**Batch:C2 Roll No.:16010122257**

**Experiment No. 04**

|  |
| --- |
| **TITLE:** **Develop and demonstrate JavaScript with POP-UP boxes and functions** |

**AIM:** To demonstrate the functionalities of JavaScript using HTML and CSS

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Expected Outcome of Experiment:** Design static web pages using various HTML tags.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Books/ Journals/ Websites referred:**

1. <https://www.freecodecamp.org/learn/javascript-algorithms-and-data-structures-v8/.>
2. <https://www.w3schools.com/js/>
3. <https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/JavaScript_basics>

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Describe and utilize Javascript programming concepts such as variables, arrays, conditionals, and loops.

Write and deploy Javascript code to solve practical web design problems.

**Problem Statement: Description of the application implemented with output**:

Online e-commerce web site.

Name Validation Problem Statement:

* Write JavaScript code to validate the user’s input for the name field.
* The input should contain only alphabets (no special characters or numbers).
* The length of the name should be at least 6 characters.

Password Validation Problem Statement:

* Write JavaScript code to validate the user’s input for the password field.
* The password should not be less than 6 characters in length.

E-mail ID Validation Problem Statement:

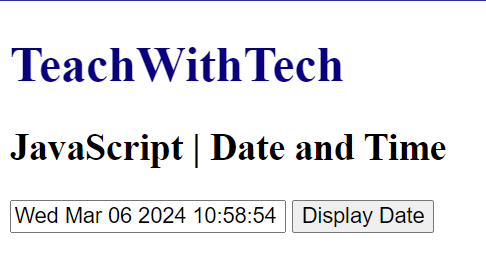
* Write JavaScript code to validate the user’s input for the email field.
* The email should follow the standard pattern name@domain.com.

Phone Number Validation Problem Statement:

* Write JavaScript code to validate the user’s input for the phone number field.
* The phone number should contain exactly 10 digits.

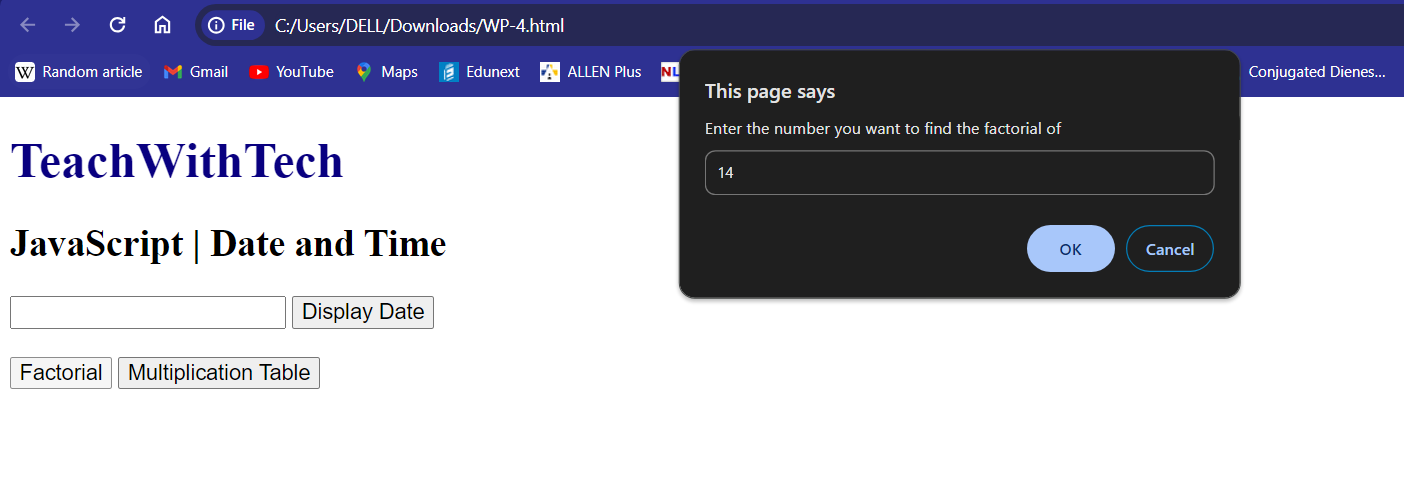
**a) Input**: Click on Display Date button using onclick( ) function

