10 - 8 bit binary representation

0 0 0 0 1 0 1 0

-10 = 1 0 0 0 1 0 1 0

Sign bit

-3 = 1 0 0 0 0 0 1 1

-4 = 1 0 0 0 0 1 0 0

× 0 0 0 0 0 1 1 1

⇒ +7

-3 = 1 0 0 0 0 0 1 1

10 : 0 0 0 0 1 0 1 0

= -13 × 1 0 0 0 1 1 0 1

0 0 0 0 0 0 0 0 = 0

1 0 0 0 0 0 0 0 = -0

→ $-a = 2's\ a = is\ a + 1 = \sim a + 1$

$-a = \sim a + 1$ ⟶ $0 \rightleftarrows 1$

10: 0 0 0 0 1 0 1 0

~10: 1 1 1 1 0 1 0 1

+1: 0 0 0 0 0 0 0 1

	7	6	5	4	3	2	1	0
	-2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
-10:	1	1	1	1	0	1	1	0

MSB

$128 + 64 + 32 + 16 + 4 + 2 = 246$

$-128 + 64 + 32 + 16 + 4 + 2 = -10$

	-2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
23:	0	0	0	1	0	1	1	1

~23: 1 1 1 0 1 0 0 0

+1: +1

---

1 1 1 0 1 0 0 1

$-128 + 64 + 32 + 8 + 1$

$= -128 + 105 = -23$

1 0 0 0 ← +10 * Max = +7, Min = −8

$-2^3\ \ 2^2\ \ 2^1\ \ 2^0$ → [−8, +7]

n bits

n

2 — — → $-2^1\ 2^0$

3 — — — → $-2^2\ 2^1\ 2^0$

4 — — — — → $-2^3\ 2^2\ 2^1\ 2^0$

5 — — — — — → $-2^4\ \ 2^1\ 2^0$

n bits number

Min

1 0 → -2^1

1 0 0 → -2^2

1 0 0 0 → -2^3

1 0 0 0 0 ⇒ -2^4

$= -2^{N-1}$

Max

0 1 → 2^0

0 1 1 → $2^0+2^1 \rightarrow 2^2-1$

0 1 1 1 → $2^0+2^1+2^2 \rightarrow 2^3-1$

0 1 1 1 1 → $2^0+2^1+2^2+2^3 \rightarrow 2^4-1$

$2^{N-1}-1$

// n bit number → $[-2^{N-1}, 2^{N-1}-1]$

	#bits	
// byte ⇒	$n = 8$	$[-2^7, 2^7 - 1] \Rightarrow [-128, 127]$
short int ⇒	$n = 16$	$[-2^{15}, 2^{15} - 1] \Rightarrow [-32768, 32767]$
int ⇒	$n = 32$	$[-2^{31}, 2^{31} - 1] \Rightarrow [-2147483648, 2147483647]$ $\Rightarrow [-2 * 10^9, 2 * 10^9]$
long int ⇒	$n = 64$	$[-2^{63}, 2^{63} - 1] \Rightarrow [-8 * 10^{18}, 8 * 10^{18}]$

$$\Rightarrow 2^{10} = 1024 \approx 1000 \approx 10^3$$

$$(2^{10})^3 \approx (10^3)^3$$

// Cube on both sides

$2^{30} \approx 10^9 \Rightarrow$ Multiply 2 $\Rightarrow$ $2^{31} \approx 2 * 10^9$

// Square on both sides

$2^{60} \approx 10^{18} \Rightarrow$ Multiply 2^3 $\Rightarrow$ $2^{63} \approx 8 * 10^{18}$

⇒ // Given N calculate 2^N

N = 3 → 8

N = 10 → 2^{10} = 1024

N = [0, 62]

— power2(N) {

| return 1 << N

| 1L << N

}

N = 30 → 2^{30} ✓

N = 31 → 2^{31} * overflow

N = 60 → 2^{60} ✓

N = 61 → 2^{61} ✓

N = 62 → 2^{62} ✓

N = 63 → 2^{63} *

type of 1 becomes long

Python: Strings

Java: Big Integer

C++:

1 = 0 0 0 0 0 0 0 1

1 << 1 = 0 0 0 0 0 0 1 0 = 2

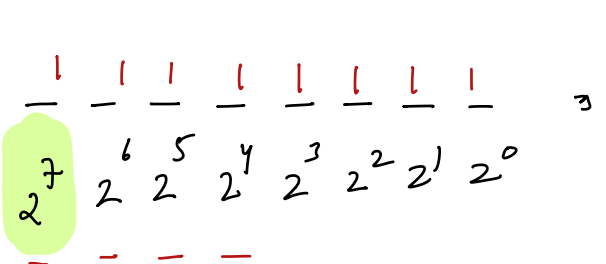
1 << 2 = 0 0 0 0 0 1 0 0 = 2^2

1 << 3 = 2^3

1 << N = 2^N

c/c++ → { unsigned / signed }

unsigned 8 bit Number



→ { 0, 255 }

signed 8	unsigned 8
[-128, 127]	[0, 255]

// unsigned MSB is +ve

By default it is signed

int x;

↓

Signed int

keyword

unsigned int x

↳ ↓

unsigned int

→ a - b

a + (-b)

↳ Is can we get binary of -b
↳ Can we get binary of a

→ MSB → Bit Wise Operators

→ Ranges → Basic problems : → checkBit(N, i)

→ Overflow → 27th Contest : 2hrs } 4 Problems

2-3 days : 2 hrs

Jan - 3rd

Strings	Recursion - 1	LinkedList	Subsets Subsequence
Hashing 1	Recursion - 2	Trees - 1	Problem Solving - I
Hashing 2	Stacks	Trees - 2	Problem Solving - II

Feb : Adv :

1) Arrays -

2) Bits -

3)

1) 10 days holidays

- :→ per day 10/15 to revise a session Notes
- :→ Clear all your backlogs
- :→ Please watch your doubts

2) Hackerrank : Solve topic by topic