K. J. Somaiya College of Engineering, Mumbai-77
(A Constituent College of Somaiya Vidyavihar University)

Batch: A1 **Roll No.: 16010123012**

Experiment No. 03

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://developer.mozilla.org/en-US/docs/Web/JavaScript

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:

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).

Javascript Basic Concepts Learned with Syntax
Handling User Input and Events - onclick() event to trigger JavaScript functions**,** prompt() to take user input, alert() to show outputs

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-
scale=1.0" />
    <title>JavaScript Basic Concepts</title>
```

K. J. Somaiya College of Engineering, Mumbai-77
(A Constituent College of Somaiya Vidyavihar University)

```
</head>
<body>
  <h1>First JavaScript</h1>
  <script>
    document.write("<hr>");
    document.write("Hello World!");
    document.write("<hr>");

    // 1. Variables and Data Types
    var name = "Aaryan";
    let age = 20;
    const PI = 3.14;
    console.log(name, age, PI);
    console.log(typeof name, typeof age, typeof PI);
    alert("Hello " + name);
    console.log("Hello " + name);
    console.log(typeof name);
    console.log(typeof age);
    console.log(typeof document);

    // 2. Operators
    n1 = 10;
    n2 = 20;
    n3 = "30";
    console.log(n1 + n2);
    console.log(n1 - n2);
    console.log(n1 * n2);
    console.log(n1 / n2);
    console.log(n1 % n2);
    console.log(n1 > n2);
    console.log(n1 == n2);
    console.log(n1 + n3);
    console.log("Square of " + n1 + " is " + n1 * n1);
    console.log(`Square of ${n1} is ${n1 * n1}`);

    // 3. Conditional Statements
    let num = 10;
    if (num > 0) {
      console.log("Positive Number");
    } else {
      console.log("Negative Number");
    }

    // 4. Loops
    for (i = 1; i <= 10; i++) {
      console.log(i);
    }
    for (i = 1; i <= 10; i++) {
      rem = i % 2;
```

```
        if (rem == 0) {
          console.log("Value of i is " + i + " and it is even");
        }
      }

      // 5. Functions
      function greet(name) {
        return "Hello, " + name + "!";
      }
      console.log(greet("Aaryan"));

      // 6. Arrays
      users = ["Aaryan", "Rahul", "Rohit", "Raj", "Ravi"];
      console.log(users);
      console.log(users[0]);
      console.log(users[1]);
      console.log(users[2]);
      console.log(users[3]);
      console.log(users[4]);
      users = ["Aaryan", 20, "Rahul", 21, "Rohit", 22, "Raj", 23,
"Ravi", 24];
      console.log(users);
      console.log(typeof users);

      record = [10, 30, 50, 20, 40];
      console.log(record);
      record.sort((a, b) => b - a);
      console.log("Sorted record:", record);

      // 7. Regular Expressions for Validation
      const emailPattern = /^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-
Z]{2,}$/;
      console.log(emailPattern.test("user@example.com"));

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

Description of the application implemented with output:

1. Displaying the Current Date

A button labeled "Display Date" triggers a JavaScript function using onclick(). The function retrieves the current date and displays it inside a readonly text box. When the user clicks the button, the date appears in the textbox.

2. Factorial Calculation

A button labeled "Calculate Factorial" prompts the user to enter a number. The function computes the factorial of the number and displays it using an alert().

3. Multiplication Table

A button labeled "Multiplication Table" prompts the user for a number. It generates a multiplication table (1 to 10) for the given number. The result is displayed in an alert() box.

4. Registration Form with Validation

Users enter Name, Password, Email, and Phone Number in the form. JavaScript validates the input, Name must contain only alphabets and be at least 6 characters long, Password must be at least 6 characters long, Email must follow the standard email format, Phone number must contain exactly 10 digits.
If the input is invalid, an error message appears below the respective field.
If all inputs are correct, an alert box confirms the form submission. If the user enters invalid data, error messages appear. If the input is valid, an alert box displays, Form submitted successfully!

