JavaScript_Day – 5_Hands_On -- Harshitha Kamatam

Problem 1

Problem Statement:

Greeting Generator (Level-1)

Scenario

A simple webpage should greet users based on the name they enter. This helps beginners understand how JavaScript functions accept parameters and how variables behave inside functions.

Requirements

- Create an input box to enter a user's name.
- Create a button labeled "Generate Greeting".
- When the button is clicked:
- A JavaScript function should accept the name as a parameter.
- Display a greeting message like:
  "Hello, Rahul! Welcome to our website."
- Display the greeting inside a <p> element.

Technical Constraints

- Use only vanilla JavaScript (no libraries).
- Use function keyword (not arrow functions for this task).
- Use document.getElementById() for DOM manipulation.
- Variable for the greeting message must be declared inside the function.

Learning Outcome

You should be able to:

- Understand how functions receive input using parameters.
- Learn local variable scope.
- Manipulate webpage elements using the DOM.

Source Code:

File Name: Index.html

```html
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Greeting Generator</title>
    <style>
        body {
            text-align: center;
            background-color: azure;
            margin: 10px;
        }
    </style>
</head>
<body>
    <h1>Greeting Generator</h1>
    <input type="text" id="username" placeholder="Enter your name">
    <button onclick="generateGreeting()">Generate Greeting</button>
    <p id="greeting"></p>

    <script>
        function generateGreeting() {
            var name = document.getElementById("username").value;
            var message = "Hello, " + name + "! Welcome to the Greeting Generator Website.";
            document.getElementById("greeting").innerText = message;
        }
    </script>
</body>
</html>
```

Output:

Output File: index.html

Explanation:

This code creates a simple **Greeting Generator**. When you enter your name and click the button, JavaScript takes the name from the input box and creates a personalized welcome message.

Problem 2

Problem Statement:

Simple Object-Based User Info Display (Level-1)

Scenario

A webpage needs to display basic user information stored inside a JavaScript object.

Requirements

Create a JavaScript object user with properties:

- name
- age
- city

Create a function displayUserInfo(userObj):

- Accepts the object as a parameter.
- Displays user details in separate <p> elements.

A button click should trigger the function.

Technical Constraints

- Object properties must be accessed using dot notation.
- Function should not use global variables.
- DOM elements should be updated dynamically.

Learning Outcome

You will be able to:

- Understand JavaScript objects.
- Pass objects to functions.
- Display object data dynamically on a webpage.

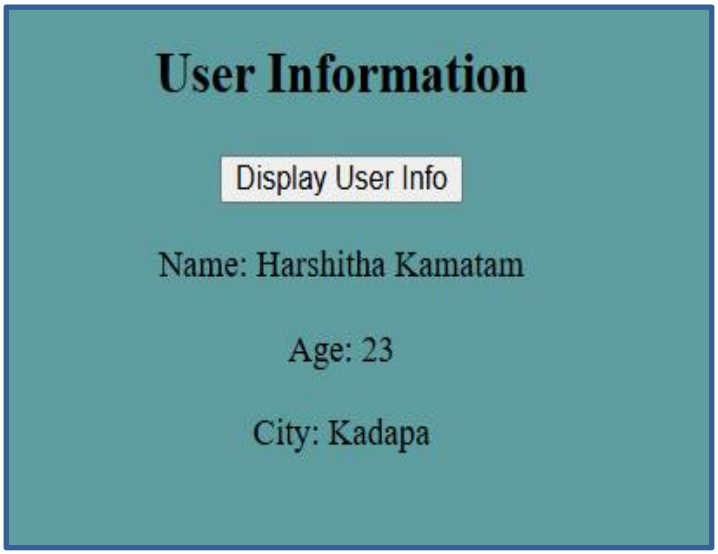
Source Code:

File Name: index.html

```html
<!--JS_Day5_Hands_on_Problem_Statement2_Harshitha Kamatam-->
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>User Info Display</title>
    <style>
        body {
            text-align: center;
            background-color: cadetblue;
        }
    </style>
</head>
<body>
    <h2>User Information</h2>
    <button onclick="showUser()">Display User Info</button>
    <p id="name"></p>
    <p id="age"></p>
    <p id="city"></p>

    <script>
        var user = {
            name: "Harshitha Kamatam",
            age: 23,
            city: "Kadapa"
        };
        function displayUserInfo(userObj) {
            document.getElementById("name").innerText = "Name: " + userObj.name;
            document.getElementById("age").innerText  = "Age: "  + userObj.age;
            document.getElementById("city").innerText = "City: " + userObj.city;
        }
        function showUser() {
            displayUserInfo(user);
        }
    </script>
</body>
</html>
```

Output:

Output File: index.html



Explanation:

This code creates a simple **User Information display** using a JavaScript object. When you click the "Display User Info" button, it reads the user object (name, age, city) and shows the details on the screen.

