CS & IT ENGINEERING

Data structure & Programming

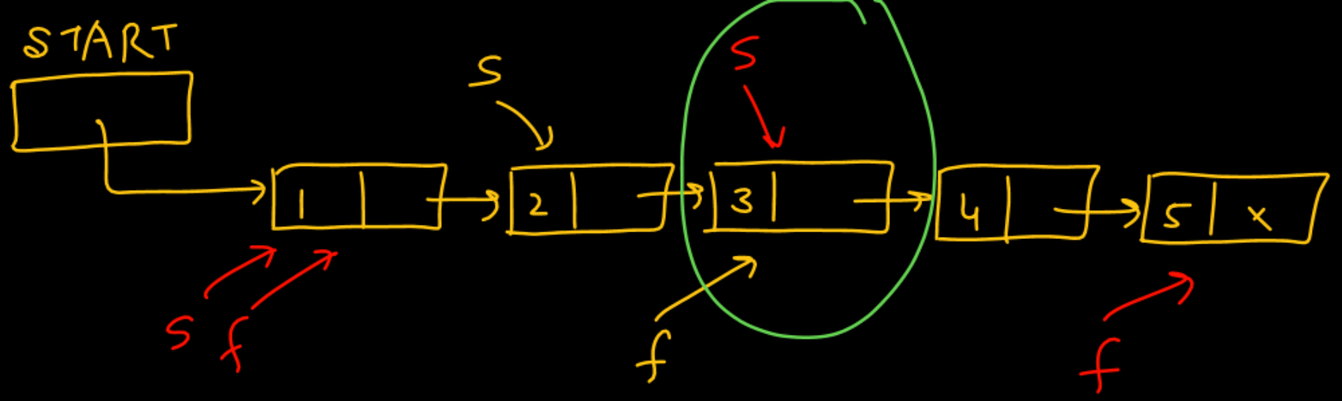
Linked List

Lec- 04

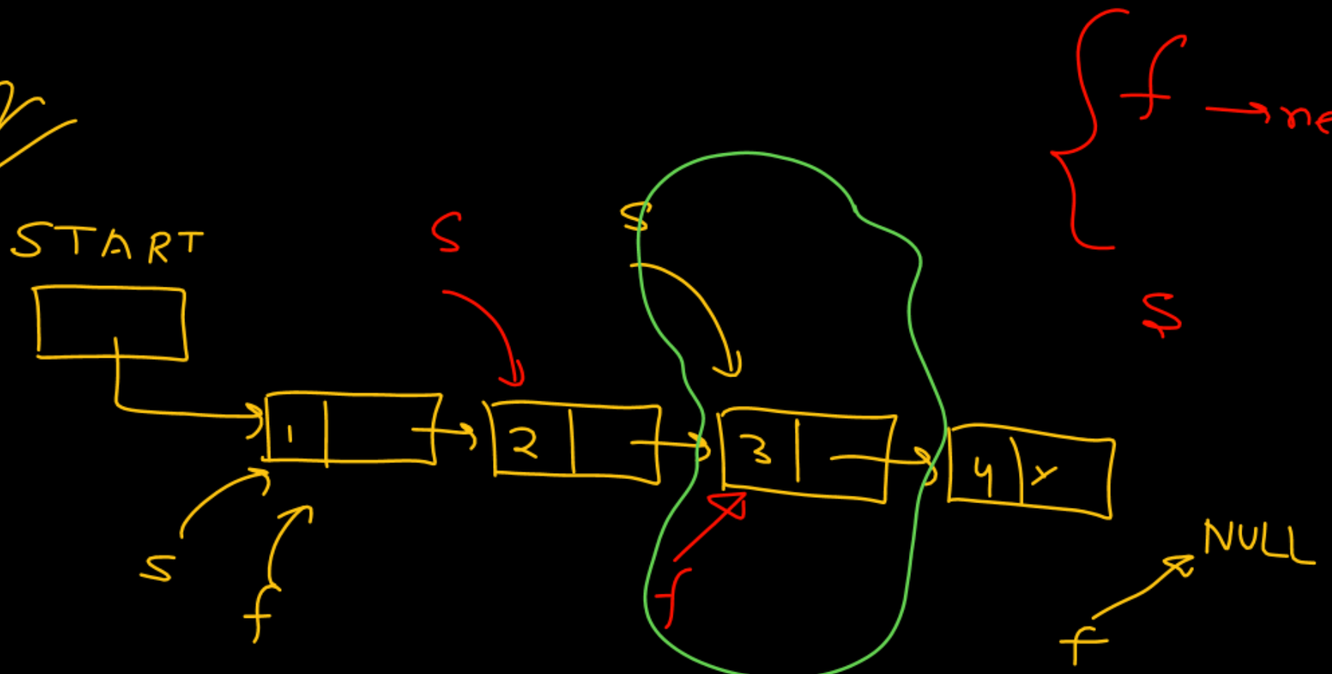
By- Pankaj Sharma sir

Linked List -4

Case1



Case2

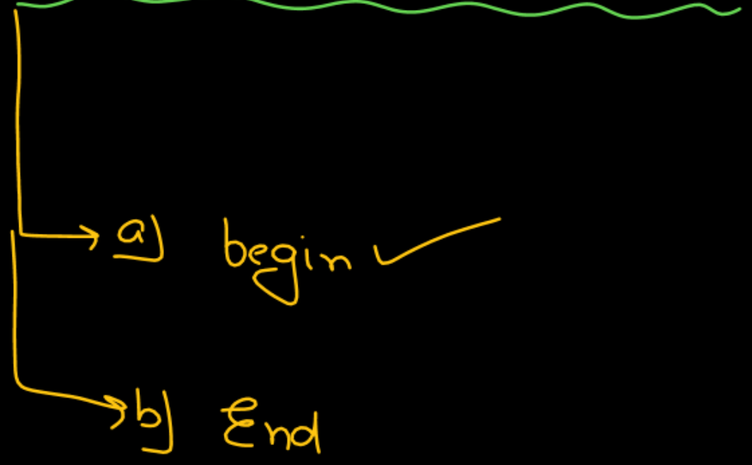


f → next == NULL
stop

s

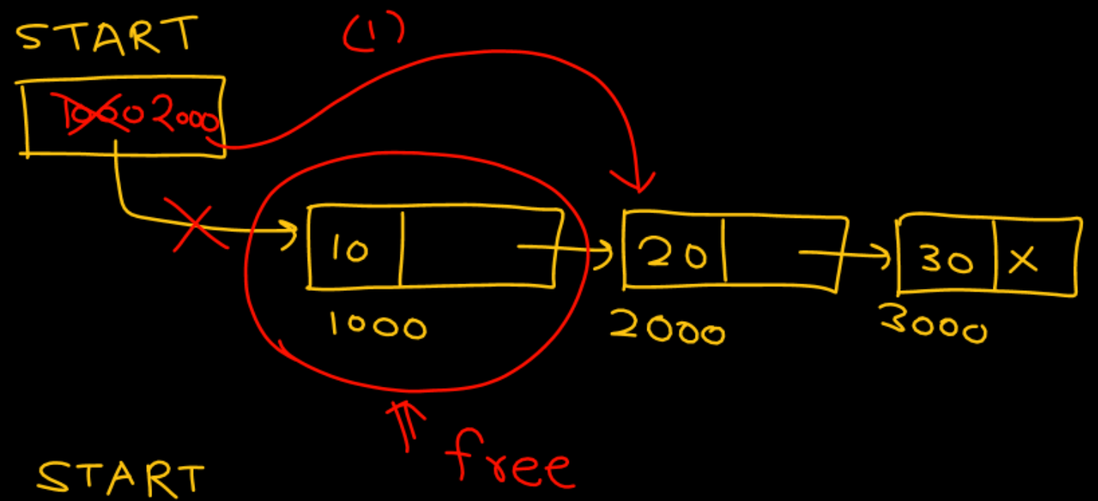
```
while( fast != NULL && fast->next != NULL) {
    slow = slow->next;
