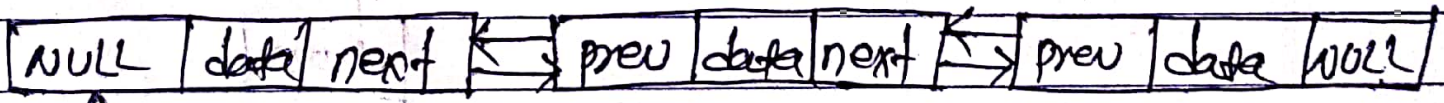
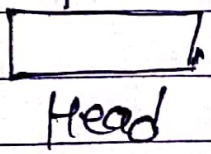
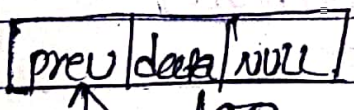
Doubly linked list

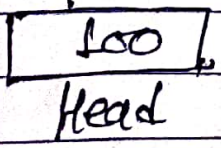
A doubly linked list is different from a singly linked list in a way that each node has an extra pointer that points to the previous node, together with the next pointer & data similar to singly linked list.



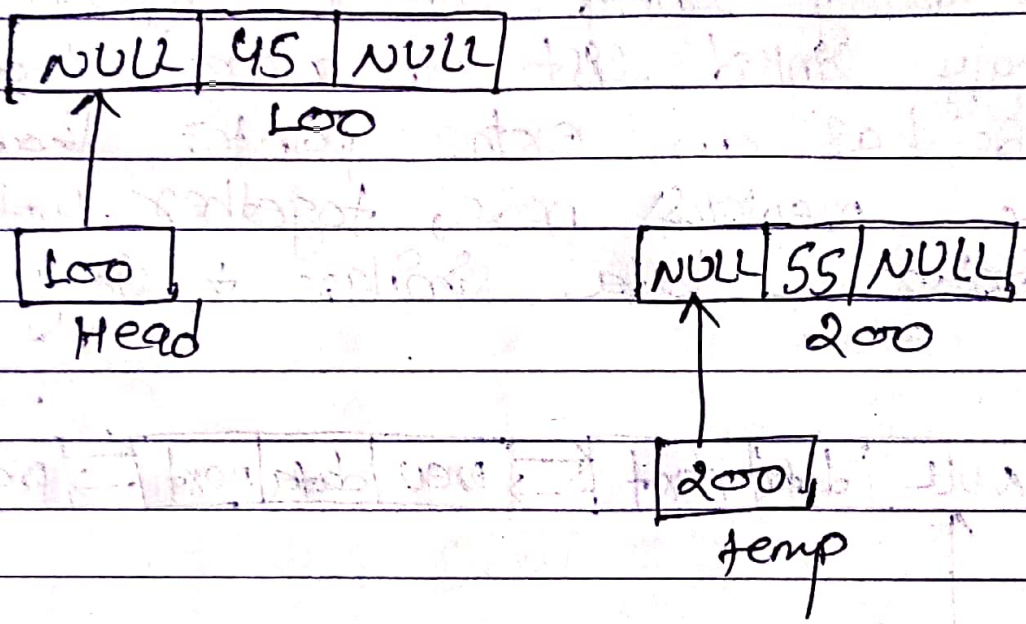



```
struct node {
    struct node *prev;
    int data;
    struct node *next;
};
```

