Instructions!!

① Keep a copy and a pen

② Be interactive

Bit Manipulation

By Sanika Khandija

What matters the most in CP?

→ faster coding time

→ faster run-time

Bit Manipulation

It is a very important topic to handle questions related to binary.

Number System

Decimal (0 - 9) → 10

Binary (0, 1) → 2

octal (0 - 7) → 8

Hexa Decimal (0 - 9, A - F) → 16

Basics of bit manipulation

① Decimal to Binary

2	102	
2	51	0
2	25	1
2	12	1
2	6	0
2	3	0

$(102)_{10} = (1100110)_2$

Decimal ↓ Binary

Code it up!

1 1

② Binary to Decimal

$(1101101)_2$

$2^6\ 2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0$

code it up!

\+ = 126

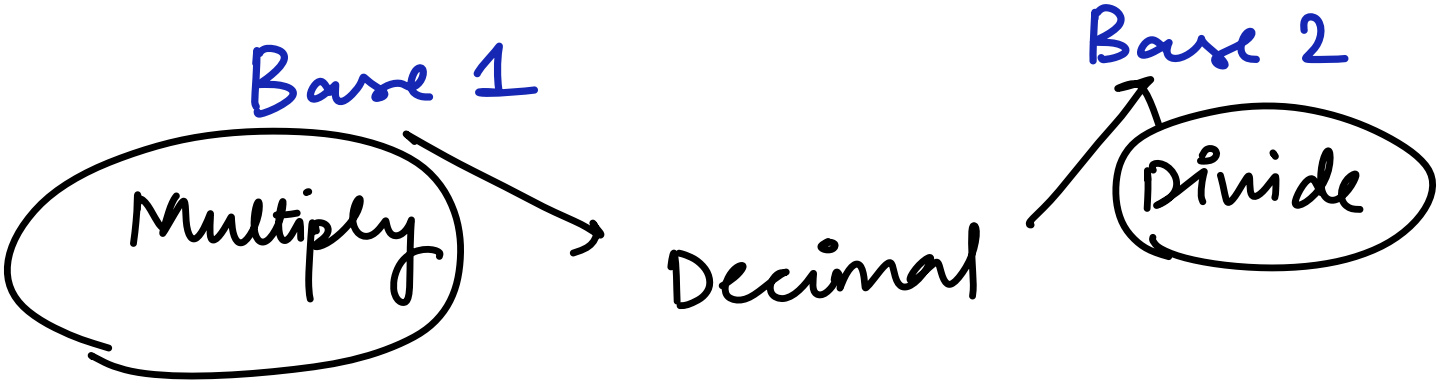
similary other conversions can be done like

octal — binary — octal - decimal
decimal - binary

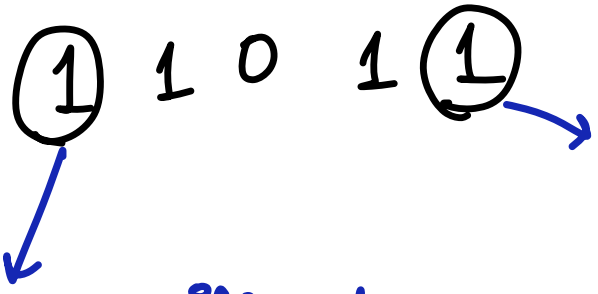
Hexadecimal — octal — Hexa - decimal
decimal - octal

binary — hexadecimal —

Simple shortcut



Now we are only concerned with Binary numbers.



Most significant bit (MSB)

(Last bit)

Least significant bit (LSB)

(1st bit)

Bitwise operators

① OR operator (|)

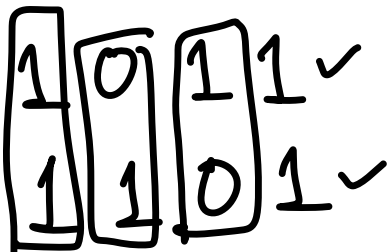
Bitwise OR gives 1 if either of the bits is set.

② AND operator (&)

Bitwise AND gives 1 if both the bits are set.

