**Operators in Java**

Operators are used to perform some operations on variables. The following operators are supported by JAVA.

* Arithmetic operators
* Logical operators
* Relational operators
* Assignment operators
* Bitwise operators
* Unary operators
* Ternary operators
* Shift operators
* new operators
* . operators

1. Arithmetic operators-

This operator is used to perform some mathematical operation such as addition (+), subtraction (-)

, Multiplication (\*), Division (/) and modules (%), etc.

**Example**-

**Output-**

1. Logical operators

This operators are used to perform logical AND & OR operation.

1. **Logical AND (&&) operators**-

Logical && operator doesn't check second condition if first condition is false. It checks second condition only if first one is true.

**Fig- Truth Table for Logical AND operator**

**Example- Scenario- 1**

In this example, first condition 10<20 is becomes true and second condition 10<30 is becomes true, both conditions are true, hence output is true.

**Output-**

**Example- Scenario 2**

In second example, first condition 10>20 is becomes false and second condition 10<30 is becomes true, hence output is false.

**2. Logical OR (||) operators-**

Logical || operator doesn't check second condition if first condition is true. It checks second condition only if first one is false.

**Fig- Truth table for Logical OR Operator**

**Example- Scenario-1**

In this example, first condition 10<20 is becomes true and second condition 10<30 is becomes true, hence output is true.

**Output-**

**Example- Scenario-2**

In this example, first condition 10>20 is becomes false and second condition 10>30 is becomes false, hence output is false.

**Output**-

3. Relational Operators-

This operators are used to perform greater than (>), less than (<), greater than or equal to (>=), less than or equal to (<=), equal to (==), not equal to (!=), etc.

**Example**-

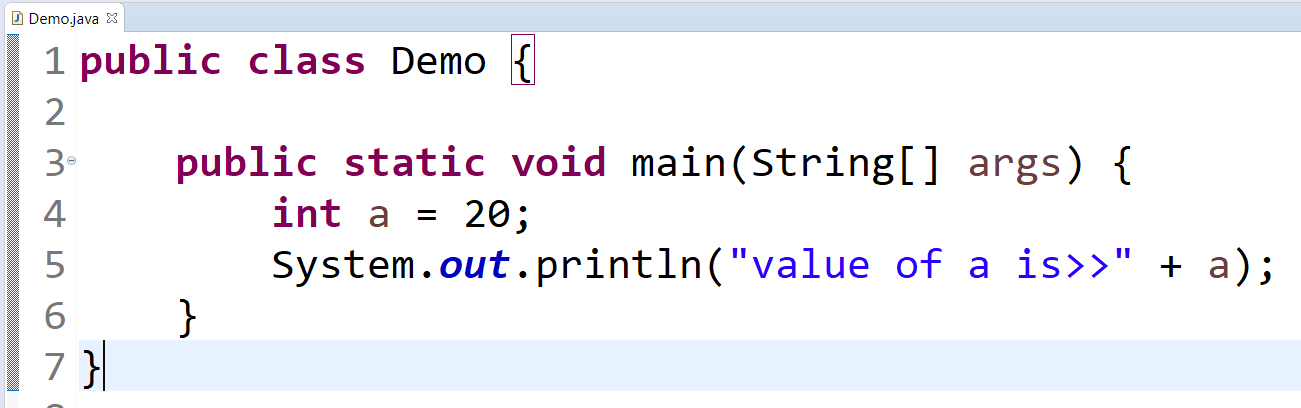
**Output-**

4. Assignment operators-

This operator is used to assign the values to variable.

**Syntax-** Variable =value;

**Example**-

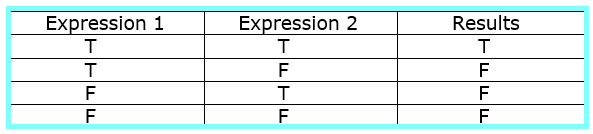


5. Bitwise operators-

This operators are used to perform Bitwise AND & OR operation.

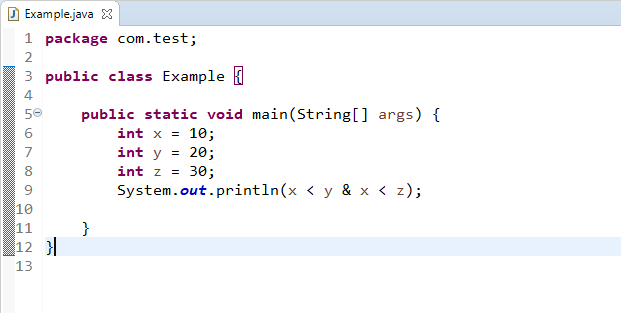
1. **Bitwise AND(&) operators-**

The bitwise & operator always checks both conditions whether first condition is true or false.



