Bitwise Operators :-

& | ^ ~ << >>

Operations on bits

exclusive OR ↑

truth table

		AND	OR	XOR	
a	b	&	|	^	~ [not] on 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

byte a :- 28	⇒	0	0	0	1	1	1	0	0		
byte b :- 18	⇒	0	0	0	1	0	0	1	0		
a&b	⇒	0	0	0	1	0	0	0	0	⇒	16
a|b	⇒	0	0	0	1	1	1	1	0	⇒	30.
a^b	⇒	0	0	0	0	1	1	1	0	⇒	14.
~a (1's compliment)	⇒	1	1	1	0	0	0	1	1	⇒	−29
		-2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0		

−128 + 64 + 32 + 2 + 1

Qqq.

int a = 10, print (a & 1)

a = 1 0 1 0

1 = 0 0 0 1

& 0 0 0 0 → 0.

10 & 1 => 0.

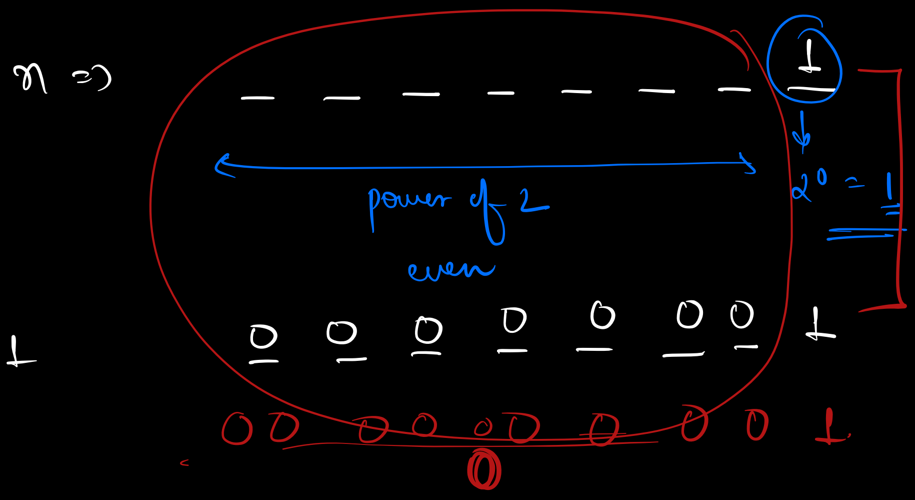
int a = 11, a & 1

a => 1 0 1 1

1 => 0 0 0 1

& => 0 0 0 1

11 & 1 => 1.



odd & 1 => 1.

even & 1 => 0.

if(a&1 ==0)
//a is even

if(a&1 ==1)
//a is odd.

: Properties:

i) a & 0 => 0.

ii) a & a => a

iii) a | 0 => a

iv) a | a => a

v) a ^ 0 => a

vi) a ^ a => 0

vii) a & b = b & a

viii) a | b = b | a

ix) a ^ b = b ^ a

x) a & b & c = b & c & a = c & b & a

xi) a | b | c = b | c | a = c | b | a

xii) a ^ b ^ c = b ^ c ^ a = c ^ b ^ a.

Q.1. $a \wedge b \wedge b \wedge c \wedge a$

$= a \wedge a \wedge b \wedge b \wedge c$

$= 0 \wedge b \wedge b \wedge c$

$= b \wedge b \wedge c$

$= 0 \wedge c$

$= c$ ← $\underline{\underline{\text{ans}}}$.

Q.2 $c \wedge d \wedge a \wedge c \wedge a \wedge g \wedge d$

$= g$

$c \wedge d \wedge a \wedge c \wedge a \wedge g \wedge d$

$\Rightarrow \underbrace{c \wedge c}_{0} \wedge \underbrace{a \wedge a}_{0} \wedge \underbrace{d \wedge d}_{0} \wedge g$

$\Rightarrow \underline{\underline{g}}$.

Q.3. Given N arr elements, every element repeats twice except any one element, find that unique element.

ex ⇒ arr[7] = 3 8 4 8 3 7 4

O/p = 7

$3 \wedge 8 \wedge 4 \wedge 8 \wedge 3 \wedge 7 \wedge 4 \Rightarrow 3 \wedge 3 \wedge 4 \wedge 4 \wedge 8 \wedge 8 \wedge 7$

$\Rightarrow 7$

ex ⇒ arr[5] = 2 2 7 9 7

O/P = 9.

