Data Science & AI & NIC - Param

Python-For Data Science

Linked List



Lecture No.- 03

Recap of Previous Lecture











Topic

Linked List Part-02

Topics to be Covered











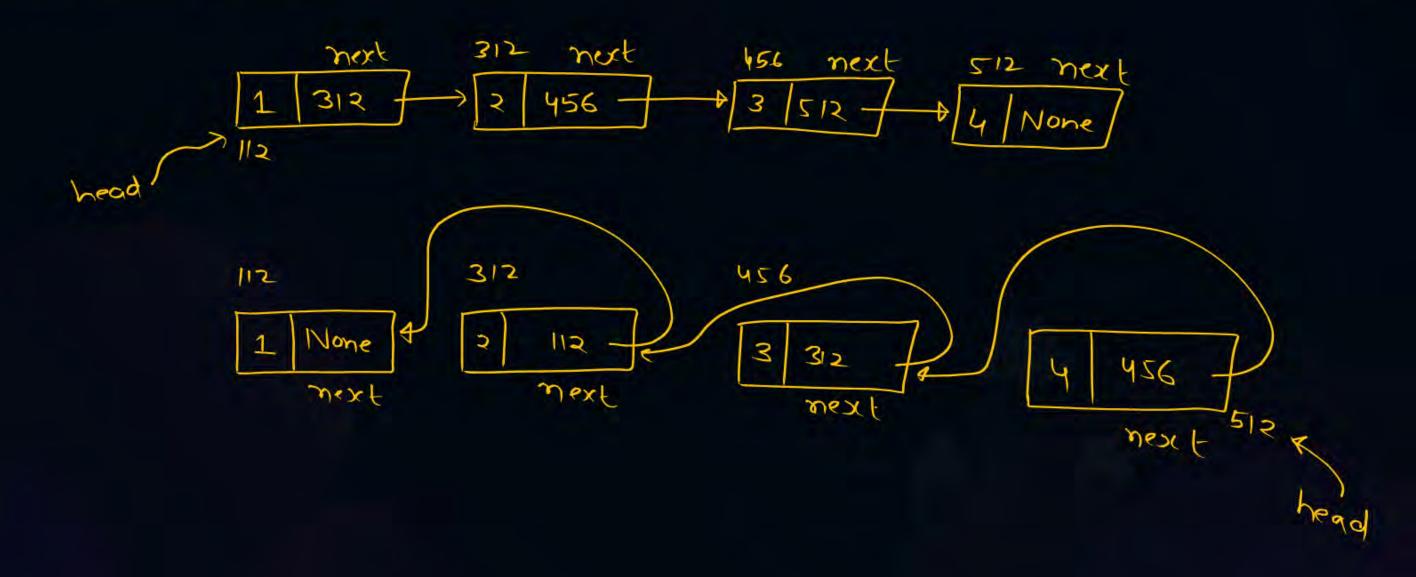
Topic

