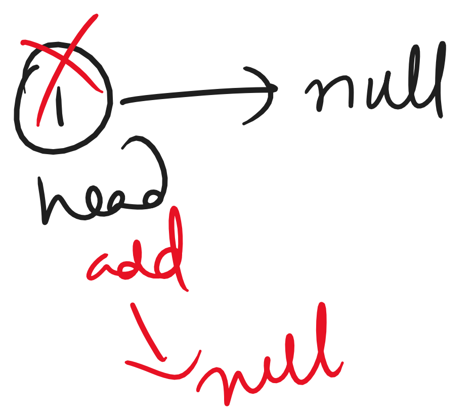
```
DeleteEndNode (Node** head) {

}
```

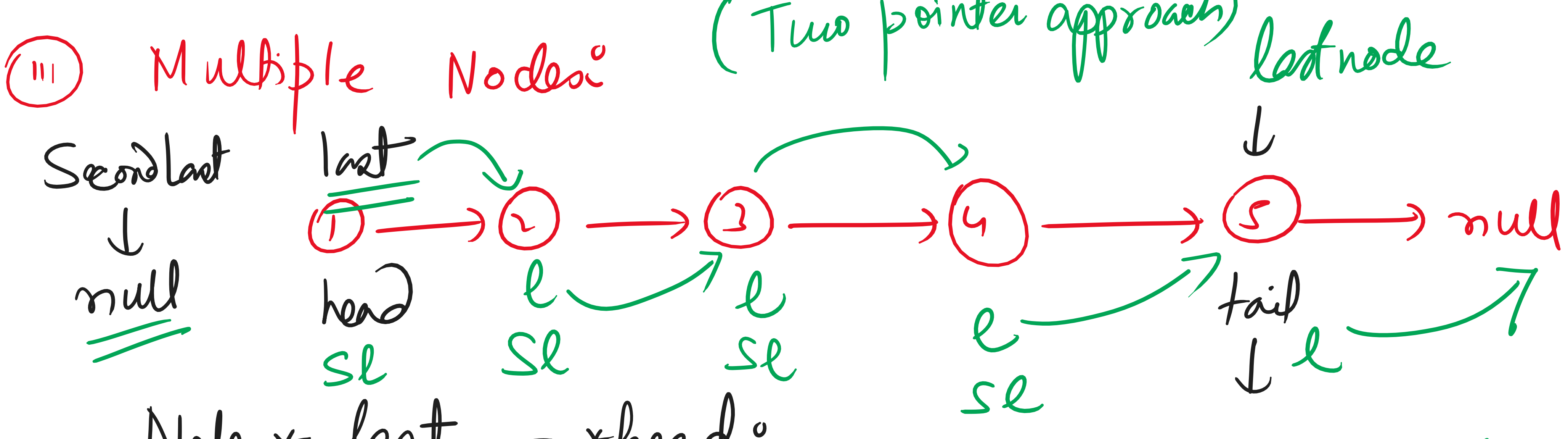
① Condition check : if empty → can't delete.

⑪ Single Element :→ (h→next == null)



```
→ delete *head;
→ *head = null;
```

⑪ Multiple Nodes: (Two pointer approach)



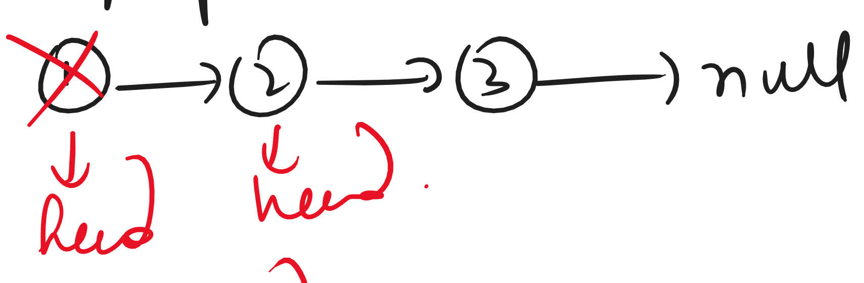
```
Node* last = *head;
Node* secondLast = nullptr;
→ while (last→next != nullptr)
  {
  → secondLast = last;
    last = last→next;
  }
```

```
delete last;
sl→next = null;
```

Delete Target Node :→

1. Empty → can't delete

2.

