AGENDA

Bitwise operators and Properties

Left and Right shift operator concepts

Prob
- Check i^{th} bit
- Count set bits
- Unset i^{th} bit
- Set bits in range

(9 - 10:30 PM)

Adv Contest 1 → 13 oct Friday

Syllabus → Arrays 1, 2, 3

Bit Manipulations

10:30
Discussion

Friday 6 Oct → Normal lecture

Bits → 0 1

unset set

Bitwise operators (&, |, ^, ~, <<, >>)

atleast 1 set bit

A	B	A & B	A | B	A ^ B	~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

same
same
pubby
shame

Basic AND Properties

```
int A = 10
int B = 4
int C = A & B
```

C = 0

10 → 1010

4 → & 0100

0 ← 0000

1) A & 1

A 0th bit is set → 1

0th bit is unset → 0

```
A → ---------1
& 1 → 00000001
    00000001
```

```
A → ---------0
& 1 → 00000001
    00000000
```

odd

A → 6 5 4 3 2 1 0

A → _ _ _ _ _ _ 1 → odd no.

2^0

A → 10 → 0 0 0 0 1 0 1 0 → even

A → 3 → 0 0 0 0 0 0 1 1 → odd

A → 7 → 0 0 0 0 0 1 1 1 → odd

0^{th} bit is 1 → odd

0^{th} bit is 0 → even

A & 1

1 → 0^{th} bit is set → odd

0 → 0^{th} bit is unset → even

2) A & 0 = 0

A = _ _ _ _ _ _

& 0 = 0 0 0 0 0 0

0 0 0 0 0 0

3) A & A = A

1 & 1 = 1

A → 0 1 0 1 1

& A → 0 1 0 1 1

0 1 0 1 1

Basic OR properties

$0 \mid 0 = 0$

otherwise

$1 = 1$

1. $A \mid 0 = A$

$1 \mid 1 = 1$

otherwise $= 0$

$A = 1\ 0\ 1\ 0\ 1$

$1 \quad 0 = 0\ 0\ 0\ 0\ 0$

