Today's Content:

- Number system basics
- Binary to decimal & Viceversa
- Adding 2 Binary numbers
- Bitwise operations
  - Basic properties
  - Basic problems

Satya Sai

2019 CSE grad

2+ years

(Array Interview Problems)

not depend: Bit Man

Extra Session

Number System Basics → Decimal Number System → [0 - 9]
→ [10]

$\underset{10^2}{7}\ \underset{10^1}{3}\ \underset{10^0}{4}$: 700 + 30 + 4

$\underset{10^3}{6}\ \underset{10^2}{5}\ \underset{10^1}{9}\ \underset{10^0}{4}$: 6000 + 500 + 90 + 4

2 4 5 : 200 + 40 + 5

Other number Systems
→ binary → trinary
→ Hexa →
→ Octal →

Number System :

Octal - $(125)_8$ → power [8]
→ every digit [0 - 7]

$\underset{8^2}{\overset{2}{1}}\ \underset{8^1}{\overset{1}{2}}\ \underset{8^0}{\overset{0}{5}}$ = $1 \times 8^2 + 2 \times 8^1 + 5 \times 8^0$ = 85
64 + 16 + 5

Binary
→ every digit [0 - 1]
→ every power [2]

1) $(\overset{4}{1}\,\overset{3}{0}\,\overset{2}{1}\,\overset{1}{1}\,\overset{0}{0})_2$ = $2^4 + 0 + 2^2 + 2^1 + 0$ = 16 + 4 + 2 = 22

2^4 2^3 2^2 2^1 2^0

Binary Number System

{ Binary System → Decimal System }

1) $(1\,0\,1\,0\,0)_2 = 16 + 4 + 0 + 0 = 20$

$2^4 \quad 2^3 \quad 2^2 \quad 2^1 \quad 2^0$

Invalid Binary Representation

2) $\overset{0}{(1}\,\overset{1}{2}\,\overset{2}{0})_2 = 1 \times 2^2 + 2 \times 2^1 + 0 \times 2^0 = 4 + 4 + 0 = 8$

$2^2 \quad 2^1 \quad 2^0$

Decimal to Binary? = { Repeated division } → (? next Session)

2	37	– 1
2	18	– 0
2	9	– 1
2	4	– 0
2	2	– 0
2	1	– 1
	0	

2	25	– 1
2	12	– 0
2	6	– 0
2	3	– 1
2	1	– 1
	0	

$2^4 \; 2^3 \; 2^2 \; 2^1 \; 2^0$

1 1 0 0 1

$= 16 + 8 + 1 = 25$

2	19	– 1
2	9	– 1
2	4	– 0
2	2	– 0
2	1	– 1
	0	

1 0 0 1 1

$2^4 \qquad 2^1 \; 2^0$

$= 16 + 2 + 1 = 19$

5 4 3 2 1 0

$(1\,0\,0\,1\,0\,1)_2$

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

$= 32 + 4 + 1 = 37$

Add 2 Decimal Numbers

$d = s \% 10$, $c = s / 10$

decimal

```
   1   1
   7   8   9
0  1   4   2
-------------
0  9   3   1
```

```
      17/10  8/10  17/10
        1     0     1
11/10   7     8     3     9
  1     3     9     4     8
---------------------------
      11%10 17%10  8%10
  1     1     7     8     7
```

$s = 17$

$d = 17 \% 10$

$c = 17 / 10$

Add 2 Binary Numbers → $d = s \% 2$, $c = s/2$

Eg:

```
       1/2  3/2  2/2  1/2
        0    1    1    0
1/2  1    0    1    1    0   → 22
0    0    0    1    1    1   → 7
     ------------------------
     1%2  1%2  3%2  2%2  1%2
0    1    1    1    0    1       29
```

```
         2/2  1/2  2/2  1/2
          1    0    1    0
3/2    1    1    0    1    1   → 27
1      1    1    0    1    0   → 26
       ------------------------
       3%2  2%2  1%2  2%2  1%2
1      1    0    1    0    1   → 53
32    16    8    4    2    0
```

logical operators → AND, OR

&&, ||

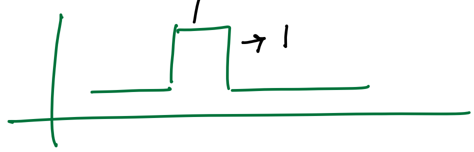
Why binary?

