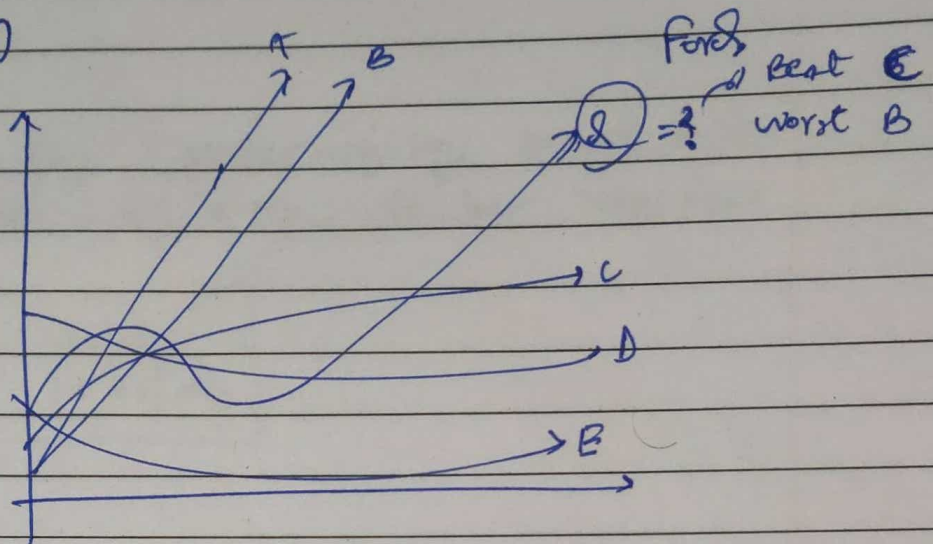


Time Complexity Analysis.

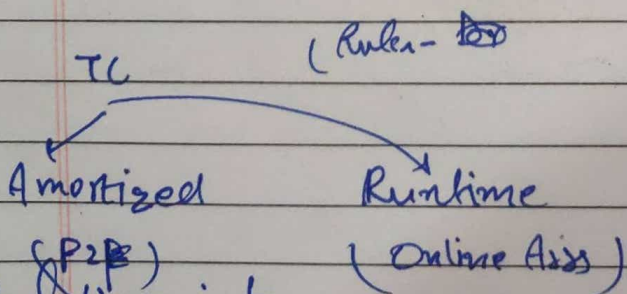
1 →
2 →

- 3 → Any Good way
- 4 → now imp
- 5 → Time Analysis
- 6 → Testing



- Best case → highest UB
- Worst case lowest UB

Big(O) if $f(n) = o(g(n))$
then $f(n) \leq c \cdot g(n)$



- Somebody there is to hear you out. Q. $100 \times 10^5 \times \log_{10} 10^5$

~~$10^5 \times 5$~~

500×10^5
ans.

- Drop Constant
- Do not drop constant

AP, GP, HP.

a
a+d
a+2d
⋮
a+(n-1)d

$$T_n = a + (n-1)d$$

$$S_n = \frac{n}{2} [2a + (n-1)d]$$

$$= \frac{n}{2} [a + l]$$

GP.

a, ar, ar², ..., arⁿ⁻¹

$$S_n = \frac{a(1-r^n)}{1-r} \quad \forall |r| < 1$$

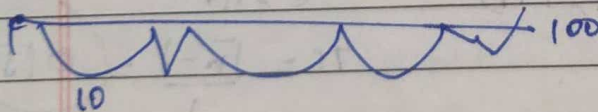
$$S_{\infty} = \frac{a}{1-r}$$

r: Common Ratio

HP: - reciprocal of AP.