$A \rightarrow 1\ 0\ 1\ 0\ 1$

2. $A \mid A = A$

$A = 1\ 0\ 1\ 0\ 1$

$1 \quad A = 1\ 0\ 1\ 0\ 1$

$1\ 0\ 1\ 0\ 1$

Basic XOR Properties

1. $A \wedge 0 = A$

$A \rightarrow 1\ 0\ 1\ 0\ 1$

$\wedge\ 0 \rightarrow 0\ 0\ 0\ 0\ 0$

$1\ 0\ 1\ 0\ 1$

bit

$x \wedge 0 =$

x	
1	$1 \wedge 0 = 1$
0	$0 \wedge 0 = 0$

$A \rightarrow 0\ 1\ 0\ 1\ 1\ 0$

$\wedge\ 0 \rightarrow 0\ 0\ 0\ 0\ 0\ 0$

$0\ 1\ 0\ 1\ 1\ 0$

2. $A \wedge A = 0$

$A \rightarrow 1\ 0\ 1\ 0\ 1$

$\wedge \quad A \rightarrow 1\ 0\ 1\ 0\ 1$

$0\ 0\ 0\ 0\ 0$

Commutative Property

AND, OR, XOR

$5+3=3+5$

↓

Order of operands does not change answer

$2-3=3-2$

$A \& B = B \& A$

$A \mid B = B \mid A$

$A \wedge B = B \wedge A$

Associative Property

$(2+5)+7$

$7+7$

$=14$

↓

grouping of operands does not change answer

$A \& B \& C$

$2+(5+7)$

$2+12$

14

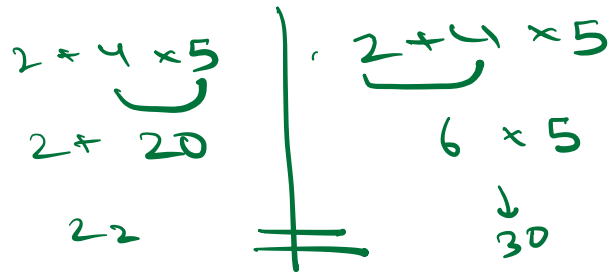
$(A \& B) \& C = A \& (B \& C)$

$A \mid B \mid C$

$(A \mid B) \mid C = A \mid (B \mid C)$

$A \wedge B \wedge C$

$(A \wedge B) \wedge C = A \wedge (B \wedge C)$



Evaluate expression a ^ b ^ a ^ d ^ b

a ^ b ^ a ^ d ^ b

a ^ a ^ b ^ d ^ b

a ^ a ^ b ^ b ^ d

0 b 0 d

ans → d

Evaluate expression 1 ^ 3 ^ 5 ^ 3 ^ 2 ^ 1 ^ 5

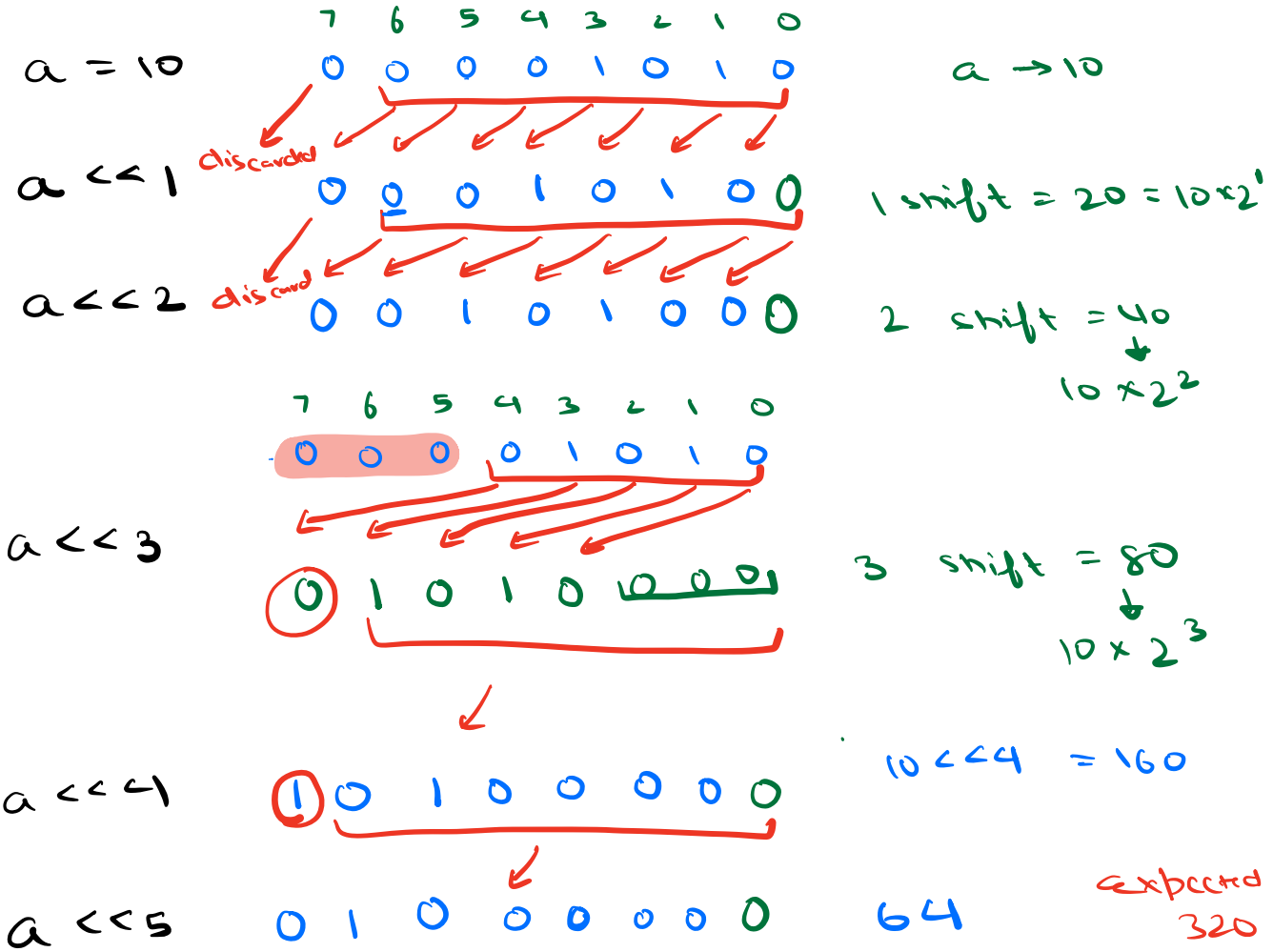
1 ^ 1 ^ 3 ^ 3 ^ 5 ^ 5 ^ 2

0 ^ 0 ^ 0 ^ 2

ans = 2

Left shift (<<)