③ XOR operator (^)

even 1's → 0



odd 1's → 1

0 1 1 0

④ Bitwise Left shift operator <<

→ shift all the bits by one position to the left

0 1 1 0 1

1 1 0 1 0

Equivalent to multiply the number by 2

a << b

→ a << b = a * 2 ^ b ✓

5 , 5<<1 → 10 5 x 2^1

5<<2 → 20 5 x 2^2

a<<b → 5<<3 → 40 5 x 2^3

a>>b →

⑤ Bitwise Right shift operator

Shift all bits one position to the right

last

0 1 1 0 1

0 0 1 1 0

Equivalent to divide the number by 2

5 → 1 0 1 (2)

5 >>1 → 0 1 0 (2)

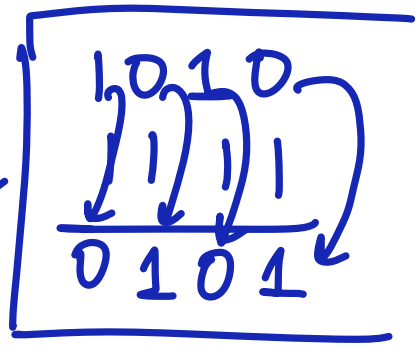
→ a >> b = a / 2 ^ b

In bit manipulation we will only be work with integers and not decimal point numbers.

5 >> 2 → 001

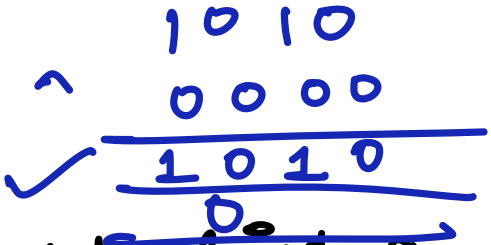
int — 32 bits ✓

long long — 64 bits



Properties of bitwise operators

① OR, AND, XOR are all commutative and associative



② $A \wedge A = 0$ → useful to find a single number

③ $A \wedge 0 = A$ ✓ (XOR with 0 gives same element)

④ $A \wedge 1 = A'$ (XOR with 1 toggles the bit) 1→0, 0→1

⑤ $a \& b <= \min(a, b)$ ✓✓

As we keep doing and the value ✓

keeps decreasing (a & b & c <= min(a,b,c))

⑥ a | b >= max(a, b)

As we keep doing or the value keeps increasing

⑦ cyclic XOR

A ^ B = C

A ^ C = B

B ^ C = A

10 → 1 0 1 0

b → 0 1 1 0

1 1 1 0

14

2 → 010, 4 → 100

3 → 011

5 → 101

7 → 111

⑧ For a number to be odd the lsb is always one, How to check.

if (num & 1 != 0) { → odd

cout << "number is odd" << endl

} (Better than % remainder check → faster)

n%2==0 → even

⑨ How to check if power of 2 or not?

A & A-1 == 0 then Yes

else No

How?

```
1000  → 8
0111  → 7
&
----
0000
----
```

Tricks and Tips using Bit Manipulation

① Set the k^{th} bit

0 → 1
1 → 1

```
1 0 1 0 1 1 0
| 0 0 0 1 0 0 0
-------------
1 0 1 1 1 1 0
```

0 | 1 = 1
1 | 1 = 1

0 based ← A | (1<<k)

A | (1<<k-1)

A | (1<<k-1)

1 based

② Check if the k^{th} bit is set or not?

0 0 0 1 0 1 0 1 0 1 1 0

& 0 0 0 0 0 0 0 1

0 0 0 0 0 0 0 0 ✗

& 0 0 0 1 0 0 0 0

0 0 0 1 0 0 0 0 ✓

$A \,\&\, (1 << k-1) \;!= 0$

$A \,\&\, (1 << k) \;!= 0$ → then set

$(A >> k) \,\&\, 1 \;!= 0$ → else not set

③ Toggle the k^{th} bit

1 0 1 0 1 1 0

^ 0 0 0 1 0 0 0

$A \,\hat{}\, (1 << k-1)$ ✓