🧩 1️⃣ STRING MANIPULATION BASICS

🔹 Declaring and Accessing

```
let str = "hello";
console.log(str[0]); // 'h'
console.log(str.length); // 5
```

🔹 Common String Methods

Operation	Syntax	Description
Convert to uppercase	`str.toUpperCase()`	→ `"HELLO"`
Convert to lowercase	`str.toLowerCase()`	→ `"hello"`
Slice	`str.slice(start, end)`	Extracts substring
Substring	`str.substring(start, end)`	Similar to slice, but handles negatives differently
Split	`str.split(" ")`	Converts string → array
Join	`arr.join(" ")`	Converts array → string
Trim	`str.trim()`	Removes extra spaces
Replace	`str.replace('a', 'b')`	Replaces first occurrence
Replace all	`str.replaceAll('a', 'b')`	Replaces all
Includes	`str.includes('lo')`	true / false
Index	`str.indexOf('e')`	→ 1
Reverse (via array)	`str.split('').reverse().join('')`	→ `"olleh"`

🧩 2️⃣ ARRAY MANIPULATION BASICS

🔹 Creating Arrays

```
let arr = [1, 2, 3];
let fruits = new Array("apple", "banana", "cherry");
```

🔹 Common Array Methods

Operation	Syntax	Description
Add element (end)	`arr.push(x)`	Adds at end
Add element (start)	`arr.unshift(x)`	Adds at start
Remove last	`arr.pop()`	Removes last element
Remove first	`arr.shift()`	Removes first element
Slice	`arr.slice(start, end)`	Copies part of array
Splice	`arr.splice(index, count)`	Removes/replaces elements
Concatenate	`arr.concat(arr2)`	Combines arrays
Join	`arr.join(',')`	Convert to string
Includes	`arr.includes(x)`	Checks existence
Index	`arr.indexOf(x)`	Returns index or -1
Reverse	`arr.reverse()`	Reverses array
Sort	`arr.sort((a,b)=>a-b)`	Sort numerically
Length	`arr.length`	Returns number of elements

🧩 3️⃣ MAP, FILTER, REDUCE (🔥 MUST MASTER)

These are **higher-order functions** — they take another function as an argument.

🔹 map()

📘 Creates a *new array* by applying a function to every element.
Does not modify original array.

```
let nums = [1, 2, 3, 4];
let doubled = nums.map(n => n * 2);
console.log(doubled); // [2, 4, 6, 8]
```

💡 Use when you want to **transform** each element.

🔹 filter()

📘 Creates a *new array* with elements that pass a test (returns `true`).

```
let nums = [1, 2, 3, 4, 5];
let even = nums.filter(n => n % 2 === 0);
console.log(even); // [2, 4]
```

💡 Use when you want to **select some elements** based on condition.

🔹 reduce()

📘 Reduces array to a *single value* (sum, product, max, etc.)

```
let nums = [1, 2, 3, 4];
let sum = nums.reduce((acc, curr) => acc + curr, 0);
console.log(sum); // 10
```

💡 Use when you want to **accumulate** values into one (e.g., total, object, string).

💡 Combined Example:

```
let nums = [1, 2, 3, 4, 5, 6];

let result = nums
  .filter(n => n % 2 === 0) // [2,4,6]
  .map(n => n * n)          // [4,16,36]
  .reduce((a, b) => a + b); // 56
```

```
console.log(result); // 56
```

🧩 DOM (Document Object Model) — JavaScript Notes

The **DOM** represents your HTML document as a tree of nodes.
JavaScript can access and modify HTML and CSS dynamically through the DOM.

📘 1️⃣ Accessing Elements

✅ By ID

```
const title = document.getElementById("myId");
```

- Selects a **single element** by its unique `id`.
- Returns the element object or `null` if not found.

✅ By Class Name

```
const items = document.getElementsByClassName("myClass");
```

- Returns an **HTMLCollection** (live list).
- Use loop or `Array.from()` to iterate.

Example:

```
Array.from(items).forEach(el => el.style.color = "red");
```

✅ By Tag Name

```
const divs = document.getElementsByTagName("div");
```

- Returns all elements with that tag name.
- Also an **HTMLCollection**.

✅ By Query Selector

```
const firstItem = document.querySelector(".myClass"); // First match
```

- Returns the **first** element matching the CSS selector.

✅ By Query Selector All

```
const allItems = document.querySelectorAll(".myClass");
```

- Returns a **NodeList** (can use `forEach()` directly).
- Not live (does not auto-update if DOM changes).

🧠 2️⃣ Modifying Content

Change Text

```
document.getElementById("demo").textContent = "New Text!";
```

Change HTML

```
document.getElementById("demo").innerHTML = "<b>Bold Text</b>";
```

Change Attributes

```
const img = document.querySelector("img");
img.src = "newImage.png";
img.alt = "Updated Image";
```

Change Styles

```
const box = document.getElementById("box");
box.style.backgroundColor = "lightblue";
box.style.fontSize = "20px";
```

🎨 3️⃣ Creating and Removing Elements

Create

```
const newEl = document.createElement("p");
newEl.textContent = "Hello, I was created dynamically!";
document.body.appendChild(newEl);
```

Remove

```
newEl.remove(); // removes the element
```

Append to Specific Container

```
const container = document.getElementById("container");
container.appendChild(newEl);
```

⚡ 4️⃣ Event Handling

Add Event Listener

```
const btn = document.getElementById("clickBtn");
btn.addEventListener("click", () => {
  alert("Button Clicked!");
});
```

Common Events:

Event	Description
click	Fired when user clicks an element
mouseover	When cursor moves over an element
keydown	When a key is pressed
input	When input field value changes
submit	When a form is submitted

🧩 5️⃣ DOM Traversal (Moving Around)

Property	Description
parentElement	Returns parent node
children	Returns all child elements
firstElementChild	First child element
lastElementChild	Last child element
nextElementSibling	Next element on same level
previousElementSibling	Previous element on same level

Example:

```
const first = document.querySelector("li");
console.log(first.nextElementSibling.textContent); // logs text of next <
```

🧱 6️⃣ Useful Real-World Examples

Change Button Text on Click

```
document.querySelector("button").addEventListener("click", function() {
  this.textContent = "Clicked!";
});
```

Toggle Visibility

```
const box = document.getElementById("box");
box.style.display = box.style.display === "none" ? "block" : "none";
```

Add New List Item

```
const ul = document.querySelector("ul");
const li = document.createElement("li");
li.textContent = "New Item";
ul.appendChild(li);
```

🧩 7️⃣ Differences Between Collections

Type	Returned By	Can use forEach()	Live Update
HTMLCollection	getElementsByClassName / TagName	❌ No	✅ Yes
NodeList	querySelectorAll	✅ Yes	❌ No

💡 8️⃣ Summary Cheatsheet

Operation	Method
Select by ID	`getElementById("id")`
Select by Class	`getElementsByClassName("class")`
Select by Tag	`getElementsByTagName("tag")`
Select first match	`querySelector("selector")`
Select all matches	`querySelectorAll("selector")`
Change text	`element.textContent = "..."`
Change HTML	`element.innerHTML = "..."`
Add event	`element.addEventListener("event", fn)`
Create element	`document.createElement("tag")`
Add element	`parent.appendChild(child)`

Operation	Method
Remove element	`element.remove()`

🧠 **Tip for Interviews:**

Be comfortable using `querySelector()` and `querySelectorAll()` — they're the most flexible and used in **modern JavaScript codebases**.