Module (JAVASCRIPT BASIC & DOM) – 4

Q.1 What is JavaScript. How to use it?

Ans.

JavaScript (js) is a light-weight object-oriented programming language which is used by several websites for scripting the webpages. It is an interpreted, full-fledged programming language that enables dynamic interactivity on websites when applied to an HTML document. It was introduced in the year 1995 for adding programs to the webpages in the Netscape Navigator browser. With JavaScript, users can build modern web applications to interact directly without reloading the page every time.

We can use JavaScript 2 ways:

(1)Embedding JavaScript in HTML: JavaScript code can be embedded directly within HTML documents using `<script>` tags.

(2)External JavaScript Files: For larger JavaScript codebases, it's common to separate JavaScript into external `.js` files and link them to your HTML documents using `<script>` tags

Q.2 How many type of Variable in JavaScript?

Ans.

There are two types of variables in JavaScript : local variable and global variable.

Q.3 Define a Data Types in js?

Ans.

JavaScript provides different data types to hold different types of values. There are two types of data types in JavaScript.

1. Primitive data type :

String -represents sequence of characters e.g. "hello"

Number - represents numeric values e.g. 100

Boolean - represents boolean value either false or true

Undefined - represents undefined value

Null - represents null i.e. no value at all

2. Non-primitive (reference) data type :

Object - represents instance through which we can access members

Array - represents group of similar values

Q.4 Write a mul Function Which will Work Properly When invoked With Following Syntax.

Ans.

```
function mul(num1) {
  function mul1(num2) {
      function mul2(num3) {
          return num1 * num2 * num3;
      }; // end of mul2()
      return mul2;
  }; // end of mul1()
  return mul1;
} // end of mul()
```

Q.5 What the deference between undefined and undeclare in JavaScript?

Ans.

(1) **Undefined**:

- In JavaScript, "undefined" is a primitive value that means a variable has been declared but has not been assigned a value.

(2) **Undeclared**:

- "Undeclared" refers to a situation where a variable is used without being declared first using "`var`, `let` or `const`."

Q.6 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().

Ans.
```
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.");
```

Q.7 Check if typeof '10' is exactly equal to 10. If not make it exactly equal?

Ans.

```
// Check if typeof '10' is exactly equal to 10
    if (typeof '10' !== 'number') {
        // Convert '10' to a number
        let num = Number('10'); // Using Number to convert '10' to 10
        console.log(num); // Output: 10
    } else {
        console.log('typeof "10" is exactly equal to 10');
    }
```

Q.8 Write a JavaScript Program to find the area of a triangle?

Ans.

```
// Function to calculate the area of a triangle
function calculateTriangleArea(base, height) {
    // Calculate the area using the formula: A = 1/2 * base * height
    let area = 0.5 * base * height;
    return area;
}

// Example usage:
let base = 5;
let height = 8;
let area = calculateTriangleArea(base, height);

console.log(`The area of the triangle with base ${base} and height ${height} is:
${area}`);
```

Q.9 Write a JavaScript program to calculate days left until next Christmas?

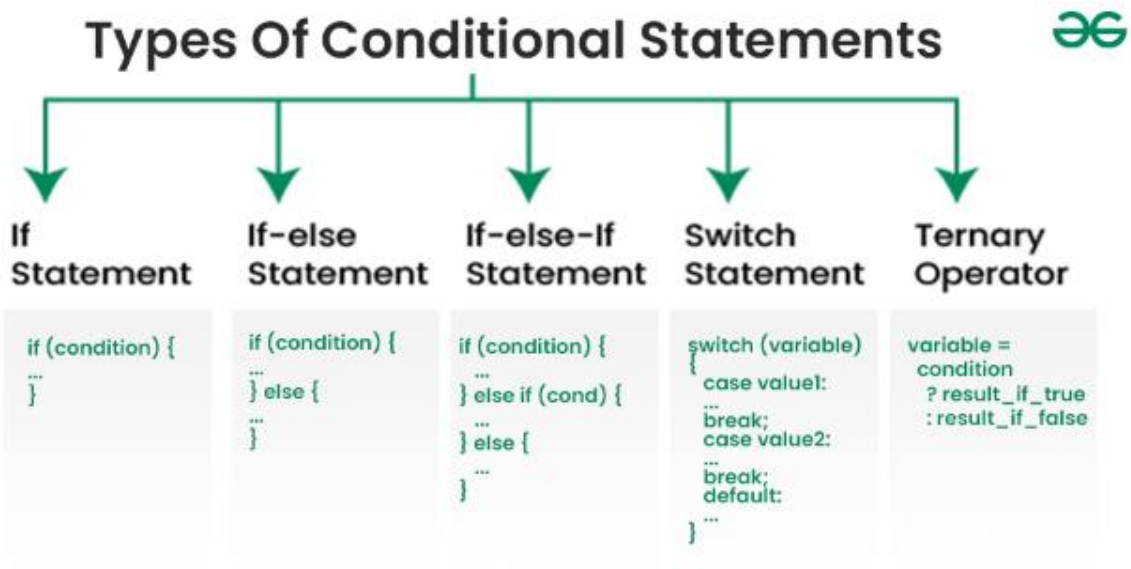
Ans.

```
function daysUntilChristmas() {
    // Get the current date
    let today = new Date();

    // Get the current year
    let currentYear = today.getFullYear();

    // Christmas for the current year
    let christmas = new Date(currentYear, 11, 25); // Month is 0-based, so 11 is
December
```

```
    // Check if Christmas has already passed this year
    if (today.getMonth() === 11 && today.getDate() > 25) {
        christmas.setFullYear(currentYear + 1); // Set to next year if Christmas
has passed
    }

    // Calculate the difference in milliseconds between the current date and
Christmas
    let difference = christmas.getTime() - today.getTime();

    // Convert milliseconds to days
    let daysLeft = Math.ceil(difference / (1000 * 60 * 60 * 24));

    return daysLeft;
}

// Call the function to get the days left until Christmas
let daysLeft = daysUntilChristmas();

// Display the result
console.log(`There are ${daysLeft} days left until Christmas.`);
```

Q.10 What is Condition Statement?

Ans.



Q.11 Find circumference of Rectangle formula : C = 4 * a ?

Ans.

Rectangles have a perimeter, which is the total distance around the outside edges of the shape. The perimeter PPP of a rectangle is calculated differently based on its dimensions:

For a rectangle with sides length and width, the perimeter P is given by:

P=2×(L+W)

```
// Function to calculate the perimeter of a rectangle
function calculatePerimeter(length, width) {
    // Calculate perimeter using the formula P = 2 * (L + W)
    let perimeter = 2 * (length + width);
    return perimeter;
}

// Example usage:
let length = 8;
let width = 5;
let result = calculatePerimeter(length, width);

console.log(`The perimeter of the rectangle with length ${length} and width
${width} is: ${result}`);
```

Q.12 WAP to convert years into days and days into years?

