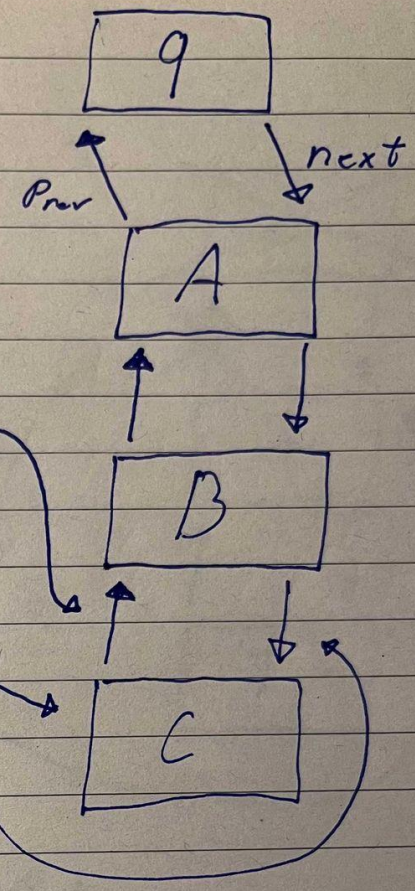
Doubly Linked list

```
addNodeLast(newNode):
    if tail == null:
        head = newNode
        tail = newNode
    else
        newNode.Prev = tail
        tail.next = newNode
        tail = newNode
```

```
addNodeFirst(newNode):
    if head == null :
        head = newNode
        tail = newNode
    else
        newNode.next = head
        head.Prev = newNode
        head = newNode
```



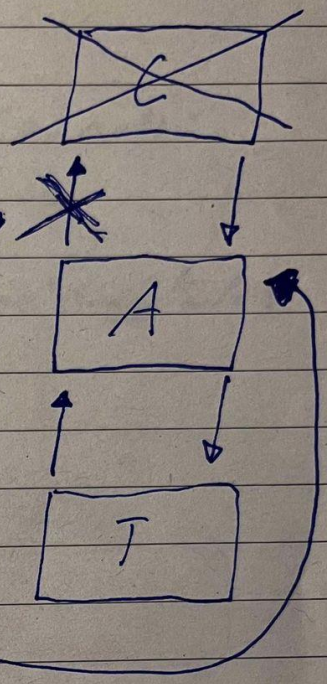
Doubly Linked List

```
removeFirst :
    if head == null :
      return null

    removedNode = head

    if head == tail :
      head = null
      tail = null
    else
      head = head.next
      head.prev = null

    removedNode. return removedNode
```



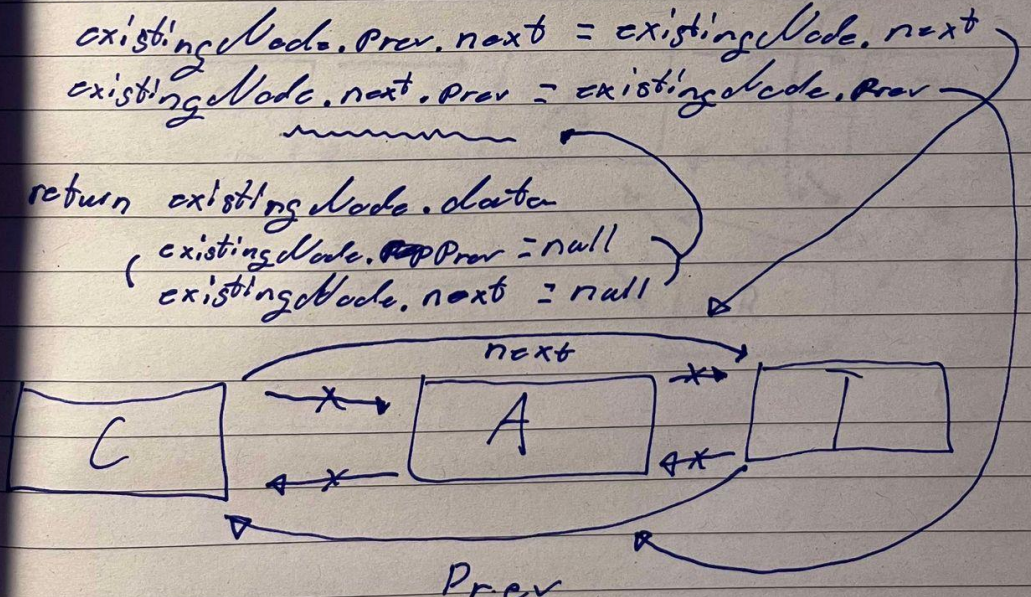
Doubly Linked List

```
removeNode(existingNode):

    if existingNode == Null || head == null:
        return null

    if head === tail && head === existingNode:
        head = null
        tail = null
    else if existingNode === head:
        head = head.next
        head.prev = null
    else if existingNode === tail:
        tail = tail.prev
        tail.next = null
    else:
        existingNode.prev.next = existingNode.next
        existingNode.next.prev = existingNode.prev

return existingNode.data
    (existingNode.prev = null
    existingNode.next = null
```