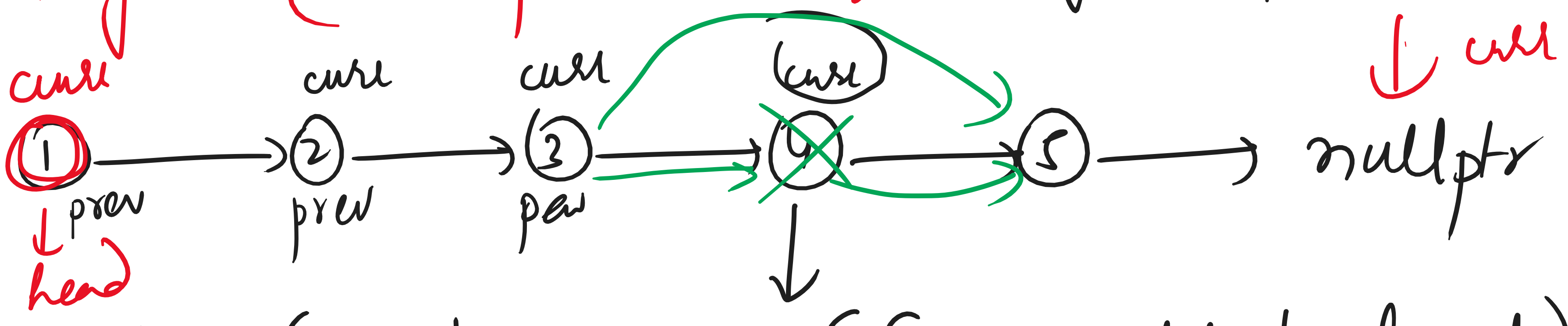


target is at the head

target = 1

```
temp = head
head = head→next
delete temp;
```

3. Delete Target (Multiple Nodes) target = 4 ⑦

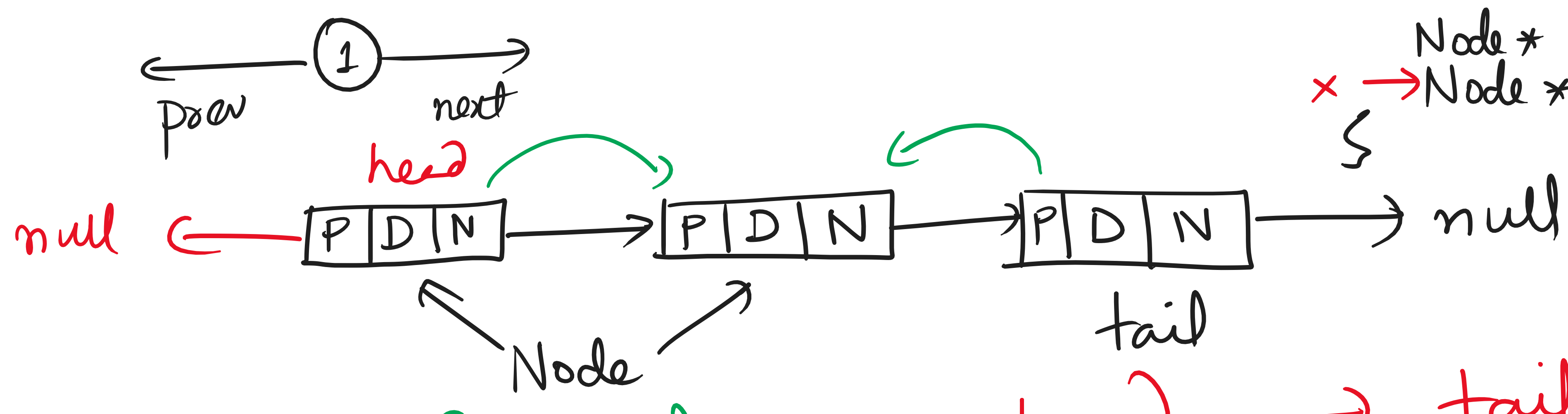


```
prev = null
curr = head
while (curr != nullptr && curr→data != target)
{
  prev = curr;
  curr = curr→next;
}
```

```
prev→next =
  curr→next
delete curr;
```

Doubly Linked List :→ Traversals:

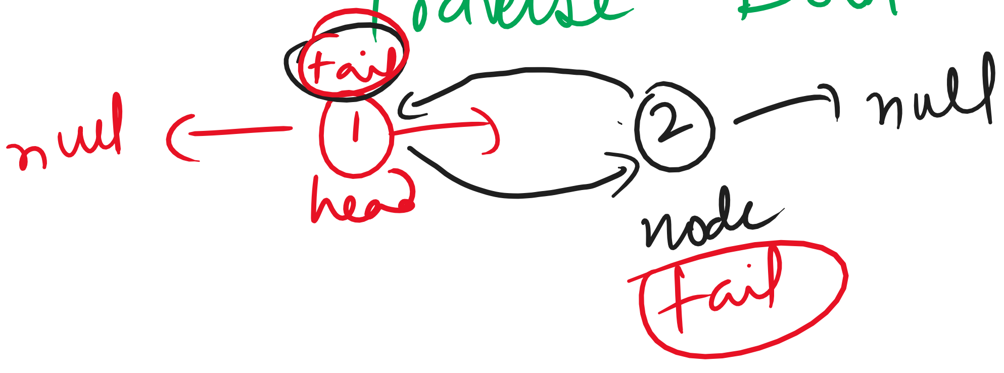
```
Class Node {
  int data;
  Node* next;
× → Node* prev;
}
```



Traverse Both Ways :

head → tail

tail → head



null
tail
1
head
2
null
node
tail