|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Data type | description | Size (in bytes) | Format specifier | example |
| int | It stores all +,- and zero values | 4 bytes | %d | int num=10 |
| char | All values like symbols, numeric and alphabets | 1 byte | %c | char ch=’@’ |
| float | It stores decimal values also | 4 bytes | %f | float pi=3.14 |
| double | It stores numeric as well as decimal | 8 bytes | %lf | double d=300.14822 |

Note : 2 bytes in 16 bit devices 4 bytes in 32/64 bit devices

* Always use single quote for char.

1 byte = 8 bits

1kb =1024 bytes

1mb =1024 k**b**

**1 GB= 1024 mb**

-2n-1 to 2n-1 -1 🡪 where n= no. of bits

Int = 4 bytes 🡪 1 byte = 8 bits 🡪 4\*8 = 32 bits

-232-1 to 232-1 -1 🡪 -2,14,74,83,648 to 2,14,74,83,647 ans.

Convert 0001 0000 into decimal

🡪 0 x 27 + 0 x 26 + 0 x 25 + 1 x 24 + 0 x 23 + 0 x 22 + 0 x 21 + 0 x 20

* 16 ans.

19 🡪 10011 -> 0001 0011

1. Bitwise and( & ) 🡪 (16 & 19)

0001 0000

0001 0011

\_\_\_\_\_\_\_\_\_

0001 0000 🡪 16 ans.

\_\_\_\_\_\_\_\_\_

1. Bitwise or (|)

0001 0000

0001 0011

\_\_\_\_\_\_\_\_\_

0001 0011 🡪 19 ans.

\_\_\_\_\_\_\_\_\_

1. Bitwise not ( ~ )

0001 0011 🡪 11101100 🡪 236 ans.

1. Left shift ( << )

(16<<3) 🡪 00010000 🡪 1000 0000 🡪 128 ans.

1. Right shift ( >> )

(16>>3) 🡪 00010000 🡪 0000 0010 🡪 2 ans.

(01010101)₂ = (0 × 2⁷) + (1 × 2⁶) + (0 × 2⁵) +

(1 × 2⁴) + (0 × 2³) + (1 × 2²) + (0 × 2¹) + (1 × 2⁰)

= (85)₁₀ ans.

Miscellaneous operators

++ ( increment )

\_ \_ ( decrement )

Int num =10;

++num 🡺 10+1 🡺 11 ans.

--num 🡺 10-1 🡺 9 ans.