JavaScript Tutorial

# 1. How to Create a JavaScript File and Execute Code using Node.js and VS Code

To create and execute JavaScript code, you need two tools: Node.js (to run JavaScript outside a browser) and a code editor like Visual Studio Code (VS Code).

Follow these steps:

1. Install Node.js:   
 - Go to https://nodejs.org/ and download the latest version of Node.js for your system.   
 - After installation, open your terminal (Command Prompt/PowerShell for Windows, Terminal for macOS/Linux).   
 - Verify installation by typing:   
   
 node --version

2. Install Visual Studio Code (VS Code):   
 - Download and install VS Code from https://code.visualstudio.com/.   
 - Open VS Code and create a new file. Save it with a `.js` extension (e.g., `app.js`).

3. Write and Execute JavaScript Code:   
 - Inside `app.js`, write some simple JavaScript code:  
   
 console.log('Hello, World!');  
   
 - To execute this code, open the terminal in VS Code (View > Terminal) and type:  
   
 node app.js  
   
 You should see `Hello, World!` printed in the terminal.

# 2. Define Variables and Data Types in JavaScript

Variables store data values that can be used and manipulated in your programs. In JavaScript, there are three ways to declare variables: `var`, `let`, and `const`. The basic data types include numbers, strings, booleans, arrays, and objects.

Example:  
  
let age = 25; // Number  
const name = 'John'; // String  
let isStudent = true; // Boolean  
let hobbies = ['reading', 'gaming', 'sports']; // Array  
let person = { name: 'Jane', age: 30 }; // Object

Use `let` for variables whose values can change and `const` for constants (values that won’t change).

# 3. Comparison Operations in JavaScript

Comparison operators are used to compare two values. They return a boolean (true or false). Common comparison operators are:

let a = 10;  
let b = 20;  
  
console.log(a == b); // false (equal to)  
console.log(a != b); // true (not equal to)  
console.log(a > b); // false (greater than)  
console.log(a < b); // true (less than)  
console.log(a >= b); // false (greater than or equal to)  
console.log(a <= b); // true (less than or equal to)

# 4. Use Control Structures in JavaScript

Control structures allow you to execute code based on conditions or repeatedly through loops. The main control structures are `if/else`, `switch/case`, and loops (`for`, `while`).

## If/Else Example

let age = 18;  
  
if (age >= 18) {  
 console.log('You are an adult.');  
} else {  
 console.log('You are a minor.');  
}

## Switch/Case Example

let day = 2;  
  
switch (day) {  
 case 1:  
 console.log('Monday');  
 break;  
 case 2:  
 console.log('Tuesday');  
 break;  
 default:  
 console.log('Other day');  
}

## Loop Example

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

# 5. Define and Manipulate Strings in JavaScript

Strings are sequences of characters, typically used to represent text. You can create strings using single quotes (' ') or double quotes (" ").

Example:  
  
let greeting = 'Hello';  
let name = 'Alice';  
let message = greeting + ', ' + name + '!';  
console.log(message); // Output: Hello, Alice!

# 6. Demonstrating String Methods

JavaScript provides several built-in methods to manipulate strings.

let text = 'Hello World';  
  
// Length of the string  
console.log(text.length); // Output: 11  
  
// Convert to uppercase  
console.log(text.toUpperCase()); // Output: 'HELLO WORLD'  
  
// Convert to lowercase  
console.log(text.toLowerCase()); // Output: 'hello world'  
  
// Extract a substring  
console.log(text.substring(0, 5)); // Output: 'Hello'  
  
// Replace part of a string  
console.log(text.replace('World', 'JavaScript')); // Output: 'Hello JavaScript'