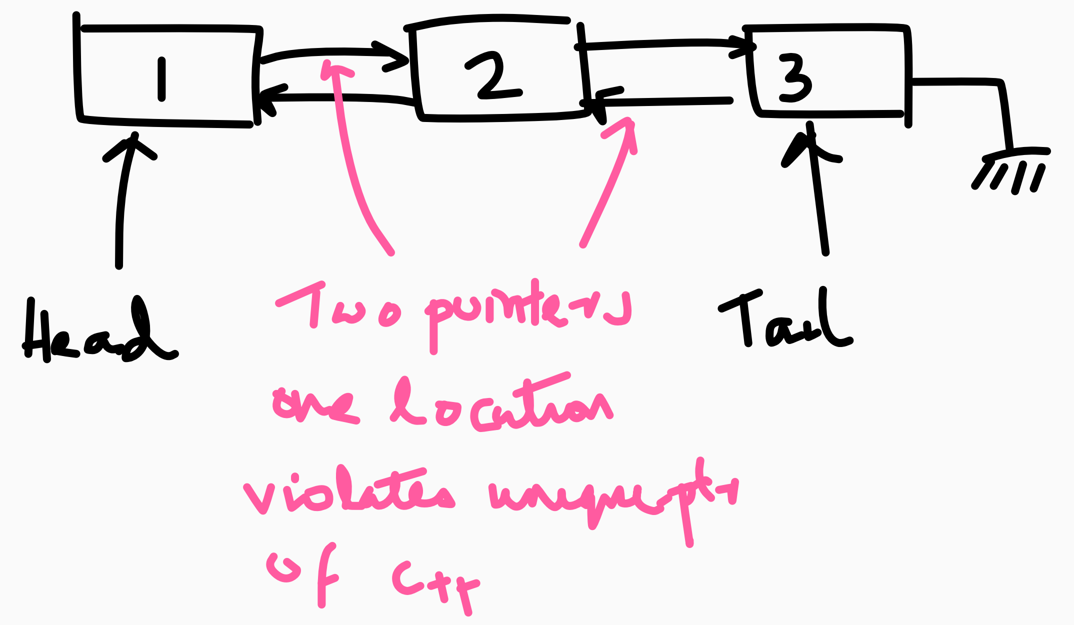
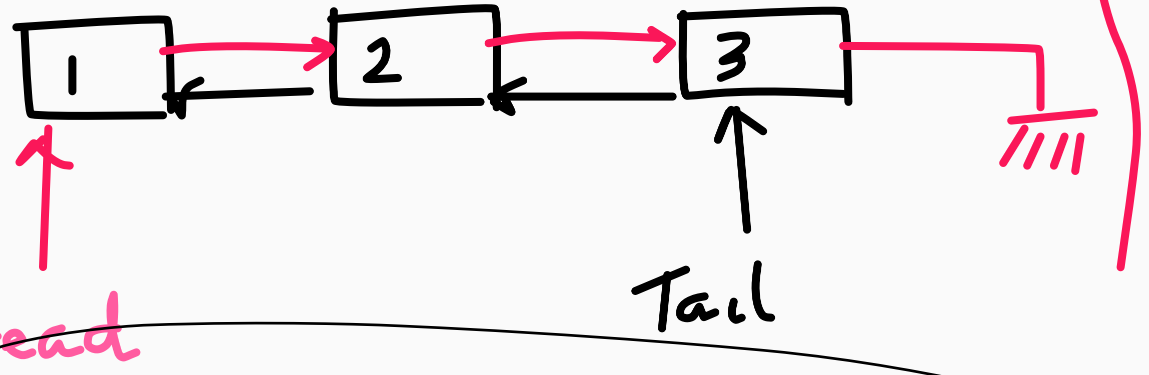
Doubly Linked List

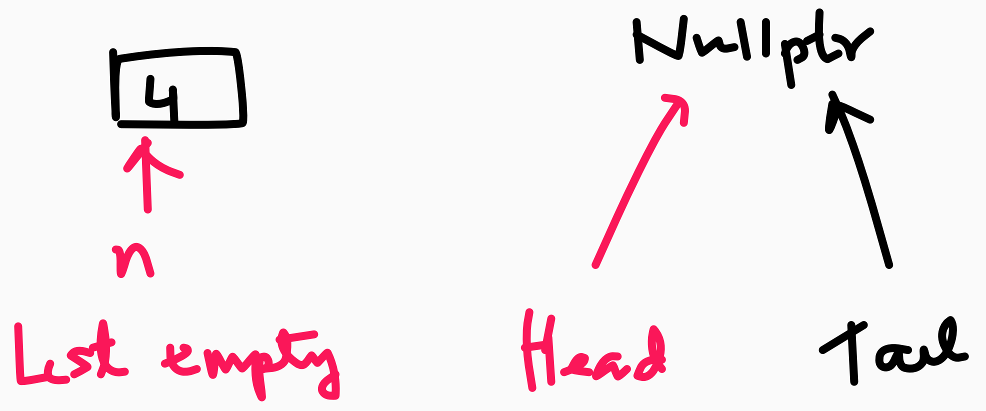




```
// n is a smart pointer
n->right = move(head) // right is a smart pointer
// head doesn't point to anything now
n->right->left = n.get() // assignment is possible as l-value is not a smart pointer
// r-value needs to wrapped around with get() to get raw pointer
head = move(n) // both are smart pointers
```

