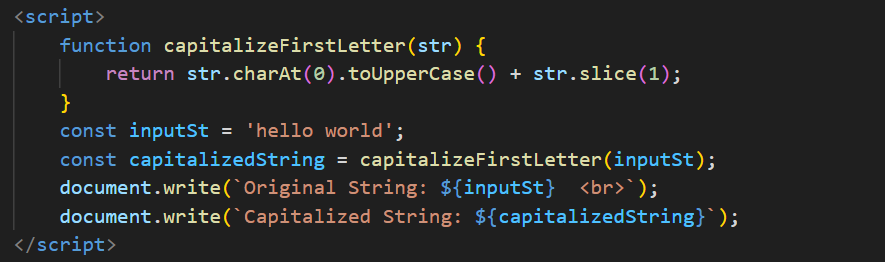
Q. Write a JavaScript program to compare two objects?

Ans.

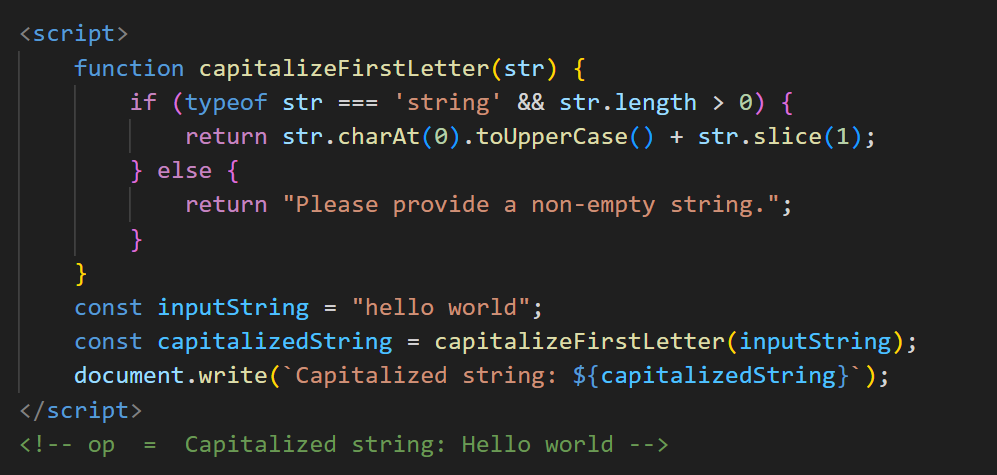


Q. Write a JavaScript program to convert an array of objects into CSV string?

Ans.

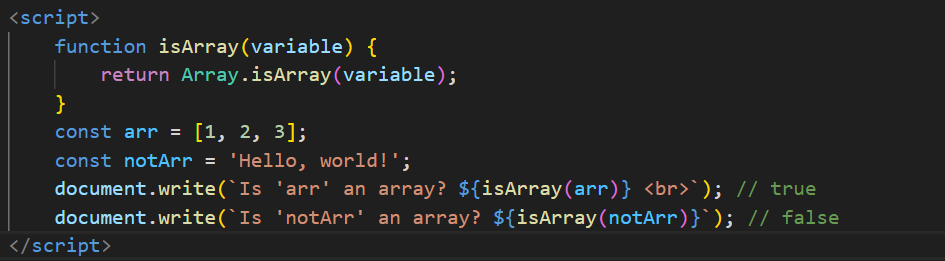


Q. Write a JavaScript program to capitalize first letter of a string?

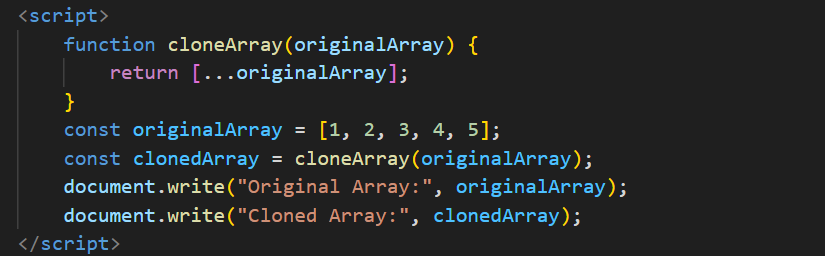


Q. Write a JavaScript program to determine if a variable is array

Ans.