    fast = fast->next->next;
}
```

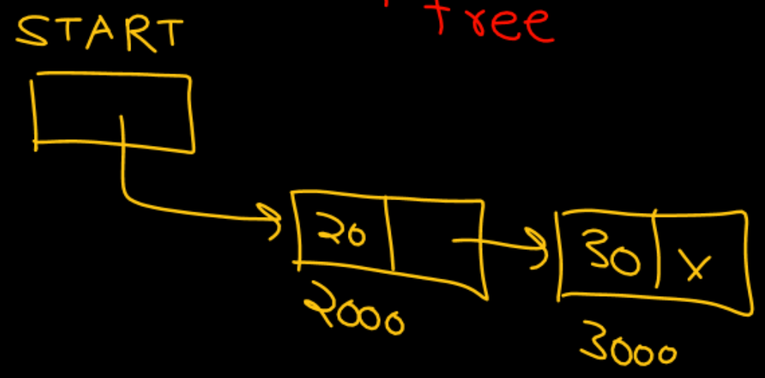
Deletion in linked list



① START → NULL → X

If l.l is Empty ⇒ do nothing

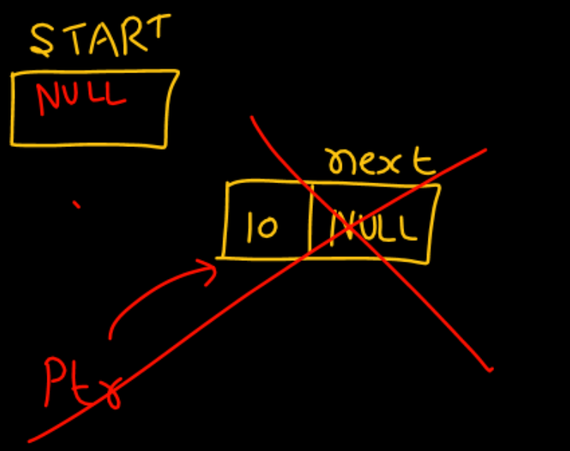
② 



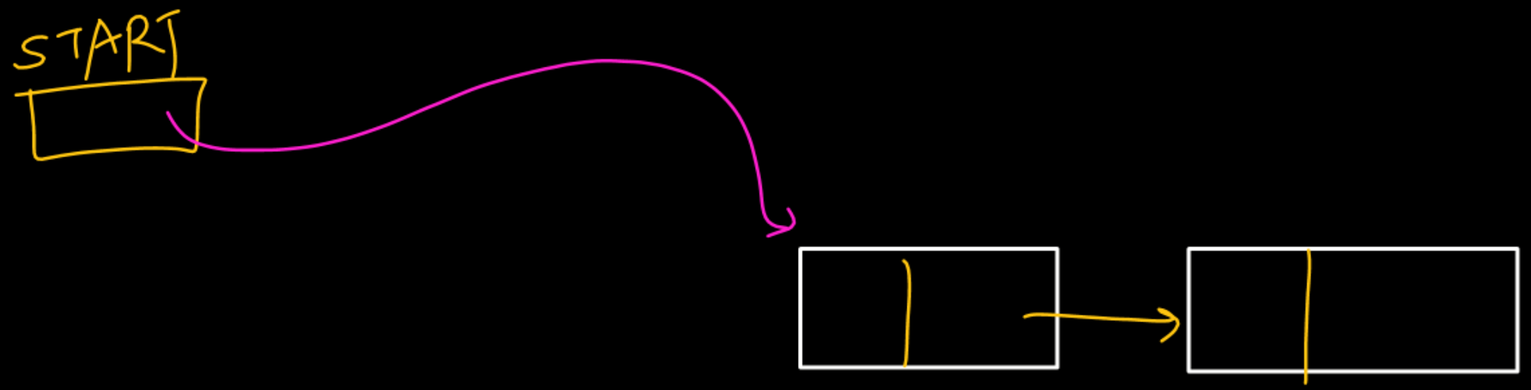
After deletion of 1st node
START ⇒ must point 2nd node.

```
if (START == NULL)
        return ;
Ptr = START ;
START = START -> next ;
free (Ptr)
```

Case 1:



