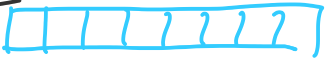
04 November 2018 00:28

Linked List

→ Linear data structure

→ A linked list is a dynamically allocated DS.

arr

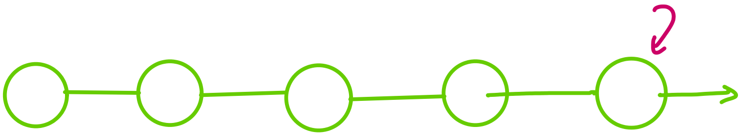


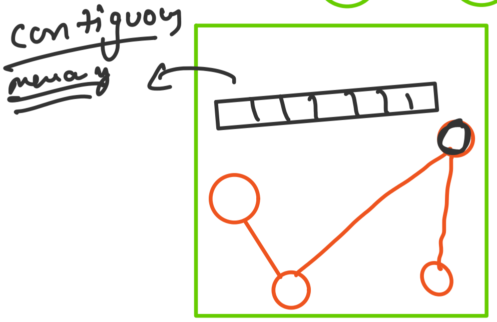
→ arrays

↳ contiguous linear block of memory

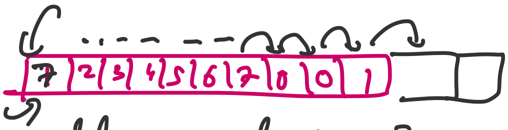
direct access to the indices of array

arr[1] arr[3]



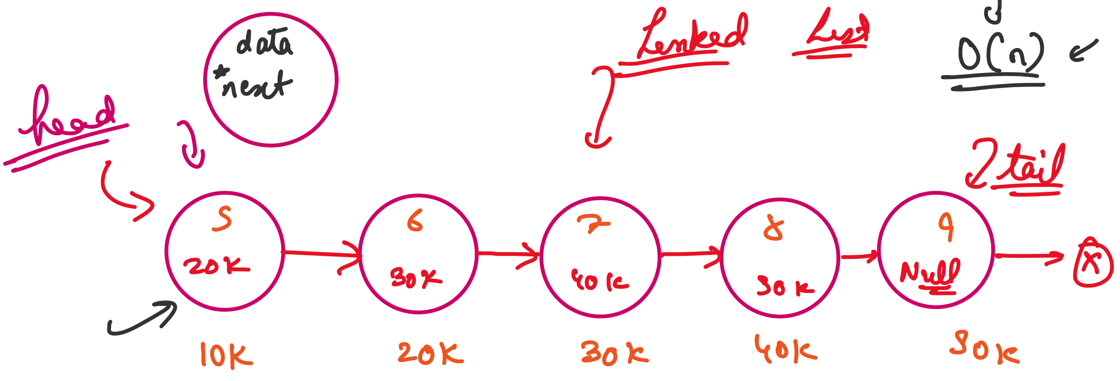


Scattered form inside the memory



↪ add new element → 7

O(n)



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

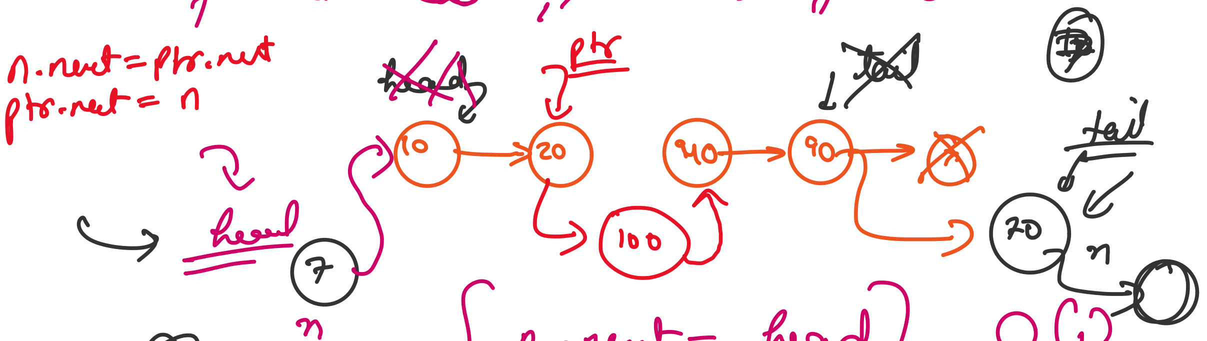
new object

malloc

new Node()