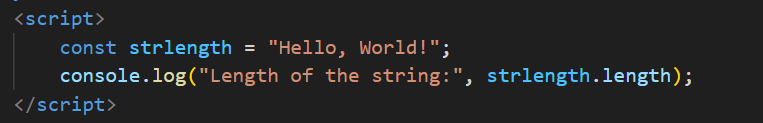
Q. Write a JavaScript program to clone an array?



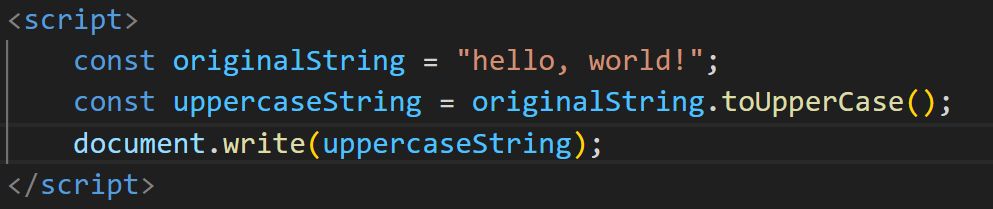
Q. What is the drawback of declaring methods directly in JavaScript objects?

* Memory Usage: When you declare methods directly within an object, those methods are copied into each instance of that object. This may result in more memory being used, particularly if the item is used frequently. It might be more memory-efficient to define the methods only once in the prototype chain if they carry out the same tasks for every instance.
* No Method Sharing: Instances do not share methods that are explicitly specified in objects. Every instance of an object you create will have a copy of the methods on its own. Redundancy may arise from this, and modifications made to one instance's method won't have an impact on other instances' methods.
* Updates to a method can be challenging because you have to update it in every instance of the object if you later decide to expand or update it. This can be prone to mistakes and could result in inconsistent data if you neglect to update a specific instance.
* Problems with Prototypal Inheritance: Methods stated explicitly in an object will not be inherited by its prototypes if you subsequently choose to extend it via prototypal inheritance. If your expectation is that inherited objects would have the same methods, this could result in unexpected behavior.

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



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



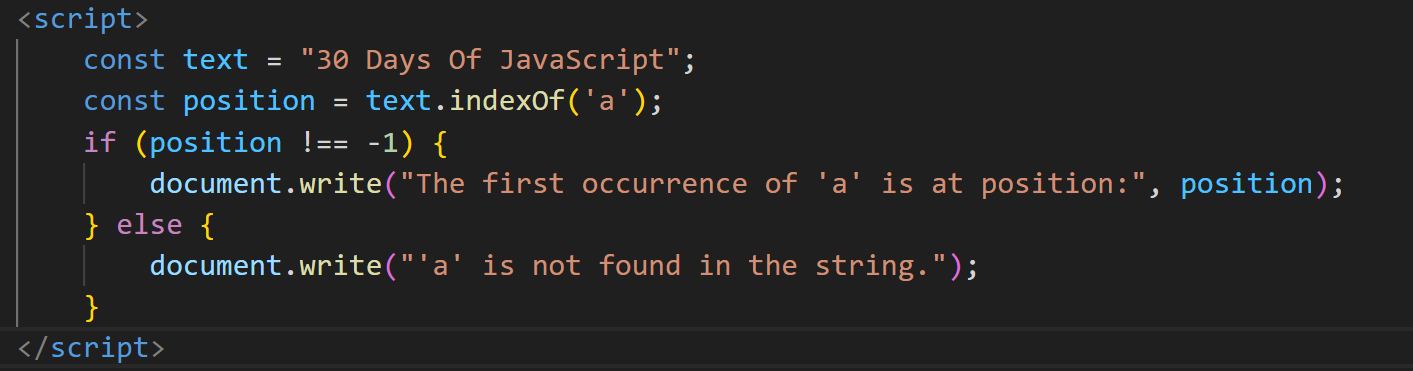
Q. What is the drawback of declaring methods directly in JavaScript objects?

