Operators

Javascript operators are used to perform different types of mathematical and logical computations.

1. Arithmetic Operators
2. Assignment Operators
3. Comparison Operators
4. String Operators
5. Logical Operators
6. Bitwise Operators
7. Ternary Operators
8. Type Operators

Arithmetic Operators:

Used to perform arithmetic operations on numbers.

• Addition (+)

• Subtraction (-)

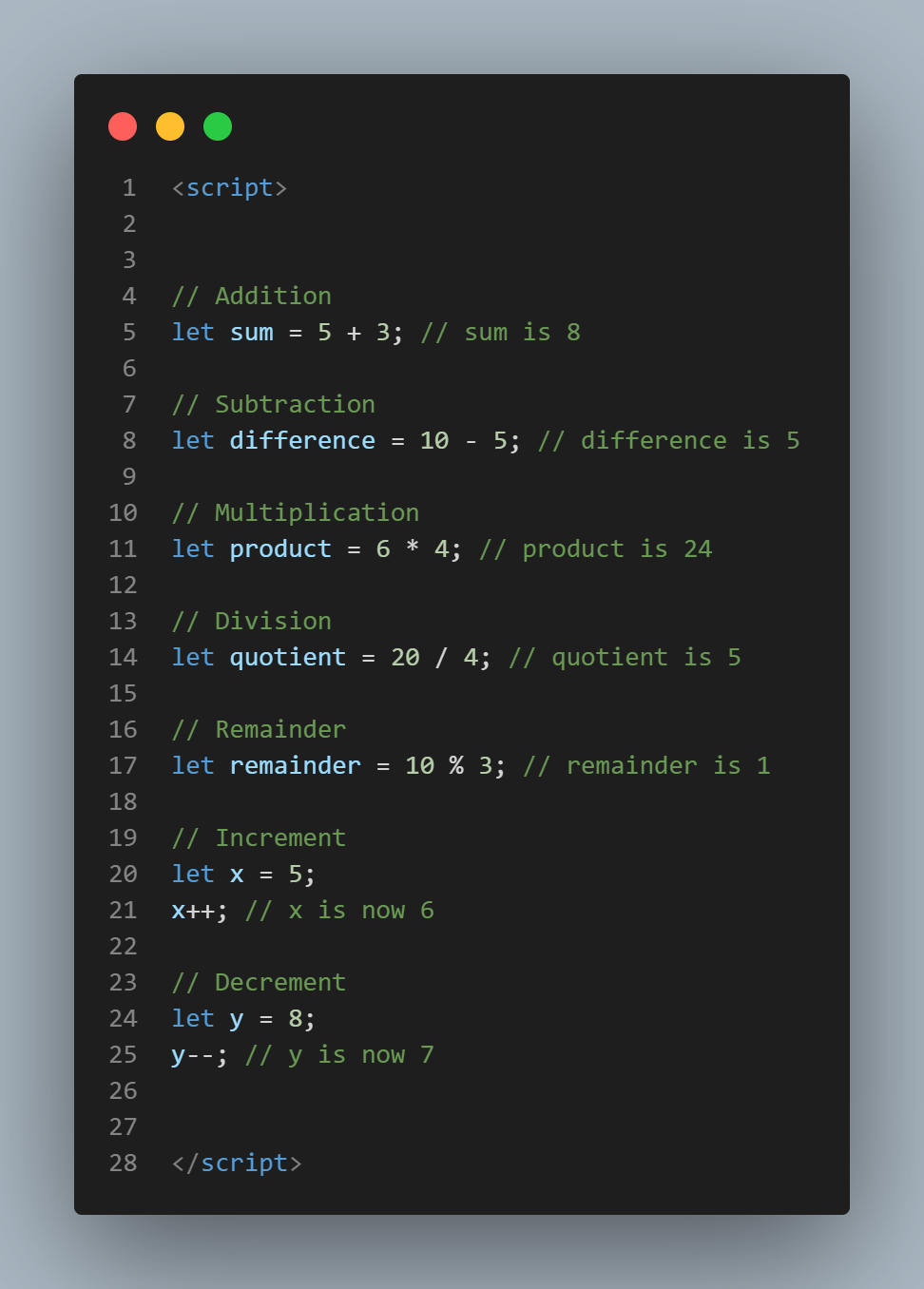
• Multiplication (\*)