① for (i=1; i<N; i++)
 {
 // TC
 // (end-start)
 // iter

$$\Rightarrow \frac{N}{\sqrt{N}} = \sqrt{N}$$



$$\frac{100}{10} = 10$$

② for (i=1; i<N; i++)
 for (j=1; j<N; j+=sqrt(N))
 print(x);

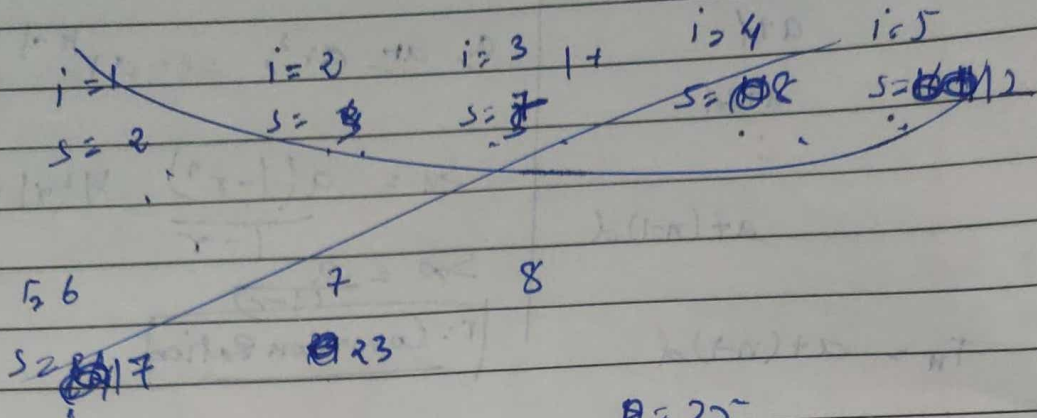
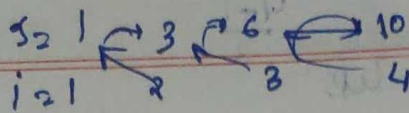
$$O(\sqrt{N})$$

$$\sqrt{1} + \sqrt{2} + \dots + \sqrt{N}$$

36

6th
13
19
25
31

③



Stop when $k^{th} \text{ jump} = N$

$$\frac{k(k+1)}{2} = N$$

$$O(k^2 + k) = O(2n)$$

$$i=1; j=1$$

$$O(k^2) = O(n)$$

$$O(k) = O(\sqrt{n})$$

while ($j \leq N$)

{ $i++$;

$j = j + i$;

count (*);

}

④ for ($i=1$; $i^2 \leq N$; $i++$)

$$s=1$$

$$e = \sqrt{N}$$

$$TC = \frac{\sqrt{n}-1}{1} = O(\sqrt{n})$$

⑤

$$O(n)$$

$$1+2+3 \dots N$$

for ($i=1$; $i \leq N$; $i++$)

for ($j=1$; $j \leq i$; $j++$)

$$O(n)$$

$$O(n)$$

Replace with $\frac{n(n+1)}{2}$

for ($k=1$; $k \leq 10$; $k++$)

$$O(n^2)$$

6) for (i=1; i<=N; i++)
 for (j=1; j<=i^2; j++)
 for (k=1; k<=N/2; k++)

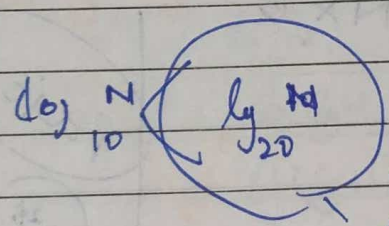
$O(N^3)$
 $O(N^2)$
 $O(N/2)$
 $O(N^4)$

7) for (i=1; i<=N; i=i^2)
 print(x)

$\log n$

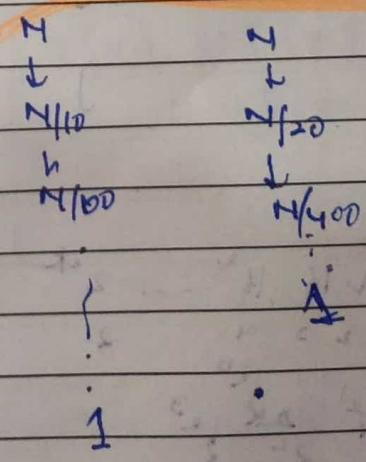
j: 1 2 4 8 16 ... 2^{k-1}
 i: 1 2 3

$2^{k-1} > n$
 $\log_2 2^{k-1} > \log n$
 $k < \log_2 n + 1$
 TC $\log_2 n$



Higher Compression factor
 (how fast N can be compressed to 1)