In a linked list, we have the access

for the head, sometimes for tail



[n.next = head
head = n]

O(1)

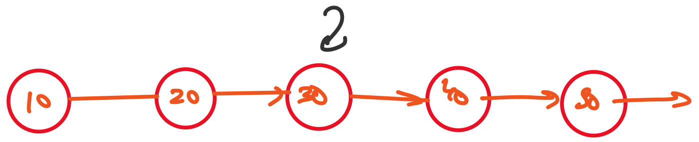
K

O(K)   O(n)

Can I add at tail → [tail.next = n
tail = n] → O(1)

[temp = head
traverse   temp.next==null] → O(n)

Q>



Middle Node of Linked List

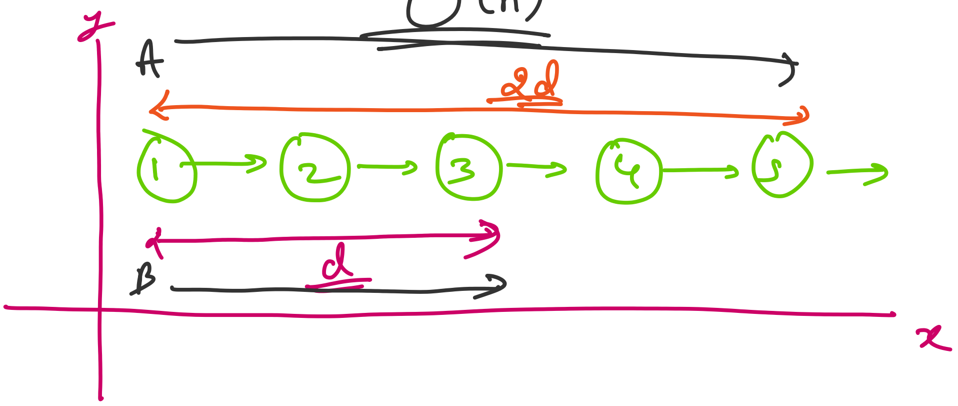
2 min

n/2

Brute force Sol^n → calculate the length of the linked list in one pass   n

→ Do one more traversal, but only n/2 time & the encountered node, will be middle node

O(n)