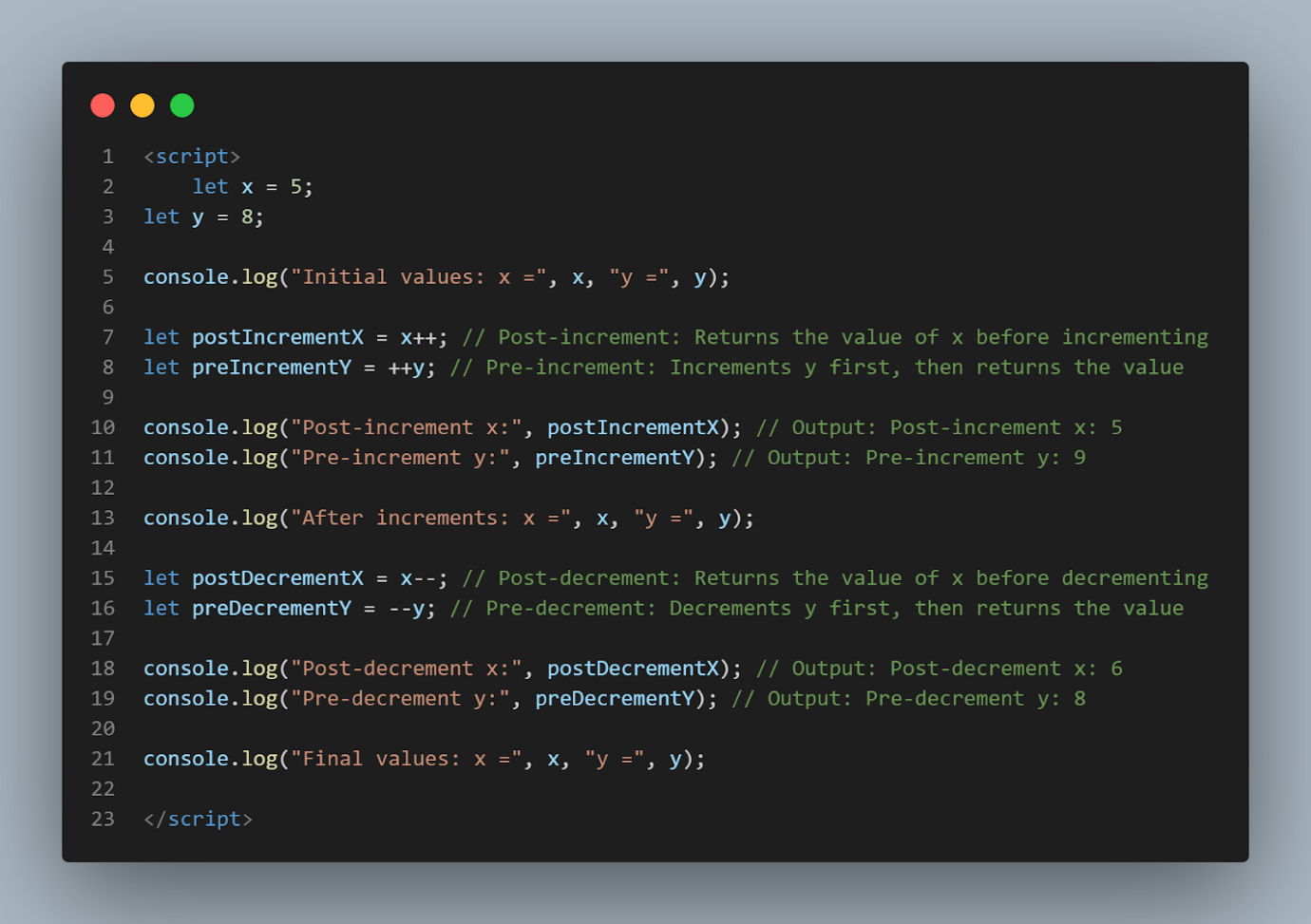
• Division (/)

• Remainder (%)

• Increment (++)

