

Bits Manipulation - I



Binary Number System

✓ Decimal \leftrightarrow binary

(13)₁₀ = (1101)₂

To convert

LCM,

$$\begin{array}{r} 2 \mid 13 & 1 \\ \quad\quad\quad 2 \mid 6 & 0 \\ \quad\quad\quad 2 \mid 3 & 1 \\ \quad\quad\quad 1 & \\ \end{array} \quad (1101)_2$$

Computer \rightarrow binary
 $\rightarrow 1010111$



4.5 MB

int \rightarrow 4 bytes
byte \rightarrow 1 byte. LSB
MSB

31 32 bits. 3 2 1 0

binary to Decimal

$$(10101)_2$$

4 3 2 1 0

$$\rightarrow 1 * 2^4 + 0 * 2^3 + 1 * 2^2 + 0 * 2^1 + 1 * 2^0 \\ = 16 + 0 + 4 + 0 + 1 = (21)_{10}$$

$$\begin{matrix} 2 & 1 & 0 \\ (135)_{10} \\ \downarrow \end{matrix}$$

$$1 * 10^2 + 3 * 10^1 + 5 * 10^0 \\ = 100 + 30 + 5 \\ = \underline{\underline{135}}$$

Binary Addition & Subtraction

$$\begin{array}{r} & 1 & 1 & 1 \\ & 9 & 4 & 8 \\ + & 1 & 8 & 7 & 9 \\ \hline & 2 & 8 & 2 & 7 \end{array}$$

$$\begin{array}{r} & 1 & 1 & 1 \\ & 1 & 0 & 1 & 1 & 1 \\ + & 0 & 1 & 1 & 0 \\ \hline & 1 & 1 & 1 & 0 & 1 \end{array}$$

$$\begin{array}{r} 1+1 = 2 \\ \downarrow \\ 1+1+1 = 3 \\ \downarrow \\ 11 \end{array}$$

$$9 - \underline{3} = 6$$

↓

$$9 + \underline{(-3)} = 6$$

Additive Inverse

$$(1100)_2 - (101)_2$$

$$12 - 5 \rightarrow 12 + \underline{(-5)}$$

[2's Compliment
is the additive inverse
of binary number]

Two Steps

1. Invert all bits

2. Add 1

$$\begin{array}{r} (0000000101) \rightarrow 5 \\ \downarrow \text{invert} \end{array}$$

$$\begin{array}{r} 1111111010 \\ \downarrow \text{add 1} \end{array}$$

$$(1111111011) \rightarrow 2's \text{ of } 5$$

$$12 \rightarrow \begin{array}{r} 1x \\ + 111111011 \\ \hline 0000000111 \\ \downarrow 421 \end{array} \rightarrow 7$$

$$\begin{array}{r} 1010 \\ 8020 \rightarrow \underline{10} \end{array}$$

$$\begin{array}{r|rr|l} & 12 & 0 \\ \hline 2 & 6 & 0 \\ 2 & 3 & 1 \\ - & - & \\ \hline & 1100 \end{array}$$

~~12 681~~

$$\begin{array}{r} 1100 \\ \hline \end{array}$$

+ , - , / , * , %

Bitwise Operators → l, &, ^, ~, >>, <<

a	b	a b	<u>a & b</u>	<u>a ^ b</u>	<u>~a</u>	000101
1	1	1	1	0	0	111111010
1	0	1	0	1	1	111111010
0	1	1	0	1	1	111111010
0	0	0	0	0	1	111111010

$a = 5 \quad 111111010$

$\sim a =$

$\sim a = 1(\underline{\hspace{10cm}})$

$32^m \quad 31 \text{ bits}$

$000101 \rightarrow \underline{0000010}$

$5 \gg 1 \leftarrow 00101$

$5 \ll 2 \leftarrow 10100$

Bit Masking

$\&$ with 1 \rightarrow remains same
 $\&$ with 0 \rightarrow number becomes 0

rob



Input



Mask



Output

\wedge with 0 \rightarrow remains the same



\wedge with 1 \rightarrow toggle

$$\begin{array}{rcl} \underline{1} \wedge \underline{1} & = & \underline{0} \\ \underline{0} \wedge \underline{1} & = & \underline{1} \end{array}$$

$$\begin{array}{rcl} \underline{1} \wedge 0 & = & \underline{1} \\ \underline{0} \wedge 0 & = & \underline{0} \end{array}$$