8) $O(\frac{n}{2})$ for (i=N/2; i<=N; i++)
 $O(N/2)$ for (j=1; j<=N/2; j++)
 $\frac{n}{2} \times \frac{n}{2} \times \log n$
 $O(\log n)$ for (k=1; k<=N; k=k^2)
 print(x)



$n^2 \log n$

$$\log n = \left(1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \dots + \frac{1}{N}\right)$$

$$\log(1+x) = x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \dots \quad (-1 \leq x \leq 1)$$

⑪ for (i=1; i < N; i++) $O(n)$ ✓

for (j=1; j < N; j=j+i) log

i=1 j=1, 2, 3, ..., N	j=2 j=1, 3, 5, ..., N $N/2$	i=3 j= $\frac{N}{3}$ times	i=N j=1 time.
--------------------------	-----------------------------------	-------------------------------	------------------

$$N \left(1 + \frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{N}\right) = \underline{O(n \log n)}$$

⑫ $N = 2^{2^k}$

for (i=1; i < N; i++)

j=2;

while (j < N)

{ j=j²

}

i=1

2, 4, 16, 16², ..., 2^{2^k}

1 2 3 4 2

2^{2⁰} 2^{2¹} 2^{2²} 2^{2³} 2^{2⁴}

$$2^{2^k} > N$$

$$\log 2^{2^k} \geq N$$

$$2^k \geq N$$

$$k = \log \log N$$

$$k = \log \log N$$

waylog

ident 2^{2^k}

i=1

2, 4

$$2^{2^0} < 2^{2^1} < 2^{2^2} < 2^{2^3} < 2^{2^4}$$

$$2^{2^k}$$

For Explanation

$j \rightarrow 2$			
$8 \rightarrow N$			
$K=1$	$K=2$	$K=3$	$K=4$
$N=2^8$	$N=2^{2^2} = 2^4$	$N=2^{2^3} = 8$	$N=2^{2^4} = 16$
$j=2 \text{ times}$	$j=3 \text{ times}$	$j=4 \text{ times}$	$j=5 \text{ times}$
$2 \leq 2^2$	$2 \leq 2^2 \leq 2^4$		$2 \leq 2^2 \leq 2^4$
			$2 \leq 2^4$

