Advanced Functions and Form Validation(16 hours)

Task 1: Advanced Function Operations

Requirements

- Create a function named `calculateSum` that accepts two parameters.
- Return the sum of the two parameters from the function.
- Create another function named `calculateProduct` that accepts two parameters.
- Return the product of the two parameters from the function.
- Call both functions with different values.
- Store the returned values in variables.
- Display the results using `console.log()` with clear labels.

Task 2: Function Expressions and Arrow Functions

Requirements

- Create a function expression that accepts a string and returns its length.
- Create an arrow function that accepts a number and returns its square.
- Call both functions with appropriate values.
- Store the returned results in variables.
- Display the outputs using `console.log()` with meaningful labels.

Task 3: Callback Function Implementation

Requirements

- Create a function named `processNumber` that accepts two parameters:
  - a number
  - a callback function
- Inside the function, pass the number to the callback and return the result.
- Create two callback functions:
  - one that doubles a number
  - one that halves a number
- Call `processNumber` using both callback functions.
- Display the results using `console.log()`.

Task 4: Function with Default Parameters

Requirements

- Create a function named `calculateDiscount` that accepts:
  - price
  - discount percentage (default value should be provided)
- Calculate the final price after discount.
- Return the calculated value.
- Call the function with and without passing the discount value.
- Log both results using `console.log()`.

Task 5: Immediately Invoked Function Expression (IIFE)

Requirements

- Create an IIFE that:
  - declares a variable inside the function
  - performs a mathematical operation
- Display the result using `console.log()`.
- Ensure the variable is not accessible outside the function.

Task 6: Form Validation – Input Presence Check

Requirements

- Create an HTML form with:
  - name input field
  - email input field
  - submit button
- Write JavaScript to validate the form on submission.
- Check if the name and email fields are empty.
- Display validation messages using `console.log()`.
- Prevent form submission if any field is empty.

Task 7: Form Validation – Email Format Check

Requirements

- Create an HTML form with an email input field.
- Write JavaScript code to validate the email format.
- Check whether the email contains `@` and `.` characters.
- Display a success or error message using `console.log()`.

- Prevent form submission if the email format is invalid.

Task 8: Form Validation – Password Length Check

Requirements

- Create an HTML form with a password input field.
- Write JavaScript to validate password length.
- Ensure the password has at least 8 characters.
- Display validation result using `console.log()`.
- Prevent form submission if the condition fails.

Task 9: Combined Form Validation Program

Requirements

- Create an HTML form with:
  - name
  - email
  - password
- Validate all fields using JavaScript.
- Check for empty values.
- Validate email format.
- Validate password length.
- Display validation messages using `console.log()`.

Task 10: Advanced Function Validation Utility

Requirements

- Create separate functions for:
  - checking empty input
  - validating email format
  - validating password length
- Use these functions inside a main validation function.
- Call the main function on form submission.
- Display validation results using `console.log()`.