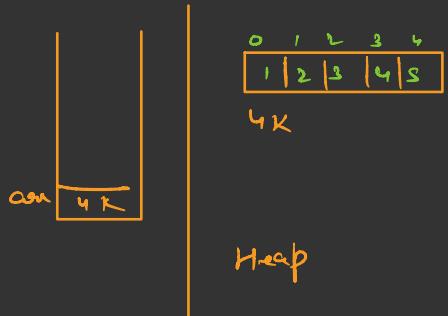


Linked list

int arr[] = new int[5]

20 bytes



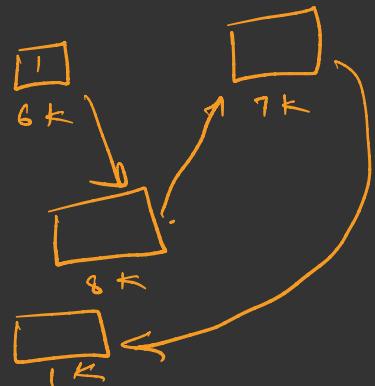
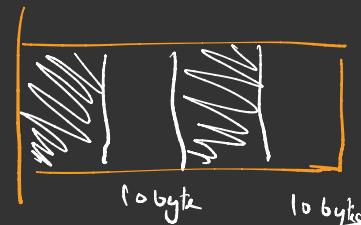
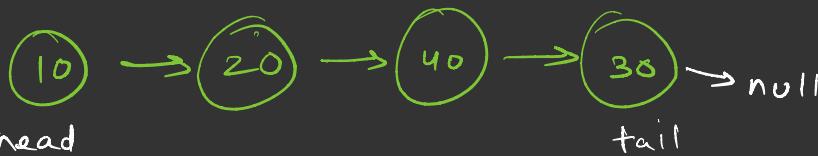
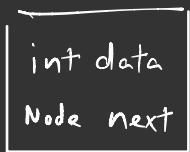
LL was introduced



↓
Node

- ① data
- ② reference to next node

Node



Class LL {

Node head

Node tail

int size

3

add

addFirst (int val)

addLast (int val)

add (int val, int idx)

get

remove

set

addFirst (u)

