Do participate in the contest tomorrow! will share the link on Discord. Time ⇒ 4:00PM

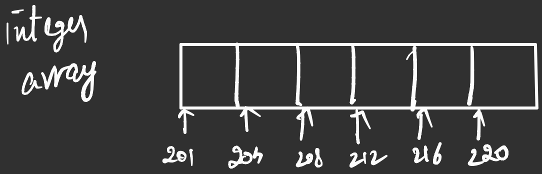
L8

Linked List : Intuition and Implementation

Join Discord - https://bit.ly/ly-discord

RECAP

Let's start with Arrays



1.) Random access

⇓

$O(1)$

2.) Insert/Delete from an index

⇓

$O(N)$

What’s the problem?

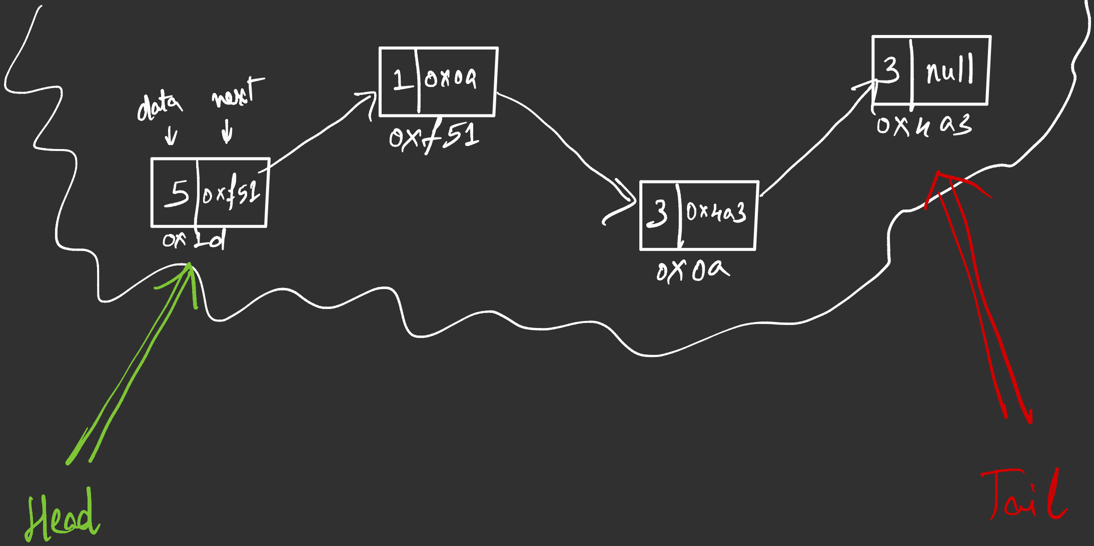
1.) Insertion / Deletion takes time.

2.) The space required for an Array needs to be continuous

Deleting a picture from a gallery of 1000s of pictures and 100s of videos

Let’s understand Linked List

The idea:
A bunch of nodes that contain data, and those are **linked** to each other **sequentially**


data
next
5
0xf51
0x1d
1
0x0a
0xf51
3
0x4a3
0x0a
3
null
0x4a3
Head
Tail

Everything has its goods & bads

Good ⇒ Insertion /Delete got better

Bad ⇒ Random Access takes time.

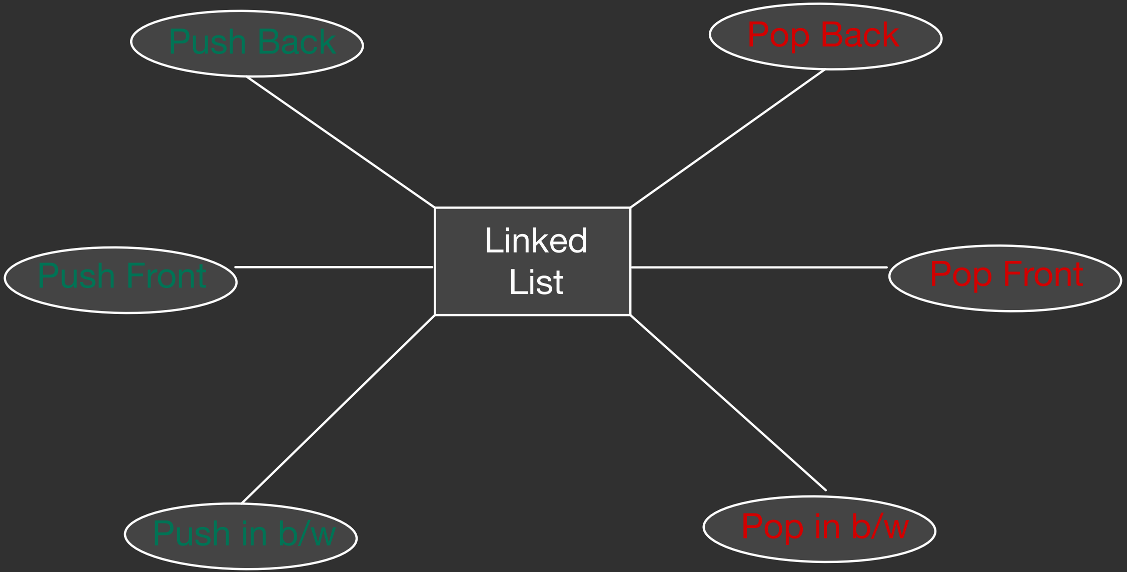
The Structure

Java

```
class Node {
    int data;
    Node next;
}

class LL {
    Node head;
    int length;
}
```

