Report:

JavaScript Loops, Arrays, Functions, and Scope

# 1. JavaScript Loops:

Loops are used to execute a block of code multiple times.

## a) for Loop:

Runs a block of code a specific number of times.

Example:

for (let i = 0; i < 5; i++) {  
 console.log(i);  
}

## b) while Loop:

Executes a block of code as long as the condition is true.

Example:

let i = 0;  
while (i < 5) {  
 console.log(i);  
 i++;  
}

## c) do...while Loop:

Executes the block of code at least once before checking the condition.

Example:

let i = 0;  
do {  
 console.log(i);  
 i++;  
} while (i < 5);

# 2. JavaScript Arrays:

Arrays are used to store multiple values in a single variable. They are zero-indexed.

Example:

let fruits = ["apple", "banana", "mango"];

Common Array Methods:

- push(): Add item to end  
- pop(): Remove last item  
- shift(): Remove first item  
- unshift(): Add item to start  
- length: Get array size  
- indexOf(): Get index of item  
- includes(): Check if item exists  
- forEach(), map(), filter(), reduce(): Loop and transform data

# 3. JavaScript Functions:

Functions are blocks of code designed to perform specific tasks.

## a) Regular Function:

Defined using the function keyword.

Example:

function greet(name) {  
 return "Hello, " + name;  
}

## b) Arrow Function:

A shorter way to write functions introduced in ES6.

Example:

const greet = (name) => {  
 return "Hello, " + name;  
}

# 4. JavaScript Scope:

Scope defines where variables can be accessed in code.

## a) Global Scope:

Variables declared outside any function or block are in global scope and accessible everywhere.

## b) Function Scope:

Variables declared inside a function using var are only accessible within that function.

## c) Block Scope:

Variables declared with let or const inside a block (like if or for) are accessible only inside that block.

# Conclusion:

Understanding loops, arrays, and functions helps in building powerful and efficient programs. Knowing the scope of variables is important to prevent errors and manage data flow in JavaScript.