

Learn CODE: Start with C

Article is prepared by –

Sorowar Mahabub, Studies B.Sc in CSE,

Int' Islamic University Chittagong (IIUC)

Cell: 01834756433, 01521564157(whasApp available)

Email : sorowarmahabub1709vip@gmail.com

Article Reference : Google, DataFlair_blog

4 Types of **Operators in C** and C++ (Enhance Your Fundamental Skills)

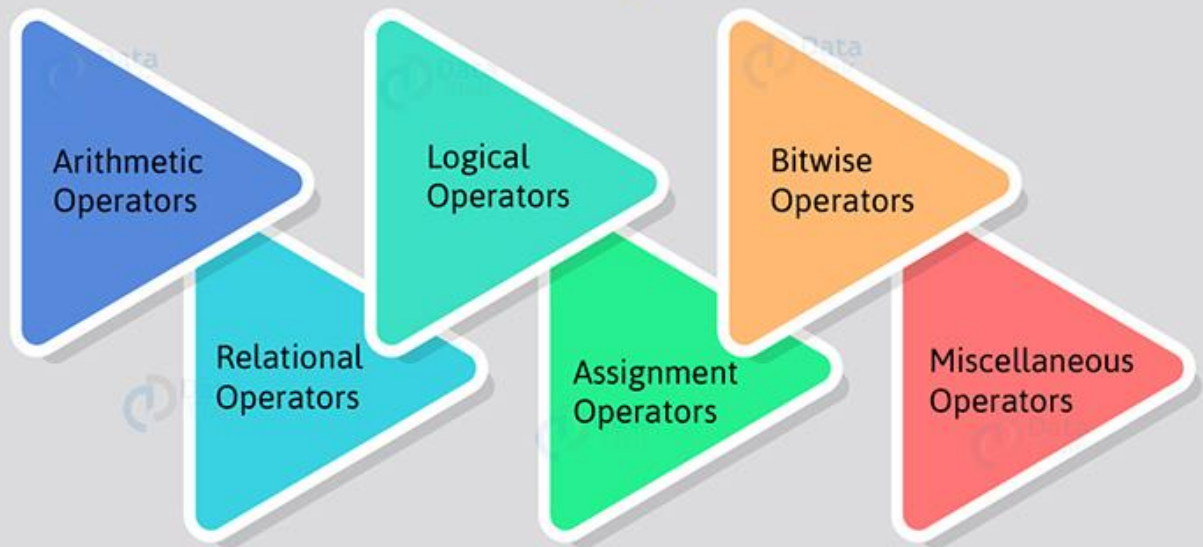
Operators are the basic concept of any programming language, used to build a foundation in programming for freshers. Operators can be defined as basic symbols that help us work on logical and mathematical operations. Operators in C and C++, are tools or symbols that are used to perform mathematical operations concerning arithmetic, logical, conditional and, bitwise operations.

There are many sub-operators presents in each type of Operators in C/C++. Let's discuss one by one with their examples.

Types of Operators in C and C++

There are 6 types of Operators in C/C++

Operators in C / C++



(This Picture Collected from DataFlair)

Let us discuss in detail the function of each type of operator.

1. Arithmetic Operators

It includes basic arithmetic operations like addition, subtraction, multiplication, division, modulus operations, increment, and decrement.

