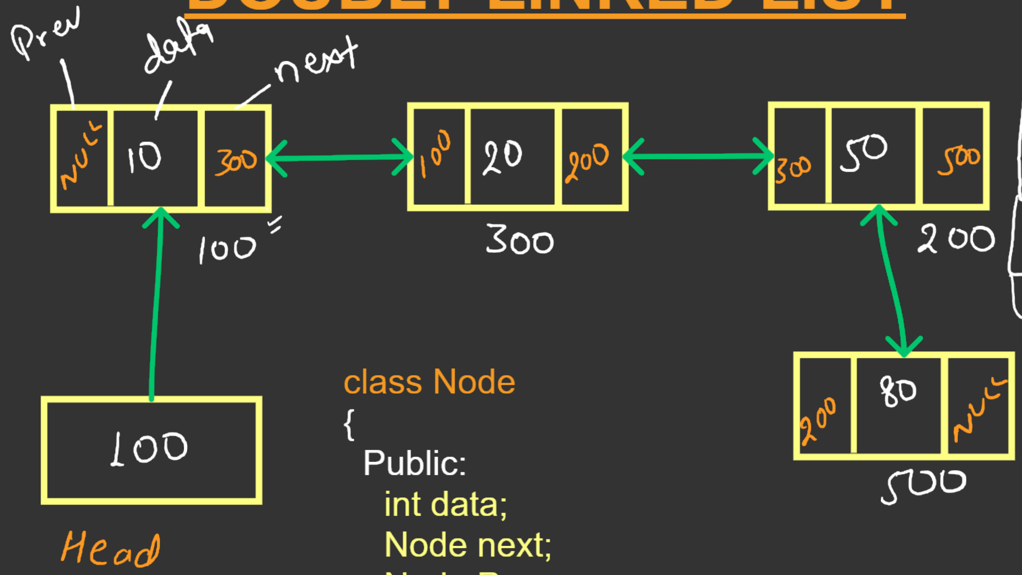
DOUBLY LINKED LIST



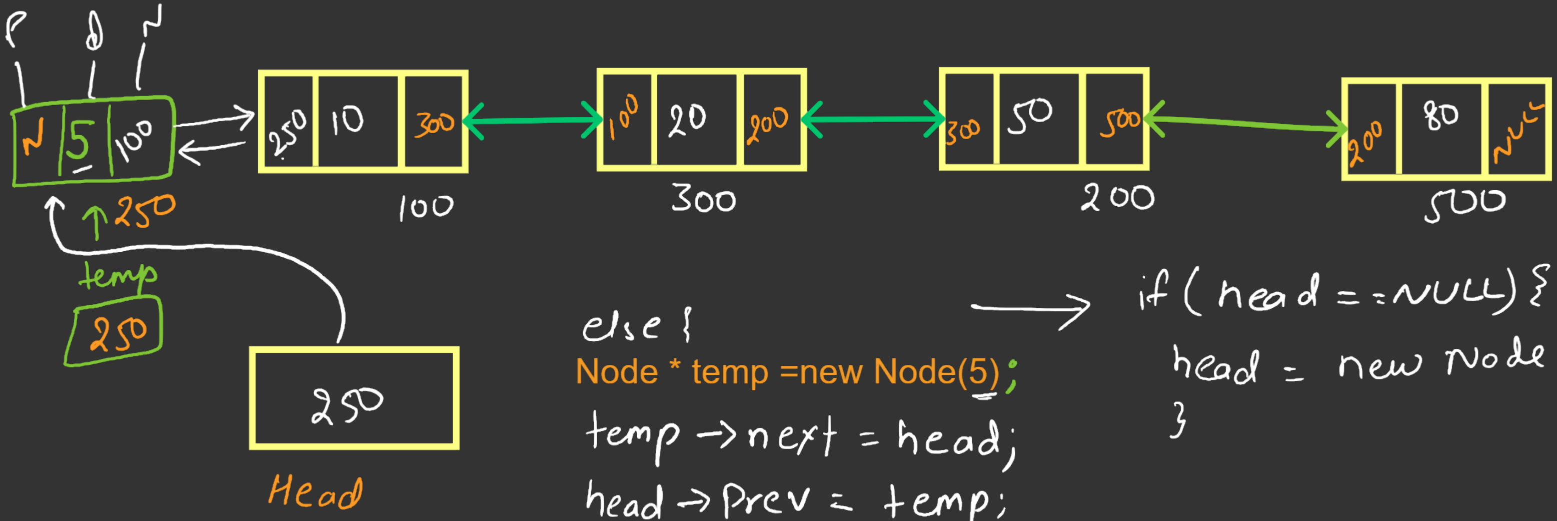
```
class Node
{
  Public:
    int data;
    Node next;
    Node Pre;
};
```