8 bit no



Overflow → bigger value cannot be stored in smaller data type memory space

10 << 5 → 320
10 << 6 → 640
} bij no. won't fit in 8 bits

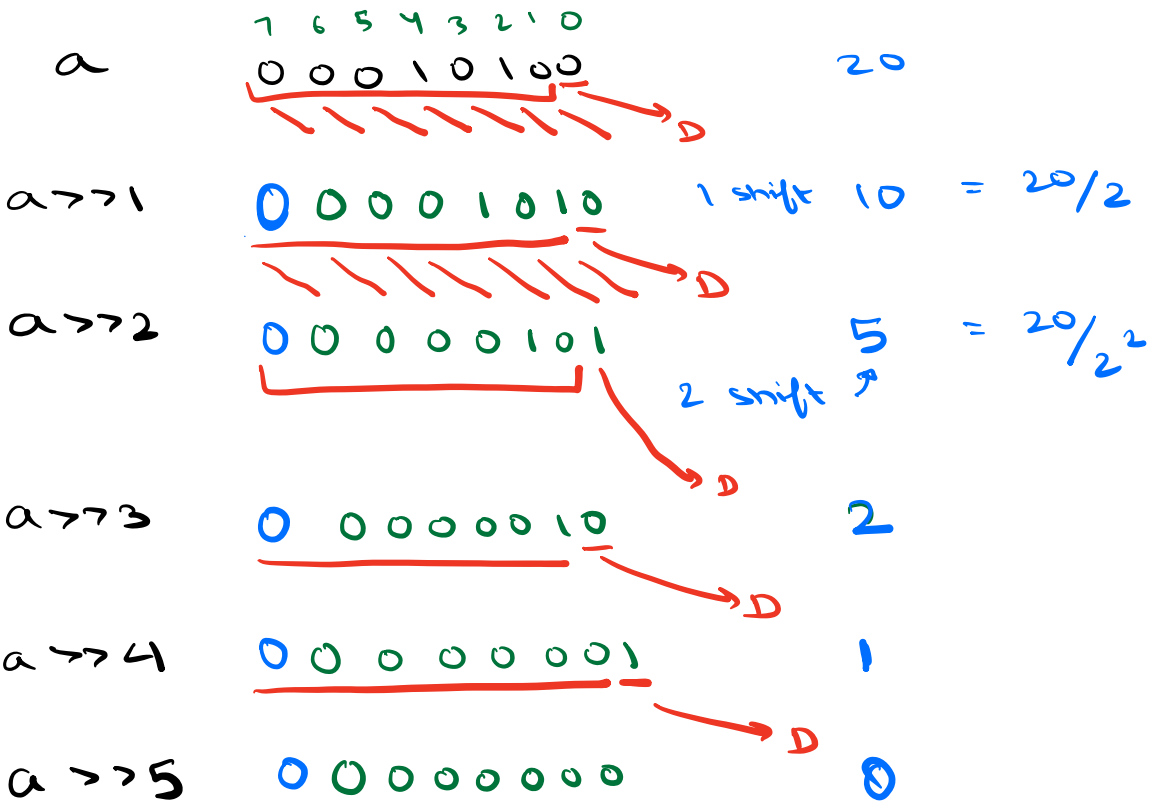
$10 << i = 10 \times 2^i$

$$a << n = a \times 2^n$$

$1 << n = 2^n$

$2^4 \rightarrow 1 << 4$

Right shift (>>)



Overflow does not occur

$$A >> N = \frac{A}{2^N}$$

$$1 >> N = \frac{1}{2^N}$$

$1 << 3$

$$1 << n = 2^n$$

$$1 << 3 = 2^3 = 8$$

---

10 : 29

	7	6	5	4	3	2	1	0	
1	0	0	0	0	0	0	0	1	set bit → 0th pos
1<<3	0	0	0	0	1	0	0	0	set bit → 3rd pos