* Methods are exposed as a component of the public interface of an object when they are declared directly in the object. Since encapsulation cannot be enforced natively, all methods are available and changeable from outside the object.
* An object's methods are copied into each instance when it is created in multiples. This may result in redundant memory usage, particularly if there are numerous instances of the object or if the methods are complicated.
* Methods stated explicitly in objects may not be shared across instances efficiently if you subsequently decide to use inheritance and build a prototype chain. This may lead to a code structure that is harder to maintain and uses less memory.
* Directly stated methods within objects might not be as reusable between projects or other objects. Distinguishing functions or modules among methods improves code reusability.
* You must alter or change a method for each instance of an object separately if you wish to do so for all instances of the object. This can be error-prone and time-consuming, particularly in bigger codebases.
* Naming conflicts may arise if various methods on various objects have the same name. In larger codebases where numerous developers participate, this problem becomes increasingly common.

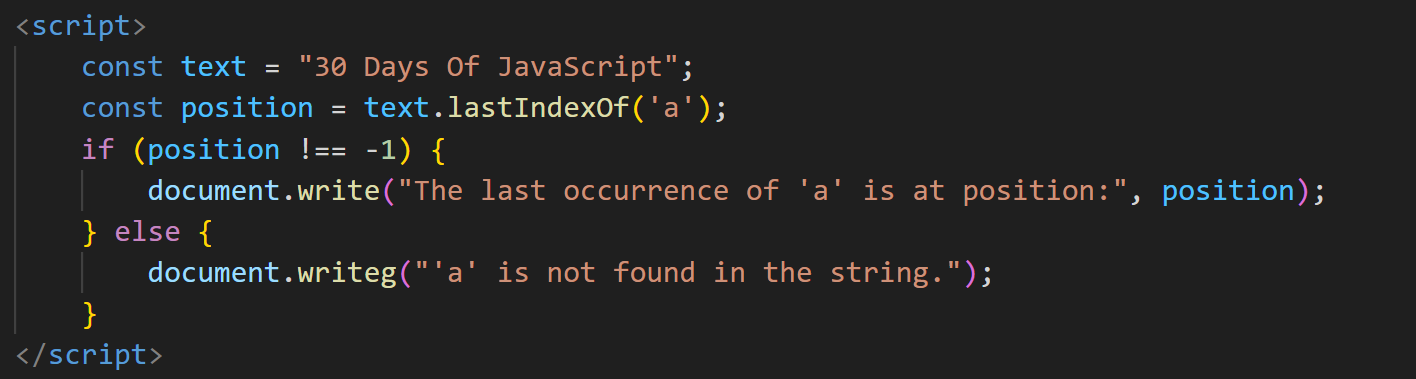
Q. Write a JavaScript program to get the current date. Expected Output : mm-dd-yyyy, mm/dd/yyyy or dd-mm-yyyy, dd/mm/yyyy?



Q. Use indexOf to determine the position of the first occurrence of a in 30 Days Of JavaScript?



Q. Use lastIndexOf to determine the position of the last occurrence of a in 30 Days Of JavaScript?



Q. how many types of JS Event? How to use it?

Ans.

Mouse Events:

* click: Triggered when the user clicks an element.
* dblclick: Triggered when the user double-clicks an element.
* mousedown: Triggered when the user presses a mouse button over an element.
* mouseup: Triggered when the user releases a mouse button over an element.
* mousemove: Triggered when the user moves the mouse pointer over an element.
* mouseover and mouseout: Triggered when the mouse pointer enters or leaves an element.

Keyboard Events:

* keydown: Triggered when a key is pressed down.
* keyup: Triggered when a key is released.
* keypress: Triggered when a key is pressed and released.

Structured Events:

* submit: Started as soon as a form is sent in.
* change: Initiated when an input element's value shifts.
* input: Always activated when an input element's value fluctuates.
* blur and focus: Initiated when an element experiences a change of focus.

