Bit Manipulation 1

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

Properties

1. $A \mid 0 = A$

$A = 1010$
0000
1010

2. $A \mid A = A$ $\quad 1 \mid 1 = 1 \quad 0 \mid 0 = 0$

3. $A \wedge 0 = A$ $\quad 1 \wedge 0 = 1 \quad 0 \wedge 0 = 0$

4. $A \wedge A = 0$ $\quad 1 \wedge 1 = 0 \quad 0 \wedge 0 = 0$

5. A & 0 = 0

6. A & A = A

0 & 0 = 0 1 & 1 = 1

7. Odd / Even

4 → 1 0 0

6 → 1 1 0

8 → 1 0 0 0

10 → 1 0 1 0

Even → 0

5 → 1 0 1

7 → 1 1 1

9 → 1 0 0 1

11 → 1 0 1 1

Odd → 1

last bit determines even / odd.

how to check last bit ?

A & 1 → 0 even no.

→ 1 odd no.

A = 5

```
  1 0 1
& 0 0 1
-------
  0 0 1
-------
```

A = 10

```
  1 0 1 0 0
& 0 0 0 0 1
-----------
  0 0 0 0 0
-----------
```

8. Commutative Property

$A \& B = B \& A$

$A \mid B = B \mid A$

$A \wedge B = B \wedge A$

9. Associative Property

$(A \& B) \& C = A \& (B \& C)$

$(A \mid B) \mid C = A \mid (B \mid C)$

$(A \wedge B) \wedge C = A \wedge (B \wedge C)$

Quiz

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

$= (a \wedge a) \wedge (b \wedge b) \wedge d$

$= (0 \wedge 0) \wedge d$

$= 0 \wedge d = d$

Quiz

$1 \wedge 3 \wedge 5 \wedge 3 \wedge 2 \wedge 1 \wedge 5$

$\Rightarrow (1 \wedge 1) \wedge (3 \wedge 3) \wedge (5 \wedge 5) \wedge 2$

$= 0 \wedge 0 \wedge 0 \wedge 2 = 2$

Question

Given an integer array where every no. occurs twice except one number. find that unique no.?

Soln : ans = $\wedge_i$, a[i]

```
ans = a[0]
for(i=1 to n-1){
    ans ^= a[i]
}
return ans
```