Ans.

(1)Year to Days:

```
function yearsToDays(years) {
    return years * 365;
}

// Example usage:
let years = 1;
let days = yearsToDays(years);

console.log(`${years} years are equal to ${days} days.`);
```

(2)Days to Year:

```
// Function to convert days to years
function daysToYears(days) {
    return days / 365;
}

// Example usage:
let days = 1825; // Example: 5 years * 365 days/year = 1825 days
let years = daysToYears(days);

console.log(`${days} days are equal to approximately ${years.toFixed(2)}
years.`);
```

Q.13 Convert temperature Fahrenheit to Celsius? (Conditional logic Question).\

Ans.

```
// Function to convert Fahrenheit to Celsius
function fahrenheitToCelsius(fahrenheit) {
    let celsius = (5 / 9) * (fahrenheit - 32);
    return celsius;
}

// Example usage:
let fahrenheit = 68; // Example temperature in Fahrenheit
let celsius = fahrenheitToCelsius(fahrenheit);

console.log(`${fahrenheit} degrees Fahrenheit is equal to ${celsius.toFixed(2)}
degrees Celsius.`);
```

Q.14 Write a JavaScript exercise to get the extension of a filename.?

Ans.

```
// Function to get the extension of a filename
function getFileExtension(filename) {
    // Split the filename by dot (.)
    let parts = filename.split('.');

    // Get the last part (which is the extension)
    let extension = parts[parts.length - 1];
```

```
    return extension;
}

// Example usage:
let filename1 = "script.js";
let filename2 = "index.html";

let extension1 = getFileExtension(filename1);
let extension2 = getFileExtension(filename2);

console.log(`The extension of ${filename1} is: ${extension1}`);
console.log(`The extension of ${filename2} is: ${extension2}`);
```

Q.15 What is the result of the expression (5 > 3 && 2 < 4)?

Ans.

Result of the Expression (5 > 3 && 2 < 4) is True.

```
// Evaluating the expression (5 > 3 && 2 < 4)
let result = (5 > 3 && 2 < 4);

// Output the result
console.log(result); // This will print: true
```

Q.16 What is the result of the expression (true && 1 && "hello")?

Ans.

Result of the Expression (true && 1 && "hello") is “hello”.

```
// Evaluating the expression (true && 1 && "hello")
let result = (true && 1 && "hello");

// Output the result
console.log(result); // This will print: hello
```

Q.17 What is the result of the expression true && false || false && true?

Ans.

Result of the Expression true && false || false && true is “false”.

because true&&false is “false” and false&&true is also “false”,

so false || false is “false”

Q.18 What is a Loop and Switch Case in JavaScript define that ?

Ans.

loops and switch cases are control flow mechanisms that allow you to execute code repeatedly or conditionally based on certain conditions. Here’s an explanation of each:

There are 3 types of **Loops**:

(1) for loop: Executes a block of code a specified number of times.

```
for (initialization; condition; iteration) {
    // code block to be executed
}
```

(2)while loop: Executes a block of code as long as the specified condition is true.

```
while (condition) {
    // code block to be executed
}
```

(3)do...while loop: Similar to a while loop, but the block of code is executed once before the condition is tested.

```
do {
    // code block to be executed
} while (condition);
```

Switch Case:

```
switch (expression) {
    case value1:
```

```
        // code block to be executed if expression === value1
        break;
    case value2:
        // code block to be executed if expression === value2
        break;
    ...
    default:
        // code block to be executed if expression doesn't match any case
}
```

Q.19 What is the use of isNaN function?

Ans.

In JavaScript `NaN` is short for "Not-a-Number".

The `isNaN()` method returns true if a value is NaN.

The `isNaN()` method converts the value to a number before testing it.

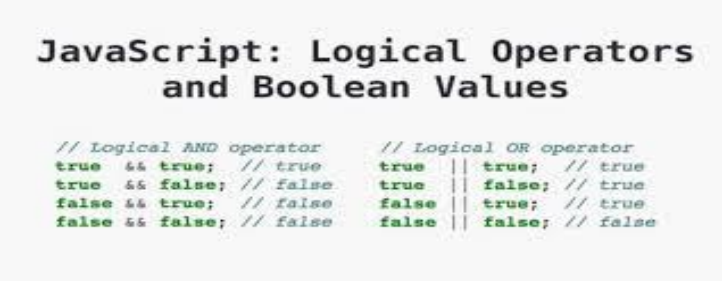
Example:

```
isNaN(NaN);         // true
isNaN(123);         // false
isNaN("123");       // false (converted to number 123)
isNaN("hello");     // true (cannot be converted to a number)
isNaN(true);        // false (converted to number 1)
isNaN(undefined);   // true (undefined is converted to NaN)
isNaN("");          // false (empty string is converted to 0)
```

Q.20 What is the difference between && and || in JavaScript?

Ans.

In JavaScript, `&&` (logical AND) and `||` (logical OR) are both operators used to perform logical operations on boolean operands.



Q.21 What is the use of Void (0)?

Ans.

Use of Void(0) in JavaScript :

- Returns `undefined` when evaluated.
- Historically used to prevent navigation when used in `<a>` tags (`<a href="javascript:void(0);">Click Me</a>`).
- Ensures an expression evaluates to `undefined` without causing side effects.
- Less commonly used in modern JavaScript due to improvements in web standards and alternative approaches to handling event actions.

Q.22 Check Number Is Positive or Negative in JavaScript?

Ans.

```
function checkNumber(num) {
    if (num > 0) {
        return "Positive";
    } else if (num < 0) {
        return "Negative";
    } else {
        return "Zero";
    }
}
```

Q.23 Find the Character Is Vowel or Not ?

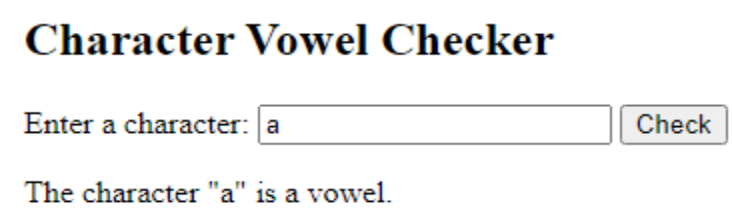
Ans.

```
<!DOCTYPE html>
```

```html
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Character Vowel Checker</title>
    <script>
        // Function to check if a character is a vowel
        function isVowel(char) {
            char = char.toLowerCase();
            const vowels = ['a', 'e', 'i', 'o', 'u'];
            return vowels.includes(char);
        }

        // Function to handle form submission
        function checkCharacter() {
            // Get the input value from the form
            let userInput = document.getElementById("charInput").value.trim();

            // Validate if userInput is a single character
            if (userInput.length === 1) {
                // Check if the character is a vowel using the isVowel function
                if (isVowel(userInput)) {
                    document.getElementById("result").textContent = `The
character "${userInput}" is a vowel.`;
                } else {
                    document.getElementById("result").textContent = `The
character "${userInput}" is not a vowel.`;
                }
            } else {
                alert("Please enter a single character.");
            }

            // Prevent form submission
            return false;
        }
    </script>
</head>
<body>
    <h2>Character Vowel Checker</h2>
    <form onsubmit="return checkCharacter()">
        <label for="charInput">Enter a character:</label>
        <input type="text" id="charInput" name="charInput" maxlength="1"
required>
        <button type="submit">Check</button>
    </form>
    <p id="result"></p>
```

```
</body>
</html>
```

Output:



Q.24 Write to check whether a number is negative, positive or zero?