Q. What is Bom vs Dom in JS?

Ans. DOM (Document Object Model):

* The DOM represents the structure of a document as a tree of objects. Each object corresponds to a part of the document, such as elements, attributes, and text.
* Document: The top-level object that represents the entire HTML or XML document.
* Element: Represents HTML or XML elements in the document, like <div>, <p>, etc.
* Attribute: Represents attributes of elements, like id, class, etc.
* Text: Represents the actual text content within elements.

Usage:

* DOM is used to dynamically manipulate and interact with the content of a web page. It allows you to create, modify, and delete elements, attributes, and text dynamically.

BOM (Browser Object Model):

* The BOM represents the browser itself and provides objects and methods for interacting with the browser window and its components.
* Window: The top-level object that represents the browser window. It provides properties and methods for controlling the browser window
* Navigator: Provides information about the browser and its capabilities.
* Screen: Represents the user's screen and provides information like screen width and height.
* Location: Represents the URL of the current page.
* History: Represents the browser's session history.

Usage:

* BOM is used for tasks related to the browser itself, such as controlling the size and position of the browser window, navigating to different pages, and managing the browser's history.

Q. Array vs object defences in JS?

Arrays:

* An array is an ordered, indexed collection of values. Each element in an array is identified by an index, starting from 0.
* Arrays can hold values of any data type, including numbers, strings, objects, or even other arrays.

Syn = const myArray = [1, 'two', { key: 'value' }, true];

Use Cases:

* Best suited for when you have a list of similar or related items and need to perform operations on them collectively.
* Useful for operations like iteration, filtering, mapping, and reducing elements.

Objects:

* An object is an unordered, key-value pair collection of data. Each key (also called a property) is associated with a value.
* Objects can hold values of any data type, and the keys are typically strings or symbols.

Syn = const myObject = {

key1: 'value1',

key2: 42,

key3: { nestedKey: 'nestedValue' },

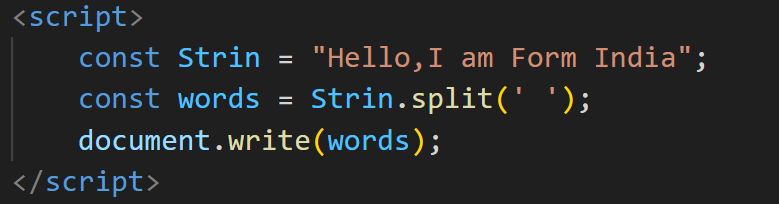
key4: true

};

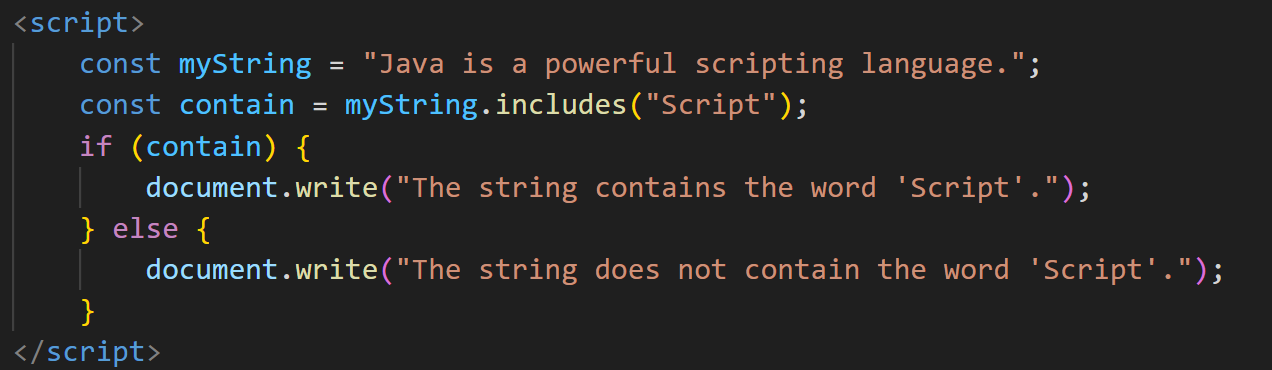
Use Cases:

* Best suited for representing entities with properties or attributes.
* Useful when you need to access values using descriptive keys.

