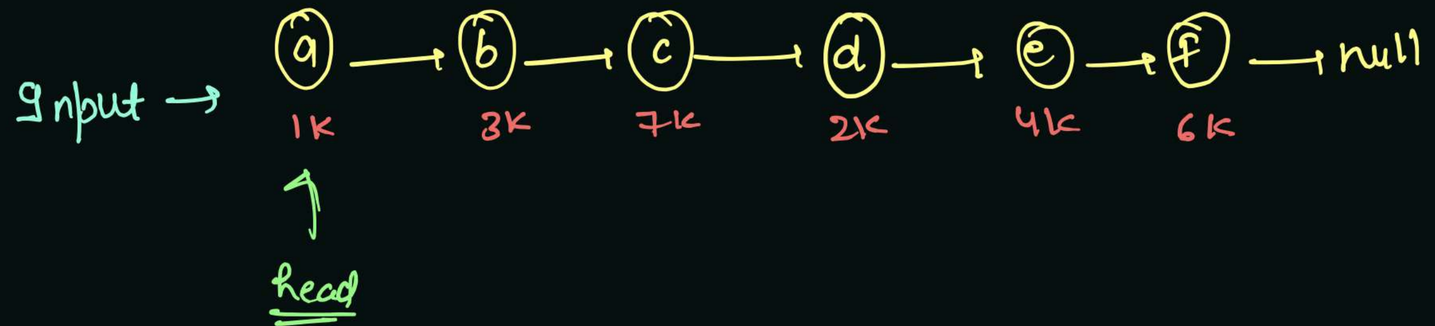
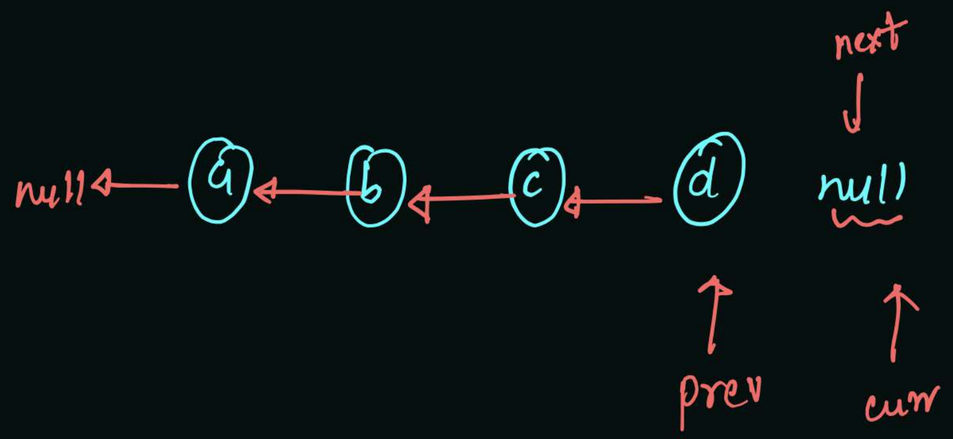
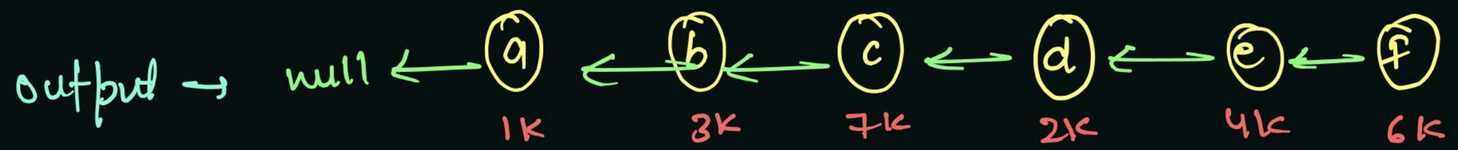
Reverse A LinkedList

Saturday, 14 August 2021 2:27 PM





new head = prev.