Ans.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Number Sign Checker</title>
    <script>
        // Function to check whether a number is negative, positive, or zero
        function checkNumber() {
            // Get the input value from the HTML input field
            let userInput = document.getElementById("numberInput").value.trim();

            // Convert user input to a number (assuming the user enters valid
numeric input)
            let num = parseFloat(userInput);

            // Validate if userInput is a valid number
            if (!isNaN(num)) {
                if (num > 0) {
                    document.getElementById("result").textContent = `${num} is
Positive`;
                } else if (num < 0) {
                    document.getElementById("result").textContent = `${num} is
Negative`;
                } else {
                    document.getElementById("result").textContent = `${num} is
Zero`;
                }
            } else {
```

```
                alert("Invalid input. Please enter a valid number.");
            }
        }
    </script>
</head>
<body>
    <h2>Number Sign Checker</h2>
    <p>Enter a number to check if it is Negative, Positive, or Zero:</p>
    <input type="text" id="numberInput" placeholder="Enter number...">
    <button onclick="checkNumber()">Check</button>
    <p id="result"></p>
</body>
</html>
```

Output:

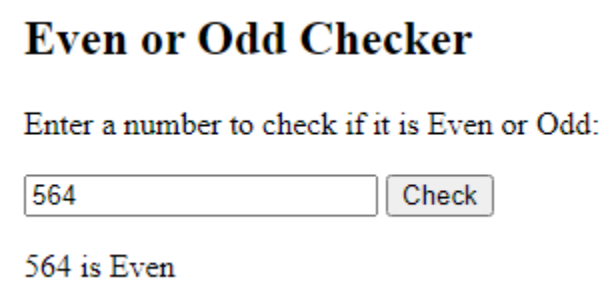
Q.25 Write to find number is even or odd using ternary operator in JS?

Ans.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Even or Odd Checker</title>
    <script>
        // Function to check if a number is even or odd using ternary operator
        function checkEvenOdd() {
            // Get the input value from the HTML input field
            let userInput = document.getElementById("numberInput").value.trim();

            // Convert user input to a number (assuming the user enters valid
numeric input)
            let num = parseInt(userInput);
```

```
            // Validate if userInput is a valid number
            if (!isNaN(num)) {
                // Use ternary operator to check if number is even or odd
                let result = (num % 2 === 0) ? "Even" : "Odd";

                // Display the result
                document.getElementById("result").textContent = `${num} is
${result}`;
            } else {
                alert("Invalid input. Please enter a valid number.");
            }
        }
    </script>
</head>
<body>
    <h2>Even or Odd Checker</h2>
    <p>Enter a number to check if it is Even or Odd:</p>
    <input type="text" id="numberInput" placeholder="Enter number...">
    <button onclick="checkEvenOdd()">Check</button>
    <p id="result"></p>
</body>
</html>
```

Output:



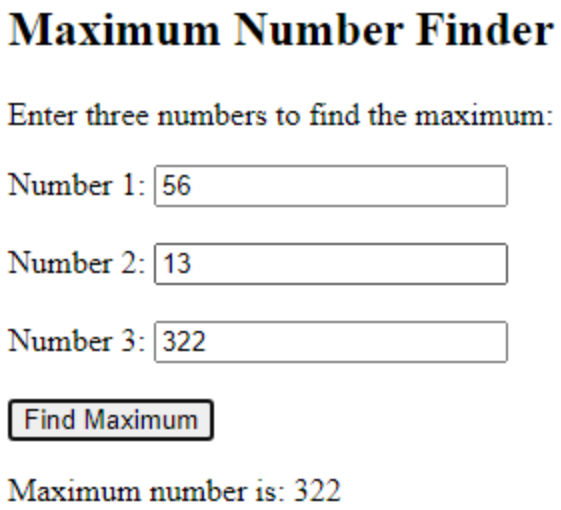
Q.26 Write find maximum number among 3 numbers using ternary operator in JS?

Ans.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Maximum Number Finder</title>
    <script>
```

```
        // Function to find maximum number among three numbers using ternary
operator
        function findMaxNumber() {
            // Get the input values from the HTML input fields
            let num1 = parseFloat(document.getElementById("num1").value.trim());
            let num2 = parseFloat(document.getElementById("num2").value.trim());
            let num3 = parseFloat(document.getElementById("num3").value.trim());

            // Validate if userInput is valid numbers
            if (!isNaN(num1) && !isNaN(num2) && !isNaN(num3)) {
                // Use nested ternary operators to find the maximum number
                let max = (num1 >= num2) ? (num1 >= num3 ? num1 : num3) : (num2
>= num3 ? num2 : num3);

