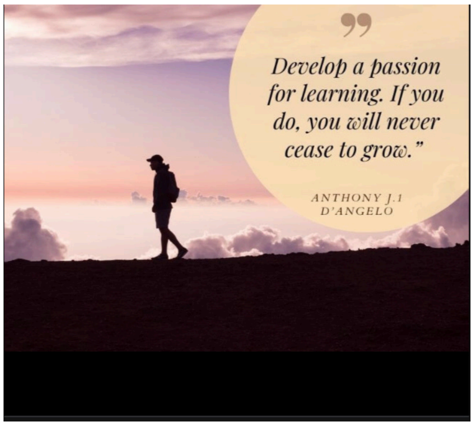
Today's Quote



Today's content:

- power of left shift
- set / unset i^{th} bit
- check i^{th} bit is set or unset
- Q.1 Count no. of set-bits in N
- Negative no's
- Range of Integers / Long
- Importance of constraints
- Binary no. subtraction.

$1 \ll 0 = 1 \quad [2^0]$

$1 \ll 1 = 2 \quad [2^1]$

$1 \ll 2 = 4 \quad [2^2]$

$1 \ll 3 = 8 \quad [2^3]$

$1 \ll 4 = 16 \quad [2^4]$

$1 \ll i = \quad [2^i]$

$N \ll 0 = N$

2	45	1
2	22	0
2	11	1
2	5	1
2	2	0
2	1	1
	0	

Power of left shift

$(45)_{10} = (101101)_2$

Case-1

OR

	5	4	3	2	1	0	
45 →	1	0	1	1	0	1	
(1<<2) →	0	0	0	1	0	0	
	1	0	1	1	0	1	= (45)

Case-2.

	5	4	3	2	1	0
45 →	1	0	1	1	0	1
(1<<4) →	0	1	0	0	0	0
	1	1	1	1	0	1

→ (61)

$\boxed{N \mid (1 \ll i)}$ → Setting i^{th} bit

[set / on → 1
unset / off → 0]

XOR

45 →	1	0	1	1	0	1	
(1<<2) →	0	0	0	1	0	0	
	1	0	1	0	0	1	[<N]

45 →	1	0	1	1	0	1	
(1<<4) →	0	1	0	0	0	0	
	1	1	1	1	0	1	[>N]

$\boxed{N \wedge (1 \ll i)}$ → Toggling i^{th} bit

AND

	5	4	3	2	1	0
45 →	1	0	1	1	0	1
(1<<2) →	0	0	0	1	0	0
	0	0	0	1	0	0

	5	4	3	2	1	0
45 →	1	0	1	1	0	1
(1<<4) →	0	1	0	0	0	0
	0	0	0	0	0	0 → 0

$\boxed{N \,\&\, (1 << i)}$ →
- 0 [ith-bit is unset]
- (1<<i) [ith-bit is set]

Check whether ith-bit is set/unset.

Q: Unset ith-bit of the given number if it is set, otherwise no change.

N=45, i=2.   1 0 1 1 0 1 (bits 5 4 3 2 1 0) ⟹ 1 0 1 0 0 1 [41]

N=45, i=4   1 0 1 1 0 1 (bits 5 4 3 2 1 0) ⟹ 1 0 1 1 0 1 [45]

```
if (checkBit(N, i) is set) {
    N = N ^ (1 << i)
}
else {
    // do nothing.
}
```

How to check ith-bit is set ?

① $N \& (1<<i) == (1<<i)$

② $N \mid (1<<i) == N$

③ $N \wedge (1<<i) < N$

$N \wedge (1<<i)$ → $<N$ [ith-bit is set] ; → $>N$ [ith-bit is unset]

$N\&1$ → 0 [0th-bit is 0] [even no.] ; → 1 [0th-bit is 1] [odd no.]

(i=2) $N = 45$.

$(N>>i)$

$$\begin{array}{cccccc} 5 & 4 & 3 & 2 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{array}$$

④ $(N>>i) \& 1 == 1$

$N=10$ ⇒
$$\begin{array}{cccc} 3 & 2 & 1 & 0 \\ 1 & 0 & 1 & 0 \end{array}$$

$i=3$.

$N>>3$ ⇒ $0\,0\,0\,1$

$\& \; 1$

$0\,0\,0\,1$

Q) Count the no. of set-bits in N.

Eg: N=45.

$\begin{array}{cccccc} 5 & 4 & 3 & 2 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 \end{array}$ [ans → 4]

N=35 1 0 0 0 1 1 [ans → 3]

idea: [check for every bit whether it is set or unset]

int → 4 Bytes → 32 bits.

pseudo-code.

```
count = 0
for( i = 0 ; i < 32 ; i++) {
    if (checkBit(N, i) is set) {
        count ++
    }
}
return count
```

int.	long.	no. of bits.
0→32	0→64	= 100.
$\approx 2^{31}$	$\approx 2^{63}$	$\approx 2^{100}$.