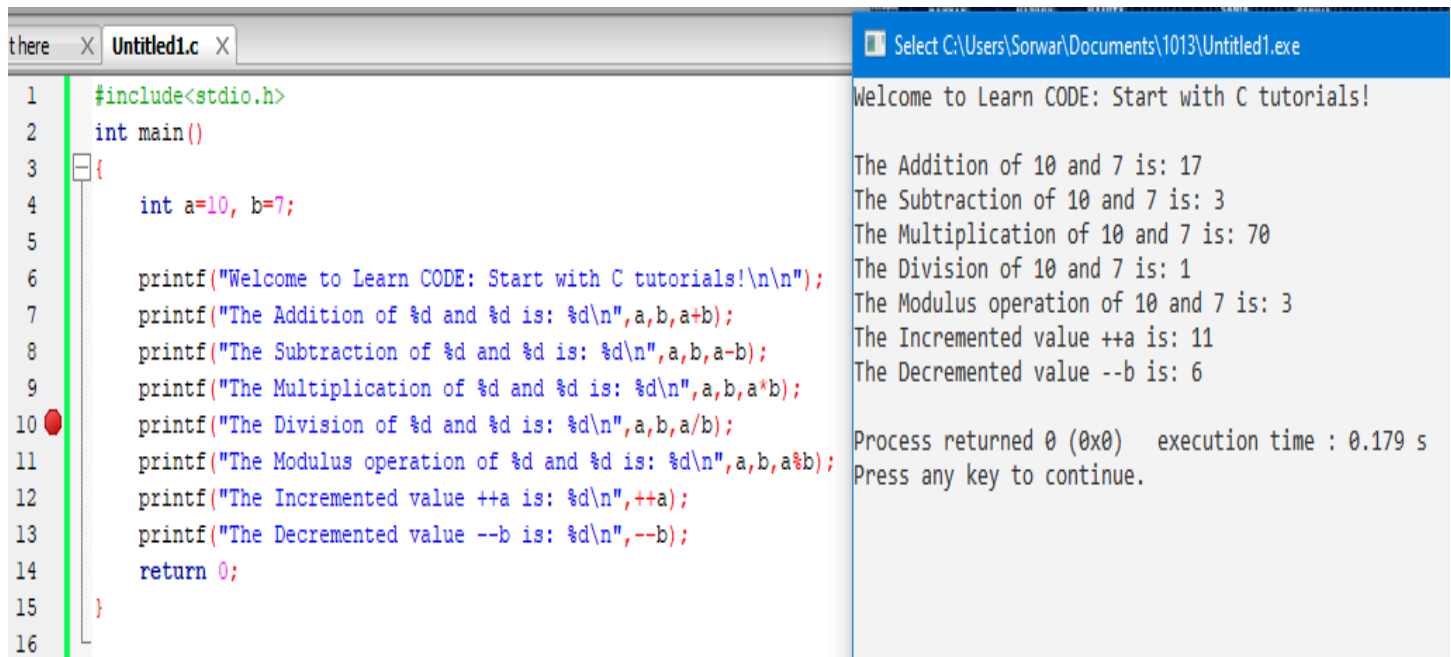
The Arithmetic Operators in C and C++ include:

1. **+** (**Addition**) – This operator is used to add two operands.
2. **-** (**Subtraction**) – Subtract two operands.
3. ***** (**Multiplication**) – Multiply two operands.
4. **/** (**Division**) – Divide two operands and gives the quotient as the answer.
5. **%** (**Modulus operation**) – Find the remains of two integers and gives the remainder after the division.
6. **++** (**Increment**) – Used to increment an operand.
7. **—** (**Decrement**) – Used to decrement an operand.

Example of Arithmetic Operators in C

```
1. #include<stdio.h>
2. int main()
3. {
4. int a=10, b=7;
5.
6. printf("Welcome to Learn CODE: Start with C tutorials!\n\n");
7. printf("The Addition of %d and %d is: %d\n",a,b,a+b);
8. printf("The Subtraction of %d and %d is: %d\n",a,b,a-b);
9. printf("The Multiplication of %d and %d is: %d\n",a,b,a*b);
10. printf("The Division of %d and %d is: %d\n",a,b,a/b);
11. printf("The Modulus operation of %d and %d is: %d\n",a,b,a%b);
12. printf("The Incremented value ++a is: %d\n",++a);
13. printf("The Decrementd value --b is: %d\n",--b);
14. return 0;
15. }
```

Mentioned CODE & OUTPUT on Screen:

The screenshot shows a code editor window titled 'Untitled1.c' with the following C code:

```
1 #include<stdio.h>
2 int main()
3 {
4     int a=10, b=7;
5
6     printf("Welcome to Learn CODE: Start with C tutorials!\n\n");
7     printf("The Addition of %d and %d is: %d\n",a,b,a+b);
8     printf("The Subtraction of %d and %d is: %d\n",a,b,a-b);
9     printf("The Multiplication of %d and %d is: %d\n",a,b,a*b);
10    printf("The Division of %d and %d is: %d\n",a,b,a/b);
11    printf("The Modulus operation of %d and %d is: %d\n",a,b,a%b);
12    printf("The Incremented value ++a is: %d\n",++a);
13    printf("The Decrementd value --b is: %d\n",--b);
14    return 0;
15 }
```

