CLASS 41

25 March 2023 11:09

CLASS-41

Number + DP + Rec

Number Theory → Difficult

optimization → Medium

Euler Totient → Easy

Bit Manipulation

Comp → 0 or 1

0 → False

1 → True

10 12 abcd a

⇓

Binary Nos

Sequence of 0 and 1

10

10 → 1010

a → 97

↓

Binary

10

1010

int → 4 bytes → 32 bits

1 bit

1 byte = 8 bits

0 or 1

10

1 0 1

0 or 1

32 space [Combination of 0 or 1]

Represent 10

| 0 | 0 | 0 | 0 | 0 | 0 | - - - - - 0 | 0 | 1 | 0 | 1 | 0 |

← 32 →

10 → Binary NO [0 and 1 combination)

10 → Binary No (0 and 1 combination)

Char → 1 byte → | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |

8 bits

int → 32 bits

1010 , 0010 , 1111

Decimal to Binary

int → Binary

2	14
2	7	0
2	3	1
2	1	1

1110

17 → ~~0001~~ 10001

23 → ~~10001~~ 10111

Binary to Decimal

$(23)_{10} = (10111)_2$

10111 → $2^0 * 1 + 2^1 * 1 + 2^2 * 1 + 2^3 * 0 + 2^4 * 1$

$1 + 2 + 4 + 16 = 23$

Bits

$\log_2 n$

2^n

11111 ⇒ 31

$2^4 * 1$, $2^3 * 1$, $2^2 * 1$, $2^1 * 1$ — 2, $2^0 * 1$ → 1

16, 8, 4

$8 + 16 + 4 + 1 + 2 = 31$

1000 ⇒ 8

No contribution in ans

2^n = 2, 4, 8, 16, 32, 64, 128, 256

Binary

Binary

$2^4 = 16$

10000

$2^1, 2^2, 2^3, 2^4, 2^5, 2^6, 2^7, 2^8$

10000

$2^6 * bit$

00100

$2^2 * 1 =$

0001000

$2^3 * 1$

$2^1, 2^2, 2^3, \ldots$

2^8

index

int → 32 bit

5 - 1

2^3

8 → 1000

7 → 0111

$2^n - 1$

2^4

16 → 10000

15 → 01111

16 - 1 = 15

32 → 100000

31 → 011111

Operations on Bits

10 → 1010

8 → 1000

10 + 8 = 18

10 - 8 = 2

10 * 8 = 80

$\frac{10}{8} = 1$

10 % 8 = 2

1. and (&)
2. or (|)
3. ^ (xor)
4. << (left shift)
5. >> (Right shift)
6. ~ (Negation)

10 + 8 = 18

Boolean Table

1 1 0 0

(15)

& 1 1 0 0
1 0 0 0

output = 1

356
+ 255

Both Input = 1 => output 1

X	Y	X & Y
1	1	1
1	0	0
0	1	0
0	0	0

1101 => 13
& 1001 => 9
1001 => 9

13 & 9

13 & 9 &

even odd

| (or) operator

atleast 1 → output 1

X	Y	X | Y
1	1	1
1	0	1
0	1	1
0	0	0

1100 → 12
| 0011 → 3
1111 → 15

=> 12 | 3 = 15

16 ?

^ (xor)

Both same => 0
other wise => 1

X	Y	X ^ Y
1	1	0
1	0	1
0	1	1
0	0	0

1 0 0 1 1 1 1
^ 0 1 0 0 1 1 1
1 1 0 1 0 0 0

$\frac{0}{0}$ $\frac{1}{0}$ $\frac{1}{0}$

~ (Negation)

~ 110011

$1 \rightarrow 0$ [worry]
$0 \rightarrow 1$

$\boxed{001100}$

Right shift operator [Binary Exponentiation] (>>)

n = $\boxed{110011}$ $\boxed{n >> 1}$

$\boxed{11001}$

110011 n >> 2

0 32

↓

31st

-- - 0011001

$1111 \rightarrow \boxed{31}$ $\boxed{31 >> 1}$ $\frac{31}{2} = 15$

$111 \rightarrow 15$

15 >> 1 $n/2 \Rightarrow \boxed{n >> 1}$

$111 \rightarrow 7$

$\boxed{n = n >> 1}$

$31 >> 2$ $\frac{31}{2} = \frac{15}{2} = 7$

$111 \rightarrow 7$

$n >> i \Rightarrow \boxed{\frac{n}{2^i}}$

$1111 \rightarrow [2^0 \times 1 + 2^1 \times 1 + 2^2 \times 1 + 2^3 \times 1] / 2$

