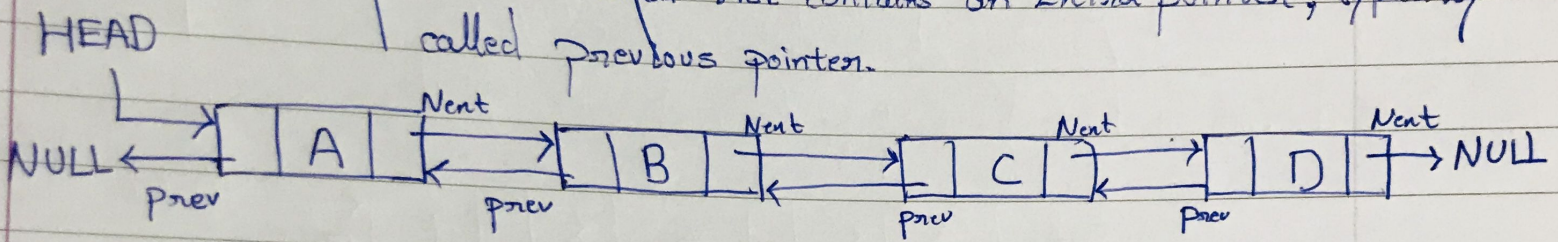
Double Linked List

A double linked list contains an extra pointer, typically called previous pointer.




```
class Node :
    def __init__(self, next=None, prev=None, data=None):
        self.next = next
        self.prev = prev
        self.data = data
```

Advantages :-

i) DLL can be traversed in both forward & backward.

ii) Delete operation is more efficient.

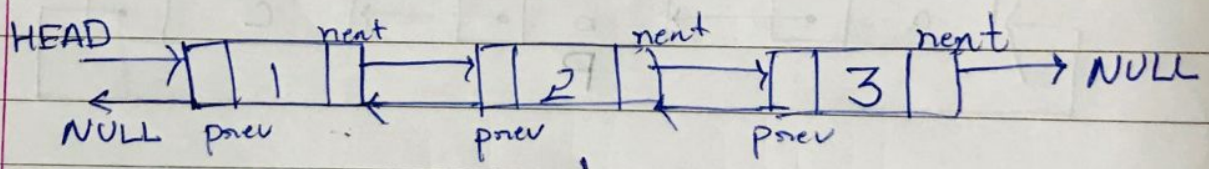
Double Linked List :-

Date:

=> Set a head :-

If element enist in the Linked List
i) remove it from the Linked List
ii) Set to head.

```
def setHead(self, node):
```



↓ To set head

Suppose we want to make node(2) as head

self.insertBefore(head, node)

i) InsertBefore (self.head, node)

↑ We will call insert before function

→ self.remove (node)

↳ Insert Before will call remove node function

=> Remove a node :

We write ... before function

→ self.remove(node)

↳ Insert Before will call remove node function

=> Remove a node :

```
def remove(self, node):
    if (node == self.head):
        self.head = self.head.next
    if (node == self.tail):
        self.tail = self.tail.prev
```

