H.W

a, b

$\min((a \wedge x) + (b \wedge x)) = ?$

x = can be any value you want..

x = a & b

ans = a ^ b

0 → 1
1 → 0

0 ^ 0 = 0

if x = 0

^0/1 x

a = 1 0 0 1 0 0 ^(1 0 0 0)

+ b = 1 1 0 0 1 0 ^(1 0 0 0)

0 1 0

a ^ 1

1 ^ 1 = 0

0 ^ 1 = 1

1 ^ 1 = 0
0 ^ 1 = 1
1 ^ 0 = 0
0 ^ 0 = 0

1/0

x = _ _ _ _ _ 0

1

0

sum/2

1 0 0 1
0 1 1 1

0

1 + 1

0 + 1 = 1
^1 ^1
1 + 0

Sum/.2

1 bit

A ^ 0 = A

a → 1 0 | 0 | 1 | 0
+
b → 1 0 | 1 | 0 | 0

^ x = 0

^ x

0

Take XOR with 1

0 ^ 0 + 1 ^ 0 = 0+1 = 1

0 ^ 1 + 1 ^ 1 = 1+0 = 1

= 0

Take XOR with 0

1 ^ 1

1 ^ 0 + 1 ^ 0 = 1 + 1 = 10 = $(2)_{10}$

0 0 → 0
0 1
1 0
1 1 → 1

$x = a \mid b$ or $a \& b$

$a = 1 \qquad b = 0$

$\min(a \wedge x + b \wedge x) = ?$

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

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

MIN $a \wedge (a \& b) + b \wedge (a \& b) = \text{Ans}$

$= a \wedge b$

$0 + 0 = 0$

bits that are ON in both a & b

$a \& b = 0$

$A \wedge 0 = A$

1, 1 → 0

1, 0 → 1

a	a	a	b	b	b
1	0	1	1	1	1
0	0		0	0	0

sum $= a \wedge b$

a | 0 1 0 1

b | 0 0 1 1

$\wedge x$

x | 0 0 0 0

1 1 OR 1

$x = 0/1$

$1 \wedge 0 + 1 \wedge 0$

$= 2$

$1 \wedge 1 + 1 \wedge 1 = (10)_2$

$= 0$

$a \wedge x + b \wedge x = \boxed{0}$

$= (10)_2$

$1 + 0 = 1$ $0 + 1 = 1$

$0 + 0 = 1$ $1 + 0 = 1$

$x = 0$

a	b		x
0	0	→	0
1	1	→	1

$\boxed{\begin{array}{l} 0 \ \underline{OR} \ 0 = 0 \\ 1 \ \underline{OR} \ \phi = 1 \end{array}}$

AND

AND

$x = a$

For any no. of bits

$\boxed{x = a \,\&\, b}$

On

$x = a | b$

Left Shift Operator (<<)

$a = (10)_{10} \rightarrow$ binary $(1010)_2$

8 bit system $\rightarrow$

garbage

7 6 5 4 3 2 1 0

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

$\boxed{a << 1} \rightarrow$ $(0\,0\,0\,1\,0\,1\,0\,0)_2$

16 8 4

$= (20)_{10}$

a << 2

7 6 5 4 3 2 1 0

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

$\hookrightarrow (40)_{10}$

a << 3 = (80)

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

$(15)_{10}$ (15 << 2

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

$(0\,0\,1\,1\,1\,1\,0\,0)_2 = (\quad 60 \quad)_{10}$

32 16 8 4

```
int a = 5;
int b = (a<<2);
cout << b;
```

20

(a << n)

a * power(2, _)

$\boxed{a^b = power(a,b)}$ 2^n

Overflow

$a = (0\,1\,0\,1\,0\,0\,0\,0)_2 = 80$

$\boxed{a << 2}$

$(0\,1\,0\,0\,0\,0\,0\,0)_2 = (64)_{10}$

Right shift Operator (>>)

$a = (10)_{10} = (0\,0\,0\,0\,1\,0\,1\,0)_2$

$a >> 1 = (0\,0\,0\,0\,0\,1\,0\,1)_2 = 5$

$a >> 2 = (0\,0\,0\,0\,0\,0\,1\,0)_2$

2

integer division by 2

$a >> n = \left(\frac{a}{2^n}\right)$. Integer division

$\boxed{cout << (a >> n);}$

$a + b$

$a = 50$ $\qquad a >> 2$

$(00110010) \rightarrow (00001100)_2 \rightarrow 12$

$a >> 2$

$50 / 2^i = \frac{50}{2^2}$

$\frac{50}{4} = 12$

cout << (a << n)

Q Given a number (Decimal Number System)
Return True if its i^{th} bit is ON
False if its i^{th} bit is OFF

$\Rightarrow$ bool isithbitOn(int a, int i);

10, 1 $\qquad$ 00001010

$a >> i$ then check the last bit

sol 2: $a \,\&\, (1 << i)$

sol 1: $(a >> i) \% 2$

7 6 5 4 3 2 1 0

0 1 0 0 1 1 0 0

$a = (\ )_{10}$

$i = 3$

$a >> i$

0 000 1001

last bit is ON

i^{th} bit ON

$a \, \& \, (1 << i)$

0 0 0 0 0 0 0 1

$<< i$

0 0 0 0 1 0 0 0

$(a >> i) \, \& \, 1$

$(a >> i) \, \% \, 2$ ↳ Divisibility test by 2

0 0 0 0 0 0 0 1

if last bit is ON → means the number is ODD

0 1 0 0 1 1 0 0
& 0 0 0 0 1 0 0 0
0 0 0 0 1 0 0 0

NON zero ⇒ It means that i^{th} bit was set.

