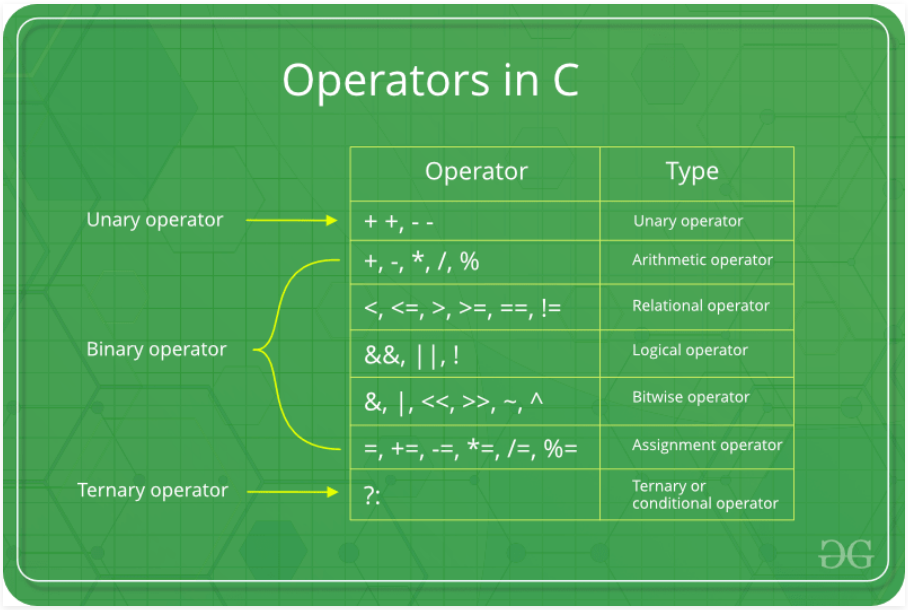
Operators in C | Set 1 (Arithmetic Operators)

Difficulty Level : Easy • Last Updated : 28 Jun, 2021



Operators are the foundation of any programming language. Thus the functionality of C language is incomplete without the use of operators. Operators allow us to perform different kinds of operations on operands. In C, operators in Can be categorized in following categories:

- **Arithmetic Operator**s (+, -, *, /, %, post-increment, pre-increment, post-decrement, pre-decrement)
- **Relational Operators** (==, !=, >, <, >= & <=) Logical Operators (&&, || and !)
- **Bitwise Operators** (&, |, ^, ~, >> and <<)
- **Assignment Operator**s (=, +=, -=, *=, etc)
- **Other Operators** (conditional, comma, sizeof, address, redirection)

- **Modulus:** The '%' operator returns the remainder when first operand is divided by the second. For example, **x%y**.

C

```c
// C program to demonstrate
// working of binary arithmetic
// operators
#include <stdio.h>

int main()
{
    int a = 10, b = 4, res;

    // printing a and b
    printf("a is %d and b is %d\n", a, b);

    res = a + b; // addition
    printf("a+b is %d\n", res);

    res = a - b; // subtraction
    printf("a-b is %d\n", res);

    res = a * b; // multiplication
    printf("a*b is %d\n", res);

    res = a / b; // division
    printf("a/b is %d\n", res);

    res = a % b; // modulus
    printf("a%%b is %d\n", res);

    return 0;
}
```

C++

```cpp
#include <iostream>
```

```
    // subtraction
    res = a - b;
    cout << "a-b is: "<< res << "\n";

    // multiplication
```



```
    // modulus
    res = a % b;
    cout << "a%b is: "<< res << "\n";

    return 0;
}
```

Output:

```
a is 10 and b is: 4
a+b is: 14
a-b is: 6
a*b is: 40
a/b is: 2
a%b is: 2
```

The ones falling into the category of unary arithmetic operators are:

- **Increment:** The **'++'** operator is used to increment the value of an integer. When placed before the variable name (also called pre-increment operator), its value is incremented instantly. For example, **++x**.
  And when it is placed after the variable name (also called post-increment operator), its value is preserved temporarily until the execution of this statement