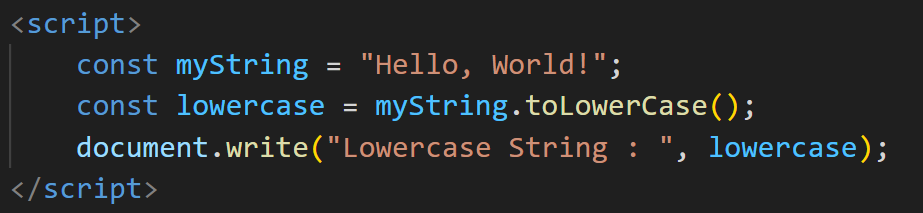
Q. Split the string into an array using split() Method?



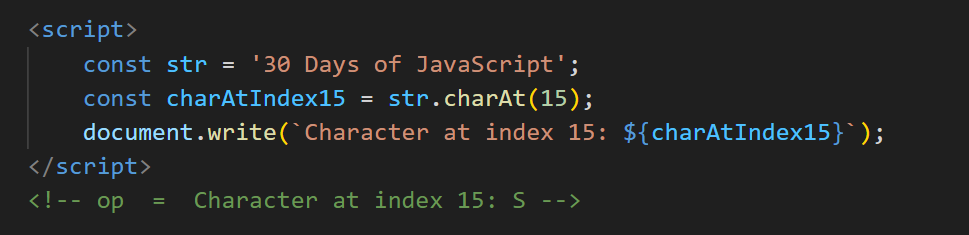
Q. Check if the string contains a word Script using includes() method?



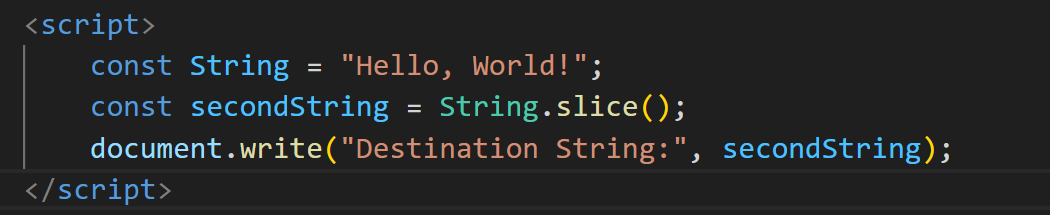
Q. Change all the string characters to lowercase letters using toLowerCase() Method.



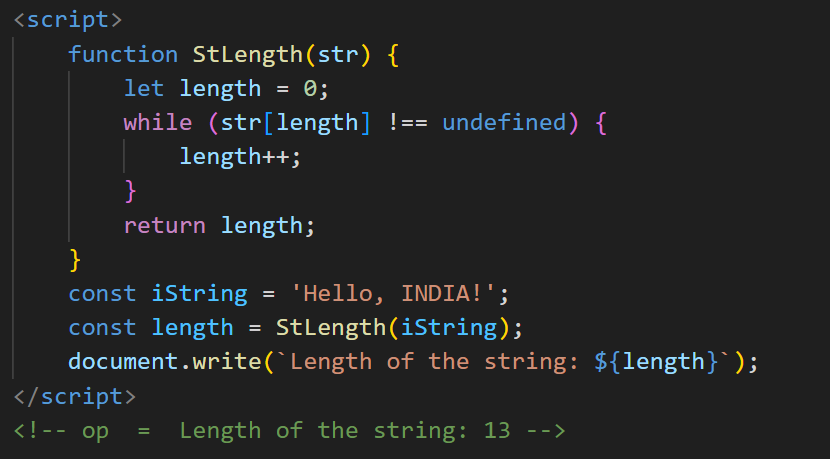
Q. What is Character at index 15 in ’30 Days of JavaScript’ string? Use charAt() method.



