### **Operator:**

Operator is a symbol that performs certain operations.

Java provides the following set of operators

- 1. Arithmetic operators
- 2. Increment and decrement operators
- 3. Relational or comparison operators
- 4. Bitwise operators, Shift operators
- 5. Short circuit Logical operators
- 6. Assignment operators
- 7. Conditional (?:) operator

#### 1. Arithmetic operators:

+	addition
-	subtraction
*	multiplication
/	Division operator
%	Modulo operator

If we apply any arithmetic operation b/w 2 variables a & b, the result type is always

## max ( int, type of a , type of b)

Ex:

byte + byte = int

byte + short = int

short + short = int

short + long = long

double + float = double

int + double = double

char + char = int

char + int = int

char + int = int

char + double = double

In integral arithmetic (byte, int, short, long) there is no way to represent infinity, if infinity is the result we will get the ArithmeticException/ by zero

System.out.println(10/0); //output ArithmeticException/ by zero

But in floating point arithmetic (float,double), there is a way represents infinity.

System.out.println(10/0.0);
//output: infinity

#### **Arithmetic exception:**

- 1. It is a RuntimeException but not compile time error
- 2. It occurs only in integral arithmetic but not in floating point arithmetic.
- 3. The only operations which cause ArithmeticException are: '/' and '%'

# 2. Increment & decrement operators:

Increment	Pre-	Ex:	y = ++x;
operator	increment		
	Post-	Ex:	y = x++;
	increment		

Decrement	Pre-	Ex: $y =x$ ;
operator	decrement	
	Post-	Ex: $y = x;$
	decrement	

The following table will demonstrate the use of increment and decrement operators.

<b>Expression</b>	<b>Initial</b>	<b>Value</b>	<u>Final</u>
	<u>value</u>	of y	<u>value</u>
	of x		of x
y = ++x	20	21	21
y = x++	20	20	21
y =x	20	19	19
y= x	20	20	19

- 1. Increment and decrement operators we can apply <u>only for variables</u> but not for constant values. Otherwise we will get compile time error.
- 2. We can apply increment or decrement operators even for primitive data types <u>except</u> boolean.

#### 3. Relational operators

$$(<,<=,>,>=)$$

We can apply relational operators for every primitive type except <u>boolean</u>.

#### Ex:

System.out.println(10>10.5); //false

#### 4. Equality operators:

We can apply equality operators for every primitive type including boolean type also

Ex:

System.out.println(10==20); //false

#### **5. Shift operators:**

#### << left shift operator:

After shifting the empty cells we have to fill with zero.

System.out.println(10<<2);

//40

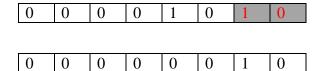
0 0 0	0	1	0	1	0
-------	---	---	---	---	---

0	0	1	0	1	0	0	0

### >> right shift operator :

After shifting the empty cells, we have to fill with sign bit. (0 for +ve and 1 for -ve)

System.out.println(10>>2); //2



#### 6. Bitwise operators(& ,| ,^)

operator	Description
&(AND)	If both arguments are true
	then only result is true.
(OR)	if atleast one argument is true.
	Then the result is true.
^( <b>X-OR</b> )	if both are different
	arguments. Then the result is
	true
~1	Bitwise complement operator
	i.e, 1 means 0 and 0 means 1

Ex:

System.out.println(true&false); //false

We can apply bitwise operators even for integral types also.

System.out.println(4&5); //4

# **Bitwise complement operator:**

**(~**)

We can apply this operator only for integral types but not for boolean types.

We have to apply complement for total bits.

Ex:

System.out.println(~4); //-5
------------------------------

<u>Note:</u> The most significant bit acts as sign bit. O value represents +ve number where as 1 represents -ve value

Positive numbers will be represented directly in the memory where as –ve numbers will be represented indirectly in 2's complement form

# **Boolean complement operator** (!)

This operator is applicable only for Boolean types but not for integral types

System.out.println(!true);

//false

# 7. Short circuit logical operators (&& , || )

Applicable only for Boolean types but not for integral types.

x&&y: y will be evaluated if and only if x is true.

 $\mathbf{x} \parallel \mathbf{y} : \mathbf{y}$  will be evaluated if and only if  $\mathbf{x}$  is false.

#### 8. Assignment operator:

There are 3 types of assignment operators

#### 1. simple assignment:

Example:

$$int x = 10;$$

#### 2. chained assignment:

Example:

#### **3.compound assignment:**

Example:

#### 9. Conditional operator(?:)

The only possible ternary operator in java is conditional operator.

Syntax:

X=firstValue	if	condition	else
secondValue			

If condition is True then firstValue will be considered else secondValue will be considered

Ex 1:

int x=(10>20)?30:40;		
System.out.println(x);	//40	

Note: nesting of ternary operator is possible.

**operator:** we can use this operator to declare under construct/ create arrays.

## Java operator precedence:

1. unary	Highest
	(),[],
	++,, ~,!
2. Arithmetic	*, /, %,+,-
3. Shift	>>,>>>,<<
4. Relational	<,<=,>,>=
5. Equality	==,!=
6. Bitwise	&, ^,
7. Short circuit	&&,
logical	
8. Conditional	?:
9. Assignment	=
	Lowest