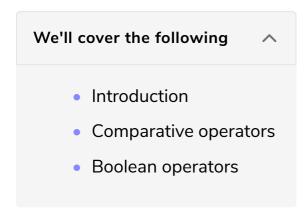
## **Logical Expressions**

In this lesson, an explanation of the use of logical expressions in Java programs is provided.



## Introduction #

Logical expressions are also known as **Boolean expressions**. It will always evaluate to a value of either **true** or **false**. Therefore, they will be represented by the data type **boolean**. While they may seem similar to mathematical operators, the difference lies in how they are used with **comparative** or **boolean operators**. Let's look at both types of operators in detail.

## Comparative operators #

Java has several operators that can be used to *compare* value. Comparison implies knowing which value is greater than the other, or equal to it, and so on. The table below shows the entire list of operators available in Java.

| Comparative operators    |
|--------------------------|
| Less than                |
| Greater than             |
| Less than or equal to    |
| Greater than or equal to |
|                          |

| == | Equal to     |
|----|--------------|
|    |              |
| != | Not equal to |

Note here that these comparative operators can be used on any **primitive data type** except for boolean.

Now that we have an idea of what these operators are let's see how we can use them in code.

```
class log op {
    public static void main(String[] args) {
        int x = 5;
        int y = 10;
        System.out.println("x is equal to: " + x);
        System.out.println("y is equal to: " + y);
        System.out.println("x is greater than y");
        System.out.println(x > y);
        System.out.println("x is less than y");
        System.out.println(x < y);
        System.out.println("x is greater than or equal to y");
        System.out.println(x \ge y);
        System.out.println("y is less than or equal to y");
        System.out.println(y <= y);</pre>
        System.out.println("x is equal to y");
        System.out.println(x == y);
        System.out.println("x is not equal to y");
        System.out.println(x != y);
    }
}
```







## Boolean operators #

These operators rely on **boolean algebra**. Hence, boolean operators will work directly on boolean values. There are **four** types of boolean operators. Let's first look at their symbols and what they are in the table below before we discuss what functionality they perform.

| Symbols | Boolean operators     |
|---------|-----------------------|
| !       | Boolean NOT           |
| &&      | Boolean AND           |
|         | Boolean OR            |
| ^       | Boolean exclusive XOR |

- The boolean NOT inverts the value of the boolean expression that follows it.
- The boolean AND will return **true** if and only if, expressions on both sides of the operator are **true**.
- The boolean OR will return **true** if the expression on either or both sides of the operator is **true**.
- The boolean exclusive XOR will return **true** if one of the expressions evaluates to **true** and the other to **false**.

Now that we have some understanding of these operators let's see how to use them in code.

```
class bool ops {
    public static void main(String[] args) {
        boolean x = true;
        boolean y = false;
        System.out.println("Value of x: " + x);
        System.out.println("Value of y: " + y);
        System.out.println("Boolean NOT of x");
        System.out.println(!x);
        System.out.println("Boolean AND of x and y");
        System.out.println(x && y);
        System.out.println("Boolean OR of x and y");
        System.out.println(x \mid\mid y);
        System.out.println("Boolean exclusive XOR of x and y");
        System.out.println(x ^ y);
    }
}
```





Now that we have understood the fundamentals of Maths and Logic in Java, we will do some practice before moving on to the next lesson!