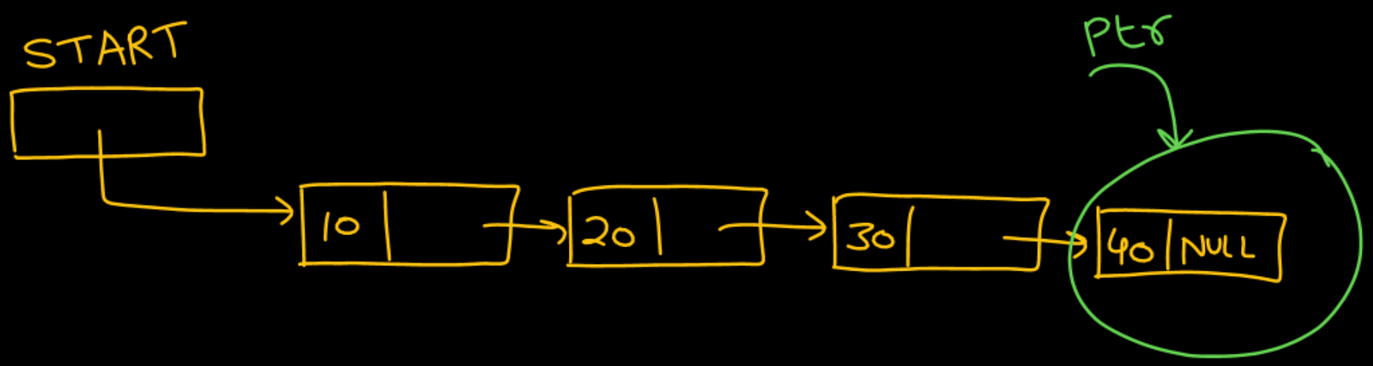
START = NULL



START → next

START = START → next ;

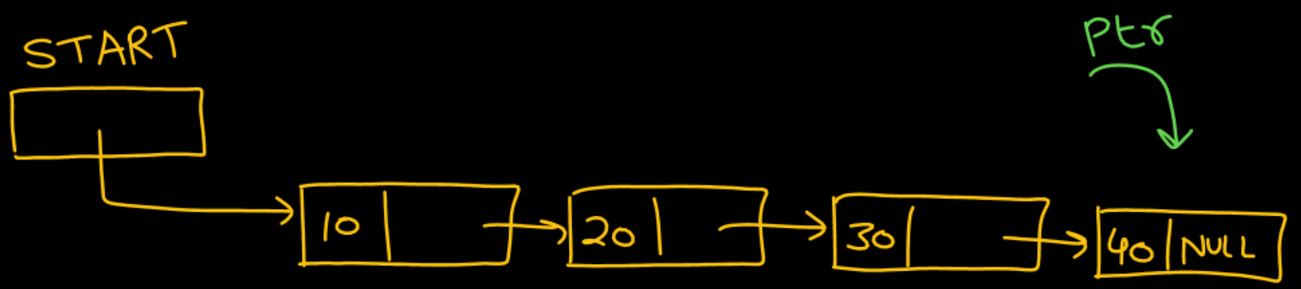
Deletion from End



1. Traverse

2nd last node ka next field ⇒ NULL

Deletion from End



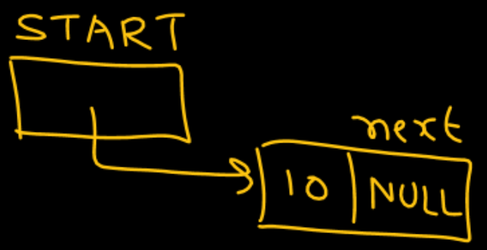
If ~~case1~~, ~~case2~~ fails

Ensure
⇒ Atleast 2 nodes

Case 1:

⇒

Case 2:

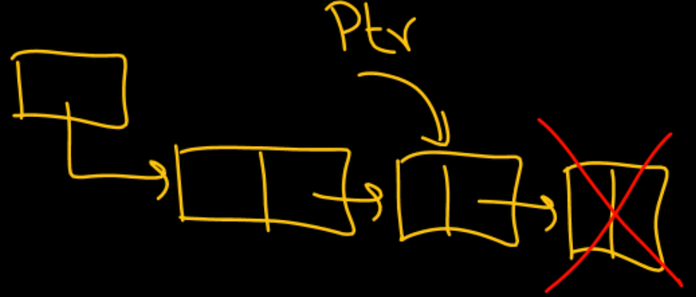


⇒ START = NULL

free

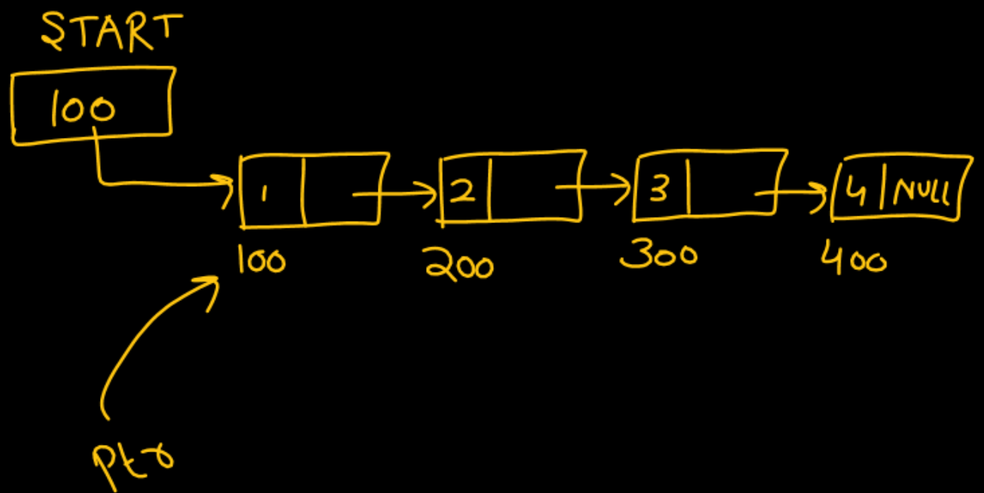
return

```
Ptr = START;
if (START == NULL)
    return;
else if (START->next == NULL)
    {
        START = NULL;
        free(Ptr);
        return;
    }
while(Ptr->next->next != NULL)
    Ptr = Ptr->next;
free(Ptr->next);
Ptr->next = NULL;
```