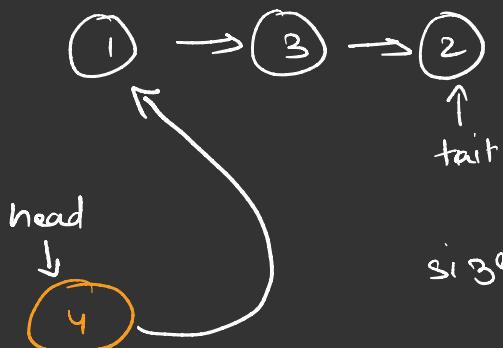
1. Create new node

2. point next to

head

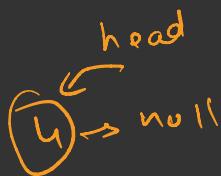
3. point head to new node

4. increase the size



size = 3

size = 0

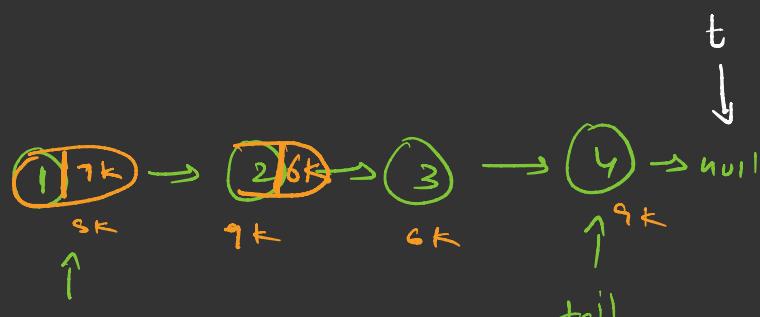


head = null
tail = null
size = 0

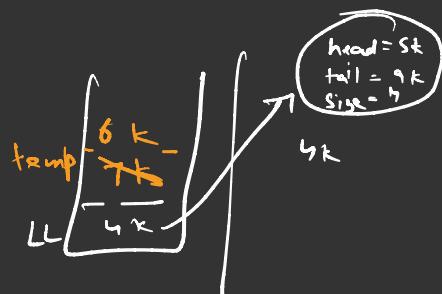
if (tail == null)
tail = n

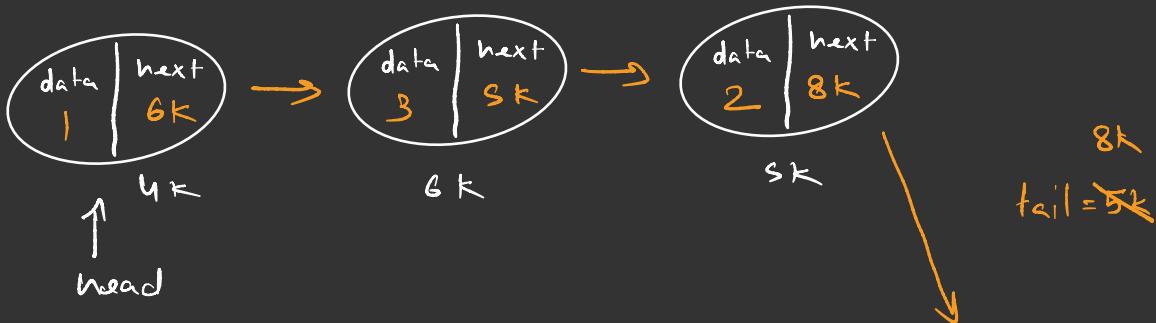
display

1 → 2 → 3 → 4 →



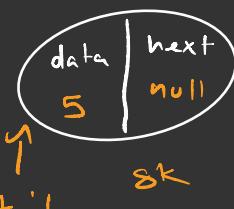
```
public void display() {  
    Node temp = head;  
    while(temp != null) {  
        System.out.print(temp.data + " ->");  
        temp = temp.next;  
    }  
    System.out.println("null");  
}
```





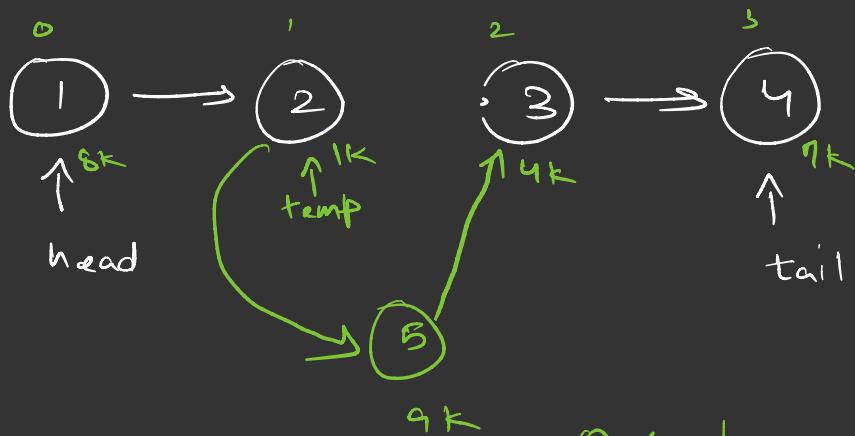
addLast(3)

- 1) Create new node
- 2) tail.next should point to new node (n)
- ③ tail should point to n
- 4) Increase size



