Bitwise exclusive OR operator

Bitwise exclusive OR operator is denoted by ^. Two operands are written on both sides of the exclusive OR operator. If the corresponding bit of any of the operand is 1 then the output would be 1, otherwise 0.

For example,

Let us consider two variables a and b,

```
a = 16;
b = 11;
```

The binary representations are:- a = 00010000
b = 00001011

a = 0 0 0 1 0 0 0 0

b = 0 0 0 0 1 0 1 1

---

a^b = 0 0 0 1 1 0 1 1 = 27

Hence, result of a=16, b=11 would be

a ^ b = 27.

Let us understand with an example:—

```
#include <stdio.h>
    int main(){ int a = 16, b=11;
printf("\n The a^b = %d", a^b);
    return 0;
}
```

Bitwise complement operator (~)

The bitwise complement operator is also known as one's complement operator. It takes only one operand or variable and performs complement operation on an operand. When we apply the complement operation on any bit, then 0 becomes 1 and 1 becomes 0.

For example, if we have int a=8;

The binary representation of the above variable is given as:-

a = 1000;

When we apply ~a; it becomes 0111 that means → -9.

Let us understand with the example:-

```
#include <stdio.h>
int main(){ int a=8;
printf("\n Complement of a ~a = %d", ~a);
return 0;
}
```

Output: Complement of a ~a = -9

Bitwise shift operators :-

There are two types of shift operators used on bits in C programming. The bitwise shift operators will shift the bits either on the left side or right side. Therefore, we can say that the bitwise shift operator is divided into two categories :-

→ Left-shift operator <<

→ Right-shift operator >>

Left-shift operator shifts the number of bits to the left-side.

Syntax :- Operand << n;

where, operand is an integer on which we apply the left-shift operation.

n is the number of bits to be shifted.

In the case of Left-shift operator, 'n' bits will be shifted on the left-side. The 'n' bits on the left side will be popped out, and n bits on the right-side are filled with 0.

For example :-

Suppose we have a statement :-

```
int a = 5;
```

The binary form of a is given as :-

a = 0101

If we want to left shift by 2, the statement would be :-

```
a << 2;
```

00010100 which is 20

Example:-

```
#include <stdio.h>
int main() { int i=5;
printf ("\n The value of a<<2 is: %d", a<<2);
return 0;
}
```

Output would be:—

The value of a <<2 is: 20