7. Operators

$$\underbrace{\overset{\text{operands}}{\text{sum}} = a + b}$$

sum = a + b → operator expression

operands operands

Here operator is ADDITION

Types of Operator

① Arithmetic Operators (Binary / Unary)
② Relational "
③ Logical "
④ Bitwise " (Bit Manipulation)
⑤ assignment "

① Arithmetic Operators

Binary [2 operands]

- + → a+b
- − → a-b
- x → a*b
- / → a/b
- % (modulo) → A%B

for remainder

eg 10/5 = 2
10%5 = 0 [Remainder]
5%3 = 2

Unary [single operands]

- ++ ⇒ a = a+1
- -- ⇒ a = a-1

Meaning

a=a+1 → a++ or ++a
a=a-1 → a-- or --a

- ++ → increment operator

Pre increment
पहले ++a
1. value change
then, 2. value use होगा

Post increment
बाद मे a++
1. value use
2. value change

++

[++a]
Pre Increment

```
Int a = 10;
Int b = ++a;
System.out.println(a);
System.out.println(b);
```

Output: 11
11

[a++]
Post Increment

```
Int a = 10;
Int b = a++;
System.out.println(a);
System.out.println(b);
```

Output: 11
10

```
Int a = 10;
Int b = ++a;
```

a b
|~~10~~| | |11|
11

In this wherever the value of 'a' is stored in memory will change first then the same value will assign to b and print

① पहले 'a' खुद change होगा
② फिर वो value 'b' मे store करने के लिए use होगा |

Memory मे a = 10 assign है फिर

|~~10~~| | |10|
a b
11

Java 'b' ~~[illegible]~~ बनेगा और उसके अंदर a की जो पुरानी value वो store होगा फिर Increment होगा |

```
Int b = a++;
```

then a Increment

--

[Pre Decrement]
↓
[--a]
↓
same as above

```
Int a = 10;
Int b = ~~[illegible]~~ --a;
System.out.println(a);
System.out.println(b);
```

Output: 9
9

[Post Decrement]
↓
[a--]
↓
same as above.
↓

```
Int a = 10;
Int b = ~~a++~~ a--;
System.out.println(a);
System.out.println(b);
```

Output: 9
10

This operator is used to check if something equal or not.

Relational Operators

```
a = 10;
b = 5;
System.out.println(a == b);
  Output: false
```

1. == [equal to equal to]
2. != [not equal to]
3. > [greater than]
4. <
5. >= [Greater than equal to]
6. <=

(2) not equal to or inequality

```
a = 10;                     a = 10;
b = 5;                      b = 10;

sout(a != b);               Sout(a != b);

Output: true                Output: false
```

This operator is opposite of '=='.

(3) >

```
int a = 10;                 int a = 10;
int b = 5;                  int b = 5;
Sout(a > b);                Sout(b > a);
Output: true                Output: false
```

(4) Works same as above (3).

(5) >= [Greater than equal to]

```
int a = 10;
int b = 10;
Sout(a > b);          Sout(a >= b)
Output: false         Output: true
```

To check logic

Logical Operator

① && (Logical AND)

Ans	Statement①	Statement②
T	T	T
F	T	F
F	F	T
F	F	F

```
System.out.print((3>2) && (5>0));
```

Output : true

```
System.out.println((3<2) && (5<0));
```

Output: false

(Stt1) && (Stt2)
T T → True

② || (Logical OR)

Ans	Stt①	Stt②
T	T	F
T	F	T
T	T	T
F	F	F

```
System.out.println((3<2) || (5<0));
```

Output : false

Only false when both statements are false.

③ ! (Logical Not)

Not operator

Ans	Stt
F	T
T	F

```
System.out.println(!(3>2));
```

Output: false

Assignment Operator

1) = A = B [Java assign right value to left)

2) +=

3) -=

4) *=

5) /=

2) +=

A = A + 10 [operand is same]

↓

A+ = 10

or operand should same only that we use this.

3) -=

B = B-5

↓

B- = 5

```
1   A = 10;
2   A = A+10;
3   A+= 10;  // Output is as same as line 2 but Java execute
             it fast (some nano seconds).
```

And like this other assignment operators also works same.

* Operater Precedence

↓

Check Pdf for table.