The output window on the right shows the following text:

```
Welcome to Learn CODE: Start with C tutorials!

The Addition of 10 and 7 is: 17
The Subtraction of 10 and 7 is: 3
The Multiplication of 10 and 7 is: 70
The Division of 10 and 7 is: 1
The Modulus operation of 10 and 7 is: 3
The Incremented value ++a is: 11
The Decrementd value --b is: 6

Process returned 0 (0x0)   execution time : 0.179 s
Press any key to continue.
```

Table for Arithmetic Operators in C and C++

Operator	Operand	Operation	Elucidation
+	a, b	$a + b$	Addition
-	a, b	$a - b$	Subtraction

*	a, b	a * b	Multiplication
/	a, b	a / b	Division
%	a, b	a % b	Modulus operator – to find the remainder when two integral digits are divided
++	a	a ++	Increment
—	a	a —	Decrement

2. Relational Operators

It is used to compare two numbers by checking whether they are equal or not, less than, less than or equal to, greater than, greater than or equal to.

1. **== (Equal to)**– This operator is used to check if both operands are equal.
2. **!= (Not equal to)**– Can check if both operands are not equal.
3. **> (Greater than)**– Can check if the first operand is greater than the second.
4. **< (Less than)**- Can check if the first operand is lesser than the second.
5. **>= (Greater than equal to)**– Check if the first operand is greater than or equal to the second.
6. **<= (Less than equal to)**– Check if the first operand is lesser than or equal to the second

If the relational statement is satisfied (it is true), then the program will return the value 1, otherwise, if the relational statement is not satisfied (it is false), the program will return the value 0.

Example of Relational Operators in C-

```

1. #include <stdio.h>
2. int main()
3. {
4.   int a=10, b=10, c=20;
5.
6.   printf("Welcome to Learn CODE: Start with C tutorials!\n\n");
7.
8.   printf("For %d == %d : The output is: %d \n", a, b, a == b); // condition is true

```

```

9. printf("For %d == %d : The output is: %d \n", a, c, a == c); // condition is false
10.
11. printf("For %d != %d : The output is: %d \n", a, c, a != c); // condition is true
12. printf("For %d != %d : The output is: %d \n", a, b, a != b); // condition is false
13.
14. printf("For %d > %d : The output is: %d \n", a, b, a > b); // condition is false
15. printf("For %d > %d : The output is: %d \n", a, c, a > c); // condition is false
16.
17. printf("For %d < %d : The output is: %d \n", a, b, a < b); // condition is false
18. printf("For %d < %d : The output is: %d \n", a, c, a < c); // condition is true
19.
20. printf("For %d >= %d : The output is: %d \n", a, b, a >= b); // condition is true
21. printf("For %d >= %d : The output is: %d \n", a, c, a >= c); // condition is false
22.
23. printf("For %d <= %d : The output is: %d \n", a, b, a <= b); // condition is true
24. printf("For %d <= %d : The output is: %d \n", a, c, a <= c); // condition is true
25. return 0;
26. }

```

Code on Screen:

<pre> 1 #include <stdio.h> 2 int main() 3 { 4 int a=10, b=10, c=20; 5 6 printf("Welcome to Learn CODE: Start with C tutorials!\n\n"); 7 8 printf("For %d == %d : The output is: %d \n", a, b, a == b); // condition is true 9 printf("For %d == %d : The output is: %d \n", a, c, a == c); // condition is false 10 11 printf("For %d != %d : The output is: %d \n", a, c, a != c); // condition is true 12 printf("For %d != %d : The output is: %d \n", a, b, a != b); // condition is false 13 14 printf("For %d > %d : The output is: %d \n", a, b, a > b); // condition is false 15 printf("For %d > %d : The output is: %d \n", a, c, a > c); // condition is false 16 17 printf("For %d < %d : The output is: %d \n", a, b, a < b); // condition is false 18 printf("For %d < %d : The output is: %d \n", a, c, a < c); // condition is true 19 20 printf("For %d >= %d : The output is: %d \n", a, b, a >= b); // condition is true 21 printf("For %d >= %d : The output is: %d \n", a, c, a >= c); // condition is false 22 23 printf("For %d <= %d : The output is: %d \n", a, b, a <= b); // condition is true 24 printf("For %d <= %d : The output is: %d \n", a, c, a <= c); // condition is true 25 return 0; 26 } 27 </pre>	<pre> Select C:\Users\Sorwar\Documents\1013\Untitled1.exe Welcome to Learn CODE: Start with C tutorials! For 10 == 10 : The output is: 1 For 10 == 20 : The output is: 0 For 10 != 20 : The output is: 1 For 10 != 10 : The output is: 0 For 10 > 10 : The output is: 0 For 10 > 20 : The output is: 0 For 10 < 10 : The output is: 0 For 10 < 20 : The output is: 1 For 10 >= 10 : The output is: 1 For 10 >= 20 : The output is: 0 For 10 <= 10 : The output is: 1 For 10 <= 20 : The output is: 1 Process returned 0 (0x0) execution time : 0.044 s Press any key to continue. </pre>
---	---

Table for Relational Operators in C and C++

Operator	Operand	Operation	Elucidation
==	a, b	(a==b)	Used to check if both operands are equal
!=	a, b	(a!=b)	Used to check if both operands are not equal
>	a, b	(a>b)	Used to check if the first operand is greater than the second
<	a, b	(a<b)	Used to check if the first operand is lesser than the second
>=	a, b	(a>=b)	Used to check if the first operand is greater than or equal to the second
<=	a, b	(a<=b)	Used to check if the first operand is lesser than or equal to the second

3. Logical Operators

It refers to the boolean values which can be expressed as:

- Binary logical operations, which involves two variables: AND and OR
- Unary logical operation: NOT

Logical Operators in C/C++ Includes –

1. **&& (AND)** – It is used to check if both the operands are true.
2. **|| (OR)** – These operators are used to check if at least one of the operand is true.
3. **! (NOT)** – Used to check if the operand is false

If the logical statement is satisfied (it is true), then the program will return the value 1, otherwise, if the relational statement is not satisfied (it is false), the program will return the value 0.

Example of Logical Operators in C Programming-

```
1. #include <stdio.h>
2. int main()
3. {
4.     int a = 10, b = 10, c = 20, answer;
5.
6.     printf("Welcome to Learn CODE: Start with C tutorials!\n\n");
7.
8.     answer = (a == b) && (c > b);
9.     printf("For (%d == %d) && (%d != %d), the output is: %d \n",a,b,b,c,answer); //condition is true
10.
11.    answer = (a == b) && (c < b) && (c>0);
12.    printf("For (%d == %d) && (%d <= %d), the output is: %d \n",a,b,b,c,answer); //condition is false
13.
```

```

14. answer = (a == b) || (b > c);
15. printf("For (%d == %d) || (%d < %d), the output is: %d \n",a,b,c,b,answer); //condition is true
16.
17. answer = (a != b) || (a <= b) || (a > c);
18. printf("For (%d != %d) || (%d < %d), the output is: %d \n",a,b,c,b,answer); //condition is true
19.
20. answer = !(a == b);
21. printf("For !(%d == %d), the output is: %d \n",a,b,answer); //condition is false
22.
23. answer = !(a != b);
24. printf("For !(%d == %d), the output is: %d \n",a,b,answer); //condition is true
25. return 0;
26. }

