Bit Manipulation

TABLE OF CONTENTS

- 1. Number System and Conversion
- 2. Addition of two binary numbers
- 3. Bitwise Operators
- 4. Negative Numbers
- 5. Ranges of Numbers
- 6. Importance of Constraints



$$aw(1) + \{5, 2, 8, 2, 7, 5, 3, 7, 5\}$$

$$\begin{cases} Sorting \\ Arill = \{2, 2, 3, 5, 5, 7, 7, 8\} \end{cases}$$

Decimal Number System

→ digif can vary from 0 to 9. (base-10)
$$342 \rightarrow 3 \times 10^{2} + 4 \times 10^{6} + 2 \times 10^{6}$$

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

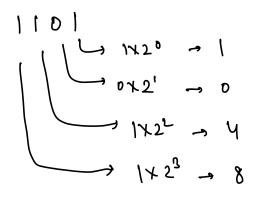
$$-2 \times 10^{2} + 5 \times 10^{2} + 6 \times 10^{6} + 3 \times 10^{6}$$

Binary Number System

- digif can either be
$$0$$
 or $1 \cdot (bas(-2))$
 $110 - 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 4 + 2 + 0 = 6$.

 $1011 \rightarrow 1 \times 2^2 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$
 $-3 \quad 8 + 0 + 2 + 1 \Rightarrow 11$

1. Binary to Decimal Conversion



•
$$(1011010)_2 = (90)_{10}$$

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

| $(101100)_2 = (90)_{10}$

| $(101100)_2 = (90)_{10}$

| $(101100)_2 = (90)_{10}$

| $(101100)_2 = (90)_{10}$

| $(101100)_2 = (90)_{10}$

| $(101100)_2 = (90)_{10}$

| $(101100)_2 = (90)_{10}$

| $(101100)_2 = (90)_{10}$

| $(101100)_2 = (90)_{10}$

| $(101100)_2 = (90)_{10}$

| $(101100)_2 = (90)_{10}$

| $(101100)_2 = (90)_{10}$

| $(101100)_2 = (90)_{10}$

| $(101100)_2 = (90)_{10}$

| $(101100)_2 = (90)_{10}$

| $(101100)_2 = (90)_{10}$

| $(101100)_2 = (90)_{10}$

| $(101100)_2 = (90)_{10}$

| $(101100)_2 = (90)_2$

| $(101100)_2 = (90)_2$

| $(101100)_2 = (90)_2$

| $(101100)_2 = (90)_2$

| $(101100)_2 = (90)_2$

| $(101100)_2 = (90)_2$

| $(101100)_2 = (90)_2$

| $(101100)_2 = (90)_2$

| $(101100)_2 = (90)_2$

| $(101100)_2 = (90)_2$

| $(101100)_2 = (90)_2$

| $(101100)_2 = (90)_2$

| $(101100)_2 = (90)_2$

| $(101100)_2 = (90)_2$

| $(101100)_2 = (90)_2$

| $(101100)_2 = (90)_2$

| $(101100)_2 = (90)_2$

| $(101100)_2 = (90)_2$

| $(101100)_2 = (90)_2$

| $(101100)_2 = (90)_2$

| $(101100)_2 = (90)_2$

|

$$N \rightarrow 110$$
 $V \rightarrow 110$
 $V \rightarrow 11$
 $V \rightarrow 0 \times 2^{1}$
 $V \rightarrow 1$
 $V \rightarrow 1$

```
Code. -
 inf ans = 0;
int power= 1 Minitally, it is 2°.
                                                      = \frac{2}{5}
                                                  an = 0 +1 + (0x3+4
 while( N > 0) {
                                   Nº YOX
                                                  Power = 12 48
     int r = N % 10;
     N = N/10;
                                                v → 1 p 1
     ans += (r + power);
     power *= 2;
                                  T.C - O(log.N)
  return ans;
```

2. Decimal to Binary

•
$$(20)_{10} = (0)_{00}$$

•
$$(45)_{10} = (10)_{10})_2$$

```
# code =

inf ans = 0;

inf power = 1;

while ( N > 0) {

inf r = N/2;

N = N/2;

Ans + = (r + power);

power * = 10;

Yeturn ans;
```

Addition of two decimal numbers -

$$n1 \rightarrow 3 \quad 6 \quad 8$$
 $n2 \rightarrow 1 \quad 4 \quad 5$
 $am \rightarrow 5 \quad 1 \quad 3$

Addition of two binary numbers -

$$8um^2 | +0+0=1$$

$$8 \text{ lm} = 1 + 0 + 0 = 1 = \frac{2}{v}, \frac{0}{1}$$

Code -

```
//NI, N2 -> given.
  inf ans=0, power=1;
  int carry = 0;
   while (n1 >0 11 n2 >0 11 carry >0) {
          1 = n1 /.10; } last digit of n1 & n2
          22 = n2 /, 10;
          n(=n/10), n2=n2/10; 3 reducing n1 l n2
          sym = 11 + 12 + carry;
            q = Sum/2;
            r = sum 1.2;
            ans += (r* power);
            carry = 9;
            power = 10;
                                Tic \rightarrow 0 (log 10 Max(n1,n2))

S.C \rightarrow 0 (1)
     return ans;
```

Bitwise Operators

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

same same puppy shame

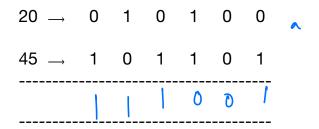
a	b	a&b	a b	a^b	~a/!a		
0	0	O	O	O	1		
0	1	0	1	1	1		
1	0	O	1	1	0		
1	1	1	1	D	0		

$$5 \rightarrow 1 \quad 0 \quad 1$$
 $6 \rightarrow 1 \quad 1 \quad 0$

| 0 0



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





Negative Numbers

2's compliment

$$(-2^{7})$$
 + $(2^{6} + 2^{5} + 2^{6} + 2^{1} + 2^{0})$ => -128 + 64 + 32 + 16 +2+1
=> -128 + 115

+ 1



Bit Manipulation

0

$$\downarrow$$

No. is positive





No. is negative

$$2^{0} + 2^{1} + 2^{2} + 2^{3} + 2^{3} + 2^{3}$$

$$\frac{1\left[2^{3^{1}}-1\right]}{\left(2-1\right)} \Rightarrow 2^{3^{1}}-1$$

Binary representation of -3 (in 8 bits)

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

$$\frac{32 \text{ bits}}{2} \rightarrow \left[-2^{21}, 2^{21} - 1\right]$$

$$-2 \times 10^{9}$$
, 2×10^{9}

$$long range - [-2^{62}, 2^{63}-1]$$

Importance of Constraints.

int a -> 105, 6-106

inf c = a + b;

print (c); 3 = wrong and because of overflow

int a -> 105, b -> 106

long c = a * b; overflow happens have print (c);

int a -> 105, 6-106

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

Is the following code correct?

```
14 N 4 105
14 A(1) 2 106
```

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

[
$$lo^b$$
, lo^6 , lo^6 , lo^6 , lo^6 , lo^6 , lo^6]

105 such elements.

11

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

integer carit hold this.

Rivise assignment problems - Noks.

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

+N(.