ES6+ Features in JavaScript

ES6 (ECMAScript 2015) introduced many modern and powerful features that make JavaScript cleaner, shorter, and easier to write.
Let's cover the most important ones — **Modules, Destructuring, Spread/Rest, Template Literals, and Classes**

□ 1. Modules (import / export)

Modules let you **split your code** into multiple files for better organization and reuse.

Example:

math.js

```
export function add(a, b) {
  return a + b;
}

export const PI = 3.1416;
```

main.js

```
import { add, PI } from './math.js';

console.log(add(5, 10));  // Output: 15
console.log(PI);          // Output: 3.1416
```

→ You can also use **default export** if exporting only one main thing:

```
// math.js
export default function square(x) {
  return x * x;
}
// main.js
import square from './math.js';
console.log(square(5)); // Output: 25
```

📝 **Note:** You must use `type="module"` in your HTML:

```
<script type="module" src="main.js"></script>
```

□ 2. Destructuring (Arrays & Objects)

Destructuring allows you to **unpack values** from arrays or objects easily.

👉 Array Destructuring:

```
const fruits = ["apple", "banana", "mango"];
const [first, second, third] = fruits;
console.log(first); // apple
```

👉 Object Destructuring:

```
const person = { name: "Raj", age: 24, country: "Nepal" };
const { name, age } = person;
console.log(name); // Raj
console.log(age);  // 24
```

You can also rename:

```
const { name: fullName } = person;
console.log(fullName); // Raj
```

🌀 3. Spread and Rest Operators (...)

👉 Spread Operator — expands elements

Used to copy or merge arrays/objects.

```
const arr1 = [1, 2, 3];
const arr2 = [...arr1, 4, 5];  // spread
console.log(arr2); // [1, 2, 3, 4, 5]
```

With objects:

```
const user = { name: "Raj", age: 24 };
const updated = { ...user, country: "Nepal" };
console.log(updated);
```

👉 Rest Operator — collects remaining values

Used to group remaining items into an array.

```
function sum(...numbers) {
  return numbers.reduce((total, n) => total + n, 0);
}

console.log(sum(1, 2, 3, 4)); // Output: 10
```

💬 4. Template Literals (Backticks ``)

Used to embed variables or expressions directly in strings.
No more `"Hello " + name + "!"` — you can now use `${}` inside backticks.

```
let name = "Raj";
let age = 24;
console.log(`My name is ${name} and I am ${age} years old.`);
```

You can also write **multi-line strings**:

```
let message = `
Hello ${name},
Welcome to JavaScript ES6 world!
`;
console.log(message);
```

□ 5. Classes (Object-Oriented JavaScript)

ES6 introduced `class` syntax for creating objects easily.

A **class** is a **template for creating objects**.
It lets you group related **properties** (data) and **methods** (functions) together neatly.

Basic Syntax

```
class ClassName {
  constructor(parameter1, parameter2) {
    this.property1 = parameter1;
    this.property2 = parameter2;
  }

  methodName() {
    // method logic
  }
}
```

👉 Example:

```
class Person {
  constructor(name, age) {
    this.name = name;
    this.age = age;
  }

  greet() {
    console.log(`Hi, I'm ${this.name} and I'm ${this.age} years
old.`);
  }
}

const raj = new Person("Raj", 24);
raj.greet(); // Output: Hi, I'm Raj and I'm 24 years old.
```

👉 Inheritance:

You can create subclasses using `extends`.

```
class Student extends Person {
  constructor(name, age, course) {
    super(name, age); // call parent constructor
    this.course = course;
  }

  study() {
    console.log(`${this.name} is studying ${this.course}.`);
  }
}

const student1 = new Student("Raj", 24, "BCA");
student1.greet(); // inherited
student1.study();
```

□ Mini Assignment

1. Create a module with two exported functions (`add`, `multiply`) and import them.
2. Use destructuring to extract name and age from a `person` object.
3. Use spread to merge two arrays.
4. Use a rest parameter in a function that sums all given numbers.
5. Create a class `Car` with `model`, `year`, and a method `start()`.