Find the ith Bit

Number →

8 7 6 5 4 3 2 1 0
1 0 0 0 0 1
8 0 0 0 0 1 0 0 0 0
<hr/>
0 0 0 0 1 0 0 0 0

find 4th bit → 1

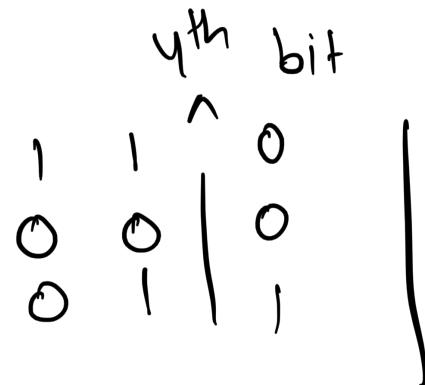
```
if ((n & mask) == 0)  
    return 0 ;  
  
else  
    return 1;  
  
mask = 1 << i
```

1. $n \geq mask$

2. find mask

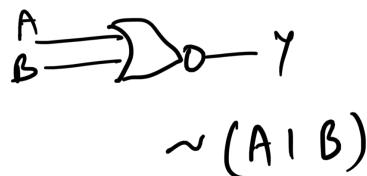
3. $mask = 1 \ll i$

Toggle the ith Bit



input \rightarrow 1 0 0 1 0 1 1 0 1
o/p \rightarrow (1 0 0 1 1 1 1 0 1)

$$\begin{array}{r} 1001\cancel{0}1101 \\ \wedge 000010000 \\ \hline 100101101 \end{array}$$



create OR from NAND

(63)

$$2^6 = 64$$

$$\begin{array}{r} 1000000 \\ - 111111 \\ \hline 0000000 \end{array}$$

Check if a number is power of Two

$$2^4 = 16 - 1$$

$$8 \rightarrow 1000$$

$$\begin{array}{r} 10000 \\ - 801111 \\ \hline 000000 \end{array}$$

$$\begin{array}{r} 100000 \\ - 101111 \\ \hline 000000 \end{array}$$

$$32 - 1 = 31$$

$$\begin{array}{r} 1001 \\ - 1000 \\ \hline 1000 \end{array}$$

Count the number of set bits in a Number

$N =$

Count = $\phi \times 2^8 + 5$

[and with 1]

$$\begin{array}{r} 1011010 \\ 20000001 \\ \hline 0000000 \end{array}$$
$$\begin{array}{r} 1011010 \\ 2000001 \\ \hline 000001 \end{array}$$