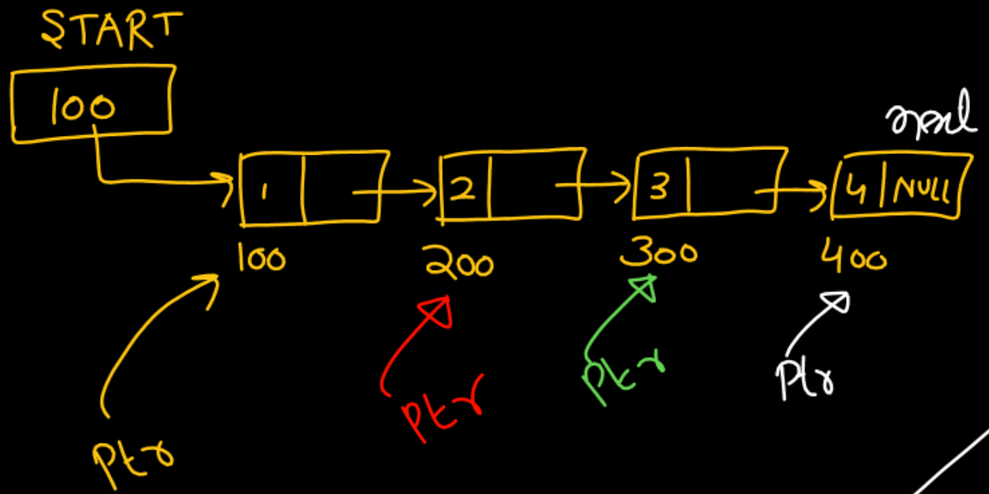
Reverse

1. Printing
2. actual l.l. reverse

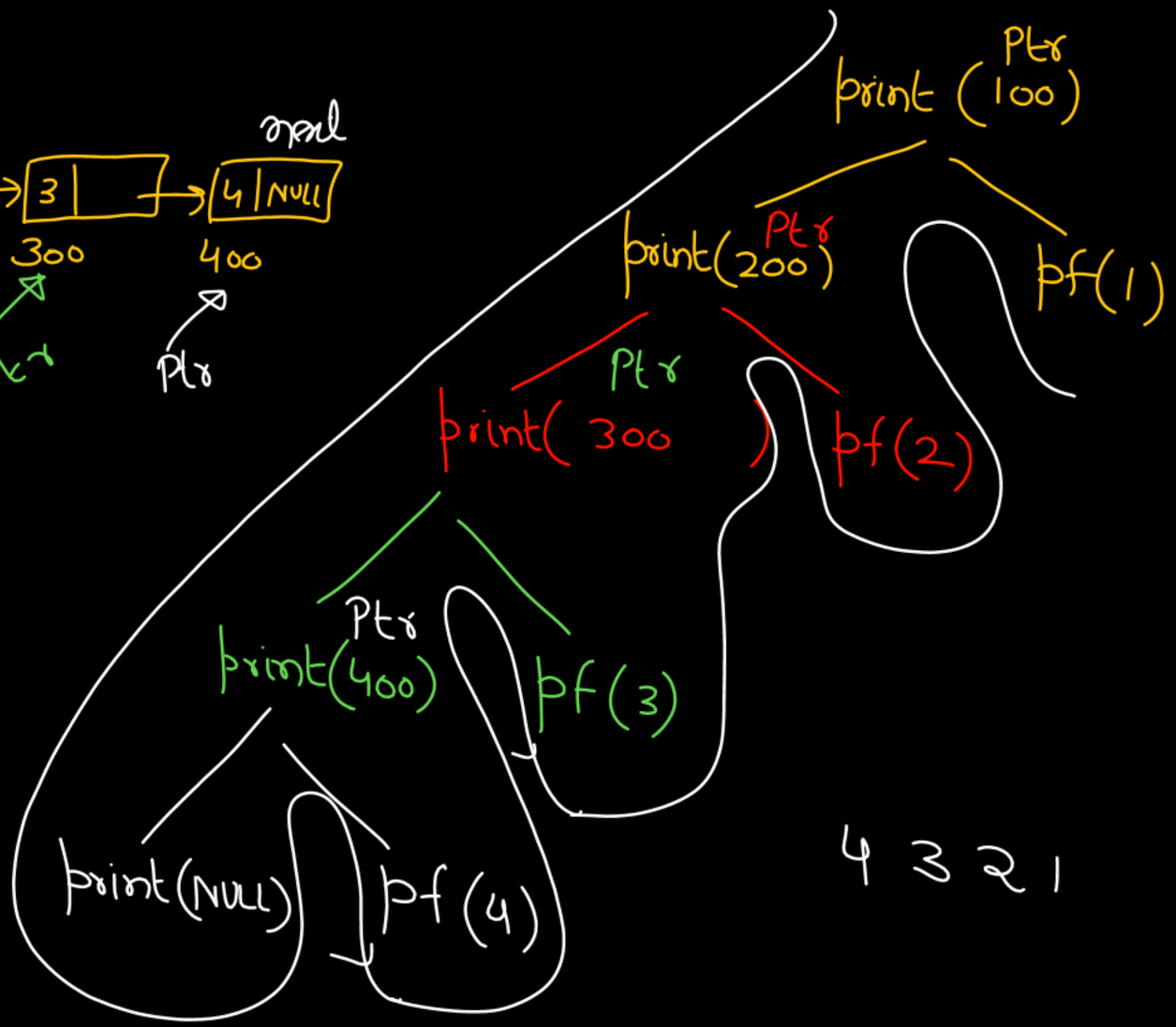


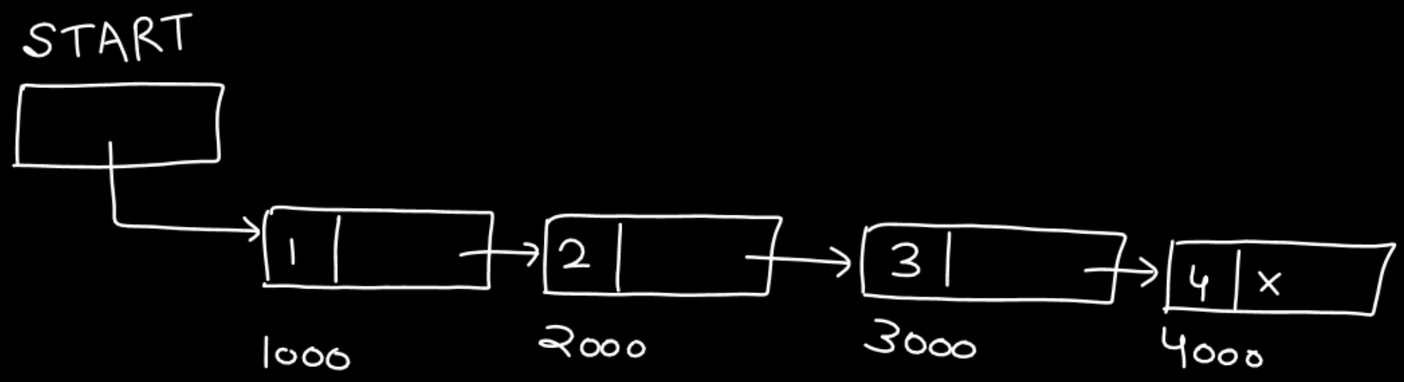

```
main( )
 {

 print( START )

 }
```

```
Void print ( struct Node *Ptr )
 {

   if (Ptr) {

 print( Ptr -> next );
 pf ("%d", Ptr -> data);
   }

 }
```




```
Void print ( struct Node *Ptr )
    {
        if(Ptr) {
        print( Ptr->next );
        pf ("%d", Ptr->data);
            }
    }
```

