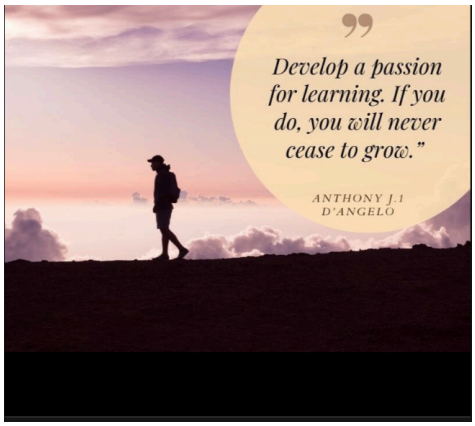
Today's Quote



Today's content:

- → power of left shift
- → set/ unset i^{th} bit
- → check i^{th} bit is set or unset
- → Q1 Count no. of set-bits in N
- → Negative no's
- → Range of Integers
- → Importance of constraints
- → Binary no. substraction.

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

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

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

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

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

$1 << i = \cdot \quad [2^i]$

$$\begin{array}{c} 3\ 2\ 1\ 0 \\ (1\ 0\ 0\ 0)_2 \end{array} = (\quad)_{10}$$

$\uparrow$ 2^3

Power of Left Shift

N = 45 $\quad$ 1 0 1 1 0 1

Case 1:

$\boxed{OR}$

	5	4	3	2	1	0	
	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	
	1	0	1	1	0	1	
(1<<4)	0	1	0	0	0	0	
	1	1	1	1	0	1	⇒ (61)

$\boxed{N \mid (1 << i)}$ ⇒ Setting i^{th} - bit

$\boxed{XOR}$

	5	4	3	2	1	0	
	1	0	1	1	0	1	→ [45]
(1<<2)	0	0	0	1	0	0	
	1	0	1	0	0	1	→ [41]

	5	4	3	2	1	0	
	1	0	1	1	0	1	→ [45]
(1<<4)	0	1	0	0	0	0	
	1	1	1	1	0	1	→ [61]

$\boxed{N \wedge (1 << i)}$ ⇒ Toggle i^{th} - bit

AND

```
        1 0 1 1 0 1
(1<<2)  0 0 0 1 0 0
        -----------
        0 0 0 1 0 0   ⇒ [2^2]
```

```
        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 ⇒ i^{th}-bit is unset

→ $(1 << i)$ ⇒ i^{th}-bit is set

⇑

{check if i^{th}-bit is set or unset}

Q.) Unset i^{th} bit of the given number if it is set, else no change.

N = 45 , i = 2

```
5 4 3 2 1 0
1 0 1 1 0 1   ⇒   1 0 1 0 0 1
```

N = 45 , i = 4

```
5 4 3 2 1 0
1 0 1 1 0 1   ⇒   1 0 1 1 0 1
```

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

Check Bit:

$N \& (1<<i) == (1<<i)$ — Bit is set.

OR.

$N | (1<<i) == N$ — Bit is set.

Q. Check if i^{th}-bit is set.

① $N \& (1<<i) == (1<<i)$

② $N | (1<<i) == N$

③ $N \wedge (1<<i) < N$

$N \wedge (1<<i)$
- $< N$ — i^{th}-bit is set.
- $> N$ — i^{th}-bit is unset

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

$N = 45$

$i = 2$

$(N >> i)$

5	4	3	2	1	0
1	0	1	1	0	1
0	0	1	0	1	1

④ $(N >> i) \& 1 == 1$

$N = 10 \rightarrow$ 1010 (3 2 1 0)

$i = 3$

$(N << i)$

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

$$\begin{array}{lcccccc} & 5 & 4 & 3 & 2 & 1 & 0 \\ N=45 & 1 & 0 & 1 & 1 & 0 & 1 \\ i=4 & 0 & 0 & 0 & 0 & 1 & 0 \\ & 0 & 0 & 0 & 0 & 0 & 1 \\ \hline & 0 & 0 & 0 & 0 & 0 & 0 \end{array} \quad \&$$

Q.) Count the number of set bits in N.

eg: 45 → $\overset{5}{1}\,\overset{4}{0}\,\overset{3}{1}\,\overset{2}{1}\,\overset{1}{0}\,\overset{0}{1}$ [ans → 4]

35 → 1 0 0 0 1 1 [ans → 3].

idea: [Check for every bit, if it is set or unset.]

int → 4 bytes → 32 bits.

$\underset{7}{1}\ \underset{6}{1}\ \underset{5}{1}\ \underset{4}{1}\ \underset{3}{1}\ \underset{2}{1}\ \underset{1}{1}\ \underset{0}{1} \Rightarrow 2^7+2^6+2^5+ --- 2^0$

$\approx 2^8$

pseudo-code.

```
int Count SetBits (N) {
    ans = 0
    for ( i → 0 to 31) {
        if ( checkBit (N, i) == 1) {
            ans = ans + 1
        }
    }
    return ans;
}
```

int:	long.
31	63

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

T.C → O(log n)

S.C → O(1)

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

2nd approach.

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 →	[0]

pseudo-code.