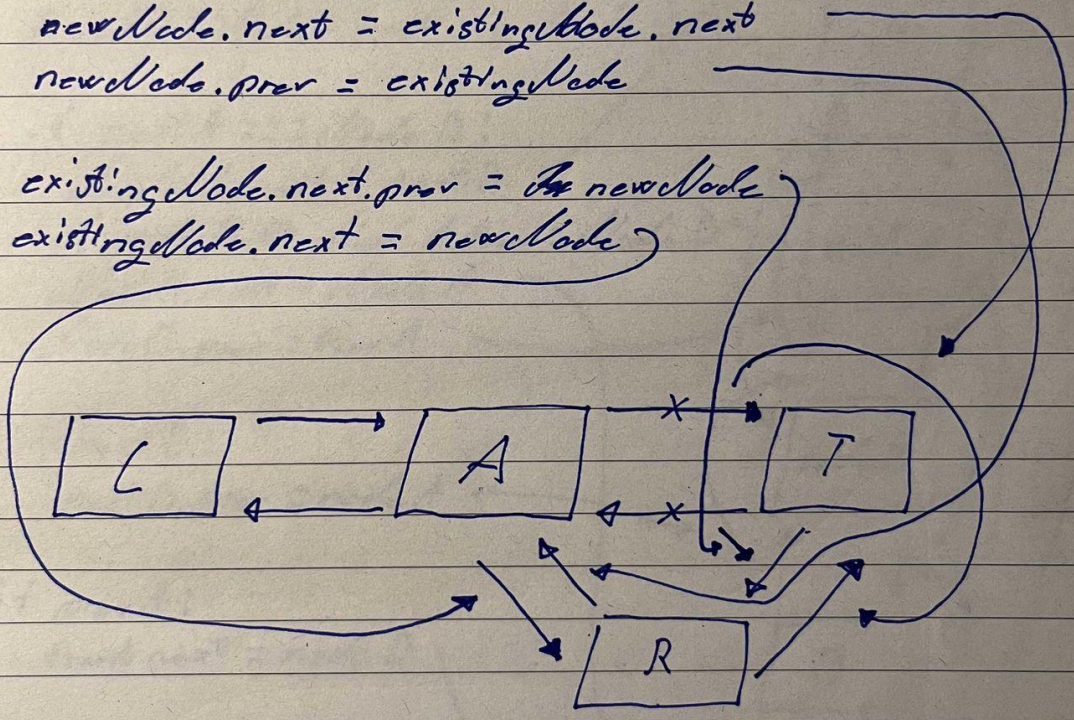
B Doubly Linked list

```
insertAfterNode(newNode, existingNode):

  if existingNode === null:
    return

  if existingNode === tail:
    addNodeLast(newNode)
  else:
    newNode.next = existingNode.next
    newNode.prev = existingNode

    existingNode.next.prev = newNode
    existingNode.next = newNode
```



Doubly Linked list

SwapNode (NodeA, NodeB):