Q. copy to one string to another string in JS?



Q. Find the length of a string without using libraryFunction?



Q. What is JavaScript?

* JavaScript is a high-level, interpreted programming language primarily used for client-side web development. It enables interactive web pages and is an essential technology for building modern web applications. JavaScript, often abbreviated as JS, is a versatile language that can also be used for server-side development, mobile app development, and more.
* JavaScript is mainly known for its role in client-side scripting. It is embedded in HTML documents and executed by web browsers, allowing developers to create dynamic and interactive web pages.
* JavaScript is a dynamically-typed language, meaning you don't need to declare the data type of a variable explicitly. It is also weakly typed, allowing for flexibility in variable types.
* JavaScript is object-oriented, supporting object creation and manipulation. It uses prototypes rather than classical inheritance.
* With the advent of technologies like Node.js, JavaScript can also be used for server-side development. This allows developers to use JavaScript for both the client and server sides of an application.

Q. What is the use of isNaN function?

* JavaScript's isNaN function is used to ascertain whether a given integer is NaN (Not-a-Number). Whether the supplied value is NaN is indicated by the boolean value that is returned.
* The method returns true if the parameter supplied to isNaN is a string that can be converted to a numeric value or a numeric value, and the result is NaN.
* The function returns false if the argument is a string that can be transformed to a numeric value or a numeric value, and the output is a legitimate integer.
* The function returns true if the argument is any other kind of data type.

Q. What is negative Infinity

Ans. Infinity Negative:

* Infinity, which stands for negative infinity, is the opposite of infinity. It's employed to signify a value that's less than any other number.
* It might be the outcome of mathematical processes that produce values larger than JavaScript's minimum representable number.

Q. Which company developed JavaScript?

Ans. JavaScript was developed by Netscape Communications Corporation. It was created by Brendan Eich in 1995 while he was working at Netscape. Initially, it was named "Mocha," but later it was renamed to "LiveScript" and then to "JavaScript." The renaming coincided with a partnership between Netscape and Sun Microsystems, and JavaScript was intended to complement Java in web browsers. Despite the similar name, JavaScript and Java are distinct languages with different purposes and features.

Q. What are undeclared and undefined variables?

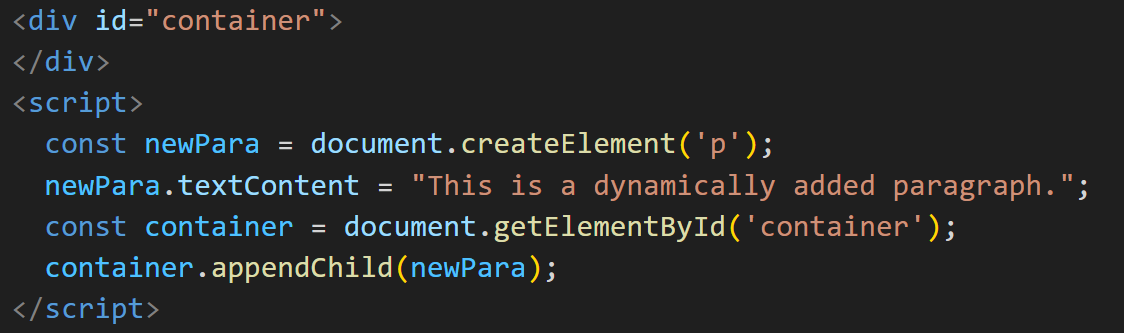
Ans . Undeclared Variables:

* An undeclared variable is a variable that has been used in code without being declared using the var, let, or const keyword.
* When you try to use a variable without declaring it, JavaScript will implicitly create a global variable (if not in strict mode) or raise a reference error (in strict mode).

Undefined Variables:

* An undefined variable is a variable that has been declared but not assigned a value or a variable explicitly set to the value undefined.
* When you declare a variable but don't assign a value to it, JavaScript initializes it with the value undefined.
* Variables explicitly set to undefined may indicate intentional lack of value or a reset to an undefined state.