Code:

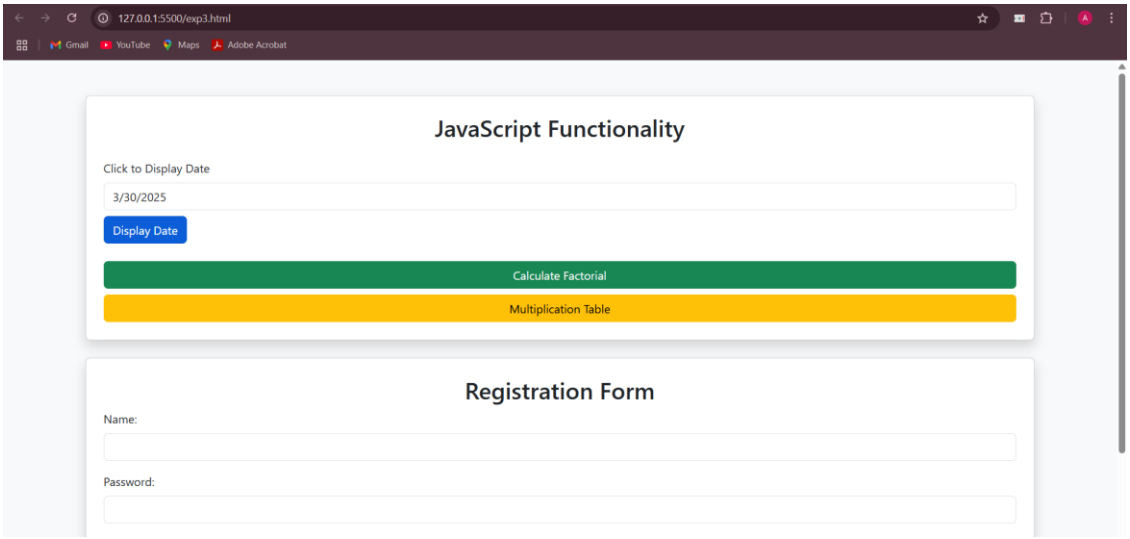
```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-
scale=1.0" />
    <title>JavaScript</title>
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.mi
n.css" rel="stylesheet"/>
  </head>
  <body class="bg-light">
    <div class="container mt-5">
      <div class="card shadow p-4">
        <h2 class="text-center mb-4">JavaScript Functionality</h2>

        <div class="mb-3">
          <label class="form-label">Click to Display Date</label>
          <input type="text" id="dateBox" class="form-control" readonly
/>
          <button class="btn btn-primary mt-2" onclick="displayDate()">
            Display Date
          </button>
        </div>

        <button class="btn btn-success mt-2"
onclick="calculateFactorial()">
          Calculate Factorial
        </button>
        <button class="btn btn-warning mt-2"
onclick="showMultiplicationTable()">
```

```
          Multiplication Table
        </button>
      </div>

      <div class="card shadow p-4 mt-4">
        <h2 class="text-center">Registration Form</h2>
        <form onsubmit="return validateForm()">
          <div class="mb-3">
            <label class="form-label">Name:</label>
            <input type="text" id="name" class="form-control" />
            <div id="nameError" class="text-danger"></div>
          </div>

          <div class="mb-3">
            <label class="form-label">Password:</label>
            <input type="password" id="password" class="form-control" />
            <div id="passwordError" class="text-danger"></div>
          </div>

          <div class="mb-3">
            <label class="form-label">Email:</label>
            <input type="email" id="email" class="form-control" />
            <div id="emailError" class="text-danger"></div>
          </div>

          <div class="mb-3">
            <label class="form-label">Phone Number:</label>
            <input type="text" id="phone" class="form-control" />
            <div id="phoneError" class="text-danger"></div>
          </div>

          <button type="submit" class="btn btn-primary w-
100">Register</button>
        </form>
      </div>
    </div>

    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bund
le.min.js"></script>
    <script>
      function displayDate() {
        document.getElementById("dateBox").value = new
Date().toLocaleDateString();
      }

      function calculateFactorial() {
        let n = prompt("Enter a number:");
        n = parseInt(n);
```