                // Display the result
                document.getElementById("result").textContent = `Maximum number
is: ${max}`;
            } else {
                alert("Invalid input. Please enter valid numbers.");
            }
        }
    </script>
</head>
<body>
    <h2>Maximum Number Finder</h2>
    <p>Enter three numbers to find the maximum:</p>
    <label for="num1">Number 1:</label>
    <input type="text" id="num1" placeholder="Enter number...">
    <br><br>
    <label for="num2">Number 2:</label>
    <input type="text" id="num2" placeholder="Enter number...">
    <br><br>
    <label for="num3">Number 3:</label>
    <input type="text" id="num3" placeholder="Enter number...">
    <br><br>
    <button onclick="findMaxNumber()">Find Maximum</button>
    <p id="result"></p>
</body>
</html>
```

Output:



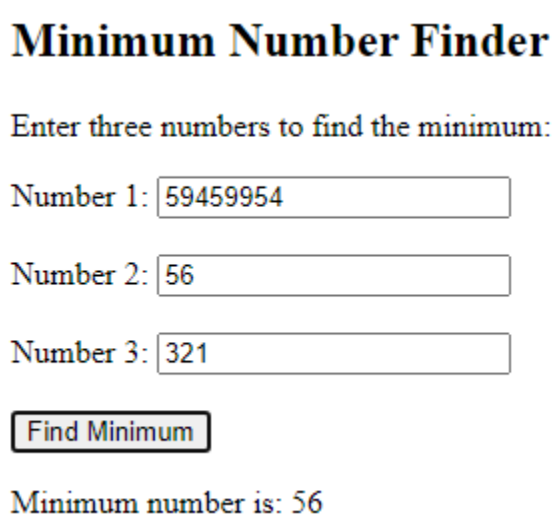
Q.27 Write to find minimum number among 3 numbers using ternary operator in JS?

Ans.

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <title>Minimum Number Finder</title>
</head>

<body>
    <h2>Minimum Number Finder</h2>
    <p>Enter three numbers to find the minimum:</p>
    <label for="num1">Number 1:</label>
    <input type="text" id="num1" placeholder="Enter number...">
    <br><br>
    <label for="num2">Number 2:</label>
    <input type="text" id="num2" placeholder="Enter number...">
    <br><br>
    <label for="num3">Number 3:</label>
    <input type="text" id="num3" placeholder="Enter number...">
    <br><br>
    <button onclick="findMinNumber()">Find Minimum</button>
    <p id="result"></p>
</body>
<script>
    // Function to find minimum number among three numbers using ternary operator
    function findMinNumber() {
        // Get the input values from the HTML input fields
```

```
        let num1 = parseFloat(document.getElementById("num1").value.trim());
        let num2 = parseFloat(document.getElementById("num2").value.trim());
        let num3 = parseFloat(document.getElementById("num3").value.trim());

        // Validate if userInput is valid numbers
        if (!isNaN(num1) && !isNaN(num2) && !isNaN(num3)) {
            // Use nested ternary operators to find the minimum number
            let min = (num1 <= num2) ? (num1 <= num3 ? num1 : num3) : (num2 <=
num3 ? num2 : num3);

            // Display the result
            document.getElementById("result").textContent = `Minimum number is:
${min}`;
        } else {
            alert("Invalid input. Please enter valid numbers.");
        }
    }
</script>

</html>
```

Output:



Q.28 Write to find the largest of three numbers in JS?

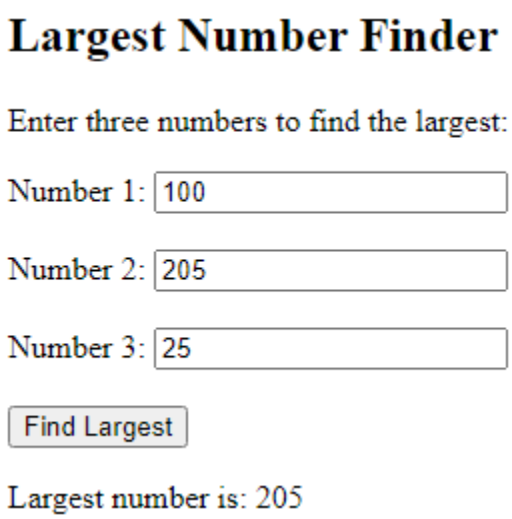
Ans.

```
<!DOCTYPE html>
<html lang="en">
```

```
<head>
    <meta charset="UTF-8">
    <title>Largest Number Finder</title>
</head>

<body>
    <h2>Largest Number Finder</h2>
    <p>Enter three numbers to find the largest:</p>
    <label for="num1">Number 1:</label>
    <input type="text" id="num1" placeholder="Enter number...">
    <br><br>
    <label for="num2">Number 2:</label>
    <input type="text" id="num2" placeholder="Enter number...">
    <br><br>
    <label for="num3">Number 3:</label>
    <input type="text" id="num3" placeholder="Enter number...">
    <br><br>
    <button onclick="findLargestNumber()">Find Largest</button>
    <p id="result"></p>
</body>
<script>
    // Function to find the largest number among three numbers
    function findLargestNumber() {
        // Get the input values from the HTML input fields
        let num1 = parseFloat(document.getElementById("num1").value.trim());
        let num2 = parseFloat(document.getElementById("num2").value.trim());
        let num3 = parseFloat(document.getElementById("num3").value.trim());

        // Validate if userInput is valid numbers
        if (!isNaN(num1) && !isNaN(num2) && !isNaN(num3)) {
            // Determine the largest number using conditional statements
            let largest;

            if (num1 >= num2 && num1 >= num3) {
                largest = num1;
            } else if (num2 >= num1 && num2 >= num3) {
                largest = num2;
            } else {
                largest = num3;
            }

            // Display the result
            document.getElementById("result").textContent = `Largest number is:
${largest}`;
        } else {
```

```
            alert("Invalid input. Please enter valid numbers.");
        }
    }
</script>

</html>
```

Output:



Q.29 Write to show

i. Monday to Sunday using switch case in JS?

ii. Vowel or Consonant using switch case in JS?

Ans.

i. Monday to Sunday using switch case in JS:

```
// Function to display the day of the week using switch case
function displayDay(dayNumber) {
    let dayName;

    switch (dayNumber) {
        case 1:
            dayName = "Monday";
            break;
        case 2:
            dayName = "Tuesday";
            break;
        case 3:
```

```
            dayName = "Wednesday";
            break;
        case 4:
            dayName = "Thursday";
            break;
        case 5:
            dayName = "Friday";
            break;
        case 6:
            dayName = "Saturday";
            break;
        case 7:
            dayName = "Sunday";
            break;
        default:
            dayName = "Invalid day number";
    }

    return dayName;
}

// Example usage:
for (let i = 1; i <= 7; i++) {
    console.log(`Day ${i}: ${displayDay(i)}`);
}
```

ii. Vowel or Consonant using switch case in JS:

```
// Function to check if a character is a vowel or consonant using switch case
function checkVowelConsonant(char) {
    // Convert input to lowercase to handle both cases (uppercase and lowercase)
    char = char.toLowerCase();

    switch (char) {
        case 'a':
        case 'e':
        case 'i':
        case 'o':
        case 'u':
            return `${char} is a vowel`;
        default:
            return `${char} is a consonant`;
    }
}
```

```
// Example usage:
console.log(checkVowelConsonant('a'));  // Output: "a is a vowel"
console.log(checkVowelConsonant('b'));  // Output: "b is a consonant"
console.log(checkVowelConsonant('E'));  // Output: "e is a vowel"
console.log(checkVowelConsonant('Z'));  // Output: "z is a consonant"
```

(Conditional looping logic Question)

Q.30 What are the looping structures in JavaScript? Any one Example?

Ans.

In JavaScript, there are several looping structures that allow you to repeatedly execute a block of code. The main looping structures include:

(1) for Loop
(2) while Loop
(3) do-while Loop
(4) for…in Loop
(5) for…of Loop

Here is Structure of For Loop:

```
// Example: Printing numbers from 1 to 5 using a for loop
for (let i = 1; i <= 5; i++) {
    console.log(i);
}
```

Q.31 Write a print 972 to 897 using for loop in JS?

Ans.