2 ^ 2 ^ 7 ^ 9 ^ 7 ⇒ 2 ^ 2 ^ 7 ^ 7 ^ 9

⇒ 9

sol^n ⇒ XOR all elements

pseudo

```
uniqueElement (arr[]) {
    ans = 0
    for( i=0; i<N; i++) {
        ans = ans ^ arr[i]
    }
    return ans
}
```

TC ⇒ O(N)
SC ⇒ O(1)

dry run ⇒ 2 9 7 2 7

ans = 0 → 2 11 12 14 → 9 ans

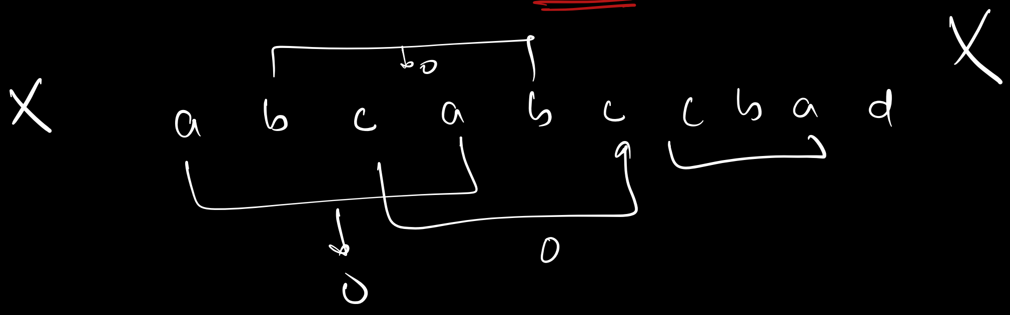
2 ⇒ 0010
9 ⇒ 1001
1011

11 → 1011
7 → 0111
1100

12 → 1100
2 → 0010
1110

14 → 1110
7 → 0111
1001 → 9

Q: will this soln work, if the nos. repeat themselves thrice ⇒ NO



Q: if repetition → 4 times Yes
8 times Yes
6 times Yes.

works if repetition is even times

break → 10PM

: Left Shift :-

$\underbrace{\text{byte } a = 10}_{10}$:- $\boxed{0}$ 0 0 0 1 0 1 0 = 10

discarded

b = $\underbrace{a<<1}_{10 * 2^1}$ => 0 0 0 1 0 1 0 0 = 20

b<<1 => disc 0 0 1 0 1 0 0 0 => 40

or, $\underbrace{a<<2}_{10 * 2^2}$ disc

=> 0 1 0 1 0 0 0 0 => $\underline{80}$

$\underbrace{a<<3}_{10 * 2^3}$

$\underbrace{a<<4}_{10 * 2^4}$ => $\boxed{\underline{160}}$ → overflow

→ a is byte [-128, 127]

a<<3 => 0 1 0 1 0 0 0 0

a<<4 => 1 0 1 0 0 0 0 0 =>

-128 + 32 => $\underline{\underline{-96}}$

int a = ______________

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

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

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

⋮

a<<N => $a * 2^N$ [till no overflow occurs] what ⇒ shifting the bits 1 unit to left

what happens ⇒ $a << 1$ ⇒ $a * 2$

$a << N$ ⇒ $a * 2^N$ [assuming no overflow]

Q4. Find 2^N, given N.

soln 1 ⇒ power func

soln 2 ⇒ loop

soln 3 ⇒ $a << N = a * 2^N$.

$1 << N = 1 * 2^N$

→ return $\underline{1 << N}$.

$$\boxed{\begin{array}{l} TC \Rightarrow O(1) \\ SC \Rightarrow O(1) \end{array}}$$

ex ⇒ 1 is integer ⇒ o/p will be integer

$N = 29$, 2^{29} ✓

$N = 30$; 2^{30} ✓

$\left.\begin{array}{l} N = 31 \\ N > 31 \end{array}\right\}$ $\left.\begin{array}{l} 2^{31} \\ > 2^{31} \end{array}\right]$ ✗✗!

$a << N \Rightarrow a * 2^N$

$2 << N \Rightarrow 2 * 2^N$

$\Rightarrow \underline{\underline{2^{N+1}}}$

$\Rightarrow$ for larger nos. $\Rightarrow$ 1L << N.

$\underline{\underline{2^{62}}}$ long range

$N > 62 \Rightarrow 2^{63}$ [xx].

python → no issues [impl. by strings]

Java → Big Integer → cust impl —

C# → " "

C → move to C++/ Java.

Q Calculate 5^N.

