UNIT-II
Building Blocks of Language

Introduction

- **Operator in java is a symbol that is used to perform operations.**
  - Unary Operator
  - Arithmetic Operator
  - shift Operator
  - Relational Operator
  - Bitwise Operator
  - Logical Operator
  - Ternary Operator
  - Assignment Operator

Bitwise Operator

- Java defines several bitwise operators, which can be applied to the integer types, long, int, short, char, and byte.

- **Bitwise operator works on bits and performs bit-by-bit operation.**

- Assume if a = 60 and b = 13;

Bit	Operator	Description
&	AND	The bitwise AND operator, &, combines corresponding bits in its two operands such that if both bits are 1, the result is 1—otherwise the result is 0
|	OR	The bitwise OR operator, |, combines corresponding bits such that if either or both bits are 1, then the result is 1. Only if both bits are 0 is the result 0
^	Exclusive OR	The bitwise exclusive OR (XOR) operator, ^, combines corresponding bits such that if both bits are the same the result is 0; otherwise, the result is 1.
~	Complement	The complement operator, ~, takes a single operand in which it inverts all the bits, so that each 1 bit becomes 0, and each 0 bit becomes 1.

Ex: a = 0011 1100

b = 0000 1101

Then,

a&b = 0000 1100

a|b = 0011 1101

a^b = 0011 0001

~a = 1100 0011

Assignment Operator

- It used to assigns the value on its right to the operand on its left.
- Datatype Variable = expression;
- Ex: int x=5;

Operator	Purpose	Example	Equivalent
+=	Addition	x += 2	x = x + 2
-=	Subtraction	x -= 2	x = x - 2
/=	Division	x /= 2	x = x / 2
*=	Multiplication	x *= 2	x = x * 2
%=	Modulus	x %= 2	x = x % 2

Unary Operator

- **The unary operators require only one operand.**
- They perform various operations such as incrementing/decrementing a value by one, negating an expression, or inverting the value of a Boolean.
- **Prefix and postfix**

Operator	Description
+	**Unary plus operator;** indicates positive value (numbers are positive without this)
-	**Unary minus operator;** negates an expression
++	**Increment operator;** increments a value by 1
--	**Decrement operator;** decrements a value by 1
!	**Logical complement operator;** inverts the value of a boolean

Logical Operators

- These operators are **used to perform "logical AND" and "logical OR", "Logical NOT" operation**

- Similar function as AND gate and OR gate in digital electronics.

- One thing to keep in mind is the second condition is not evaluated if the first one is false, i.e. it has short-circuiting effect.

- Used extensively to test for several conditions for making a decision.

Logical Operators

- **&& , Logical AND :**
  - returns true when both conditions are true.
- **|| , Logical OR :**
  - returns true if at least one condition is true.
- **!, Logical Not:**
  - Returns true if the value is false and vice versa.

Shift Operator

- These operators are used to shift the bits of a number left or right thereby multiplying or dividing the number by two respectively.

- They can be used when we have to multiply or divide a number by two.

- **<< , Left shift operator:** shifts the bits of the number to the left and fills 0 on voids left as a result. Similar effect as of multiplying the number with some power of two.

- **>> , Signed Right shift operator:** shifts the bits of the number to the right and fills 0 on voids left as a result. The leftmost bit depends on the sign of initial number. Similar effect as of dividing the number with some power of two.

- **>>> , Unsigned Right shift operator:** shifts the bits of the number to the right and fills 0 on voids left as a result. The leftmost bit is set to 0.

Relational Operators

- These operators are used to check for relations like equality, greater than, less than.

- They return boolean result after the comparison and are extensively used in looping statements as well as conditional if else statements.

Relational Operators

- **== , Equal to :**
  - returns true of left hand side is equal to right hand side.
- **!= , Not Equal to :**
  - returns true of left hand side is not equal to right hand side.
- **< , less than :**
  - returns true of left hand side is less than right hand side.

Relational Operators

- **<= , less than or equal to :**
  - returns true of left hand side is less than or equal to right hand side.
- **> , Greater than :**
  - returns true of left hand side is greater than right hand side.
- **>= , Greater than or equal to:**
  - returns true of left hand side is greater than or equal to right hand side.

Ternary Operators

- **Ternary operator is a shorthand version of if-else statement. It has three operands and hence the name ternary.**
- **Syntax:**
  - **expr1? expr2 : expr3**
- **Ex:** (a>b)?a:b