**JavaScript Functions and Control Structures**

**Objective:** Students will learn about JavaScript functions, including how to declare and invoke them. They will also explore control structures such as conditionals and loops to control the flow of their programs.

**Functions in JavaScript**

**What is a Function?**

A function is a block of reusable code designed to perform a particular task. Functions help in organizing code into modular, manageable sections.

**Declaring a Function:**

Functions in JavaScript can be declared using the function keyword followed by a name, a list of parameters (enclosed in parentheses), and a block of code (enclosed in curly braces).

function greet(name) {

console.log('Hello, ' + name + '!');

}

function sayHello() {

console.log(‘Hi, welcome to my page’);

}

**Calling a Function:**

To execute the code inside a function, you call the function by its name followed by parentheses, optionally including arguments.

greet('Alice'); // Outputs: Hello, Alice!

greet(“Kehinde”);

sayHello();

**Function Parameters and Arguments:**

Parameters are the variables listed in the function's declaration. Arguments are the actual values passed to the function when it is called.

function add(a, b) {

return a + b;

}

add(2, 3)

let sum = add(5, 10); // sum is 15

**Return Statement:**

The return statement stops the execution of a function and returns a value to the function caller.

function multiply(a, b) {

return a \* b;

}

let product = multiply(4, 5); // product is 20

**Function Expressions:**

Functions can also be defined as expressions. These can be **anonymous** (without a name) and assigned to variables.

let square = function(number) {

return number \* number;

};

let result = square(5); // result is 25

**Arrow Functions:**

Introduced in ES6, arrow functions provide a shorter syntax for writing function expressions.

let double = (n) => n \* 2;

let doubledValue = double(4); // doubledValue is 8

Let square = (num) => {

return num \* num;

}

let result = square(5); //result - 25;

**Conditionals (if-else statements)**

**Understanding Conditionals:**

Conditionals are used to perform different actions based on different conditions. The **if** statement is the most basic conditional, followed by **else** and **else if** for more complex conditions.

let temperature = 30;

if (temperature > 25) {

console.log('It is hot outside.');

} else {

console.log('It is cool outside.');

}

**Using else if:**

The else if statement allows you to specify a new condition if the first condition is false.

let score = 85;

if (score >= 90) {

console.log('Grade: A');

} else if (score >= 80) {

console.log('Grade: B');

} else if (score >= 70) {

console.log('Grade: C');

} else {

console.log('Grade: F');

}

**Logical Operators in Conditionals:**

You can combine multiple conditions using logical operators (&& for AND, || for OR).

let age = 20;

let hasTicket = true;

if (age >= 18 && hasTicket) {

console.log('You can enter the concert.');

} else {

console.log('You cannot enter the concert.');

}

&& - logical AND - (\*), 1-true, 0-false

|| - logical OR - (+)

AND

1 && 1 = 1

1 && 0 = 0

0 && 1 = 0

0 && 0 = 0

OR

1 || 1 = 1

1 || 0 = 1

0 || 1 = 1

0 || 0 = 0

**If-Else Statement syntax**

if(condition){

code to be executed if condition is true

}else if(condition2) {

code to be executed if condition2 is true

} else {

code to be executed if condition and condition2 are false

}

**Loops**

**Understanding Loops:**

Loops allow you to run the same block of code multiple times. There are several types of loops in JavaScript: for, while, and do-while.

**for Loop:**

The for loop is used when you know how many times you want to execute a statement or a block of statements.

for (let i = 0; i < 5; i++) {

console.log('Number: ' + i);

}

**Initialization**: let i = 0 starts the loop with i set to 0.

**Condition**: i < 5 runs the loop as long as i is less than 5.

**Increment**: i++ increases i by 1 after each loop iteration.

**while Loop:**

The while loop continues to execute a block of code as long as the specified condition is true.

let count = 0;

while (count < 5) {

console.log('Count: ' + count);

count++;

}

**do-while Loop:**

The do-while loop is similar to the while loop, but it executes the code block at least once before checking the condition.

let n = 0;

do {

console.log('n is: ' + n);

n++;

} while (n < 5);

**Practical Exercise: Simple Quiz**

**Objective:** Create a simple quiz application using functions, conditionals, and loops to assess the students' understanding of basic JavaScript concepts.

**HTML Structure:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Simple Quiz</title>

</head>

<body>

<h1>Quiz</h1>

<p id="question"></p>

<input type="text" id="answer" placeholder="Your answer">

<button onclick="checkAnswer()">Submit</button>

<p id="result"></p>

<script>

let currentQuestion = 0;

const questions = [

{ question: "What is 2 + 2?", answer: "4" },

{ question: "What is the capital of France?", answer: "Paris" },

{ question: "What is the color of the sky?", answer: "blue" }

];

function showQuestion() {

document.getElementById('question').innerText = questions[currentQuestion].question;

}

function checkAnswer() {

let userAnswer = document.getElementById('answer').value;

if (userAnswer.toLowerCase() === questions[currentQuestion].answer.toLowerCase()) {

document.getElementById('result').innerText = 'Correct!';

} else {

document.getElementById('result').innerText = 'Wrong, try again.';

}

}

showQuestion();

</script>

</body>

</html>

**Explanation:**

*The HTML structure includes a paragraph to display the question, an input field for the user's answer, a button to submit the answer, and another paragraph to display the result.*

*The JavaScript part initializes an array of question objects, each containing a question and its corresponding answer.*

*The showQuestion function displays the current question on the webpage.*

*The checkAnswer function retrieves the user's answer, compares it with the correct answer, and updates the result paragraph accordingly.*