$a << N \to a * 2^N$

$\downarrow$

$5 \longrightarrow \underline{\underline{5 * 2^N}}$

(5<<N) wrong

$5 << (N-1) \to \underline{5 * 2^{N-1}}$

use power, use loop.

$\underline{\underline{6^N}}$

$\downarrow$

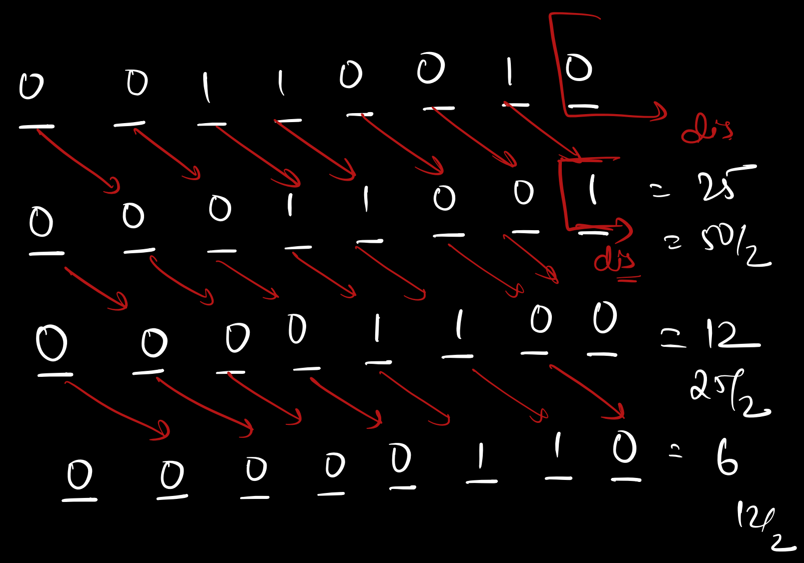
$\cancel{2^N * 3^N}$

$3 * (1 << N)$

$= 3 * 2^N$

$= \underline{\underline{6 * 2^{N-1}}}$

: Right Shift :-



byte a = 50 ⇒ 0 0 1 1 0 0 1 0 → dis

a>>1 ⇒ 0 0 0 1 1 0 0 1 = 25 = $50/2$ → dis

a>>2 ⇒ 0 0 0 0 1 1 0 0 = 12 $25/2$

a>>3 ⇒ 0 0 0 0 0 1 1 0 = 6 $12/2$

$a >> N \Rightarrow a/2^N$

a>>3 = 6

a>>4 = 3

a>>5 = 1

a>>6 = 0

$$\boxed{a >> N = \frac{a}{2^N}}$$

Q 5: Given N and i, check if ith bit in N is set or not?

$N = 26 \Rightarrow \begin{array}{ccccc} 4 & 3 & 2 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 \end{array}$

$i = 2$

return false

cons:

$0 \leq N \leq 10^9$

↓

$\underbrace{\text{int}}$

↓

32 bits

31st bit is unset as N is +ve

↓

$\underbrace{0 \leq i \leq 30}$

$N = 35 \Rightarrow \begin{array}{cccccc} 5 & 4 & 3 & 2 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 \end{array}$

$i = 1$

return true.

$N = \begin{array}{cccccccc} 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \end{array}$

$\underline{\underline{i = 0;}}$

$N \& 1 == 1$ → true

$\underbrace{\text{last bit set}}_{\text{odd}}$

$N \& 1 == 0$ → false

$\underbrace{\text{last bit unset}}_{\text{even}}$

last bit

N even => 0

N odd => 1.

$$N = \begin{array}{cccccccc} 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \end{array}$$

$i = 1$

→ $(N >> 1) \& 1 == 1$ return true

else

return false.

$i = 2$

$(N >> 2) \& 1 == 1$ return true

else

return false.

```
boolean checkBit(N, i){
    if((N>>i) & 1 ==1)
        return true
    else
        return false.
```

$TC \Rightarrow O(1)$

TODO :- Solve this problem using left shift

Q 6. Given N, set ith bit in N:-

if ith bit is already set, don't make any changes.

$$N = 26 \Rightarrow \begin{array}{ccccc} 4 & 3 & 2 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 \end{array}$$

$i = 2$

set i=2

$$\Rightarrow 1\ 1\ 1\ 1\ 0 \Rightarrow \underline{\underline{30}}.\ \text{O/p.}$$

$$N = 35 \Rightarrow \begin{array}{cccccc} 5 & 4 & 3 & 2 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 \end{array}$$

$i = 1$

O/p = 35.