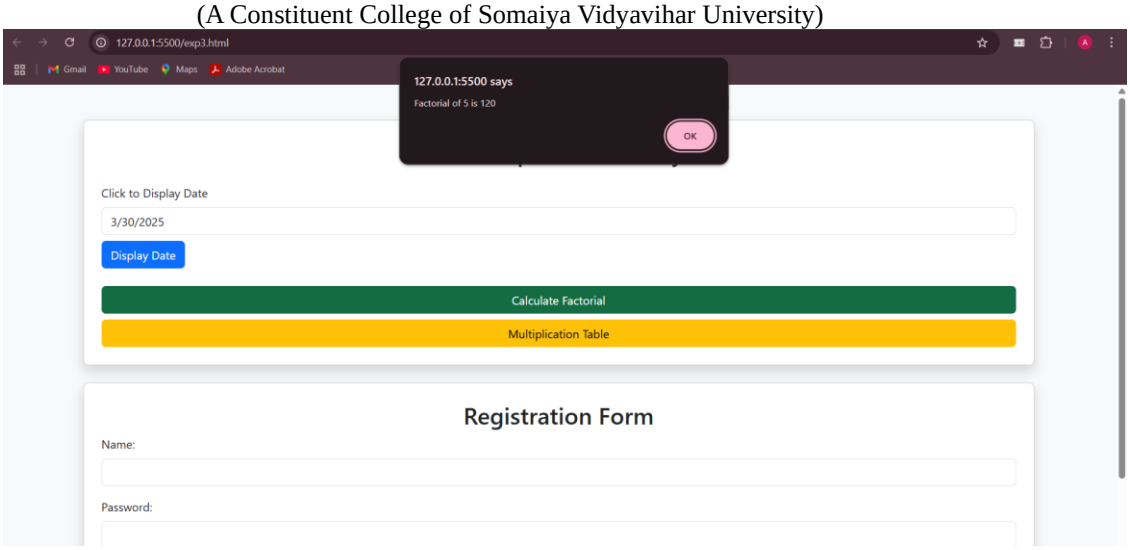
```
        if (isNaN(n) || n < 0) {
          alert("Please enter a valid positive number");
          return;
        }

        let factorial = 1;
        for (let i = 1; i <= n; i++) {
          factorial *= i;
        }
        alert(`Factorial of ${n} is ${factorial}`);
      }

      function showMultiplicationTable() {
        let n = prompt("Enter a number:");
        n = parseInt(n);
        if (isNaN(n)) {
          alert("Please enter a valid number");
          return;
        }

        let table = `Multiplication Table of ${n}:\n`;
        for (let i = 1; i <= 10; i++) {
          table += `${n} * ${i} = ${n * i}\n`;
        }
        alert(table);
      }

      function validateForm() {
        let name = document.getElementById("name").value.trim();
        let password = document.getElementById("password").value.trim();
        let email = document.getElementById("email").value.trim();
        let phone = document.getElementById("phone").value.trim();

        let nameError = document.getElementById("nameError");
        let passwordError = document.getElementById("passwordError");
        let emailError = document.getElementById("emailError");
        let phoneError = document.getElementById("phoneError");

        nameError.textContent = passwordError.textContent =
emailError.textContent = phoneError.textContent = "";

        let isValid = true;
        if (!/^[A-Za-z]{6,}$/.test(name)) {
          nameError.textContent = "Name must contain only letters and be
at least 6 characters long";
          isValid = false;
        }

        if (password.length < 6) {
```