1. Insertion
2. Deletion

- Insert at start
- Insert at end
- Insert at any random position

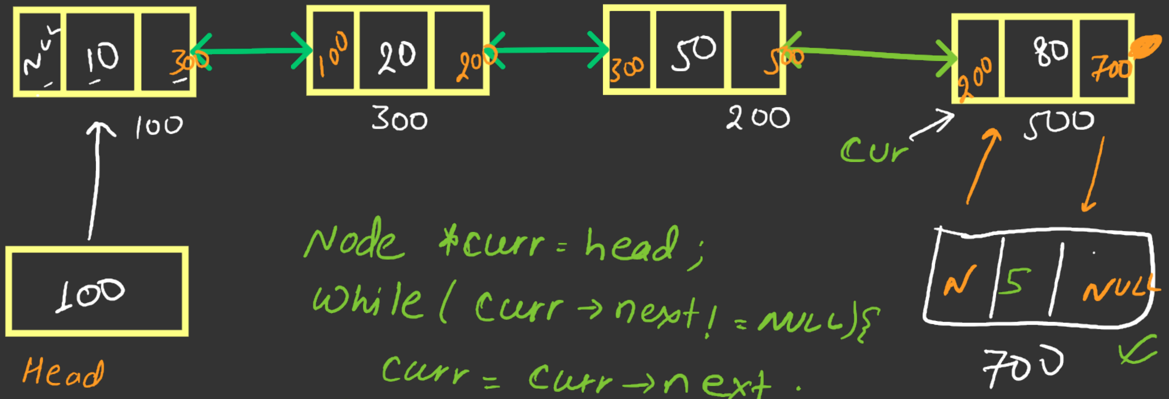
```
class Node
{
  Public:
    int data;
    Node *Prev;
    Node *Next;
    Node(int value)
    {
        data =value; ✓
        Prev = NULL; ✓
        Next = NULL; ✓
    }
}
```

