Arithmetic Operators

An **operator** is a symbol that operates on a value to perform specific mathematical or logical computations. They form the foundation of any programming language. In C++, we have built-in operators to provide the required functionality.

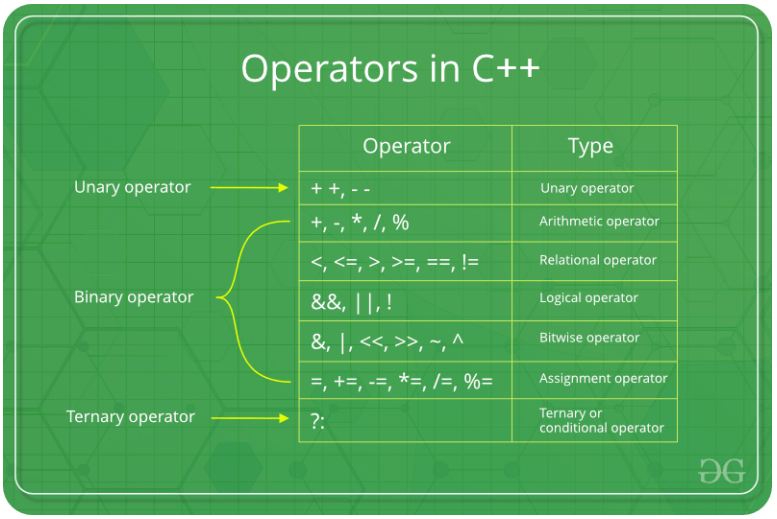
An operator operates the **operands**. For example,

```
int c = a + b;
```

Here, '+' is the addition operator. 'a' and 'b' are the operands that are being 'added'.

Operators in C++ can be classified into 6 types:

1. **Arithmetic Operators**
2. Relational Operators
3. Logical Operators
4. Bitwise Operators
5. Assignment Operators
6. Ternary or Conditional Operators



Arithmetic Operators

These operators are used to perform arithmetic or mathematical operations on the operands. For example, '+' is used for addition, '-' is used for subtraction '*' is used for multiplication, etc.

Arithmetic Operators can be classified into 2 Types:

A) Unary Operators: These operators operate or work with a single operand. For example: Increment(++) and Decrement(–) Operators.

Name	Symbol	Description	Example
Increment Operator	++	Increases the integer value of the variable by one	int a = 5; a++; // returns 6
Decrement Operator	—	Decreases the integer value of the variable by one	int a = 5; a–; // returns 4

▼ **Example:**

```
// CPP Program to demonstrate the increment
// and decrement operators
#include <iostream>
using namespace std;

int main()
{
    int a = 10;
    cout << "a++ is " << a++ << endl;
    cout << "++a is " << ++a << endl;

    int b = 15;
    cout << "b-- is " << b-- << endl;
    cout << "--b is " << --b << endl;

    return 0;
}
```

Output

```
a++ is 10
++a is 12
b-- is 15
--b is 13
```

> Note: ++a and a++, both are increment operators, however, both are slightly different.

> *In ++**a**, the value of the variable is incremented first and then It is used in the program. In **a**++, the value of the variable is assigned first and then It is incremented. Similarly happens for the decrement operator.*

B) Binary Operators: These operators operate or work with two operands. For example: Addition(+), Subtraction(-), etc.

Name	Symbol	Description	Example
Addition	+	Adds two operands	int a = 3, b = 6; int c = a+b; // c = 9
Subtraction	–	Subtracts second operand from the first	int a = 9, b = 6; int c = a-b; // c = 3
Multiplication	*	Multiplies two operands	int a = 3, b = 6; int c = a*b; // c = 18
Division	/	Divides first operand by the second operand	int a = 12, b = 6; int c = a/b; // c = 2
Modulo Operation	%	Returns the remainder an integer division	int a = 8, b = 6; int c = a%b; // c = 2

> Note: The Modulo operator(%) operator should only be used with integers.

▼ **Example:**

```
// CPP Program to demonstrate the Binary Operators#include

int main()
{
    int a = 8, b = 3;

    // Addition operator
    cout << "a + b = " << (a + b) << endl;

    // Subtraction operator
    cout << "a - b = " << (a - b) << endl;
```

```
    // Multiplication operator
    cout << "a * b = " << (a * b) << endl;

    // Division operator
    cout << "a / b = " << (a / b) << endl;

    // Modulo operator
    cout << "a % b = " << (a % b) << endl;

    return 0;
}
```

Output

```
a + b = 11
a - b = 5
a * b = 24
a / b = 2
a % b = 2
```