Question 1: What are variables in JavaScript? How do you declare a variable using var, let and const?

* **Variables** in JavaScript are containers used to **store data values** such as numbers, strings, booleans, objects, etc. They allow you to reference and manipulate values in your program.
* **Ways to Declare a Variable**

JavaScript provides **three keywords** to declare a variable:

| **Keyword** | **Scope Type** | **Reassignment Allowed** | **Redeclaration Allowed** | **Hoisting Behavior** |
| --- | --- | --- | --- | --- |
| var | Function-scoped | ✅ Yes | ✅ Yes | Hoisted (initialized as undefined) |
| let | Block-scoped | ✅ Yes | ❌ No | Hoisted (but not initialized) |
| const | Block-scoped | ❌ No (read-only) | ❌ No | Hoisted (but not initialized) |

**✅ Syntax and Examples**

**1. Using var**

*var name = "Jayesh";*

*var age = 25;*

*console.log(name); // Jayesh*

* Can be redeclared and reassigned.
* Function scope.

**2. Using let**

*let city = "Ahmedabad";*

*city = "Surat"; // ✅ Allowed*

*console.log(city); // Surat*

* Cannot be redeclared in the same block.
* Block scope (inside {}).

**3. Using const**

*const country = "India";*

*// country = "USA"; ❌ Error: Assignment to constant variable*

*console.log(country); // India*

* Must be **initialized at the time of declaration**.
* Cannot be changed (used for constants).
* Block scope.

Question 2: Explain the different data types in JavaScript. Provide examples for each.

* JavaScript has **8 main data types**, divided into two categories:

**🧾 1. Primitive Data Types (Immutable)**

* These are simple, **single values**, not objects.

| **Data Type** | **Description** | **Example** |
| --- | --- | --- |
| String | Represents text | "Hello" |
| Number | Any number (integer or float) | 42, 3.14 |
| BigInt | For very large integers | 12345678901234567890n |
| Boolean | True or false value | true, false |
| undefined | Variable declared but not assigned | let x; |
| null | Represents intentional absence of value | let y = null; |
| Symbol | Unique and immutable value (for keys) | Symbol("id") |

**🧱 2. Non-Primitive (Reference) Data Types**

* These hold **collections of values** or **complex structures**.

| **Type** | **Description** | **Example** |
| --- | --- | --- |
| Object | Key-value pairs | { name: "Jayesh", age: 25 } |
| Array | Ordered collection (list) | [1, 2, 3, 4] |
| Function | Callable object | function greet() {} |

Question 3: What is the difference between undefined and null in JavaScript?

**undefined vs null**

| **Feature** | **undefined** | **null** |
| --- | --- | --- |
| **Meaning** | Variable has been declared but not assigned | Variable has been explicitly set to "no value" |
| **Type** | undefined (primitive) | object (this is a known JavaScript bug) |
| **Set by** | JavaScript (automatically) | Developer (manually) |
| **Usage** | Indicates uninitialized variable | Used to intentionally clear a variable |
| **Equality (==)** | undefined == null → true | Same |
| **Strict Equality (===)** | undefined === null → false | Different types |

**🧪 Examples**

**✅ undefined Example**

*let x;*

*console.log(x); // undefined (not assigned)*

*console.log(typeof x); // "undefined"*

* JavaScript assigns undefined by default when no value is given.

**✅ null Example**

*let y = null;*

*console.log(y); // null*

*console.log(typeof y); // "object" (JavaScript quirk)*

* null is used **intentionally** to represent "no value".

**❓ Key Difference**

* undefined = system-level default (no value assigned)
* null = developer-level intention (explicitly empty)

**✅ When to Use**

| **Scenario** | **Use** |
| --- | --- |
| Variable declared but not initialized | undefined (automatic) |
| Intentionally clearing a value | null (manual) |