```
int countSetBits (N){
        count = 0
        while ( N > 0) {
            if ((N & 1) == 1)
                count += 1
            }
            N = N >> 1
        }
    return count
}
```

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

$$\left[\begin{array}{l} N << i \Rightarrow N * 2^i \\ N >> i \Rightarrow \dfrac{N}{2^i} \end{array}\right]$$

{000000 — — 011111}

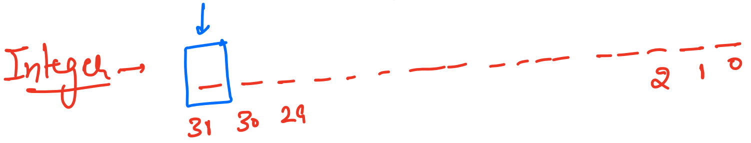
{11111111111111111111111111111111}

Negative no's.

$(-45)_{10} = (?)_2$

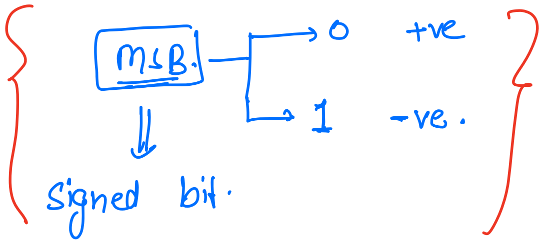
→ {8-bit number}

M.S.B [Most Significant Bit]



$2^{31} > [2^{30} + 2^{29} + \text{— —} \; 2^2 + 2^1 + 2^0]$

$2^{31} > 2^0 \frac{[2^{31}-1]}{[2-1]} = 2^{31}-1$



	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

$(-2^7) + 2^6 + 2^4 + 2^1 + 2^0 = 128 + 64 + 16 + 2 + 1 = 211$

$= -128 + 64 + 16 + 2 + 1 = -45$

$(-12)_{10} = (\ ?\ )_2$ {8 bit integer}

	7	6	5	4	3	2	1	0	
12 →	0	0	0	0	1	1	0	0	
toggle all bits →	1	1	1	1	0	0	1	1	→ 1's compliment.
							+	1	
	$\boxed{1}$	1	1	1	0	1	0	0	→ 2's compliment

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

$= -128 + 116$

$= \underline{\underline{-12}}$

Minimum number ⇒ $\overset{7}{1}\ \overset{6}{0}\ \overset{5}{0}\ \overset{4}{0}\ \overset{3}{0}\ \overset{2}{0}\ \overset{1}{0}\ \overset{0}{0} = -128 = -2^7$

Maximum number ⇒ $\overset{7}{0}\ \overset{6}{1}\ \overset{5}{1}\ \overset{4}{1}\ \overset{3}{1}\ \overset{2}{1}\ \overset{1}{1}\ \overset{0}{1} = 127 = 2^7 - 1$

Range of integers : $[-2^{31}, 2^{31}-1]$

Minimum number ⇒ $\overset{31}{1}\ \overset{30}{0}\ \overset{29}{0}$ — — — $\overset{2}{0}\ \overset{1}{0}\ \overset{0}{0} = -2^{31}$

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

$= -2.147483648 \times 10^9$

Maximum number ⇒ $\overset{31}{0}\ \overset{30}{1}\ \overset{29}{1}$ — — — $\overset{2}{1}\ \overset{1}{1}\ \overset{0}{1} \Rightarrow 2^{31} - 1$

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

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

Minimum no. → $-2^{63} \approx -9 \times 10^{18}$

Maximum no. → $2^{63}-1 \approx 9 \times 10^{18}$

Q) Calculate sum of all array elements.

```
long sum = 0
for ( i → 0 to N-1) {
        sum += arr[i]
}
return sum
```

Constraints

$1 <= N <= 10^5$

$1 <= arr[i] <= 10^6$

$A = [10^6 \ 10^6 \ 10^6 \ \_\_ \ 10^6]$

$N = 10^5$

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

∴ overflow condition.

constraints → T.L.E
→ overflow

Q:) Given two +ve integers [a & b], return a*b

$a <= 2*10^9$
$b <= 2*10^9$

$a*b = 2\times10^9 \times 2\times10^9$
$= 4\times10^{18}$

```
int ans = a * b
return ans
```

[X]

⇓

```
long ans = a * b
return ans
```

[X]

overflow at time of multiplication step.

⇓

```
long ans = long (a*b)
return ans
```

[X]

⇓

```
long ans = long(a) * b
return ans
```

[✓]

long * int = long.

⇓

```
long ans = a ;
ans *= b ;
return ans.
```

[✓]

Subtraction of binary no.'s

45 − 12.

⇒ 45 + (−12)

	(1)	(1)	(1)	(1)	(1)			
	7	6	5	4	3	2	1	0
45 →	0	0	1	0	1	1	0	1
−12 →	1	1	1	1	0	1	0	0
	0	0	1	0	0	0	0	1

$2^5 + 2^0 = 32 + 1 = 33.$

Doubts :

[10:00 A.M.] → By tommorow.

[Friday to Sunday 11 A.M 6:30] continue 2:30 hrs.

12 − 40 = −28

12 + (−40) =

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

$(-2^7) + (2^6) + (2^5) + 2^2$

$= -128 + 64 + 32 + 4 = -128 + 100$

$= -28$

2	40	
2	20	0
2	10	0
2	5	0
2	2	1
2	1	0
	0	1

0 0 1 0 1 0 0 0

1 1 0 1 0 1 1 1

+ 1

1 1 0 1 1 0 0 0