2016 ජූනි 18, සෙනසුරාදා

CO 225

(2) Operators and Expressions (continue…)

Operators

- (6) Conditional Operator

Ternary operator (operator with three parts)

General form is:

    <expression1> ? <expression2> : <expression3>

Order of execution: If <expression1> is true, then the <expression2> get executed and becomes the value. Otherwise <expression3> get executed and becomes the value.

eg:- a = 10, b = 20

x = (a > b) ? a : b;

This can be expressed by if-else structure as well:

```
if (a > b)
        x = a;
else
        x = b;
```

- (7) Bitwise Operator

Manipulate data of values of bit level

This operator can be used to test the bit, shift to left and shift to right.

This is not for float or double.

Operator	**Meaning**
&	bitwise AND
!	bitwise NOT
^	bitwise EXCLUSIVE OR
~	one's complement
<<	shift left
>>	shift right
>>>	shift right with zero fill

```
class TestBitwise{
        public static void main (String args[]){
                int x = 5;
                int y = 6;
                System.out.println("x =" + x); => Output x = 5
                System.out.println("y =" + y); => Output y = 6
                System.out.println("x & y =" + (x&y)); => Output x & y = 4
                System.out.println("x ^ y =" + (x^y)); => Output x ^ y = 3
                }
        }
```

```
class TestShift{
        public static void main (String args[]){
                int x = 7;
                System.out.println("x =" + x); => Output x = 7
                System.out.println("x>>>2=" + (x>>>2)); => Output x>>>2=1
                System.out.println("x<<1=" + (x<<1)); => Output x<<1=14
                System.out.println("x>>>1=" + (x>>>1)); => Output x>>>1=3
                }
        }
```

- (8) Special Operators

01. “InstanceOf” operator

    return TRUE or FALSE

    if your object on the left is an instance of the class given on the right, then returns TRUE

eg:- Let say there are two classes named Student and Employee

Student s = new Student (); => this will create an object s of the class Student

```
if (s instanceOf Student){

        <some_method1>;

        } => TRUE, <some_method1> will be called

if (s instanceOf Employee){

        <some_method2>;

        } => FALSE, <some_method2> will not b called
```

This operator can be used to make sure that the object s is of the class Student, when calling a method.

02. Dot operator

use to access instance variables, constants and operators

eg:-

```
class Student{
        int age;
        public int getAge(){
                return this.age;
                }
        }



Student s = new Student ();
System.out.println(s.getAge());
        => use the dot operator (.) to access the method getAge() of the object s.
```

Expressions

Combination of variables, constants and operators arranged according to the syntax of the language.

General form	Java form
ab + c	a * b + c
(m+n)(x+y)	(m+n)*(x+y)
ab/c	a*b/c
$3x^2$+2x+1	3*x*x + 2*x + 1

General form of evaluation (how Java evaluate order)

(1) Increment/decrement

(2) arithmetic

(3) comparison

(4) logical

(5) assignment

Operator Associativity

If there are more than one operators in the same type appear in an expression, then associativity defines he way of expression

eg:- int result = 3+4-5-1 = 1

int result = 3*6/2 = 9

int result =3*6/2+3+4 = 9+3+4 = 16

Changing order of evaluation by using parentheses ().

eg:- result = 3+2*3 = 9 whereas (3+2)*3 = 15