



By Parthasarathi Swain

# What is an Operator?

**An operator is a symbol that performs an operation on variables and values.**

## Types of Operators:

**1.Arithmetic Operators**
- Perform basic math operations
- +, -, *, /, %
- **Example**: a + b, x * y

**2.Assignment Operators**
- Assign values to variables
- =, +=, -=, *=, /=, %=
- **Example**: a = 5, b += 2

**3.Relational (Comparison) Operators**
- Compare two values
- ==, !=, >, <, >=, <=
- **Example**: a == b, x < y

**4.Logical Operators**
- Combine or invert logical values
- && (AND), || (OR), ! (NOT)
- **Example**: a > 0 && b < 10

**5.Increment/Decrement Operators**
- Increase or decrease value by 1
- ++, -- (prefix/postfix)
- **Example**: ++x, y--

**6.Bitwise Operators**
- Operate on bits
- &, |, ^, ~, <<, >>
- **Example**: a & b, x << 2

**7.Conditional (Ternary) Operator**
- Short form of if-else
- ? :
- **Example**: max = (a > b) ? a : b;

**8.Sizeof Operator**
- Returns size of a data type or variable
- **Example**: sizeof(int), sizeof(arr)

## Example 1 : Arithmetic Operators

```cpp
#include <iostream>
using namespace std;

int main() {
    int a = 10, b = 3;
    cout << "Addition: " << a + b << endl;
    cout << "Subtraction: " << a - b << endl;
    cout << "Multiplication: " << a * b << endl;
    cout << "Division: " << a / b << endl;
    cout << "Modulus: " << a % b << endl;
    return 0;
}
```

```
Addition: 13
Subtraction: 7
Multiplication: 30
Division: 3
Modulus: 1
```

## Example 2 : Assignment Operators

```
#include <iostream>
using namespace std;

int main() {
    int a = 10;
    cout << "Original a: " << a << endl;
    a += 5;
    cout << "After a += 5: " << a << endl;
    a -= 3;
    cout << "After a -= 3: " << a << endl;
    a *= 2;
    cout << "After a *= 2: " << a << endl;
    a /= 4;
    cout << "After a /= 4: " << a << endl;
    a %= 3;
    cout << "After a %= 3: " << a << endl;
    return 0;
}
```

```
Original a: 10
After a += 5: 15
After a -= 3: 12
After a *= 2: 24
After a /= 4: 6
After a %= 3: 0
```

# Example 3 : Relational Operators

```cpp
#include <iostream>
using namespace std;

int main() {
    int a = 5, b = 7;

    cout << "a == b: " << (a == b) << endl;
    cout << "a != b: " << (a != b) << endl;
    cout << "a > b : " << (a > b) << endl;
    cout << "a < b : " << (a < b) << endl;
    cout << "a >= b: " << (a >= b) << endl;
    cout << "a <= b: " << (a <= b) << endl;

    return 0;
}
```

```
a == b: 0
a != b: 1
a > b : 0
a < b : 1
a >= b: 0
a <= b: 1
```

# Example 4 : Relational Operators

```cpp
#include <iostream>
using namespace std;

int main() {
    int a = 6, b = 8;

    cout << "a > 5 && b < 10: " << (a > 5 && b < 10) << endl;
    cout << "a > 5 || b > 10: " << (a > 5 || b > 10) << endl;
    cout << "!(a > 5): " << !(a > 5) << endl;

    return 0;
}
```

```
a > 5 && b < 10: 1
a > 5 || b > 10: 1
!(a > 5): 0
```

# Example 5 : Increment and Decrement Operators

```
#include <iostream>
using namespace std;

int main() {
    int a = 5;

    cout << "Post-increment a++: " << a++ << endl;  // prints 5, then a becomes 6
    cout << "After post-increment, a: " << a << endl;

    cout << "Pre-increment ++a: " << ++a << endl;   // a becomes 7, then prints 7

    cout << "Post-decrement a--: " << a-- << endl;  // prints 7, then a becomes 6
    cout << "Pre-decrement --a: " << --a << endl;   // a becomes 5, then prints 5

    return 0;
}
```

1. Prefix (e.g., ++x or --x)
First changes the value, then uses it.
2. Postfix (e.g., x++ or x--)
First uses the current value, then changes it.


# Thank You