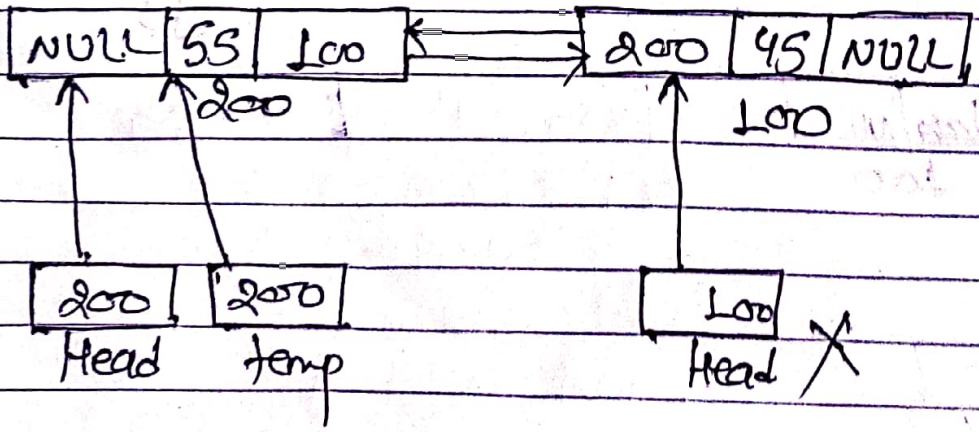




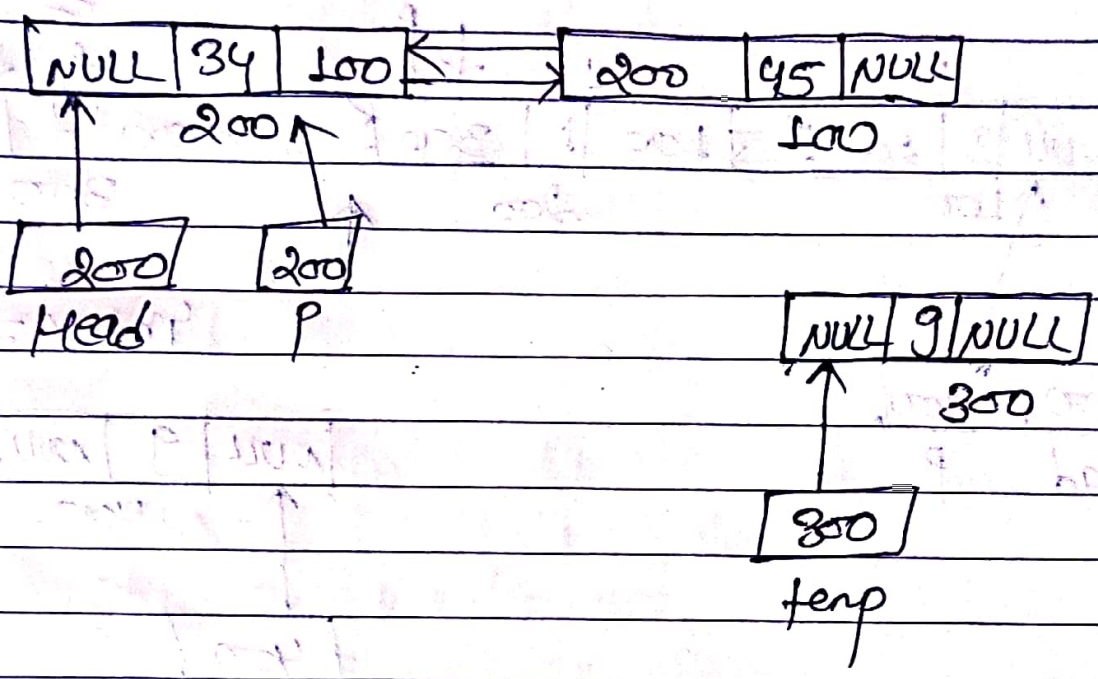
Insertion at the beginning of the Doubly linked list




```
temp -> next = head;
temp -> prev = NULL;
head -> prev = temp;
head = temp;
```

