Sheryians Coding School

Live Cohort

Notes Day 17

Day 17 : JavaScript Mastery

1 . Working with Strings in JavaScript

JavaScript provides various methods to manipulate and work with strings. Since strings are immutable, any modification results in a new string.

1. **slice()**
   - Extracts a section of a string and returns a new string without modifying the original.
   - Example:

```
let str = "Hello World";
console.log(str.slice(0, 5)); // Output: "Hello"
```

2. **Template Strings**
   - Uses backticks (`` ` ``) to allow embedding expressions within a string.
   - Example:

```
let name = "John";
console.log(`Hello, ${name}!`); // Output: "Hello, John!
```

3. **split()**
   - Splits a string into an array based on a separator.
   - Example:

```
let words = "Hello World".split(" ");
console.log(words); // Output: ["Hello", "World"]
```

Day 17 : JavaScript Mastery

1 . Working with Strings in JavaScript

4. **replace()**
   - Replaces a specified substring with another.
   - Example:

```
let text = "Hello";
console.log(text.replace("H", "J")); // Output: "Jello"
```

5. **includes()**
   - Checks if a substring exists within a string.
   - Example:

```
console.log("JavaScript".includes("Java")); // Output: true
```

Day 17 : JavaScript Mastery

2 . Conditional Operators in JavaScript

Conditional Statements

1. **if statement**
   - Executes code if a condition is true.
   - Example:

```
let x = 10;
if (x > 5) {
  console.log("Big number");
}
```

2. **if-else statement**
   - Runs different code blocks based on a condition.
   - Example:

```
let num = 4;
if (num > 5) {
  console.log("Greater than 5");
} else {
  console.log("Less than or equal to 5");
}
```

Day 17 : JavaScript Mastery

2 . Conditional Operators in JavaScript

3. **else-if ladder**
   - Used for multiple conditions.
   - Example:

```
let score = 85;
if (score >= 90) {
  console.log("A grade");
} else if (score >= 80) {
  console.log("B grade");
} else {
  console.log("Below B grade");
}
```

4. **Ternary Operator**
   - Shorter way to write an if-else statement.
   - Example:

```
let age = 18;
let status = age >= 18 ? "Adult" : "Minor";
console.log(status); // Output: "Adult"
```

Day 17 : JavaScript Mastery

2 . Conditional Operators in JavaScript

5. **switch statement**
   - Alternative to multiple if-else conditions.
   - Example:

```
let day = "Monday";
switch (day) {
  case "Monday":
    console.log("Start of the week");
    break;
  case "Friday":
    console.log("Weekend is coming!");
    break;
  default:
    console.log("Normal day");
}
```

Day 17 : JavaScript Mastery

3 . Loops in JavaScript

🔹 Loop Types

1. **for loop**
   - Repeats a block of code a specific number of times.
   - Example:

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

2. **while loop**
   - Executes a block while a condition is true.
   - Example:

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

3. **do...while loop**
   - Executes the block at least once before checking the condition.
   - Example:

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

Day 17 : JavaScript Mastery

3 . Loops in JavaScript

4. **forEach loop (for arrays)**
   - Iterates over each element in an array.
   - Example:

```
[1, 2, 3].forEach(num => console.log(num));
```

5. **for...in loop (for objects)**
   - Iterates over object properties.
   - Example:

```
let obj = { a: 1, b: 2 };
for (let key in obj) {
  console.log(key, obj[key]);
}
```

6. **for...of loop (for iterables)**
   - Iterates over iterable objects like arrays and strings.
   - Example:

```
let arr = [10, 20, 30];
for (let value of arr) {
  console.log(value);
}
```

Day 17 : JavaScript Mastery

4 . Functions in JavaScript

🔹 Function Types

1. **Regular Function**
   - Example:

```
function greet(name) {
  return `Hello, ${name}`;
}
console.log(greet("Alice"));
```

2. **Arrow Function**
   - Shorter syntax for defining functions.
   - Example:

```
const add = (a, b) => a + b;
console.log(add(2, 3)); // Output: 5
```

3. **Immediately Invoked Function Expression (IIFE)**
   - A function that runs immediately after being defined.
   - Example:

```
(function() {
  console.log("This runs immediately!");
})();
```

Day 17 : JavaScript Mastery

4 . Functions in JavaScript

4. **Higher-Order Function**
   - A function that takes another function as an argument.
   - Example:

```
function operate(fn, a, b) {
  return fn(a, b);
}
console.log(operate((x, y) => x + y, 10, 20));
// Output: 30
```

Day 17 : JavaScript Mastery

5 . Scoping & Closures in JavaScript

🔹 Scoping Rules

1. **Global Scope**
   - Variables declared outside any function are accessible anywhere.
   - Example:

```
let globalVar = "Hello";
function show() {
  console.log(globalVar);
}
```

2. **Function Scope**
   - Variables declared inside a function are not accessible outside.
   - Example:

```
function example() {
  let localVar = 10;
}
console.log(localVar); // Error
```

3. **Closures**
   - A function that remembers variables from its outer function.
   - Example:

```
function outer(x) {
  return function inner(y) {
    return x + y;
  };
}
const addFive = outer(5);
console.log(addFive(10)); // Output: 15
```