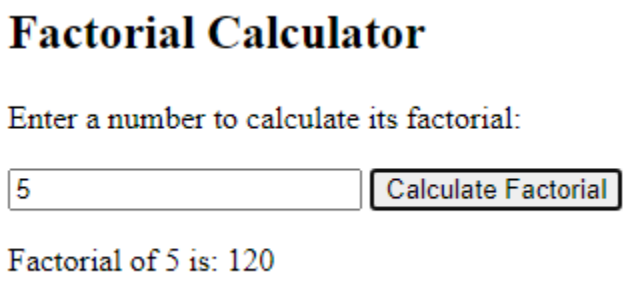
```
// Using a for loop to print numbers from 972 to 897
for (let i = 972; i >= 897; i--) {
    console.log(i);
}
```

Q.32 Write to print factorial of given number?

Ans.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Factorial Calculator</title>
    <script>
        // Function to calculate factorial
        function calculateFactorial() {
            // Get the input value from the HTML input field
            let userInput =
parseInt(document.getElementById("numberInput").value.trim());

            // Validate if userInput is a valid number
            if (!isNaN(userInput)) {
                // Check if the number is non-negative
                if (userInput < 0) {
                    alert("Factorial is not defined for negative numbers.");
                    return;
                }

                // Calculate factorial
                let factorial = 1;
                for (let i = userInput; i >= 1; i--) {
                    factorial *= i;
                }

                // Display the result
                document.getElementById("result").textContent = `Factorial of
${userInput} is: ${factorial}`;
            } else {
                alert("Invalid input. Please enter a valid number.");
            }
        }
    </script>
</head>
<body>
    <h2>Factorial Calculator</h2>
    <p>Enter a number to calculate its factorial:</p>
    <input type="text" id="numberInput" placeholder="Enter number...">
    <button onclick="calculateFactorial()">Calculate Factorial</button>
    <p id="result"></p>
</body>
</html>
```

Output:



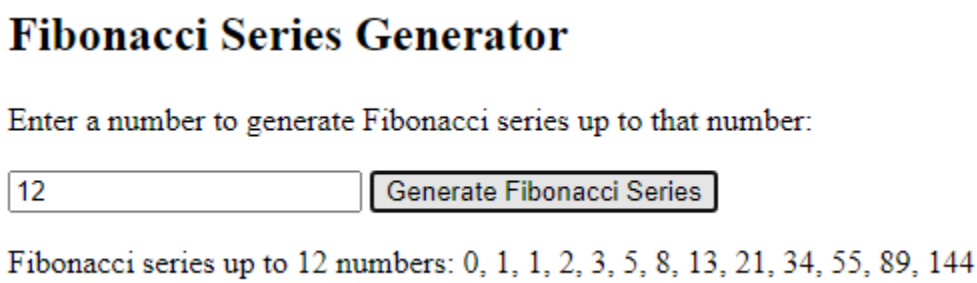
Q.33 Write to print Fibonacci series up to given numbers?

Ans.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Fibonacci Series</title>
    <script>
        // Function to generate and display Fibonacci series
        function generateFibonacci() {
            // Get the input value from the HTML input field
            let userInput =
parseInt(document.getElementById("numberInput").value.trim());

            // Validate if userInput is a valid number
            if (!isNaN(userInput) && userInput > 0) {
                // Initialize variables for Fibonacci sequence
                let fibSeq = [];
                let num1 = 0, num2 = 1, nextTerm;

                // First two terms of Fibonacci sequence
                fibSeq.push(num1);
                fibSeq.push(num2);

                // Generate Fibonacci series up to userInput
                for (let i = 2; i <= userInput; i++) {
                    nextTerm = num1 + num2;
                    fibSeq.push(nextTerm);
                    num1 = num2;
                    num2 = nextTerm;
                }

                // Display the Fibonacci series
```

```
                document.getElementById("result").textContent = `Fibonacci series
up to ${userInput} numbers: ${fibSeq.join(', ')}`;
            } else {
                alert("Invalid input. Please enter a valid positive number.");
            }
        }
    </script>
</head>
<body>
    <h2>Fibonacci Series Generator</h2>
    <p>Enter a number to generate Fibonacci series up to that number:</p>
    <input type="text" id="numberInput" placeholder="Enter number...">
    <button onclick="generateFibonacci()">Generate Fibonacci Series</button>
    <p id="result"></p>
</body>
</html>
```

Output:



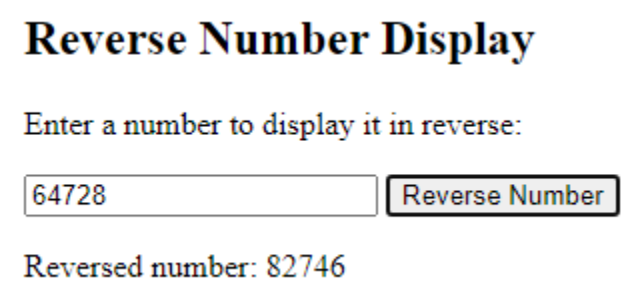
Q.34 Write to print number in reverse order e.g.: number = 64728 ---> reverse =82746 in JS?

Ans.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Reverse Number Display</title>
    <script>
        // Function to reverse and display the number
        function reverseNumber() {
            // Get the input value from the HTML input field
            let userInput = document.getElementById("numberInput").value.trim();

            // Validate if userInput is not empty and contains only digits
            if (userInput !== "" && /^\d+$/.test(userInput)) {
                // Convert the input number to a string and split into an array
of characters
```

```
                let digitsArray = userInput.split('');

                // Reverse the array of digits
                digitsArray.reverse();

                // Join the reversed array back into a string
                let reversedNumber = digitsArray.join('');

                // Display the reversed number
                document.getElementById("result").textContent = `Reversed number:
${reversedNumber}`;
            } else {
                alert("Invalid input. Please enter a valid number.");
            }
        }
    </script>
</head>
<body>
    <h2>Reverse Number Display</h2>
    <p>Enter a number to display it in reverse:</p>
    <input type="text" id="numberInput" placeholder="Enter number...">
    <button onclick="reverseNumber()">Reverse Number</button>
    <p id="result"></p>
</body>
</html>
```

Output:



Q.35 Write a program make a summation of given number (E.g., 1523 Ans: - 11) in JS?

Ans.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Sum of Digits</title>
    <script>
```

```
        // Function to calculate sum of digits
        function calculateSumOfDigits() {
            // Get the input value from the HTML input field
            let userInput = document.getElementById("numberInput").value.trim();

            // Validate if userInput is not empty and contains only digits
            if (userInput !== "" && /^\d+$/.test(userInput)) {
                // Convert the input number to a string and split into an array
of characters
                let digitsArray = userInput.split('');

                // Initialize sum of digits
                let sum = 0;

                // Calculate sum of digits
                for (let digit of digitsArray) {
                    sum += parseInt(digit);
                }

                // Display the result
                document.getElementById("result").textContent = `Sum of digits of
${userInput} is: ${sum}`;
            } else {
                alert("Invalid input. Please enter a valid number.");
            }
        }
    </script>
</head>
<body>
    <h2>Sum of Digits</h2>
    <p>Enter a number to calculate the sum of its digits:</p>
    <input type="text" id="numberInput" placeholder="Enter number...">
    <button onclick="calculateSumOfDigits()">Calculate Sum of Digits</button>
    <p id="result"></p>
</body>
</html>
```

