



SP Group of Institute

UNIT 2: OPERATORS - Identity & Precedence

This unit explains Identity Operators and Operator Precedence in C programming with examples.

1. IDENTITY OPERATORS

It is used to compare the memory location of two operands.

Operators :

`==` (is equal to)

`!=` (is not equal to)

Example :

```
int a = 10, b = 10;
int *p = &a, *q = &b;
if (p == q)
    ↳ Same address
else
    ↳ Different address
```

Common Mistakes :

- Confusing `==` (equality) with `=` (assignment).
- Comparing values instead of addresses.

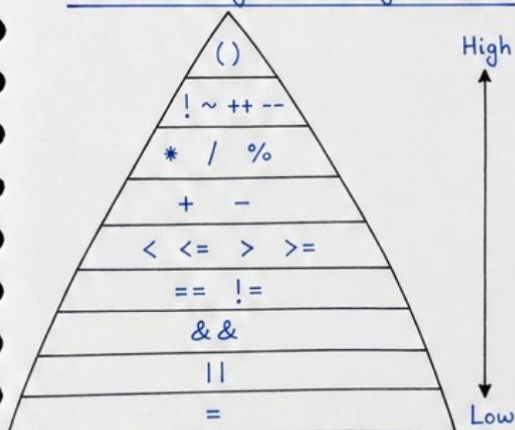
Example :

```
if (a == b) // compares values
if (&a == &b) // compares addresses
```

2. PRECEDENCE & ASSOCIATIVITY

Operators are evaluated according to their precedence. Higher precedence operators are evaluated first.

Precedence Pyramid (High → Low)



Example 1 :

```
int a = 10, b = 5, c = 2;
int result = a + b * c;
printf("%d", result);
```

Output :

20

Explanation :

First $b * c = 5 * 2 = 10$
Then $a + 10 = 10 + 10 = 20$

Example 2 :

```
int a = 10, b = 5, c = 2;
int result = (a + b) * c;
printf("%d", result);
```

Output :

30

Explanation :

First $(a + b) = 10 + 5 = 15$
Then $15 * c = 15 * 2 = 30$

Notes :

- Precedence decides which operator is evaluated first.
- Operators in same precedence group are evaluated according to associativity.
- Most binary operators are left to right associative.
- Assignment operator (`=`) is right to left associative.

Practice
Makes
Perfect!





SP Group of Institute

UNIT 2 : OPERATORS - Arithmetic & Relational

Operators are special symbols that perform operations on variables and values.



1. Arithmetic Operators

These operators are used to perform mathematical operations.

Operator	Symbol	Concept	Code Example	Output
Addition	+	Adds two values	<code>a = 10 ; b = 5 ; c = a + b ;</code>	<code>c = 15</code>
Subtraction	-	Subtracts right value from left value	<code>a = 10 ; b = 5 ; c = a - b ;</code>	<code>c = 5</code>
Multiplication	*	Multiplies two values	<code>a = 10 ; b = 5 ; c = a * b ;</code>	<code>c = 50</code>
Division	/	Divides left value by right value	<code>a = 10 ; b = 5 ; c = a / b ;</code>	<code>c = 2</code>
Modulus	%	Returns remainder of division	<code>a = 10 ; b = 3 ; c = a % b ;</code>	<code>c = 1</code>



2. Relational Operators

These operators are used to compare two values.

Operator	Symbol	Concept	Code Example	Output
Equal to	<code>==</code>	Checks if two values are equal	<code>a = 5 ; b = 5 ; c = (a == b) ;</code>	<code>c = true</code>
Not equal to	<code>!=</code>	Checks if two values are not equal	<code>a = 5 ; b = 3 ; c = (a != b) ;</code>	<code>c = true</code>
Greater than	<code>></code>	Checks if left value is greater	<code>a = 7 ; b = 3 ; c = (a > b) ;</code>	<code>c = true</code>
Less than	<code><</code>	Checks if left value is smaller	<code>a = 2 ; b = 9 ; c = (a < b) ;</code>	<code>c = true</code>
Greater than or equal to	<code>>=</code>	Checks if left value is greater or equal	<code>a = 5 ; b = 5 ; c = (a >= b) ;</code>	<code>c = true</code>
Less than or equal to	<code><=</code>	Checks if left value is less or equal	<code>a = 4 ; b = 7 ; c = (a <= b) ;</code>	<code>c = true</code>

Note : Relational operators return boolean values: true (1) or false (0).

