

## Linked Lists, July - 20

20 July 2020 18:47

Insertion at nth position ✓

Delete a LL

Delete a node from beginning of a LL

Delete a node from the end of a LL

Delete a node from the nth position of a LL

Find the middle element of a LL

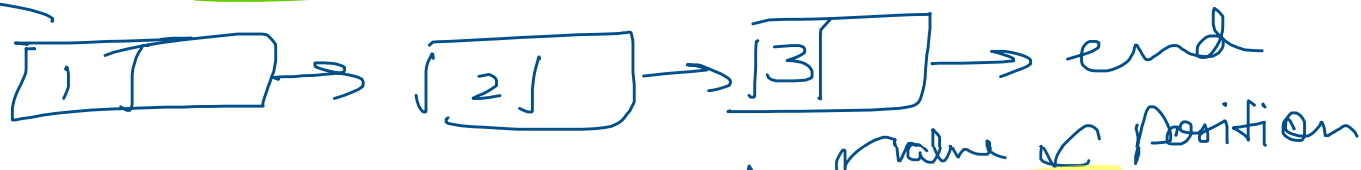
Reverse a Linked List

← H.W

### 3 Insertion at nth position

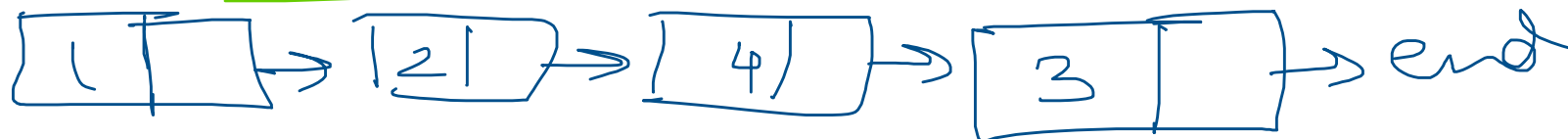
head  
↑

what we have

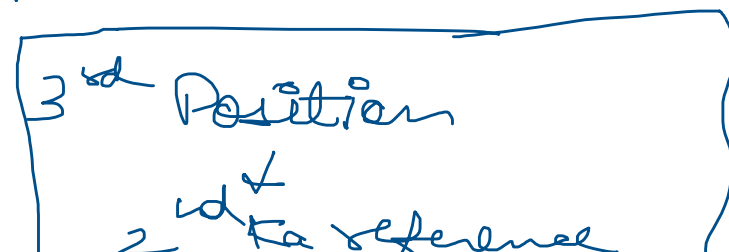


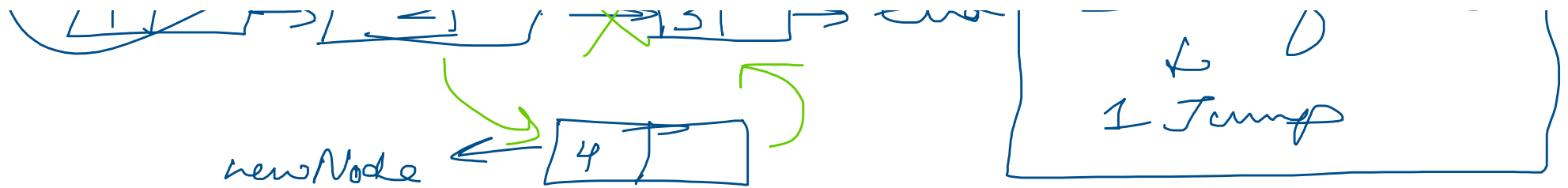
insert at Pos ( 4 , 3 )

what we want



Steps





$newNode.next = temp.next;$   
 $temp.next = newNode;$

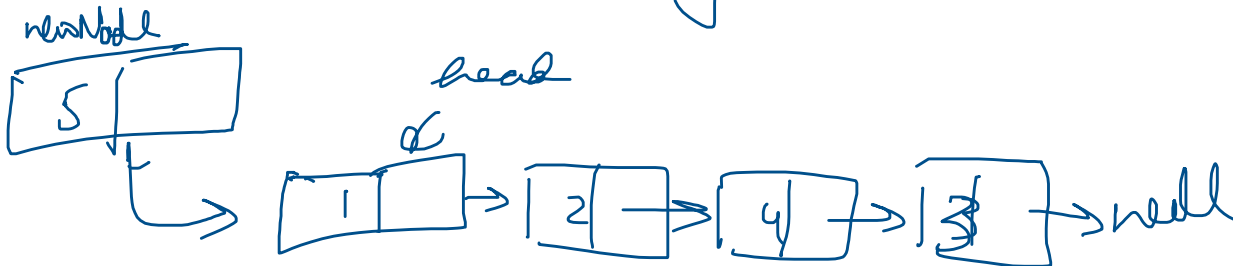
$n-2$  Jumps

Steps:

1. Add newNode
2. Create temp = head;
3. Take n-2 Jumps
4. newNode.next = temp.next;

5. temp.next = newNode;  
 Insert At Position 1  
 case:

Handle Separately

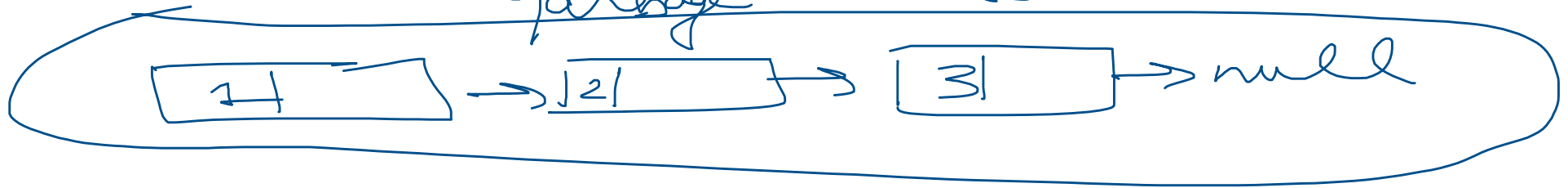


IAP(5, 1)

