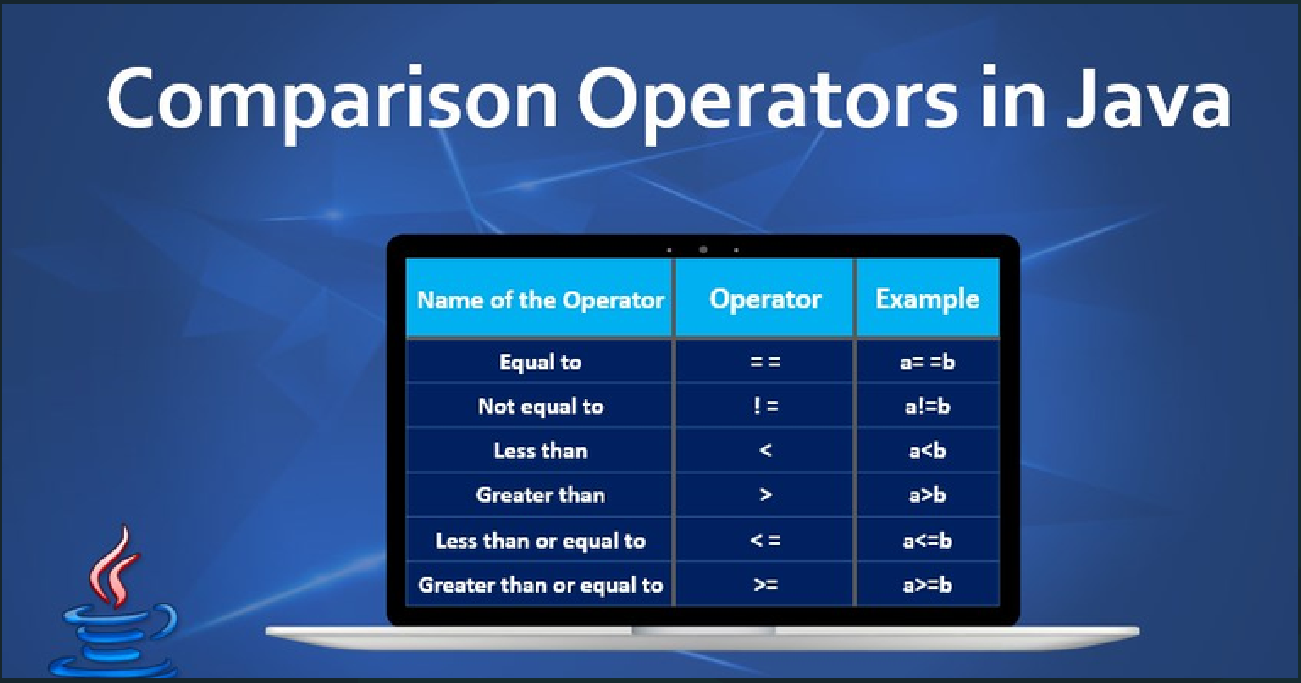
Introduction to comparison and logical operators

Comparison Operators

When writing code, we often need to include conditions to determine whether certain code should be executed. This is where comparison operators come into play, as they enable us to compare conditions and check their validity.


Comparison Operators in Java

Name of the Operator	Operator	Example
Equal to	= =	a= =b
Not equal to	! =	a!=b
Less than	<	a<b
Greater than	>	a>b
Less than or equal to	< =	a<=b
Greater than or equal to	>=	a>=b

- In Java, comparison operators are used to compare two values and determine whether they are equal, not equal, greater than, less than, greater than or equal to, or less than or equal to each other.
- These operators return a boolean value (true or false) depending on the outcome of the comparison

Here are the comparison operators in Java:

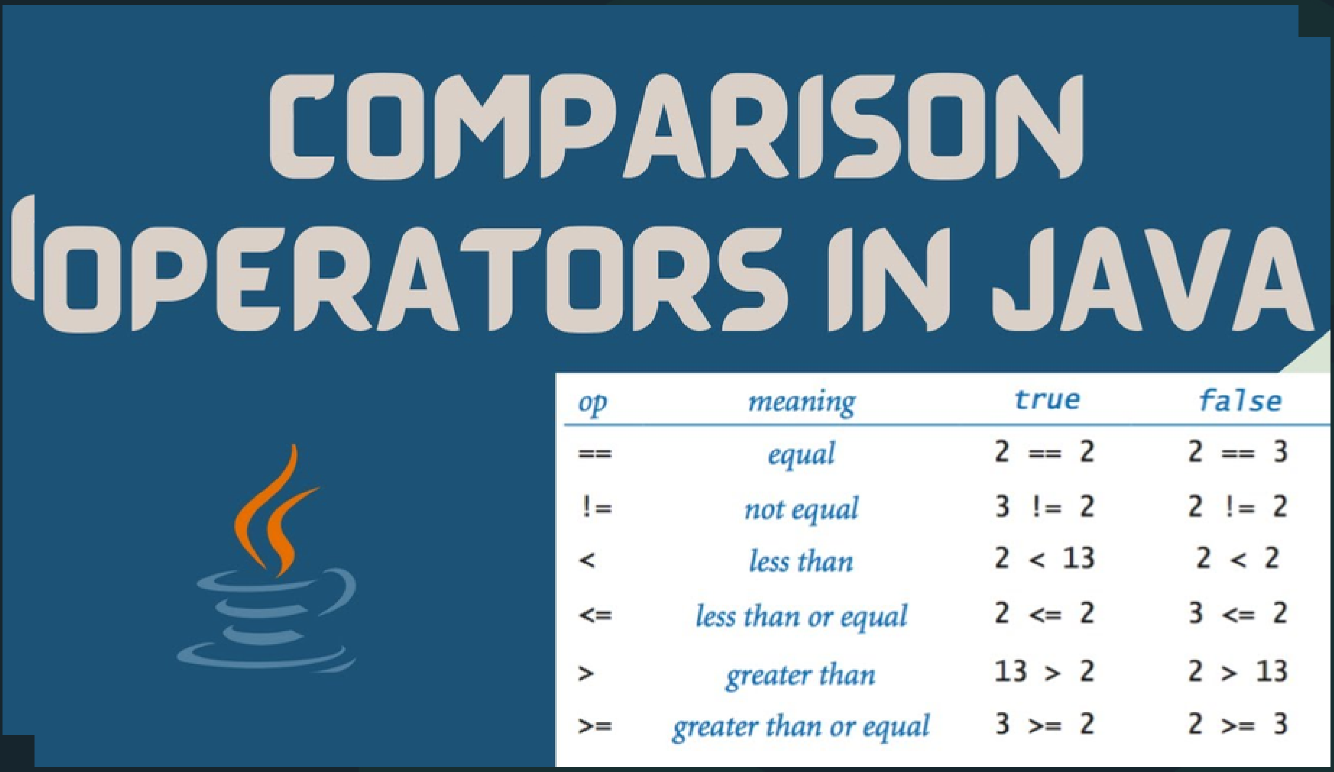
- Equal to (==): This operator compares two values and returns true if they are equal. For example, 2 == 2 would return true because 2 is equal to 2.

- Not equal to (!=): This operator compares two values and returns true if they are not equal. For example, 2 != 3 would return true because 2 is not equal to 3.

- Less than (<): This operator compares two values and returns true if the first value is less than the second value. For example, 2 < 3 would return true because 2 is less than 3.

- Greater than or equal to (>=): This operator compares two values and returns true if the first value is greater than or equal to the second value. For example, 3 >= 2 would return true because 3 is greater than 2, or equal to 2.

- Less than or equal to (<=): This operator compares two values and returns true if the first value is less than or equal to the second value. For example, 2 <= 3 would return true because 2 is less than 3, or equal to 3.


COMPARISON OPERATORS IN JAVA

op	meaning	true	false
==	equal	2 == 2	2 == 3
!=	not equal	3 != 2	2 != 2
<	less than	2 < 13	2 < 2
<=	less than or equal	2 <= 2	3 <= 2
>	greater than	13 > 2	2 > 13
>=	greater than or equal	3 >= 2	2 >= 3

Logical operators

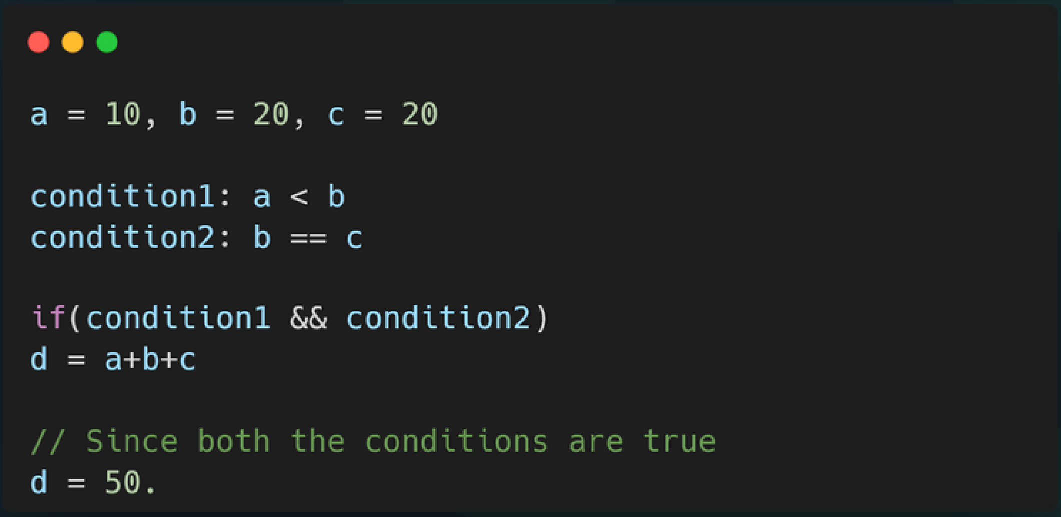
These operators are used to perform logical “AND”, “OR” and “NOT” operation, i.e. the function similar to AND gate and OR gate in digital electronics. They are used to combine two or more conditions/constraints or to complement the evaluation of the original condition under particular consideration.