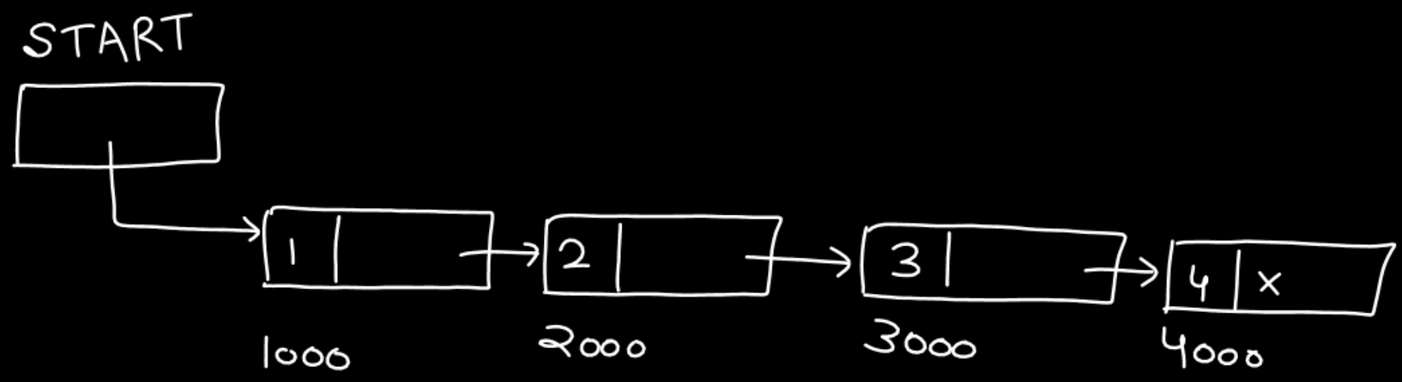


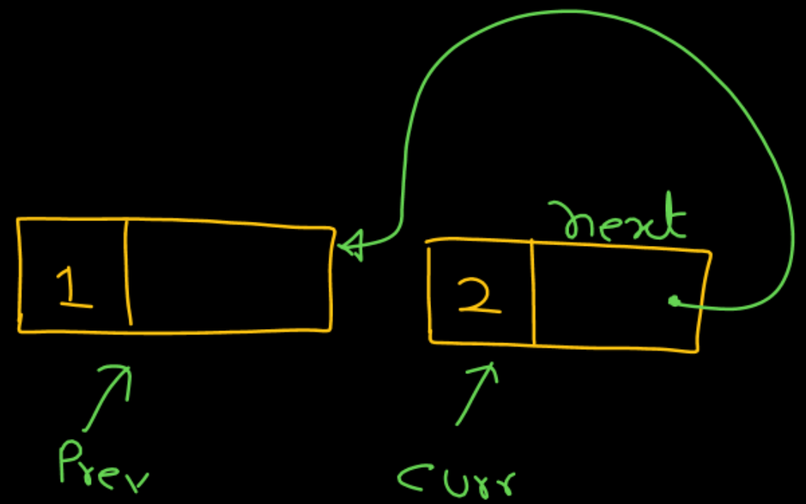


case 1 : L.L is empty → 0 node

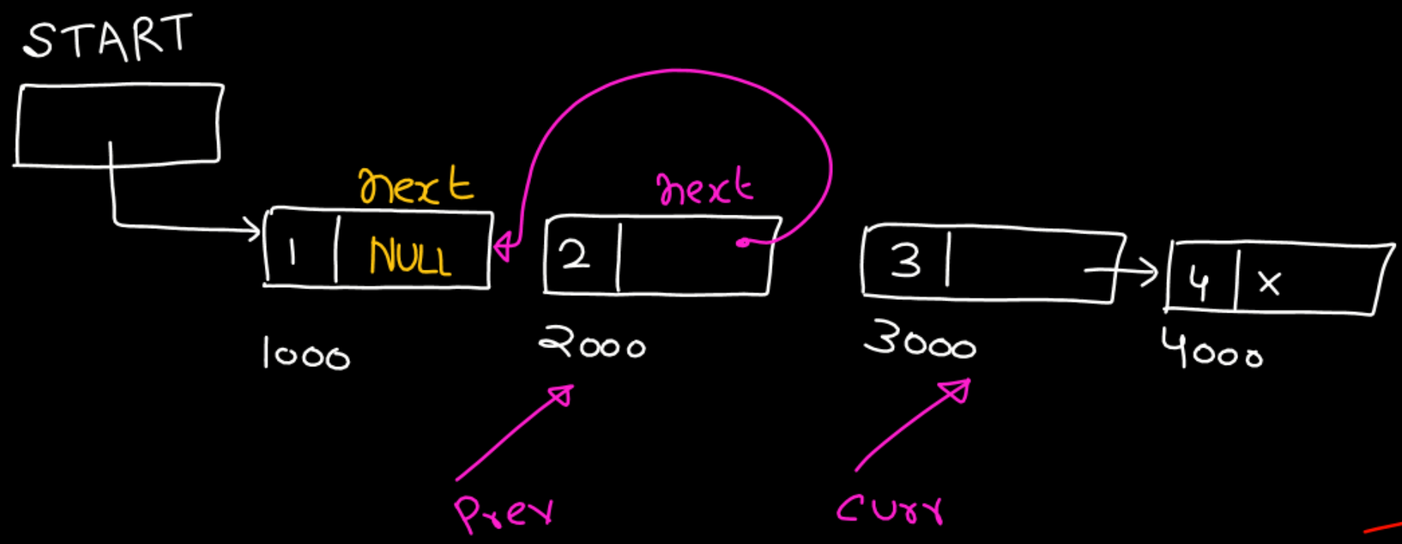
case 2 : 1 node → 1 node

at least 2 nodes exist

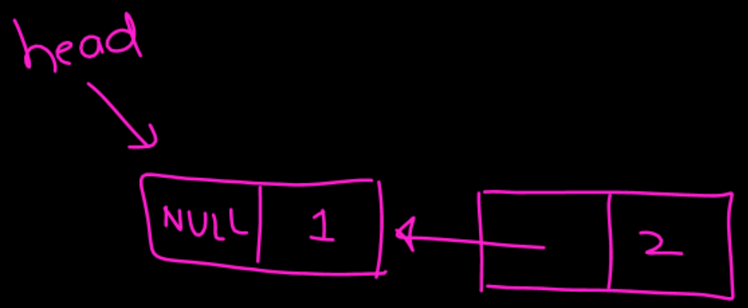




Curr → next = Prev



Save next node before rev.

