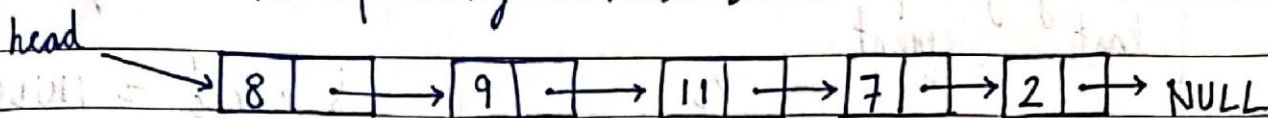


Deletion in a Linked List

Consider the following Linked List

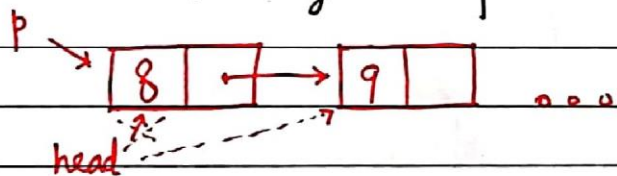


Deletion can be done for the following cases:

- 1> Deleting the first Node
- 2> Deleting the node at an index
- 3> Deleting the last Node
- 4> Deleting the ~~last~~ node with a given value.

The deletion just like insertion is done by rewiring the pointer connections, the only caveat being: we need to free the memory of the deleted node using `free()`.

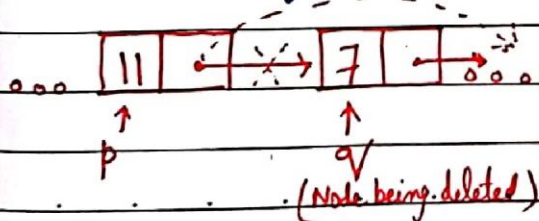
Case 1: Deleting the first node



```
struct Node * p = head;
head = head->next;
free(p);
```

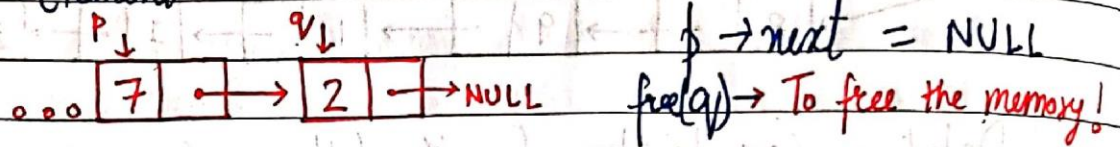
Case 2: Deleting the node at an index

for deleting a given node, we first bring a temporary pointer p before element to be deleted and q on the element being deleted

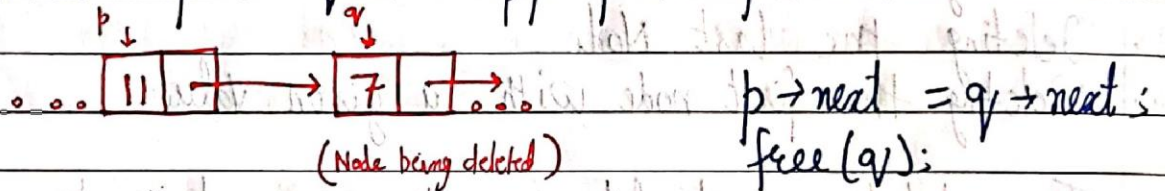


```
p->next = q->next;
free(q);
```

Case 3: Deleting the last Node
Last node can be deleted just like case 2 by bringing p on second last element and q on last element.



Case 4: Delete the first node with a given value
This can be done exactly like case 2 by bringing pointers p & q to appropriate positions



head = p = q

head = head = head

free(q)