SCALER

Bit Manipulation

TABLE OF CONTENTS

Notes

arr[] → [5, 2, 8, 2, 7, 5, 3, 7, 5]

↓ Sorting

arr[] → [2, 2, 3, 5, 5, 5, 7, 7, 8]

Decimal Number System

→ digit can vary from 0 to 9. (base-10)

$342 \rightarrow 3\times10^2 + 4\times10^1 + 2\times10^0$

$2563 \rightarrow 2000 + 500 + 60 + 3$

$\rightarrow 2\times10^3 + 5\times10^2 + 6\times10^1 + 3\times10^0$

Binary Number System

→ digit can either be 0 or 1. (base→2)

$110 \rightarrow 1\times2^2 + 1\times2^1 + 0\times2^0 = 4+2+0 = \underline{6}.$

$1011 \rightarrow 1\times2^3 + 0\times2^2 + 1\times2^1 + 1\times2^0$

$\rightarrow 8 + 0 + 2 + 1 \Rightarrow \underline{\underline{11}}$

1. Binary to Decimal Conversion

- $(1101)_2 = (13)_{10}$

1 1 0 1

$1 \times 2^0 \to 1$

$0 \times 2^1 \to 0$

$1 \times 2^2 \to 4$

$1 \times 2^3 \to 8$

$N \to 1101$

$\downarrow 10$

$N \to 110$, $r \to 1 \times 2^0$

$\downarrow 10$ $+$

$N \to 11$, $r \to 0 \times 2^1$

$\downarrow 10$ $+$

$N \to 1$, $r \to 1 \times 2^2$

$\downarrow 10$ $+$

$N \to 0$, $r \to 1 \times 2^3$

13

- $(1011010)_2 = (90)_{10}$

$0 \times 2^0 \to 0$

$1 \times 2^1 \to 2$

$0 \times 2^2 \to 0$

$1 \times 2^3 \to 8$

$1 \times 2^4 \to 16$

$0 \times 2^5 \to 0$

$1 \times 2^6 \to 64$

Code. →

```
int ans = 0;
int power = 1   // initially, it is 2^0 .

while( N > 0) {
    int r = N % 10;
    N = N /10;
    ans += (r * power);
    power *= 2;
}
return ans;
```

N → $\cancel{1}\cancel{0}\cancel{1}$

$= 5$

ans $= 0 + 1 + (0 \times 2) + 4$

power $= \cancel{1}\ \cancel{2}\ \cancel{4}\ 8$

$r \to \cancel{1}\ \cancel{0}\ 1$

$$\boxed{\begin{array}{l} T.C \to O(\log_{10} N) \\ S.C \to O(1) \end{array}}$$

2. Decimal to Binary

- $(20)_{10}$ = $(10100)_2$

2	20	
2	10	→ 0
2	5	→ 0
2	2	→ 1
2	1	→ 0
	0	→ 1

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

2	45		
2	22	→ 1	$\times 10^0$
2	11	→ 0	$\times 10^1$
2	5	→ 1	$\times 10^2$
2	2	→ 1	$\times 10^3$
2	1	→ 0	$\times 10^4$
	0	→ 1	$\times 10^5$

code →

```
int ans = 0 ;
int power = 1 ;
while ( N > 0) {
    int r = N % 2 ;
    N = N / 2 ;
    ans += (r * power);
    power *= 10;
}
return ans;
```

T.C → $O(\log_2 N)$
S.C → $O(1)$

Addition of two decimal numbers -

```
          0   1   1   0
n1 →          3   6   8
n2 →          1   4   5
             -----------
ans →         5   1   3
             -----------
```

→ find the last digit of n1 & n2

→ sum = carry + l1 + l2

→ Remove last digit from n1 & n2

→ update ans & carry.

Addition of two binary numbers -

```
        0   1   0   0   1   0
n1  →           1   1   0   1
n2  → +         1   0   0   1
       -------------------------
            1   0   1   1   0
       -------------------------
```

sum → 1 + 1 + 0 → 2 (q → 1, r → 0)

sum → 0 + 0 + 1 ⇒ 1 (q → 0, r → 1)

sum → 1 + 0 + 0 → 1 (q → 0, r → 1)

sum → 1 + 1 + 0 ⇒ 2 (q → 1, r → 0)

sum → 0 + 0 + 1 ⇒ 1 (q → 0, r → 1)