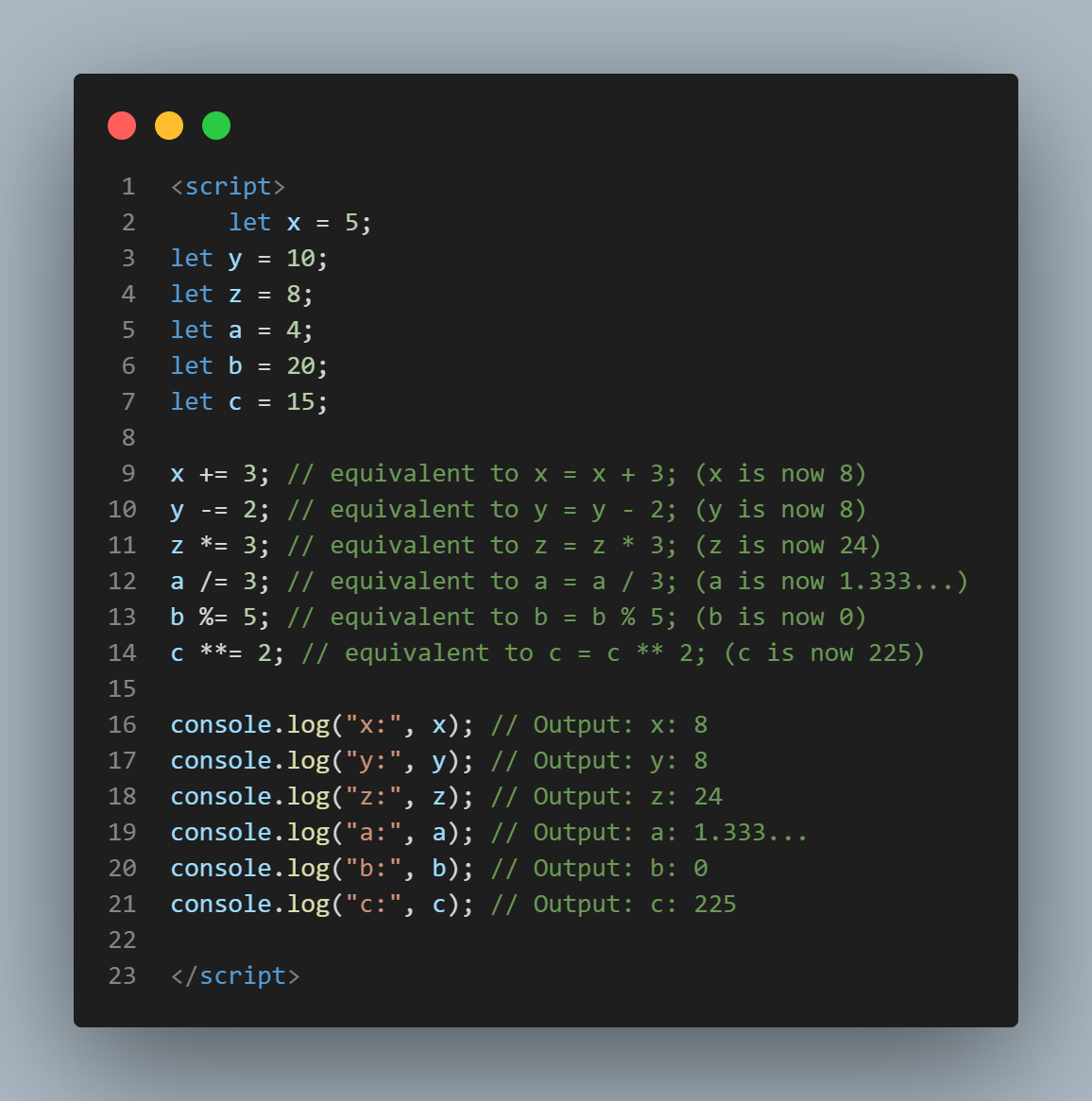
• Decrement (--)