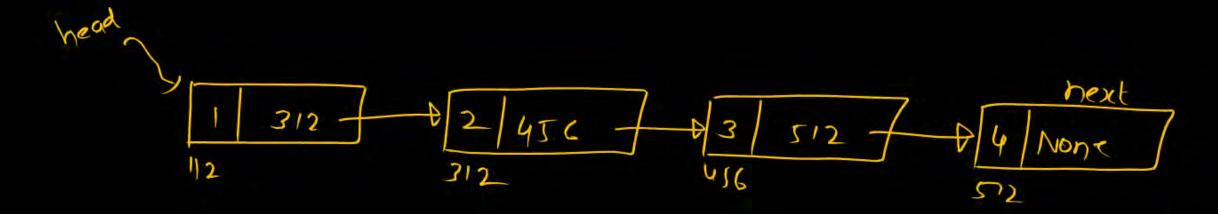
Linked List Part-03

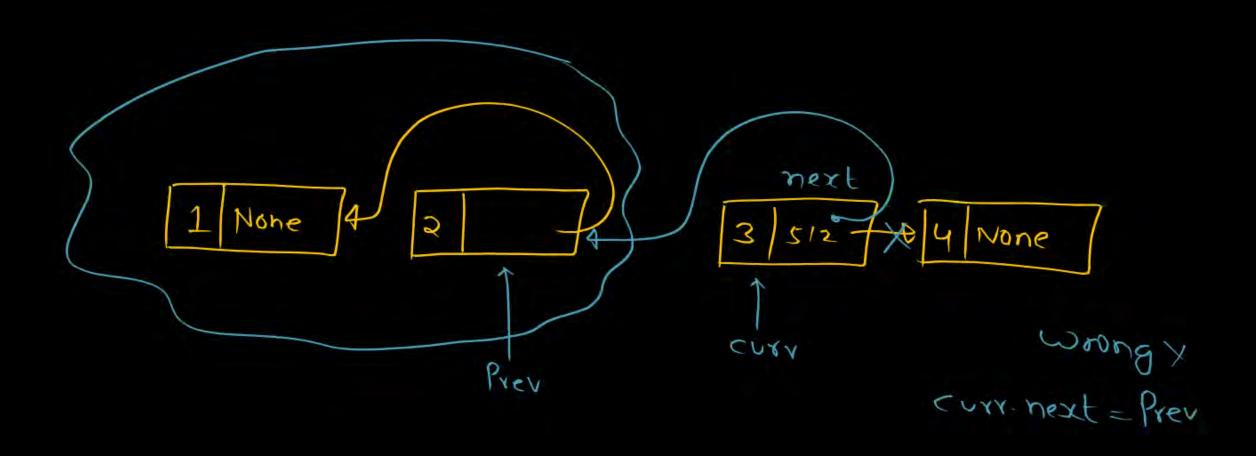


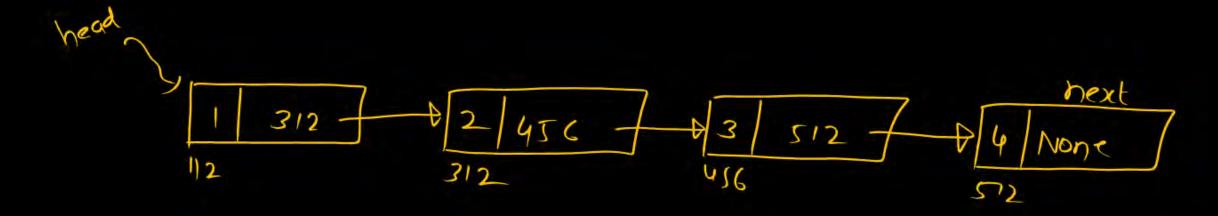
Topic: Linked List

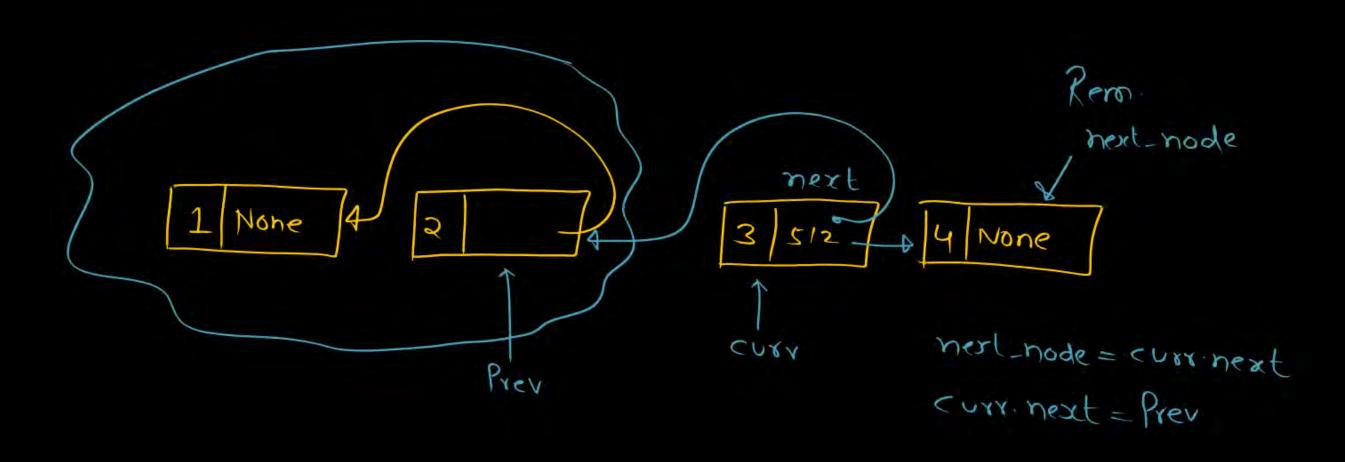


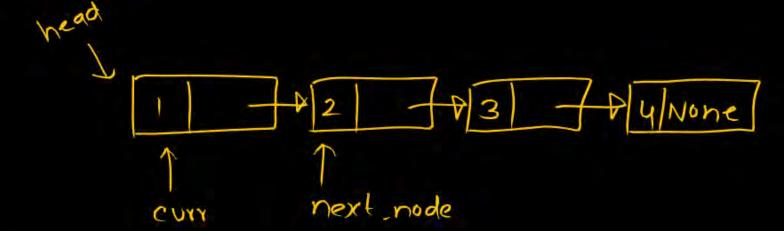




