Operators

Operators play an important role in Java. There are three kinds of operators in Java. They are (i) Arithmetic Operators (ii) Comparison / Relational Operators and (iii) Logical Operators

Objectives

Upon completion of this topic, we will be able to:

- Understand operators in Java

Operators

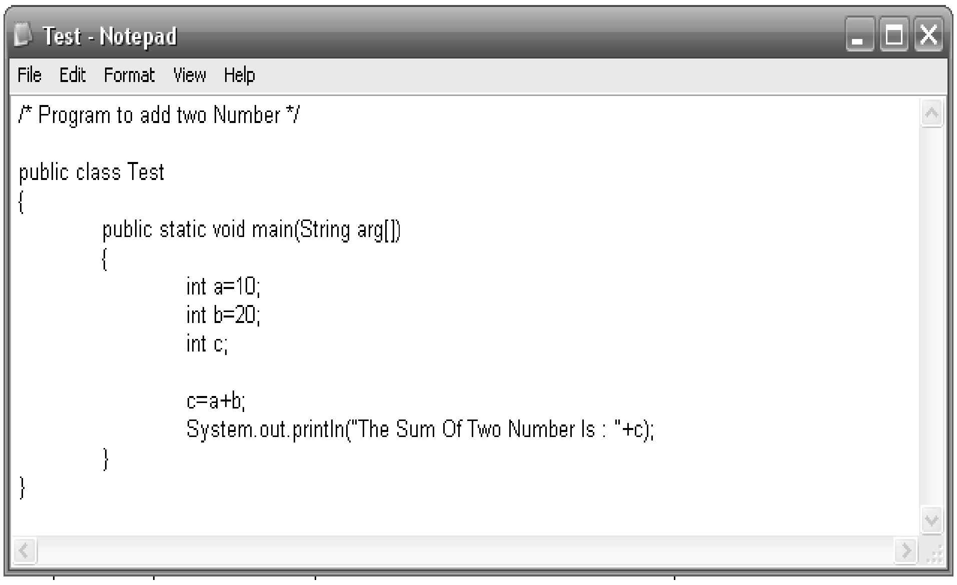
Arithmetic Operators

Addition, Subtraction, Multiplication, Division and Modulus are the various arithmetic operations that can be performed in Java.

List of Arithmetic Operators

Operator	Meaning	Use	Meaning
+	Addition	op1+op2	Adds op1 and op2
-	Subtraction	op1-op2	Subtracts op2 from op1
*	Multiplication	op1*op2	Multiplies op1 and op2
/	Division	op1/op2	Divides op1 by op2
%	Modulus	op1 % op2	Computes the remainder of dividing op1 by op2

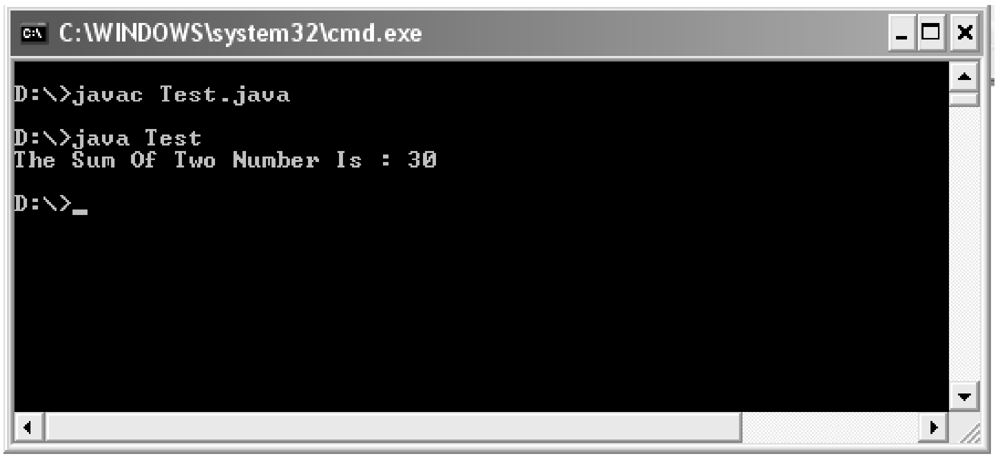
The following Java program adds two numbers and prints the result.



```
/* Program to add two Number */

public class Test
{
		public static void main(String arg[])
		{
			int a=10;
			int b=20;
			int c;

			c=a+b;
			System.out.println("The Sum Of Two Number Is : "+c);
		}
}
```

Java Program to add two numbers and printing the result

The compilation and running of the program is shown in the figure below.