Prev	data	Next

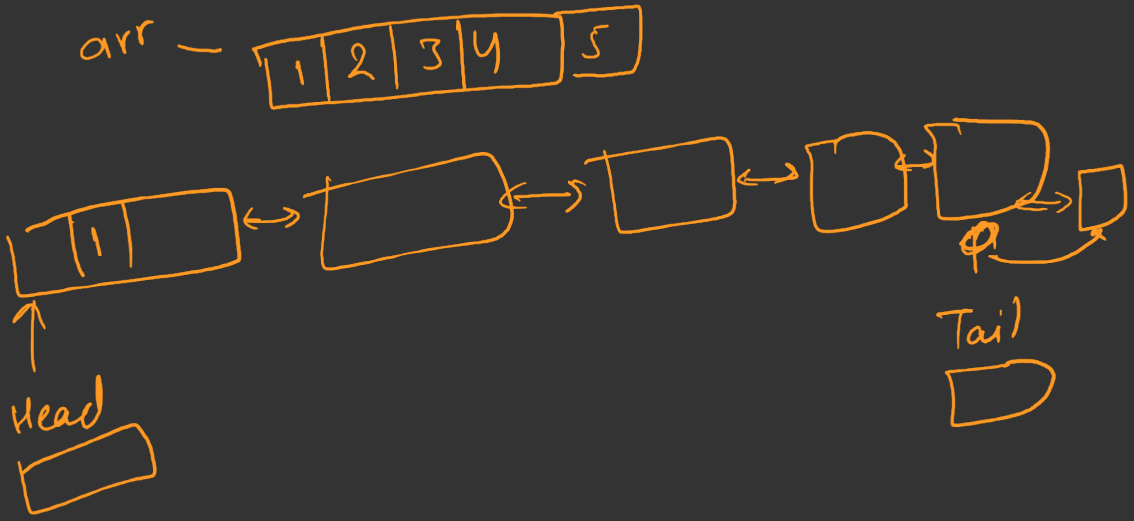


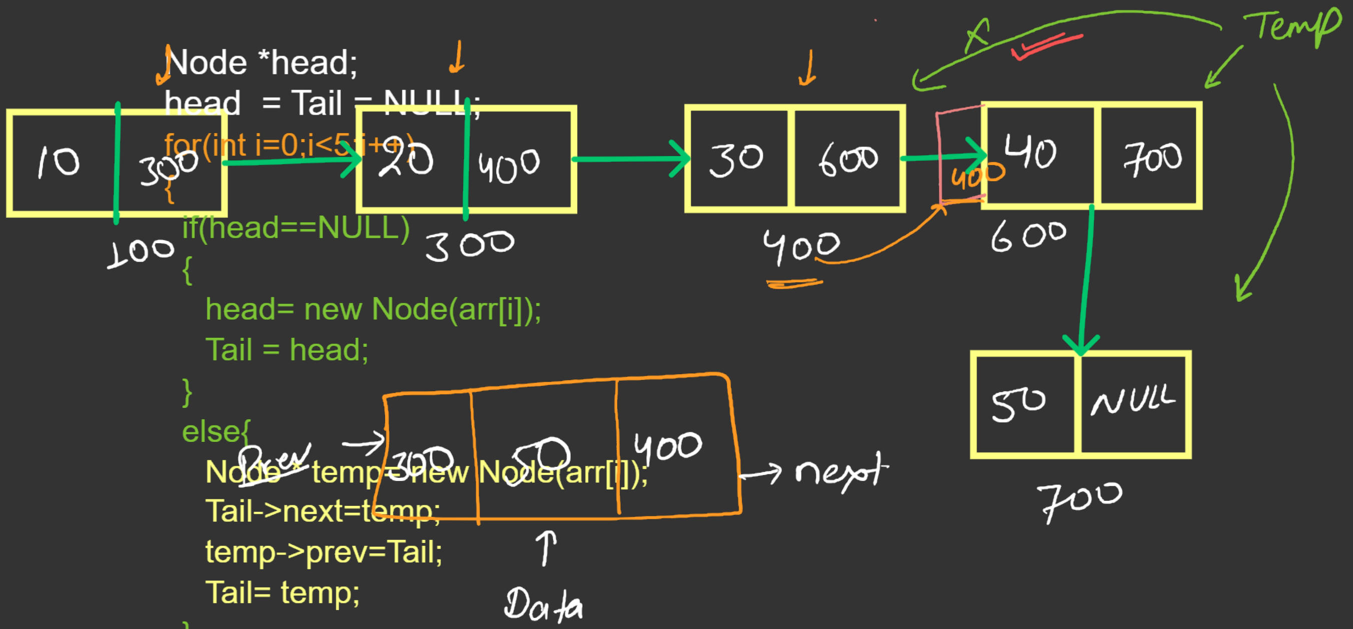
```
else {
Node * temp =new Node(5);
temp -> next = head;
head -> Prev = temp;
} head = temp:
```

```
if ( head == NULL) {
  head = new Node(5);
}
```