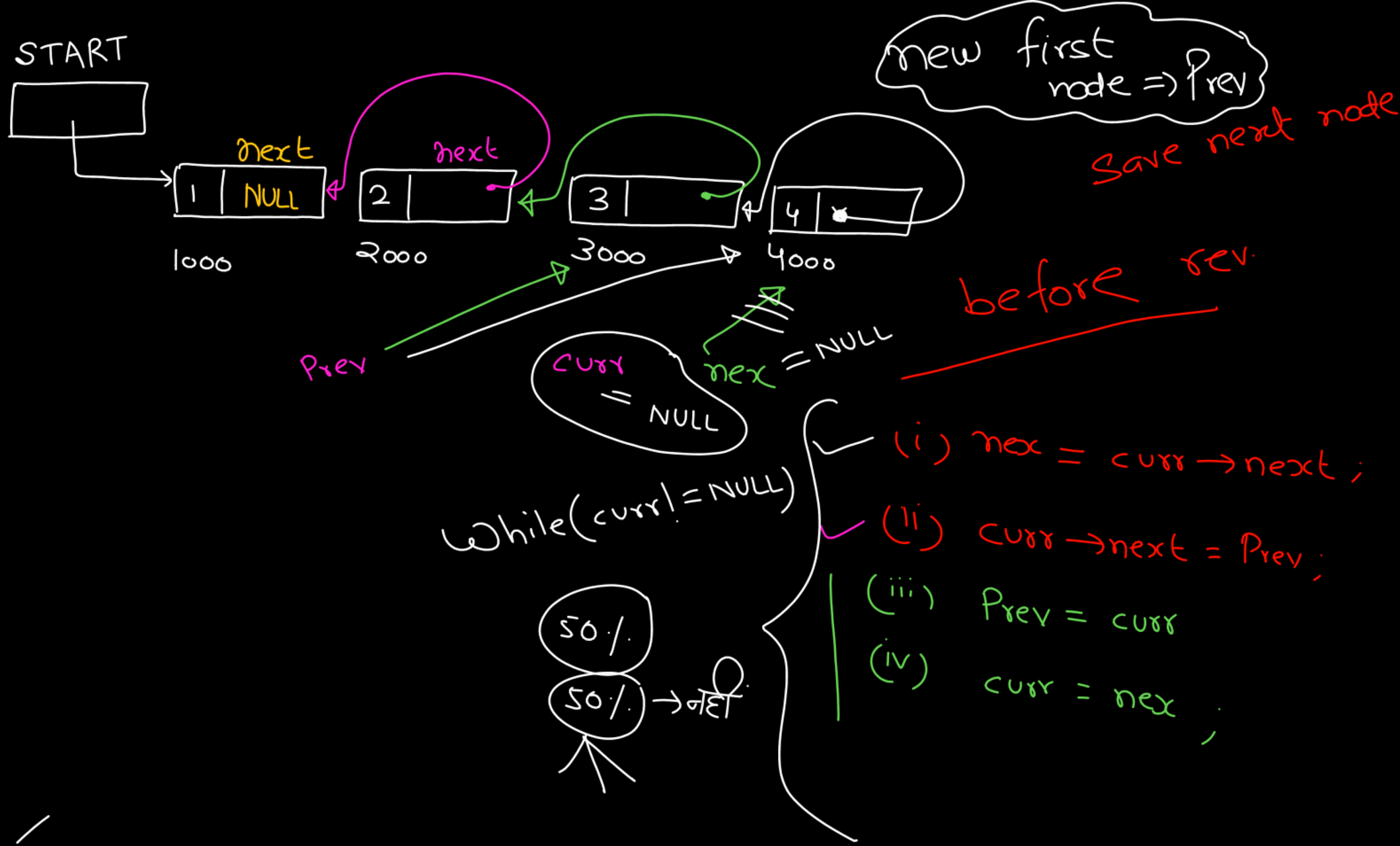


(i) `nex = curr->next;`

(ii) `curr->next = Prev;`

(iii) `Prev = curr`

(iv) `curr = nex;`

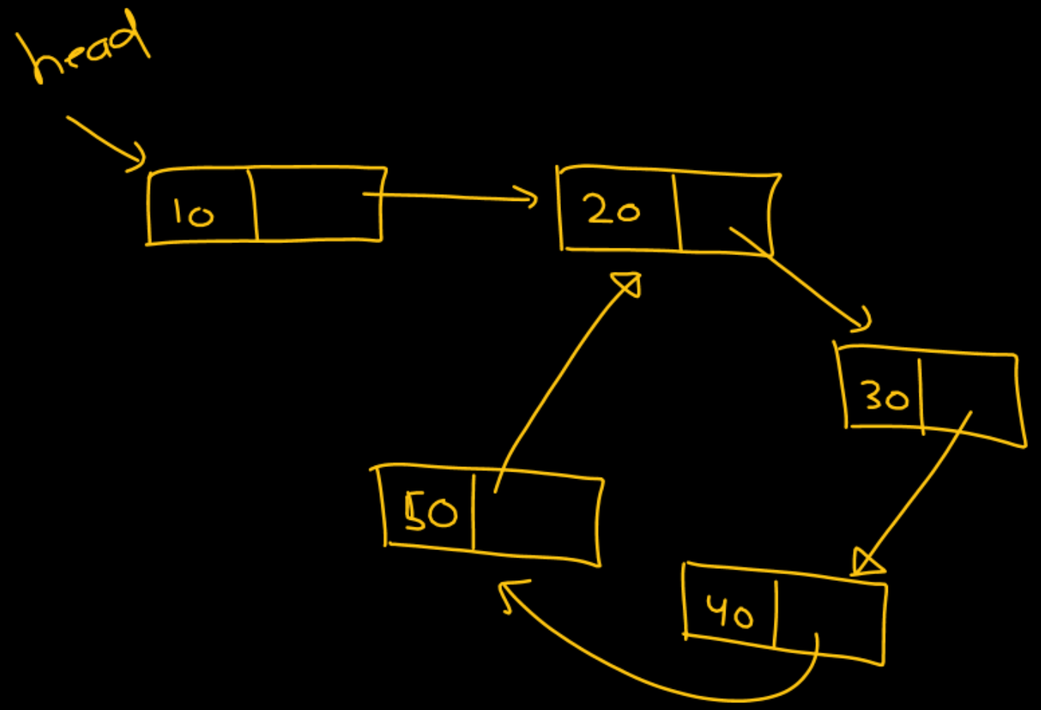

START
next
next
1
NULL
2
3
4
1000
2000
3000
4000
new first node => Prev
Save next node
Prev
curr = NULL
nex = NULL
before rev.
while(curr != NULL)
(i) nex = curr → next ;
(ii) curr → next = Prev ;