Output:

Sum of Digits

Enter a number to calculate the sum of its digits:

59 Calculate Sum of Digits

Sum of digits of 59 is: 14

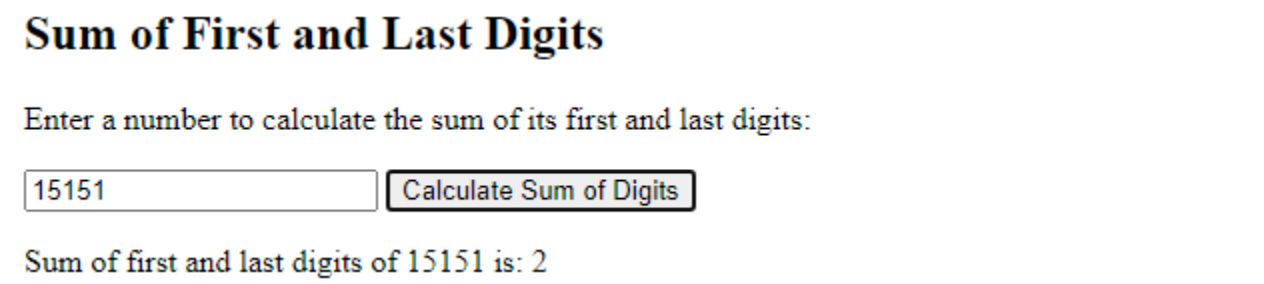
Q.36 Write a program you have to make a summation of first and last Digit. (E.g., 1234 Ans: - 5) in JS?

Ans.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Sum of First and Last Digits</title>
    <script>
        // Function to calculate sum of first and last digits
        function calculateSumOfFirstAndLastDigits() {
            // Get the input value from the HTML input field
            let userInput = document.getElementById("numberInput").value.trim();

            // Validate if userInput is not empty and contains only digits
            if (userInput !== "" && /^\d+$/.test(userInput)) {
                // Convert the input number to a string
                let numberStr = userInput.toString();

                // Extract first and last digits
                let firstDigit = parseInt(numberStr.charAt(0));
                let lastDigit = parseInt(numberStr.charAt(numberStr.length - 1));

                // Calculate sum of first and last digits
                let sum = firstDigit + lastDigit;

                // Display the result
                document.getElementById("result").textContent = `Sum of first and
last digits of ${userInput} is: ${sum}`;
            } else {
                alert("Invalid input. Please enter a valid number.");
            }
        }
    </script>
</head>
<body>
```

```
    <h2>Sum of First and Last Digits</h2>
    <p>Enter a number to calculate the sum of its first and last digits:</p>
    <input type="text" id="numberInput" placeholder="Enter number...">
    <button onclick="calculateSumOfFirstAndLastDigits()">Calculate Sum of
Digits</button>
    <p id="result"></p>
</body>
</html>
```

Output:



Q.37 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.

```
// Function to print the pattern
function printPattern() {
    // Iterate through each row from 1 to 5
    for (let i = 1; i <= 5; i++) {
        let row = `${i} `; // Start each row with the row number followed by a
space

        // Generate the rest of the pattern for the current row
        for (let j = 1; j <= 4; j++) {
            let value;
            if (j === 1) {
                value = 1; // First column: square of the row number
            } else {
                value = Math.pow(i, j); // Other columns: power of the row number
            }
            row += `${value} `;
```

```
        }

        // Output the row to the console
        console.log(row);
    }
}

// Call the function to print the pattern
printPattern();
```

Q.38 Use pattern in console.log in JS?

1)

1

1 0

1 0 1

1 0 1 0

1 0 1 0 1

Ans(1).

```
// Function to print the pattern
function printPattern() {
    // Iterate through each row from 1 to 5
    for (let i = 1; i <= 5; i++) {
        let row = ""; // Initialize an empty string for each row

        // Generate the pattern for the current row
        for (let j = 1; j <= i; j++) {
            if (j % 2 === 0) {
                row += "0 "; // Append '0 ' for even positions
            } else {
                row += "1 "; // Append '1 ' for odd positions
            }
        }

        // Output the row to the console
        console.log(row.trim());
    }
}

// Call the function to print the pattern
printPattern();
```

2)

A

B C

D E F

G H I J

K L M N O

Ans(2).

```
// Function to print the pattern
function printPattern() {
    // Iterate through each row from 1 to 5
    for (let i = 1; i <= 5; i++) {
        let row = ""; // Initialize an empty string for each row

        // Generate the pattern for the current row
        for (let j = 1; j <= i; j++) {
            if (j % 2 === 0) {
                row += "0 "; // Append '0 ' for even positions
            } else {
                row += "1 "; // Append '1 ' for odd positions
            }
        }

        // Output the row to the console
        console.log(row.trim());
    }
}

// Call the function to print the pattern
printPattern();
```

3)

1

2 3

4 5 6

7 8 9 10

11 12 13 14 15

Ans(3).

```
let num = 1;
for (let i = 1; i <= 5; i++) {
    let row = "";
    for (let j = 1; j <= i; j++) {
        row += num + " ";
        num++;
    }
    console.log(row);
}
```

4)

```
*

* *

* * *

* * * *

* * * * *
```

Ans(4).

