I. **Core JavaScript Fundamentals**:

Data Types:

- Primitive (Number, String, Boolean, Null, Undefined, Symbol, BigInt)
- Variables and Scoping:var, let, and const: differences in scope (function, block) and hoisting behavior.
- Global scope, function scope, block scope.
- Temporal Dead Zone (TDZ) for let and const.

Operators:

- Equality operators (== vs. ===) and type coercion.
- Logical operators (AND, OR, NOT).
- Ternary operator.
- Control Flow:if/else, switch statements.
- Loops (for, while, do-while, for...in, for...of).
- break and continue.

Functions:**Function declarations vs. function expressions**.

- Arrow functions (ES6): syntax, this binding, limitations.
- Parameters and arguments.
- Default parameters.
- Immediately Invoked Function Expressions (IIFEs).

Objects and Arrays:

- Creating objects (literal, constructor, Object.create()).
- Accessing properties (dot vs. bracket notation).
- Array methods (map, filter, reduce, forEach, splice, slice, includes, etc.).
- Shallow vs. deep copy of objects/arrays.
- Destructuring assignment.
- Spread operator (...).
- Object.freeze(), Object.seal(), Object.preventExtensions().
- Getters and Setters

.

II. **Advanced JavaScript Concepts**:

II. Advanced JavaScript Concepts:

- this Keyword:Understanding its context in different scenarios (global, function, method, constructor, arrow function, call/apply/bind).

- Closures:Definition, how they work, and common use cases (e.g., data encapsulation, private variables, currying).

- Hoisting:Understanding how variable and function declarations are "hoisted" during compilation.

- Differences in hoisting behavior for var, let, const, and function declarations/expressions.

Prototypal Inheritance:

- The [[Prototype]] chain.
- How objects inherit properties and methods.
- Constructor functions and the new keyword.

Asynchronous JavaScript:

- Callbacks: Basic understanding, callback hell.
- Promises: States (pending, fulfilled, rejected), then(), catch(), finally(), Promise.all(), Promise.race().
- async/await: How it simplifies asynchronous code, error handling with try...catch.

Event Loop:

- Microtask queue, macrotask queue, how JavaScript handles concurrency.
- setTimeout, setInterval.

Workspace API for HTTP requests.

- Map and Set:Differences from plain objects and arrays, use cases.
- WeakMap and WeakSet:Differences from Map and Set, garbage collection.
- Memoization:Caching function results for performance optimization.