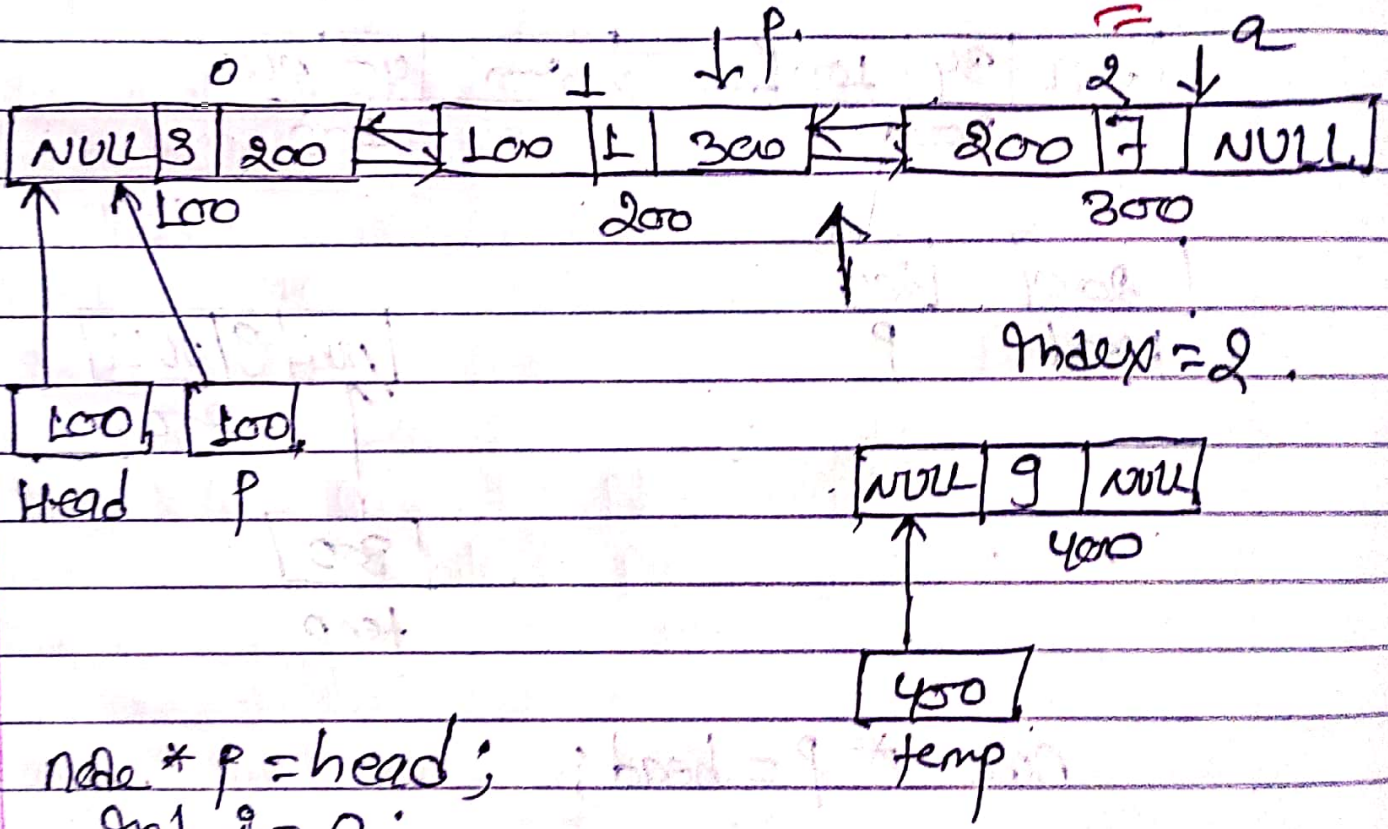


Insertion At the End in Doubly Linked List




```
node * p = head;
while (p->next != NULL) {
    p = p->next;
}
p->next = temp;
temp->prev = p;
temp->next = NULL;
```