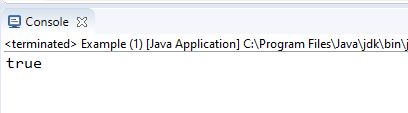
**Fig- Truth table for Bitwise AND operator**

**Example- Scenario-1**

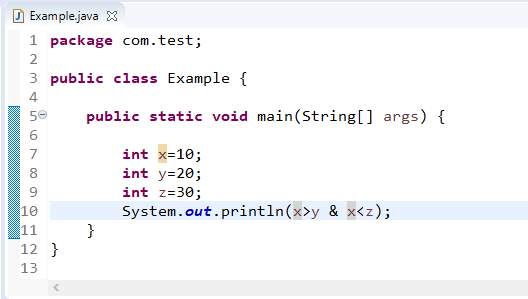


In this example, first condition 10<20 is becomes true and second condition 10<30 is becomes true, hence output is true.

**Output**-

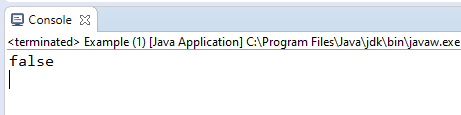


**Example- Scenario- 2**



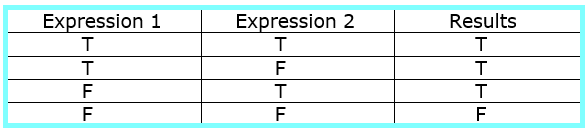
In second example, first condition 10>20 is becomes false and second condition 10<30 is becomes true, hence output is false.

**Output-**



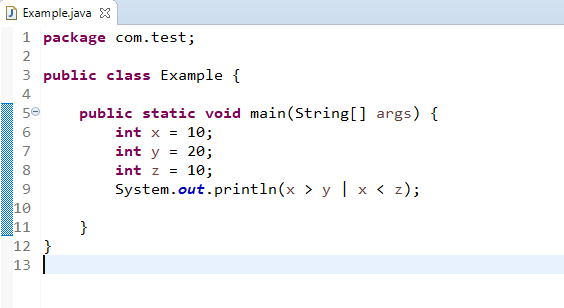
1. **Bitwise OR(|) operators-**

The bitwise (|) operator always checks both conditions whether first condition is true or false.



**Fig- Truth table for Bitwise OR operator**

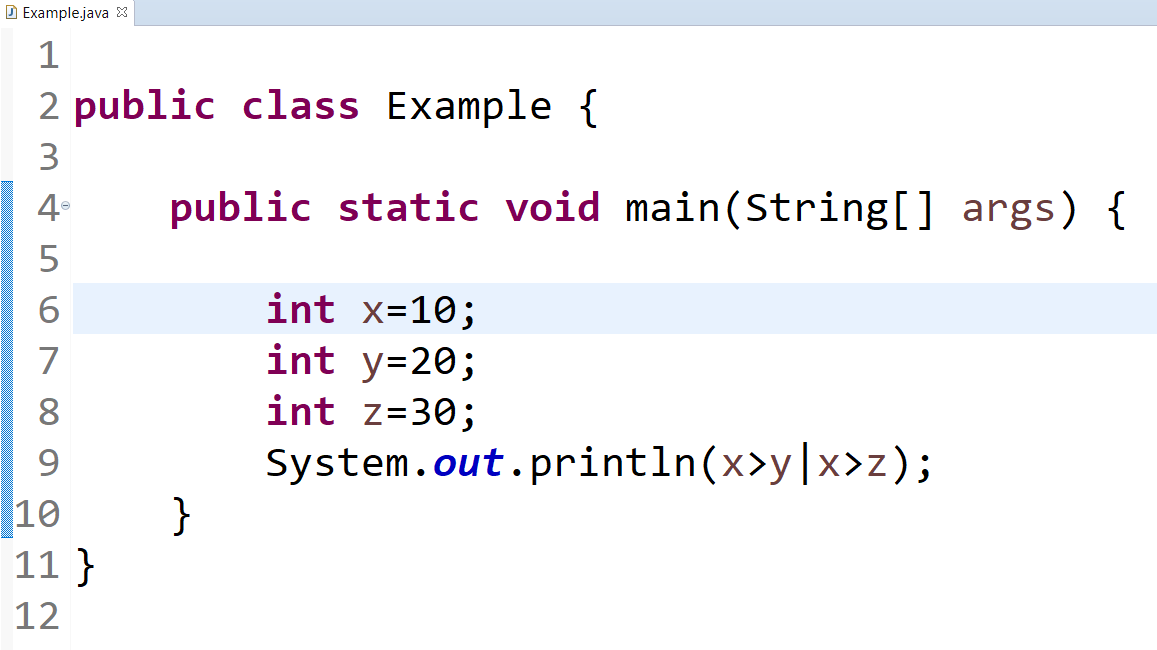
**Example-Scenario-1**



In this example, first condition 10>20 is becomes false and second condition 10<30 is becomes true, hence output is true.