```
for (let i = 1; i <= 5; i++) {
    let row = "";
    for (let j = 1; j <= i; j++) {
        row += "* ";
    }
    console.log(row);
}
```

Q.39 Accept 3 numbers from user using while loop and check each numbers palindrome?

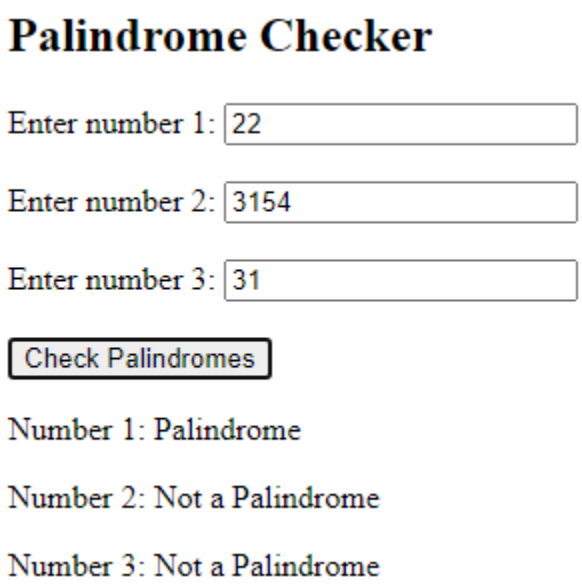
Ans.

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Palindrome Checker</title>
</head>
<body>
<h2>Palindrome Checker</h2>
```

```
<form id="palindromeForm">
    <label for="num1">Enter number 1:</label>
    <input type="number" id="num1" required><br><br>

    <label for="num2">Enter number 2:</label>
    <input type="number" id="num2" required><br><br>

    <label for="num3">Enter number 3:</label>
    <input type="number" id="num3" required><br><br>

    <button type="button" onclick="checkPalindromes()">Check Palindromes</button>
</form>

<div id="result"></div>

<script>
function checkPalindromes() {
    let num1 = document.getElementById('num1').value;
    let num2 = document.getElementById('num2').value;
    let num3 = document.getElementById('num3').value;

    let results = [];

    results.push(checkPalindrome(num1));
    results.push(checkPalindrome(num2));
    results.push(checkPalindrome(num3));

    let resultElement = document.getElementById('result');
    resultElement.innerHTML = '';

    for (let i = 0; i < results.length; i++) {
        let message = `Number ${i + 1}: ${results[i] ? 'Palindrome' : 'Not a
Palindrome'}`;
        let p = document.createElement('p');
        p.textContent = message;
        resultElement.appendChild(p);
    }
}

function checkPalindrome(number) {
    // Convert number to string to easily reverse
    let numString = number.toString();
    let reversedNumString = numString.split('').reverse().join('');

    // Compare original number with reversed number
```

```
    return numString === reversedNumString;
}
</script>

</body>
</html>
```

Output:



• What is negative Infinity?

Ans.

In JavaScript, `Negative Infinity` is a special value that represents the lowest possible numeric value, essentially the opposite of positive Infinity (`Infinity`). It is denoted by `-Infinity`.

• Which company developed JavaScript?

Ans.

JavaScript was developed by Netscape Communications Corporation, specifically by Brendan Eich. Brendan Eich created JavaScript in 1995 while he was working at Netscape.

• What are undeclared and undefined variables?

Ans.

Undeclared Variables:

An undeclared variable in JavaScript is one that has been used in code without being formally declared (i.e., without using `var`, `let`, or `const` keywords) within the current scope or any outer scope. When you try to reference an undeclared variable, JavaScript does not find any declaration for it in the scope chain.

Undefined Variables:

An undefined variable in JavaScript is one that has been declared but has not been assigned a value. When a variable is declared using `var`, `let`, or `const`, but no value is assigned to it, JavaScript assigns it a special value called `undefined`.

• Write the code for adding new elements dynamically?

Ans.

Let's assume we have a simple HTML structure with a button that will add new elements when clicked:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Add Elements Dynamically</title>
</head>
<body>
<div id="container">
    <!-- Existing content where new elements will be added -->
</div>
<button id="addElementBtn">Add New Element</button>

<script src="script.js"></script>
</body>
</html>
```

Here's how we can write the JavaScript code to add new elements dynamically when the button is clicked:

```
// Get a reference to the container where new elements will be added
const container = document.getElementById('container');

// Get a reference to the button that triggers adding new elements
const addElementBtn = document.getElementById('addElementBtn');

// Function to add new elements
function addNewElement() {
    // Create a new element (for example, a div)
```

```
    const newElement = document.createElement('div');

    // Set some content or attributes to the new element
    newElement.textContent = 'New Element Added';
    newElement.classList.add('new-element'); // Example: Adding a CSS class

    // Append the new element to the container
    container.appendChild(newElement);
}

// Event listener to call the addNewElement function when the button is clicked
addElementBtn.addEventListener('click', addNewElement);
```

• What is the difference between ViewState and SessionState?

Ans.

ViewState	SessionState
Maintained at page level only.	Maintained at session level.
View state can only be visible from a single page and not multiple pages.	Session state value availability is across all pages available in a user session.
It will retain values in the event of a postback operation occurring.	In session state, user data remains in the server. Data is available to user until the browser is closed or there is session expiration.
Information is stored on the client's end only.	Information is stored on the server.
used to allow the persistence of page-instance-specific data.	used for the persistence of user-specific data on the server's end.
ViewState values are lost/cleared when new page is loaded.	SessionState can be cleared by programmer or user or in case of timeouts.

• What is === operator?

Ans.

The === operator in JavaScript is known as the strict equality operator. It is used to compare two values for equality without performing type coercion.

Ex.

```
5 === 5;       // true
```

```
'hello' === 'hello';   // true
5 === '5';     // false (different types)
```

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

Ans.

In JavaScript, we can change the style or class of an HTML element dynamically using various methods provided by the Document Object Model (DOM).

1. Using `element.style.property`:

You can directly manipulate the inline styles of an element using the `style` property. This method is useful for setting or modifying individual style properties.

```
// Assuming you have an element with id="myElement"
let element = document.getElementById('myElement');

// Changing the background color
element.style.backgroundColor = 'blue';

// Changing the font size
element.style.fontSize = '20px';