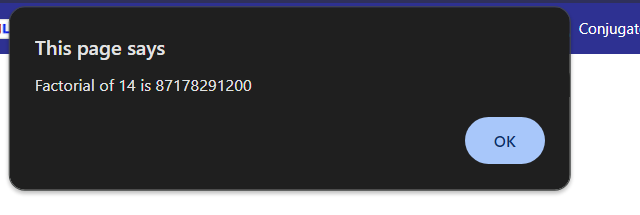
**Output:** Display date in the textbox



**b) Input:** A number n obtained using prompt

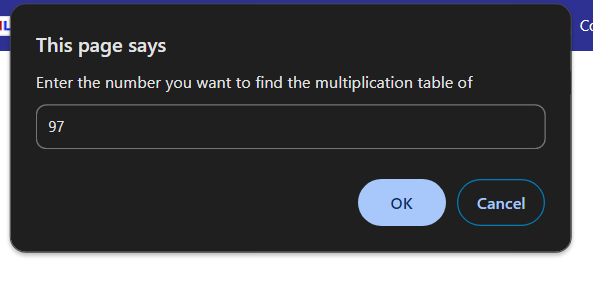
**Output:** Factorial of n number using alert

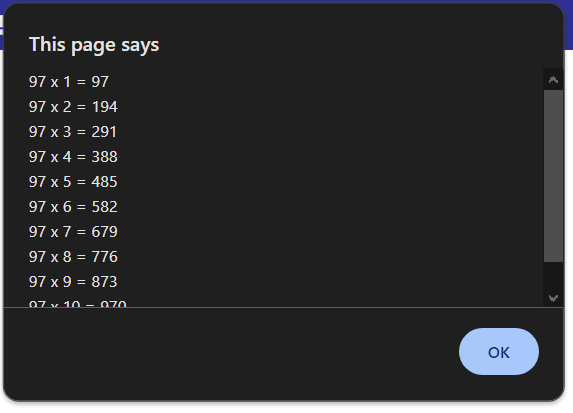




**c) Input:** A number n obtained using prompt

**Output:** A multiplication table of numbers from 1 to 10 of n using





d**) Write JavaScript to validate the following fields for the registration page**.

Name (Name should contain alphabets and the length should not be less than 6 characters).

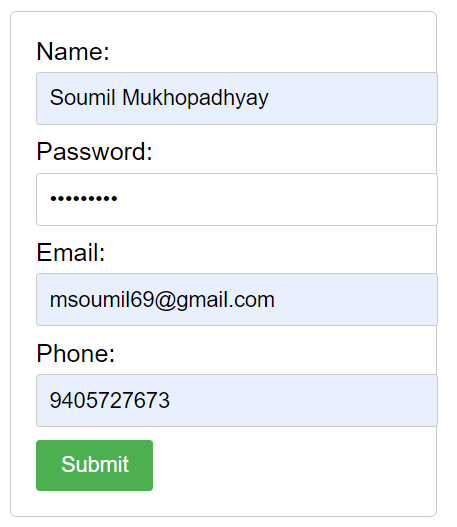
Password (Password should not be less than 6 characters length).