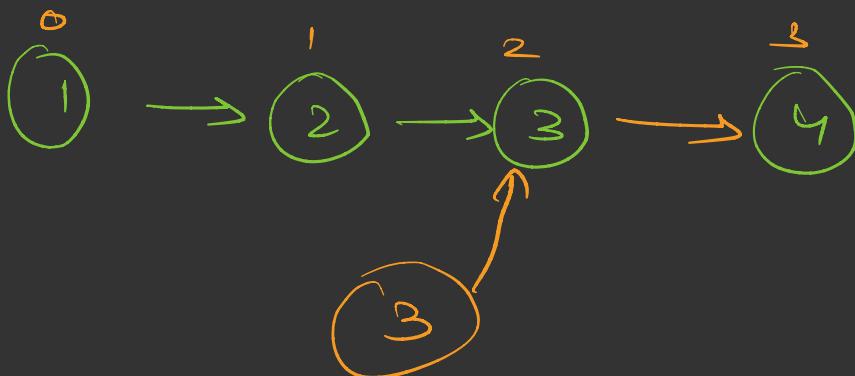
`add(int val, int idx)`

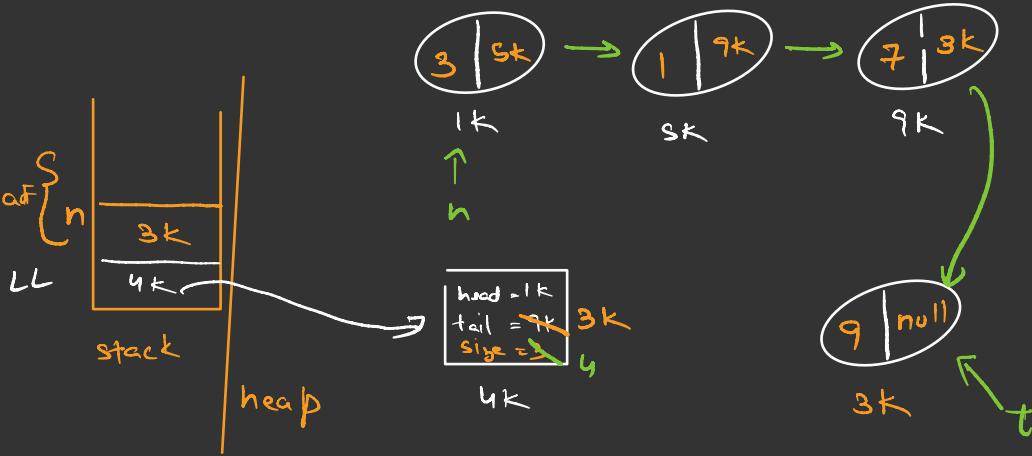
`add(s, 2)`



`temp = get(idx - 1)`

- ① Create
- ② `get(idx - 1)`
- ③ `n.next = temp.next;`
- ④ `temp.next = n`
- ⑤ `size++`

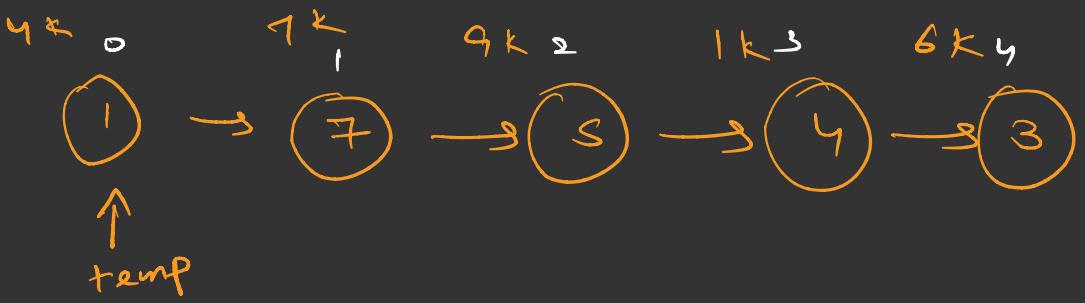




```
public void addLast(int data) {
    if(this.size == 0) {
        this.addFirst(data);
        return;
    }
    Node n = new Node(data);
    this.tail.next = n;
    this.tail = n;
    this.size++;
}
```