// Adding more styles
element.style.padding = '10px';
```

• How to read and write a file using JavaScript?

Ans.

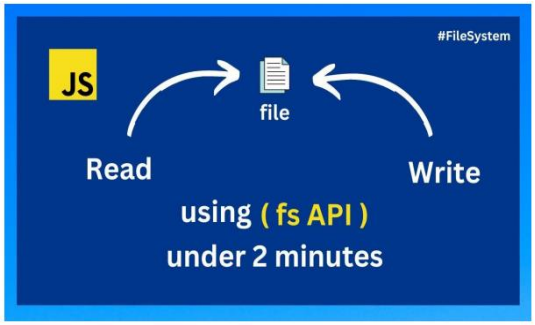
→ The fs.readFile() and rs.writeFile()

→ methods are used to read and write of a file using javascript.

→ The file is read using the fs.readFile() function, which is an inbuilt method.

→ This technique reads the full file into memory and stores it in a buffer.

Syntax :- fs.readFile(file_name, encoding, callback_function)



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

Ans.

Using `parseInt` Function:

JavaScript's `parseInt` function converts strings representing numbers in different bases (binary, octal, hexadecimal) to integers.

Syntax: `parseInt(string, radix)`

- `string`: The string to convert.
- `radix`: Optional. Specifies the base of the numeral system (2 for binary, 8 for octal, 16 for hexadecimal).

• What is the function of the delete operator?

Ans.

In JavaScript, the `delete` operator is used to remove a property from an object or to remove an element from an array.

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

Ans.

There are mainly 3 pop up boxes available in javascript:

- **`alert`**: Displays an alert message with an "OK" button.
- **`confirm`**: Displays a message with "OK" and "Cancel" buttons to confirm or cancel an action.
- **`prompt`**: Displays a message with an input field for the user to enter data, along with "OK" and "Cancel" buttons.

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

Ans.

In JavaScript, we can force a page to load another page by changing the `window.location` property to the URL of the new page. This technique triggers the browser to navigate to the specified URL, effectively loading a new page.

• What are the disadvantages of using innerHTML in JavaScript?

Ans.

→ It is very slow because as inner HTML already parses the content even we have to parse the content again so that's why it takes time.

→ When we have used the event handlers then the event handlers are not automatically attached to the new elements created by innerHTML.

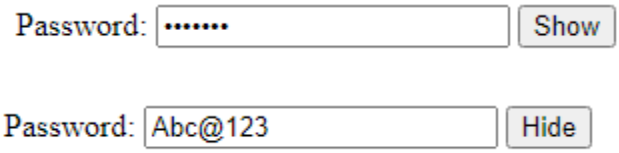
• Create password field with show hide functionalities

Ans.

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <title>Password Field with Show/Hide Functionality</title>
    <style>
        /* Optional: Styling for the container */
        .password-container {
            margin: 20px;
        }
    </style>
</head>

<body>

    <div class="password-container">
        <label for="password">Password:</label>
        <input type="password" id="password" name="password">
        <button id="togglePassword">Show</button>
    </div>

    <script>
        document.addEventListener('DOMContentLoaded', function () {
            const passwordInput = document.getElementById('password');
```

```
            const togglePasswordButton =
document.getElementById('togglePassword');

            togglePasswordButton.addEventListener('click', function () {
                // Toggle the type attribute
                if (passwordInput.type === 'password') {
                    passwordInput.type = 'text';
                    togglePasswordButton.textContent = 'Hide';
                } else {
                    passwordInput.type = 'password';
                    togglePasswordButton.textContent = 'Show';
                }
            });
        });
    </script>


</body>

</html>
```

Output:



• Create basic math operation in JS.

Ans.

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Basic Math Operations</title>
    <style>
        body {
            height: 100vh;
            width: 100vw;
            display: flex;
            justify-content: center;
```

```
            align-items: center;
            /* border: 2px solid black; */
        }

        .heading {
            font-size: 25px;
            /* border: 2px solid black; */
        }

        .calc {
            font-size: 20px;
            padding: 20px;
            height: 500px;
            width: 500px;
            border: 2px solid black;
        }
    </style>
</head>

<body>
    <table class="calc">
        <tr>
            <th class="heading" colspan="2">Maths Operations</th>
        </tr>
        <tr>
            <td>Enter 1st Number: </td>
            <td><input type="text" id="inp1"></td>
        </tr>
        <tr>
            <td>Enter 2nd Number: </td>
            <td><input type="text" id="inp2"></td>
        </tr>
        <tr>
            <td colspan="2">
                <button onclick="calculation('+')">+</button>
                <button onclick="calculation('-')">-</button>
                <button onclick="calculation('*')">*</button>
                <button onclick="calculation('/')">/</button>
                <button onclick="calculation('%')">%</button>
                <button onclick="resetbtn()">Reset</button>
            </td>
        </tr>
        <tr>
            <td colspan="2">Answer is : <span id="answer"></span></td>
        </tr>
```

```
    </table>

    <script>
        function calculation(operation) {
            const val1 = parseFloat(document.getElementById('inp1').value);
            const val2 = parseFloat(document.getElementById('inp2').value);
            let result;

            if (isNaN(val1) || isNaN(val2)) {
                answer = 'Please Enter a valid Input';
            } else {
                switch (operation) {
                    case '+':
                        answer = val1 + val2;
                        break;
                    case '-':
                        answer = val1 - val2;
                        break;
                    case '*':
                        answer = val1 * val2;
                        break;
                    case '/':
                        answer = val1 / val2;
                        break;
                    case '%':
                        answer = val1 % val2;
                        break;
                    default:
                        answer = 'Invalid Operation';
                }
            }

            document.getElementById('answer').textContent = answer;
        }

        function resetbtn() {
            document.getElementById('inp1').value = '';
            document.getElementById('inp2').value = '';
            document.getElementById('answer').textContent = '';
        }
    </script>
</body>

</html>
```

• Create result.

Ans.

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <style>
        body {
            height: auto;
            width: 50vw;
            display: flex;
            flex-direction: column;
            justify-content: center;
            align-items: center;
            border: 2px solid black;
        }

        td {
            padding: 8px;
            text-align: left;
            /* border: 2px solid red; */
        }

        .tbl {
            /* border: 1px solid blue; */
        }

        button {
            padding: 5px 20px;
        }
    </style>
</head>

<body>

    <h1>Marksheet for Information Technology</h1>
    <h3>Enter Marks</h3>
    <table class="tbl">
        <tr>
            <td>1. C Language</td>
```

```
        <td><input type="text" id="clng" /></td>
    </tr>
    <tr>
        <td>2. C++ Language</td>
        <td><input type="text" id="cpplng" /></td>
    </tr>
    <tr>
        <td>3. Database</td>
        <td><input type="text" id="database" /></td>
    </tr>
    <tr>
        <td>4. HTML</td>
        <td><input type="text" id="html" /></td>
    </tr>
    <tr>
        <td>5. CSS</td>
        <td><input type="text" id="css" /></td>
    </tr>
    <tr>
        <td>6. PHP</td>
        <td><input type="text" id="php" /></td>
    </tr>

    <td>7. Core Java</td>
    <td><input type="text" id="coreJava" /></td>
    </tr>
    <tr>
        <td></td>
        <td colspan="2">
            <button onclick="calculateResults()">Result</button>
        </td>
    </tr>
    <tr>
        <td>Total is:</td>
        <td>
            <h3 id="total">0</h3>
        </td>
    </tr>
    <tr>
        <td>Percentage is:</td>
        <td>
            <h3 id="percentage">0%</h3>
        </td>
    </tr>
```

```
    </table>

</body>
<script>

    function calculateResults() {
        const subjects = ['clng', 'cpplng',
            'database', 'html', 'css', 'php', 'coreJava'];

        let total = 0;
        let isValid = true;
        for (let i = 0; i < subjects.length; i++) {
            const mark =
                parseFloat(document.getElementById(subjects[i]).value);
            if (isNaN(mark) || mark < 0 || mark > 50) {
                alert("Please enter valid marks between 0and 50 for all
subjects.");

                isValid = false;
                break;
            }
            total += mark;
        }
        if (!isValid) return;
        const percentage = (total / (subjects.length * 50)) * 100;
        document.getElementById('total').textContent
            = total;

        document.getElementById('percentage').textContent = percentage.toFixed(2)
+ '%';
    }

</script>

</html>
```