Insert at head. list nonempty

Insert at head

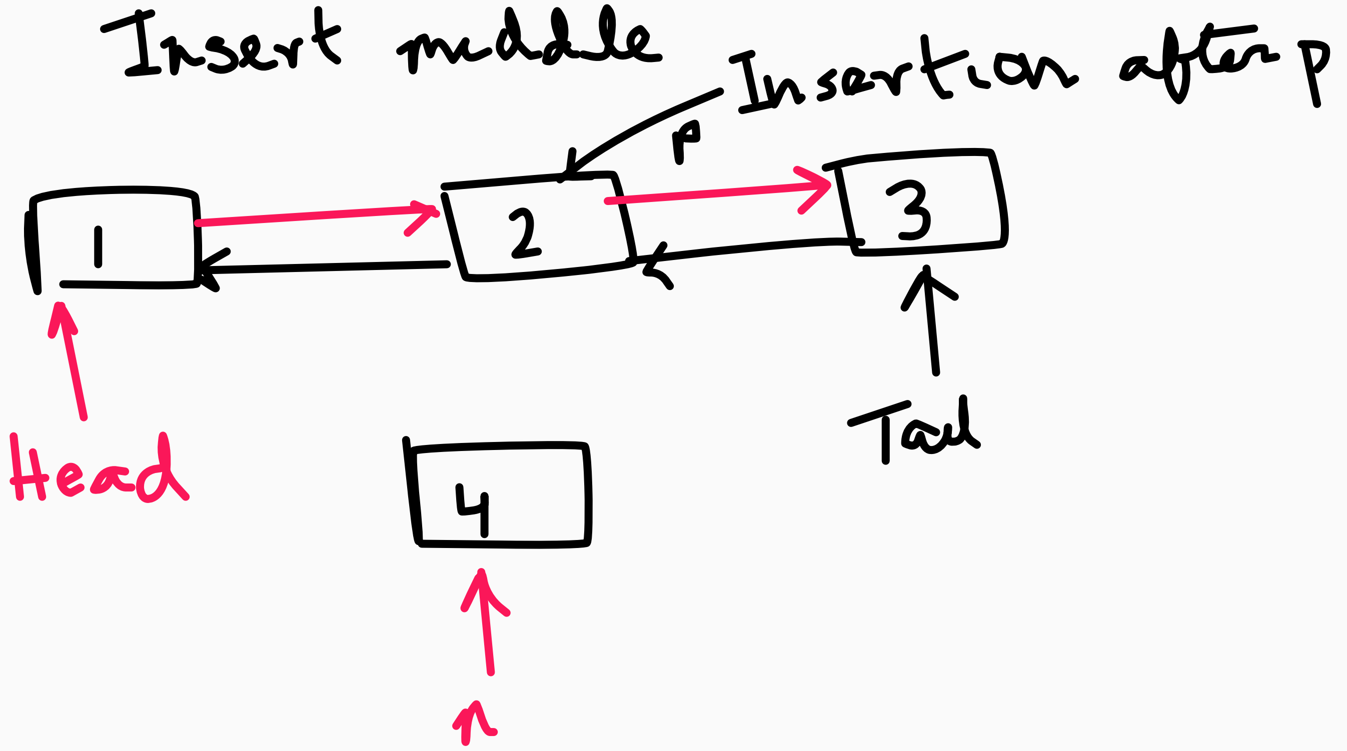


```
head = move(n) // both are smart pointers
tail = head.get() // r-value is converted to raw pointer
head->left = nullptr // regular assignment will work
```

Insert at tail

```
n->left = tail// none of them is smart pointer
n->left->right = move(n)// n doesn't point to anything now
// tail still points to previous node. adjust that
tail = tail->right.get() // get raw pointer before assignment
```

Insert middle



```
// p is not a smart pointer so id n->left
n->left = p;
n->right = move(p->right);
// after the above step, p->right points to nothing
// so always need to go through n->right
n->right->left = n.get() // r-value involves dumb pointer
n->left->right = move(n) // n is smart ptr - so move ownership
```