Suppose I want to have 2 runners

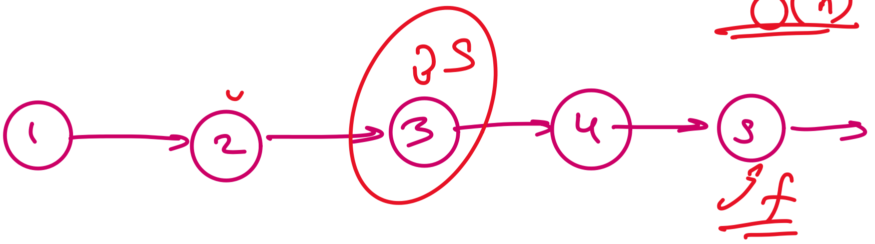
2x ← A → last node

x — B → middle node

} → same time

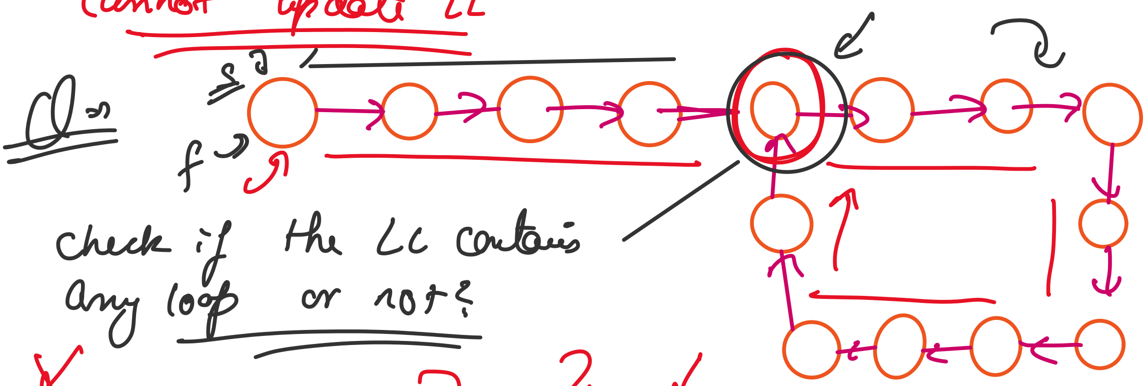
O(n + n/2)

O(n)



speed of fast = 2 × speed of slow

Cannot update LL



check if the LL contains any loop or not?

~~value~~

address

? ? ?

unordered_map

visited

O(n)

Object → HashMap

O(1)

Space O(n)

Floyd cycle algorithm

4

dist → 3

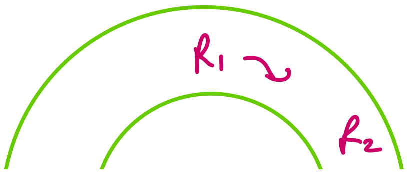
<0

S → S.next ?

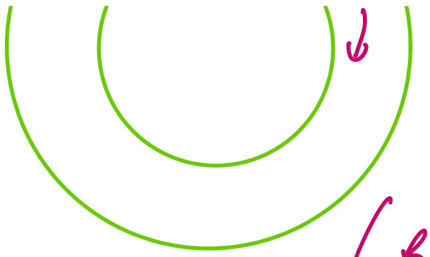
f → f.next.next ?

Why ?

[speed of fast = 2 × speed of slow]