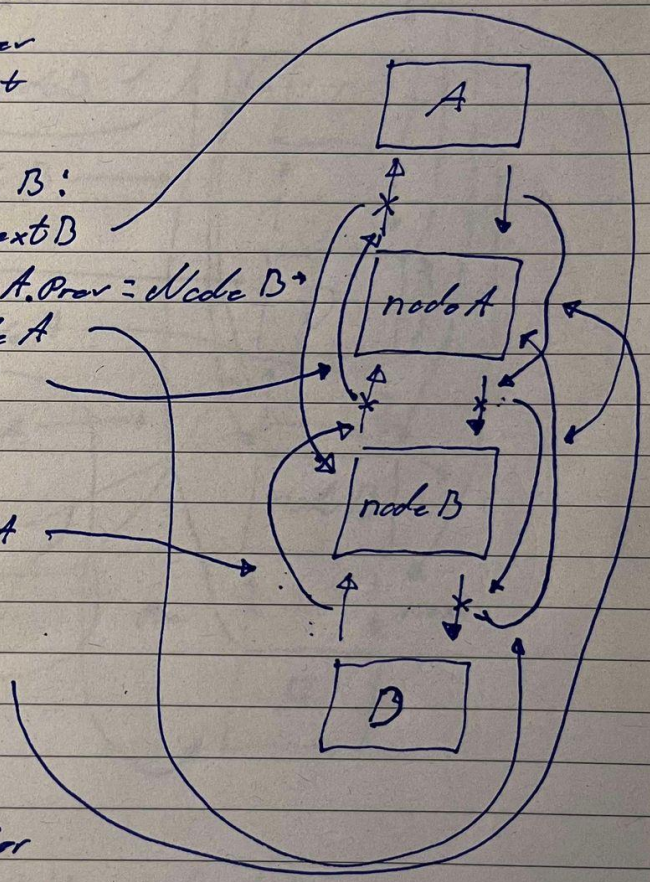
if NodeA == NodeB || NodeA == null || NodeB == null:
return

PrevA = NodeA.prev
nextA = NodeA.next
PrevB = NodeB.Prev
NextB = NodeB.next

if nextA == NodeB:
NodeA.next = nextB
~~NodeB.next~~ NodeA.Prev = NodeB
NodeB.next = NodeA
NodeB.prev = PrevA

if nextB:
nextB.prev = nodeA

if prevA:
PrevA.next = nodeB



hvis nodeA kommer før nodeB og de er naboer.

