1. Bitwise Operators:

Bitwise operators are operators that operate on 'ints' and 'uints' at the binary levels. This means they look directly at the binary digits or bits of an integer. The mathematical operations like: addition, subtraction, multiplication and division are done in bit-level. In C, the following 6 operators are bitwise operators: 1. & (bitwise AND), 2. | (bitwise OR), 3. ^ (bitwise XOR), 4. << (left shift), 5. >> (right shift), 6. ~ (bitwise NOT)

Example:

```
#include<stdio.h>

int main()

{

        //a=5(00000101), b=9(00001001)

        unsigned char a = 5, b = 9;


        // The result is 00000001

        printf("a = %d, b = %d/n", a, b);

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


        //The result is 00001101

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


        //The result is 00001100

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


        //The result is 11111010

        printf("~a = %d/n", a = ~a);
```

```
//The result is 00010010
printf("b<<1 = %d/n", b << 1);

//The result is 00000100
printf("b>>1 = %d/n", b >> 1);
return 0;
```

Output:

a = 5, b = 9

a&b = 1

a|b = 13

a^b = 12

~a = 250

b>1 = 4

2. Conditional or Ternary Operator:

The ternary operator is used to execute code based on the result of a binary condition. It takes in a binary condition as input, which makes it similar to an 'if-else' control flow block. It also, however, returns a value, behaving similar to a function. The ternary cannot be used to execute code. It must be either returned in a function, or set equal to a variable with the same data type as the returned values. It is also known as Ternary Operator.

Syntax:

variable = Expression1 ? Expression2 : Expression3

It can be visualised into if-else statement as:

if(Expression1)

{

```
        variable = Expression2;
}
else
{
        variable = Expression3;
}
```

Example:

```
#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.
```