Types of Operator in Java

1)Basic Arithmetic Operators
2) Assignment Operators
3) Auto-increment and Auto-decrement Operators
4) Logical Operators
5) Comparison (relational) operators & Terinary

1) Basic Arithmetic Operators

Basic arithmetic operators are: +, -, *, /, %
+ is for addition.

– is for subtraction.

***** is for multiplication.

/ is for division.

% is for modulo.
Note: Modulo operator returns remainder, for example 10 % 5 would return 0

Example of Arithmetic Operators

```java
public class ArithmeticOperatorDemo {
   public static void main(String args[]) {
      int num1 = 100;
      int num2 = 20;

      System.out.println("num1 + num2: " + (num1 + num2) );
      System.out.println("num1 - num2: " + (num1 - num2) );
      System.out.println("num1 * num2: " + (num1 * num2) );
      System.out.println("num1 / num2: " + (num1 / num2) );
      System.out.println("num1 % num2: " + (num1 % num2) );
   }
}
```

2) Assignment Operators

Assignments operators in java are: =, +=, -=, *=, /=, %=
num2 = num1 would assign value of variable num1 to the variable.

num2+=num1 is equal to num2 = num2+num1

num2-=num1 is equal to num2 = num2-num1

num2*=num1 is equal to num2 = num2*num1

num2/=num1 is equal to num2 = num2/num1

num2%=num1 is equal to num2 = num2%num1

```
public class AssignmentOperatorDemo {
   public static void main(String args[]) {
      int num1 = 10;
      int num2 = 20;

      num2 = num1;
      System.out.println("= Output: "+num2);

      num2 += num1;
      System.out.println("+= Output: "+num2);

      num2 -= num1;
      System.out.println("-= Output: "+num2);

      num2 *= num1;
      System.out.println("*= Output: "+num2);

      num2 /= num1;
      System.out.println("/= Output: "+num2);

      num2 %= num1;
      System.out.println("%= Output: "+num2);
   }
}
```

3) Auto-increment and Auto-decrement Operators

++ and —
num++ is equivalent to num=num+1;

num–- is equivalent to num=num-1;

4) Logical Operators

Logical Operators are used with binary variables. They are mainly used in conditional statements and loops for evaluating a condition.

Logical operators in java are: &&, ||, !

Let’s say we have two boolean variables b1 and b2.

b1&&b2 will return true if both b1 and b2 are true else it would return false.

b1||b2 will return false if both b1 and b2 are false else it would return true.

!b1 would return the opposite of b1, that means it would be true if b1 is false and it would return false if b1 is true..

Example of Logical Operators

```java
public class LogicalOperatorDemo {
   public static void main(String args[]) {
      boolean b1 = true;
      boolean b2 = false;

      System.out.println("b1 && b2: " + (b1&&b2));
      System.out.println("b1 || b2: " + (b1||b2));
      System.out.println("!(b1 && b2): " + !(b1&&b2));
   }
}
```

5) Comparison(Relational) operators

We have six relational operators in Java: ==, !=, >, <, >=, <=

== returns true if both the left side and right side are equal

!= returns true if left side is not equal to the right side of operator.

> returns true if left side is greater than right.

< returns true if left side is less than right side.

>= returns true if left side is greater than or equal to right side.

<= returns true if left side is less than or equal to right side.

Example of Relational operators

Note: This example is using if-else statement which is our next tutorial, if you are finding it difficult to understand then refer if-else in Java.

```java
public class RelationalOperatorDemo {
   public static void main(String args[]) {
      int num1 = 10;
      int num2 = 50;
      if (num1==num2) {
         System.out.println("num1 and num2 are equal");
      }
      else{
         System.out.println("num1 and num2 are not equal");
      }

      if( num1 != num2 ){
         System.out.println("num1 and num2 are not equal");
      }
      else{
         System.out.println("num1 and num2 are equal");
      }

      if( num1 > num2 ){
         System.out.println("num1 is greater than num2");
      }
      else{
         System.out.println("num1 is not greater than num2");
      }

      if( num1 >= num2 ){
         System.out.println("num1 is greater than or equal to num2");
      }
      else{
         System.out.println("num1 is less than num2");
      }

      if( num1 < num2 ){
         System.out.println("num1 is less than num2");
      }
      else{
         System.out.println("num1 is not less than num2");
      }

      if( num1 <= num2){
         System.out.println("num1 is less than or equal to num2");
      }
      else{
         System.out.println("num1 is greater than num2");
      }
   }
}
```

Ternary Operator

This operator evaluates a boolean expression and assign the value based on the result.

Syntax:

```
variable num1 = (expression) ? value if true : value if false
```

If the expression results true then the first value before the colon (:) is assigned to the variable num1 else the second value is assigned to the num1.

Example of Ternary Operator

```
public class TernaryOperatorDemo {

   public static void main(String args[]) {
        int num1, num2;
        num1 = 25;
        /* num1 is not equal to 10 that's why
         * the second value after colon is assigned
         * to the variable num2
         */
        num2 = (num1 == 10) ? 100: 200;
        System.out.println( "num2: "+num2);

        /* num1 is equal to 25 that's why
         * the first value is assigned
         * to the variable num2
         */
        num2 = (num1 == 25) ? 100: 200;
        System.out.println( "num2: "+num2);
   }
}
```