**React 9**

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

Some key features introduced in ES6 include:

* let and const (block-scoped variables)
* Arrow functions (=>)
* Template literals (`Hello, ${name}`)
* Default parameters (function greet(name = 'User'))
* Destructuring (e.g., const [a, b] = arr)
* Spread (...) and Rest (...) operators
* Promises
* Classes and Inheritance
* Modules (import, export)
* Enhanced Object Literals
* Map, Set, WeakMap, WeakSet
* for...of loop
* Symbols

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

* Introduced in ES6.
* Used to declare **block-scoped** variables (i.e., scoped within {}).
* Can be **reassigned**, but **not re-declared** in the same scope.

let x = 10;

x = 20; // allowed

let x = 30; // Error: Identifier 'x' has already been declared (in the same scope)

1. **Differences between var and let**

| **Feature** | **var** | **let** |
| --- | --- | --- |
| Scope | Function-scoped | Block-scoped ({}) |
| Hoisting | Hoisted, initialized as undefined | Hoisted, but not initialized |
| Redeclaration | Allowed in same scope | Not allowed in same scope |
| Usage before declaration | Allowed (but undefined) | ReferenceError |

**Example:**

console.log(a); // undefinedvar a = 5;

console.log(b); // ReferenceErrorlet b = 5;

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

* Used to declare **block-scoped constants**.
* Must be **initialized at the time of declaration**.
* Cannot be **reassigned**.
* For **objects or arrays**, their **contents can be changed**, but not the reference.

const x = 10;

x = 20; // Error

const arr = [1, 2];

arr.push(3); // allowed

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

* Syntactic sugar over JavaScript's existing prototype-based inheritance.
* Supports constructor, methods, getters/setters.

class Person {

constructor(name) {

this.name = name;

}

greet() {

return `Hello, ${this.name}`;

}

}

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

* Use extends keyword to inherit from another class.
* Use super() to call the parent constructor.

class Animal {

constructor(name) {

this.name = name;

}

speak() {

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

}

}

class Dog extends Animal {

speak() {

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

}

}

const d = new Dog("Rex");

d.speak(); // Rex barks.

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

* Shorter syntax for writing functions.
* No this, arguments, super, or new.target binding.
* Useful for anonymous functions and callbacks.

const add = (a, b) => a + b;console.log(add(2, 3)); // 5

const greet = name => `Hello, ${name}`;

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

#### Set

A collection of **unique values**.

Maintains insertion order.

const s = new Set();

s.add(1);

s.add(2);

s.add(1); // duplicate, ignoredconsole.log(s); // Set(2) {1, 2}

#### Map

A collection of **key-value pairs**.

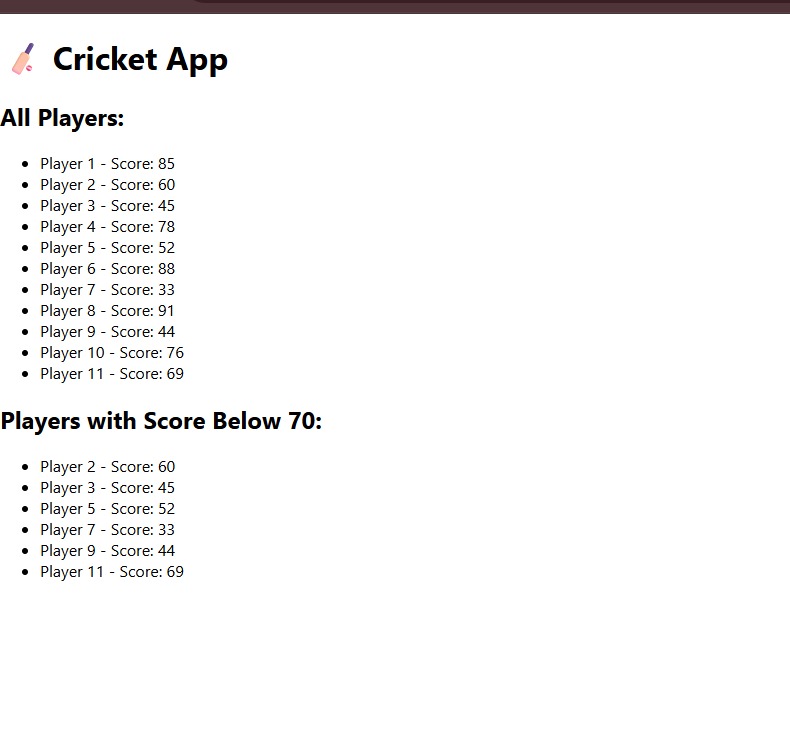
Keys can be of any type (objects, functions, etc.).

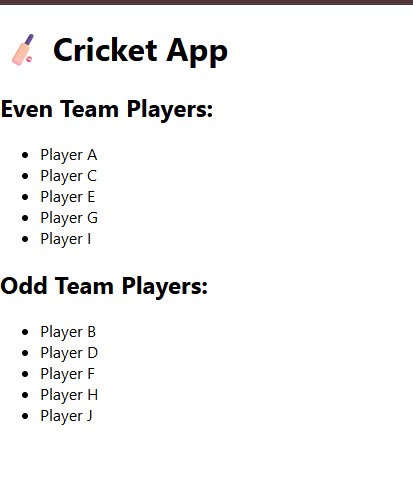
const m = new Map();

m.set('a', 1);

m.set('b', 2);console.log(m.get('a')); // 1

**Output for the Lab:**

****

****