Decimal Number System !

$(2437)_{10} \Rightarrow (2 \times 10^3) + (4 \times 10^2) + (3 \times 10^1) + (7 \times 10^0)$

(3 2 1 0)

Base $\Rightarrow$ 10 $\Rightarrow$ 10 unique characters

$\downarrow$

[0 - 9]

A bit of maths.

$2^0, 2^1, 2^2, 2^3, 2^4 \Rightarrow$ GP

Sum of a GP sequence $= \frac{a(r^n - 1)}{r - 1}$

$a =$ first item $\Rightarrow 1$

$r \Rightarrow 2$

$2^0 + 2^1 + 2^2 + 2^3 \ldots + 2^{n-1} =$

Sum $\Rightarrow \frac{1(2^n - 1)}{2 - 1} = 2^n - 1$

Binary Number System.

↳ Unique characters {0 & 1}

↓

Base ⇒ 2

$$(101101)_2$$ (5 4 3 2 1 0)

$$\Rightarrow (1\times2^5)+(1\times2^3)+(1\times2^2)+(1\times2^0)$$

$$\Rightarrow 32+8+4+1$$

$$\Rightarrow (45)_{10}$$

Quiz 1 (10110) (4 3 2 1 0)

$$\Rightarrow (1\times2^4)+(1\times2^2)+(1\times2^1)$$

$$\Rightarrow (22)_{10}$$

Decimal to binary !

Quotient ⇒ D/Q

Remainder ⇒ D%Q

Ex

2 | 19		
2 | 9	1	↑
2 | 4	1	↑
2 | 2	0	↑
2 | 1	0	↑
0	1	↑

$$\Rightarrow (10011)_2$$

Q

2 | 25		
2 | 12	1	
2 | 6	0	
2 | 3	0	
1 | 1	1	
0	1	

$(11001_1)_2$

Add 2 Decimal numbers

```
  0  1  1   ------>  Sum/10
  2  3  4  7
  1  2  5  8
-------------
  3  6  10 15  <------ Sum
-------------
  3  6  0  5   <------ Sum%10
-------------
```

Adding 2 Binary Numbers

```
    1  1     1  1   <------ Sum/2
    0  1  1  0  0  1  1
    0  0  1  0  1  1  0
-------------------------
    1  2  2  1  2  2  1   <------ Sum
-------------------------
    1  0  0  1  0  0  1   <------ Sum%2
-------------------------
```

Negative numbers in binary-

Byte x ⇒ 10 ⇒ 0 0 0 0 1 0 1 0

x ⇒ -10 ⇒ 1 0 0 0 1 0 1 0

y ⇒ 6 ⇒ 0 0 0 0 0 1 1 0

-4 1 0 0 1 0 0 0 0

↳ -16

Issue 1: Arithmetic operations not working

Issue 2: $(00000000)_2 \Rightarrow 0$

$(10000000)_2 \Rightarrow 0$

How to store negative values in binary number system.

$-a \Rightarrow$ 2's complement of a

↓

1's complement + 1

↓

toggle all bits.

$10 \Rightarrow 00001010$

$-10 \Rightarrow$ toggle $(00001010) + 1$

$\Rightarrow 11110101 + 1$

11110110

MSB (most significant bit)

LSB (least significant bit)

↓

$2^7 + 2^6 + 2^5 + 2^4 + 2^2 + 2^1$

↓

$-128 + 118 \Rightarrow -10$

$-10 \Rightarrow 11110110$

$6 \quad\quad 00000110$

11111100

$\Downarrow$

$-2^7 + 2^6 + 2^5 + 2^4 + 2^3 + 2^2$

$-128 + 124$

$\Rightarrow -4$

Is MSB alway negative ??

Signed

MSB $\Rightarrow$ -ve

-2^7

unsigned.

MSB $\Rightarrow$ +ve.

2^7

JAVA / C / C++ $\Rightarrow$ signed / unsigned.