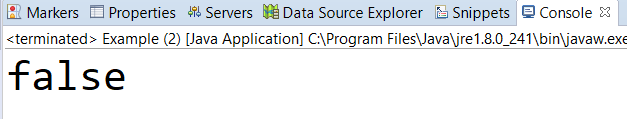
**Output**-

**Example- Scenario- 2**



In second example, first condition 10>20 is becomes false and second condition 10>30 is becomes false, hence output is false.

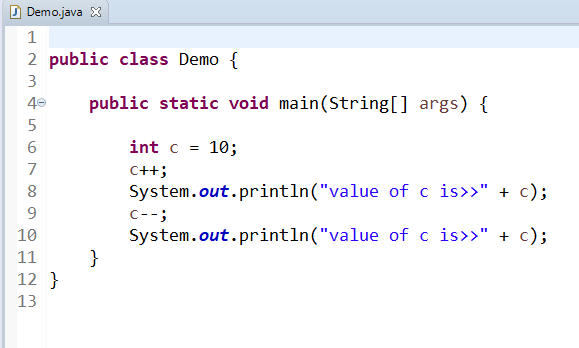
**Output-**



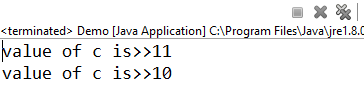
6. Unary operators-

This operators are used to perform an operation like increment (++) or decrement (--).

**Example**



**Output-**



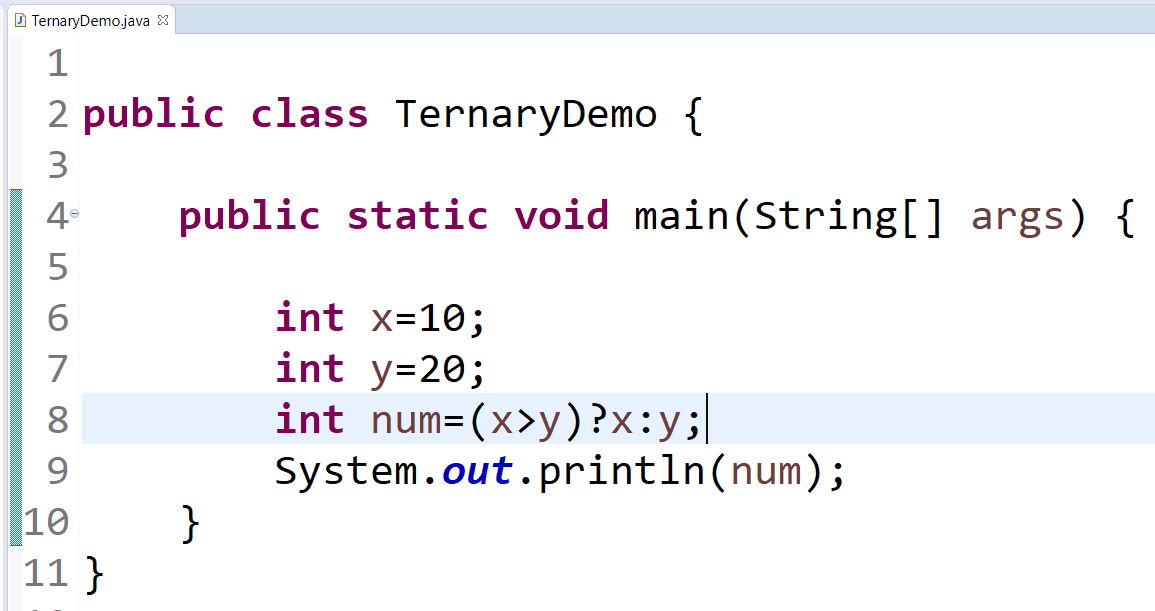
7. Ternary operators-

It includes three operands.

**Why?**

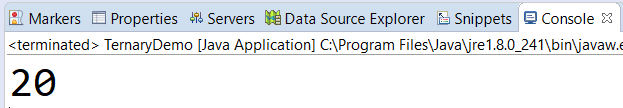
If else statement requires group of line code to execute the statement but by using this, we can write the code into one line only.