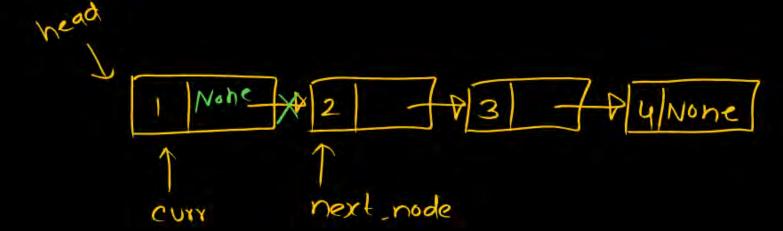




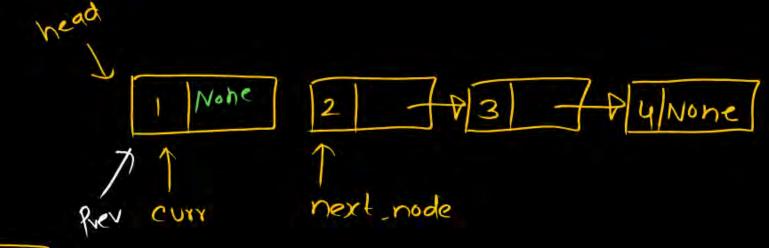


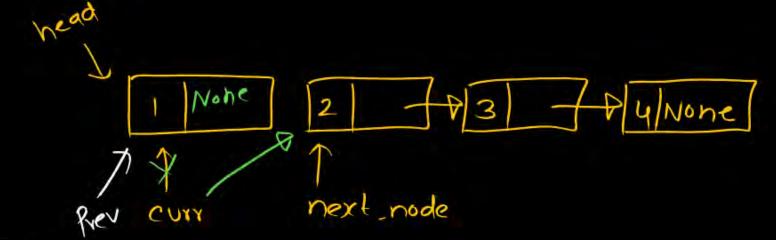


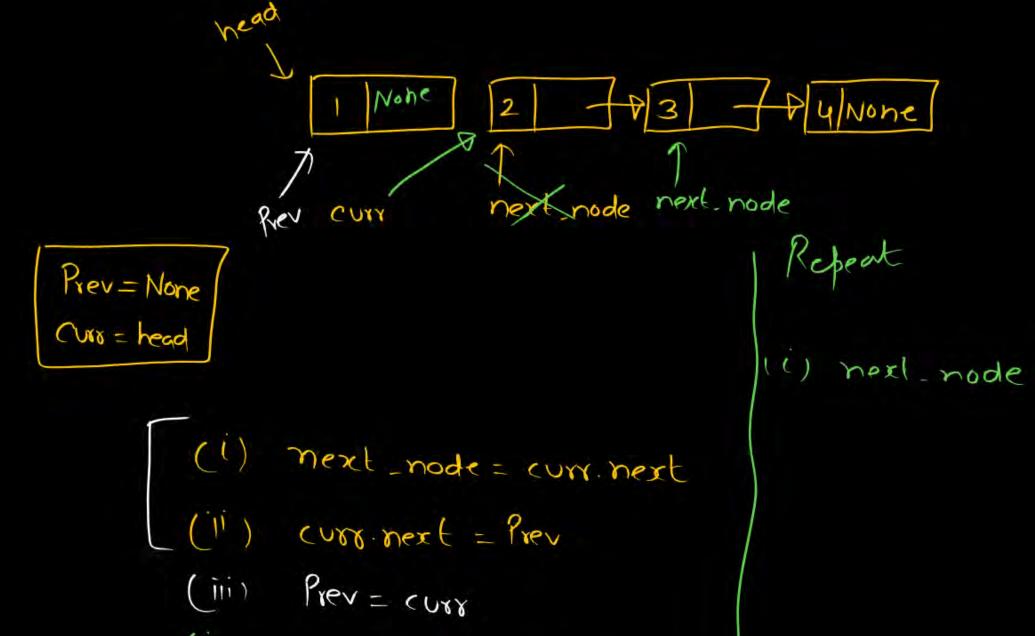
- (1) next_node = curr.next
- (11) curs next = Prev



