BITWISE OPERATORS

In arithmetic-logic unit, mathematical operations like: addition, subtraction, multiplication and division are done in bit-level. To perform bit-level operations in C programming, bitwise operators are used.

| **Operators** | **Meaning of operators** |
| --- | --- |
| & | [Bitwise AND](https://www.programiz.com/c-programming/bitwise-operators" \l "and) |
| | | [Bitwise OR](https://www.programiz.com/c-programming/bitwise-operators" \l "or) |
| ^ | [Bitwise XOR](https://www.programiz.com/c-programming/bitwise-operators" \l "xor) |
| ~ | [Bitwise complement](https://www.programiz.com/c-programming/bitwise-operators" \l "complement) |
| << | [Shift left](https://www.programiz.com/c-programming/bitwise-operators" \l "left-shift) |
| >> | [Shift right](https://www.programiz.com/c-programming/bitwise-operators" \l "right-shift) |

## **Bitwise AND operator**

The output of bitwise AND is 1 if the corresponding bits of two operands is 1. If either bit of an operand is 0, the result of corresponding bit is evaluated to 0.

### EXAMPLE:- Bitwise AND

#include <stdio.h>int main(){

int a = 12, b = 25;

printf("Output = %d", a&b);

return 0;

}

**Output** = 8

## **Bitwise OR operator**

The output of bitwise OR is 1 if at least one corresponding bit of two operands is 1. In C Programming, bitwise OR operator is denoted by |.

### Example: Bitwise OR

#include <stdio.h>int main(){

int a = 12, b = 25;

printf("Output = %d", a|b);

return 0;

}

**Output** = 29

## **Bitwise XOR (exclusive OR) operator ^**

The result of bitwise XOR operator is 1 if the corresponding bits of two operands are opposite. It is denoted by ^.

### Example: Bitwise XOR

#include <stdio.h>int main(){

int a = 12, b = 25;

printf("Output = %d", a^b);

return 0;

}

**Output** = 21

## Bitwise complement operator ~

Bitwise compliment operator is an unary operator (works on only one operand). It changes 1 to 0 and 0 to 1. It is denoted by ~.

### Bitwise complement of any number N is -(N+1). Eg:-

bitwise complement of N = ~N (represented in 2's complement form)

2'complement of ~N= -(~(~N)+1) = -(N+1)

### Example #4: Bitwise complement

#include <stdio.h>int main()

{

printf("Output = %d\n",~35);

printf("Output = %d\n",~-12);

return 0;

}

**Output** = -36

Output = 11

## **Shift Operators in C programming**

There are two shift operators in C programming:

* Right shift operator
* Left shift operator.

### Right Shift Operator

Right shift operator shifts all bits towards right by certain number of specified bits. It is denoted by >>.

## Left Shift Operator

Left shift operator shifts all bits towards left by a certain number of specified bits. The bit positions that have been vacated by the left shift operator are filled with 0. The symbol of the left shift operator is <<.

### Example #5: Shift Operators

#include <stdio.h>int main()

{

int num=212, i;

for (i=0; i<=2; ++i)

printf("Right shift by %d: %d\n", i, num>>i);

printf("\n");

for (i=0; i<=2; ++i)

printf("Left shift by %d: %d\n", i, num<<i);

return 0;

}

OUTPUT :-

Right Shift by 0: 212

Right Shift by 1: 106

Right Shift by 2: 53

Left Shift by 0: 212

Left Shift by 1: 424

Left Shift by 2: 848

# Ternary Operator

The conditional operator is of the form

variable = Expression1 ? Expression2 : Expression3

It can be visualized into if-else statement as:

if(Expression1)

{

variable = Expression2;

}

else

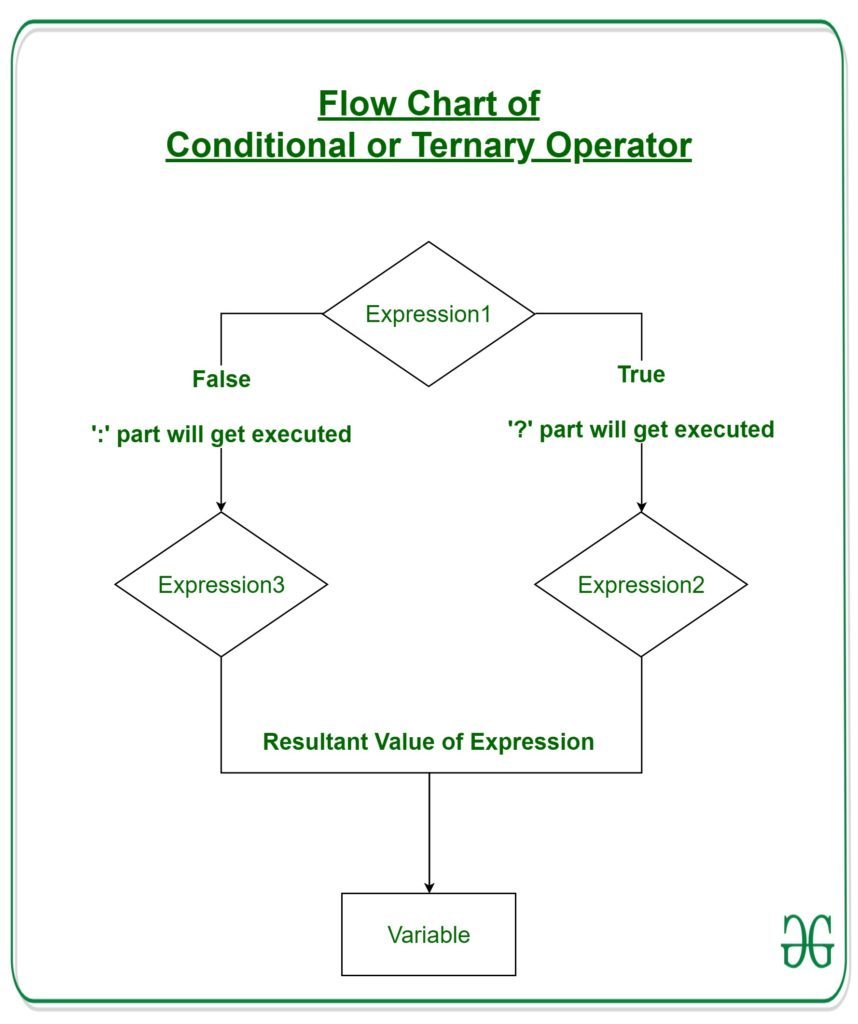
{

variable = Expression3;

}

Since the Conditional Operator ‘?:’ takes three operands to work, hence they are also called **ternary operators**

**Working:**  
Here, **Expression1** is the condition to be evaluated. If the condition(**Expression1**) is True then **Expression2** will be executed and the result will be returned. Otherwise, if the condition(**Expression1**) is false then **Expression3** will be executed and the result will be returned.

[](https://media.geeksforgeeks.org/wp-content/uploads/20190920114837/Flow-Chart-of-Conditional-or-Ternary-Operator-__-in-C_C.jpg)

**Example:** Program to Store the greatest of the two Number.

|  |
| --- |
| // C program to find largest among two  // numbers using ternary operator    #include <stdio.h>    int main()  {      // variable declaration      int n1 = 5, n2 = 10, max;        // Largest among n1 and n2      max = (n1 > n2) ? n1 : n2;        // Print the largest number      printf("Largest number between"             " %d and %d is %d. ",             n1, n2, max);        return 0;  } |

**Output:**

Largest number between 5 and 10 is 10.