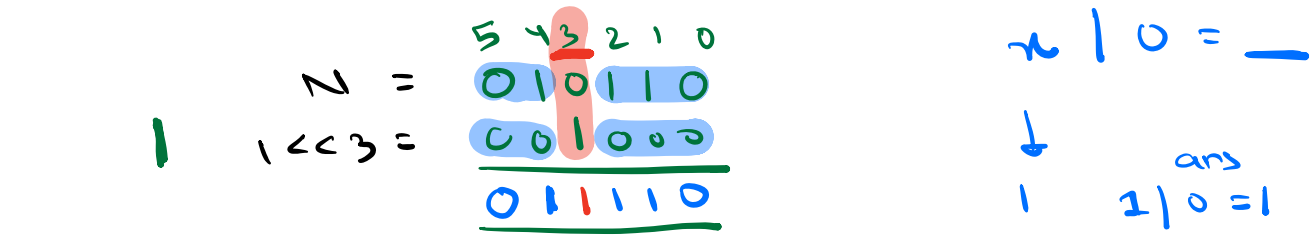
	7	6	5	4	3	2	1	0
1	0	0	0	0	0	0	0	1

	7	...	i	...	2	1	0
1<<i	0	0	0	1	0	0	0

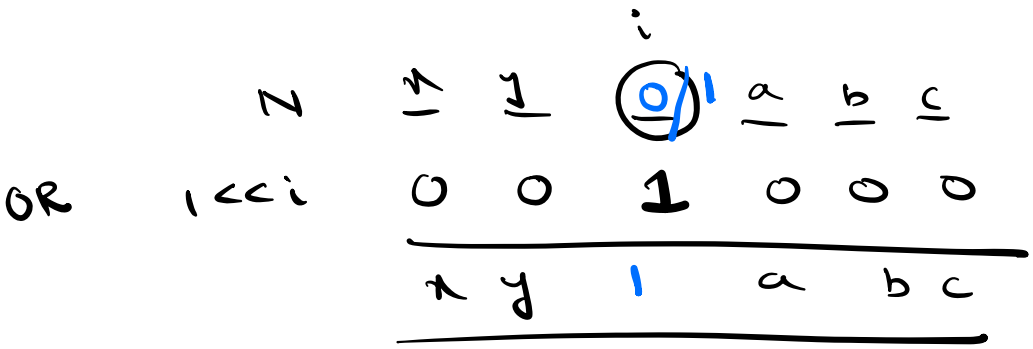
Set ith bit

i = 3

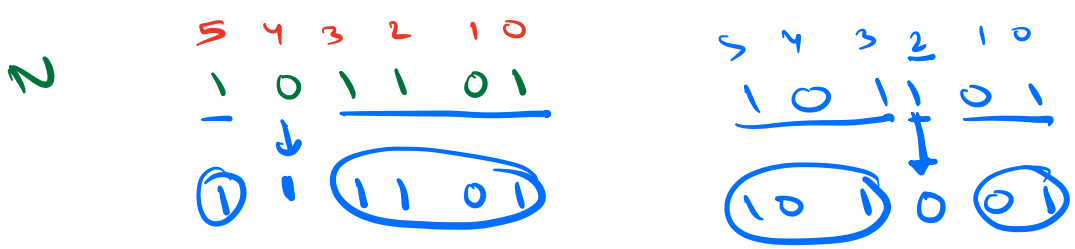
N = N | (1<<i)

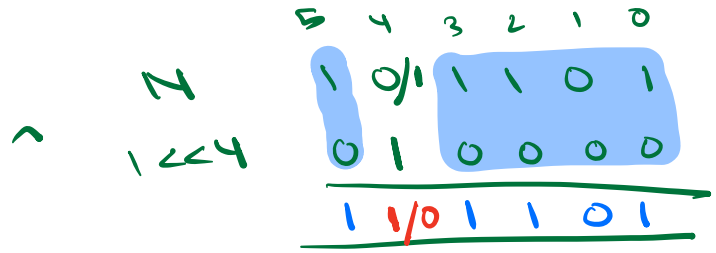


(a) bit | 0 = bit



Toggle ith bit in N





i = 4

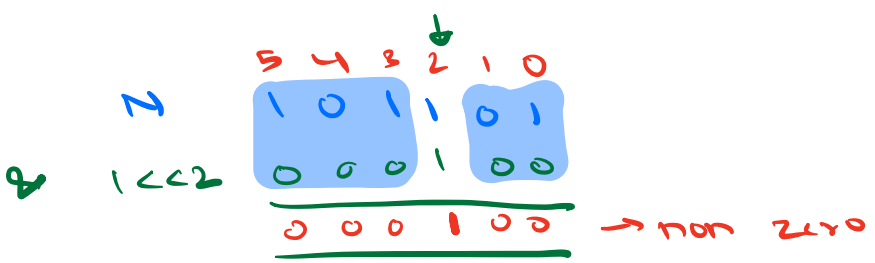
bit ^ 0.

bit	ans
1	1
0	0

(a) bit ^ 0 = bit

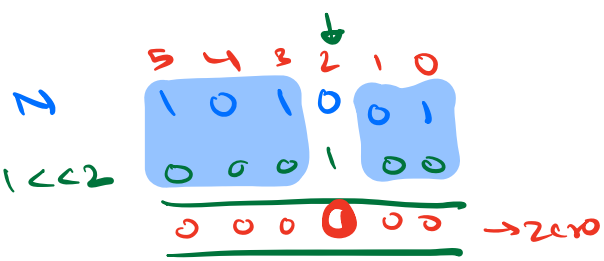
$N = N \wedge (1 << i)$ Toggle ith bit in N

Check whether ith bit is set or not in N



N = 1 0 1 0
& 1 = 0 0 0 1

i = 2 → True
i = 4 → False



if ans is 0, then ith bit is unset

ans is non 0, then ith bit is set

$ans = N \ \& \ (1 << i)$

Set ith bit (OR)

$N = N \mid (1 << i)$

Toggle ith bit (XOR)

$N = N \wedge (1 << i)$

Check if ith bit is set (AND)

ans = N & (1<<i)

0 → bit unset

≠0 → bit set

1. Check if ith bit is set or not

N = 3

i = 4 false

i = 1 true

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

```
fn checkbit (N, i) {
    if (N & (1<<i) ==0)
        return false
    else
        return true
}
```

TC : O(1)

SC : O(1)

```
           i
N      _ _ _ 0 _ _
& 1<<i  0 0 0 1 0 0
       ------------
        0 0 0   0 0
```