★ Key Takeaway:

Arithmetic operators perform calculations, while relational operators perform comparisons.



SP Group of Institute

UNIT 2 : OPERATORS - Logical & Assignment

Operators are special symbols that perform operations on variables and values.

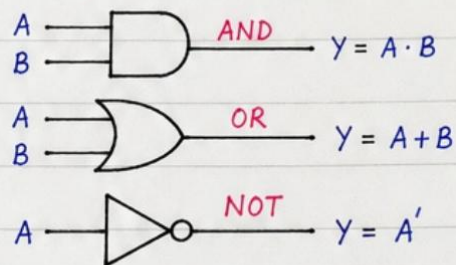
* LOGICAL OPERATORS

Logical operators are used to combine two or more conditions. They return either True (1) or False (0).

Truth Table

A	B	A AND B	A OR B	NOT A
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0

Logic Gate Representation



Note: In C/C++, 0 is False and non-zero value is True.

* ASSIGNMENT OPERATORS

Assignment operators are used to assign values to variables. The left operand must be a variable.

Common Assignment Operators

Operator	Example	Equivalent To
=	$a = b$	$a = b$
+=	$a += b$	$a = a + b$
-=	$a -= b$	$a = a - b$
*=	$a *= b$	$a = a * b$
/=	$a /= b$	$a = a / b$
%=	$a \% = b$	$a = a \% b$

Code Examples

```
// Example 1
int a = 10, b = 3;
a += b; // a = a + b
cout << a; // Output: 13
```

```
// Example 2
int x = 20;
x -= 5; // x = x - 5
cout << x; // Output: 15
```

Remember:

- Use && for AND, || for OR, ! for NOT in C/C++.
- Assignment operators make code short and easy.

Example (Logical Operators in C++)

```
int p = 5, q = 10;
if (p < q && q != 0)
    cout << "Both conditions are True";
else
    cout << "At least one condition is False";
```



Output:
Both conditions
are True



SP Group of Institute

UNIT 2 : OPERATORS -

Bitwise & Membership

* BITWISE OPERATORS

Bitwise operators work on individual bits of integers. They treat numbers in binary form.

Operator	Name	Symbol	Example (a=5, b=3)	Binary	Result
AND	Bitwise AND	&	5 & 3 = 1	$\begin{array}{r} 0101 \\ 0011 \\ \hline \end{array}$	0001 (1)
OR	Bitwise OR		5 3 = 7	$\begin{array}{r} 0101 \\ 0011 \\ \hline \end{array}$	0111 (7)
XOR	Bitwise XOR	^	5 ^ 3 = 6	$\begin{array}{r} 0101 \\ 0011 \\ \hline \end{array}$	0110 (6)
NOT	Bitwise NOT	~	~5 = -6	0101 →	...1010 (-6)
Left Shift	Shift Left	<<	5 << 1 = 10	$\begin{array}{r} 0101 \\ 1010 \end{array}$	1010 (10)
Right Shift	Shift Right	>>	5 >> 1 = 2	$\begin{array}{r} 0101 \\ 0010 \end{array}$	0010 (2)

Interview Question :

→ Q. What is the result of 12 & 5 ?

12 = 1100 , 5 = 0101

$$\begin{array}{r} 1100 \\ \& 0101 \\ \hline 0100 = 4 \end{array}$$

Answer : 4

* MEMBERSHIP OPERATORS

Membership operators are used to test whether a value or variable is found in a sequence (string, list, tuple, set, etc.).

Operators :

- in → True if present
- not in → True if not present

Note :

The 'in' and 'not in' operators improve code readability and are widely used in loops and conditional statements.

Examples :

1. x = 'Computronics'
 print('Comp' in x) # Substring check
 print('tron' not in x) # Not present
 Output: True
 False

2. lst = [10, 20, 30, 40]
 print(20 in lst) # Element check
 print(50 not in lst) # Not present
 Output: True
 True

3. s = {1, 2, 3, 4}
 print(3 in s)
 print(5 not in s)
 Output: True
 True

* Summary : Bitwise operators work at the bit level, while membership operators check existence of elements in a sequence. 😊