C++

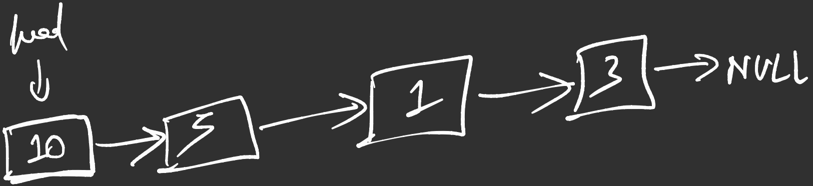
```
class Node {
    int data;
    Node * next;
}

class LL {
    Node * head;
    int length;
}
```


Push Back
Pop Back
Push Front
Linked List
Pop Front
Push in b/w
Pop in b/w

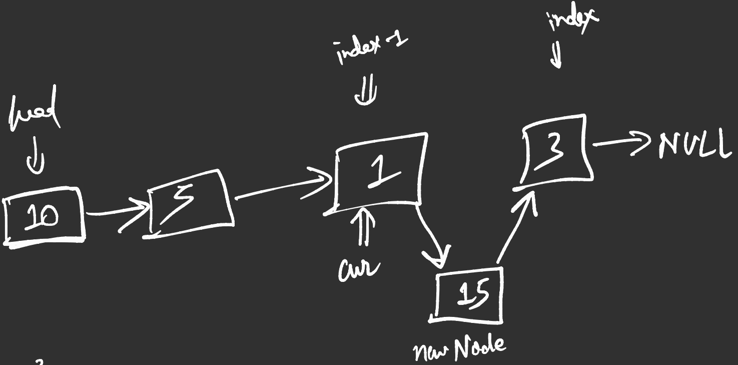
```
Class X {
    Int x;
  public:
    void update(Int x) {
      this->x = x;
    }
}
```

Let’s implement

addAtHead()

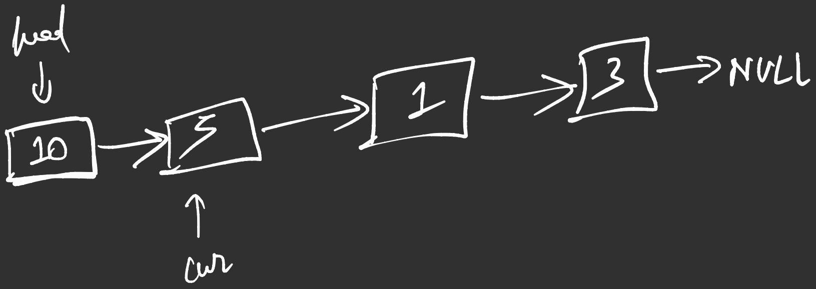


```
newHead.next = this.head;

this.head = newHead;
```



Index = 3
val = 15

(Insert at Index)

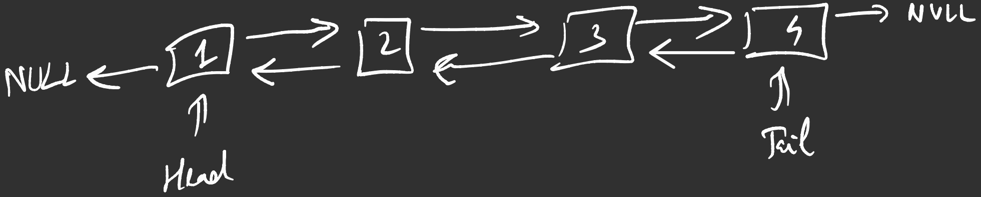


- index = 2

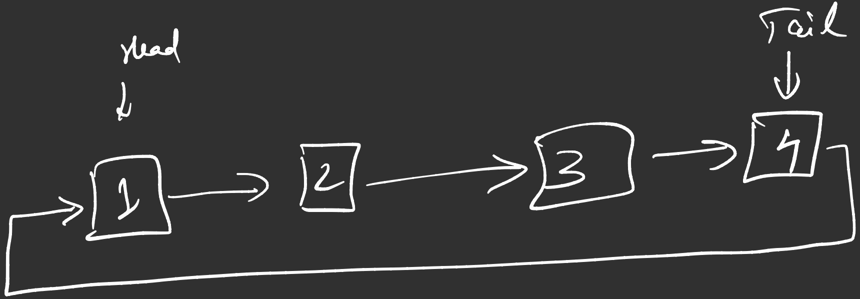
(Delete at Index)

Other types?

1) Doubly LL



2.) Circular LL



Thank You!

Reminder: Going to the gym & observing the trainer work out can help you know the right technique, but you'll muscle up only if you lift some weights yourself.

So, PRACTICE, PRACTICE, PRACTICE!