Bits Manipulation - I

Binary Number System

Computer → binary

→ 1 0 1 0 1 1 1

→ 4.5 MB

Decimal ⟷ binary

$(13)_{10} = (1101)_2$

To convert

LCM.

2	13	1
2	6	0
2	3	1
	1	

$(1101)_2$

int → 4 bytes

byte → 1 byte.

MSB

LSB

32 bits.

binary to Decimal

$(10101)_2$

4 3 2 1 0

$\rightarrow 1*2^4 + 0*2^3 + 1*2^2 + 0*2^1 + 1*2^0$

$= 16 + 0 + 4 + 0 + 1 = (21)_{10}$

2 1 0

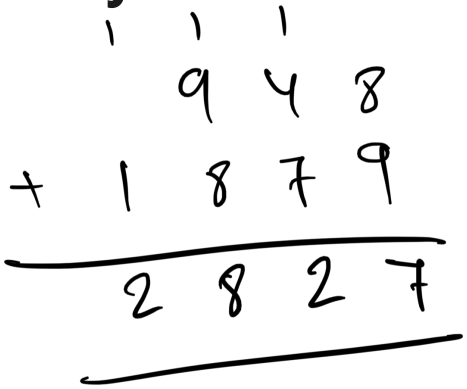
$(135)_{10}$

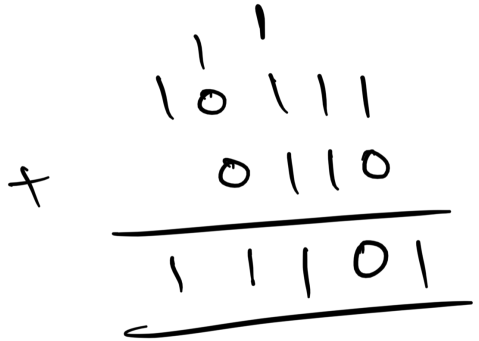
$1*10^2 + 3*10^1 + 5*10^0$

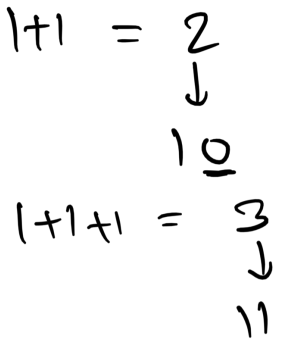
$= 100 + 30 + 5$

$= 135$

Binary Addition & Subtraction







$9 - 3 = 6$

$\downarrow$

$9 + (-3) = 6$

Additive Inverse

$(1100)_2 - (101)_2$

$12 - 5 \rightarrow 12 + (-5)$

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

Two steps

1. Invert all bits
2. Add 1

$(00000000101) \rightarrow 5$

$\downarrow$ invert

11111111010

$+1$ add 1

$\downarrow$

$(11111111011) \rightarrow$ 2's of 5

12 ⟶

```
   1 1
   0000001100
+  1111111011
------------
   0000000111   ⟶ 7
------------
       ↓
       4 2 1
```