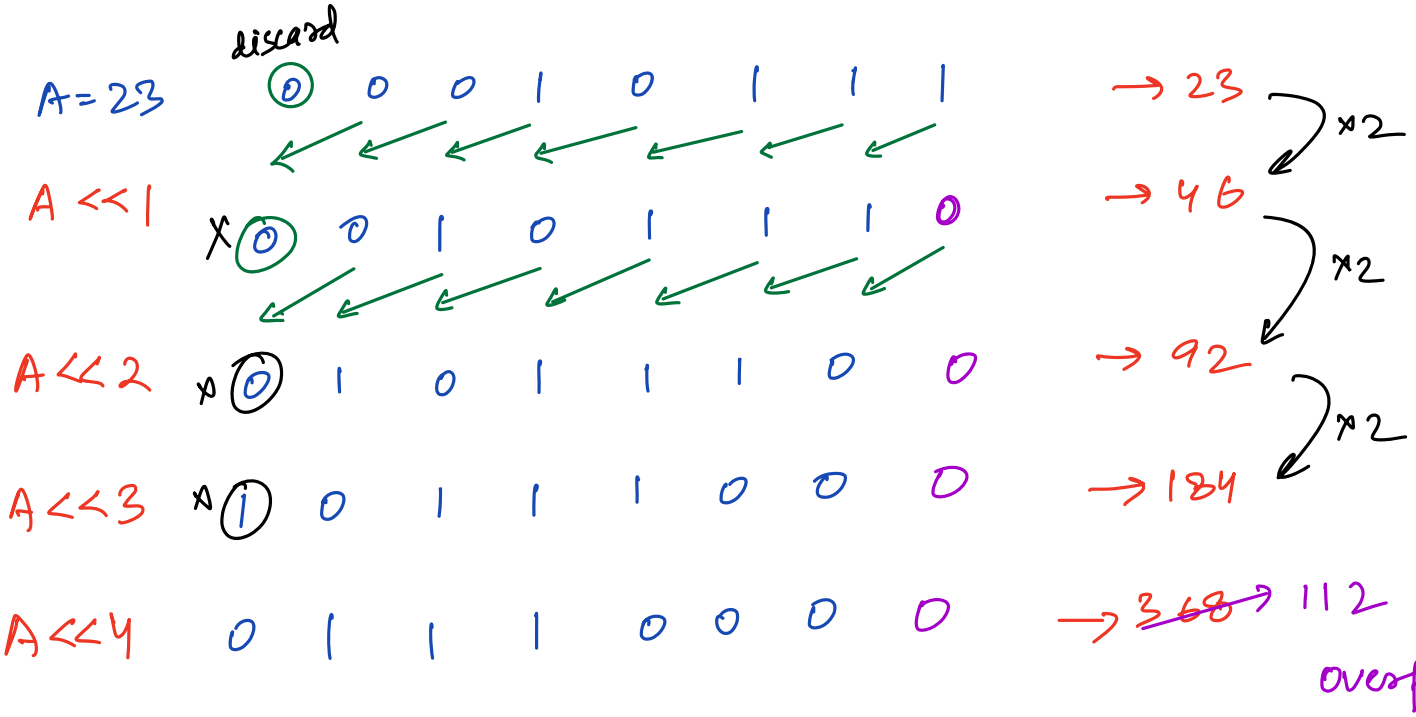
TC = O(N)

SC = O(1)

Left Shift operator (<<)