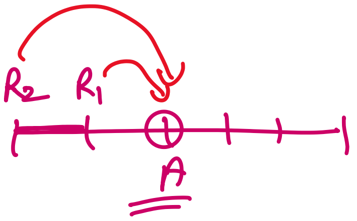


$S R_2 = 2 \times S R_1$



3

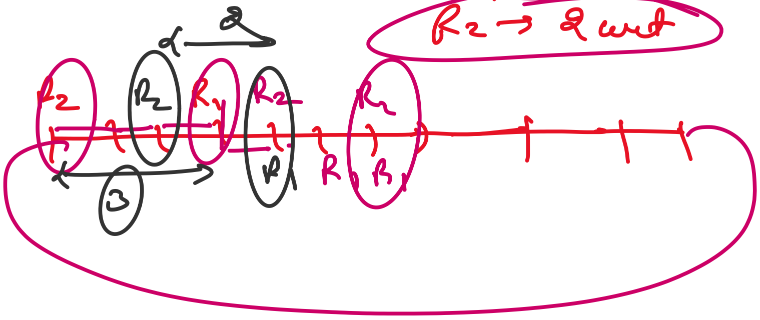
4



R2 will either overlap R1 or cross R1

R1 → 1 unit

R2 → 2 unit



d

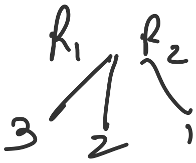
R1 & R2

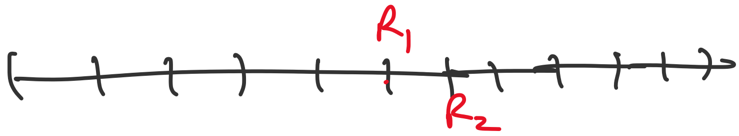
Speed of R2 = 3x speed R1

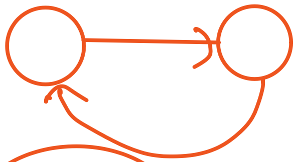
Speed → twice

4, 5

d = decreasing → 1







d x

1

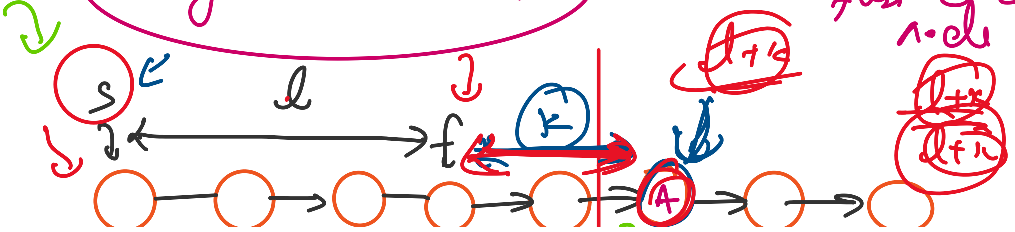
2

d ≠ 0

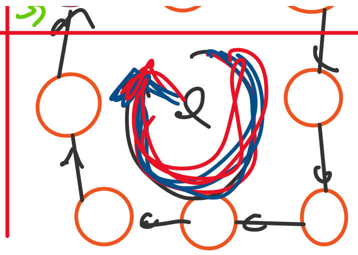
< 0

why this works?

k → distance between fast & start loop node



if now both slow & fast moves by 1, well they meet



cool

f = f.next

length of cycle of LL is 'l'

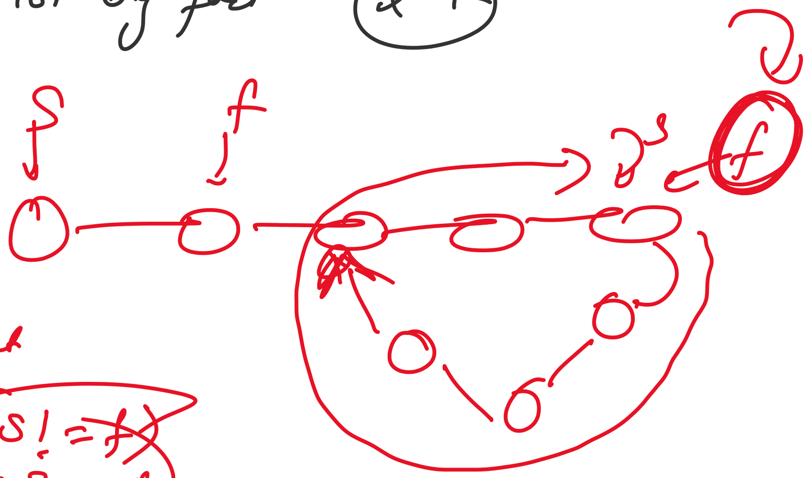
2 case → fast is at right of axis
→ fast is at left of axis

Dist by slow = l−K

Dist by fast = l−K

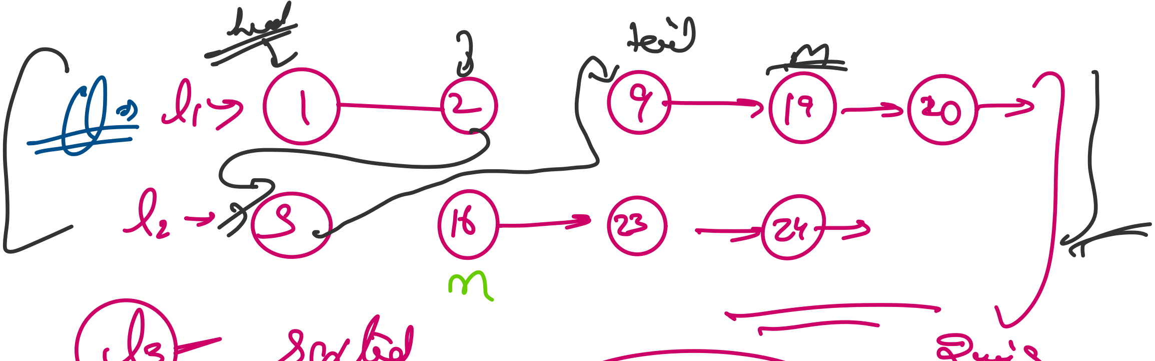
O(n)