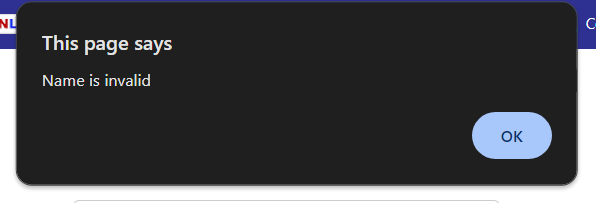
E-mail id (should not contain any invalid and must follow the standard pattern

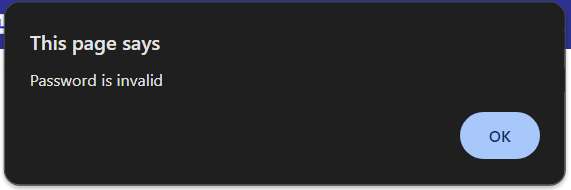
name@domain.com)

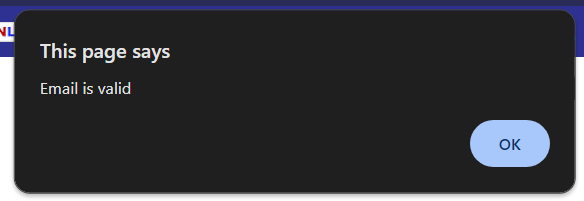
Phone number (Phone number should contain 10 digits only).

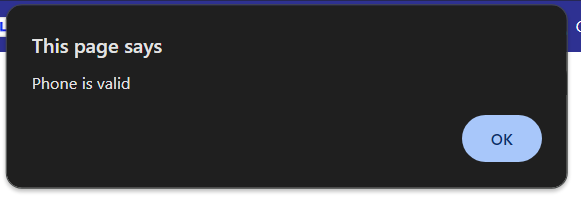
*OUTPUT(S):*







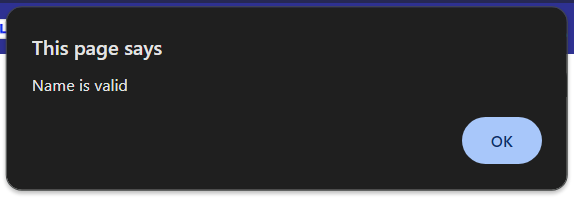


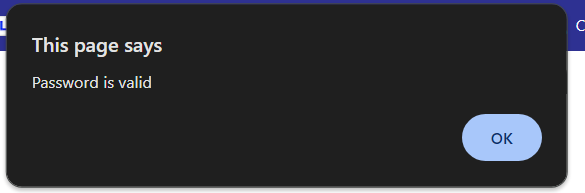


Name and Password were showing as invalid because of not satisfying all conditions

(It was only accepting a single word,no blank spaces were accepted;there was a special character in password).

After corrections were made(removed the special character):





**CODE:**

<!DOCTYPE html>

<html>

    <head>

        <title>

            JavaScript | Date and Time

        </title>

    </head>

    <body>

        <h1 style="color: rgb(9, 0, 128);">

            TeachWithTech

        </h1>

        <h2>

            JavaScript | Date and Time

        </h2>

        <form>

            <input type="text" id="date">

            <input type="button" value="Display Date"

                onclick="displayDate()">

        </form>

        <br>

        <button onclick="factorial()">

            Factorial

        </button>

        <button onclick="multiplicationTable()">

            Multiplication Table

        </button>

        <!DOCTYPE html>

<html>

<head>

    <title>Form Validation</title>

    <link rel="stylesheet" type="text/css" href="style.css">

</head>

<body>

    <form id="registrationForm">

        <label for="name">Name:</label>

        <input type="text" id="name" name="name" required>

        <br>

        <label for="password">Password:</label>

        <input type="password" id="password" name="password" required>

        <br>

        <label for="email">Email:</label>

        <input type="email" id="email" name="email" required>

        <br>

        <label for="phone">Phone:</label>

        <input type="tel" id="phone" name="phone" required>

        <br>

        <input type="submit" value="Submit" onclick="validateForm(); return false;">

    </form>

    <style>

        body {

    font-family: Arial, sans-serif;

}

form {

    width: 300px;

    margin: 0 auto;

    padding: 20px;

    border: 1px solid #ccc;

    border-radius: 5px;

}

label {

    display: block;

    margin-bottom: 5px;

}

input[type="text"], input[type="password"], input[type="email"], input[type="tel"] {

    width: 100%;

    padding: 10px;

    margin-bottom: 10px;

    border: 1px solid #ccc;

    border-radius: 3px;