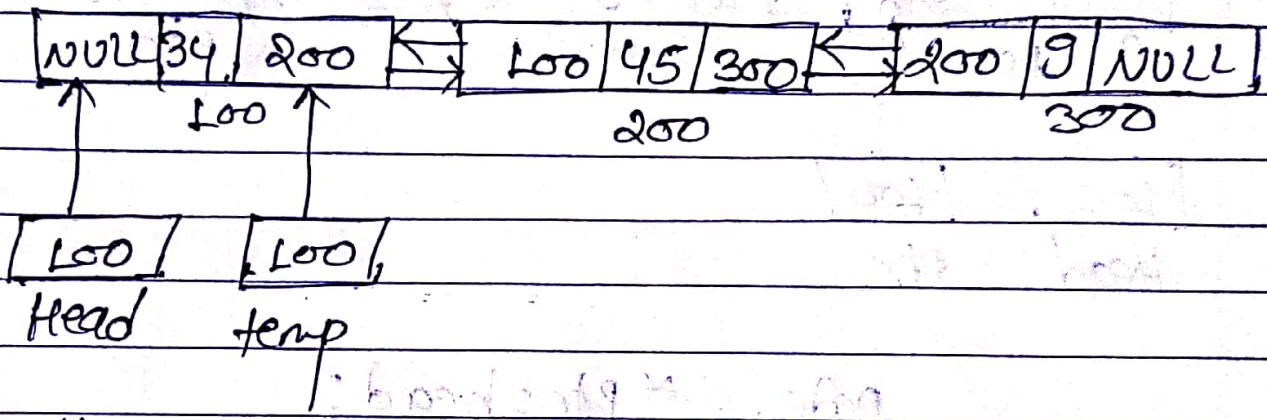
Insertion After given position in Doubly Linked List.




```
node * p = head;
int i = 0;
while ( i != index - 1 ){
        p = p -> next;
        i++;
}
node * q = p -> next;
temp -> next = q;
temp -> prev = p;
p -> next = temp;
q -> prev = temp;
```

Deleting Doubly linked list

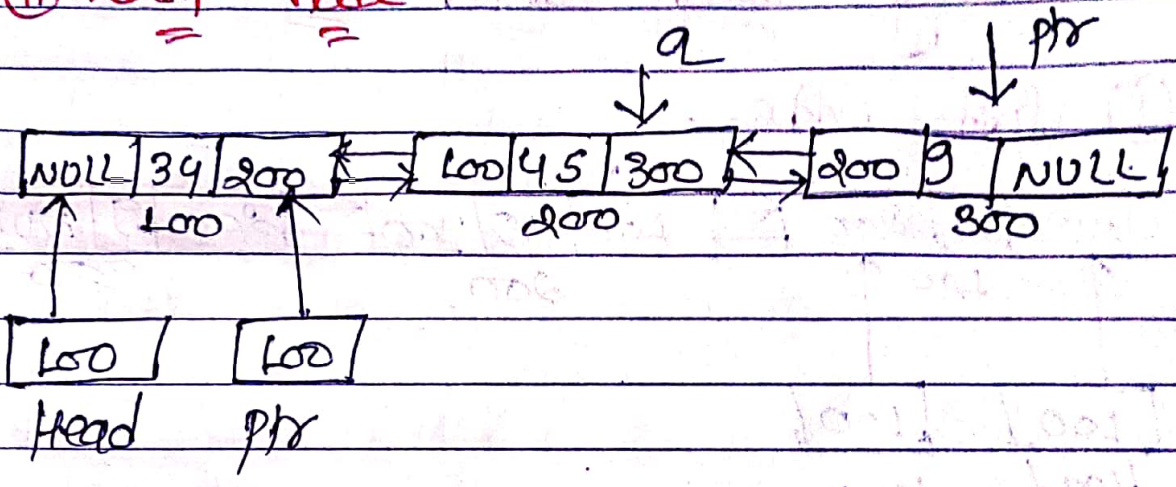
① first node.




```
// method 1.
    head = head->next;
    free(head->prev);
    head->prev = NULL;

// method 2.
    head = head->next;
    free(temp);
    head->prev = NULL;
```