1010

8020 → 10

2	12	0
2	6	0
2	3	1
	1	

1100

~~12 6 3 1~~

1100

+, -, /, *, %

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

a	b	a|b	a&b	a^b
1	1	1	1	0
1	0	1	0	1
0	1	1	0	1
0	0	0	0	0

~a 000101

a = 5 11111111 010 → 6

~a =

1 ( ) 32 bits

000101 → 000010

5 >> 1 - 00101

5 << 2 - 10100

Bit Masking

& with 1 → remains same

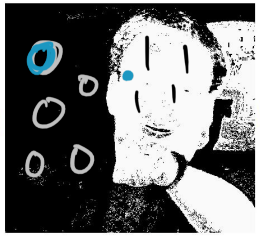
& with 0 → number becomes 0

rgb

white → 1
black → 0

Input

&



Mask

=

Output

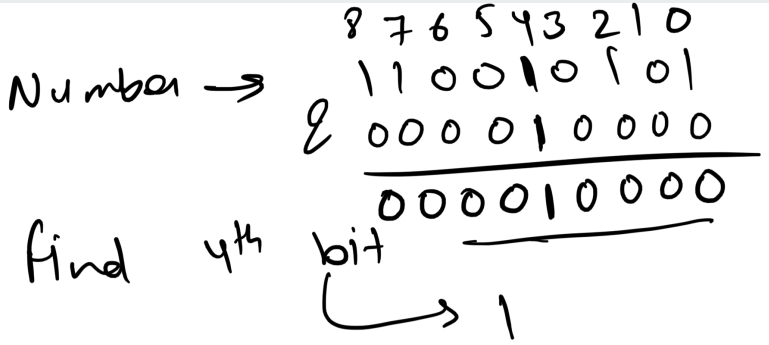
^ with 0 → remains the same

^ with 1 → toggle

1 ^ 1 = 0
0 ^ 1 = 1

1 ^ 0 = 1
0 ^ 0 = 0

Find the ith Bit

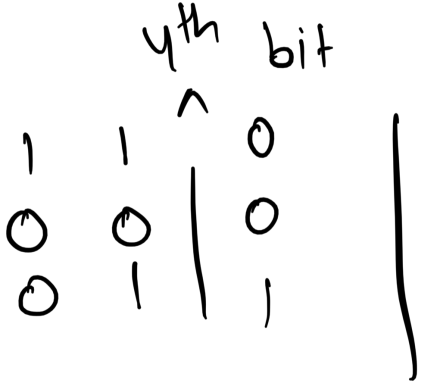


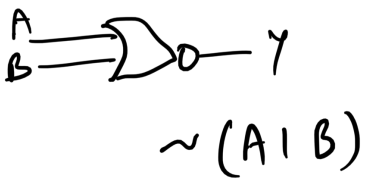

```
if ((n & mask) == 0)
    return 0;
else
    return 1;
```