Break 5 mins

Set a bit

int setBit(int a, int i)

Set bit at i^{th} position

7 6 5 4 3 2 1 0

Example: 0 0 1 0 ~~0~~ 1 1 1 0

$i = 3$

0 0 1 0 1 1 1 0

Ans =

return $\boxed{a \mid (1 \ll i)}$

$a \wedge (1 \ll i)$ ✗ $a \mid (2^{\wedge} i)$

$\boxed{1 \ll i = 2^i}$

$a \ll i = a * 2^i$

$a = 1$

$1 \ll i =$ 0 0 0 0 1 0 0 0

0 1 1 0 1 1 0 0

0 1 1 0 0 1 0 0 ← Turn off the bit

Clear a bit (Unset a bit)

int clearabit (int a, int i)

7 3 2 1 0

0 1 0 1 0 0 1 0

4th bit

A & 0 = 0

&

1 1 1 0 1 1 1 1

0 1 0 0 0 0 1 0

ANS

$a \& (!(1<<i))$

0 0 0 1 0 0 0 0

1 1 1 0 1 1 1 1

Toggle bit

$a \land (1<<i)$

a ~~X~~ 2^i

Q Given a number, Tell if the number has only a single bit switched ON.

OR Find if no. is power of 2 or NOT.

1, 2, 4, 8, 16 . . . 1024, —

0000 00 10   1 0 0 0 0 0 0 0 0 0 0

code

B.F

```
count = 0
for ( i = 0 → 32 )
    if ( a & ( 1 << i ) ) count++
if ( count == 1 ) return true
else return false
```

a & !a = 0

X = - - - - 1 - - -

& 0 1 1 1

0

X = 1000

  1 1 0 0
& 1 0 1 1
  1 0 0 0

0000 1

0111

0 0000

000000

1001100   001000

− 1   − 1

& 1001011   000111

≠ 0

If a number has a single bit turned ON subtracting −1 will toggle the ~~res~~

$\boxed{a \,\&\, (a-1) == 0}$

Q count set bits in a number

1101

$(13)_{10} \rightarrow 3$

H.W

can you solve it O(no. of set bits)

O(32)

no. of bits

You are given an array of N numbers.
All the numbers are in range of
$\boxed{[1 - N+1]}$ and occurring only once.
except one number that is missing
Find out that number

N = 4

arr = [5 4 1 2]

1 → N+1

(1) Sort and check

(2) Hash Map

1 2 4 5

(3) → $\frac{(N+1)(N+2)}{2}$ − actual sum of array

$\cancel{1} + \cancel{2} + 3 + \cancel{4} + \cancel{5} - \cancel{5} - \cancel{4} - \cancel{1} - \cancel{2}$

Long long

n 10 ans = 0

```
for ( i = 1 → n+1 )
    ans ^= i ;
```

```
for ( i = 0 → n )
    ans ^= ( arr[i] ^ (i+1) )

ans ^= n+1
```

```
for ( i = 0 → n )
    ans ^= arr[i]
```

return ans ;

$ans = \cancel{1} \wedge \cancel{2} \wedge 3 \wedge \cancel{4} \wedge \cancel{5}$
$\wedge \cancel{5} \wedge \cancel{4} \wedge \cancel{1} \wedge \cancel{2}$

0000 ×
0001 → 4th
0010 → 3rd
0011 → 4,3

Reverse Bits

3 → 0 0 0 1 1

unsigned int ans = 0

1 1 0 0 0

```
for (i=0 → 32)
    if (a & (1<<i))
        ans |= (1<<(31-i))
```

$ans += 2^{(31-i)}$

1 1 0 0 ...

31 30 29

a = 1 0 0

b = 1 1

i = a.size();

j = b.size()

i--

j--

String 1

1+1

ans = (a+b) % 2 = ans

carry = (a+b) / 2

7 6 5 4 3 2 1 0

$-$ 2^6 2^5 2^4 2^3 2^2 2^1 2^0

-2^7

$0 \rightarrow \boxed{2^8 - 1}$

0 1 1 1 1 1

$2^0 + 2^1 + \dots 2^6$

$= \boxed{2^7 - 1}$

1 0 0 0 0 0 0 0 $= -2^7$

$-1 = ?$

1 0 0 0 0 0 0 1 $= ?$

$-2^7 +$ 2^0

$-2^7 + 2^0 = \times$ -1

4 3 2 1 0

1 0 0 1 0 0 0 1

$-2^7 + 2^4 + 2^0$

$-128 + 16 + 1$

$= (-111)_{10}$

$\boxed{1\ 1\ 1\ 1\ 1\ 1\ 1\ 1}$

$\cancel{-2^7} + (2^6 + \dots 2^0) = \cancel{-2^7}\ -1$

2^7

2^s complement

1 bit sign | 7 bit magn

1 1 1 1 1 1 1 1

-2^7 -2^7+1

- Take a no
- invert all bits
- and add 1

-10

0 0 0 0 1 0 1 0

1 1 1 1 0 1 0 1

\+ 1

1 1 1 1 0 1 1 0

$-128 + 64 + 32 + 16 + 4 + 2$

80   38

$118 - 128$

$= -10$

10 = 0000 1010

& 0000 1000

0000 1000 = 8

= (1<<i) or 0