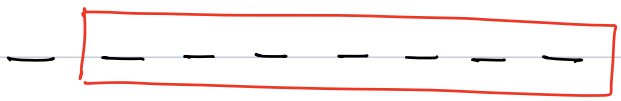
All other languages $\Rightarrow$ By default we have signed.

Ranges

Range for int data type ⇒ $[-2\times10^9, 2\times10^9]$

Byte x ⇒ 8 bits.

⇒ $2^0+2^1+2^2+2^3+2^4+2^5+2^6$



$[-2^7, 2^7-1] \Rightarrow [-128, 127]$

Int x ⇒ 32 bits.

$[-2^{31}, 2^{31}-1]$

$2^{10} \Rightarrow 1024 \simeq 10^3$

$2^{30} \Rightarrow (2^{10})^3 \Rightarrow (10^3)^3 \Rightarrow 10^9$

$2^{31} \Rightarrow (2\times10^9)$

$\Rightarrow [-2\times10^9, 2\times10^9]$

Bitwise Operators

[&, |, ^, ~, <<, >>]

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

a = 29 ⇒ 0 0 0 1 1 1 0 1

b = 18 ⇒ 0 0 0 1 0 0 1 0

a&b ⇒ 0 0 0 1 0 0 0 0

a|b ⇒ 0 0 0 1 1 1 1 1

a^b ⇒ 0 0 0 0 1 1 1 1

~a ⇒ 1 1 1 0 0 0 1 0

$a \& b = b \& a$

$a \mid b \Rightarrow b \mid a$

$a \wedge b \Rightarrow b \wedge a$

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

$A \& 1 \Rightarrow 1$.

$A \Rightarrow$ _ _ _ _ _ _ _ _ → These should be 0 or 1
$\&$ 0 0 0 0 0 0 0 1
next.

0 0 0 0 0 0 0 1

$A \Rightarrow$ odd

$A \& 1 \Rightarrow 0$

$A \Rightarrow$ _ _ _ _ _ _ _ _ $0 \Rightarrow$ even

0 0 0 0 0 0 0 1

0 0 0 0 0 0 0 0

$A \& A \Rightarrow A$ | $A \& 0 \Rightarrow 0$

$A \mid A \Rightarrow A$ | $A \mid 0 \Rightarrow A$

$A \wedge A \Rightarrow 0$ | $A \wedge 0 \Rightarrow A$

$1 \wedge 0 \Rightarrow 1$ | $1 \wedge 1 \Rightarrow 0$

$0 \wedge 0 \Rightarrow 0$ | $0 \wedge 1 \Rightarrow 1$

Break : 10:42

0 0 1 1 1 1 1 1

↳

1 1 0 0 0 0 0 0

Left Shift (<<)

	7	6	5	4	3	2	1	0	
Byte a = 5 =	0	0	0	0	0	1	0	1	
a<<1 =	0	0	0	0	1	0	1	0	= 10 ($a \times 2^1$)
a<<2 =	0	0	0	1	0	1	0	0	= 20 ($a \times 2^2$)
a<<3 =	0	0	1	0	1	0	0	0	= 40 ($a \times 2^3$)
a<<4 =	0	1	0	1	0	0	0	0	= ($a \times 2^4$)
a<<5 =	1	0	1	0	0	0	0	0	= ($a \times 2^5$)
a<<6 =	0	1	0	0	0	0	0	0	= 64

$a << n \Rightarrow a \times 2^n \Rightarrow$ if there is no overflow

$1 << n \Rightarrow 1 \times 2^n$

$1 << n \Rightarrow 2^n$

pow(2,n)

