BITWISE OPERATORS:

These operators are used to perform mathematical operations at bit level. Decimal values are converted into binary values which are the sequence of bits and bitwise operators work on these bits.

1) &- Bitwise AND

2) |- Bitwise OR

3) ~- Bitwise NOT

4) ^- Bitwise XOR

5) <<- Left Shift

6) >>- Right shift

Consider x=40 and y=80. Binary form of these values is given below.

x = 00101000

y = 01010000

All bit wise operations for x and y are given below.

1. x & y = 00000000 (binary) = 0 (decimal)
2. x | y = 01111000 (binary) = 120 (decimal)
3. ~x=11111111111111111111111111111111111111111111010111 = -41 (decimal)
4. x ^ y = 01111000 (binary) = 120 (decimal)
5. x << 1 = 01010000 (binary) = 80 (decimal)

6. x >> 1 = 00010100 (binary) = 20 (decimal)

TERNARY OPERATOR:

The conditional/ternary operator is similar to if-else construct as it follows the same function but it takes up less space and helps write if-else statements in the shortest way possible.

Syntax:

variable = Expression1 ? Expression2 : Expression3

Here, Expression1 is evaluated and if it holds true, then Expression2 is executed and if it's false, Expression3 is executed. The resultant of the expression is stored in the variable.

Ex:

// C program to find minimum among two given numbers

```
#include <stdio.h>

int main ()
{
```

```
    int n1 = 5, n2 = 10, min;



    min = (n1 < n2) ? n1 : n2;

    printf("Minimum of %d and %d is %d. ", n1,n2,min);

    return 0;
}
```

The output of the code:

Minimum of 5 and 10 is 10.