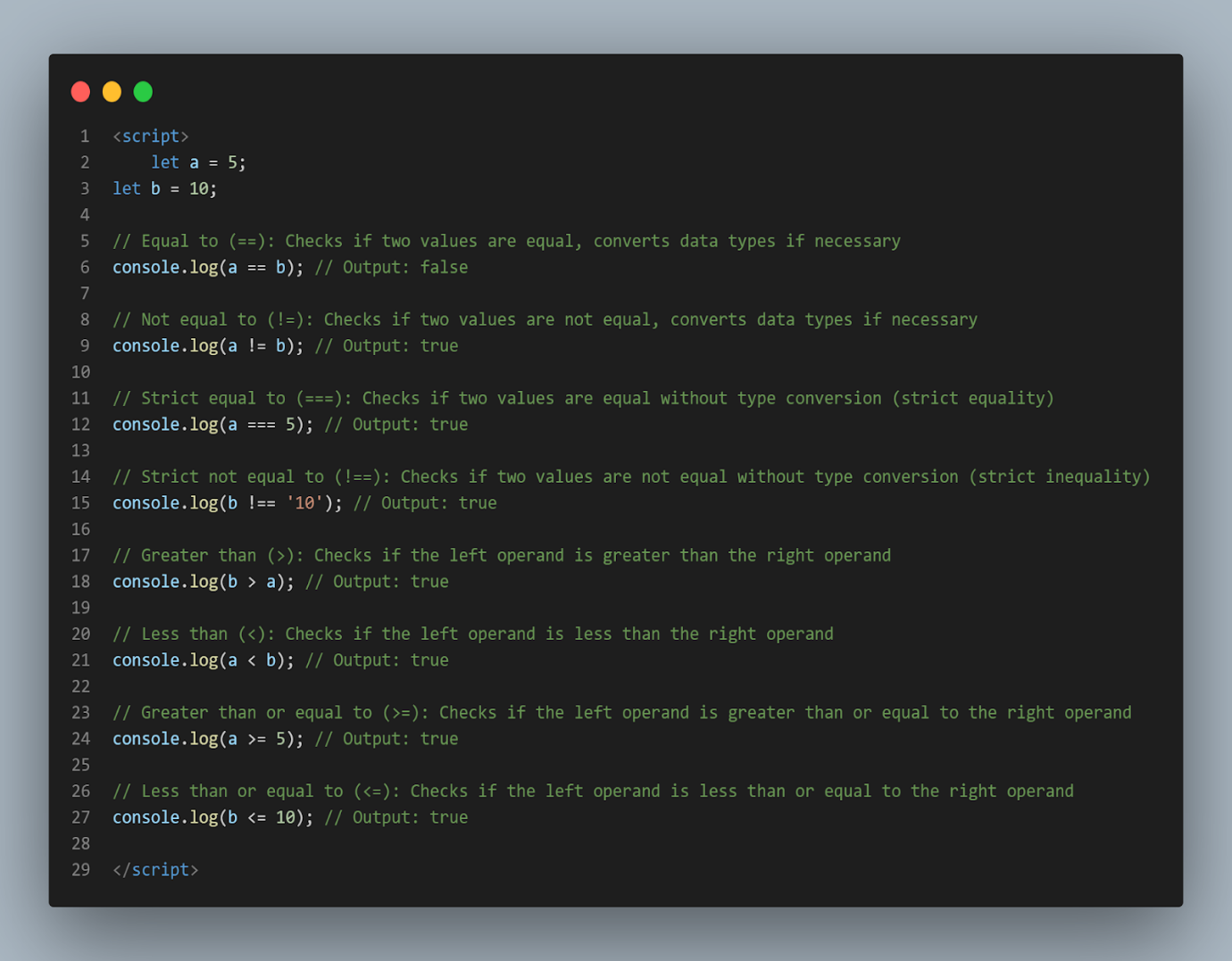
Assignment Operators:

Assignment operators in JavaScript are used to assign values to variables. They combine the operation of assigning a value to a variable with some other operation, such as addition, subtraction, multiplication, division, or modulus.



