# Lec 22 Operators:

This lesson covers the main types of operators in Java, including arithmetic, relational, logical, assignment, unary, ternary, instance of, and bitwise operators, as well as the basics of number systems (decimal, binary, octal, hexadecimal). Understanding these operators is essential for performing calculations, making decisions, and handling data in Java programs.

## Java Operators Overview

Java operators are special symbols or keywords used to perform operations on variables and values. They are fundamental for tasks like calculations, comparisons, and logic flow in programming.

## Arithmetic Operators

Arithmetic operators (+, -, *, /, %, ++, --) allow you to add, subtract, multiply, divide, find remainders, and increment or decrement values. They are used for basic mathematical operations in code.

## Relational Operators

Relational operators (==, !=, >, <, >=, <=) compare two values and return a boolean result (true or false). They are used for decision-making in conditions.

## Logical Operators

Logical operators (&& for AND, || for OR, ! for NOT) combine multiple boolean expressions. 'AND' returns true only if both conditions are true, 'OR' returns true if at least one is true, and 'NOT' inverts the result.

## Logical Operators in Practice

Using logical operators, you can check for complex conditions, such as eligibility based on age or matching multiple values. 'AND' requires all conditions to be met; 'OR' requires at least one; 'NOT' excludes specific cases.

## Assignment Operators

Assignment operators (=, +=, -=, *=, /=) assign or update the value of a variable. For example, 'x += 5' adds 5 to x. These shortcuts help update variables efficiently.

## Unary Operators

Unary operators (like ++, --, +, -, !) operate on a single operand. They are used for incrementing, decrementing, negating, or inverting a value.

## Ternary Operator

The ternary operator (condition ? value_if_true : value_if_false) is a compact way to write simple if-else logic in one line, returning one of two values based on a condition.

## Instanceof Operator

The 'instanceof' operator checks if an object is an instance of a particular class or subclass. It is useful for type checking before casting or using objects.

## Bitwise Operators Introduction

Bitwise operators (& for AND, | for OR, ^ for XOR, ~ for NOT, << for left shift, >> for right shift) manipulate individual bits in binary numbers. They are mainly used for low-level programming, such as optimizing performance or working with binary data.

## Bitwise AND and OR

Bitwise AND (&) results in 1 only if both bits are 1; OR (|) results in 1 if at least one bit is 1. These are applied to each bit of integer values.

## Bitwise XOR and NOT

XOR (^) returns 1 if bits are different; NOT (~) flips each bit. These can be used for toggling values or finding differences at the bit level.

## Right Shift and Left Shift Operators

Right shift (>>) moves bits to the right, effectively dividing by powers of two; left shift (<<) moves bits to the left, multiplying by powers of two. They are used for fast arithmetic or manipulating binary data.

## Java Number Systems

Java supports decimal, binary (prefix 0b), octal (prefix 0), and hexadecimal (prefix 0x) number systems. Knowing how to represent and convert between these is important for certain programming tasks.

## Summary and Next Steps

Mastering operators and number systems lays the foundation for deeper Java concepts like inheritance and polymorphism. Reviewing these basics helps in interviews and real-world programming.