$111 \rightarrow [2^0 \times 1 + 2^1 \times 1 + 2^2 \times 1]$

111 → $2^0 \times 1 + 2^1 \times 1 + 2^2 \times 1$

2^i 2^{i-1} 2

$\frac{2^i}{2} = 2^{i-1}$

left shift (<<)

1 0 1 1 1 << 1 => 1 0 1 1 1 0

2^0 2^1

23 → 46

$2^i \to 2^{i+1}$ $\times 2$

$(n << i) \Rightarrow n \times 2^i$

111

7 << 3

~~000~~ 111000

overflow

int → 32 bits

int → 2^{31} 10^9

$10^5 << 1 \Rightarrow 2 \times 10^5$

+ve -ve

± or −

int 32 bits 30

10^9 << 1 => overflow

sign 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 LSB

-ve no, Random

31 0

MSB 2^{31} 0 − 30 $2^0 \times 1$

0 − 30 31 bits = 1

int → +ve no, -ve no

unsigned int

Decimal = 5

int → Max value

−5 → 1 | 0 0 - - - - - - 101 Value

$2^{31} - 1$

1000 − 1

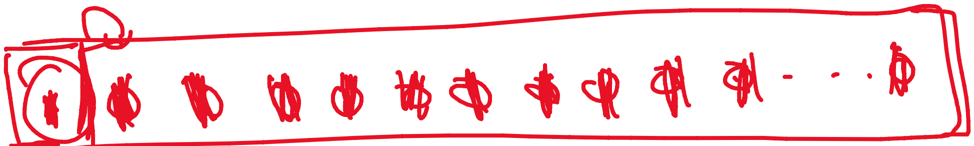
111

1000 − 1

111

3 → 111 ⇒ 7

2^{31} →



31 30

int

$2^{31}-1$

$-2^{31}-1$ to $2^{31}-1$

long long → 8 ⇒ 64 bits

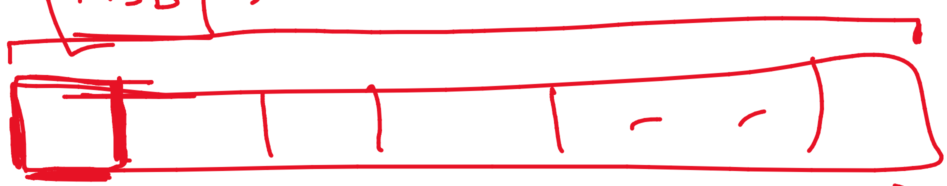
unsigned int

−ve values

+ve value

−ve

MSB → contribute to value



31 0

unsigned int = double of int

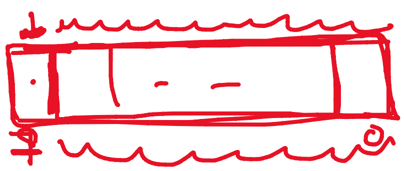
$2^{32}-1$

$2^{31}-1 + 2^{31}$ ⇒ $2^{32}-1$

Binary level

Char →

NO



7 0

8 bits ⇒ 2^8 = 256

2^7 ⇒ 128

Max value

−128 to 127

256

7 6 5 4 3 2 1 0

→ 2^7-1

127

1 byte

64

Max ⇒ $2^{63}-1$

2^8-1

= 255

63 → MSB

bool

MSB → LSB

1 0 1 0

Setbit → 1

1010

unsetbit → 0

operators

1) How to check if i^{th} bit is set

1 0 1 0

Right shift

i ⇒ index = 1

i = 2

a >> (i+1)

a >> i

1 → binary → 0 0 0 1

1 1 ⇒ 1

1 0 1

& 1 → 1

& → 1 1 → 1

0

(a >> i) & 1

1 0 1 0 ⇒ 0 0 0 1

& 1

0

0 1 0 1 0

& 1 0 0 0

1 0 0 0

1 << i

1 << i

1 → 0 0 0 1

1 0 0 0

1 << 3

1 << 3

0 0 0 0 0

& 0 0 0 0 1

0

0 0 0 0 1

& 0 0 0 0 1

1

0 1

0 0 0 0 1 0

0 0 1 0 → 0 0 0 1

&

1 0 0 0

0 0 0 0

$\boxed{\begin{array}{l}(a >> i) \& 1 \\ a \& (1 << i)\end{array}}$

a & b

0 1 ⇒ 0

left shift

if i^{th} bit is set or not