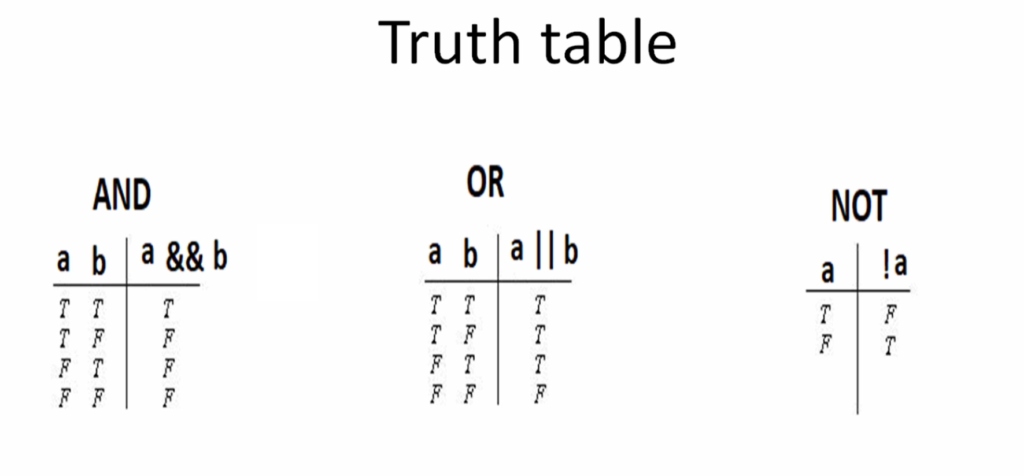
Comparison Operators

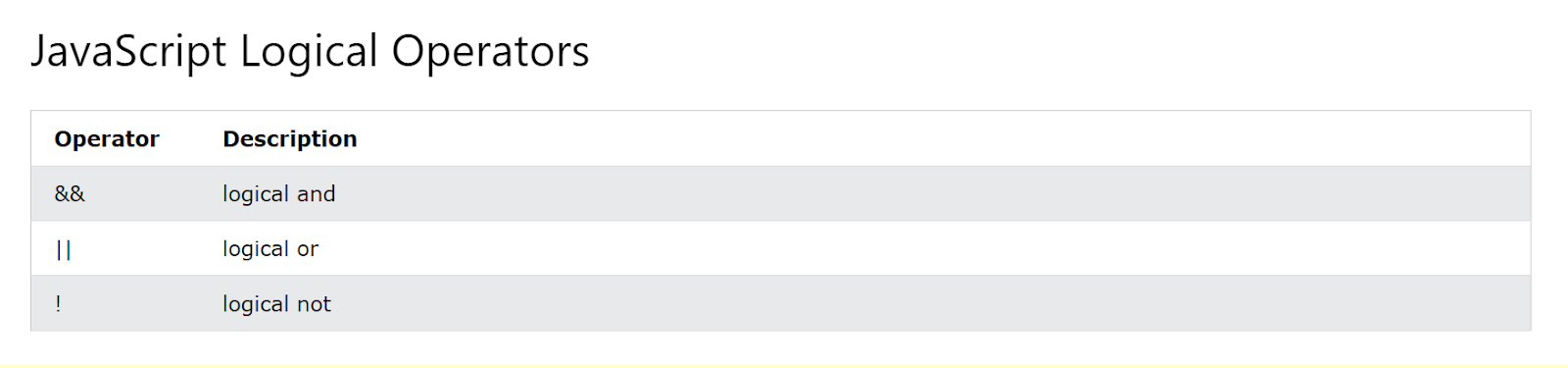
Comparison operators in JavaScript are used to compare two values and return a Boolean result