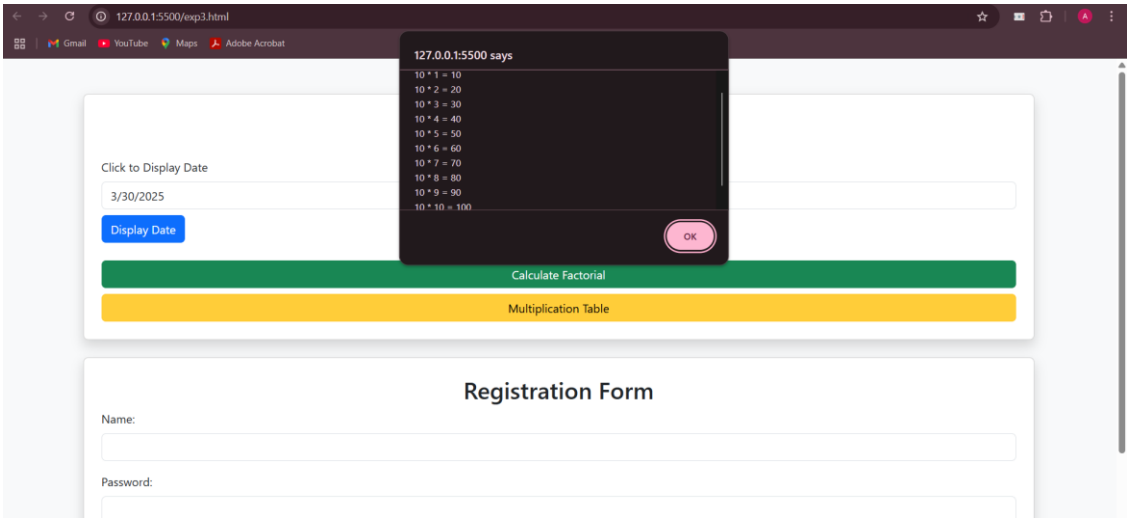
```
          passwordError.textContent = "Password must be at least 6
characters long";
          isValid = false;
        }

        let emailPattern = /^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-
Z]{2,}$/;
        if (!emailPattern.test(email)) {
          emailError.textContent = "Enter a valid email (e.g.,
aaryan@somaiya.edu)";
          isValid = false;
        }

        if (!/^\d{10}$/.test(phone)) {
          phoneError.textContent = "Phone number must contain exactly 10
digits";
          isValid = false;
        }

        if (isValid) {
          alert("Form submitted successfully!");
        } else {
          alert("Please correct the errors in the form.");
        }
        return false;
      }
    </script>
  </body>
</html>
```

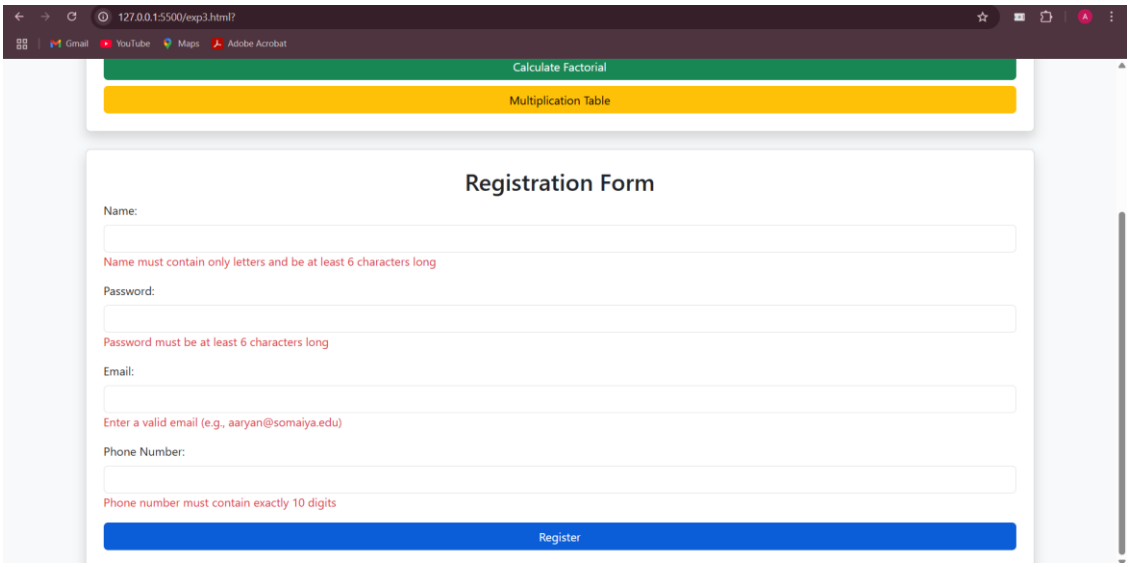
Output:

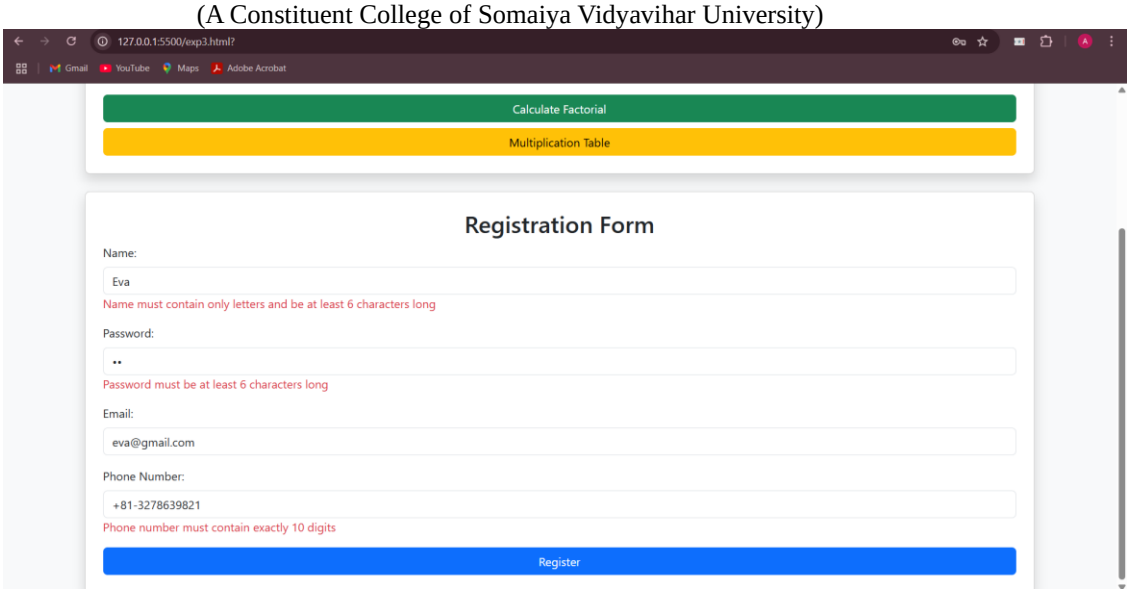


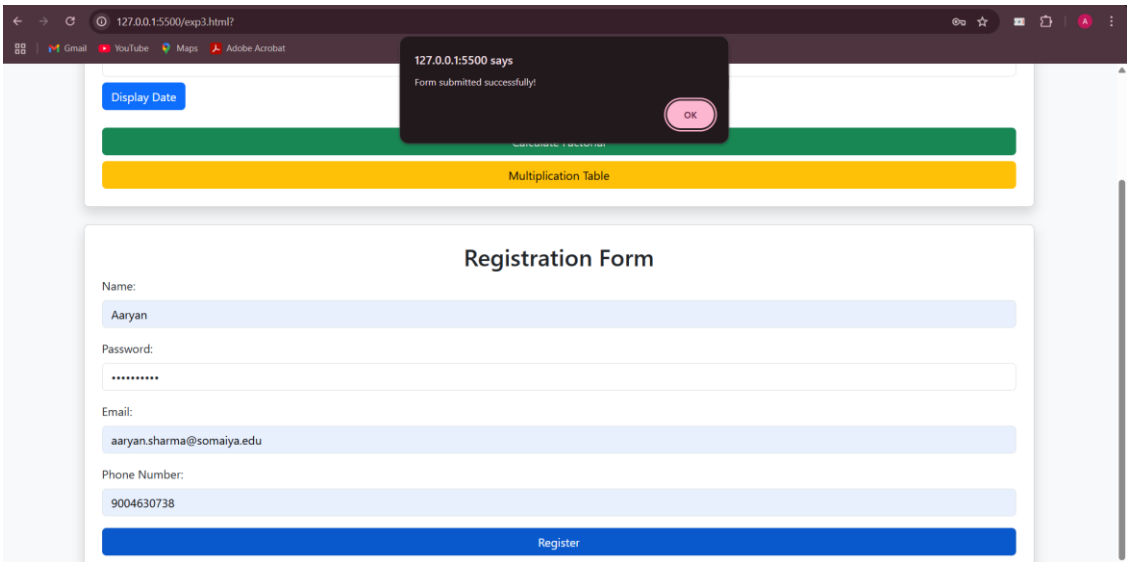
K. J. Somaiya College of Engineering, Mumbai-77
(A Constituent College of Somaiya Vidyavihar University)