One thing to keep in mind is the second condition is not evaluated if the first one is false, i.e. it has a short-circuiting effect. Used extensively to test for several conditions for making a decision. Let’s look at each of the logical operators in a detailed manner:

‘Logical AND’ Operator(&&):

This operator returns true when both the conditions under consideration are satisfied or are true. If even one of the two yields false, the operator results false. For example, cond1 && cond2 returns true when both cond1 and cond2 are true (i.e. non-zero).

Example:

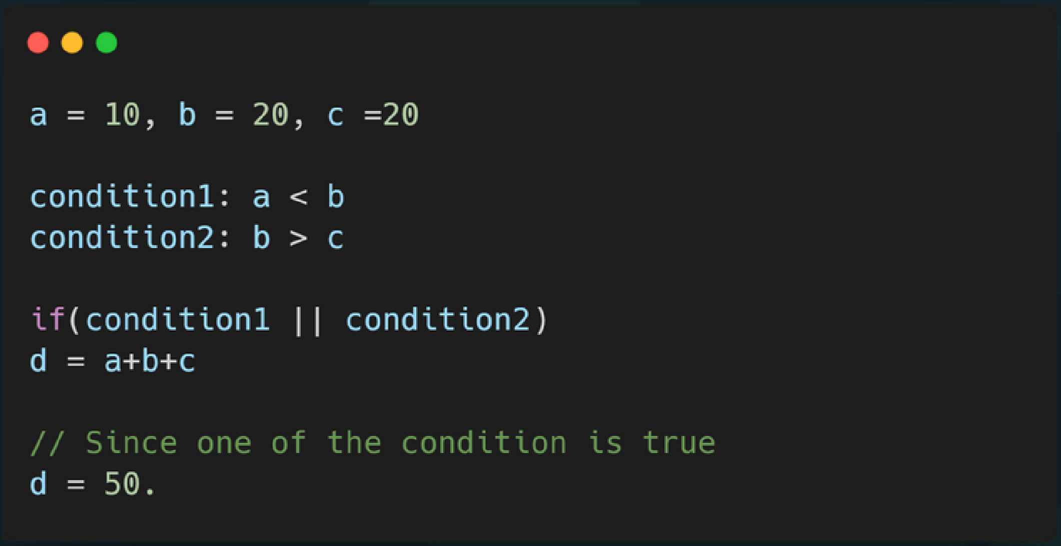


```
a = 10, b = 20, c = 20

condition1: a < b
condition2: b == c

if(condition1 && condition2)
d = a+b+c

// Since both the conditions are true
d = 50.
```

'Logical OR' Operator(||):

This operator returns true when one of the two conditions under consideration are satisfied or are true. If even one of the two yields true, the operator results true. To make the result false, both the constraints need to return false

Example:

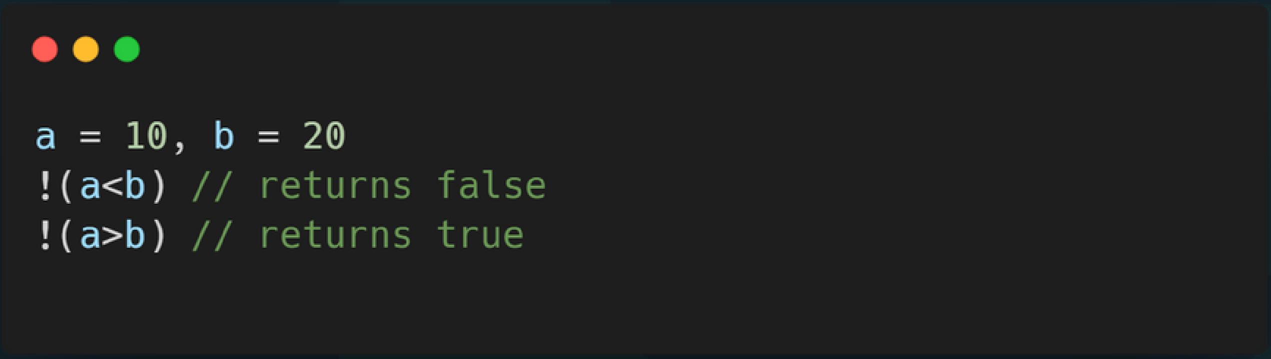


```
a = 10, b = 20, c =20

condition1: a < b
condition2: b > c

if(condition1 || condition2)
d = a+b+c

// Since one of the condition is true
d = 50.
```

'Logical NOT' Operator(!):

Unlike the previous two, this is a unary operator and returns true when the condition under consideration is not satisfied or is a false condition. Basically, if the condition is false, the operation returns true and when the condition is true, the operation returns false.

Example:



```
a = 10, b = 20
!(a<b) // returns false
!(a>b) // returns true
```