0	y	0	0	1	0	0	0
		1	1	0	1	0	1
+		1	0	0	1	1	0
	1	0	1	1	0	1	1

sum = 1 + 0 + 0 = 1 $\xrightarrow{q}$ 0, $\xrightarrow{r}$ 1

sum = 0 + 1 + 0 = 1

sum = 1 + 1 + 0 = 2 $\xrightarrow{q}$ 1, $\xrightarrow{r}$ 0

sum = 0 + 0 + 1 = 1 $\xrightarrow{q}$ 0, $\xrightarrow{r}$ 1

sum = 1 + 0 + 0 = 1 $\xrightarrow{q}$ 0, $\xrightarrow{r}$ 1

sum = 1 + 1 + 0 = 2 $\xrightarrow{q}$ 1, $\xrightarrow{r}$ 0

sum = 0 + 0 + 1

sum = 1 → 01

sum = 2 → 10

sum = 3 → 11

	0	0	1	1	0	0
n1 →		1	0	1	1	0
n2 +		0	0	1	1	1
ans -		1	1	1	0	1

code →

//n1, n2 → given.

```
int ans = 0 , power = 1 ;
int carry = 0;
while ( n1 > 0 || n2 > 0 || carry > 0 ) {
        l1 = n1 % 10;          } last digit of n1 & n2
        l2 = n2 % 10;
        n1 = n/10 , n2 = n2/10; } reducing n1 & n2
        sum = l1 + l2 + carry;
         q = sum/2;
         r = sum % 2;
         ans += (r * power);
         carry = q ;
         power *= 10;
}
return ans;
```

$T.C \rightarrow O(\log_{10} \max(n1, n2))$

$S.C \rightarrow O(1)$

Bitwise Operators

! , & , | , ^ , << , >> - Advanced BM

same same puppy shame

a	b	a&b	a|b	a^b	~a/!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

5 & 6 = ? 4

20 & 45 = ? 4

```
5  →   1   0   1
                     &
6  →   1   1   0
--------------------------
       1   0   0
--------------------------
```

```
20  →   0   1   0   1   0   0
                                 &
45  →   1   0   1   1   0   1
--------------------------------------
        0   0   0   1   0   0
--------------------------------------
```

20 | 45 = ? 61

20 ^ 45 = ? 57

20	→	0	1	0	1	0	0
45	→	1	0	1	1	0	1
		1	1	1	1	0	1

\|

20	→	0	1	0	1	0	0
45	→	1	0	1	1	0	1
		1	1	1	0	0	1

^

Negative Numbers

int → 4B → 32 bits.

Assumption 1 int → 8 bits

sign bit

-13

0 0 0 0 1 1 0 1 (13)

1's compliment 1 1 1 1 0 0 1 0

+ 1

2's compliment 1 1 1 1 0 0 1 1

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

$(-2^7) + (2^6 + 2^5 + 2^4 + 2^1 + 2^0) \Rightarrow -128 + 64 + 32 + 16 + 2 + 1$

$\Rightarrow -128 + 115$

$\Rightarrow -13$

0

↓

No. is positive

2^{31}

↓

1 1 1 1 1 ... 1

↓

No. is negative

$2^0 + 2^1 + 2^2 + 2^3 + \_\_\_\_ \; 2^{30}$

$\Rightarrow \frac{1[2^{31}-1]}{(2-1)} \Rightarrow 2^{31}-1$

Binary representation of -3 (in 8 bits)

0 0 0 0 0 0 1 1

↓

1 1 1 1 1 1 0 0

+ 1

1 1 1 1 1 1 0 1

$-2^7 + 2^6 + 2^5 + 2^4 + 2^3 + 2^2 + 2^0$

1 0 0 0 0 0 0 0 $\leftarrow$ $\underline{-128}$

1 0 0 0 0 0 0 1

$\boxed{0}$ 0 0 0 0 0 0 0

1

1 0

1 1

⋮

0 1 1 1 1 1 1 1 $\boxed{\underline{127}}$

$\underline{8 \text{ bits}} \rightarrow [-128, 127] \rightarrow [-2^7, 2^7-1]$

$\underline{32 \text{ bits}} \rightarrow [-2^{31}, 2^{31}-1]$

$\rightarrow [-2147483648, 2147483647]$

$\rightarrow [-2 \times 10^9, 2 \times 10^9]$

integer range → $[-2^{31}, 2^{31}-1]$

approx → $[-2\times10^{9}, 2\times10^{9}]$

long range → $[-2^{63}, 2^{63}-1]$

approx → $[-9\times10^{18}, 9\times10^{18}]$

Importance of Constraints →

int $a \to 10^5$, $b \to 10^6$

```
int c = a*b;
print(c);
```

} ⇒ wrong ans because of overflow.

---

int $a \to 10^5$, $b \to 10^6$

```
long c = a*b;
print(c);
```

↳ overflow happens here

---

int $a \to 10^5$, $b \to 10^6$

```
long c = (long)(a) x b;
print(c);
```

1 <= N <= 105

1 <= A[i] <= 106

$1 \le N \le 10^5$

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

Is the following code correct ?

long

```
int sum = 0;
for (int i = 0; i < N; i++) {
    sum = sum + A[i];
}
print(sum)
```

$$\underbrace{[10^6, 10^6, 10^6, 10^6, 10^6, \sim\!\sim - 10^6]}_{10^5 \text{ such elements.}}$$

$\Downarrow$

Sum of all $\rightarrow$ 10^{11}

integer can't hold this.

Revise assignment problems → Notes.

From every 5-7 problem - pick 1 & try to code it.

+ve.

0 0 1 0 1 1 0 1

↓

1 1 0 1 0 0 1 0

$+ \; 1$

1 1 0 1 0 0 1 1

-ve.