}

input[type="submit"] {

    background-color: #4CAF50;

    color: white;

    padding: 10px 20px;

    border: none;

    border-radius: 3px;

    cursor: pointer;

}

input[type="submit"]:hover {

    background-color: #45a049;

}

    </style>

        <script>

            function displayDate(){

                var date = new Date();

                document.getElementById("date").value = date;

            }

            function factorial(){

                var num = prompt("Enter the number you want to find the factorial of");

                var fact = 1;

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

                    fact \*=i;

                }

                alert("Factorial of " + num + " is " + fact);

            }

            function multiplicationTable(){

                var num = prompt("Enter the number you want to find the multiplication table of ");

                var table = "";

                for(var i = 1; i <= 10; i++){

                    table += num + " x " + i + " = " + (num \* i) + "\n";

                }

                alert(table);

            }

            function validateForm(){

                var name = document.getElementById("name").value;

                var password = document.getElementById("password").value;

                var email = document.getElementById("email").value;

                var phone = document.getElementById("phone").value;

                var namePattern = /^[a-zA-Z]{6,}$/;

                var passwordPattern = /^[a-zA-Z0-9]{6,}$/;

                var emailPattern = /^[a-zA-Z0-9.\_-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,4}$/;

                var phonePattern = /^[0-9]{10}$/;

                if(namePattern.test(name)){

                    alert("Name is valid");

                }else{

                    alert("Name is invalid");

                }

                if(passwordPattern.test(password)){

                    alert("Password is valid");

                }else{

                    alert("Password is invalid");

                }

                if(emailPattern.test(email)){

                    alert("Email is valid");

                }else{

                    alert("Email is invalid");

                }

                if(phonePattern.test(phone)){

                    alert("Phone is valid");

                }else{

                    alert("Phone is invalid");

                }

            }

            </script>

    </body>

</html>

**Javascript Basic Concepts Learned With Syntax**

JavaScript code is typically incorporated into a webpage using the <script>...</script> tags. While these tags can be positioned anywhere in the HTML document, it is generally advisable to place them within the <head> tags. The <script> tag serves as a signal to the browser to commence interpreting the enclosed text as a JavaScript script.

* Single Line Comment

It is represented by double forward slashes (//). It can be used

before and after the statement.

* Multiline Comment

It can be used to add single as well as multi line comments. So,

it is more convenient.

/\* your code \*/

* JavaScript is case sensitive
* getElementById().

This example uses the method to "find" an HTML element (with id="demo") and changes the element content (innerHTML) to "Hello JavaScript":

Example

document.getElementById("demo").innerHTML = "Hello JavaScript";

* Used for variable declarations:

let , const

Redundant: var

* length property of an array returns the length of an array(the number of array elements)

Example:var array = {“name”, “class” , “section”};

array.length;

Output: 3

* Add a new element to an array using push method.

Example: array.push(“roll\_no”);

console.log(array);

Output: “name, “class” , “section” , “roll\_no”

* console.log( ) is used to print out something to the console/output.

Example:console.log(“hello world!”);

* Loops can execute a block of code as long as a specified condition is true.

while loop loops through a block of code as long as a specified condition is true

Syntax

while (condition)

{

code block to be executed

}

* Do-while loop will execute the code block once, before checking if the condition is true, then it will repeat the loop as long as the condition is true

do

{

code block to be executed

} while (condition);

* The for loop is often the tool you will use when you want to create a loop

Syntax

for (statement 1; statement 2; statement 3) {

code block to be executed

}

Statement 1 is executed before the loop (the code block) starts.

Statement 2 defines the condition for running the loop (the

code block).

Statement 3 is executed each time after the loop (the code

block) has been executed.

Use the switch statement to select one of many blocks of code to

be executed.

case n:

code block

break;

case n:

code block

break;

default:

code block

The switch expression is evaluated once.

The value of the expression is compared with the values of each case.

If there is a match, the associated block of code is executed.

* A JavaScript function is a block of code designed to perform a

particular task.