} Check for head & tail of the node

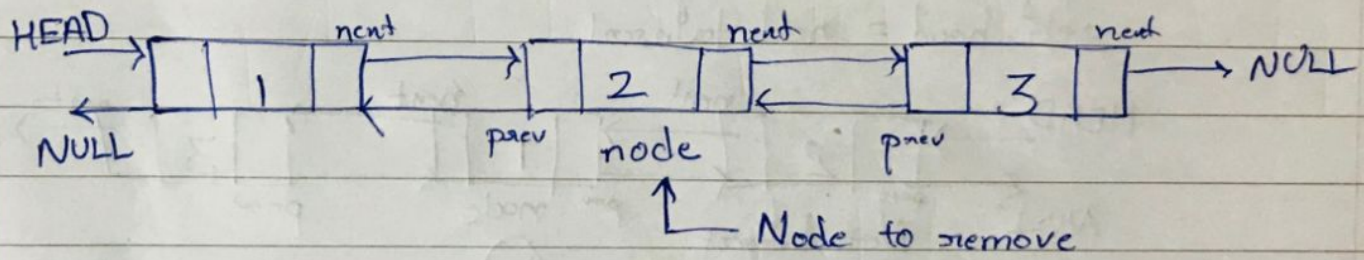
Here we break a single Link

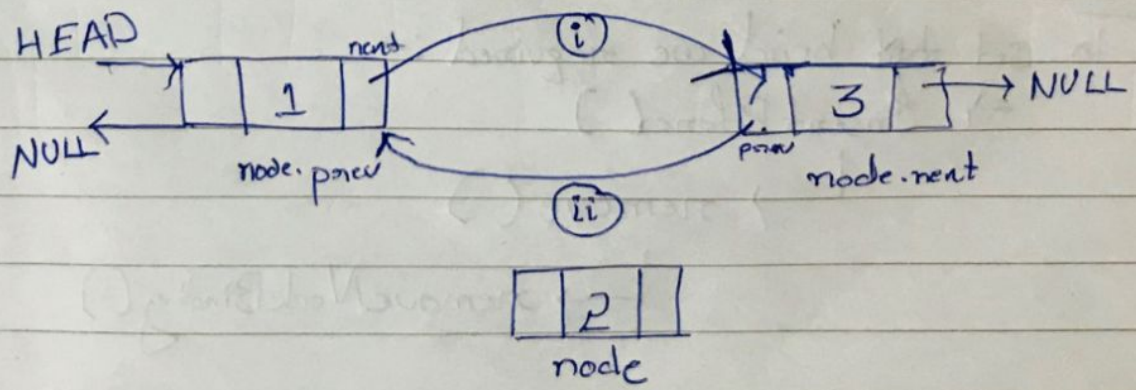
self.removeNodeBindings(node)

↳ to break and bind.

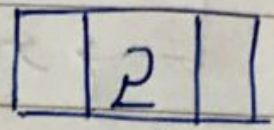
Called by removeAll()

```
def removeNodeBindings(self, node):
    if node.prev is not None:
        node.prev.next = node.next        (i)
    if node.next is not None:
        node.next.prev = node.prev        (ii)
    node.next = None
    node.prev = None
```





So we have removed 2 from the linked list, Now perform insertBefore()



node

So we have removed 2 from the Linked List, Now perform insertBefore()

Insert Before a node :-

self.remove(node) # Head none

(self.insertBefore (self.head, node))

def insertBefore(self, node, nodeToInsert):

self.remove(nodetoInsert) → self.head → node

nodetoInsert.prev = node.prev (i)

nodetoInsert.next = node

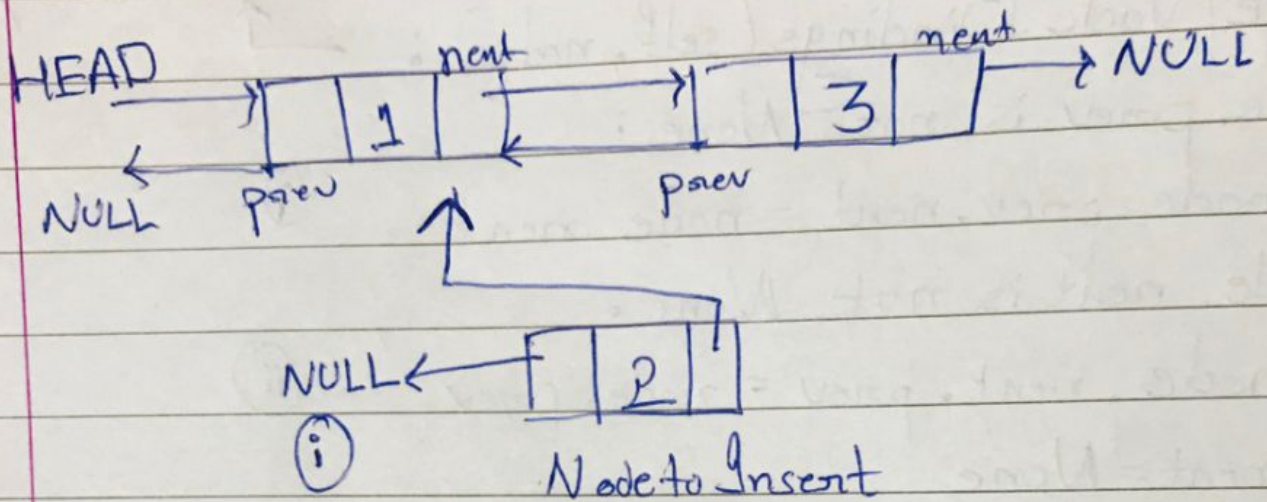
if node.prev is None:

self.head = nodeToInsert (ii)

else: (iii)

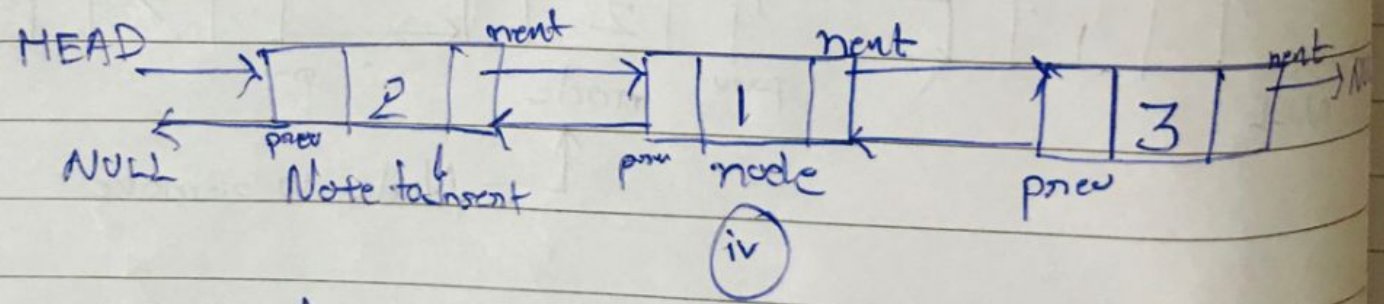
node.prev.next = nodetoInsert

node.prev = nodetoInsert (iv)