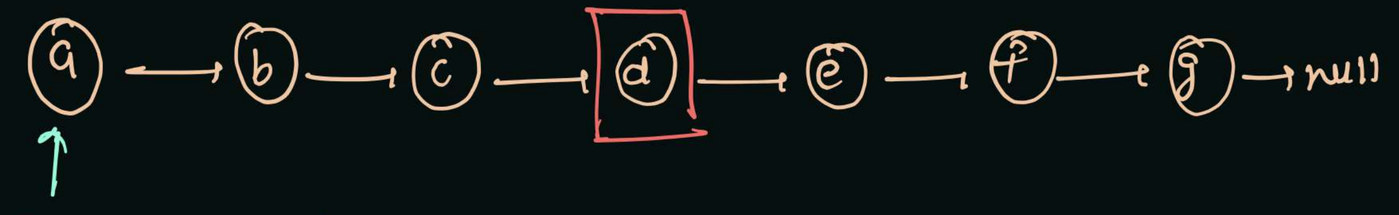
task →

```
while(curr != null) {
    next = curr.next;
    curr.next = prev;
    prev = curr
    curr = next
}
return
```

Middle Of A Linked List

Saturday, 14 August 2021 2:31 PM

Odd size

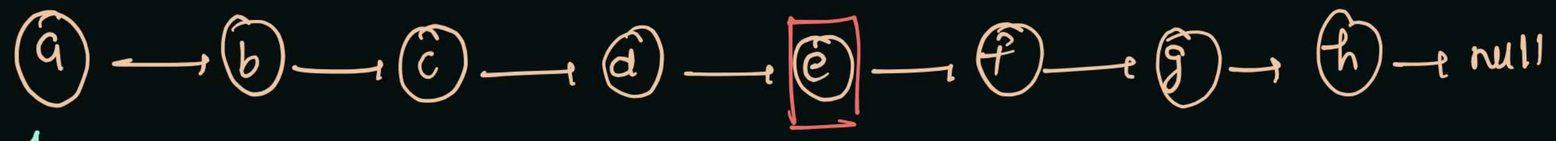


Odd → Middle Element

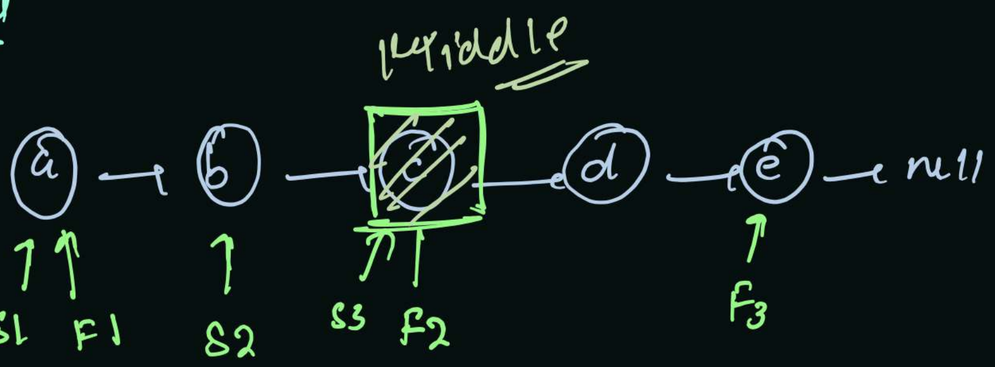
Mid 2:

Leetcode

Even size-



Even → first Element of Second half



Slow = head = speed x

Fast = head → speed 2x

Speed of slow = x

Speed of fast = 2x

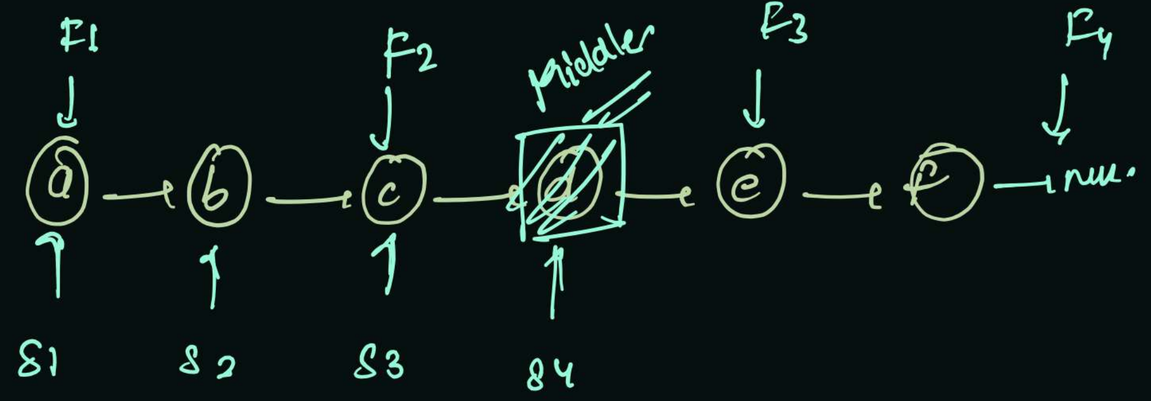
distance travel by slow in T time = x × T

distance travel by fast in T time = 2x × T.