$$\rightarrow j = (K+1)$$

$$2^{2^K} = N$$

$$\log_2 2^{2^K} = \log_2 N$$

$$2^K = \log_2 N$$

$$\log_2 2^K = \log_2 \log_2 N$$

$$K = \log(\log N)$$

$$T(n) = T(n-1) + 1$$

$$T(n-1)$$

$$T(n-1) = 1 + T(n-2)$$

$$T(n-2) = 1 + T(n-3)$$

$$T(n) = 1 + (1 + (1 + T(n-3)))$$

$$= 3 + T(n-3)$$

$$T(n-3) = T(1)$$

$$T(n) = 1 + T(n-1)$$

$$= 2 + T(n-2)$$

\vdots

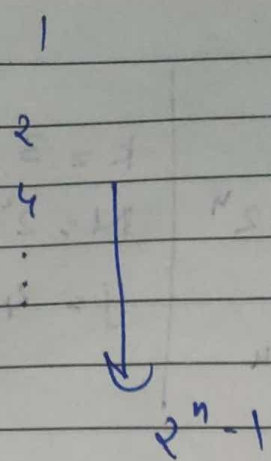
$$K + T(n-K)$$

\vdots

$$(n-1) + T(n-(n-1))$$

$$(n-1) + T(1)$$

$$\underline{\underline{n}}$$



$$S_{\infty} = \frac{1(2^n - 1)}{2 - 1} = 2^n - 1$$

Master's Theorem

$$T(n) = aT(n/b) + O(n^k \log^p n)$$

$$a > 1, b > 1, k \geq 0 \quad | \quad p \rightarrow \text{real No.}$$

$\Theta = \text{WC}$
 $\Omega = \text{BL}$
 $\Theta = \text{Avg Case}$

1) $(a > b^k)$ then $T(n) = O(n^{\log_b a})$

2) $a = b^k$ then $p > -1$, then $T(n) = O(N^{\log_b a} \cdot \log^{p+1} N)$

$p = -1$ then $T(n) = O(N^{\log_b a} \cdot \log \log N)$

$p < -1$ then $T(n) = O(N^{\log_b a})$

3) $(a < b^k)$ then $p > 0$, then $T(n) = O(n^k \log^p n)$

$p < 0$, then $T(n) = O(n^k)$

Q1 $4T(n/2) + n^2$

$a=4$ $b=2$ $k=2$ $P=0$

Q1

$a = b^k$

$\left[n^{\log_2 4} \cdot \log n \right]$

$\Rightarrow n^2 \log n$

Q2

$T(n) = 2T(n/2) + \sqrt{n}$

$a=2$ $b=2$ $k=\frac{1}{2}$

$b^k = 2^{1/2} = 1.4$

$a > b^k$

$\Theta(n^{\log_2 2}) = N^1 = \underline{n}$

Latode

\Rightarrow $< 10^8$ Computations \rightarrow (1 sec) 50% acceptance
will work

$1 \leq N \leq 20$

$1 \leq N \leq 10^4$

$\Rightarrow O(n \log n), n \sqrt{n}$

10^{12}
 10^9

$\Rightarrow \log n$

$10^6 \rightarrow 1M$

$\Rightarrow \log n, \sqrt{n}$

$10^9 \rightarrow 1B$

$10^7 / 10^4$

$\Rightarrow n$

$\log_2 10^9 \approx 32$

$\log_2 10^8 = 28$

10^5

$\Rightarrow n \log n$

10^4

$\Rightarrow n \log n, n \sqrt{n}$

10^3

$\Rightarrow O(n^2)$

10^2

$\Rightarrow n^3$

$10^8 \rightarrow 100 \text{ am}$

$1 < N < 10^5$ }
Recursion X
DP $O(n^2)$ X
Greedy/BS ✓
If 10^8 then ✓

$(n-1) \times n$

$\bar{a} = (1/n) \sum_{i=1}^n a_i$

$$\frac{1}{x} = \frac{1}{x}$$

$$p \cdot 1 = \frac{1}{x}$$

$$x = \frac{1}{p} \quad \left(\frac{1}{p} \right) \cdot n = \frac{1}{p} \cdot n$$

Start at

end

←

initialization

(1)

$$x = \frac{1}{p} \cdot n$$

$$p \cdot 1 = \frac{1}{x}$$

$$101 \times 101$$

$$101 \times 101$$

$$101 \times 101$$

$$101 \times 101$$

$$101 \times 101$$

$$101 \times 101$$

$$101 \times 101$$

$$101 \times 101$$

$$101 \times 101$$

$$101 \times 101$$

$$101 \times 101$$

$$101 \times 101$$

$$101 \times 101$$

$$101 \times 101$$

$$101 \times 101$$

$$101 \times 101$$

$$101 \times 101$$

$$101 \times 101$$

$$101 \times 101$$

$$101 \times 101$$

$$101 \times 101$$

$$101 \times 101$$

$$101 \times 101$$

$$101 \times 101$$

$$101 \times 101$$

Bit magic:

bitwise AND

$$x: 0 \dots 0011$$

$$y: 0 \dots 0110$$

$$(x \& y): 0 \dots 0010$$

```
public class test {
    public static void main (String[] args) {
        int x = 3, y = 6;
        System.out.println (x & y);
    }
}
```

0	0	—	1
0	1	—	1
1	0	—	1
1	1	—	0

$$\begin{array}{r} 011 \\ 4110 \\ \hline 010 = 2 \\ \text{OR } 101 = 5 \end{array}$$