So node.prev is None : (ii)

self.head = node to Insert



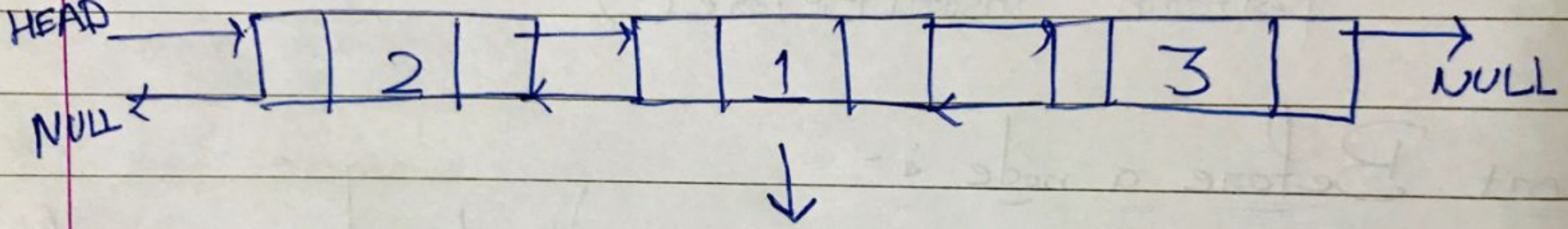
Summary : To set ~~text~~ head we required :-

→ InsertBefore( )

↳ remove( )

↳ removeNodeBinding( )

⇒ Set tail() :-



To set tail

So here we have to do :-

→ insertAfter()

└ remove()

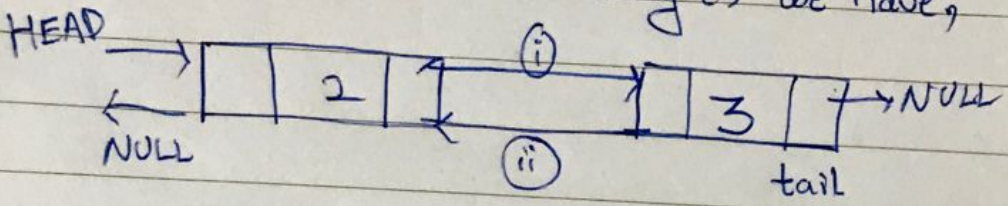
└ removeNodeBinding()

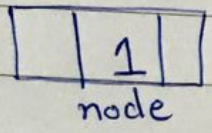
* we call self.insertAfter(self.tail, node)

4(1)

Node

After remove() and removeBinding() we have,





=> Insertion After a Node :-

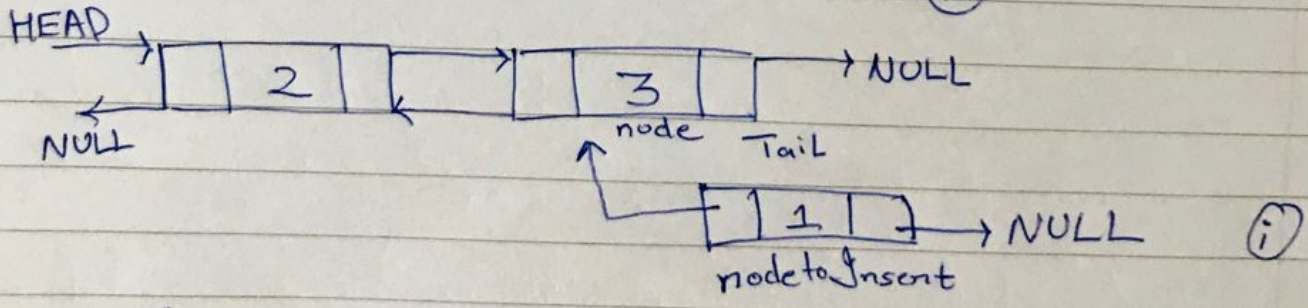
```
def insertAfter(self, node, nodeToInsert):
                    ↳ tail          ↳ node
    self.remove(nodeToInsert)  # Done
    nodeToInsert.prev = node
    nodeToInsert.next = node.next
    if node.next is None:
        self.tail = noteToInsert
    else:
        node.next.prev = nodeToInsert
    node.next = nodeToInsert
```

3 (i)

(ii)

(iii)

(iv)



So, node.next is None in our case