A JavaScript function is executed when "something" invokes it

(calls it).

Example

function myFunction(p1, p2) {

return p1 \* p2; // The function returns the product of p1 & p2

}

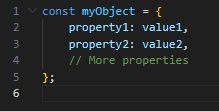
**Post Lab Objective with Ans :**

What are the possible ways to create objects in JavaScript?

**There are different ways to create new objects:**

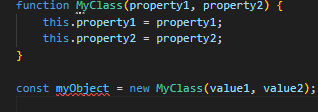
* **Create a single object, using an object literal.**

**This is the most straightforward and commonly used method. You define a new object by simply listing its properties and values within curly braces {}. For example:**



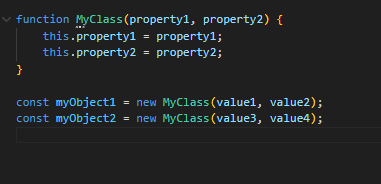
* **Create a single object, with the keyword new.**

**You can use the ‘new’ keyword along with a constructor function to create a new instance of an object. Constructor functions are regular functions but are typically named with an uppercase first letter to distinguish them from regular functions. Inside the constructor, you use the this keyword to refer to the newly created object. For example:**



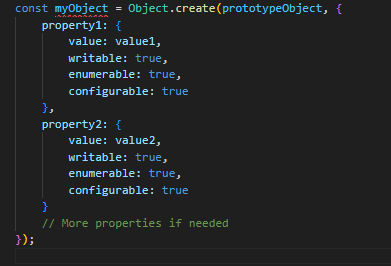
* **Define an object constructor, and then create objects of the constructed type.**

**This method is similar to the previous one but separates the definition of the constructor function from the instantiation of objects. First, you define the constructor function, and then you create new objects using the new keyword and the constructor function. For example:**



* **Create an object using Object.create().**

**This method allows you to create a new object with a specified prototype object and optional properties. It gives you more control over the prototype chain of the newly created object. For example:**



What is the Difference between == and === operators

**Equality (==):**

* **The == converts the operands to the same type,if they are not of, before making the comparison.**
* **If the two values have different types, JavaScript will attempt to convert one or both values to a common type, and then compare the values.**
* **For example:**



**Strict Equality (===):**

* **The === operator, also known as strict equality, does not perform type coercion. It requires both the value and the type to be the same for the comparison to be true.**
* **If the types are different, the comparison returns false without attempting to convert the values.**

**Example:**



What is the difference between let and var?

**In JavaScript, `let` and `var` are both used for variable declaration, but they have some important differences:**

**1. Scope:**

**- Variables declared with `var` are function-scoped. This means that their scope is limited to the function in which they are defined, or they are globally scoped if declared outside of any function.**

**- Variables declared with `let` are block-scoped. This means that their scope is limited to the block (enclosed by curly braces `{}`) in which they are defined, including `if`, `for`, `while`, and other block statements.**

**2. Hoisting:**

**- Variables declared with `var` are hoisted to the top of their containing function or global scope. This means that you can use the variable before it's declared without raising an error, although its value will be `undefined`.**

**- Variables declared with `let` are also hoisted, but they are not initialized. If you try to use a variable declared with `let` before its declaration, you'll encounter a ReferenceError.**

**3. Re-declaration:**

**- Variables declared with `var` can be re-declared within the same scope without raising an error. This can lead to potential issues and bugs in your code.**

**- Variables declared with `let` cannot be re-declared within the same block scope. Attempting to do so will result in a SyntaxError.**

**4. Global Object Property:**

**- Variables declared with `var` become properties of the global object (`window` in browsers, `global` in Node.js).**

**- Variables declared with `let` do not become properties of the global object.**

**In summary, `let` is generally preferred over `var` for variable declaration in modern JavaScript code because it provides better scoping behavior, helps avoid common bugs related to hoisting and re-declaration, and does not pollute the global object. However, `var` may still be used in specific scenarios where function scope or hoisting behavior is desired.**