```

Code on Screen-

```

1  #include <stdio.h>
2  int main()
3  {
4      int a = 10, b = 10, c = 20, answer;
5
6      printf("Welcome to Learn CODE: Start with C tutorials!\n\n");
7
8      answer = (a == b) && (c > b);
9      printf("For (%d == %d) && (%d != %d), the output is: %d \n",a,b,c,b,answer); //condition is true
10
11     answer = (a == b) && (c < b) && (c > 0);
12     printf("For (%d == %d) && (%d <= %d), the output is: %d \n",a,b,c,b,answer); //condition is false
13
14     answer = (a == b) || (b > c);
15     printf("For (%d == %d) || (%d < %d), the output is: %d \n",a,b,c,b,answer); //condition is true
16
17     answer = (a != b) || (a <= b) || (a > c);
18     printf("For (%d != %d) || (%d < %d), the output is: %d \n",a,b,c,b,answer); //condition is true
19
20     answer = !(a == b);
21     printf("For !(%d == %d), the output is: %d \n",a,b,answer); //condition is false
22
23     answer = !(a != b);
24     printf("For !(%d == %d), the output is: %d \n",a,b,answer); //condition is true
25     return 0;
26 }

```

Welcome to Learn CODE: Start with C tutorials!

For (10 == 10) && (10 != 20), the output is: 1
For (10 == 10) && (10 <= 20), the output is: 0
For (10 == 10) || (20 < 10), the output is: 1
For (10 != 10) || (20 < 10), the output is: 1
For !(10 == 10), the output is: 0
For !(10 == 10), the output is: 1

Process returned 0 (0x0) execution time : 0.086 s
Press any key to continue.

Table for Logical Operators in C and C++

Operator	Operand	Operation	Elucidation
&&	a, b	(a && b)	AND: Used to check if both the operands are true
	a, b	(a b)	OR: Used to check if at least one of the operand is true
!	a	!a	NOT: Used to check if the operand is false

4. Assignment Operators

It is used to assign a particular value to a variable. We will discuss it in detail in the later section with its shorthand notations.

1. **= (Assignment)**- Used to assign a value from right side operand to left side operand.
2. **+= (Addition Assignment)**- To store the sum of both the operands to the left side operand.
3. **-= (Subtraction Assignment)** – To store the difference of both the operands to the left side operand.
4. ***= (Multiplication Assignment)** – To store the product of both the operands to the left side operand.
5. **/= (Division Assignment)** – To store the division of both the operands to the left side operand.
6. **%= (Remainder Assignment)** – To store the remainder of both the operands to the left side operand.

Example of Assignment Operators in C

```

1. #include<stdio.h>
2. int main()
3. {
4.     printf("Welcome to Learn CODE: Start with C tutorials!\n\n");
5.
6.     int number = 10, result;
7.     result = number;
8.
9.     printf("result = %d \n", result);
10.
11.    result += number; //Same as result = result + a
12.    printf("result = %d \n", result);
13.

```



```

14.result -= number; //Same as result = result - a
15.printf("result = %d \n", result);
16.
17.result *= number; //Same as result = result * a
18.printf("result = %d \n", result);
19.
20.result /= number; //Same as result = result / a
21.printf("result = %d \n", result);
22.
23.result %= number; //Same as result = result % a
24.printf("result = %d \n", result);
25.return 0;
26.}

```

Table for Assignment Operators in C and C++

Operator	Operand	Operation	Elucidation
=	a, b	a=b	Used to assign a value from right side operand to left side operand
+=	a, b	a+=b	a=a+b: The value of a+b is stored in a
-=	a, b	a-=b	a=a-b: The value of a-b is stored in a
=	a, b	a=b	a=a*b: The value of a*b is stored in a
/=	a, b	a/=b	a=a/b: The value of a/b is stored in a
%=	a, b	a%=b	a=a %b: The value of a%b is stored in a

Summary

Operators are the basic foundation of the C/C++ Programming language. Now, you can perform any operation of mathematical, logical, relational, with other condition. We learned each operator in C and C++ with their examples. As a beginner, you should know each operator, and how, why, when to use it.

Bismillah Hir Rahmanir Rahim

"Let's appear in thy smiling"



Md. Sorowar Mahabub Rabby | 01834-756433 01756-043655

*Studies B. Sc in CSE(Computer Science & Engineering),
Int' Islamic University Chittagong (IIUC).*

Baroiyerhat, Mirsharai, Chittagong.

 SOrowar Mahabub
 

 SOrowar Mahabub
 sorowarmahabub1709vip@gmail.com