for explaining ⟶ 8 bit numbers

↳ 0 - 255



N << 1	=	N * 2
N << K	=	N * 2^K

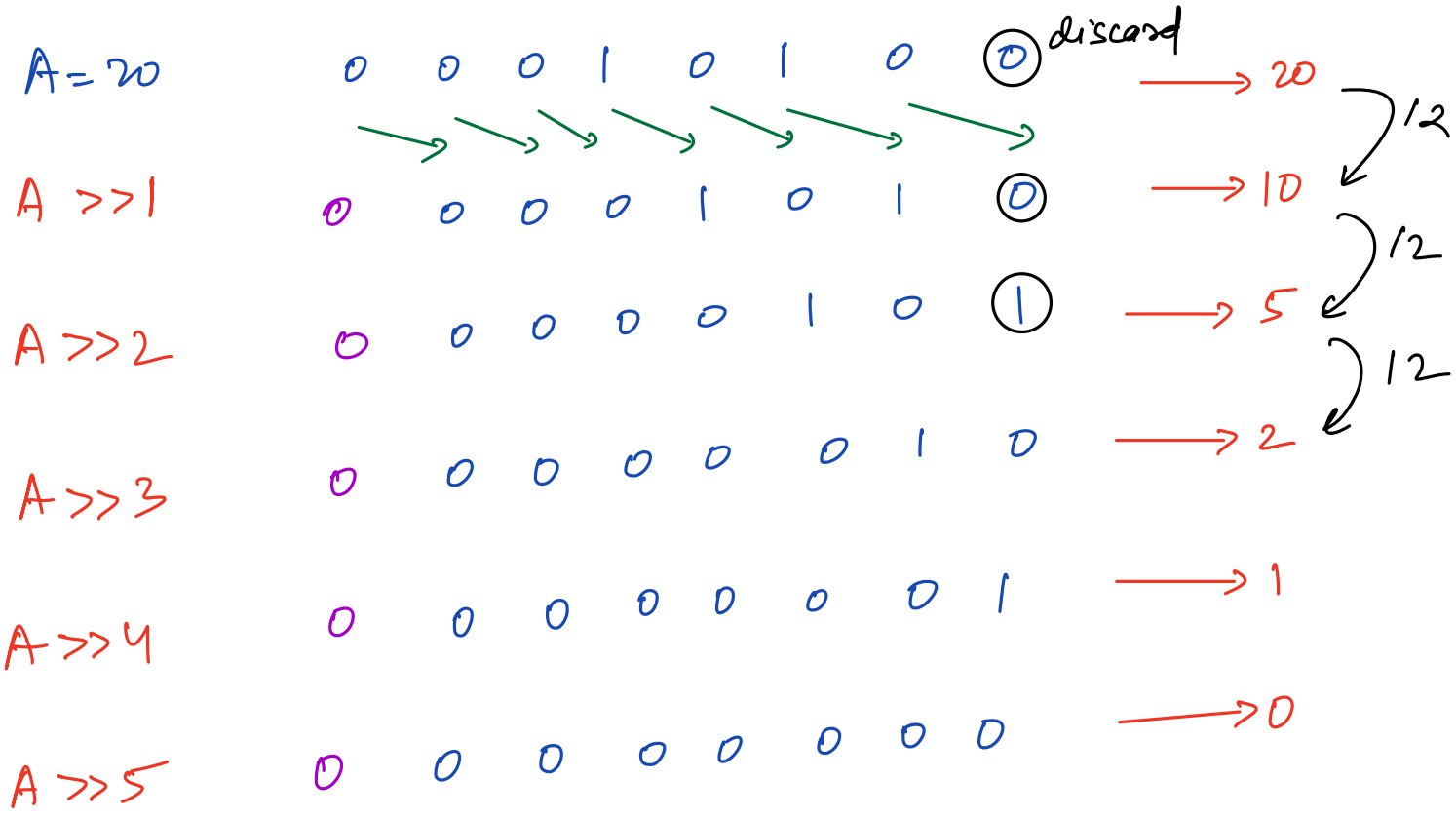
1 << n = 2^n

take care of overflow

x * 2 > INT_MAX ✗

$x > \frac{INT\_MAX}{2}$ ✓ overflow condition

Right Shift (>>)



$N >> 1 = N/2$
$N >> K = N/2^K$

no overflow

Quiz

$1 << 3$

$1 << n = 2^n$

$\Rightarrow 2^3 = 8$

Power of left shift operator

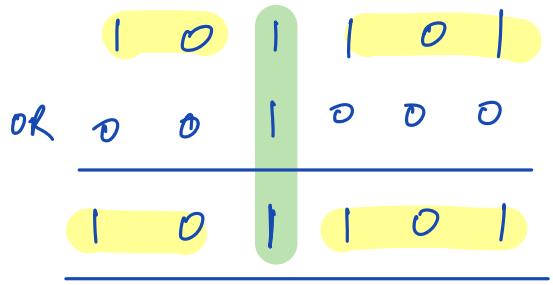
$(1 << K) = 2^K$ → 0 00...0 1 0 00...0 (only K^{th} bit is set)

(K^{th} bit)

1. OR

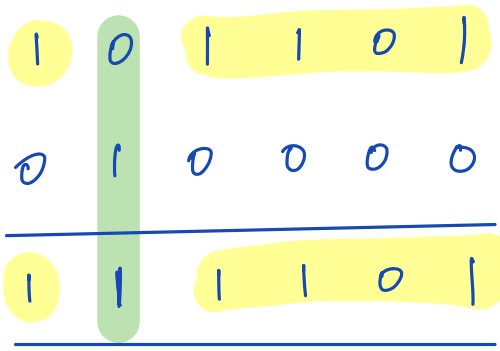
$N \mid (1 << K)$ ⇒ set K^{th} bit of N

N = 45
K = 3



= 45 if K^{th} bit is already 1 ⇒ no change

N = 45
K = 4



= 61

else → $N + 2^K$

2. XOR

$N \wedge (1 << K)$ → toggle K^{th} bit

N = 45
K = 3

XOR

```
  1 0 1 1 0 1
  0 0 1 0 0 0
-------------
  1 0 0 1 0 1
-------------
```

= 37

(if bit is 1 = $N - 2^K$)

