**Logical operators**

There are three logical operators in JavaScript: || (OR), && (AND), ! (NOT).

**[|| (OR)](https://javascript.info/logical-operators" \l "or)**

The “OR” operator is represented with two vertical line symbols:

result = a || b;

In classical programming, logical OR is meant to manipulate boolean values. If any of it’s arguments is true, then it returns true, otherwise – returns false.

There are four possible logical combinations:

alert( true || true ); // true

alert( false || true ); // true

alert( true || false ); // true

alert( false || false ); // false

As we can see, the result is always true except for the case when both operands are false.

Most of time, OR || is used in if to test if *any* of given conditions is correct.

For example:

let hour = 9;

if (hour < 10 || hour > 18) {

alert( 'The office is closed.' );

}

We can pass more conditions:

let hour = 12;

let isWeekend = true;

if (hour < 10 || hour > 18 || isWeekend) {

alert( 'The office is closed.' ); // it is weekend

}

**[&& (AND)](https://javascript.info/logical-operators" \l "and)**

The AND operator is represented with two ampersands &&:

result = a && b;

In classical programming AND returns true if both operands are truthy and false – otherwise:

alert( true && true ); // true

alert( false && true ); // false

alert( true && false ); // false

alert( false && false ); // false

An example with if:

let hour = 12;

let minute = 30;

if (hour == 12 && minute == 30) {

alert( 'Time is 12:30' );

}

[**AND seeks the first falsy value**](https://javascript.info/logical-operators#and-seeks-the-first-falsy-value)

Given multiple AND’ed values:

result = value1 && value2 && value3;

The AND "&&" operator is doing the following:

* Evalutes operands from left to right.
* For each value converts it to a boolean. If the result is false, stops and returns the original value.
* If values finished (all are truthy), returns the last one.

In other words, AND returns the first falsy value or the last value if none found.

The rules above are similar to OR. The difference is that AND returns the first *falsy* value while OR returns the first *truthy* one.

Examples:

// if the first operand is truthy,

// AND returns the second one:

alert( true && false ); // false

alert( true && true ); // true

We can also pass several values in a row. See how the first falsy one is returned:

alert( true && true && false && true ); // false

When all values are truthy, the last value is returned:

alert( true && true && true ); // true, the last one

**AND && executes before OR ||**

The precedence of the AND && operator is higher than OR ||, so it executes before OR.

In the code below true && false is calculated first:

alert( true || true && false ); // true

**[! (NOT)](https://javascript.info/logical-operators" \l "not)**

The boolean NOT operator is represented with an exclamation sign "!". It has precedence over AND(&&) and OR(||).

The syntax is pretty simple:

result = !value;

The operator accepts a single argument and does the following:

1. Converts the operand to boolean type: true/false.
2. Returns an inverse value.

For instance:

alert( !true ); // false

alert( !false ); // true

**Questions and Exercises**

1. Write a program to ask a user for their age and checks that age is between 14 and 90 inclusively.

“Inclusively” means that age can reach the edges 14 or 90.

1. Write another program to check that age is NOT between 14 and 90 inclusively.

Create two variants: the first one using NOT !, the second one – without it.

1. Create a program to ask a student to enter their mark as a percentage and then displays the corresponding letter grade without using else if’s and nested if’s.
2. Design and create a program that allows a boxer to enter their weight in pounds. Indicate their weight class based on the following criteria:

Heavyweight Over 200lbs

Middleweight Between 161 and 200lbs

Lightweight Between 135 and 160lbs

Featherweight Between 126 and 134lbs

Flyweight Less than 126lbs

1. Write a program that asks a user for a 2 numbers between 1 and 10. Let the user know if they selected an even or an odd number (do not use the % operator). Provide a message informing the user that they have provided an incorrect number if they did not enter a number between 1 and 10.
2. Write a small guessing game. Have the program generate and store a random number between 1 and 10 and then give the user three attempts to guess it. If a number outside 1 and 10 is entered automatically tell them they have lost their guess and do not even check to see if they’re right.
3. Write a program to determine the area of a circle given the radius. If the user enters a negative number then provide an appropriate message informing them that the area cannot be calculated until a valid value for radius is entered. If they entered a number over 1000 tell them the number is too large to be calculated.