Q. Write the code for adding new elements dynamically?



Q. What is the difference between ViewState and SessionState?

Ans. ViewState:

* Scope: During postbacks, a single web page's state is preserved via ViewState. In between rounds of requests to the server, it maintains the state of controls on a page.
* Storage: ViewState data is transmitted back and forth between the client and the server after being saved in a hidden field on the page.
* Lifetime: ViewState has a brief lifespan and is limited to the lifecycle of a single page. It is employed to store data while a single web page is being processed.
* Use: ViewState is frequently used to store little pieces of information unique to a single page, including user preferences or control attributes.

State of the Session:

* Scope: During a user's session, however, SessionState is utilized to keep state across several pages. It makes session-specific data retrieval and storage possible.
* Storage: A session identifier is supplied to the client to link subsequent requests to the right session, and SessionState data is stored on the server.
* SessionState is persistent for the duration of the user's session. It can be used to hold data that needs to be kept updated while a user interacts with the web application and is spread across several pages.
* Use: SessionState can be used to store user-specific data, such as preferences, login status, and other information pertinent to the duration of the session.

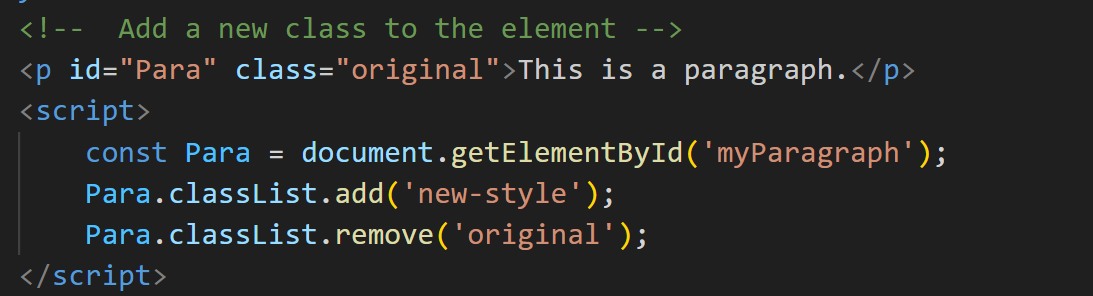
Q. What is === operator?

ANS. The === operator in JavaScript is called the "strict equality" operator. It is used to compare two values for equality without performing type coercion. In other words, it checks both the values and their types, and it returns true if they are strictly equal, and false otherwise.

* document.write(5 === 5); // true
* document.write("5" === 5); // false (different types)
* document.write(5 === '5'); // false (different types)
* document.write(5 === 6); // false (different values)

Q. How can the style/class of an element be changed?





Q. How to read and write a file using JavaScript?

Ans. JavaScript's ability to directly read and write files on the user's device is restricted by security measures in a web browser environment. Nonetheless, there are a number of indirect ways that you can interact with files.

File Input Element: To enable file selection by users, utilize the HTML <input type= "file"> element. Once a file has been chosen, you may use the File API to view its contents.

Q. What are all the looping structures in JavaScript?

1. for Loop:

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

console.log(i);

}

1. while Loop:

let i = 0;

while (i < 5) {

console.log(i);

i++;

}

1. do...while Loop:

let i = 0;

do {

console.log(i);

i++;

} while (i < 5);

1. for...of Loop:

const fruits = ['apple', 'banana', 'orange'];

for (let fruit of fruits) {

console.log(fruit);

}

1. for Loop:

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

console.log(i);

}

1. while Loop:

let i = 0;

while (i < 5) {

console.log(i);

i++;

}

1. do...while Loop:

let i = 0;

do {

console.log(i);

i++;

} while (i < 5);

1. for...of Loop:

const fruits = ['apple', 'banana', 'orange'];

for (let fruit of fruits) {

console.log(fruit);

}

Q. How can you convert the string of any base to an integer in JavaScript?

* The parseInt function in JavaScript can be used to convert a string representation of a number in any base (binary, octal, or hexadecimal, for example) to an integer. The string to be converted and the numeric system base are the two arguments passed to the parseInt function.

