### ****1. Features of ES6 (ECMAScript 2015):****

* let **and** const: Block-scoped variable declarations (unlike var).
* **Arrow Functions**: Shorter syntax for function expressions with lexical this.
* **Classes**: Syntax sugar over JavaScript’s prototype-based inheritance.
* **Template Literals**: Backtick strings with ${} for expressions.
* **Destructuring**: Extract values from arrays or objects easily.
* **Default Parameters**: Set default values for function parameters.
* **Rest & Spread Operators (**...**)**:
  + Rest: function(...args)
  + Spread: [...array1, ...array2]
* **Modules (import/export)**: Support for modular code.
* **Promises**: For asynchronous operations.
* **Enhanced Object Literals**: Shorter syntax for object properties/functions.
* **Map and Set**: New data structures.
* **For-of loop**: Loop over iterable objects.

### ****2. Explain JavaScript**** let****:****

* Introduced in ES6.
* Block-scoped (i.e., only accessible within the block {} where it's defined).
* **Not hoisted** like var (or hoisted but not initialized).
* Can be **reassigned** but **not redeclared** in the same scope.

js

CopyEdit

let x = 10;

x = 20;

let x = 30;

### ****3. Difference between**** var ****and**** let****:****

| Feature | var | let |
| --- | --- | --- |
| Scope | Function-scoped | Block-scoped |
| Hoisting | Yes (initialized as undefined) | Yes (but not initialized) |
| Redeclaration | Allowed in same scope | Not allowed in same scope |
| Temporal Dead Zone | No | Yes (access before declaration throws error) |
| Global Object | Attached to window in global | Not attached to window |

### ****4. Explain JavaScript**** const****:****

* Also block-scoped like let.
* **Cannot be reassigned** after initialization.
* **Must be initialized at declaration**.
* If assigned an object/array, its **contents can still be modified**.

js

CopyEdit

const a = 5;

a = 10;

const obj = {name: "Rishi"};

obj.name = "T";

### ****5. ES6 Class Fundamentals:****

* ES6 introduced class as syntax sugar over prototypes.
* Classes can contain constructors, methods, and static methods.

js

CopyEdit

class Person {

constructor(name) {

this.name = name;

}

greet() {

console.log(`Hi, I'm ${this.name}`);

}

static info() {

return "This is a Person class";

}

}

### ****6. ES6 Class Inheritance:****

* Use extends to create a subclass.
* Use super() to call the parent constructor.

js

CopyEdit

class Animal {

constructor(name) {

this.name = name;

}

speak() {

console.log(`${this.name} makes a noise`);

}

}

class Dog extends Animal {

speak() {

super.speak(); // optional call to parent

console.log(`${this.name} barks`);

}

}

### ****7. Define ES6 Arrow Functions:****

* Short syntax for writing functions.
* **Does not have its own** this**,** arguments**, or** super.
* Ideal for inline functions or callbacks.

js

CopyEdit

const add = (a, b) => a + b;

const greet = name => console.log(`Hello, ${name}`);

### ****8. Identify**** Set() ****and**** Map()

#### ➤ Set:

* Collection of **unique** values (no duplicates).
* Iteration order is insertion order.

js

CopyEdit

const nums = new Set([1, 2, 3, 2]);

nums.add(4);

nums.has(2); // true

#### ➤ Map:

* Stores key-value pairs.
* Keys can be **any type**, not just strings.

js

CopyEdit

const userMap = new Map();

userMap.set('name', 'Rishi');

userMap.set(1, 'One');

userMap.get(1); // 'One'