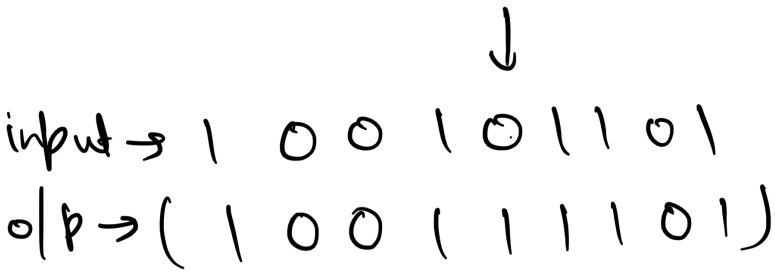
mask = 1 << i

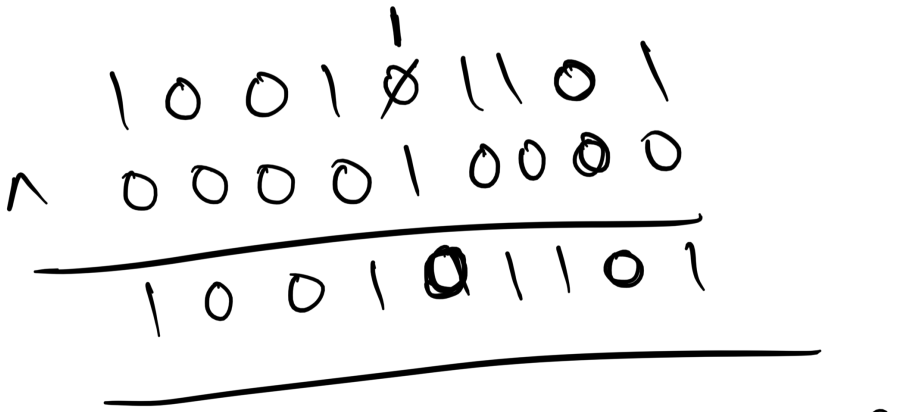
1. n & mask
2. find mask
3. mask 1<<i

Toggle the ith Bit



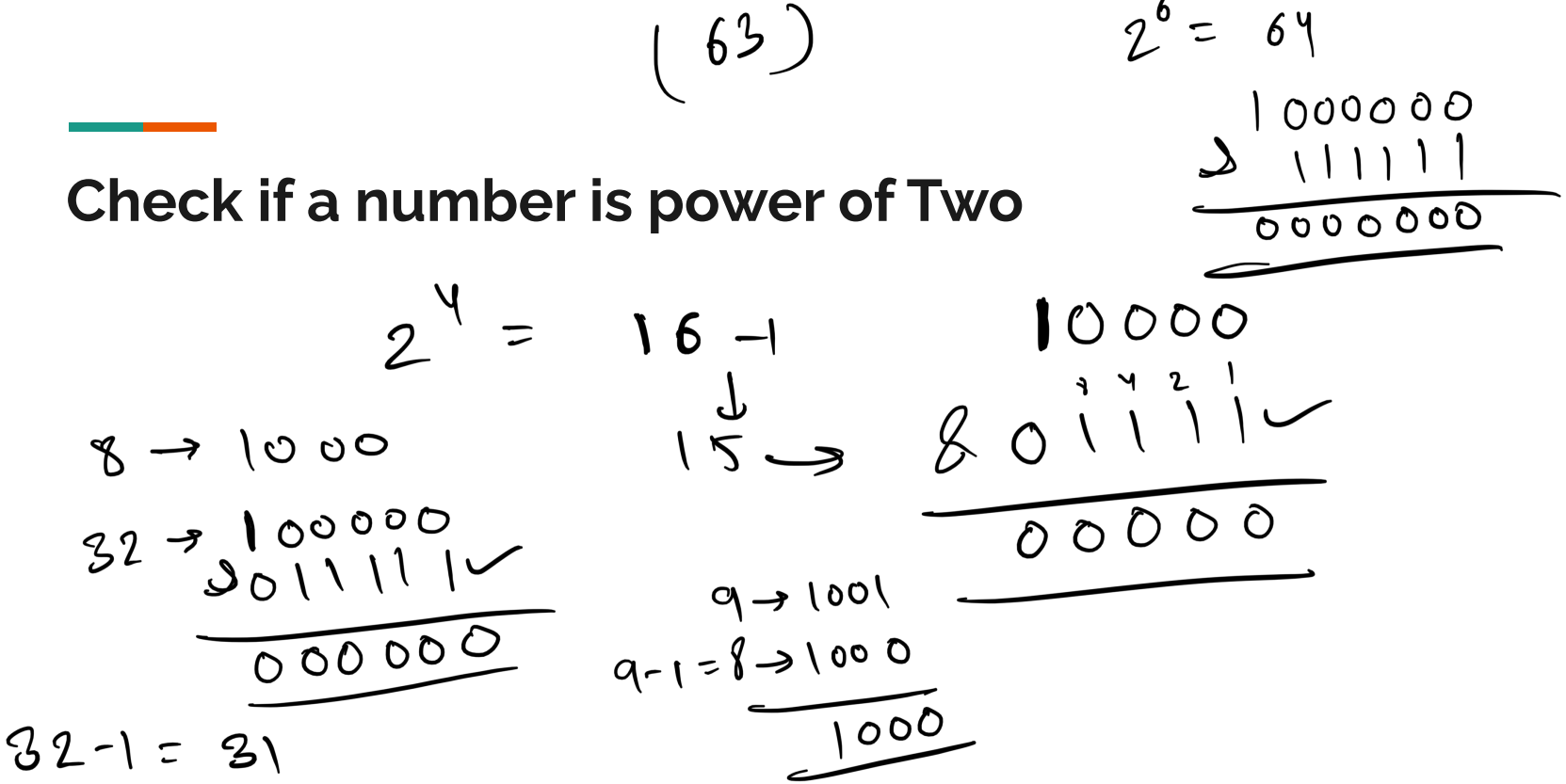




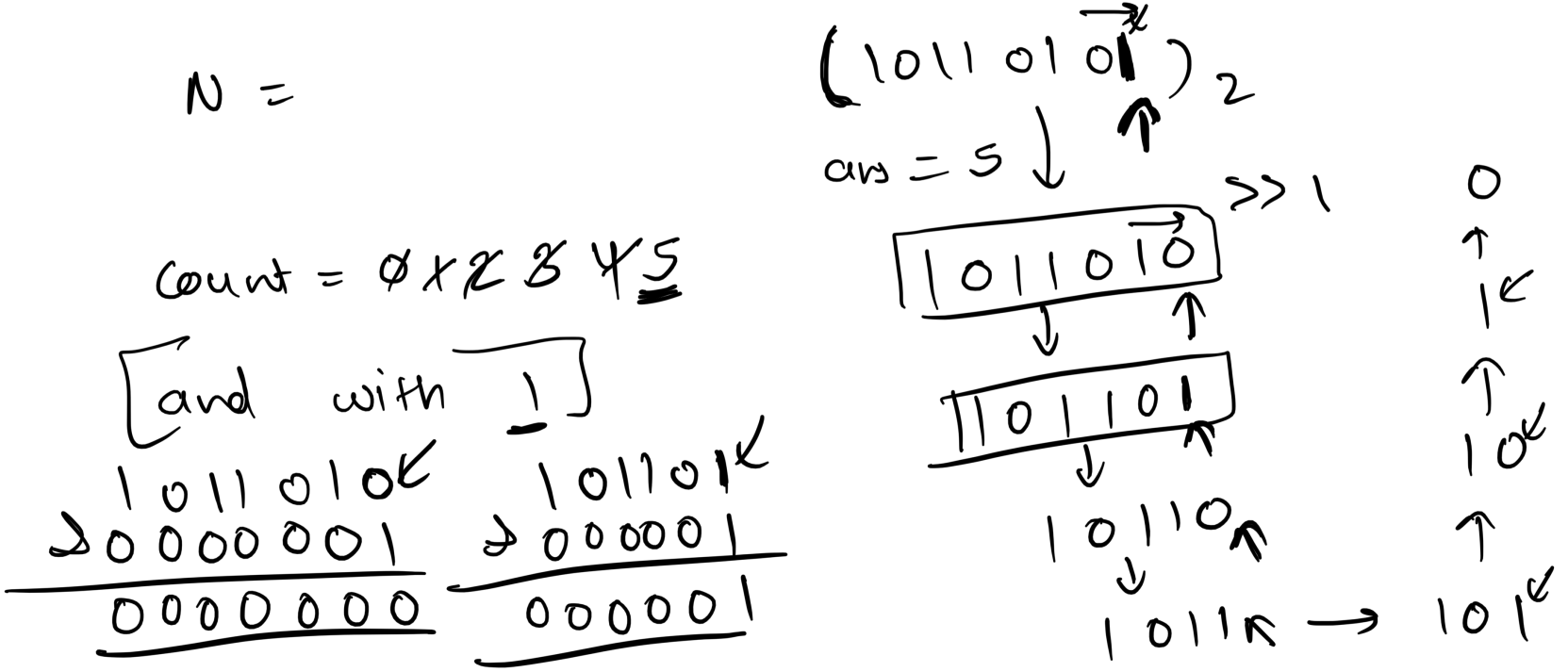


create OR from NAND

Check if a number is power of Two



Count the number of set bits in a Number



T.C.

Numer of digits in a decimal number

$\log_{10} n + 1$