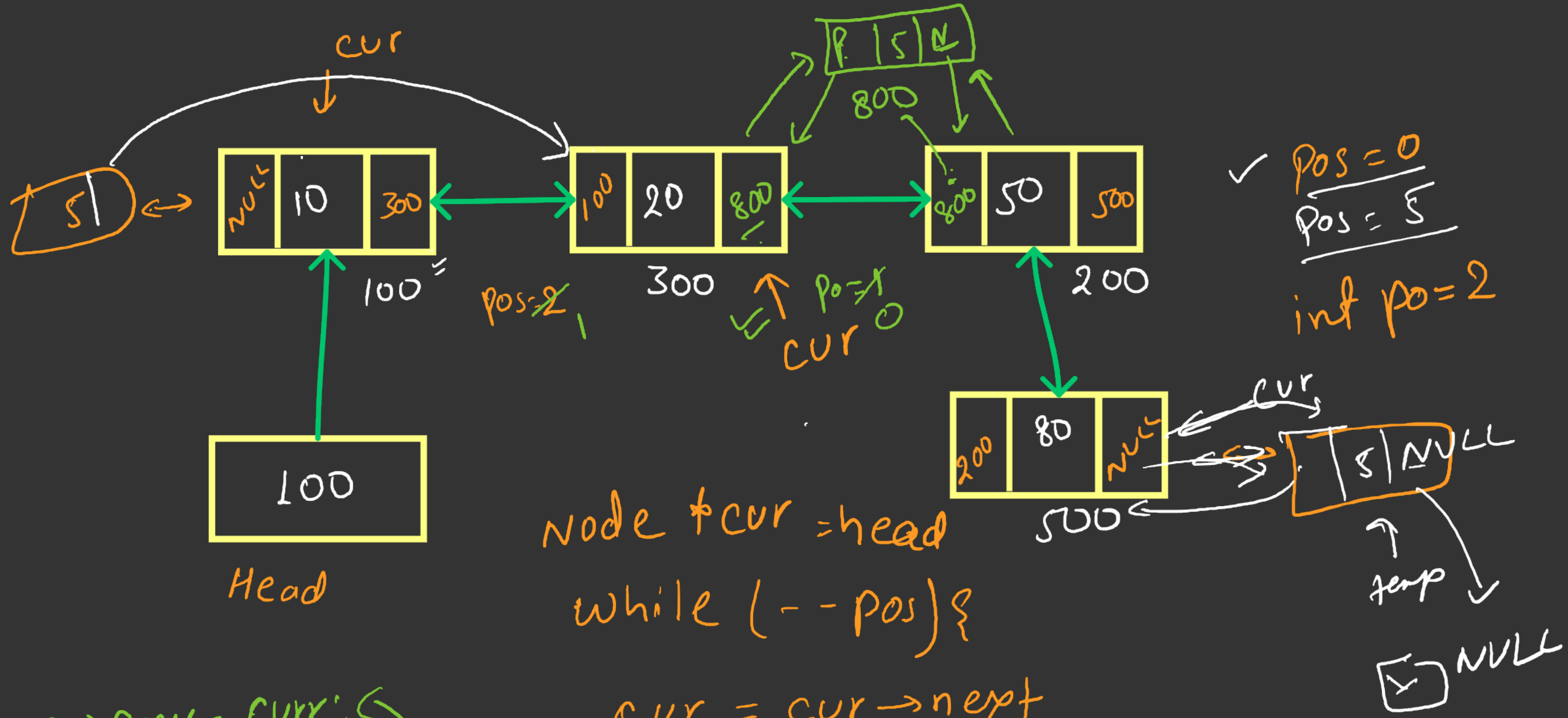


```
Node *curr = head;
while ( curr->next != NULL){
        curr = curr->next;
    }
Node *temp = new node(5);
curr->next = temp;
temp->prev = curr;
}
```


arr
1
2
3
4
5
1
φ
Head
Tail



```
Node *head;
head = Tail = NULL;
for(int i=0;i<5;i++)
{
  if(head==NULL)
  {
    head= new Node(arr[i]);
    Tail = head;
  }
  else{
    Node* temp=new Node(arr[i]);
    Tail->next=temp;
    temp->prev=Tail;
    Tail= temp;
  }
}
```



```
Node *cur = head
while (--pos){
    cur = cur->next
}
Node * temp = new Node(5);
temp->next = curr->next
temp->prev = curr;
curr->next = temp;
temp->next->prev = temp;
```