Bitwise NOT

$$x: 000 \dots 01 = 1$$

$$\text{int } x = 1$$

$$\sim x = 111 \dots 10 \quad (2^{32}-1-1) \quad (2^{32}-2)$$

$$\text{print } (\sim x),$$

$$\rightarrow \sim 1 = -2$$

In Java, negative numbers are stored in 2's complement ref

$$\text{Ref of } -x = 2^{32} - x$$

$$\text{Range of int: } -2^{31} \text{ to } 2^{32}-1$$

$$= 1111 \dots 32 \text{ times}$$

Reason

$$\begin{aligned} \text{By } 2^3 - 1 &= 111 \\ 2^3 - 1 - 1 &= 110 \end{aligned}$$

$$\text{By } \text{int } x = 5 \\ \text{print } (\sim x)$$

$$(2^{32} - 1 - 5) = 2^{32} - 6$$

$$\text{O/p is } -6$$

1's complement of

$$\sim x = \underline{\underline{-(x+1)}}$$

Formula

Left shift m

$$x \ll 1$$

~~$$x \ll 2$$~~

$$x \ll 2$$

$$x \ll 2$$

$$x \ll 2$$

Right shift D

~~$$x \gg 1$$~~

~~$$x \gg 2$$~~

$$\frac{x}{2^1}$$

$$x \gg 2 \Rightarrow x/2^2$$

Φ

$x \ll 1 = 2$

$1111 \dots 1$

$+ 111 \dots 10$

$\Rightarrow 2^{32} - 1 + 1 = -2$

$\therefore x \ll 1 = -2$

$$2^{32} - 1 \quad \{-2 = 2^{32} - 2\}$$

$$\Rightarrow 111 \dots 1 \text{ 32 times}$$

$$111 \dots 10$$

$$2^{32} - 1 - 1 = -2$$

OR

$$-1 \ll 1 \Rightarrow -1 \times 2^1 = -2$$

Right shift

$$x \gg 3$$

$$x \gg 1$$

$$\Rightarrow$$

$$\frac{33}{2^1} = 16$$

x : +ve (Chose leading bit $\rightarrow 1$)

$$x = 0 \dots 100001$$

$$x \gg 1 = 00 \dots 100000 = 16$$

$$x \gg 2 = 0 \dots 100000^x$$

$$000 \dots 1000 = 8$$

$$x \gg 1$$

$$-2/2^1 = -1$$

$$x: 1111 \dots 10^x$$

$$1111 \dots 1 = 2^{32} - 1$$

$$= -1$$

$$-2 \gg 2$$

$$-2/2^2 = 0$$

$$x: 1111 \dots 10^x$$

$$1111 \dots 10 = -1$$

So $(-2 \gg 4) = \{-1, -4 \gg 1\}$

$$1111 \dots 10 = -1$$

Change leading bit $\rightarrow 1$

x : -ve

31 bits 1 + 1 one.

Left Data Page

$2^{32} - 2$

Unsigned Right Shift: \gg

$x = -2$

$x \gg \gg \gg 1$

$x: \underbrace{111 \dots 10}_x$

$x \gg \gg 1: 0 \underbrace{111 \dots 1}$

$= 2^{31} - 1 = 2147483647$

one zero at leading bit. \rightarrow +ve number.

$x = -2 = \underbrace{111 \dots 10}_x$

$x \gg \gg 2 = 00 \underbrace{111 \dots 1}$

$= 2^{30} - 1 = 1073741823$

$\frac{111 \dots 1}{2^{31} - 1}$

$\frac{111 \dots 1}{2^{30} - 1}$

check kth bit is set or not from right side

2p has k=1
Op gen.

$\rightarrow 000 \dots 010 \downarrow$
 $k=1 \text{ pos } n$

n=8 k=2
No

$00 \dots 1000 \downarrow$

n=0 k=3
No

$00000 \dots 000 \downarrow$

$k \leq \text{No. of bit, in Binary representation}$

134

[Left shift 1 with $(k-1)$ do & with $n \Rightarrow$ if non zero = Yes
 else = No.]

$n=5$ $k=3$
 Yes.

$1 = 000 \dots 1$
 $1 \ll (k-1)$
 $00 \dots 100$

$5 \rightarrow 00 \dots 0101$
 $1 \ll (k-1) \rightarrow 00 \dots 0100$

$(1 \ll (k-1)) \& n$

7th bit of $n=5$ set bit

$\neq 0 \Rightarrow$ result is non-zero \Rightarrow Yes.
 $0 \Rightarrow$ result non-zero \Rightarrow No.

