**Operators, Precedence & Associativity:**

**Operators:** Operators are a special function that is syntactically (written) differently. Generally, operators takes two parameters and return one result.

There are many operators but here are a few examples (+, -, \*, & /)

**Example:**

var  a = 3 + 7;

console.log(a); // 7

The plus sign + is actually a function which in reality acts like the following function.

function +(a, b) {

return // add the two

}

+(3 ,4) // that would be a very long way.

JS engine provides a functionality called infix notation, in this the operator or function which in the above example is plus + sits between two paramters.

Greater than & less than operators / functions because it is reliant two numbers i.e.

var  a = 4 > 3;

console.log (a); // true - results in boolean

In short, operators are a special type of functions.

**Precedence**: It helps us to determine, which function / operator will run first.

**Associativity:** What order operator functions get called in: **(Left to Right or Right to Left)**. When functions have the same precendence.

In the references (Link below) you can see the operators with highest precedence & associtivity.

**Example:**

var a = 2 + 4 \* 5;

console.log(a); //  4 \* 5 = 20 then 20 + 3 = 23

(We know multiply has highest precedence thus it will fire up first).

**Another Example:**

var a = 2, b = 3, c = 4;

a = b = c;

console.log(a);

console.log(b);

console.log(c);

The answer would be 4 for all of them. The reason is Associativity because the assignment operator we use is equal to = it works right to left instead of left to right. which means

b = c; // 4 - first var b became the value of variable c

a = b; // 4 - then a became the value of b

**Another Example:**

var a = (3+4) \* 5;

console.log(a) // 35 - because whatever is inside the brackets has the highest precedence that is why (3 + 4) = 7 then it multiplied it with 5, that is why the answer is 35.

Without the parenthesis, multiplication would get done first then addition would follow but the parenthesis has the highest precedence thus the value inside them would take over anything inside.

**Reference for Operator precendence:**

<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Operator_Precedence>