→ ? : ~ : 1

? = ~ : 0

signals

pulse is up



→ 1

int n = 25

In your system stored in binary

$$
\begin{array}{cccccc}
 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 \\
n = & 1 & 1 & 0 & 0 & 1
\end{array}
$$

Binary to decimal & vice versa

print(n) → decimal

xor → Addition without carry

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

$2/2 \to 1$ — don't consider carry

$2\%2 = 0$

later

Bitwise operators : ⇒ { & (And), | (OR), ^ (XOR), ~ (inverse), << (leftsh), >> (rightsh) }

Truth Table?

If both are 1 only 1 (for &)

If one of them is 1 → only 1 (for |)

Same Same puppy shame (for ^)

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

// Basic problems on Bitwise operators

a = 29, b = 19

System will do it bit by bit

	4	3	2	1	0
a :	1	1	1	0	1
b :	1	0	0	1	1
print(a&b) :	1	0	0	0	1
print(a|b) :	1	1	1	1	1
print(a^b) :	0	1	1	1	0

a = 13, b = 10

Try it out?

	3	2	1	0
a :	1	1	0	1
b :	1	0	1	0
a&b :	1	0	0	0
a|b :	1	1	1	1
a^b :	0	1	1	1

// Properties:

	3	2	1	0
a = 10 :	1	0	1	0
1 :	0	0	0	1

print (a&1) : 0 0 0 0 → 0

print (a&1) : 1 0 1 0

	3	2	1	0
a = 11 :	1	0	1	1
1 :	0	0	0	1
	0	0	0	1

→ 1

print (a&1) : 1

a = 14 :	1	1	1	0
1 :	0	0	0	1

print(a&1) : 0 0 0 0 = 0

a = 13 :	1	1	0	1
1 :	0	0	0	1

print (a&1): 0 0 0 1 = 1

// observations

print (10&1) = 0
print (14&1) = 0
print (11&1) = 1
print (13&1) = 1

if (a&1 == 0){
a is even
0th bit is 0
}
if (a&1 == 1){
a is odd
0th bit is 1
}

// given a chena 0th bit is 1 or 0

0th digit in binary decides wether number is odd or even

Coming break

// Why 0^{th} bit decide even or not

7	6	5	4	3	2	1	0
_	_	_	_	_	_	_	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

$1 \times 2^0 = 1$

$0 \times 2^0 = 0$

All these bits are even

0^{th} bit pos

Even + [If 0: Even = Even]

Even + [If 1: Odd = Odd]

Basic Properties

$a \& 1$ → even: 0 ; → odd: 1

$a \& 0 = 0$

$a \& a = a$

$a \mid 1$ = TODO

$a \mid 0 = a$

$a \mid a = a$

$a \wedge 1$ → even: $a+1$; → odd: $a-1$

$a \wedge 0 = a$

$a \wedge a = 0$

$2^0 = 1$

a =	1	0	1	0	0	
1 =	0	0	0	0	1	
$a \wedge 1$ =	1	0	1	0	1	= $a+1$

a =	1	1	0	0	1	
1 =	0	0	0	0	1	
$a \wedge 1$ =	1	1	0	0	0	= $a-1$

Bitwise Properties:

$a \,\&\, b = b \,\&\, a$

$a \,|\, b = b \,|\, a$

$a \wedge b = b \wedge a$

// Commutative property

$(a \,\&\, b \,\&\, c) = (c \,\&\, b \,\&\, a)$

$a \,\&\, (b \,\&\, c)$ — x, y

$(b \,\&\, a \,\&\, c)$

$(b \,\&\, c \,\&\, a)$

$(b \,\&\, c) \,\&\, a \rightarrow (c \,\&\, b \,\&\, a)$ — x, y

same

// $(a \,\&\, b \,\&\, c) = (c \,\&\, b \,\&\, a) = (b \,\&\, a \,\&\, c)$

// $(a \,|\, b \,|\, c) = (c \,|\, b \,|\, a) = (b \,|\, a \,|\, c)$

// $(a \wedge b \wedge c) = (c \wedge b \wedge a) = (b \wedge a \wedge c)$

Associative property

// Q): 5 numbers → value?