2. Given an integer N, count no. of set bits in N.

N = 12 00001100 cnt 2

int N → 4B → 32 bits

31 . . . . . 2 1 0

N — — — — — —

BF: Go to every bit pos, check whether bit is 1. If it is 1, count it

```
int countbit (int N) {
    int cnt = 0
    for (i = 0 ; i < 32 ; i++) {
        // ith bit pos → check if set
        if (checkbit (N, i))
            cnt++
    }
    return cnt
}
```

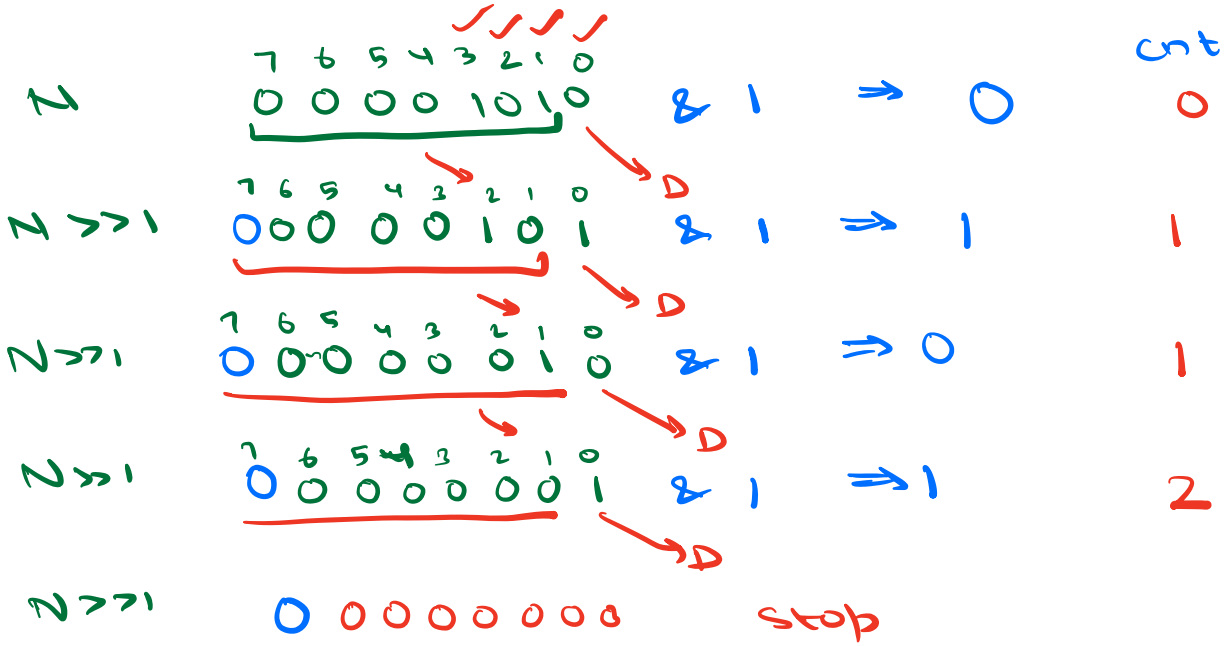
32 itr

TC : O(1)

SC : O(1)

Approach 2

N = 10




```
int countbit (int N) {
    int cnt = 0
    while (N > 0) {
        if (N & 1 == 1)
            cnt ++
        N = N >>1
        N = N/2
    }
}
```

TC : O(log N)

10 → 5 → 2 → 1 → 0

N → N/2 → N/4 → ... 0

log N steps

3. Unset ith bit

N = 12
i = 2
ans = 8

3 2 1 0
1 1 0 0
1 0 0 0

N = 12
i = 1
ans = 12

3 2 1 0
1 1 0 0
1 1 0 0

Check ith bit is set or not

Toggle ith bit

X

N = N ^ (1<<i)

ans = N & (1<<i)

```
fn unsetbit (N, i) {
    if (checkbit(N, i)) {
        N = N ^ (1<<i)
    }
}
```

true → ith bit is set

TC : O(1)
SC : O(1)

