== vs ===

Let $x = 5$

== equal $x == 5$ true

=== equal value and type $x === "5"$ false

&& vs ||

- && means all operands (values) should be true, then the condition will fulfill
- In && operator all values are checked to be true
- Example:

true && true true

true && false false

$x = 5$; $y = 6$;

$(x == 5, y == 7)$ true && false will return false

- In || (OR) operator at least one operand should be true.
- If first operand is true there is no need to check remaning.

Example

```
x = 5;  y = 6;
(x == 5 || y == 6)
```

↓ True

Truth Table

AND			OR			NOT	
T	T	T	T	T	T	T	F
T	F	F	T	F	T	T	F
F	T	F	F	T	T	F	T
F	F	F	F	F	F	F	T

If

Use to specify a block of code to be executed, if a specified condition is true.

else

Use to specify a block of code to be executed, if a specified condition is false.

else if

Use to execute a block of code, if condition 1 is not true but condition 2 is true. (if first condition is false)

```
if (time < 18) {
  greeting = "Good day"
}
```

```
else{
    greeting = "Good Evening"
}
```

```
if (time < 10){
    "Good Morning"
}
else if (time < 20){
    "Good Day"
}
else{
    "Good Evening"
}
```

Switch

- For multiple conditions
- switch executes the code blocks that matches an expression
- More than one code block can be executed if they match the expression

```
switch (expression) {
    case x:
        // code block
        break;
    case y:
        // code block
        break;
    default:
        // code block
}
```

- Use of break keyword prevent thrugh executed of other block especially the

default one.

use of assigment operator

- To make code more compact.
- For conciseness
- For readability
- code clearity in Loops

Conditions (Marks)

```
marks=Prompt("Enter Your Marks")
  if (marks >= 90){
        grade="A+"
  }
  else if (marks >= 80 &&
          marks < 90){
        grade="A"
  }
```

```
else if (marks >= 70 && marks < 80){
        grade = "B";
}
else if (marks >= 60 && marks < 70){
        grade = "C";
}
else if (marks >= 50 && marks < 60)
{
        grade = "D";
}
else {
        grade = "F";
}
```

Type	Operator	Precendence
Parenthese	( )	High
Unary Operator (Logical not)	! - ~ -- ++	
Multiplicatio / Modulus/bivision	* / %	
Addition / Substraction	+ -	
Shift	<< >>	
Comparison	> >= < <=	
Equality	== === != !==	
Bitwise AND	&	
Bitwise XOR	^	
Bitwise OR	|	
Logical AND	&&	
Logical OR	||	
Assigment	=, += -=	Low