(iii) Prev = curr
(iv) curr = nex ;
50%
50% → नहीं

comment → Yes
comment → No

4 → Today
5 → More problem
6 → types of L.L
7 → PYQs

Given a linked list, find whether there is a loop in the l.l. or not.

Yes/1

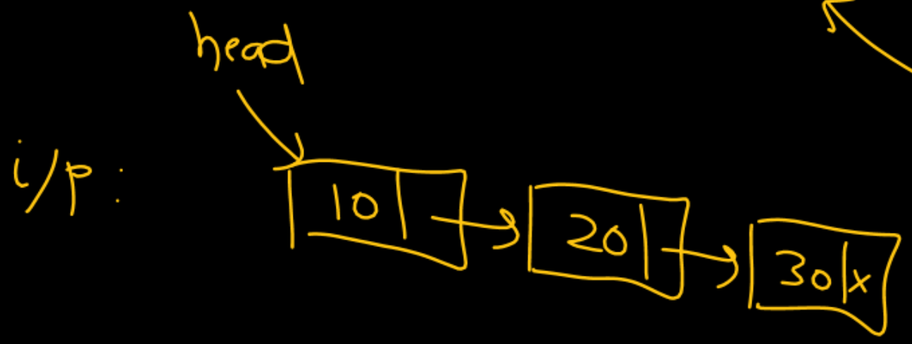


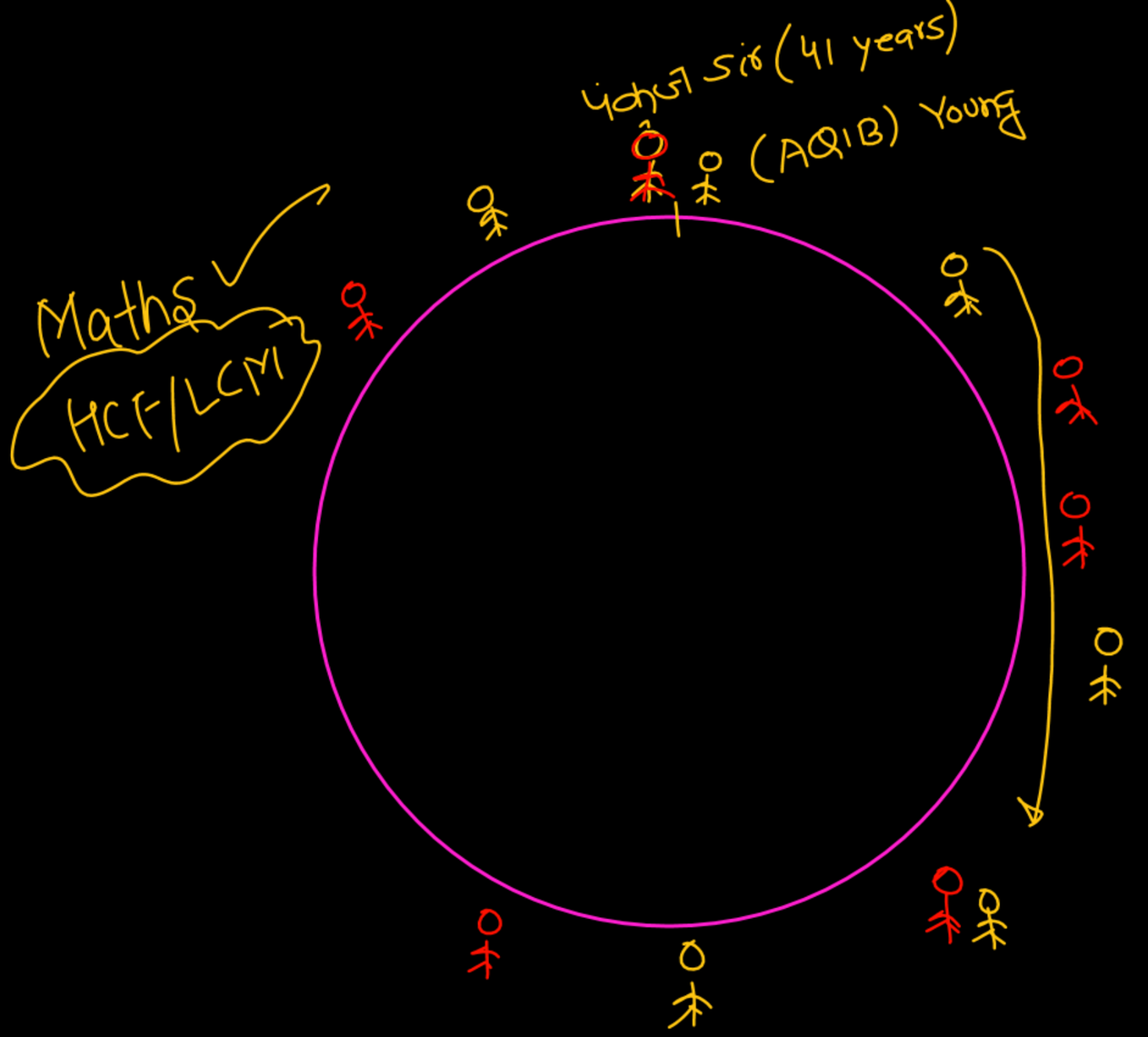
idea : ?

```
for(i=1; i<=n; i++)
{
    for(j=1; j<=n; j++)
}
```