0, d, 0/1, $b \wedge d$ — 3✓

$a \wedge b \wedge a \wedge d \wedge b = a \wedge a \wedge b \wedge b \wedge d$

$0 \quad b \quad 0 \quad d$

Q:) 7 numbers

$c \wedge f \wedge a \wedge f \wedge e \wedge g \wedge a = g$

1Q) Given N array elements, every elements repeats twice except 1, find unique element

unique element

ar[5] : { 6 9 6 10 9 } → 10

ar[7] : { 12 9 12 8 7 9 8 } → 7

ar[5] : { 2 9 7 2 7 } → 9

ans : 0 2 11 12 14 9

unique element

```
 2 : 0 0 1 0
 9 : 1 0 0 1
-------------
11 : 1 0 1 1
 7 : 0 1 1 1
-------------
12 : 1 1 0 0
 2 : 0 0 1 0
-------------
14 : 1 1 1 0
 7 : 0 1 1 1
-------------
 9 : 1 0 0 1
```

// idea:1

1) For every element, iterate in array get its frequency if == 1

frequency → occurence

```
i = 0; i < N; i++) {
    j = —; j < —; j++) {
        ____
        ____
        ____
        ____
    }
}
```

TC: $O(N^2)$ SC: $O(1)$

idea2:

1) Taking xor of all elements

```
ans = 0
i = 0; i < N; i++) {
    ans = ans ^ ar[i]
}
return ans;
```

TC: $O(N)$ SC: $O(1)$

// left shift <<

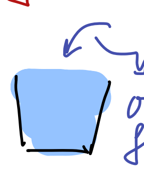
8 bit number — 8 bits

	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	
a = 10 :	0	0	0	0	1	0	1	0	$= 2^3 + 2^1 = 10 = 10 * 2^0$
discard									
a<<1 :	0	0	0	1	0	1	0	0	$= 2^4 + 2^2 = 20 = 10 * 2^1$
discard									
a<<2 :	0	0	1	0	1	0	0	0	$= 2^5 + 2^3 = 40 = 10 * 2^2$
discard									
a<<3 :	0	1	0	1	0	0	0	0	$= 2^6 + 2^4 = 80 = 10 * 2^3$
discard									
a<<4 :	1	0	1	0	0	0	0	0	$= 2^7 + 2^5 = 160 = 10 * 2^4$
discard									
a<<5 :	0	1	0	0	0	0	0	0	$= 2^6 = 64 \neq 10 * 2^5 = 320$

overflow : Exceeding what we can store

↳ we are loosing data

↳ Exceeding your capacity



over flow

// General: no overflow

$a << 1 = a * 2^1$

$a << 2 = a * 2^2$

$a << 3 = a * 2^3$

$\boxed{a << N = a * 2^N}$

$5 << N = 5 * 2^N$

$1 << 1 =$

$1 << 2 =$

$1 << 3 =$

$1 << 4 =$

$\boxed{1 << N = 2^N}$

	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0		decimal
1 :	0	0	0	0	0	0	0	1	=	1
1<<1 :	0	0	0	0	0	0	1	0	=	2^1
1<<2 :	0	0	0	0	0	1	0	0	=	2^2
1<<3 :	0	0	0	0	1	0	0	0	=	2^3

1<<4 $\Rightarrow 2^4$

$\Rightarrow$ If no overflow

$\boxed{1<<n \Rightarrow 2^N}$