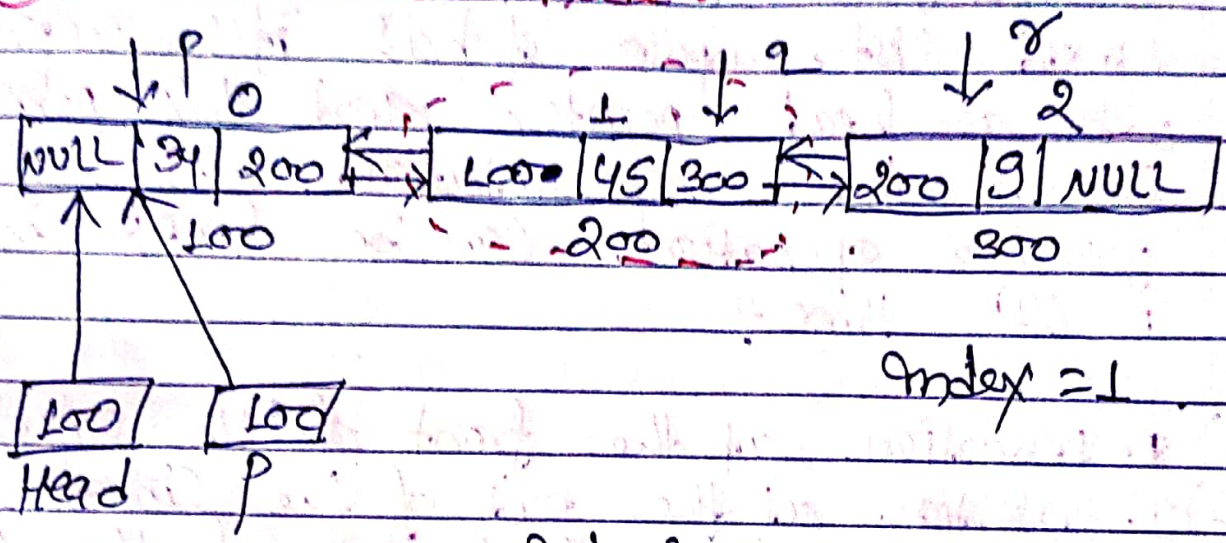
(ii) last node




```
node *ptr=head;
while(ptr->next!=NULL){
    ptr=ptr->next;
}

node *q=ptr->prev;
q->next=NULL;
free(ptr);
ptr=NULL;
```

(iii) Intermediate node :



index = 1

```
int i=0 ;
while( i != index-1 ){
        p = p->next ;
        i++ ;
}
node *q = p->next ;
node * r = q->next ;

p->next = r ;
r->prev = p ;
free (q) ;
q = NULL ;
```

Q: Consider an implementation of unsorted doubly linked list. Suppose it has its representation with a head pointer and tail pointer. Given the representation, which of the following operation can be implemented in O(1) time?

O(1) I. Insertion at the front of the linked list.
O(1) II. Insertion at the end of the linked list.
O(1) III. Deletion of the front node of the linked list.
O(1) IV. Deletion of the last node of the linked list.

a) I and II
b) I and III
c) I, II and III
d) I, II, III and IV. ✓

Q. Consider an implementation of unsorted doubly linked list. Suppose it has its representation with a head pointer only. Given the representation, which of the following operations can be implemented in O(1) time?

I. Insertion at the front of the linked list.
→ O(1)

II. Insertion at the end of the linked list.
→ O(n)

III. Deletion of the front node of the linked list.
→ O(1)

IV. Deletion of the last node of the linked list.
→ O(n)

a) I and II
b) I and III
c) I, II and III
d) I, II, III and IV

Why Doubly linked list ?

→ A doubly linked list is a variation of a singly linked list where each node is also pointing to its previous node.

i. Deletion operation is faster in doubly linked list, if the pointer to the node is given. i.e. In singly linked list traversal is required O(n) but in doubly linked list no traversal required O(1).

ii. Inserting a new node before a given node is easier in doubly linked list. i.e. traversal is required in singly linked list but in doubly linked list no traversal is required O(1).