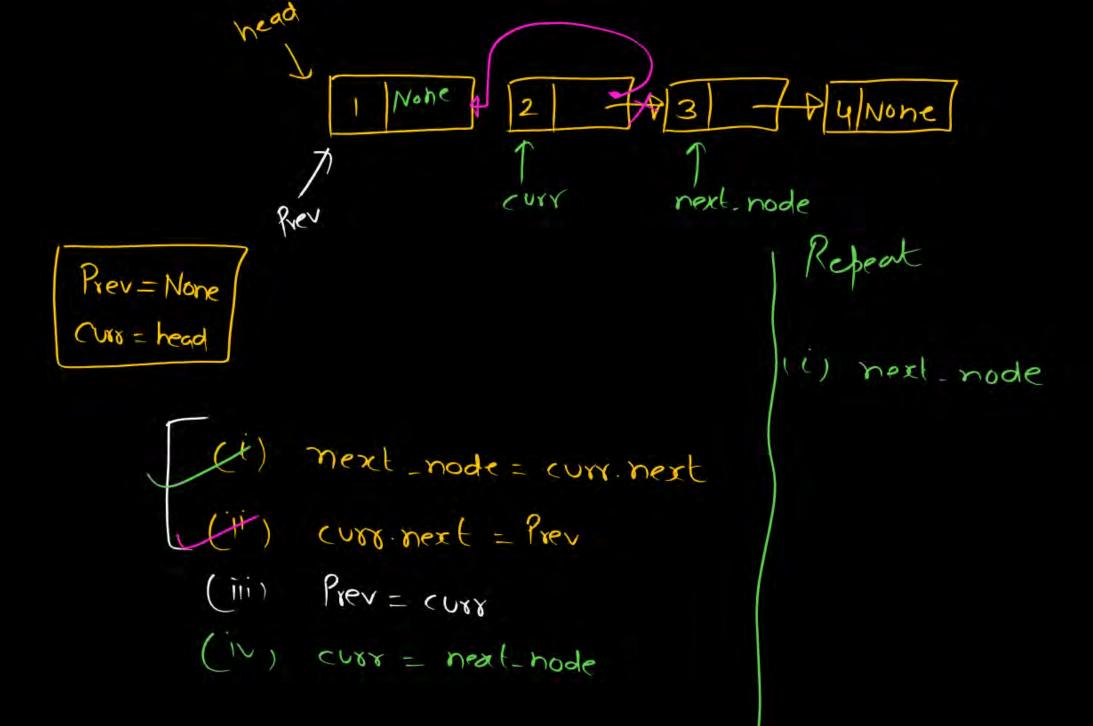
- (i) next_node = curr.next
- (11) curs next = Prev