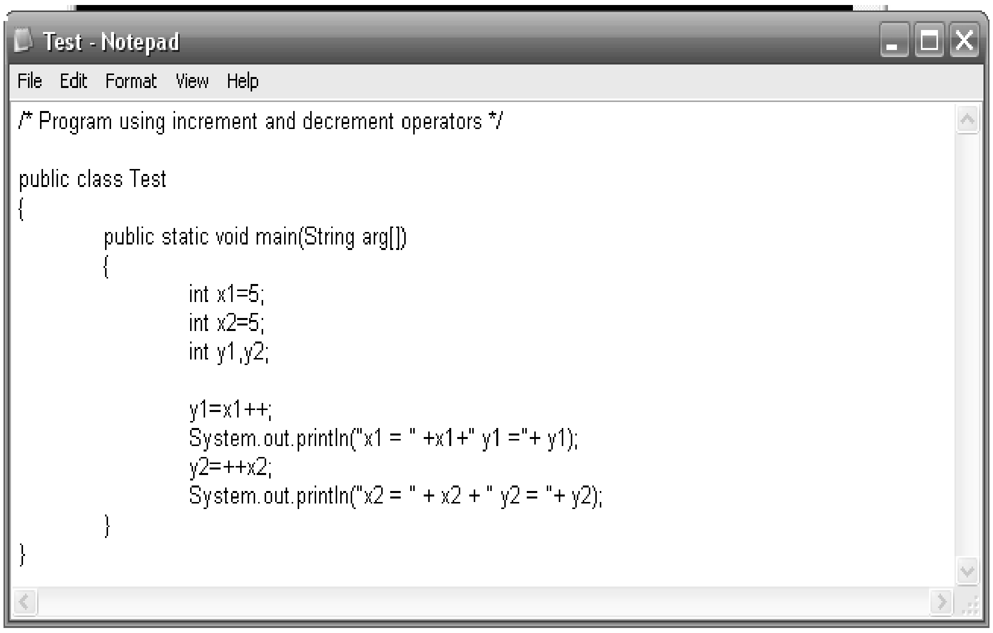


```
D:\>javac Test.java

D:\>java Test
The Sum Of Two Number Is : 30

D:\>_
```

Compilation and Running of Java Program

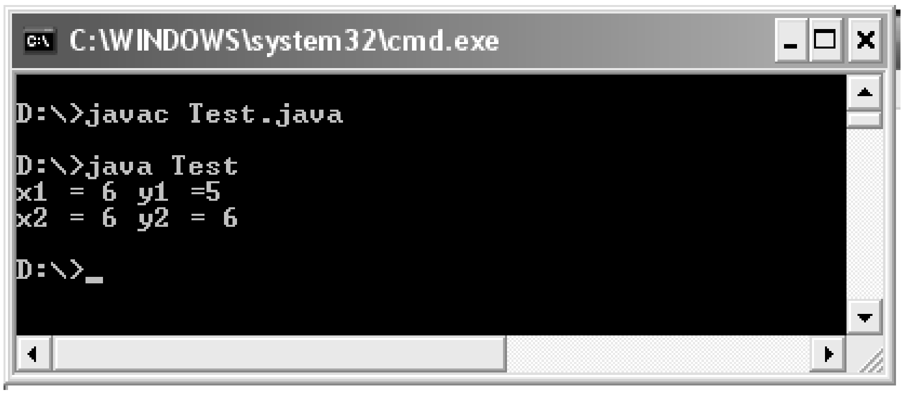
Increment and Decrement Operators

The increment operator is **++** and decrement operator is **--**. This is used to add 1 to the value of a variable or subtract 1 from the value of a variable. These operators are placed either before the variable or after the variable name. The example below shows the use of these operators.



```
/* Program using increment and decrement operators */

public class Test
{
		public static void main(String arg[])
		{
			int x1=5;
			int x2=5;
			int y1,y2;

			y1=x1++;
			System.out.println("x1 = " +x1+" y1 ="+ y1);
			y2=++x2;
			System.out.println("x2 = " + x2 + " y2 = "+ y2);
		}
}
```

Example showing increment operators in Java



```
D:\>javac Test.java

D:\>java Test
x1  = 6  y1  =5
x2  = 6  y2  =  6

D:\>_
```

Program Compilation and Running

When the operator **++** is placed after the variable name, first the assignment of the value of the variable takes place and then the value of the variable is incremented. This operation is also called ***post increment*****.** Therefore the value of y1 will remain as 5 and the value of x1 will be 6. When the operator is placed before the variable, first increment of the variable takes place and then the assignment occurs. Hence the value of both x2 and y2 will be 6. This operation is also called as ***pre increment*****.** Similarly **– –** operator can be used to perform ***post decrement*** and ***pre decrement*** operations. If there is no assignment and only the value of variable has to be incremented or decremented then placing the operator after or before does not make a difference.