Program

m2 -
 left shift

```

if (n & (1 << (k-1)) != 0)
    print("Yes");
else
    print("No");
  
```

Idea: form a new no, whose k th bit from right is 1, else 0
 $\& \text{find}(n \& \text{this no}) == 1$ ret yes.

m2 -
 Right Shift

```

if ((n >> (k-1)) & 1 == 1)
    print("Yes");
else
    print("No");
  
```

[Move k th bit of number to last position & do Bitwise
 & with 1 \Rightarrow if non 1 \Rightarrow set No.
 $0 \Rightarrow$ No.]

Eg $n=13$ $k=3$

$00 \dots 1101$
 $n \gg 2$
 $0000 \dots 11$
 00000001

$if = 1 \cdot$ Yes.

Count set bits.

$n = 5$ 101
o/p = 2

$n = 7$ 111
o/p = 3

$n = 13$ 1101
o/p = 3

$n > 1$ or $n = n/2$
Par each bit & perform &1
 if 1
 c++
 else

```

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

My

return count;

m1 Xlaive : Check if last bit is 1, count it & remove
 TC: $\Theta(\text{Total bits in } n)$ the last bit..

```

{ int res = 0;
  while (n > 0)
  { if ((n & 1) > 0) res++;
    n = n >> 1;
  }
}
  
```

$n = 5$

00 ... 0101	5
a1	
00 ... 010	2
30	
0 ... 01	1
31 bit	
00 ... 0	0
32 bits	

m2 Brian Kernigan's Algorithm.

TC: $\Theta(\text{set bit count})$

$n = 40$

Initial	00 ... 0101000
1st	00 ... 0100000
2nd	00 ... 0000000

Approach ② last set bit turned off

Subtract by 1 =

① All 1s after last set bit → 1
 5

1010000 n=40
 1001111 n-1=39
 5
 1000000 n=32

1000000 n=31
 0111111 n-1=30
 5
 0000000

```
int countbits(int n)
{
    int res = 0;
    while (n > 0)
    {
        n = (n & (n-1));
        res++;
    }
    return res;
}
```

Lookup Table Method

for 32 bit number. tab[i] ⇒ represent count of set in number.
 m3 O(1) int table[256]

Represent of set bit in 0 to 255

Memory

void initialize()

```
{
    table[0] = 0;
    for (int i = 1; i < 256; i++)
        table[i] = (i & 1) + table[i/2];
}
```

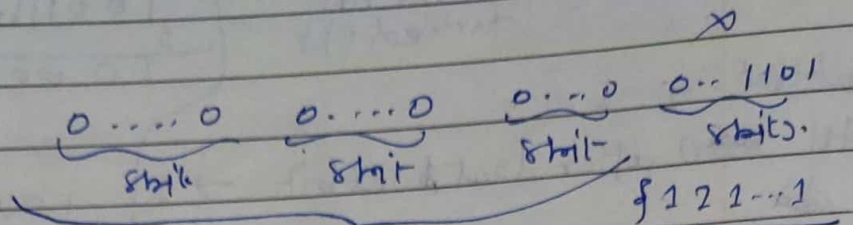
0-255
 0-255-1
 ∴ consider shifts as one unit

```
{
    int count(int n)
    {
        int res = 0;
        while (n > 0)
        {
            res += table[n & 0xFF];
            n >> 8;
        }
        return res;
    }
}
```

Mask rep of 8 set bit



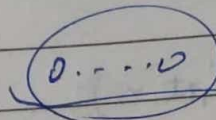
$n = 13$



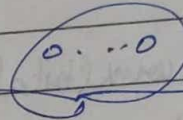
$$\begin{array}{r} 121 \dots 1 \\ 13 \\ \hline \text{rest} = \text{tab}[13] = 3 \end{array}$$

right shift by $n \gg 8$

0...0



4th time



8th time



processing
2nd time

Bitwise Part 1

AND (3)