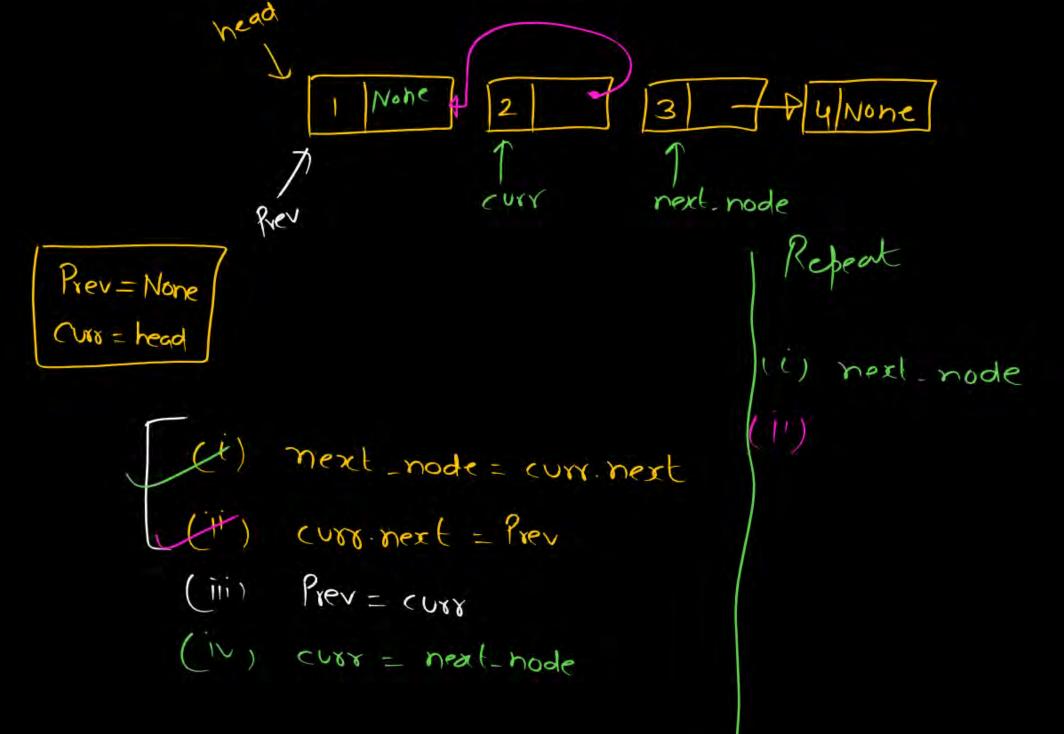






curr = next-node





None Prev = None Curs = head (i) next-node next_node = curr.next (iii) curs next = Prev IV) Prev = curr (iii) curr = next-node

None Prev = None Curs = head (i) next-node next_node = curr.next (iii) curs next = Prev (IV) Prev = curr (iii) curr = next-node

None nect-node Repeat Prev = None Curs = head (i) next-node next_node = curr.next (iii) curs next = Prev IV) Prev = curr (iii) (iv) curr = next-hode

None nect-node CUTY Prev Repeat Prev = None Curs = head next-node next_node = curr.next (iii) (iii) curs next = Prev (IV) Prev = curr (iii) (iv) curr = next-hode

None next-node Prev Repeat Prev = None Curs = head (i) next-node next_node = curr.next (iii) (iii) curs next = Prev (IV) Prev = curr (iii) (iv) curr = next-node

None 8 None next-node Prev None Prev = None Curs = head next_node = curr.next curs next = Prev Prev = curr (iii) (iv) curr = next-node

None A. None next-node Prev Prev = None None Curs = head next_node = curr.next curs next = Prev Prev = curr (iii) (iv) curr = next-node

None of 2 1 1 3 7 4 4 None of None

Prev next node

Prev next node

Prev = None Curo = head

> (it) next_node = curr.next (it) curr.next = Prev

(iii) Prev = curr

(iv) curr = next-node

head = Prev

Prev = None

curr = head

while current:

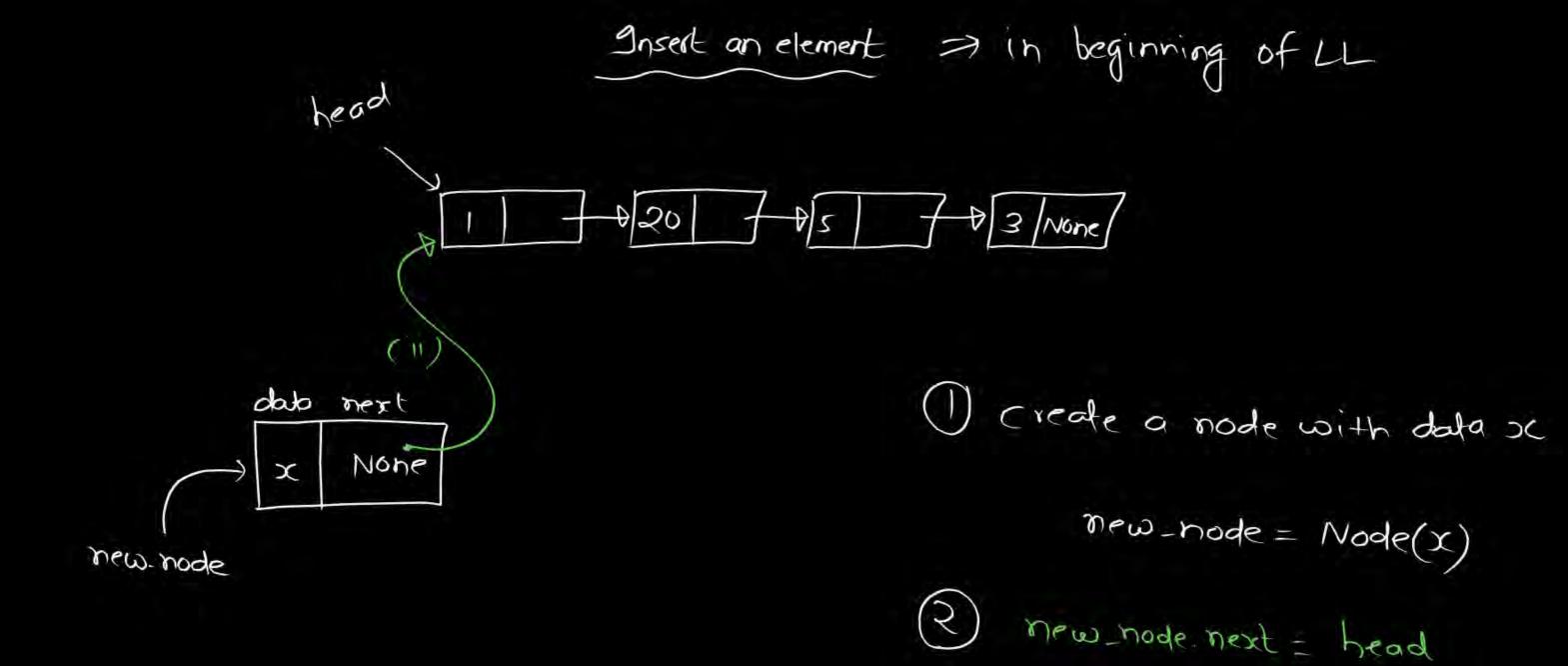
next_node = current

current = Prev

Prev = current

curr = next_node

veturn Prev



Insert an element >> in beginning of LL head Cin (ii) dub next (1) create a node with data oc None X new-node = Node(x) new node new_node. next = head

head = hew-hade

Insert an element >> in beginning of LL

LL is Empty head - None

head data next None Wake

O create a node with data oc

new-node = Node(x)

- (2) new_node next = head
- 3) head = hew-hode

Insert at End of LL head new_hode = Node (I) Curr = head if head is None o While cury next : head = new node CUTY = CUTY . next curr next = new node return head

LL is Empty

return head

CUTY 0[2] +0[3] NOCH data next new node

Insert at End head def None else None Remove 1st node

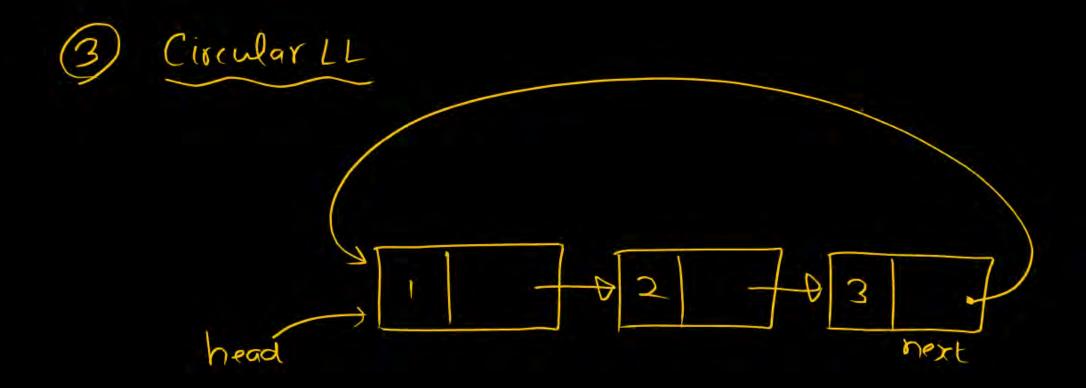
Remove last node

Shayan Shayan

statement o

Types of LL 1) Singly LL wead (2) Doubly LL Class Node " def -- init -- (self, data) self. Prev = None self data = data self hext = None Sprev data next Prev data next Prev data next 20 10 None

MON



4) Doubly Circular LL

head Prev data next saffer data next Prev data next 10 HW.

linked list

Time complexity

H.W

U Search -

2) Print

3.) Insect

4) Inserti



THANK - YOU