 <script>

        const binaryString = "1101";

        const decimalNumber = parseInt(binaryString, 2);

        console.log(decimalNumber);

    </script>

// Outputs: 13

Q. What is the function of the delete operator?

The delete operator in JavaScript is mostly used to remove a property from an object. We will study about JavaScript's delete operator in this article.

Syn= delete object , delete object.property, delete object[‘property’]

Q. What are all the types of Pop up boxes available in JavaScript?

1. Alert Box = The alert() function displays a simple dialog box with a message and an "OK" button. It is commonly used for displaying information to the user.

Syn = alert("This is an alert box!");

1. Confirm Box (confirm()): = The confirm() function displays a dialog box with a message, an "OK" button, and a "Cancel" button. It is used for asking the user to confirm or cancel an action.

Syn = const result = confirm("Do you want to proceed?");

if (result) {

console.log("User clicked OK.");

} else {

console.log("User clicked Cancel.");

}

1. Prompt Box (prompt()):= The prompt() function displays a dialog box with a message, an input field for the user to enter text, and "OK" and "Cancel" buttons. It is used for taking user input.

Syn = const userInput = prompt("Please enter your name:", "John Doe");

if (userInput !== null) {

console.log("User entered: " + userInput);

} else {

console.log("User clicked Cancel.");

}

Q. What is the use of Void (0)?

* In JavaScript, void(0) is often used to create a "void" or "undefined" value and is commonly seen in the context of anchor (<a>) elements in HTML. It is used to prevent the default action of a hyperlink while still allowing the execution of JavaScript code when the link is clicked.
* <a href="javascript:void(0);" onclick="myFunction()">Click me</a>
* The value of the href attribute is javascript:void(0);.
* There is a JavaScript function (myFunction in this case) assigned to the onclick attribute.
* Using this technique, you can make a clickable element (such as a button or link) that, when clicked, opens on an other page. The void(0) expression prevents the browser from going to a new page by returning undefined and acting as a placeholder for a URL in the href property.
* It's important to note that javascript:void(0) is no longer widely used in modern web development. Instead, developers frequently handle activities without relying on the href attribute by using event listeners in JavaScript or by using # as a placeholder in the href attribute. Another issue with using # is that it could make the page scroll to the top. Thus, using JavaScript to handle the click event is a more contemporary and streamlined method.

Q. How can a page be forced to load another page in JavaScript?

* In JavaScript, you can use the window.location object to navigate to another page. The window.location object represents the current URL of the browser. You can change its href property to a new URL to force the page to load another page.
* window.location.href = "https://www.example.com";
* This line of code sets the href property of window.location to a new URL, which effectively forces the browser to load the specified page.

Q. What are the disadvantages of using innerHTML in JavaScript?

* Inserting dynamic content with user inputs or untrusted data via innerHTML puts your application at risk of cross-site scripting (XSS) attacks. The material may run malicious programs if it is not thoroughly cleaned.
* It can be computationally costly to parse and render HTML material when modifying an element's innerHTML, particularly for big and sophisticated HTML structures. It may be more effective to use more focused techniques like textContent or to create and attach new DOM elements if you simply need to alter a piece of the content.
* Memory leaks can occur by repeatedly updating content with innerHTML without properly clearing up event listeners or other associated data. This is a result of the browser perhaps holding onto memory related to outdated content.
* If you edit or modify an element's content using innerHTML, any event listeners that are currently attached to the elements inside that container might be removed. If the listeners were not reattached, this can result in unexpected behavior.
* An element's whole content is replaced when you set content with innerHTML. Other methods, such createElement, appendChild, and setAttribute, may be more appropriate for altering the DOM if you require more granular control over specific components or attributes.
* InnerHTML may be superfluous if you're working with plain text and don't need to parse or edit HTML. The textContent attribute is more suitable in these circumstances and does not expose users to any security vulnerabilities related to HTML processing.

Q. Create password field with show hide functionalities

Q. Create result show hide functionalities

Q. Create basic math operation in JS

Q. Form Validation in JS?

All Ans in this Link = <https://github.com/kbu09/JS-TASK>