⇒ fast is at End, then slow is at Middle.

slow = a

fast = a



Running condition

```
while( fast != null &&
       fast.next != null)
{
}
```

For Mid-1

Mid-1

Portal

→ odd size → middle is obvious.

→ Even size → Last element of first half is middle >

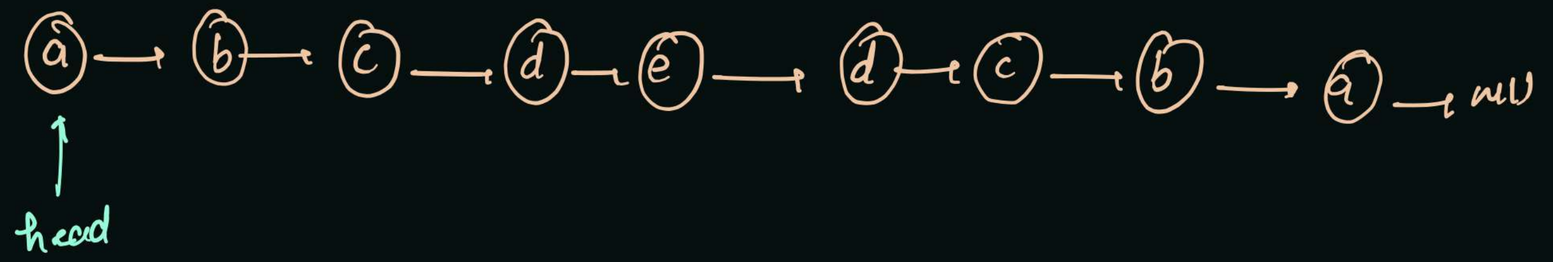
slow = head

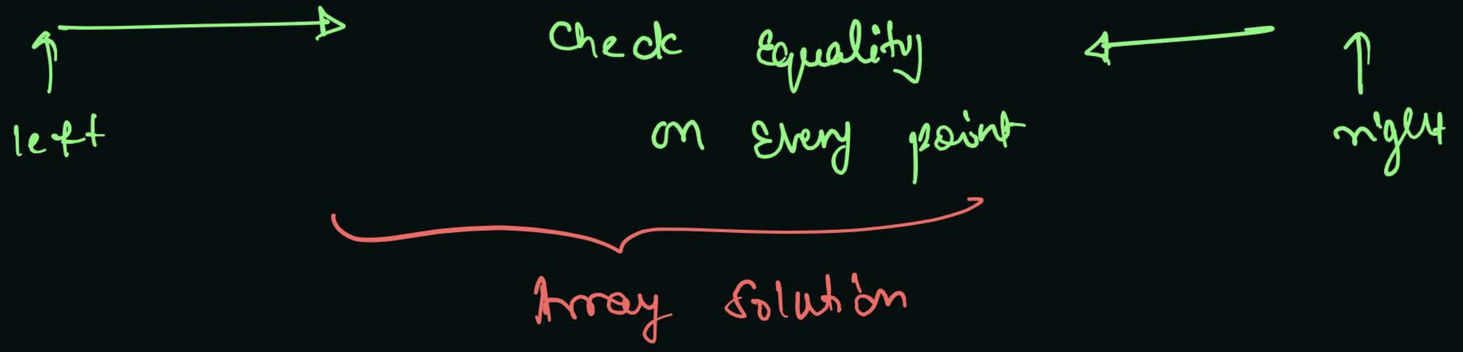
fast = head.next

Palindrome a Linked List