$\boxed{\begin{array}{l} T.C \rightarrow O(\log_2 N) \\ S.C \rightarrow O(1) \end{array}}$

$\begin{array}{ccccccccccc} 31 & 30 & 29 & 28 & 27 & & & 3 & 2 & 1 & 0 \\ | & | & | & | & | & \ldots & & | & | & | & | \end{array}$

2^{nd} Approach.

	3 2 1 0	
N = 10	1 0 1 0	count = 0
N >> 1	0 1 0 1	count = 1
N >> 2	0 0 1 0	count = 1
N >> 3	0 0 0 1	count = 2
N >> 4	0 0 0 0	{return count}.

pseudo-code.

```
count = 0
while ( N > 0) {
    if ( N & 1 == 1){
        count ++
    }
    N = N >>1        [ N = N/2 ]
}
return count ;
```

$\approx 2^{100} \Rightarrow 100$

$\approx 2^{64} \Rightarrow 64$

$\approx 2^{32} \Rightarrow 32$

$\boxed{\begin{array}{l} T.C \rightarrow O(\log_2 N) \\ S.C \rightarrow O(1) \end{array}}$

Negative Numbers

$(-45)_{10} = (\quad\quad\quad)_2$

{int → 8 bits} explaination

	7	6	5	4	3	2	1	0	
45 →	0	0	1	0	1	1	0	1	
toggle all bits	1	1	0	1	0	0	1	0	[1's compliment]
								+1	
-45 ⇒	1	1	0	1	0	0	1	1	[2's compliment]

Most Significant Bit [M.S.B]

Integer.

$\underset{31}{\boxed{\_}}\ \underset{30}{\_}\ \underset{29}{\_}\ \underset{28}{\_}\ \cdots\ \underset{3}{\_}\ \underset{2}{\_}\ \underset{1}{\_}\ \underset{0}{\_}$

$[2^{31}] \quad [\underline{2^{30} + 2^{29} + 2^{28} + \cdots + 2^{3} + 2^{2} + 2^{1} + 2^{0}}]$

$$2^{31} > \frac{1[2^{31}-1]}{[2-1]} = 2^{31}-1.$$

M.S.B → 0 → +ve

M.S.B → 1 → -ve

M.S.B ⇓ signed bit

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

$$(-2^{7}) + 2^{6} + 2^{4} + 2^{1} + 2^{0} = -128 + (64 + 16 + 2 + 1)$$

$$= -128 + 83$$

$$= -\underline{45}$$

$(-12)_{10} = (\qquad\qquad)_2$

$$\begin{array}{lcccccccccl}
 & 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 & & \\
12 \rightarrow & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & & \\
\text{toggle all bits} \rightarrow & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & & \rightarrow \text{1's compliment} \\
 & & & & & & & + & 1 & & \rightarrow \text{2's compliment} \\
\hline
 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & &
\end{array}$$

$(-2^7) + 2^6 + 2^5 + 2^4 + 2^2 = -128 + 64 + 32 + 16 + 4$

$= -128 + 116 = -12$

minimum number $\quad \overset{7}{1}\,\overset{6}{0}\,\overset{5}{0}\,\overset{4}{0}\,\overset{3}{0}\,\overset{2}{0}\,\overset{1}{0}\,\overset{0}{0} = -128 = -2^7$

maximum number $\quad 0\,1\,1\,1\,1\,1\,1\,1 = 127 = 2^7 - 1$

Range of Integer $\rightarrow [-2^{31}, 2^{31}-1]$

Minimum integer $\quad \underset{31}{1}\;\underset{30}{0}\;\underset{29}{0}\;\underset{28}{0}\;0 \;-\,-\,-\,-\; \underset{2}{0}\;\underset{1}{0}\;\underset{0}{0} = -2^{31}$

$= -2147483648 \approx -2 \times 10^9$

Maximum integer $\quad \underset{31}{0}\;\underset{30}{1}\;\underset{29}{1}\;\underset{28}{1}\;1 \;-\,-\,-\; \underset{2}{1}\;\underset{1}{1}\;\underset{0}{1} = 2^{31} - 1$

$= 2147483647 \approx 2 \times 10^9$

$= 2.147483647 \times 10^9$

Range of Long $[-2^{63}, 2^{63}-1]$

Minimum $\rightarrow$ $-2^{63} \approx -9 \times 10^{18}$

Maximum $\rightarrow$ $2^{63}-1 \approx 9 \times 10^{18}$

Q. Calculate sum of all array elements.

```
long ~~int~~ sum = 0
for( i = 0 ; i < N ; i++){
        sum += arr[i]
}
return sum
```

constraints.