**Example**-**1**

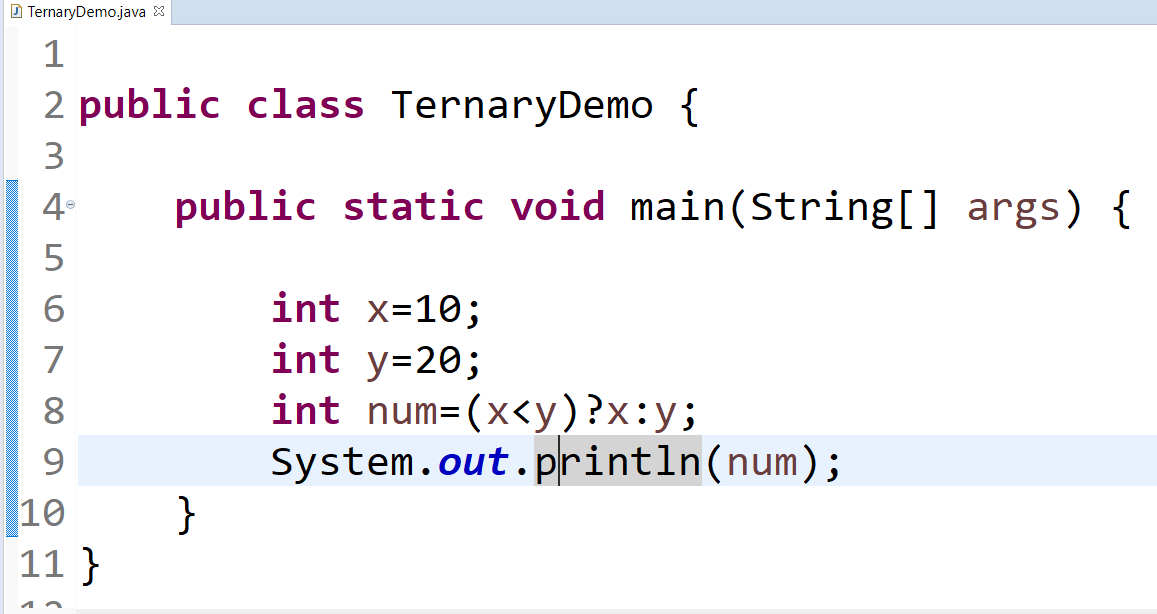


In this example, condition 10>20 becomes false, so output is 20.

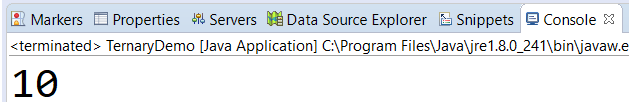
**Output**



**Example-2**



**Output-**

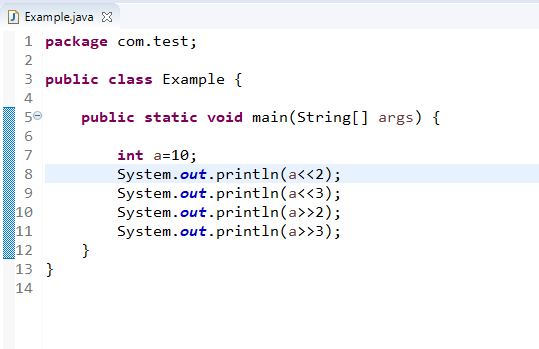


8. Shift operators (Right/Left)-

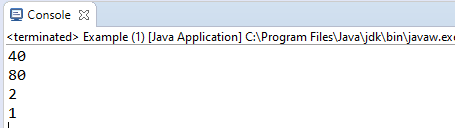
**Right shift operator >>** is used to move left operands value to right by the number of bits specified by the right operand.

**Left shift operator <<** is used to shift all of the bits in a value to the left side of a specified number of times.

**Example-**



**Output-**



1. On line 9, left shift operators occurs two times (<<), so we write it as 2, Right hand side we shift the position by 3 bits (i.e. numeric 3), Hence statement is 23.

We will always perform the multiplication operation on left shift operators. So we are putting value of a variable is 10.

Then will calculate, 10 \* 23 = ?

Cube of 2 is 8, so 10 \*8 =80.

We will get the output as **80**

1. On line 11, right shift operators occurs two times (>>), so we write it as 2, Right hand side we shift the position by 3 bits (i.e. numeric 3), Hence statement is23.

We will always perform the division operation on right shift operators. So we are putting value of a variable is 10.

Then will calculate, 10 / 23 = ?

Cube of 2 is 8, so 10 /8 =1.25 but the rounded value is 1.

We will get the output as **1**.

1. 10 /22 = 2
2. 10 /23 = 1
3. (.) operators

It is used to call the variable and method of class

10. new operators

It is used to create the object of class.