① newNode.next = head;

# Delete a LL

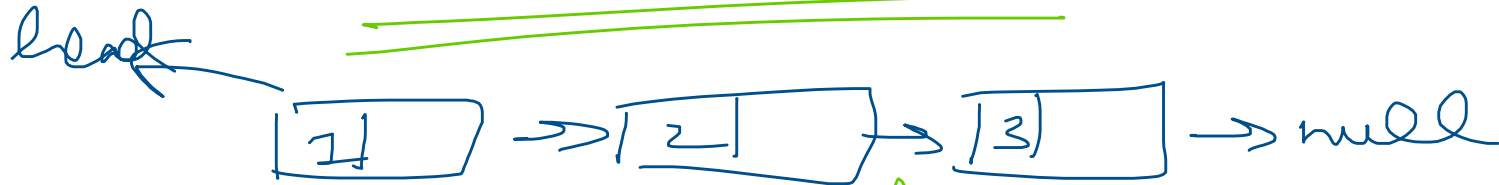
Garbage Collector



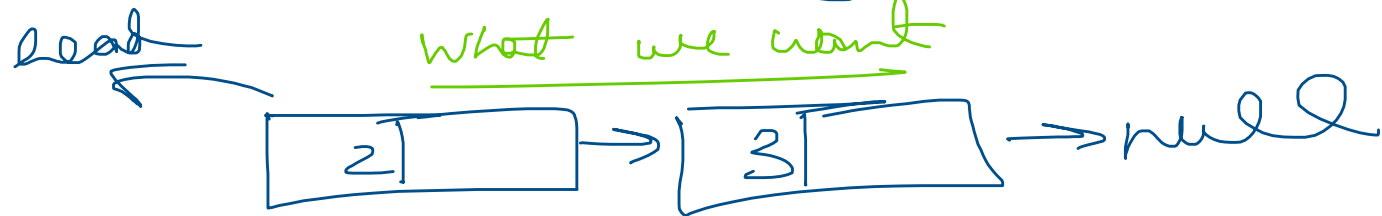
head = null

## Delete Node at Beginning of LL

what we have



what we want

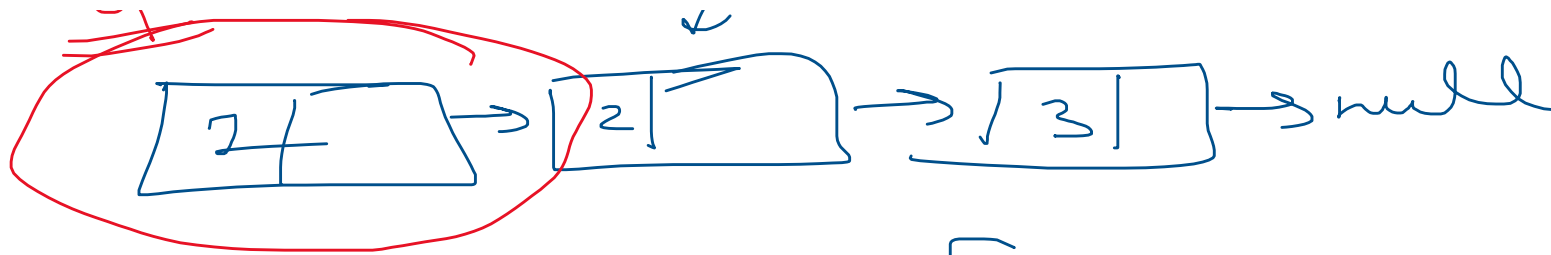


Steps

1. C

head

head = head -> next



If only 1 node [if (head->next == null)]  
 {  
     head  
     [1] --> null  
 }  
 return null;

Print the Middle Element of a LL (Second in Case of even elements)

Odd  $\rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow \text{null} \rightarrow 5/2 \rightarrow 2$   
 Even  $\rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 6 \rightarrow \text{null} \rightarrow 6/2 \rightarrow 3$

① Find length

$\xrightarrow{\text{length}} n \text{ steps}$

② Traverse till  $\frac{\text{length}}{2}$  & print value  $\rightarrow \frac{n \text{ steps}}{2}$

Total Steps  $\rightarrow \frac{3n}{2}$

### Print the Middle Element of a LL (Second in Case of even elements) - Method 2

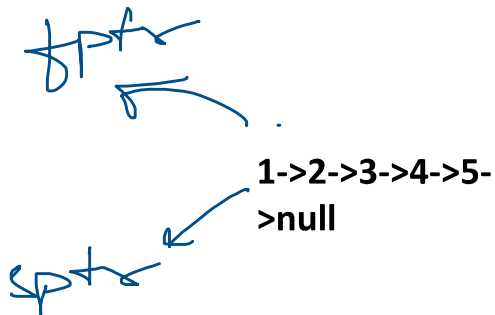


20kms/hr

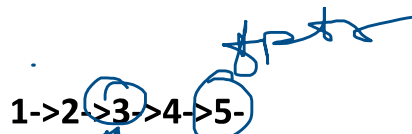
100kms



10kms/hr



**Fptr takes 2 jumps at a time.**  
**Sptr takes 1 jump at a time.**



Stop when (ptr.next == null)

~~>null~~  
~~sptr~~  
middle

1->2->3->4->5->6->null

~~sptr~~  
middle

while loop

~~sptr~~

Stop when ( ~~sptr~~ == null )