Conclusion:

I have successfully implemented and demonstrated various JavaScript functionalities, including pop-up boxes, functions, event handling, and form validation. Through this experiment, I have gained a deeper understanding of JavaScript's core concepts, such as variables, arrays, loops, and conditional statements. Additionally, I have applied these concepts to practical web applications, including date display, factorial computation, multiplication tables, and user input validation.

Post Lab Objective:

1. What are the possible ways to create objects in JavaScript?

There are several ways to create objects in JavaScript, each with its own use cases. The simplest and most commonly used method is using object literals, where an object is

defined directly with key-value pairs, such as let person = { name: "Aaryan", age: 20 };. Another approach is using the new Object() method, where an empty object is created and properties are added separately, for example, let person = new Object(); person.name = "Aaryan"; person.age = 20;. A more structured way to create objects is using constructor functions, where a function defines properties, and new objects can be instantiated with new, such as function Person(name, age) { this.name = name; this.age = age; } let person1 = new Person("Aaryan", 20);. ES6 introduced classes, which provide a cleaner and more intuitive way to create objects, where the constructor method initializes properties, for example, class Person { constructor(name, age) { this.name = name; this.age = age; } } let person1 = new Person("Aaryan", 20);. Another method is using Object.create(), which allows creating an object from an existing prototype, such as let personPrototype = { greet: function () { console.log("Hello!"); } }; let person = Object.create(personPrototype); person.name = "Aaryan"; person.age = 20;. Lastly, JSON parsing can be used to create objects from JSON data, such as let person = JSON.parse('{"name": "Aaryan", "age": 20}');, which is particularly useful when handling data from APIs.

2. What is the Difference between == and === operators?

The == operator in JavaScript compares only the values of two variables, allowing type coercion, meaning JavaScript automatically converts one operand to match the type of the other before making the comparison. For example, "5" == 5 returns true because JavaScript converts the string "5" into a number before comparing it with 5. On the other hand, the === (strict equality) operator compares both the value and the data type, ensuring that no type conversion occurs. This means "5" === 5 returns false because one is a string and the other is a number, making them strictly unequal. Therefore, using === is generally recommended for more precise comparisons, as it avoids unintended type conversions that may lead to unexpected results.

3. What is the difference between let and var?

In JavaScript, the primary difference between var and let lies in their scope. The var keyword is function-scoped, meaning a variable declared with var is accessible throughout the function, even outside the block in which it was declared. In contrast, let is block-scoped, meaning it is only accessible within the block {} where it was defined. Additionally, var allows redeclaration within the same scope, whereas let does not permit redeclaring a variable in the same scope, preventing accidental overwrites. Another key distinction is in hoisting, variables declared with var are hoisted to the top of their scope and initialized with undefined, while let is also hoisted but remains uninitialized until it is explicitly assigned a value, leading to a "Temporal Dead Zone" if accessed before initialization. For example, in a function, if var x = 10; is declared inside an if block, it remains accessible outside the block, but let y = 10; inside the same block would throw an error if accessed outside. Due to these differences, it is recommended to use let instead of var for better scope control and to avoid potential bugs caused by unintended redeclarations or hoisting behaviors.