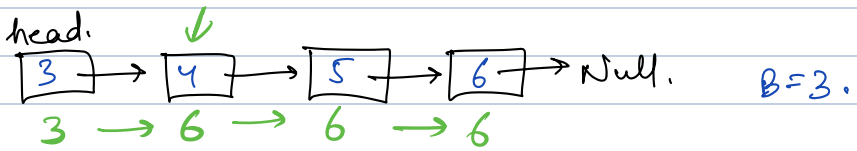
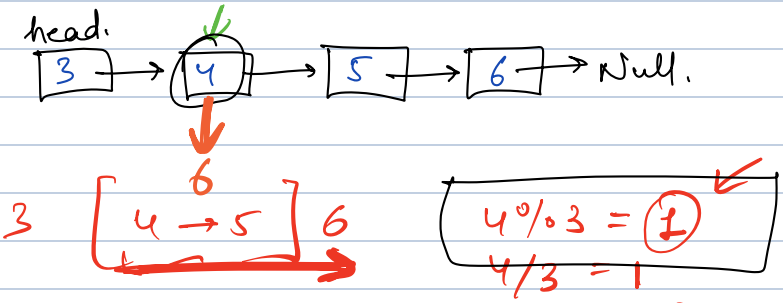
class starts at 9:05

Q1. Bhidu & Traversal.

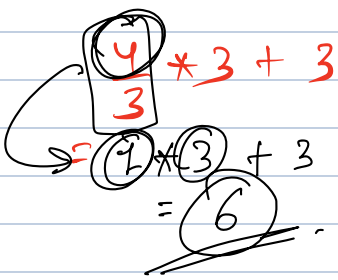
Linked List → A
Integer → B.

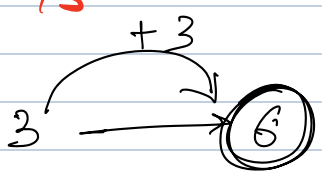


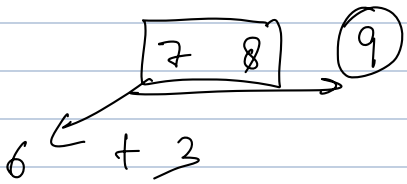
Change the value of each node to its nearest multiple of B.

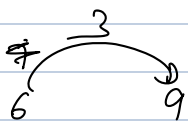


N/3







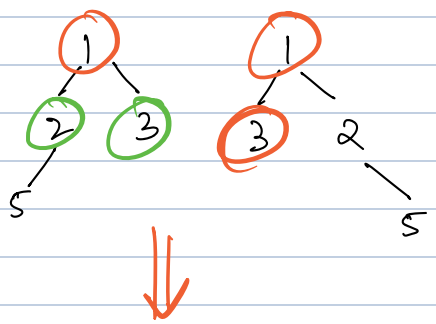


*3 + 3 =

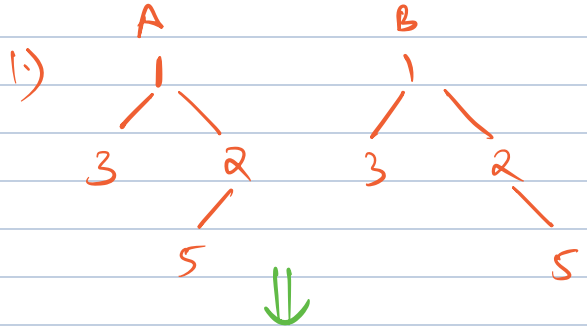
2*3 + 3 = 9.

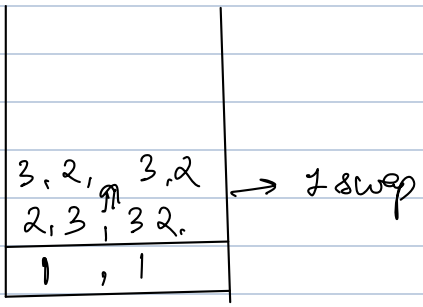
(val % N) + val.

Q2. A B → identical.

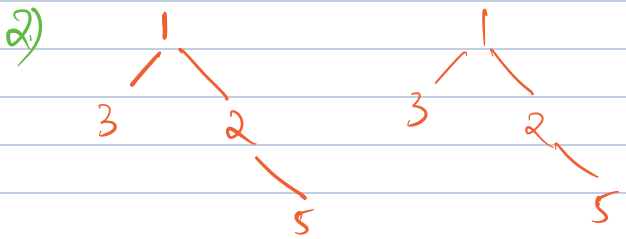


1) Try to make it identical by swapping left & right of a node → min swaps?
2) If Not possible, return -1.





→ 2 swap



1:) of

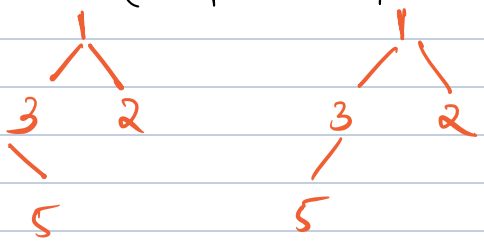
1:) Recursion

Null (3)

```
int solve (Node A, B )
1:) if (A==Null && B==Null) {
        return 0.
    if (A==Null || B==Null)
        return -1;
    if (A.val != B.val)
        return -1;

int temp1 = solve (A.left , B.left);
int temp2 = solve (A.right, B.right);

    if (temp1 != -1 && temp2 != -1)
        return. temp1 + temp2;

int temp3 = solve (A.left , B.right);
int temp4 = solve (A.right, B.left );

    if (temp3 != -1 && temp4 != -1)
        return. (temp3 + temp4 + 1)
```

TC → O(N)
O(A+B)

SC → log(N) → h.
h → height



Q Candies & Queries.

