**JAVA OPERATORS – Complete In-Depth Guide**

**🔹 1. Arithmetic Operators**

Used to perform basic mathematical operations.

| **Operator** | **Meaning** | **Example** | **Result** |
| --- | --- | --- | --- |
| + | Addition | a + b | Adds a and b |
| - | Subtraction | a - b | Subtracts b from a |
| \* | Multiplication | a \* b | Multiplies a and b |
| / | Division | a / b | Divides a by b |
| % | Modulus (Remainder) | a % b | Remainder after a / b |

🔸 **Example:**

int a = 10, b = 3;

System.out.println(a + b); // 13

System.out.println(a / b); // 3

System.out.println(a % b); // 1

📌 **Use Case:** Calculating marks, discounts, measurements.

**🔹 2. Relational (Comparison) Operators**

Used to compare values and return a boolean (true or false).

| **Operator** | **Meaning** | **Example** |
| --- | --- | --- |
| == | Equal to | a == b |
| != | Not equal to | a != b |
| > | Greater than | a > b |
| < | Less than | a < b |
| >= | Greater than or equal to | a >= b |
| <= | Less than or equal to | a <= b |

🔸 **Example:**

int x = 5, y = 10;

System.out.println(x > y); // false

System.out.println(x != y); // true

📌 **Use Case:** Login validation, eligibility checks, sorting.

**🔹 3. Logical Operators**

Used to combine multiple boolean expressions.

| **Operator** | **Meaning** | **Example** | **Result** |
| --- | --- | --- | --- |
| && | Logical AND | a > 5 && b < 10 | true if both true |
| ` | || | Logical OR |  |
| ! | Logical NOT | !true | false |

🔸 **Example:**

int age = 25;

System.out.println(age > 18 && age < 30); // true

System.out.println(!(age < 18)); // true

📌 **Use Case:** Conditions, forms, filters, login.

**🔹 4. Assignment Operators**

Used to assign or update values to variables.

| **Operator** | **Meaning** | **Example** | **Same As** |
| --- | --- | --- | --- |
| = | Assign | a = 10 | - |
| += | Add and assign | a += 5 | a = a + 5 |
| -= | Subtract and assign | a -= 3 | a = a - 3 |
| \*= | Multiply and assign | a \*= 2 | a = a \* 2 |
| /= | Divide and assign | a /= 4 | a = a / 4 |
| %= | Modulus and assign | a %= 3 | a = a % 3 |

🔸 **Example:**

int a = 10;

a += 5; // a = 15

a \*= 2; // a = 30

System.out.println(a);

📌 **Use Case:** Updating counters, progress, scores, etc.

**🔹 5. Unary Operators**

Work with only one operand.

| **Operator** | **Meaning** | **Example** | **Result** |
| --- | --- | --- | --- |
| + | Unary plus (usually ignored) | +a | a |
| - | Unary minus | -a | Negative of a |
| ++ | Increment | ++a or a++ | Adds 1 |
| -- | Decrement | --a or a-- | Subtracts 1 |
| ! | Logical NOT | !true | false |

✅ **Pre vs Post Increment:**

int x = 5;

System.out.println(++x); // 6 (pre-increment)

System.out.println(x++); // 6 (prints then increments to 7)

📌 **Use Case:** Loop counters, toggles, booleans.

**🔹 6. Bitwise Operators *(Advanced – Optional for now)***

These deal with bits (0s and 1s).

