



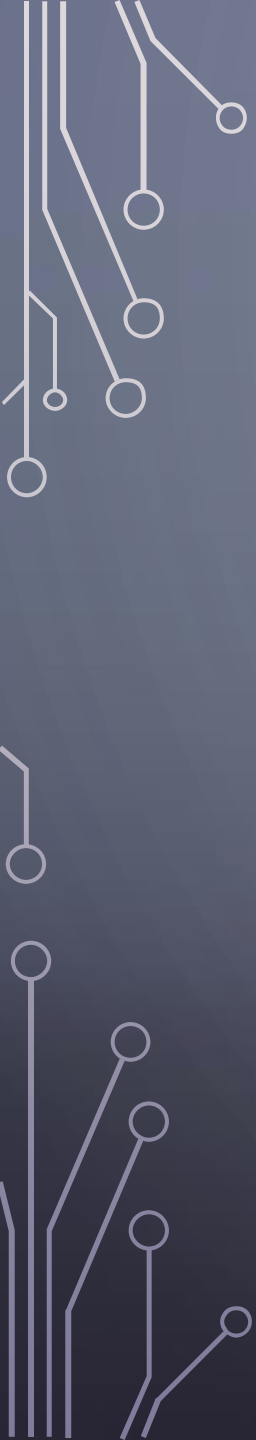
JAVA BASIC OPERATOR

MEMBER:

- 1.CHUM PISETH (E202020863)
- 2.DEN SOKUNPIDOR (E20201096)
- 3.HON RATHANA (E20201053)
- 4.BUTH KHEMERA (E20201690)
- 5.MENGHAB VATHANAK (E20201145)

CONTENT

1. What is Java Basic Operator of ?
2. Java Arithmetic Operator
3. Java Rational Operator
4. Java Bitwise Operator
5. Java Logical Operator



- Java Basic Operator: An operator, in Java, is a special symbols performing specific operations on one, two or three operands and then returning a result. The operators are classified and listed according to precedence order. Java operators are generally used to manipulate primitive data types.

- Arithmetic Operator: They are used to perform simple arithmetic operations on primitive data types.

- ❖ * : Multiplication

- ❖ / : Division

- ❖ % : Modulo

- ❖ + : Addition

- ❖ - : Subtraction



```
//a few numbers
int i = 37;
int j = 42;
double x = 27.475;
double y = 7.22;
System.out.println("Variable values...");
System.out.println("    i = " + i);
System.out.println("    j = " + j);
System.out.println("    x = " + x);
System.out.println("    y = " + y);

//adding numbers
System.out.println("Adding...");
System.out.println("    i + j = " + (i + j));
System.out.println("    x + y = " + (x + y));

//subtracting numbers
System.out.println("Subtracting...");
System.out.println("    i - j = " + (i - j));
System.out.println("    x - y = " + (x - y));

//multiplying numbers
System.out.println("Multiplying...");
System.out.println("    i * j = " + (i * j));
System.out.println("    x * y = " + (x * y));

//dividing numbers
System.out.println("Dividing...");
System.out.println("    i / j = " + (i / j));
System.out.println("    x / y = " + (x / y));

//computing the remainder resulting from dividing numbers
System.out.println("Computing the remainder...");
System.out.println("    i % j = " + (i % j));
System.out.println("    x % y = " + (x % y));

//mixing types
System.out.println("Mixing types...");
System.out.println("    j + y = " + (j + y));
System.out.println("    i * x = " + (i * x));
}
```

- **Java relational Operator:** These operators are used to check for relations like equality, greater than, and less than. They return Boolean results after the comparison and are extensively used in looping statements as well as conditional if-else statements. The general format is,

Variable relational-operator value

- ❖ Some of the relational operators are:
 - ❖ **==, Equal to** returns true if the left-hand side is equal to the right-hand side.
 - ❖ **!=, Not Equal to** returns true if the left-hand side is not equal to the right-hand side.
 - ❖ **<, less than:** returns true if the left-hand side is less than the right-hand side.
 - ❖ **<=, less than or equal to** returns true if the left-hand side is less than or equal to the right-hand side.
 - ❖ **>, Greater than:** returns true if the left-hand side is greater than the right-hand side.
 - ❖ **>=, Greater than or equal to** returns true if the left-hand side is greater than or equal to the right-hand side.

RelationalOperators.java

```
5 public static void main(String[] args) {  
6  
7     int a = 10;  
8     int b = 20;  
9  
10    System.out.println(a == b);  
11    System.out.println(a != b);  
12    System.out.println(a > b);  
13    System.out.println(a < b);  
14    System.out.println(a >= b);  
15    System.out.println(a <= b);  
16  
17    // objects support == and != operators  
18    System.out.println(new Data1() == new Data1());  
19    System.out.println(new Data1() != new Data1());  
20  
21 }
```

Problems Javadoc Declaration Console Progress

<terminated> RelationalOperators [Java Application] /Library/Java/JavaVirtualMachines/jdk-1

```
false  
true  
false  
true  
false  
true  
false  
true
```

- **Java Bitwise Operator:** These operators are used to perform the manipulation of individual bits of a number. They can be used with any of the integer types. They are used when performing update and query operations of the Binary indexed trees.

❖ **&, Bitwise AND operator:** returns bit by bit AND of input values.

❖ **~, Bitwise OR operator:** returns bit by bit OR of input values.

❖ **^, Bitwise XOR operator:** returns bit-by-bit XOR of input values.

❖ **~, Bitwise Complement Operator:** This is a unary operator which returns the one's complement representation of the input value, i.e., with all bits inverted.

eclipse-workspace - TestingDocsProject/src/com/testingdocs/bitwise/BitwiseOperatorExample.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

BitwiseOperatorExample.java ×

```
5  * Bitwise Logical Operators.
6  * Java Tutorials -- www.TestingDocs.com
7  */
8  public class BitwiseOperatorExample
9  {
10     public static void main(String args[])
11     {
12         //Declare two variables
13         int x =3;
14         int y =6;
15         System.out.println("x=: " + x);
16         System.out.println("y=: " + y);
17         // Bitwise AND operation
18         System.out.println("x&y is:= " + (x & y));
19         // Bitwise OR operation
20         System.out.println("x|y is:= " + (x | y));
21         // Bitwise XOR operation
22         System.out.println("x^y is:= " + (x ^ y));
23     }
```

Console × Progress

<terminated> BitwiseOperatorExample [Java Ap

x=: 3
y=: 6
x&y is:= 2
x|y is:= 7
x^y is:= 5

- **Java logical Operator:** These operators are used to perform “logical AND” and “logical OR” operations, i.e., a function similar to AND gate and OR gate in digital electronics. One thing to keep in mind is the second condition is not evaluated if the first one is false, i.e., it has a short-circuiting effect. Used extensively to test for several conditions for making a decision. Java also has “Logical NOT”, which returns true when the condition is false and vice-versa

❖ *Conditional operators are:*

- ❖ **&&, Logical AND:** returns true when both conditions are true.
- ❖ **||, Logical OR:** returns true if at least one condition is true.
- ❖ **!, Logical NOT:** returns true when a condition is false and vice-versa

```
1 package lh;
2
3 public class loper {
4     public static void main(String[]args) {
5         //declaring two variables
6         int a=5, b=6;
7
8         //setting condition
9         if ( a>=5 && b<6 )
10        {
11            System.out.println("Welcome to linuxhint");
12        }
13        else
14        {
15            System.out.println("Access denied! Please try again");
16        }
17    }
18 }
19 }
20
```

code

<terminated> loper [Java Application] C:\Program Files\Java\jd
Access denied! Please try again

output