JavaScript and DSA Techniques

Complete Notes on JavaScript and DSA Methods

JavaScript Closures

Closures in JavaScript:

A closure is a combination of a function bundled together with its lexical environment.

It allows a function to access variables from an enclosing scope, even after that scope has been executed.

Example:

```
function outerFunction(outerVariable) {
    return function innerFunction(innerVariable) {
        console.log(`Outer Variable: ${outerVariable}, Inner Variable: ${innerVariable}`);
    }
}
const newFunction = outerFunction('outside');
newFunction('inside');
```

Lexical Scoping in JavaScript

Lexical Scoping in JavaScript:

Lexical scoping means the scope of a variable is defined by its position within the source code, and nested functions have access to variables declared in their outer scope.

Example:

```
function outer() {
    let outerVar = 'I am outer';
    function inner() {
        console.log(outerVar); // Accessible due to lexical scoping
    }
    inner();
}
outer();
```

Array Traversal Methods

Array Traversal in JavaScript:

Methods for processing each element of an array.

1. Linear Traversal:

```
arr.forEach((value) => console.log(value));
```

2. Map Traversal:

```
const result = arr.map(value => value * 2);
```

3. Filter Traversal:

```
const filtered = arr.filter(value => value > 10);
```

4. Reduce Traversal:

```
const sum = arr.reduce((acc, value) => acc + value, 0);
```

5. Reverse Traversal:

```
arr.reverse().forEach(value => console.log(value));
```

DSA Traversal Techniques

DSA Traversal Techniques:

Traversing refers to visiting each element in a data structure.

1. Linked List Traversal:

```
function traverseLinkedList(head) {
    while (head !== null) {
        console.log(head.data);
        head = head.next;
    }
}
```

2. Tree Traversals:

a) Inorder: Left -> Root -> Right

b) Preorder: Root -> Left -> Right

c) Postorder: Left -> Right -> Root

3. Graph Traversals:

a) BFS: Visit level by level using a queue.

b) DFS: Visit depth-wise using recursion or a stack.

4. Matrix Traversal:

Iterate through rows and columns to visit each element.