The output displays the user's name, age, and city dynamically below the button.

Problem 3

Problem Statement:

Counter App with Scope Control (Level-2)

Scenario

Build a counter application where users can increment and reset a value. This problem focuses on variable scope and DOM manipulation.

Requirements

Display a counter value starting from 0.

Create two buttons:

- **Increment**
- **Reset**

Use:

- A global variable to store counter value.
- A function incrementCounter(step) that:
    - Accepts step value as a parameter.
    - Updates the counter.

Reset button should reset the counter to 0.

Technical Constraints

- Counter value must be maintained outside the function (global scope).
- DOM updates must happen inside functions only.
- No inline JavaScript in HTML.

Learning Outcome

Learners should be able to:

- Understand variable scope (global vs local).
- Apply function parameters in real scenarios.
- Control UI behavior using JavaScript functions.

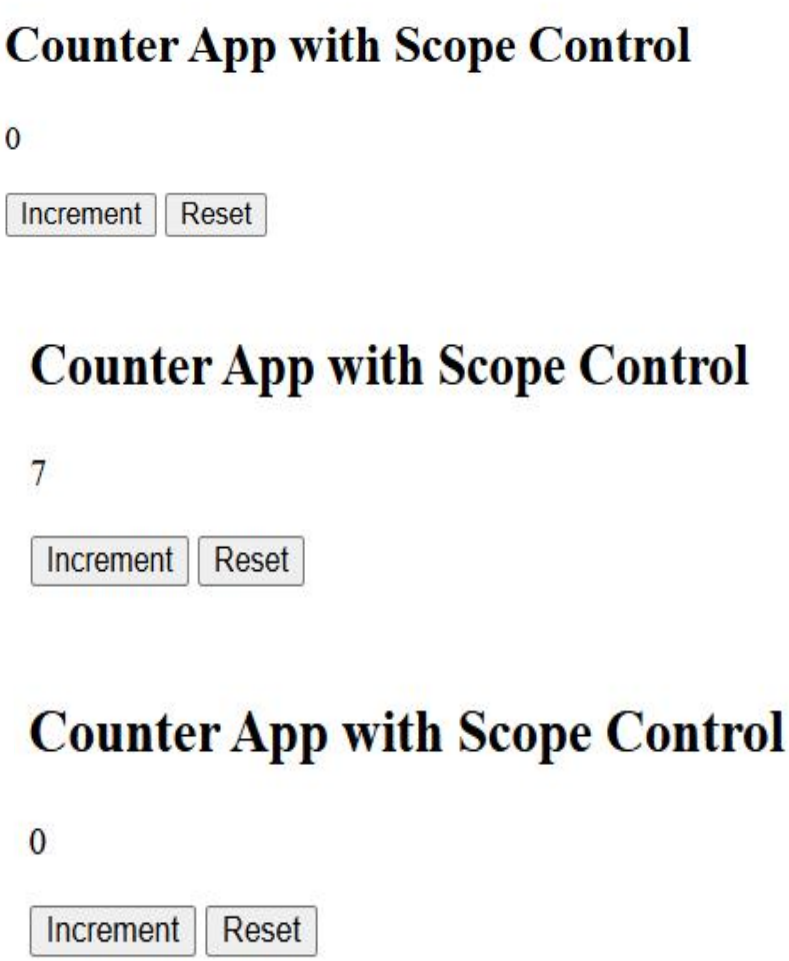
Source Code:

File Name: index.html

```html
<!--JS_Day5_Hands_on_Problem_Statement3_Harshitha Kamatam-->
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Counter App</title>
</head>
<body>
    <h2>Counter App with Scope Control</h2>
    <p id="counterValue">0</p>
    <button id="incrementBtn">Increment</button>
    <button id="resetBtn">Reset</button>

    <script>
        var counter = 0;
        function incrementCounter(step) {
            counter = counter + step;
            document.getElementById("counterValue").innerText = counter;
        }

        function resetCounter() {
            counter = 0;
            document.getElementById("counterValue").innerText = counter;
        }

        document.getElementById("incrementBtn").addEventListener("click", function() {
            incrementCounter(1);
        });

        document.getElementById("resetBtn").addEventListener("click", function() {
            resetCounter();
        });
    </script>
</body>
</html>
```

Output:

Output File: index.html



Explanation:

This code creates a simple **Counter App** using a global variable called counter. When you click **Increment**, the value increases by 1, and when you click **Reset**, it goes back to 0. The number shown on the screen updates instantly based on the button you press.

