Operators:

Operators in Java refers to the **Special Symbols.**
These **Specials Symbols** are generally used for performing **Operations** in the program.
The **Operation** may be **Arithmetic or Logic** based on the **Condition**

1.Arthematic Operators
Addition **+**
Subtraction –
Multiplication *
Division **/**
Modulus **%**

2.Unary Operators
Increment **++**
Decrement **–**

3.Assignment Operators assignment
=
+=
-=
***=**
/=
%=

4.Relational Operators (Comparison Operators)

equal to **= =**
not equal to **!=**
greater than **>**
less than **<**
greater than equal to >=
less than equal to **<=**

6.Non Short Circuit Operators / Bitwise

& (bitwise AND)
| (bitwise OR)
^ (bitwise exclusive OR)
~ (bitwise complement)
<< Left Shift Operator
>> Right Shift Operator

5.Short Circuit Operators / Logical

logical AND **&&**
logical OR ||
logical NOT !

7.Ternary Operators

ternary **? :**

Separators:

Separators are the **symbols** which are used to **divide** or **arrange** the **code**.
These are **predefined symbols** which generally used to give the **shape of our function or the program.**

Parenthesis() – Generally used to **add parameters** in the functions.
Braces{} – Generally used to define the **classes or functions** and also used to initialize the arrays.
Brackets[] – Generally used for **indexing** of an array.
Comma “,” – It is used to **separate** different elements in the program such as identifiers,
Semicolon “;” – It is used to **end** any statement in the program.
Period . Used to **separate** the **package names** from sub packages and classes.

```
//Arithmetic Operators + - * % /

int a = 15;

int b = 3;

System.out.println(a+b); // 18

System.out.println(a-b); // 12

System.out.println(a*b);  //45

System.out.println(a%b); //0

System.out.println(a/b); // 5
```

```java
//Assignment Operators += -= *= /= %=

int a = 10;
int b = 20;
System.out.println(a += b); // a = a+b // a= 10+20 // 30

a = 30;
b = 40;
System.out.println(a -= b); // b = a-b // b = 30-40 // -10

a = 5;
b = 2;
System.out.println(a *= b); // a=a*b // a= 5*2 // 10

a = 10;
b = 2;
System.out.println(a /= b); // a=a/b // a=10/2 // 5

a = 10;
b = 20;
System.out.println(a %= b); // 10
```

```
//Unary Operators ++ --

int a = 10;
System.out.println(a++); // 10
System.out.println(a++); // 11 //post Increment by 1 (increase next)
System.out.println(a++); // 12 // post Increment by 1

int b = 20;
System.out.println(++b); // 21 // pre increment by 1 (increase first)
System.out.println(++b); // 22 // pre increment by 1

int x = 10;
x++;
System.out.println(x); // 11

int y = 20;
y++;
System.out.println(y); // 21
```

```java
//Comparison Operators or Relationship Operators == < > <= >= !=

int a = 10;
int b = 20;
int c = 10;

System.out.println(a == b); // false
System.out.println(a == c); // true
System.out.println(a < b); // true
System.out.println(a > b); // false
System.out.println(a >= c); // true
System.out.println(a <= c); // true
System.out.println(a != c); // false
```

```
//ternary operator

int a, b;
a = 10;

//variable = Expression1 ? Expression2 : Expression3
b = (a==1) ? 20 : 30;  //If the left hand expression is true Expression will Execute or Else Right Hand Expression Execute
System.out.println(b); //30

b = (a==10) ? 20 : 30; //If the left hand expression is true Expression will Execute or Else Right Hand Expression Execute
System.out.println(b); //20
```

```
// Bitwise Operators Using Boolean Validation

//& (bitwise AND)
//|  (bitwise OR)
//^  (bitwise exclusive OR)
// ~ not (bitwise complement)

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

System.out.println(true | true); // true
System.out.println(false | true); // true
System.out.println(true | false); // true
System.out.println(false | false); // false
```

```
// Bitwise Operators using boolean validation
//& (bitwise AND)
//|  (bitwise OR)
//^  (bitwise exclusive OR)
// ~ not (bitwise complement)

// Both are same false
System.out.println(true ^ true); // false

// Both are same false
System.out.println(false ^ false); // false

// Both are different true
System.out.println(true ^ false); // true

// Both are different true
System.out.println(false ^ true); // true
}
}
```

```
//Bitwise Complement ~ using integers validation
//& (bitwise AND)
//|  (bitwise OR)
//^  (bitwise exclusive OR)
// ~ not (bitwise complement)
//It Inverts the value of each bit(0 to 1) and (1 to 0)

int a = 10; // 0000000000001010
System.out.println(~a); // -11

int b = -11; // 1111111111110101
System.out.println(~b); // 10
```

A	B	A&B	A|B	A^B
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

```
// Bitwise Operators
//& (bitwise AND)
//|  (bitwise OR)
//^ (bitwise  exclusive OR)
public class Eg12 {
public static void main(String[] args) {
int a = 10; // 0b1010
int b = 2;   // 0b0010

System.out.println(a & b); // 2 // 0b0010
System.out.println(a | b); //10  //0b1010
System.out.println(a^b); // 8 // 0b1000
}
}
```

// Bitwise Operators : Left Shift Operator

Use 8 bit for Left Shift Operator, actually it is 4 bit

Now left shift is 2 remove two zeros from left and add at right (append)

Note: append only zeros and remove can be 0 or 1

```
int a = 10; // 0b1010
int b = 2;   // 0b0010
```

Add four zeros Before of it	Append
0b00001010	0b00101000

```
int a = 10;
int b = 2;
System.out.println(a<<b); // 40
```

// Bitwise Operators : Right Shift Operator

Use 8 bit for Right Shift Operator, actually it is 4 bit

Remove two numbers from right side and append at left side that can be only zeros not 1’s

```
int a = 10; // 0b1010
int b = 2; // 0b0010

System.out.println(a >> b); // 2
```

Add four zeros Before of it	Append
0b00001010	0b00000010

```
package com.dl.shortcircuit.operators;

//Logical Operators
//logical AND &&
//logical OR ||

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

System.out.println(true || true); // true //Dead code
System.out.println(false || true); // true
System.out.println(true|| false); // true //Dead code
System.out.println(false || false); // false
```

```java
int a = 10;
int b = 10;
// bitwise OR |
boolean c = (a++ != 10 | b++ == 10); // false I true
System.out.println(c); // true
System.out.println(a + " " + b); // 11 11

int x = 10;
int y = 10;
//logical OR ||
// If x expression is false so y expression is evaluated
boolean c1 = x++ != 10 || y++ == 10; // false || true
System.out.println(c1); // true
System.out.println(x + " " + y); // 11 11

int x1 = 10;
int y1 = 10;
// If x expression is true so y expression is not evaluated
boolean c2 = x1++ == 10 || y1++ == 10; // true || true
System.out.println(c2); // true
System.out.println(x1 + " " + y1); // 11 10
```

```java
int a = 10;
int b = 10;
// validates both the x and y expressions
boolean c = a++ == 10 & b++ == 10; // true & true
System.out.println(c); // true
System.out.println(a + " " + b); // 11 11

int x = 10;
int y = 10;
// if x expression is true, then y expression is evaluated
boolean c1 = x++ == 10 && y++ != 10; // true && false
System.out.println(c1); // false
System.out.println(x + " " + y); // 11 11

int x1 = 10;
int y1 = 10;
// if x expression is false then y expression will not validate
boolean c3 = x1++ != 10 && y1++ == 10; // false && true
System.out.println(c3); // false
System.out.println(x1 + " " + y1); // 11 10
}
```