Sol^n

```
setBit( N, i) {
    if( checkBit( N, i) ) {
        return N.
    } else return N + $2^i$.
```

$1 << i$

→ ith is not MSB

$$N = \begin{array}{cccccc} 5 & 4 & 3 & 2 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \end{array} \Rightarrow 37$$

ith $\underline{3}$

$0 \downarrow 1$ 2^3

prev. N = $2^5 + 2^2 + 2^0 \Rightarrow 32 + 4 + 1 = 37$

after set N $\Rightarrow 2^5 + 2^3 + 2^2 + 2^0 \Rightarrow 32 + 8 + 4 + 1 = 45$

N, set ith bit [1]

$\underline{2^i}$

```
setBit( N, i) {
    if( checkBit( N, i) ) {
        return N.
} else return N + (1<<i)
```

TC => O(1)
SC = O(1)

=> approach

1 | 1 = 1

1 | 0 = 1

i = 4.

	7	6	5	4	3	2	1	0	
N =	1	0	0	1	0	1	1	0	
x =>	0	0	0	1	0	0	0	0	(1<<4)
OR,	1	0	0	1	0	1	1	0	→ N

O/P => $\underline{\underline{N | (1<<4)}}$ => $\underline{N}$

$i = 5$

$$\begin{array}{lcccccccccl}
 & 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 & \\
N = & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & \\
x \Rightarrow & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & (1<<5) \\
\hline
\text{OR,} & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & \rightarrow N
\end{array}$$

O/P $\Rightarrow$ $\underline{\underline{N \mid (1<<5)}}$

$$\boxed{\text{return} \quad N \mid (1 << i)}$$

Q 7. Given 0, set x^{th} & y^{th} bit

I/P $\Rightarrow$ $x = 2$, $y = 5$.

$$\begin{array}{lcccccc}
 & 5 & 4 & 3 & 2 & 1 & 0 \\
0 \Rightarrow & 0 & 0 & 0 & 0 & 0 & 0 \\
\hookrightarrow & 1 & 0 & 0 & 1 & 0 & 0
\end{array}$$

$$\begin{array}{lcccccc}
0 \rightarrow & 1 & 0 & 0 & 1 & 0 & 0 \\
 & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\
 & 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0
\end{array}$$

$\longleftrightarrow$

$\underline{\underline{0}}$

$\underline{\underline{1 * 2^5 + 1 * 2^2}}$

ans = $2^x + 2^y$

or, $(1<<x) + (1<<y)$

edge case

$x=2,\ y=2$

$$\begin{array}{ccc} 2 & 1 & 0 \\ 1 & 0 & 0 \end{array}$$

$\downarrow$

4

$(1<<2) + (1<<2)$

$\Rightarrow 2^2 + 2^2$

$\Rightarrow 8$

$$\boxed{\text{return } (1<<x) \mid (1<<y)}$$

Doubts

N → 4 bits ⇒

$$\begin{array}{cccc} 3 & 2 & 1 & 0 \\ \boxed{0} & 0 & 1 & 0 \end{array}$$

$i = 3$ (set)

$\phi * (-2^3) + \underbrace{\phi * 2^2}_{} + 1 * 2^1 + 0 * 2^0$

$$\boxed{\begin{array}{l} N \Rightarrow N + 2^2 \\ \text{or, } N + \textcircled{2^i} \end{array}}$$

$2^3 \mid -2^3$

4 bits $\begin{array}{cccc} 3 & 2 & 1 & 0 \\ 0 & 0 & 1 & 0 \end{array} = \underline{\underline{2}}$

$0 * (-2^3) + 0 * 2^2 + 1 * 2^1 + 0 * 2^0$

4 bits $\begin{array}{cccc} 3 & 2 & 1 & 0 \\ 1 & 0 & 1 & 0 \end{array}$

$-2^3 + 0 + 2^1 + 0$

$= -8 + 2$ $\quad 6$

$= \underline{\underline{-6}} \Rightarrow 2 - 2^3$

$\Rightarrow 2 - 8$

$\Rightarrow \underline{\underline{-6}}$

Ans $\Rightarrow (1 << x) + (1 << y)$

$0 \Rightarrow \begin{array}{cccc} 3 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{array}$

$x = 2$

$y = 2$

$N \rightarrow 0\,1\,0\,0 \Rightarrow \underline{\underline{4}}$

$(1 << 2) + (1 << 2)$

$\Rightarrow 2^2 + 2^2$ $\quad 2$

$\Rightarrow \boxed{8}$