C

```c
// C program to demonstrate working
// of Unary arithmetic
// operators
#include <stdio.h>

int main()
{
    int a = 10, b = 4, res;

    // post-increment example:
    // res is assigned 10 only, a is not updated yet
    res = a++;
    printf("a is %d and res is %d\n", a,
           res); // a becomes 11 now

    // post-decrement example:
    // res is assigned 11 only, a is not updated yet
    res = a--;
    printf("a is %d and res is %d\n", a,
           res); // a becomes 10 now

    // pre-increment example:
    // res is assigned 11 now since
    // a is updated here itself
    res = ++a;

    // a and res have same values = 11
    printf("a is %d and res is %d\n", a, res);

    // pre-decrement example:
    // res is assigned 10 only since a is updated here
    // itself
    res = --a;

    // a and res have same values = 10
    printf("a is %d and res is %d\n", a, res);

    return 0;
```

```
int main()
{
    int a = 10, b = 4, res;

    // post-increment example:
    // res is assigned 10 only,
    // a is not updated yet
    res = a++;
    // a becomes 11 now
    cout << "a is " << a
         << " and res is "
         << res << "\n";

    // post-decrement example:
    // res is assigned 11 only,
    // a is not updated yet
    res = a--;
    // a becomes 10 now
    cout << "a is " << a
         << " and res is "
         << res << "\n";

    // pre-increment example:
    // res is assigned 11 now
    // since a is updated here itself
    res = ++a;

    // a and res have same values = 11
    cout << "a is " << a
         << " and res is "
         << res << "\n";

    // pre-decrement example:
    // res is assigned 10 only
    // since a is updated here
    // itself
    res = --a;
    // a and res have same values = 10
    cout << "a is " << a
         << " and res is "
         << res << "\n";
```

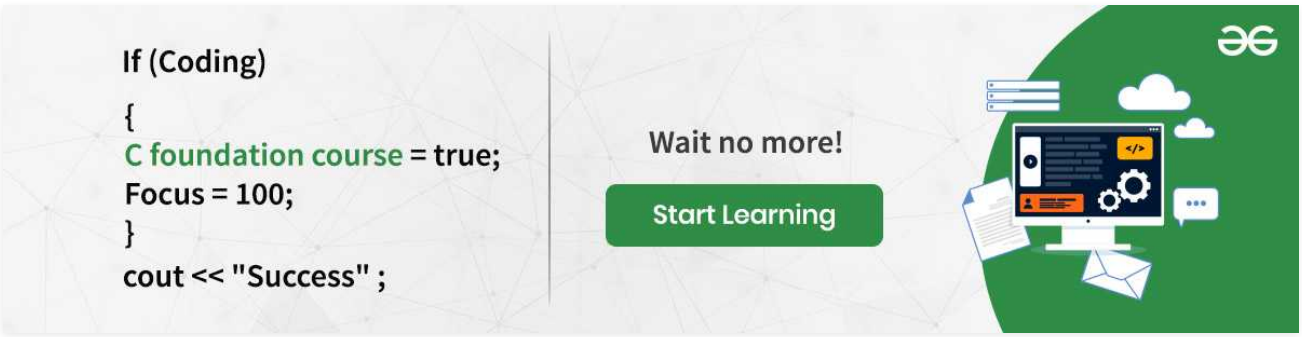
```
a is 10 and res is 10
```

We will soon be discussing other categories of operators in different posts.

To know about **Operator Precedence and Associativity**, refer this link:

Quiz on Operators in C

This article is contributed by Ayush Jaggi. Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above



Previous Next

RECOMMENDED ARTICLES Page : 1 2 3

03 Set a variable without using Arithmetic, Relational or Conditional Operator
02, Jul 16

04 Pointers in C and C++ | Set 1 (Introduction, Arithmetic and Array)
15, Dec 16

07 Output of C programs | Set 55 (Shift Operators)
20, Sep 17

08 Output of C programs | Set 58 (operators)
02, Oct 17

Article Contributed By :

GeeksforGeeks

Vote for difficulty

Current difficulty : Easy

Easy | Normal | Medium | Hard | Expert

Improved By : sirraghavgupta, akhilsharma870, kartikbagri199

Article Tags : C-Operators, cpp-operator, C Language

Practice Tags : cpp-operator

GeeksforGeeks
5th Floor, A-118,
Sector-136, Noida, Uttar Pradesh - 201305

feedback@geeksforgeeks.org