| **Operator** | **Meaning** | **Example** |
| --- | --- | --- |
| & | Bitwise AND | a & b |
| ` | ` | Bitwise OR |
| ^ | Bitwise XOR | a ^ b |
| ~ | Bitwise NOT | ~a |
| << | Left shift | a << 2 |
| >> | Right shift | a >> 2 |
| ****Common Bitwise Operators in Java****  | **Operator** | **Name** | **Symbol** | **Example** | **Description** | | --- | --- | --- | --- | --- | | AND | Bitwise AND | & | a & b | 1 if **both** bits are 1 | | OR | Bitwise OR | ` | ` | `a | | XOR | Bitwise XOR | ^ | a ^ b | 1 if **only one** bit is 1 (not both) | | NOT | Bitwise NOT | ~ | ~a | Inverts all bits (flips 0 → 1 and 1 → 0) | | Left Shift | Shift left | << | a << n | Shifts bits left by n positions | | Right Shift | Shift right | >> | a >> n | Shifts bits right by n positions |  🔢 ****Let’s Calculate With an Example****Example: int a = 5; // Binary: 0101  int b = 3; // Binary: 0011 1️⃣ a & b (AND) 0101  & 0011  -------  0001 => Decimal: 1 2️⃣ a | b (OR) 0101  | 0011  -------  0111 => Decimal: 7 3️⃣ a ^ b (XOR) 0101  ^ 0011  -------  0110 => Decimal: 6 4️⃣ ~a (NOT) a = 5 = 00000000 00000000 00000000 00000101  ~a = 11111111 11111111 11111111 11111010 => -6 (2’s complement) 5️⃣ a << 1 (Left Shift) a = 5 (0101)  << 1 → 1010 => Decimal: 10 6️⃣ a >> 1 (Right Shift) a = 5 (0101)  >> 1 → 0010 => Decimal: 2 |  |  |

**📘 Real-Life Examples:**

* **Arithmetic:** Grocery bill calculation.
* **Relational:** Is student eligible for scholarship?
* **Logical:** Login success only if username AND password are correct.
* **Assignment:** Game score updates.
* **Unary:** Level up/down.
* **Bitwise:** Used in encryption, compression, optimization.

**✅ Java Operator Examples**

**🔹 1. Arithmetic Operators**

public class ArithmeticDemo {

public static void main(String[] args) {

int a = 12, b = 5;

System.out.println("a + b = " + (a + b)); // 17

System.out.println("a - b = " + (a - b)); // 7

System.out.println("a \* b = " + (a \* b)); // 60

System.out.println("a / b = " + (a / b)); // 2

System.out.println("a % b = " + (a % b)); // 2

}

}

**🔹 2. Relational (Comparison) Operators**

public class RelationalDemo {

public static void main(String[] args) {

int x = 10, y = 20;

System.out.println("x == y: " + (x == y)); // false

System.out.println("x != y: " + (x != y)); // true

System.out.println("x > y: " + (x > y)); // false

System.out.println("x < y: " + (x < y)); // true

System.out.println("x >= y: " + (x >= y)); // false

System.out.println("x <= y: " + (x <= y)); // true

}

}

**🔹 3. Logical Operators**

public class LogicalDemo {

public static void main(String[] args) {

int age = 22;

boolean hasID = true;

System.out.println(age > 18 && hasID); // true

System.out.println(age < 18 || hasID); // true

System.out.println(!(age > 18)); // false

}

}

**🔹 4. Assignment Operators**

public class AssignmentDemo {

public static void main(String[] args) {

int a = 10;

a += 5; // a = 15

System.out.println("a += 5: " + a);

a -= 3; // a = 12

System.out.println("a -= 3: " + a);

a \*= 2; // a = 24

System.out.println("a \*= 2: " + a);

a /= 4; // a = 6

System.out.println("a /= 4: " + a);

a %= 4; // a = 2

System.out.println("a %= 4: " + a);

}

}

**🔹 5. Unary Operators**

public class UnaryDemo {

public static void main(String[] args) {

int a = 5;

System.out.println("a = " + a); // 5

System.out.println("++a = " + ++a); // 6

System.out.println("a++ = " + a++); // 6 (then a becomes 7)

System.out.println("a = " + a); // 7

System.out.println("--a = " + --a); // 6

System.out.println("a-- = " + a--); // 6 (then a becomes 5)

System.out.println("a = " + a); // 5

boolean flag = false;

System.out.println("!flag = " + !flag); // true

}

}

**🔹 6. Bitwise Operators *(Bonus Example - optional for now)***

public class BitwiseDemo {

public static void main(String[] args) {

int a = 5, b = 3;

System.out.println("a & b = " + (a & b)); // 1

System.out.println("a | b = " + (a | b)); // 7

System.out.println("a ^ b = " + (a ^ b)); // 6

System.out.println("~a = " + (~a)); // -6

System.out.println("a << 1 = " + (a << 1)); // 10

System.out.println("b >> 1 = " + (b >> 1)); // 1

}

}