# Bitwise Operators in Java - Notes

## Introduction

Bitwise operators in Java work on the binary representation of data. They can be applied to primitive data types: byte, short, int, and long.

## 1. AND Operator (&)

Performs bitwise AND operation. Each bit in the result is 1 if both corresponding bits are 1, else 0.
Example:
5 & 4 → (0101 & 0100) = 0100 = 4

## 2. OR Operator (|)

Performs bitwise OR operation. Each bit in the result is 1 if either of the corresponding bits is 1.
Example:
5 | 7 → (0101 | 0111) = 0111 = 7

## 3. XOR Operator (^)

Performs bitwise XOR operation. Each bit in the result is 1 if the corresponding bits are different.
Example:
5 ^ 7 → (0101 ^ 0111) = 0010 = 2

## 4. NOT Operator (~)

Bitwise NOT inverts all bits (1→0, 0→1).
Example:
~5 → -(5+1) = -6
Trick: ~x = -(x+1)

## 5. Left Shift (<<)

Shifts bits to the left by n positions, filling with 0s on the right.
Formula: x << n = x × (2^n)
Example:
5 << 1 = 10

## 6. Right Shift (>>)

Shifts bits to the right by n positions, keeping the sign bit (arithmetic shift).
Formula: x >> n = floor(x / 2^n)
Example:
5 >> 1 = 2
For negatives, it fills with 1 on the left.

## 7. Unsigned Right Shift (>>>)

Shifts bits to the right by n positions, filling with 0 (logical shift).
Always gives non-negative result.
Formula: x >>> n = (x mod 2^32) / 2^n
Example:
20 >>> 2 = 5
-20 >>> 2 = 1073741819

## Java Example Code

```
package BITWISE_OPERATOR;

public class Test {
  public static void main(String[] args) {
    // AND operator &
    int c = 5 & 4;
    System.out.println(c);

    // OR operator |
    int a = 5 | 7;
    System.out.println(a);

    // XOR operator ^
    int b = 5 ^ 7;
    System.out.println(b);

    // NOT operator ~
    int d = 5;
    System.out.println(Integer.toBinaryString(d));
    int e = ~d;
    System.out.println(e);
    System.out.println(Integer.toBinaryString(e));

    // Left shift <<
    int f = 5;
    int g = f << 1;
    System.out.println(g);

    // Right shift >>
    int h = 5;
    int i = h >> 1;
    System.out.println(i);

    // Unsigned right shift >>>
    int x = 20;
    int y = x >>> 2;
    System.out.println(y);
  }
}
```