`get(iidx)`

`getNode(iidx)`

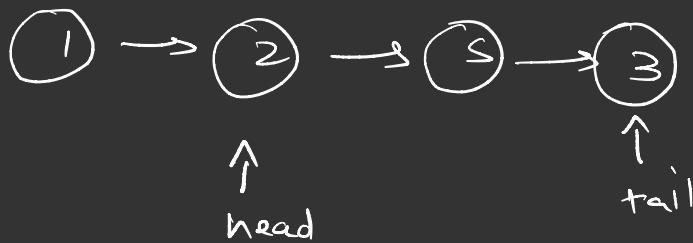


Count = 0

idx = 3

$temp = head = 4k$

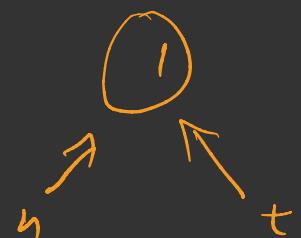
Remove First

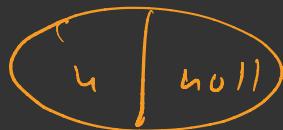


① Move head

② reduce size

$size = 1$



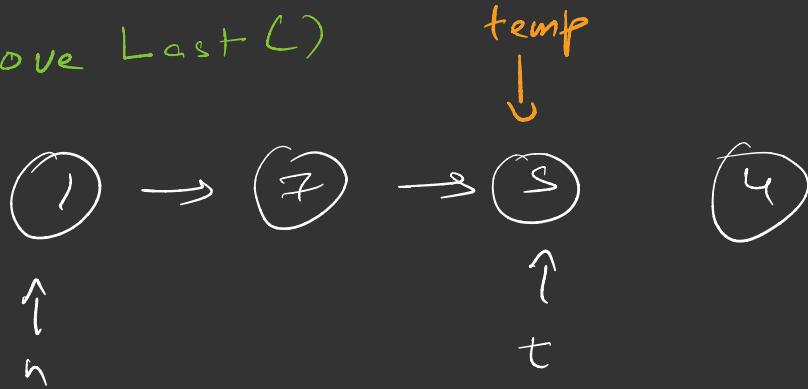


$n \rightarrow null$

$t = null$

$s--;$

remove Last()



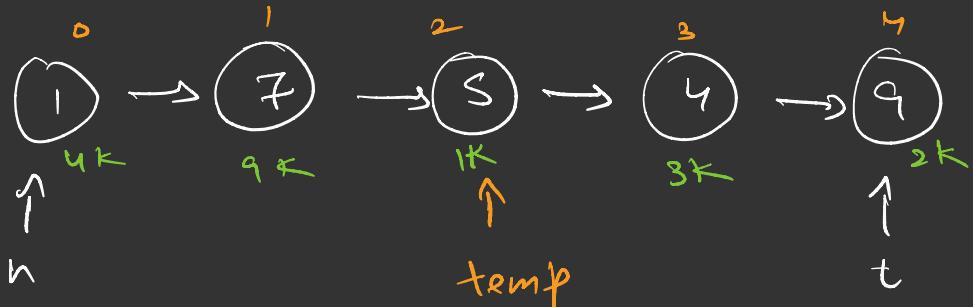
① $temp = getNode(size - 2)$

② $temp.next = null$

③ $tail = temp;$

④ $.size--$

Remove(int idx)



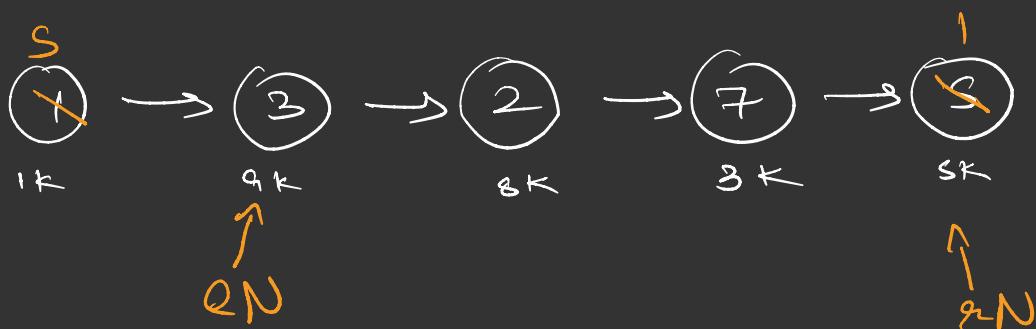
remove(3)

① get Node ($idx - 1$) -

$temp.next = temp.next.next$

Reverse Data Iteratively

Reverse DI



$$Q = \emptyset$$

$$x = 1$$

Reverse

Data

Painter