N = 45
K = 4

```
  1 0 1 1 0 1
  0 1 0 0 0 0
-------------
  1 1 1 1 0 1
-------------
```

= 61

else = $N + 2^K$

3. AND

N & (1<<K)

→ 2^K (K^{th} bit is set)

→ 0 (K^{th} bit is unset)

N = 45
K = 3

AND

```
  1 0 1 1 0 1
  0 0 1 0 0 0
-------------
  0 0 1 0 0 0
-------------
```

= 2^3 (2^K)

if K^{th} bit is 1 ⇒ 2^K

N = 45
K = 4

```
  1 0 1 1 0 1
  0 1 0 0 0 0
-------------
  0 0 0 0 0 0
-------------
```

= 0

else ⇒ 0

Unset K^{th} bit

```
def unset(N, K) {
    X = N & (1<<K)
    if(X > 0) {
        N = N ^ (1<<K)
    }
}
```

if X = 0, K^{th} bit is already unset

TC = O(1)

SC = O(1)

Check if K^{th} bit is set or unset ?

```
def isset(N, K) {
    X = N & (1<<K)
    if (X == 0) return false
    return true
}
```

Question 1

Given an integer N, count total set bits in N.

32 bit integer.

N = 12

1 1 0 0

ans = 2

```
ans = 0
for ( i = 0 to 31 ) {
    if ( N & (1<<i)  > 0 ) {
        ans ++
    }
}
return ans
```

TC = O(1)

SC = O(1)

Another approach

```
aus = 0
while ( N > 0 ) {
    if ( N&1 ) {
        ans ++
    }
```

TC = O(logN)

SC = O(1)

```
        N = N >>1    // N = N/2
    }
    return ans
```

Question

For each train, there is a special no.

Special no.: 28 bit no. where if i^{th} bit is set then train runs on that day

find train which runs the most!

A = [20 , 7 , 10]
    0   1   2

20 → 1 0 1 0 0

7 → 1 1 1

10 → 1 0 1 0

ans = 1

code

```
def findTrain(a[], N){
    curr_count = -1
    ans = -1
    for(i=0 to n-1){
        c = countBits(a[i])   // return total set bits in a[i]
        if(c > curr_count){
            curr_count = c
            ans = i
        }
    }
    return ans
}
```

TC = O(N)

SC = O(1)

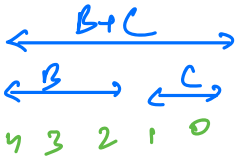
Question

Create a binary number with specific pattern.

The pattern is : A 0's followed by B 1's followed by C 0's.

A, B, C is input $0 <= A, B, C <= 20$

Return the integer.



eg A = 4
B = 3
C = 2

0 0 0 0 1 1 1 0 0 ⇒ 28

A 0's B 1's C 0's

1. Ignore first A 0's.

Set bits from C to B+C-1

```
long ans = 0
for( i = C to B+C-1 ){
    ans = ans | (1<<i)
}
return ans
```

TC = O(B)

SC = O(1)

max value of ans = 1 1 1 1 1 0 0 ... 0

20 1's 20 0's

⇒ ~40 bit no.

⇒ use long

$2^3 - 1 = 7 = 111$

$2^4 - 1 = 15 = 1111$

$2^B - 1 = \underbrace{1111\ldots1}_{B \text{ times}}$

$(2^B - 1) << C \Rightarrow \underbrace{11111}_{B \text{ times}}\underbrace{000\ldots}_{C \text{ times}}$

$((1<<B) - 1) << C$

$am = ((1L << B) - 1) << C$

OR

long am = 1

$am = ((am << B) - 1) << C$