100

n = 145 → 3

1000

n → binary

$\log_2 n + 1$

$\log_{10} 100 = 2 + 1 = 3$

$\log_{10} 145 = 2.\ldots + 1 = 3$

$\log_{10} 1000 = 3 + 1 = 4$

1000 → 4

8 → 1000

13 → (1101)

16 → 10000 → 5

$O(\log n)$

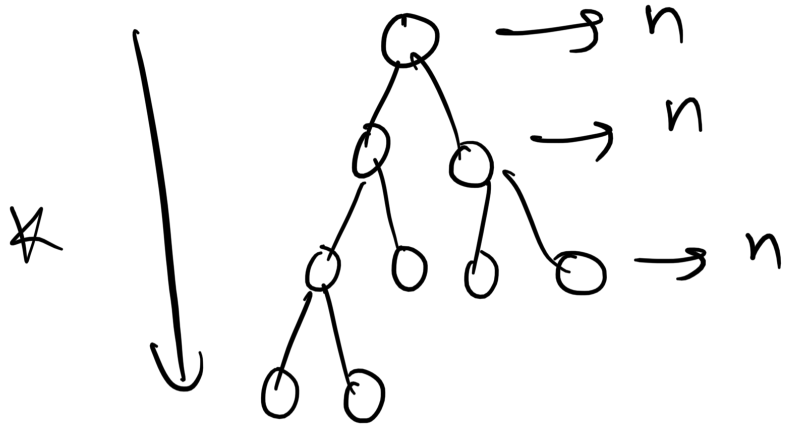
n = 13

while (n > 0)

Optimed
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.



$k = (n \log n)$

Mind Tree → MERN

[Business Analyst] →