C programming - Operators

Types of Operators (On the basis of number of operands) –

1. Unary operator –
   The operator which works on single operand at a time. For example: -5 (Negative), &n (Address of), *p (Value at), !n (NOT/negate), n++ (increment), n-- (decrement), etc.

2. Binary operator –
   The operator which works on two operands at a time. For example: a + b (Addition), a – b (Subtraction), a * b (Multiplication), a > b (Greater than), a <= b (Less than or equal to), etc.

3. Ternary operator –
   The operator which works on three operand.

   Condition ? (True block) : (False block)

   - If the "Condition" is true, then the statement inside the true block (After the '?') executes.
   - If the "Condition" is false, then the statement inside the false block (After the ':') executes.

   For example: x = (4>12) ? 15 : 25;

   The value inside the variable 'x' is 25 the condition provided is false in this case.

   NOTE – Some operators like *, &, - can act as both unary and binary operator depending upon given number of operator.

Types of Operators (On the basis of use/task) –

1. Arithmetic operators (+, -, *, /, %)–
   (+, -, *, /) – Works on any type of numeric value and returns the result in larger type among operators.

   (%) – It is known as "mod" or "modulus" operator and works only on integral number and returns remainder.

   NOTE – In case of dividing negative numbers, the sign of the remainder ONLY depends on the divisor (i.e. the number being divided).

2. Relational operators (==, !=, <=, >=, <, >) –
   == (Equals to), != (Not equal to), <= (Smaller than or equal to), >= (Greater than or equal to), < (Smaller than), > (Greater than) are the relational operators. These operators return TRUE(or 1) if the expression is true otherwise return FALSE(or 0).

   For example: (2 > 4) returns FALSE or 0.

   NOTE – Only == relational operator is commutative(Works from either side).

3. Logical operators (||, &&, !) –
   || (Logical OR), && (Logical AND) and ! (Logical NOT) are the logical operators in C programming language.

4. Assignment operator (=) –
   It works right to left i.e. value in the right gets stored into left variable.

Difference between Assignment operator (=) and Equality operator (==) :

Equality operator (==)	Assignment operator (=)
Is commutative	Not commutative
Left to right	Right to left
Return 0 or 1	Returns nothing
Doesn't compulsory need a variable on left	Must contain a variable in left

5. Address of (&) –

Returns address of a variable in the memory.

6. Sizeof() –

Returns size of a variable in bytes.

For example: int n = 423;

Sizeof(n) will return 4.

7. Increment/Decrement operator –

Types –

a. **Pre-increment/Decrement (++i, --i)** – The value of i increments or decrements before the execution.
b. **Post-increment/Decrement (i++, i--) –** The value of i increments or decrements after the execution.

NOTE – Post increment can also be written as: i = i + 1 or i += 1

8. Bitwise operator –

a. **Bitwise AND (&) –** This operator calculates bitwise AND of two operands. For example: 3&2 = 2.
b. **Bitwise OR (|) -** This operator calculates bitwise OR of two operands. For example: 3|2 = 3.
c. **Bitwise XOR (^) -** This operator calculates bitwise XOR of two operands. For example: 3^2 = 1.
d. **Bitwise Left-shift (<<) –** This operator left-shifts the bits of the number. For example: 2<<3 = 16.
e. **Bitwise Right-shift (>>) -** This operator right-shifts the bits of the number. For example: 16>>3 = 2.
f. **Bitwise Compliment (~) -** This operator returns the bitwise compliment. For example: ~724 = 275.

NOTE – Bitwise operators works on the binary number system which makes the execution faster. For better execution time we should try to use Bitwise operators.