$$(10110101)_2$$

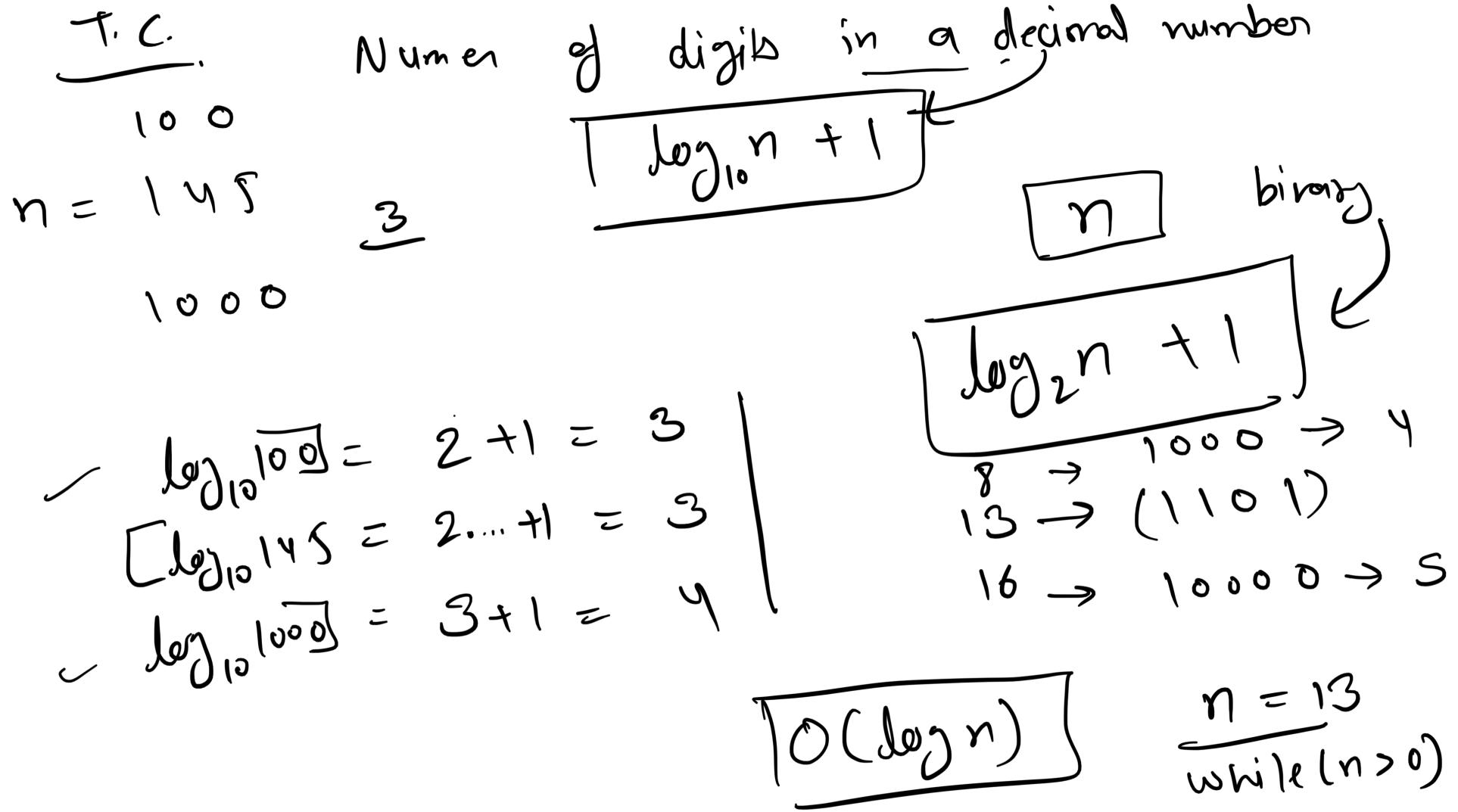
ans = 5

$$10110\overline{10}$$

$$1101101$$

$$\begin{array}{r} 10110 \\ \downarrow \\ 1011 \end{array} \rightarrow 101$$

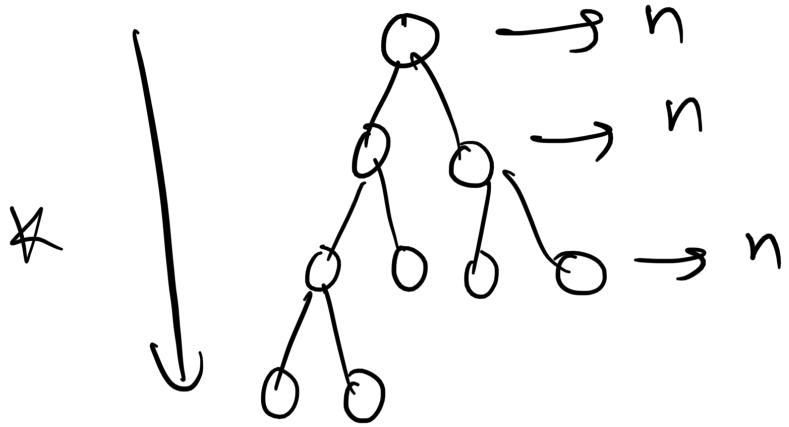
0
↑
1
↑
10
↑
100
↑
1000



Optimized
Count set bits

Practice Problems

1. Check if a number is odd or even.
2. Swap two numbers without using a third variable.
3. Set the ith Bit to 1.
4. Unset the ith bit / change ith bit to 0.
5. Given a number N, the task is to find the XOR of all numbers from 1 to N.
6. How many bits are required to change to convert one number to another.



$$k = \underline{(n \log n)}$$