$1 \le N \le 10^5$

$1 \le arr[i] \le 10^6$

$A = [10^6 \; 10^6 \; 10^6 \text{ — } 10^6]$

$N = 10^5$

$sum = 10^6 * 10^5 = 10^{11}$

$\therefore$ overflow condition.

constraints $\rightarrow$ T.L.E, overflow

Q.) Given two +ve integers. [a and b]. Return $\underline{a * b}$

$a <= 2 \times 10^9$
$b <= 2 \times 10^9$

$a * b = 2 \times 10^9 \times 2 \times 10^9$
$= \underline{\underline{4 \times 10^{18}}}.$

```
int ans = a * b          [x]
return ans
```

⇓

```
long ans = a * b         [x]
return ans
```

Overflow will happen at multiplication step.

⇓

```
long ans = long(a * b)   [x]
return ans
```

⇓

```
long ans = long(a) * b   [✓]
return ans
```

long * int = long

⇓

```
long ans = a
     ans *= b;           [✓]
return ans
```

Subtraction of Binary No's.

$1+1+1 = 11.$

$45 - 12$

$\Rightarrow 45 + (-12)$

$$
\begin{array}{r|cccccccc}
 & 1. & 1 & 1 & 1 & 1 & 1 & & \\
45 \rightarrow & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\
-12 \rightarrow & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\
\hline
 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
\hline
 & 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0
\end{array}
$$

$2^5 + 2^0 = 32+1 = \underline{33}.$

Doubts

int → 32-bit (4 Bytes).

$[+ve, -ve, 0] \rightarrow [+, -].$

[1's compliment → toggle all the bits.
2's compliment → 1's compliment + 1.]

{ -ve → 2's compliment. }

integer

no. is -ve.

$\boxed{1}$ 1 0 0 0 0 0 ——— 0 1 0 1 1 0

31 30 29 28 ... 5 4 3 2 1 0

8 → 1 0 0 0

3 2 1 0

B = C + D (split)

P ^ Q = R (merge)

Is it possible to convert A[] to size 1.

[Single element == 0].

arr → [9 , 17]

⇒ [9^17] ⇒ [24]

arr → [5 7 13 12 9]

0 1 2 3 4

[4 1 6 1 12 1 12 8 1]

→ count of odd numbers.

(count % 2 == 0) Yes.

else No.

1 0 1 0

[8-bit]

int → 32 bits.

long → 64 bits

m.s.B / sign-bit

$\boxed{1}\,\underline{0001010}$

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

$2^7 > 2^6 + 2^5 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0$

[just for explaining.]

[+ve, -ve, 0].

arr →

0	1	2
5	7	9

K = n(n+1)/2 ;

int[][] ans = new int[K][] ;

```
int t = 0 ;
for( i = 0 ; i < N ; i++) {
    for( j = i ; j < N ; j++) {
        int[] a = new int[j-i+1] ;
        for( K = i , l = 0 ; K <= j ; K++, l++) {
            a[l] = arr[K];
        }
        ans[t] = a ;
    }
    t++;
}
```

[1, 2, 3]

ArrayList <ArrayList <Integer>>

⇓

[[1], [1,2], [1,2,3], [2], [2,3], [3]]

```
ArrayList < Integer > al = new ArrayList<> ();
```

arr →

1	0	0	1	0	1	0	1	1	1	0	0
0	1	2	3	4	5	6	7	8	9	10	11

$(2*B+1)$, $B = 2.$

[length = $(2*2)+1 = 5.$]

alternating → 0, 1 or 1, 0.

idea → check every subarray of length 2B+1. [alternating or not]

[Append this in your ans. $(i+B)$]

start idx of subarray.

XOR Sum.

$(A \wedge X) + (B \wedge X)$ minimum value.

$$
\begin{array}{lccccccc}
 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \\
A \to & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\
X \to & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\
\hline
 & 0 & 0 & 1 & 0 & 0 & 1 & 0
\end{array}
\quad \wedge \quad + \quad
\begin{array}{lccccccc}
 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \\
B \to & 1 & 1 & 0 & 1 & 1 & 0 & 0 \\
X \to & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\
\hline
 & 0 & 1 & 0 & 0 & 1 & 0 & 0
\end{array}
\quad \wedge
$$

$$\Downarrow$$

$(2^4 + 2^1 + 2^5 + 2^2) \Rightarrow$ final val.

$$
\begin{array}{lccccccc}
 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \\
A \to & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\
X \to & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\
\hline
 & 0 & 1 & 0 & 0 & 1 & 0 & 0
\end{array}
\quad \wedge \quad + \quad
\begin{array}{lccccccc}
 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \\
B \to & 1 & 1 & 0 & 1 & 1 & 0 & 0 \\
X \to & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\
\hline
 & 0 & 0 & 1 & 0 & 0 & 1 & 0
\end{array}
\quad \wedge
$$

$(2^5 + 2^2 + 2^4 + 2^1) \Rightarrow$ final val.

→ If ith-bit is set in both the no's A and B, then it should be set in X.

→ All other bits they can be zero.

[take A & B.]

$X = A \& B.$