kan også bruges omvendt
nogle værdier og rækkefølge skal bare ændres lidt

Doubly linked list

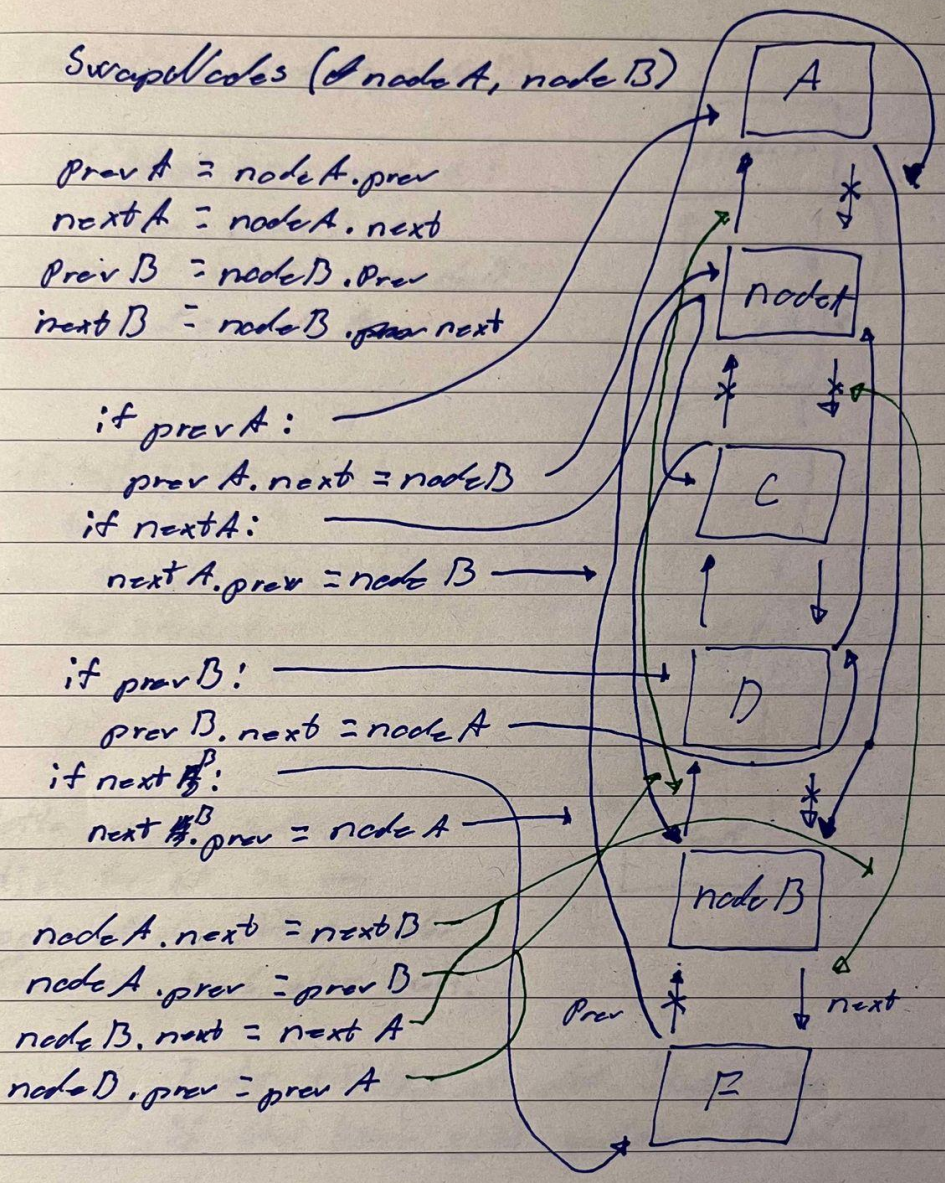
```
swapNodes(nodeA, nodeB)

prevA = nodeA.prev
nextA = nodeA.next
prevB = nodeB.prev
nextB = nodeB.next

if prevA:
    prevA.next = nodeB
if nextA:
    nextA.prev = nodeB

if prevB:
    prevB.next = nodeA
if nextB:
    nextB.prev = nodeA

nodeA.next = nextB
nodeA.prev = prevB
nodeB.next = nextA
nodeB.prev = prevA
```



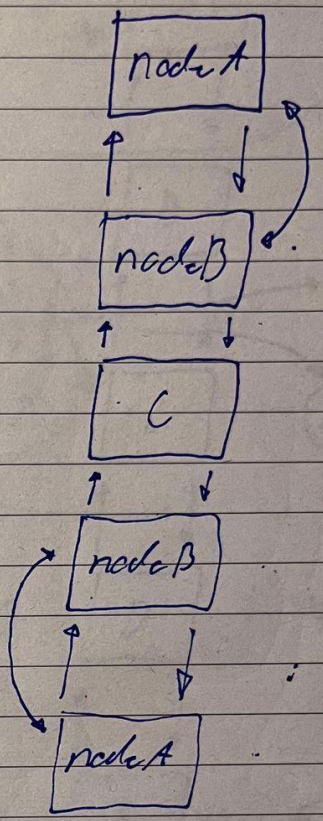
Hvis nodeA og nodeB ligger forskellige steder i den linkede list.
Og der ikke er nogle af dem som er enten head eller tail.

Doubly Linked list

```
swapNodes (nodeA, nodeB)

    if head === nodeA :
        head = nodeB
    else if head === nodeB:
        head = nodeA

if tail === nodeA:
    tail = nodeB
else if nodeB === tail:
    tail = nodeA
```



Dette er et sidste tjek for at se om noget af de skiftede noder enten var head eller tails.

I det tilfældet at det skulle ske så skal listen også opdater head eller tail

Doubly Linked List

```
nodeAT(index)

Current = head
Count = 0
while (current != null):
    if Count == index:
        return ~~Count~~ Current
    Count ++
    Current = Current.next
return null
```