Java Script Basic (LEVEL 5)

Session 1: Introduction to JavaScript and Basic Syntax

1. Introduction to JavaScript:
- Explanation: JavaScript is a programming language that allows you to implement complex features on web pages, making them interactive and dynamic. It's widely used for client-side development.

2. Linking JavaScript with HTML:
- How to Link: JavaScript can be linked to an HTML file either inline within `<script>` tags or as an external file using : `<script src="path/to/file.js"></script>` tag. It's generally a best practice to link JavaScript files externally for better organization and maintainability.

3. Output in Console:
- Usage of `console.log()`: This function is used to display messages in the browser's console, which is useful for debugging and testing.

4. Variables:
- Definition: Variables are containers for storing data values. JavaScript allows you to define variables using the `var`, `let`, or `const` keywords.

5. Variable Naming Conventions:
- Rules: Variable names can contain letters, digits, underscores, and dollar signs, but must begin with a letter. They are case-sensitive and should follow camelCase notation.

6. Variable Types:
- Difference: `let` is block-scoped and can be updated but not re-declared. `var` is function-scoped and can be updated and re-declared. `const` is block-scoped and cannot be updated or re-declared.

7. Comments:
- Types: Single-line comments use `//`, and multi-line comments use `/* ... */`.

8. Introduction to Data Types:
- Concept: Data types define the type of data a variable can hold, such as numbers, strings, and booleans.

9. Data Types Overview:
- Number: Represents numeric values.
- String: A sequence of characters.
- Null: Represents an empty or unknown value.
- NaN: Represents a "Not-a-Number" value.
- Boolean: Represents `true` or `false`.

10. Basic Operators:
- Operators: Basic mathematical operations include addition (`+`), subtraction (`-`), multiplication (`*`), division (`/`), and assignment (`=`).

Session 2: Working with Strings and Conditional Statements

1. Combining Variables in Output:

- Example: `console.log("The name is: " + name + ", age: " + age);` shows how to concatenate different data types in a print statement.

2. String Properties:

- String Length: Use `string.length` to find the length of a string.

3. Strings Concatenation:

- Method: Combining multiple strings using the `+` operator or `concat()` method.

4. Conditions:

- Usage: `if`, `else if`, and `else` statements allow you to execute different code blocks based on conditions.

5. Switch Case:

- Syntax: Used for executing one block of code among many options based on a variable's value.

6. Comparison Operators:

- Operators: Greater than (`>`), less than (`<`), greater than or equal to (`>=`), less than or equal to (`<=`), equal (`==`, `===`), not equal (`!=`, `!==`).

7. Brackets Notation:

- Usage: Curly braces `{}` are used to define code blocks for functions, loops, conditions, etc.

8. Login Validation Program:

- Implementation: Create a simple program that checks a username and password against predefined values.

Session 3: Introduction to Loops

1. Introduction to Loops:

- Concept: Loops allow you to execute a block of code multiple times.

2. For Loops:

- Syntax: `for(initialization; condition; increment){...}` is used to execute a code block a specific number of times.

3. Counter Implementation:

- Example: Create a loop that counts from 1 to 10 and prints each number.

4. While and Do While Loops:

- Syntax: `while(condition){...}` and `do{...} while(condition);`. The while loop checks the condition before executing the code, whereas the do-while loop checks after.

Session 4: Arrays and Objects

1. Introduction to Arrays:

- Concept: Arrays are used to store multiple values in a single variable.

2. Arrays Syntax:

- Example: `let arr = [1, 2, 3, 4];`

3. Arrays Indexing:

- Access: Use `arr[index]` to access elements, with indices starting at 0.

4. Array Properties and Methods:

- Length: `arr.length` returns the number of elements.
- Push & Pop: `arr.push(element)` adds an element, `arr.pop()` removes the last element.
- Join: `arr.join(separator)` joins all elements into a string.
- Find: `arr.find(callback)` returns the first element that passes a test.

5. Array Printing:

- Loops: Use loops to iterate through array elements.

6. Objects:

- Concept: Objects are collections of key-value pairs, used to store data and functionality.

Session 5: Functions

1. Introduction to Functions:

- Concept: Functions are reusable blocks of code that perform a specific task.

2. Functions Syntax:

- Example: `function functionName(parameters){...}`

3. Parameters & Arguments:

- Parameters: Variables listed as a part of the function definition.

- Arguments: Values passed to the function when calling it.

4. Function Call:

- Example: `functionName(arguments);`

5. Creating a Sum Function:

- Example: `function sum(a, b){ return a + b; }`

6. Return Statement:

- Usage: `return` ends the function and specifies the value to be returned.

Session 6: String and Array Methods

1. String Methods:

- Methods: **`concat()`, `split()`, `includes()`, `endsWith()`, `repeat()`, `toUpperCase()`, `toLowerCase()`, `at()`**.

2. Array Methods:

- Methods: **`fill()`, `find()`, `join()`, `sort()`.**