COCSIT

LANGUAGE BASIC (OPERATORS) PART II

COCSIT

Language Basics

Sensitivity: Internal & Restricted

Operators

- Java provides a set of operators to manipulate operations.
- Types of operators in java are,
  - Arithmetic Operators
  - Unary Operator
  - Relational Operators
  - Logical Operators
  - Simple Assignment Operator
  - Bitwise Operators

COCSIT

Arithmetic Operators

The following table lists the arithmetic operators

Operator	Description	Example
+	Addition	A + B
-	Subtraction	A - B
*	Multiplication	A * B
/	Division	A/B
%	Modulus	A%B

Arithmetic Operators - Example

```
/* Example to understand  Arithmetic operator */

class Sample{
  public static void main(String[ ] args){
    int a = 10;
    int b = 3;
    System.out.println("a + b = " + (a + b) );
    System.out.println("a - b = " + (a - b) );
    System.out.println("a * b = " + (a * b) );
    System.out.println("a / b = " + (a / b) );
    System.out.println("a % b = " + (a % b) );
  }
}
```

Output:

```
a + b  = 13
a - b  = 7
a * b  = 30
a / b   = 3
a % b  = 1
```

COCSIT

Unary Operators

The following table lists the unary operators

Operator	Description	Example
+	Unary plus operator	+A
-	Unary minus operator	-A
++	Increment operator	++A or A++
--	Decrement operator	--A or A--

Unary Operator - Example

```
/* Example for Unary Operators*/

class Sample{
  public static void main(String args[]) {
     int a = 10;
     int b = 20;

     System.out.println("++a    = " +  (++a) );
     System.out.println("--b    = " +  (--b) );
   }
}
```

Output:

```
++a  = 11
--b   = 19
```

COCSIT

Quiz

What will be the result, if we try to compile and execute the following code?

```
class Test {
   public static void main(String [ ] args) {
       int x=10;
       int y=5;
       System.out.println(++x+(++y));
   }
}
```

Relational Operators

The following table lists the relational operators

Operator	Description	Example
==	Two values are checked, and if equal, then the condition becomes true	(A == B)
!=	Two values are checked to determine whether they are equal or not, and if not equal, then the condition becomes true	(A != B)
>	Two values are checked and if the value on the left is greater than the value on the right, then the condition becomes true.	(A > B)
<	Two values are checked and if the value on the left is less than the value on the right, then the condition becomes true	(A < B)
>=	Two values are checked and if the value on the left is greater than equal to the value on the right, then the condition becomes true	(A >= B)
<=	Two values are checked and if the value on the left is less than equal to the value on the right, then the condition becomes true	(A <= B)

Comparing Strings

- For comparing Strings instead of using = = operator, use equals method

```
String s1="hello";
String s2="Wipro";
System.out.println(s1.equals(s2)); ➔false
```

COCSIT

Relational Operators - Example

```
/* Example to understand  Relational operator */

  class Sample{
   public static void main(String[] args){
        int a = 10;
        int b = 20;
        System.out.println("a == b = " + (a == b) );
        System.out.println("a != b = " + (a != b) );
        System.out.println("a > b = " + (a > b) );
        System.out.println("a < b = " + (a < b) );
        System.out.println("b >= a = " + (b >= a) );
        System.out.println("b <= a = " + (b <= a) );
   }
  }
```

Output:

```
a == b = false
a != b = true
a > b = false
a < b = true
b >= a = true
b <= a = false
```

COCSIT

Logical Operators

The following table lists the logical operators

Operator	Description	Example
&&	This is known as Logical AND & it combines two variables or expressions and if and only if both the operands are true, then it will return true	(A && B) is false
||	This is known as Logical OR & it combines two variables or expressions and if either one is true or both the operands are true, then it will return true	(A || B) is true
!	Called Logical NOT Operator. It reverses the value of a Boolean expression	!(A && B) is true

COCSIT

Logical Operators - Example

```
/* Example to understand  logical operator */


class Sample{
   public static void main(String[] args){
     boolean a = true;
     boolean b = false;
     System.out.println("a && b = " + (a&&b) );
     System.out.println("a || b = " + (a||b) );
     System.out.println("!(a && b) = " + !(a && b) );
   }
}
```

Output:

```
a && b = false
a || b = true
!(a && b) =
true
```

COCSIT

Simple Assignment Operator

= is the simple assignment operator which assigns right hand side value to left hand side variable.

Ex:

```
int a;
a = 10;
```

COCSIT

Bitwise Operators

The bitwise operators take two bit numbers, use OR/AND to determine the result on a bit by bit basis.

The 3 bitwise operators are :

- & (which is the bitwise AND)
- | (which is the bitwise inclusive OR)
- ^ (which is the bitwise exclusive OR)

COCSIT

Bitwise & (AND) operator

The & operator compares corresponding bits between two numbers and if both the bits are 1, only then the resultant bit is 1. If either one of the bits is 0, then the resultant bit is 0.

Example :

```
int x = 7;
int y = 9;
```

What will be x & y ?

```
7 -> 0 1 1 1
9 -> 1 0 0 1
----------------
       0 0 0 1
----------------
```

x & y results in 1.

COCSIT

Bitwise & Operator Demo

```
class BitwiseExample1 {
  public static void main(String[] args) {
    int x = 7;
    int y = 9;
    int z = x & y;
    System.out.println("z = "+z);
  }
}
```

Output :
z = 1

Bitwise | (inclusive OR) operator

The | operator will set the resulting bit to 1 if *either* one of them is 1. It will return 0 only if both the bits are 0.

Example :

```
int x = 5;
int y = 9;
```

What will be x | y ?

```
5 -> 0 1 0 1
9 -> 1 0 0 1
-----------------
       1 1 0 1
-----------------
```

x | y results in 13.

COCSIT

Bitwise | Operator Demo

```
class BitwiseExample2 {
  public static void main(String[] args) {
    int x = 5;
    int y = 9;
    int z = x | y;
    System.out.println("z = "+z);
  }
}
```

Output :
z = 13

COCSIT

Bitwise ^ (exclusive OR) operator

The ^ operator compares two bits to check whether these bits are different. If they are different, the result is 1.Otherwise, the result is 0. This operator is also known as XOR operator.

Example :

```
int x = 5;
int y = 9;
```

What will be x ^ y ?

```
5 -> 0 1 0 1
9 -> 1 0 0 1
-----------------
        1 1 0 0
-----------------
```

x ^ y results in 12.

Bitwise ^ Operator Demo

```
class BitwiseExample3 {
  public static void main(String[] args) {
    int x = 5;
    int y = 9;
    int z = x ^ y;
    System.out.println("z = "+z);
  }
}
```

Output :
z = 12

COCSIT

Quiz

What will be the output for the below code ?

```
public class Sample {
   public static void main() {
    int i_val = 10, j_val = 20;
    boolean chk;
    chk = i_val < j_val;
    System.out.println("chk value: "+chk);
   }
}
```

Thanking
You