Bitwise Operators in C and C++

* These **operators** are used to perform bit **operations**. Decimal values are converted into **binary** values which are the sequence of bits and **bit wise operators** work on these bits.
* The (**bitwise** OR) in C or **C++** takes two numbers as operands and does OR on every bit of two numbers. The result of OR is 1 if any of the two bits is 1. The ^ (**bitwise** XOR) in C or **C++** takes two numbers as operands and does XOR on every bit of two numbers.
* **Bit wise operators** in C language are & (**bitwise** AND), | (**bitwise** OR),
* ~ (**bitwise** NOT), ^ (XOR), << (left **shift**) and >> (right **shift**).
* **Bitwise operators** are special **operator** set provided by '**C** ' They are used in bit level **programming**. These **operators** are used to manipulate bits of an integer expression.
* Logical, shift and complement are three types of **bitwise operators**.

Program on Bitwise Operators:

#include <stdio.h>

main() {

unsigned int a = 60; /\* 60 = 0011 1100 \*/

unsigned int b = 13; /\* 13 = 0000 1101 \*/

int c = 0;

c = a & b; /\* 12 = 0000 1100 \*/

printf("Line 1 - Value of c is %d\n", c );

c = a | b; /\* 61 = 0011 1101 \*/

printf("Line 2 - Value of c is %d\n", c );

c = a ^ b; /\* 49 = 0011 0001 \*/

printf("Line 3 - Value of c is %d\n", c );

c = ~a; /\*-61 = 1100 0011 \*/

printf("Line 4 - Value of c is %d\n", c );

c = a << 2; /\* 240 = 1111 0000 \*/

printf("Line 5 - Value of c is %d\n", c );

c = a >> 2; /\* 15 = 0000 1111 \*/

printf("Line 6 - Value of c is %d\n", c );

}

//OUTPUT :

Line 1 - Value of c is 12

Line 2 - Value of c is 61

Line 3 - Value of c is 49

Line 4 - Value of c is -61

Line 5 - Value of c is 240

Line 6 - Value of c is 15

Process returned 0 (0x0) execution time : 0.203 s

Press any key to continue.