O(1)



s = s.next
f = f.next
wh(s != f)
s = s.next

Hints head



l3 sorted

merge sort

Quiz

O(1)

O(n+m)

O(1)

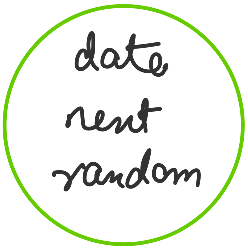
ascending

descending order

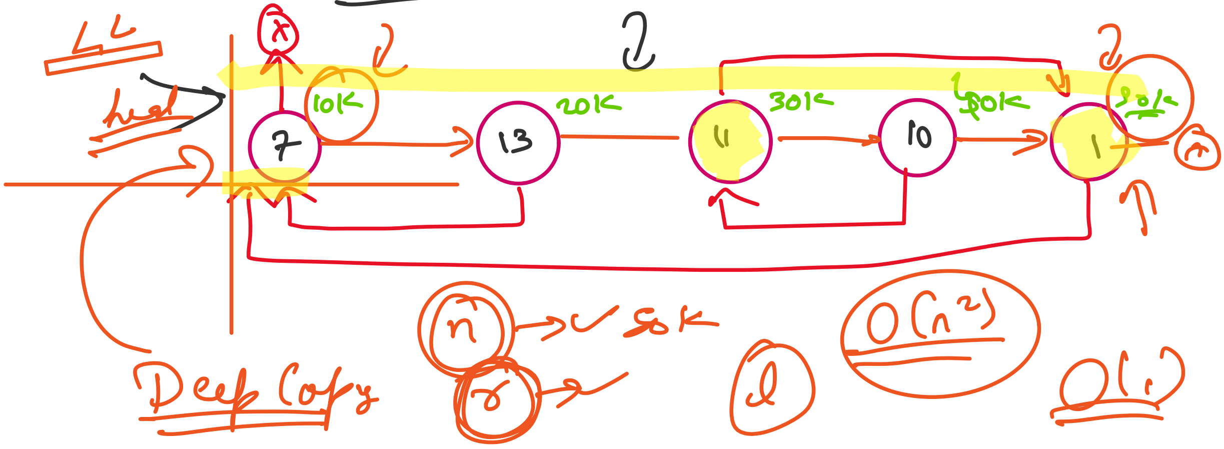
H.W

1 → 2 → 5 → 9 → 16 → 19 → 20 → 23 → 24

Q) New kind of linked list



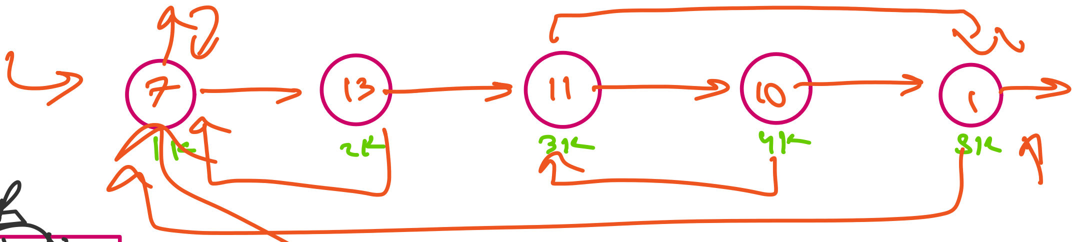
Node



Deep Copy

$O(n^2)$

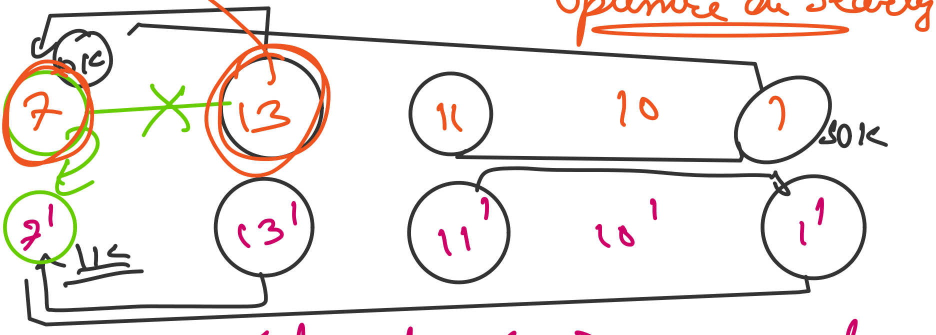
$O(1)$



map

10K	1K
20K	2K
30K	3K
40K	4K
50K	5K

Optimize on Search



mapping old node ⟷ new node

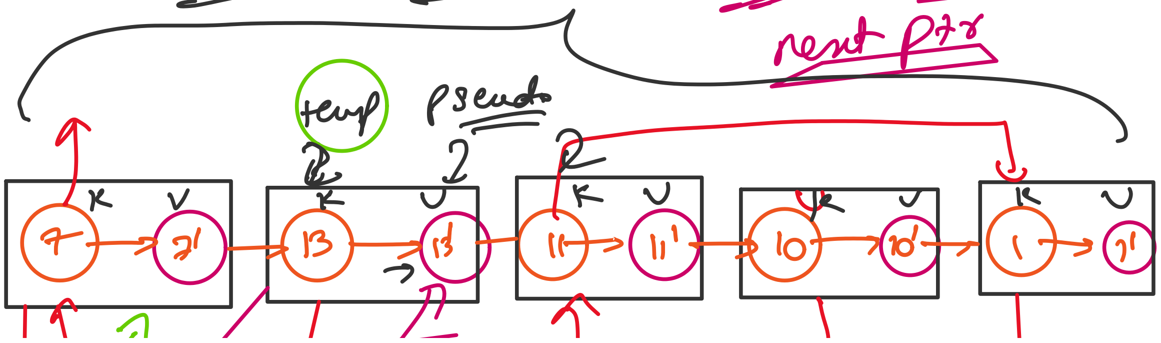
Random of 13' =
value in HM (Random of 13)

value of (50K) → 5K

$O(n)$ $O(n)$

value HM

