# Operators

* Operators are special symbols that perform specific operations on one, two, or three *operands*, and then return a result.
* Operators with higher precedence are evaluated before operators with relatively lower precedence.
* Operators on the same line have equal precedence
* The operators in the following table are listed according to precedence order.

|  |  |
| --- | --- |
| **Operator Precedence** | |
| **Operators** | **Precedence** |
| postfix | *expr*++ *expr*-- |
| unary | ++*expr* --*expr* +*expr* -*expr* ~ ! |
| multiplicative | \* / % |
| additive | + - |
| shift | << >> >>> |
| relational | < > <= >= instanceof |
| equality | == != |
| bitwise AND | & |
| bitwise exclusive OR | ^ |
| bitwise inclusive OR | | |
| logical AND | && |
| logical OR | || |
| ternary | ? : |
| assignment | = += -= \*= /= %= &= ^= |= <<= >>= >>>= |

## **Simple Assignment Operators:**

* Most common operator used to assign to assign a value on its right to the operand on its left.
* Eg: int cadence = 0; int speed = 0; int gear = 1;
* Can also be used to assign object references

**Arithmetic operators:**

* Operators that perform addition, subtraction, multiplication and division.
* **% -** is a modulo operator which divides one operand by another and returns remainder as result.

|  |  |
| --- | --- |
| **Operator** | **Description** |
| + | Additive operator (also used for String concatenation) |
| - | Subtraction operator |
| \* | Multiplication operator |
| / | Division operator |
| % | Remainder operator |

* **+** operator can also be used for concatenation of String objects.

**Unary Operators:**

* Unary operator requires only one operand.
* Performs operations like incrementing/decrementing a value by one, negating an expression, or inverting the value of a boolean

|  |  |
| --- | --- |
| **Operator** | **Description** |
| + | Unary plus operator; indicates positive value (numbers are positive without this, however) |
| - | Unary minus operator; negates an expression |
| ++ | Increment operator; increments a value by 1 |
| -- | Decrement operator; decrements a value by 1 |
| ! | Logical complement operator; inverts the value of a boolean |

**Equality and Relational operators:**

* Equality and relational operators determine if one operand is greater than, less than, equal to, or not equal to another operand.
* **‘==’** is not equal to **’=’**

|  |  |
| --- | --- |
| **Operator** | **Description** |
| == | Equal to |
| != | Not equal to |
| > | Greater than |
| < | Less than |
| >= | Greater than or Equal to |
| <= | Less than or Equal to |

**Conditional Operators:**

* && and || operators perform *Conditional-AND* and *Conditional-OR* operations on two boolean expressions
* Exhibit short-circuiting behavior which means second operator executes only when needed.
* && Conditional-AND
* || Conditional-OR
* ? : Ternary operator (if-then-else). Uses 3 operands
* Ternary operator syntax :
  + (condition) ? (value if true) : (value if false)

## **Type Comparison Operator:**

* ***instanceOf***  compares an object to specified type.
* Eg : obj instanceOf Parent

**Bitwise and Bit shift operators:**

* Performs operations on bits instead of numbers.

|  |  |
| --- | --- |
| **Operator** | **Description** |
| ~ | Negation |
| & | Bitwise AND |
| | | Bitwise OR |
| ^ | Bitwise XOR |
| >> | Right shift |
| << | Left shift |
| >>> | unsigned right shift operator |

**Examples :**

**a = 10, b = 5**

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Operation** | **Expression** | **Result** |
| + | Addition | a + b | 15 |
| - | Subtraction | a - b | 5 |
| \* | Multiplication | a \* b | 50 |
| / | Division | a / b | 2 |
| % | Modulus | a % b | 0 |
| -- | Decrement | a--, --a | 9 |
| ++ | Increment | a++, ++a | 11 |
| += | Addition assignment | a += b (same as a = a + b) | 15 |
| -= | Subtraction assignment | a -= b (same as a = a - b) | 5 |
| \*= | Multiplication assignment | a \*= b (same as a = a \* b) | 50 |
| /= | Division assignment | a /= b (same as a = a / b) | 2 |
| %= | Modulus assignment | a %= b (same as a = a % b) | 0 |

**a = 0 , b = 1**

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Operation** | **Expression** | **Result** |
| & | Bitwise AND | a & b | 0 |
| | | Bitwise OR | a | b | 1 |
| ^ | Bitwise exclusive OR | a ^ b | 1 |
| << | Shift left | a<< b | 0 |
| >> | Shift right | a >> b | 0 |
| ~ | Bitwise unary NOT | ~a | 1 |
| &= | Bitwise AND assignment | a &= b is same as a = a & b | a=0 |
| |= | Bitwise OR assignment | a |= b is same as a = a | b | a=1 |
| ^= | Bitwise exclusive OR assignment | a ^= b is same as a = a ^ b | a=1 |
| >>= | Shift right assignment | a >>= b is same as a = a >> b | a=0 |
| <<= | Shift left assignment | a <<= b is same as a = a << b | a=0 |

**Truth table for bit wise operators**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **a** | **b** | **a&b** | **a|b** | **a^b** | **~a** |
| 1 | 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 1 | 1 |
| 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 0 | 0 |

**a=10 b= 20**

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Operation** | **Expression** | **Result** |
| < | Less than | a<b | true |
| > | Greater than | a>b | false |
| <= | Less than or equal to | a<=b | true |
| >= | Greater than or equal to | a>=b | false |
| == | Equal to | a==b | false |
| != | Not equal to | a!=b | true |

**Ternary operator :**

int x = 5;

int a = 6;

int resultValue = (x > a) ? a : x; // Calculates the minimum of 2 numbers, resultValue will be 5.

a=true and b=false

|  |  |  |  |
| --- | --- | --- | --- |
| Operator | Operation | Expression | Result |
| || | Short Circuit OR | a || b | true |
| && | Short Circuit AND | a&&b | false |
| ! | NOT | !a | false |