A → | 0 | 0 | 1 | 2 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | (indices 0 1 2 3 4 5 6 7 8 9 10) ✓

M

L	R
3	4 → ✓
2	4 ✓✓
	✓
	✓

C → Queries: | 1 | 4 | 0 |

Queries

return the no of children having candies atleast 1: C[i]

Brute force.

1.) for all M rows, add +1 from L to R in Array
2.) for all Queries in C, traverse A & find count

TC → $O[(M * A) + |Q| * A]$

Optimisation –

Initially All elements in A = 0.

1.) For L^{th} index → do(+1)
2.) For $(R^{th}+1)$ index → do(−1)

L → 2 ✓
R → 4 ✓

3 → 4

1	2	3	4	5	6
0	+1	1	0	-1	0

prefix sum.

0	1	2	2	0	0

sort

TC → for M times → $O(M) * 1$

Queries: | 1 | 3 | 5 | 2 |

0 0 0 1 2 2

search for 1 or just greater than 1

Q → | 1 | 3 | 5 | 2 |

1	2	4	4	5	6

TC → $Q * \log_2(A)$

Total TC = $O(M) + O(A) + O(Q * \log_2 A) + A \log A$

prefix sum — sorting.

SC → $O(A)$

Q Little Pony & String period.

String A. → Rearrange its letters such that the period of string is min & return min possible period.
↳ character in a period

A → "abccba"

a → 2
b → 2
c → 2

abc abc

⇓

3 char ⇒ o/p = 3

A → "abccbz"

a → 1
b → 2
c → 2
z → 1

abccbz

↳ o/p → 6.

abccccab →

K.

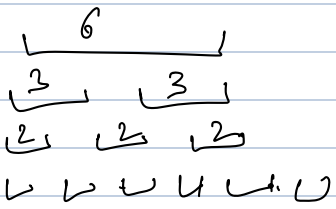
c_1 → f1
c_2 → f2
c_3 → f3

K group

K.

f1 → divisible by k.
f2, f3 → divisible by k.

f1 K.
6 → 1
6 → 2
6 → 3
6 → 6



a → 6
b → 6
c → 6

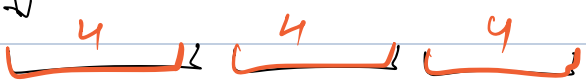
abc abc abc abc abc abc → k = 6
abbaccabc abbaccabc k = 2

f1 f2 f3 — — fn

greatest common divisor ? GCD / HCF.

Note → k to be a factor of f1 f2 — fn, & the greast factor ie GCD.
↓
K.

A = " abcbczcbb " → 12



length of A / k = no of chars in one group

A → abcabcz

a → 2
b → 2
c → 2
z → 1

GCD = 1 = K

length of string. → N.

TC → O(N) + O[log(num)]
↓ min
O(N)

5 15

5 | 15 | 3
15
0

10 15

10) 15 (1
10
5 10 (2
10
0

Q. A & B, find X such that

1.) A XOR X is min
2.) X should have set bits = B.

Ex → A = 3 X = ?
B = 3

3 set bits
A ^ X = min.

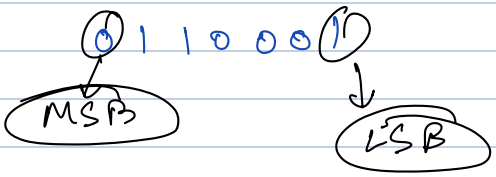
X = 3 3 ^ 3 = 0.
A = 3
0 | 1 1

X = 7 ✓ 3 set bits
3 ^ 7 = 4.

Ex. A = 15 X → 2 set bits
B = 2. X = 12.
15 ^ 12 = 3

Intuition.

1.) Setting MSB of integer increases the value of integer.



2.)

A	B	XOR
1	1	0
0	0	0
1	0	1
0	1	1

So, we will try to make i^{th} bit of X same as i^{th} bit of A.

Approach.

Start making X same as A [to minimise A ^ X = 0].

A = 0 1 1 0 1 1

X = 0 1 1 0 1 1

0 0 0 0 0 0

Case 1 B = 2. ie B ≤ set Bits in A.

MSB ←

A = 0 1 1 0 1 1

X = 0 0 1 0 0 0 ←

A ^ X 0 1 0 0 0 0 ✓

0 0 0 0 1 1 ✓ ✓

→ 3

Case 2 a) B = 5 > set Bits in A.

A = 0 1 1 0 1 1 ←

X = 0 1 1 1 0 1 1

A ^ X 0 0 0 1 0 0

b) B = 8

32 bits

[0 0 0 0 ... 0 1 1 0 1 1] → A = 0 1 1 0 1 1

X = 0 1 1 1 0 1 1

0 1 1 0 1 1

0 0 0 0 ...

A = 0 1 1 0 1 1

1 1 0 1 1 1 1

6 1's.

TC → $O(\log A) + B \cong 32$.

$32 + 32 \rightarrow O(1)$.