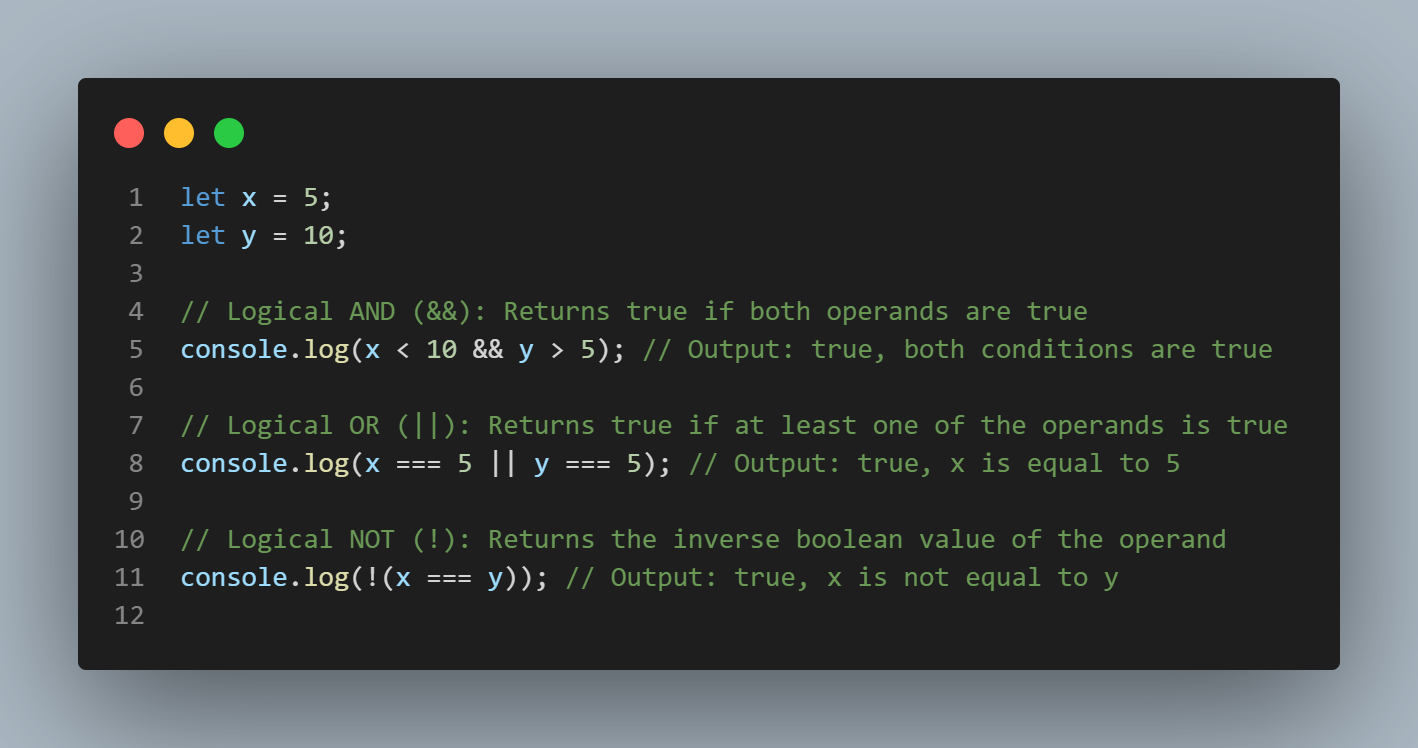


Logical operators:

Logical operators in JavaScript are used to combine or modify logical statements







In this snippet:

- `&&` (Logical AND) returns `true` if both conditions are true.

- `||` (Logical OR) returns `true` if at least one of the conditions is true.

- `!` (Logical NOT) returns `true` if the operand is false, and vice versa. It negates the boolean value of the operand.

string operators

In JavaScript, string operators are used to manipulate strings. The main string operator is the concatenation operator (`+`), which is used to concatenate or join two strings together. Here's an explanation of the string operator along with an example in one snippet:



In this snippet:

- The concatenation operator (`+`) is used to concatenate the `firstName` variable, a space, and the `lastName` variable, resulting in the full name "John Doe".

String operators are essential for manipulating and combining strings in JavaScript, enabling dynamic generation of textual content in web applications and scripts.