Comparison Operators

Comparison operators are used to compare two values and give the results.

List of Comparison Operators in Java

Operator	**Meaning**	**Example**	**Remarks**
= =	Equal	op1 = = op2	Checks if op1 is equal to op2
!=	Not Equal	op1 != op2	Checks if op1 is not equal to op2
<	Less than	op1 < op2	Checks if op1 is less than op2
>	Greater than	op1 > op2	Checks if op1 is greater than op2
<=	Less than or equal	op1 <= op2	Checks if op1 is less than or equal to op2
>=	Greater than or equal	op1 >= op2	Checks if op1 is greater than or equal to op2

Logical Operators

Logical operators are used to perform Boolean operations on the operands.

List of Logical Operators in Java

Operator	**Meaning**	**Example**	**Remarks**
&&	Short-circuit AND	op1 && op2	Returns true if both are true. If op1 is false, op2 will not be evaluated and returns false.
||	Short-circuit OR	op1 || op2	Returns true if anyone is true. If op1 is true, op2 will not be evaluated and returns true.
!	Logical unary NOT	!op	Returns true if op is false.
&	Logical AND	Op1 & op2	Returns true if both are true. Always op1 and op2 will be evaluated.
|	Logical OR	Op1 | op2	Returns true if anyone is true. Always op1 and op2 will be evaluated.

Operator Precedence

When more than one operator is used in an expression, Java will use operator precedence rule to determine the order in which the operators will be evaluated.

For example, consider the following expression:

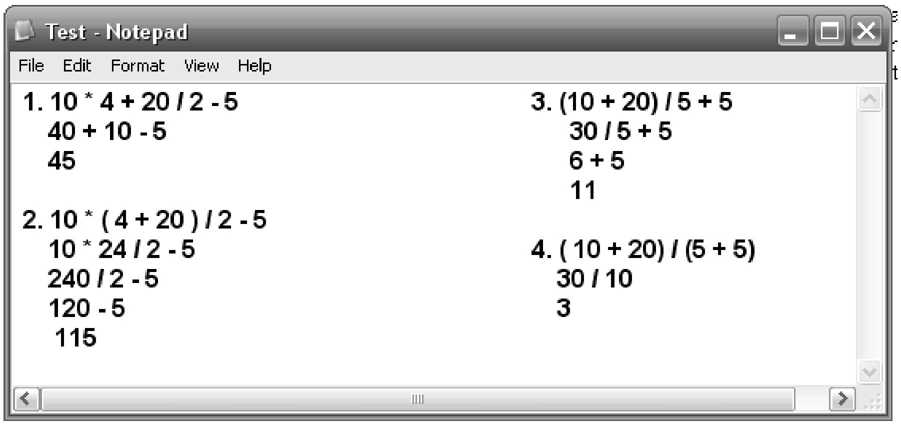
Result = 10+5*8-15/5

In the above expression, multiplication and division operations have a higher priority over addition and subtraction. Hence they perform first. Therefore the result now becomes 10+40-3.

Addition and subtraction have the same priority. When the operators have the same priority, they are evaluated from left to right in the order they appear in the expression. Hence the value of the result will become 47. In general, the following priority order is followed when evaluating an expression:

- Increment and decrement operations.
- Arithmetic operations.
- Comparisons.
- Logical operations.
- Assignment operations.

To change the order in which the expressions are evaluated, parentheses are placed around the expressions that are to be evaluated first. When the parentheses are nested together, the expressions in the innermost parentheses are evaluated first. Parentheses also improves the readability of the expressions. When the operator precedence is not clear, parentheses can be used to avoid any confusion.



```
1. 10 * 4 + 20 / 2 - 5
   40 + 10 - 5
   45

2. 10 * ( 4 + 20 ) / 2 - 5
   10 * 24 / 2 - 5
   240 / 2 - 5
   120 - 5
    115

3. (10 + 20) / 5 + 5
    30 / 5 + 5
    6 + 5
    11

4. ( 10 + 20) / (5 + 5)
   30 / 10
   3
```

Operator Precedence Example

Summary

Here are the key takeaways:

- Addition, Subtraction, Multiplication, Division and Modulus are the various arithmetic operations that can be performed in Java.
- The increment operator is ++ and decrement operator is --.
- Comparison operators are used to compare two values and gives results.
- Logical operators are used to perform Boolean operations on the operands.