Expressions & Operators

Combinations of Operator & Operand is called Expression.  
Ex: 5+20 is an expression.

Operators in JavaScript:-

1. Assignment Operators
2. Arithmetic Operators
3. Comparison Operators
4. Logical Operators
5. String Operators
6. Conditional (ternary) operator

For more info:  
https://developer.mozilla.org/en-us/docs/web/javaScript/guide/Expressions\_and\_operators#comparison

**Assignment Operator**: ‘=’  
Note:- In JS, var x = 5;  
 var y = 5;  
console.log(x==y); Output:- true  
console.log(“Hello” + x==y); Output:- false

In ES6,  
console.log(`Hello ${x==y}`); Output:- Hello true

**Arithmetic Operator**: ‘+’, ‘-’, ‘/’, ‘\*’, ‘%’, ‘++’, ‘--’

Increment & Decrement Operator:   
x++ or ++x or x-- or --x   
If used postfix, with operator after operand (for example, x++), the increment operator increments and returns the value before incrementing.

Ex: var num = 15;

Var newNum = num++ +5;

Outpu: num = 16 & newNum = 20  
Here, num++ is a post increment or Postfix operator which returning value before incrementing.

If used prefix, with operator before operand (for example, ++x), the increment operator increments and returns the value after incrementing.

Ex: var num = 15;

Var newNum = ++num + 5;

Outpu: num = 16 & newNum = 21  
Here, num++ is a pre increment or Prefix operator which returning value after incrementing.

Prefix increment operator means the variable is incremented first then the expression is evaluated using the new the value of the variable.

Similarly, with post decrement operator (x--) & pre decrement operator (--x).

**\*** What is the output of 3\*\*3?

🡪Output: 27  
**Exponentiation Operator** (\*\*), Calculates the base to the exponent power, that is, .  
3\*\*3 read as 3 exponentiation to 3 which means 3 raise to 3 or 3 power of 3 or 3^3.  
Example: 2\*\*3 🡪 2^3 returns 8  
 10\*\*-1 🡪 10^-1 returns 0.1

**\*** How to swap between two number without using third variable?

🡪var a = 5;  
var b = 10;  
a = a+b; // a = 15  
b = a-b; // b = 5  
a = a-b; // a = 10  
console.log(“the value of a is ” + a);  
console.log(“the value of a is ” + b);

OutPut: the value of a is 10  
 the value of b is 5

**Comparison Operator:**

A comparison operator compares its operands and returns a logical value based on whether the comparison is **true** or **false**.

|  |  |
| --- | --- |
| Comparison Operator | Operator Name |
| == | Equal to |
| === | Double Equal to |
| != | Not Equal |
| > | Greater Than |
| < | Less Than |
| >= | Greater Than Equal to |
| <= | Less Than Equal to |

Both ‘==’ and ‘===’ operator are responsible for comparison between two operands but ‘==’ operator only check value whereas ‘===’ operator check value as well as check variable’s value in terms of data type.

**Logical Operator: ‘&&’, ‘||’, ‘!’**

Logical operators are typically used with Boolean (logical ) values. They return Boolean value (1 / True or 0 / False).

**Logical AND** or Logical Conjunction(&&):

The logical AND (&&) operator for a set of expressions are **true** if and only if all of the expressions are true.   
In short, we can also call it as (&&) AND operator. (& = Ampercend).

**Logical OR** or Logical disjunction(||):

The logical OR (||) operator for a set of expressions are **true** if and only if, at least any one of the given expressions are true.   
In short, we can also call it as (&&) OR operator. (& = Pipe).

**Logical NOT** or Logical complement or Negation(!):

In short, we can also call it as (!) NOT operator. It makes truth to falsity and vice versa.  
Example: !( 4>5 || 5<10) Output: false

**String Operator or Concatenation Operator: ‘+’**

The concatenation operator (+) concatenates two string values together. It returns another string that is the union of the two operand strings.  
Example: var myName = “Chandan”;  
 console.log(myName + “ Kumar” );   
 OutPut: Chandan Kumar