O(n2)

CS/CC Y

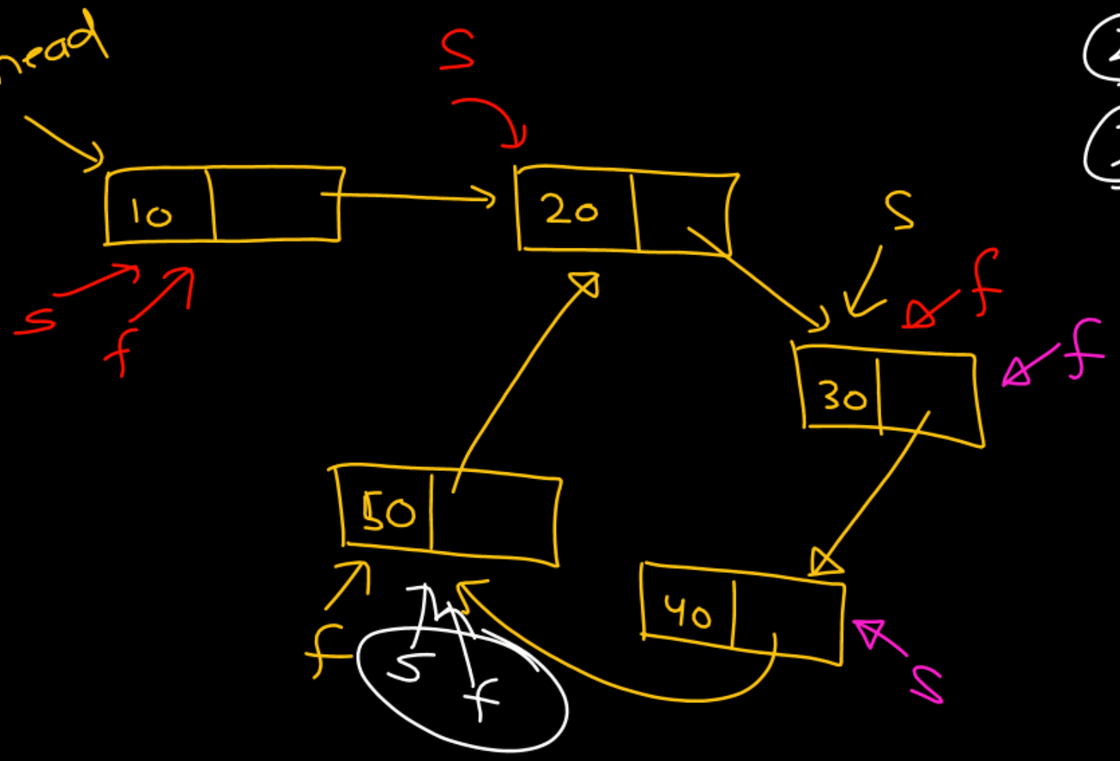



Yahya Sir (41 years)
(AQIB) Young
Maths
HCF/LCM

Given a linked list, find whether there is a loop in the l.l. or not.

1. Code
2. Special case
3. What if no loop (How to Handle)

i/p:

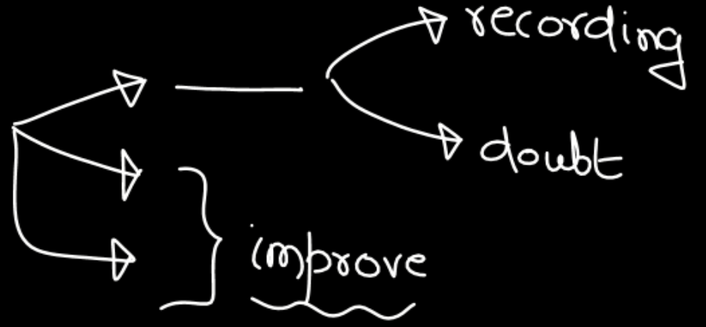


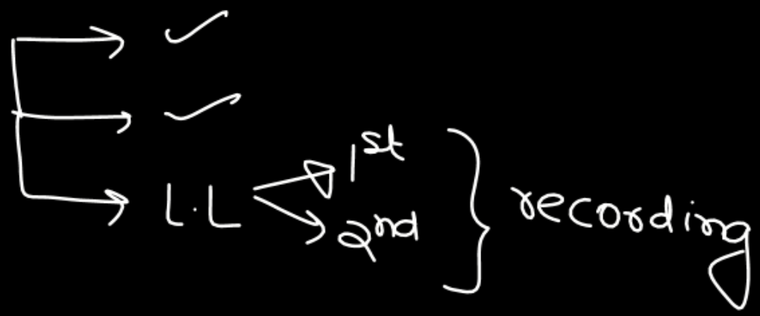
Review

Wed : 08 : 30

① ✓

② Corrective action






THANK YOU GW
SOLDIERS !