Problem 4

Problem Statement:

Dynamic Student Profile Manager (Level-2)

Scenario

Create a mini student profile system where details are stored in an object and displayed dynamically using DOM manipulation.

Requirements

Create a student object with:

- name
- rollNo
- marks

Create a function updateStudentProfile(studentObj):

- Accepts the object as a parameter.
- Displays details in a styled .

Add another function updateMarks(newMarks):

- Updates marks and refreshes the display.

Technical Constraints

- Student object must be declared in global scope.
- Functions should update object values using parameters.
- DOM should update without page refresh.

Learning Outcome

You will be able to:

- Deep understanding of object manipulation.
- Function interaction with shared data.
- Real-world use of DOM updates and scope handling.

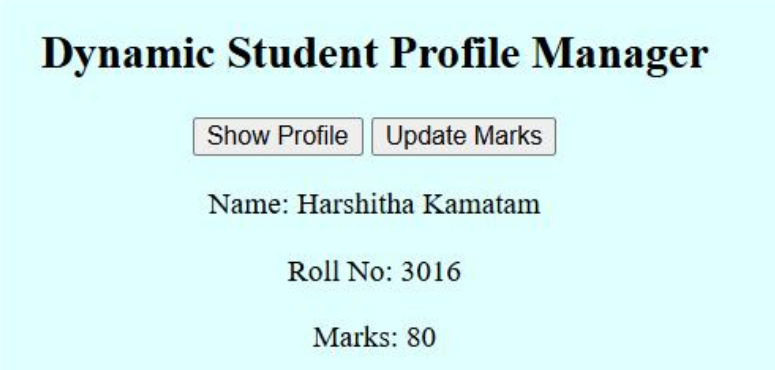
Source Code:

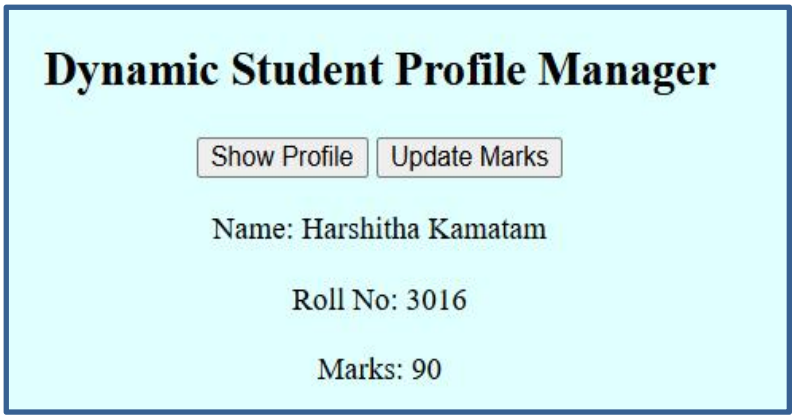
File Name: index.html

```html
<!--JS_Day5_Hands_on_Problem_Statement4_Harshitha Kamatam-->
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Student Profile</title>
    <style>
        body {
            text-align: center;
            background-color: lightcyan;
        }
    </style>
</head>
<body>
    <h2>Dynamic Student Profile Manager</h2>
    <button id="showBtn">Show Profile</button>
    <button id="updateBtn">Update Marks</button>
    <div id="profileBox"></div>

    <script>
        var student = {
            name: "Harshitha Kamatam",
            rollNo: 3016,
            marks: 80
        };

        function updateStudentProfile(studentObj) {
            document.getElementById("profileBox").innerHTML =
                "<p>Name: " + studentObj.name + "</p>" +
                "<p>Roll No: " + studentObj.rollNo + "</p>" +
                "<p>Marks: " + studentObj.marks + "</p>";
        }

        function updateMarks(newMarks) {
            student.marks = newMarks;
            updateStudentProfile(student);
        }
        document.getElementById("showBtn").addEventListener("click", function() {
            updateStudentProfile(student);
        });

        document.getElementById("updateBtn").addEventListener("click", function() {
            updateMarks(90);
        });
    </script>
</body>
</html>
```

Output:

Output File: index.html





Explanation:

This code creates a **Dynamic Student Profile Manager** using a JavaScript object to store student details (name, roll number, marks). When you click **Show Profile**, it displays the student info, and when you click **Update Marks**, it changes the marks value and updates it on the screen. The output first shows marks as 80, and after clicking Update Marks, it changes to 90 instantly.