4. Make a binary no. which has B 1s followed by C 0s

B = 3 C = 2

11100 → ans 28

---

7 6 5 4 3 2 1 0 B = 3 C = 2

0 0 0 1 1 1 0 0

7 6 5 4 3 2 1 0 B = 4 C = 3

0 1 1 1 1 0 0 0

First bit → C bit pos

Last bit → C + B - 1 bit pos

cnt = B

i —— j

j - i + 1 = B

C → j

j - C + 1 = B ⇒ j = B + C - 1

```
int num=0
for (i=C ; i <= C+B-1 ; i++) {
    num = num | (1<<i)
}
```

B iter

TC : O(B)

SC : O(1)

Doubts

A = 123 → 132

1, 2, 3

132 ✓

321 ✓

213 ✓

231 ✓

312 ✓

123

Vector <pair <int, int>> v

```
v.push_back(<1,3>)
v.push_back(<4,6>)
```

Interval

1, 3

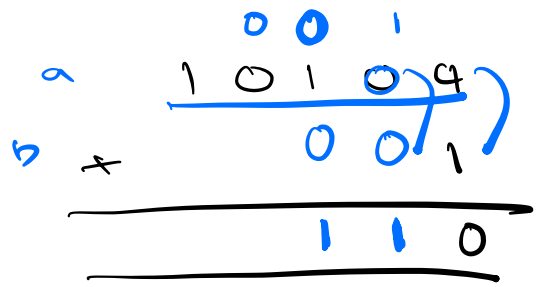
4, 6

v[0].first → 1

v[0].second → 3

Sum = 9+1

= 10



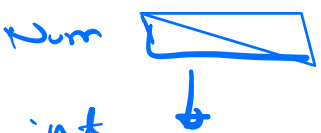
d = sum % 10

c = sum / 10

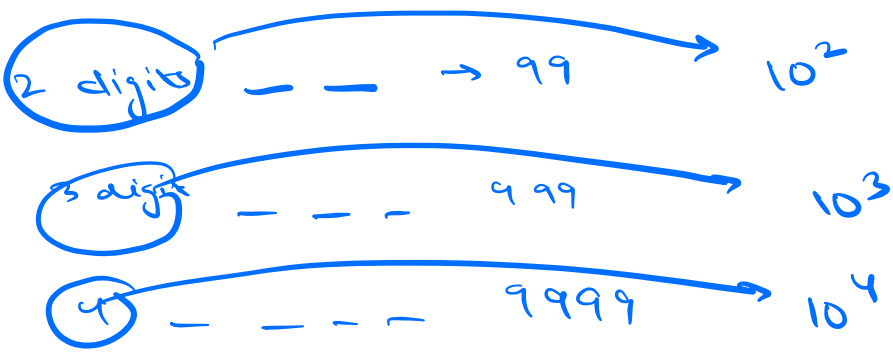
sum = dig of a + dig of b + c

Num = 122 +1

sum = 123



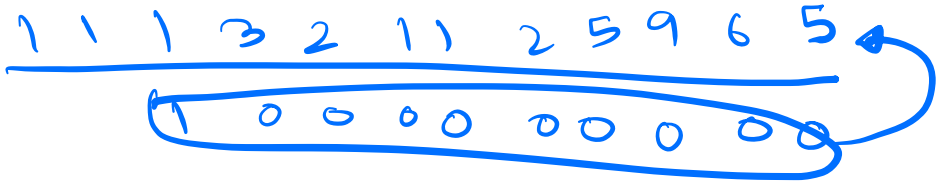
ans



A = _ _ _ _ _ _ _ _ _ _

len 100

10^{100}



int

10^9