2x2x2x2

$2^4 \Rightarrow 16 \Rightarrow 1<<4$ 00000001

↳ 00010000 $= 2^4$

Right Shift

	7	6	5	4	3	2	1	0	→ loose our lsb
byte a = 50	0	0	1	1	0	0	1	0	
a >> 1	0	0	0	1	1	0	0	1	⇒ 25 = a/2¹
a >> 2	0	0	0	0	1	1	0	0	⇒ 12 = a/2²
a >> 3	0	0	0	0	0	1	1	0	⇒ 6 = a/2³
a >> 4	0	0	0	0	0	0	1	1	⇒ 3
a >> 5	0	0	0	0	0	0	0	1	⇒ 1
a >> 6	0	0	0	0	0	0	0	0	⇒ 0

$$a >> n \Rightarrow \left(\frac{a}{2^n}\right)$$

Q1 For a given decimal number N, check if ith bit is set or Not ?

$0 \le N \le 10^9$

$0 \le i \le 30$

Ex1 $N = 29$ $i = 2$

$29 \Rightarrow$ 1 1 1 0 1 (bit positions 4 3 2 1 0) = true

$N >> i \Rightarrow$ 0 0 1 1 1

$(N >> i) \& 1 \Rightarrow 1$

$\Rightarrow$ ith bit Set

```
bool checkBit ( int N, int i ) {
    if ( (N >> i) & 1 == 1)
        return true;
    else
        return false;
}
```

Tc : O(1)

Sc : O(1)

Q2 Given a Decimal number, count the number of set bits in it.

$0 \leq N \leq 10^9$

Ex1 $N = 29$

4 3 2 1 0

$29 \Rightarrow 1\ 1\ 1\ 0\ 1 \Rightarrow 4$

```
# int countBit(int N) {
    for (int i=0; i<32; i++) {
        if (checkBit(N, i))
```

C++

3

3

TC: O(K)

↓

no of bits

SC: O(1)

7 6 5 4 3 2 1 0

_ _ _ _ _ _ _ _

(N & 1)

(N >> 1)

(N >> 2)

int countBit (int n) {

int c = 0

while (n > 0) {

if (n & 1 == 1)

c++;

n = n >> 1

}

return c

}

TC: O(log n)

SC: O(1)

Q3 Given N array elements, every element repeats twice except 1. Find that unique element.

$1 \leq N \leq 10^5$

Ex1 arr[7] = {3, 7, 5, 5, 1, 3, 7}

⇓

{3 3 5 5 7 7 1}

↓ Xor everything.

{3^3^5^5^7^7^1}

⇓

{0^0^0^1}

⇓

{0^1} ⇒ 1

TC : O(n)

SC : O(1)

Qy Given n array elements, where every element repeats thrice except 1 unique element. Find that unique element.

Ex 1 arr[n] = {1, 2, 1, 1, 3, 3, 3}

Idea 1 : Brute force. ⇒ Check for every element = it's frequency.

TC : $O(n^2)$
SC : $O(1)$.

Idea 2 : Using hashmaps.

TC : $O(n)$
SC : $O(n)$

Idea 3 : Using sorting.

TC : $O(n \log n)$
SC : $O(1)$

Aim: TC : $O(n)$ SC : $O(1)$

arr[13] : 5 7 5 4 7 11 11 9 11 7 5 4 4

	3	2	1	0
5:	0	1	0	1
7:	0	1	1	1
5:	0	1	0	1
4:	0	1	0	0
7:	0	1	1	1
11:	1	0	1	1
11:	1	0	1	1
9:	1	0	0	1
11:	1	0	1	1
7:	0	1	1	1
5:	0	1	0	1
4:	0	1	0	0
4:	0	1	0	0

Unique ⇒ 1 0 0 1

(9)

4 9 6 10

2^0

Pseudo Code

ans = 0

for (int i=0; i<32; i++) {

c = 0

for (int j=0; j<n; j++) {

if (checkBit (arr[j], i))

c++

}

if (c%3 == 1)

ans += (1 << i);

}

return ans;

TC : O(KN)

SC : O(1)

a = 5 , b = 3

$\frac{a}{b}$ ⇒

2:30 → 50%

2:45 → 40%

3 → 10%