✨⬛ LECTURE 4 — JAVASCRIPT OPERATORS

📘 **JavaScript Fundamentals — Operators Deep Dive**
🧠 *First Principles • Rules • Traps • Real-Life Logic*

⬛ FIRST PRINCIPLE — "OPERATOR HOTA KYA HAI?"

🔷 Simple Definition

Operators wo **symbols** hote hain
jo **values ya variables** par **koi operation** perform karte hain.

🟨 Real-Life Analogy

Calculator ke buttons (+, −, ×, ÷)
👉 wahi kaam JavaScript me operators karte hain

📌 Examples

```
+, -, *, /, ==, ===, &&, ||
```

⬛ ① ARITHMETIC OPERATORS ⬛

🔷 Use

👉 **Mathematical calculations** ke liye

Operator	Kaam	Example	Result
+	Addition	5 + 3	8
-	Subtraction	5 - 3	2
*	Multiplication	5 * 3	15

/	Division	10 / 2	5
%	Modulus (Remainder)	10 % 3	1
++	Increment	x++ / ++x	+1
--	Decrement	x-- / --x	−1
**	Exponent	2 ** 3	8

🟨 ORDER OF PRIORITY — BODMAS RULE

📌 JavaScript follows maths rules

1. Brackets
2. Power
3. Multiply / Divide (Left → Right)
4. Add / Subtract (Left → Right)

```
console.log(6 * 3 + 18 / (6 - 9));        // Confusing / bad

console.log(((6 * (3 + 18)) / (6 - 9)));  // Clear / good
```

🟨 Rule

Parentheses lagao → code readable + safe

🔷 Increment & Decrement (TRICKY PART)

```
let sum = 20;

console.log(sum++); // 20 (post-increment: use then increase)

console.log(sum--); // 21 (post-decrement)


let num = 23;

++num;
```

```
console.log(num);   // 24 (pre-increment)

console.log(--num); // 23 (pre-decrement)
```

🟥 **Trap**

Pre = pehle change
Post = baad me change

⬛ ② ASSIGNMENT OPERATORS ⬛

🔷 **Use**

👉 Variable me **value assign ya update** karne ke liye

Operator	Kaam	Example	Same As
`=`	Assign	x = 5	x = 5
`+=`	Add & assign	x += 3	x = x + 3
`-=`	Subtract & assign	x -= 3	x = x - 3
`*=`	Multiply & assign	x *= 4	x = x * 4
`/=`	Divide & assign	x /= 2	x = x / 2
`%=`	Modulus & assign	x %= 3	x = x % 3
`**=`	Power & assign	x **= 2	x = x ** 2

🟨 **Real-Life**

Wallet me paisa add/subtract karna → assignment operators

⬛ ③ COMPARISON OPERATORS ⬛

🔷 Use

👉 **Do values compare** karke `true` / `false` return karte hain

```
let a1 = 10;

let a2 = 20;


console.log(a1 == a2); // false

console.log(a2 > a1);  // true

console.log(a1 <= a2); // true
```

🟨 LOOSE vs STRICT EQUALITY

Operator	Meaning
`==`	Value compare (type convert karta hai)
`===`	Value + Type dono compare

```
let a = 10;

let str = "10";


console.log(a == str);  // true

console.log(a === str); // false
```

🟥 Golden Rule

Real projects me **hamesha `===` use karo**

⚠ SPECIAL CASE — null & undefined

```
console.log(null == undefined);  // true

console.log(null === undefined); // false


console.log(null <= 0); // true

console.log(null >= 0); // true

console.log(undefined == 0); // false
```

🟨 **Reason**

`null` numeric comparison me `0` jaisa behave karta hai
`undefined` numeric world me fit nahi hota

⬛ ④ LOGICAL OPERATORS ⬛

🔷 Use

👉 Multiple conditions ko combine karna
(Decision making)

Operator	Name	Kaam	Example
&&	AND	Dono true	5>2 && 2>1
||	OR	Koi True	OR
!	NOT	Ulta kar deta	!true

🟨 **Real-Life**

Login tabhi hoga jab
Email **AND** Password correct ho

⬛ ⑤ BITWISE OPERATORS ⬛

🔷 Use

👉 Numbers ke **binary (0/1)** form par kaam karte hain

Operator	Name	Example	Result
`&`	AND	5 & 3	1
`|`	OR	5|3	7
`^`	XOR	5 ^ 3	6
`~`	NOT	~5	-6
`<<`	Left Shift	5 << 1	10
`>>`	Right Shift	5 >> 1	2

🟨 **Note**

Mostly low-level / performance code me use hote hain

⬛ ⑥ NAN COMPARISON ⬛

```
let str3 = "rohit";

let str4 = "mohit";


console.log(Number(str3) == Number(str4));

// NaN == NaN → false
```

🟥 **RULE**

NaN kabhi kisi ke equal nahi hota — khud ke bhi nahi

⬛ ⑦ MULTIPLE COMPARISONS ⬛

```
let abc1 = 123;

let abc2 = "123";

let abc3 = 123;


console.log(abc1 == abc2 == abc3);

// (abc1 == abc2) → true

// true == 123 → false


abc3 = true;

console.log(abc1 == abc2 == abc3);

// true == true → true
```

🟨 **RULE**

Comparisons **left to right** evaluate hote hain

⬛ SHORT POWER SUMMARY ⬛

✅ Arithmetic → Maths
✅ Assignment → Update values
✅ Comparison → true / false
✅ Logical → Conditions
✅ Bitwise → Binary operations
✅ `===` > `==`
✅ NaN ≠ NaN
✅ Multiple comparisons → Left to Right

⬛ FINAL THOUGHT ⬛

🧠 **Operators = JavaScript ka decision-maker**
Rules samajh gaye → **bugs half khatam**
