**OBJECTIVES**

# List the features of ES6

* Let and Const keywords
* Arrow functions
* Template literals
* Default parameters
* Destructuring assignment
* Enhanced object literals
* Promises
* Classes and Inheritance
* Modules (import/export)
* Spread and Rest operators
* for...of loop
* Maps and Sets

# Explain JavaScript let

* The `let` keyword is used to declare variables that are block-scoped. It prevents variable redeclaration and ensures the variable is only accessible within the block it is defined in.

# Identify the differences between var and let



| **Feature** | **var** | **let** |
| --- | --- | --- |
| **Scope** | Function-scoped | Block-scoped ({ }) |
| **Redeclaration** | Allowed within the same scope | Not allowed within the same scope |
| **Reassignment** | Allowed | Allowed |
| **Hoisting** | Hoisted to the top of scope and initialized as undefined | Hoisted but **not initialized** (accessing before declaration throws ReferenceError) |
| **Global Object Property** | Declaring a var in global scope adds it as a property of window (in browsers) | Declaring a let in global scope does **not** add it to window |
| **Usage Recommendation** | Older syntax, avoid unless needed for function scope compatibility | Preferred for modern JavaScript due to block scoping |

# Explain JavaScript const

* `const` is used to declare variables whose values are constant and cannot be reassigned. It is block-scoped like `let`, but it does not allow reassignment after the initial value is set.

# Explain ES6 class fundamentals

* ES6 introduced a class syntax for creating objects and dealing with inheritance.  
  Key features:  
  - Defined using the `class` keyword  
  - Constructor method for initialization  
  - Can include methods  
  - Provides syntactic sugar over JavaScript's prototype-based inheritance

# Explain ES6 class inheritance

* Inheritance in ES6 is achieved using the `extends` keyword.  
  A subclass can inherit methods and properties from a parent class. The `super()` method is used to call the parent class constructor.

# Define ES6 arrow functions

* Arrow functions provide a shorter syntax for writing functions using `=>`.  
  They do not bind their own `this`, `arguments`, or `super`, making them ideal for callbacks and functional programming.

# Identify set(), map()

* `Set`: A collection of unique values. It does not allow duplicate elements.  
  `Map`: A collection of key-value pairs where keys can be of any data type. It remembers the original insertion order of the keys.