```
int x = 3, y = 6  
print (x & y)
```

1/2

$x: 0 \dots 0011$
 $y: 0 \dots 0110$

 0011

or (1)

$$\begin{array}{ll} 0 & 0 \\ 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{array} \begin{array}{l} = 0 \\ = 0 \\ = 1 \\ = 1 \end{array}$$

$$(x|y) \equiv 17$$

1110

XOR (^)

$$\begin{array}{rcl} 00 & = & 0 \\ 10 & = & 1 \\ 01 & = & 1 \\ 11 & = & 0 \end{array}$$

$$x^1 y = \underline{\underline{5}}$$

0 1 0 1

Part 2

$$2^{32}-1 = \text{all 1s} = \boxed{1111 \dots 1111}$$

NOT

$$\underline{-2} \neq \sqrt[2]{2-2} \in \mathbb{Z}^{32-1-1} \in \mathbb{N} = 111 \dots 10$$

Java -ve nos. are stored in 2's complement

$$-x = 2^{32} - x$$

Range of int = -2^{31} to $2^{31}-1$

(2)

$x = 5$


$$\begin{array}{r} 26 = 000 \dots 0101 \\ 27 = 111 \dots 1010 \end{array}$$

$2^{32} = 1.$

what did i subtracted from $2^{32}-1$ it ~~is~~ $10 = 5$

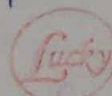
$$\sim x: 2^{32} - 1 - 5_2 \quad 2^{31} - 6 \Rightarrow -6$$

$$\approx \sqrt{2} - 6$$



$$x \ll y = x \times 2^y$$

$$11.000 \ll 1$$



Date: / /
Page:

Left shift \ll (Multiplication)

$$x = \overset{x}{000 \dots 0011} \quad x = 3$$

$$x \ll 1: \underline{000 \dots 0110} \quad \underline{6} \quad 3 \times 2^1$$

~~$x \ll 2$~~ :

$$x = \overset{x}{000 \dots 0011}$$

$x \ll 2$

$$\underline{000 \dots 01100}$$

$$\underline{12}$$

$$3 \times 2^2$$

$x \ll 4$

$$0000 \underline{110000}$$

32 16 8 4 2 1

$$\underline{48} \quad (32 + 16) \quad 3 \times 2^4 = 48$$

~~$$x = 1: 2^{32} - 1 \quad 1111 \dots 10 \quad x \ll 1: 00 \dots 01$$

$$x \ll 1: 1111 \dots 10 \quad \neg x: 1111 \dots 10$$~~

~~$x = -1$~~ $x = -1$, $x \ll 1 = 2$

$$x = -1 \Rightarrow 2^{32} - 1 \Rightarrow \overset{x}{1111 \dots 1}$$

$$\begin{aligned} & \checkmark \\ & 1111 \dots 10 \Rightarrow 2^{32} - 1 - 1 \\ & \Rightarrow 2^{32} - 2 = \underline{\underline{-2}} \end{aligned}$$

→ +ve: filling leading bits with 0

→ -ve: —————

Signed
Right shift

$$x: \quad 00 \dots 100001 \quad 33$$

$$\bullet \quad x \gg 1: \quad \underline{0} \quad 00 \dots 010000 \quad 6$$

$$x: \quad 00 \dots 100001 \quad xx$$

$$\bullet \quad x \gg 2: \quad 0000 \dots 10000 \quad 8$$

$$\bullet \quad x = -2 \Rightarrow 2^{32} - 2 \Rightarrow \begin{array}{c} 11111 \dots 10 \\ \downarrow \\ \underline{11111 \dots 1} \Rightarrow 2^{32} - 1 \\ \Rightarrow -1 \end{array}$$

$$x \gg 2 = ? \quad -1$$

$$x: \quad 11111 \dots 110 \quad xx$$

$$\underline{11111 \dots 1} \Rightarrow 2^{32} - 1 = -1$$

-2 >> y { y any value } = -1

Unsigned Right Shift >>> [+ve → both filling with 0
-ve → leading bits]

$$x = -2, \quad x \ggg 1 = ?$$

$$x = -2 = 2^{32} - 2 \Rightarrow \begin{array}{c} 1111 \dots 10 \\ \downarrow \\ \underline{01111 \dots 1} = 2^{31} - 1 \end{array}$$

$$x \ggg 2 = ?$$

$$x: \quad 1111 \dots 10 \quad xx$$

$$\underline{001111 \dots 1} = 2^{30} - 1$$