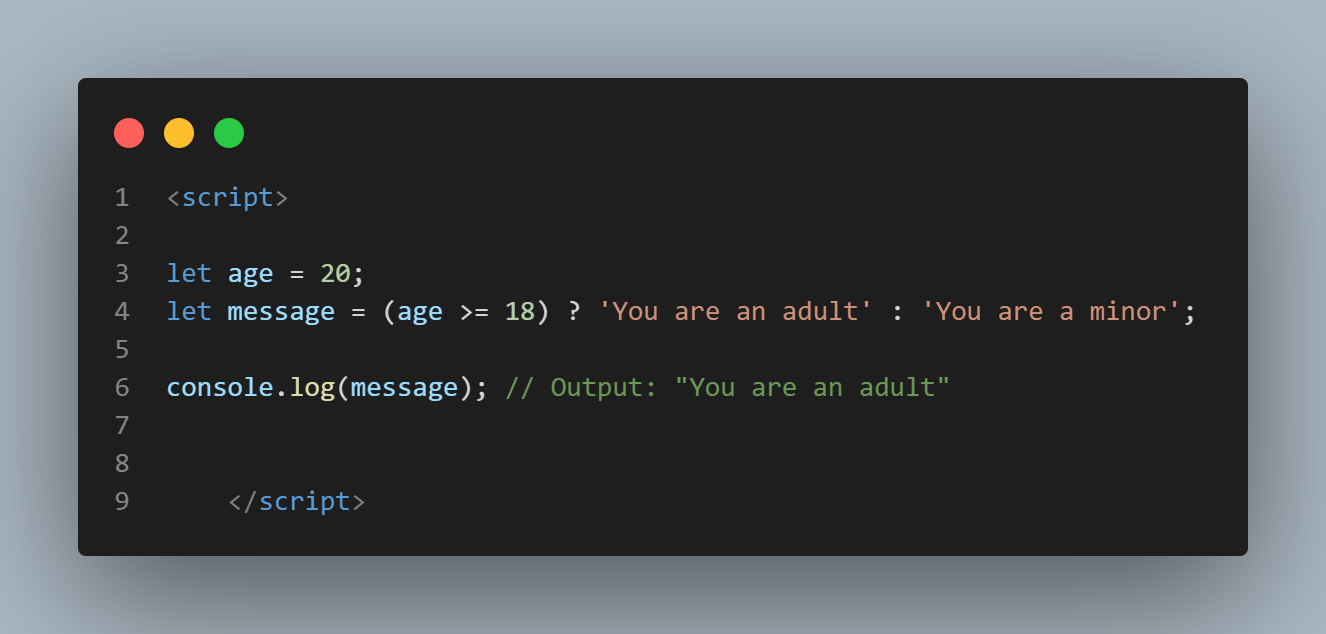
The ternary operator

also known as the conditional operator, provides a concise way to write conditional statements in JavaScript. Its syntax is:

condition ? expression1 : expression2

If the condition evaluates to true, the ternary operator returns `expression1`; otherwise, it returns `expression2`.

Here's an example demonstrating the ternary operator in action:



In this example:

- If the `age` variable is greater than or equal to 18, the condition `(age >= 18)` evaluates to true.

- Therefore, the ternary operator returns the value `'You are an adult'`, and this value is assigned to the `message` variable.

- If the condition is false, the ternary operator returns the value `'You are a minor'`.

The ternary operator is often used for assigning values to variables based on conditions, resulting in more concise and readable code compared to traditional if-else statements for simple conditions.

Type Operator:

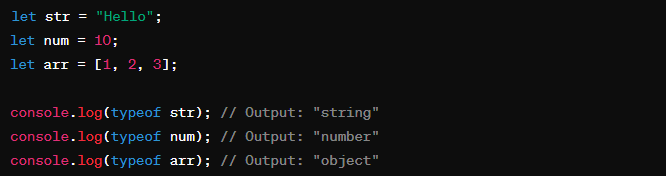
In JavaScript, there are two main type operators: `typeof` and `instanceof`. Let's explore each of them:

1. typeof Operator:

   - The `typeof` operator returns the data type of its operand.

   - It is often used to determine the type of a variable or expression.

   - The return value is a string indicating the type of the operand.

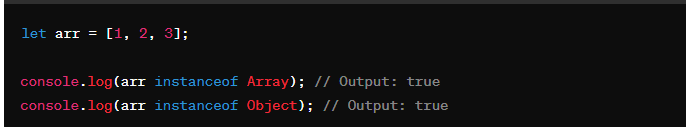


2. instanceof Operator:

   - The `instanceof` operator checks if an object is an instance of a specific object type.

   - It returns `true` if the object is an instance of the specified object type; otherwise, it returns `false`.

   - It is commonly used to check if an object is an instance of a particular class or constructor function.



   In this example, `arr` is an instance of both `Array` and `Object` because arrays are objects in JavaScript.

These type operators are useful for type checking and determining the nature of variables or objects in JavaScript code.