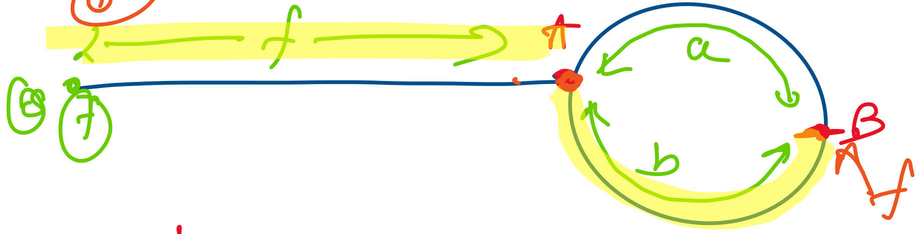
next ptr



Random of pseudo = Next of Random of temp

```
if (head == null) return null;

Node temp = head;
while (temp != null) {

    Node random = temp.next;
    Node n = new Node (temp.val)
    temp.next = n
    n.next = random
    temp = temp.next.next
}

temp = head
while (temp != null) {
    Node pseudo = temp.next;
    if (temp.random != null) {
        pseudo.random = temp.random.next;

    }
    temp = temp.next.next
}
```

sanket@interviewbit.com



S's dist = $f + a$

$f = b$

f's dist $= f+a+b+a$

$f+a+b+a+b+$

$f+a+b+a = 2(f+a)$

$f+\cancel{a}+b+\cancel{a} = 2f+\cancel{2a}$

$2f - f = b$

$\boxed{f=b}$