Saturday, 14 August 2021 2:31 PM

order of character is same from both direction

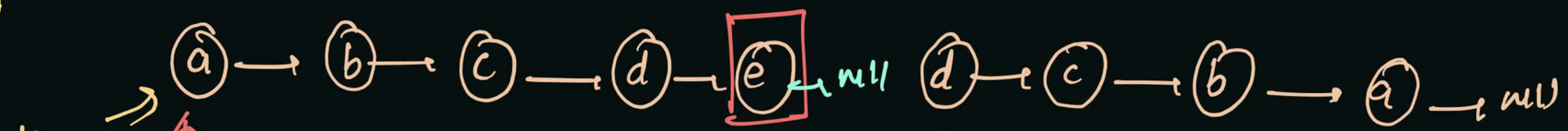




Reverse traversal in linkedlist

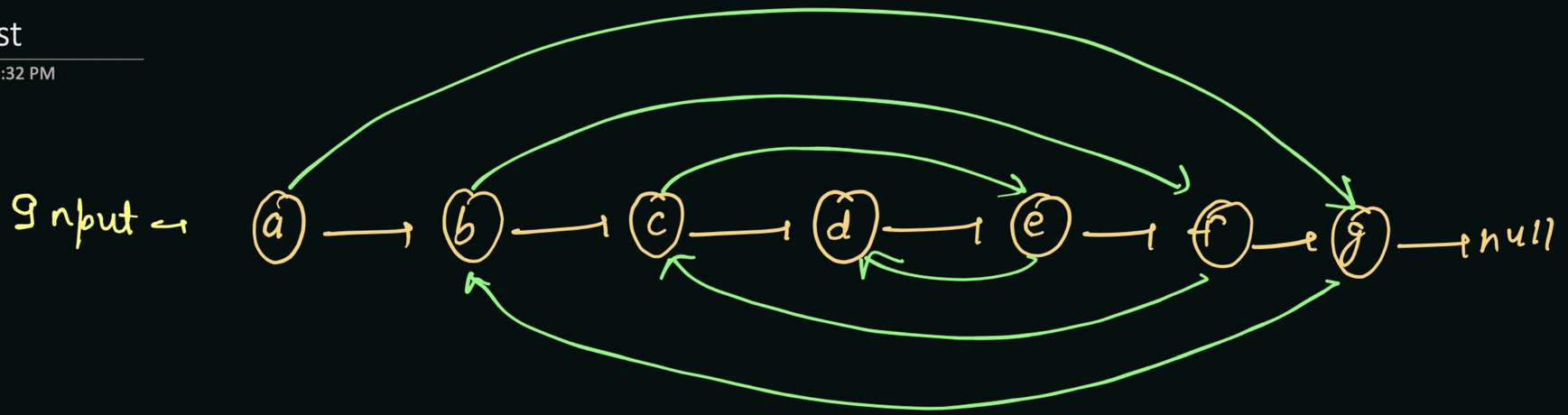
half

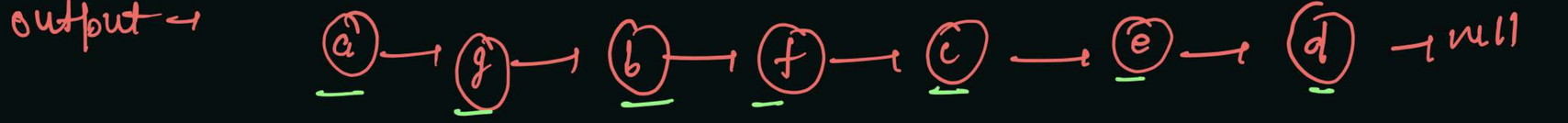
Break from middle and Reverse second half

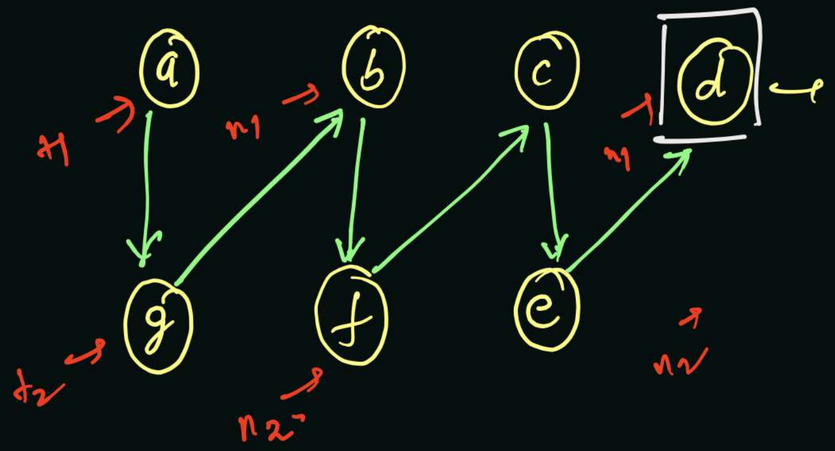


Fold of Linked List

Saturday, 14 August 2021 2:32 PM

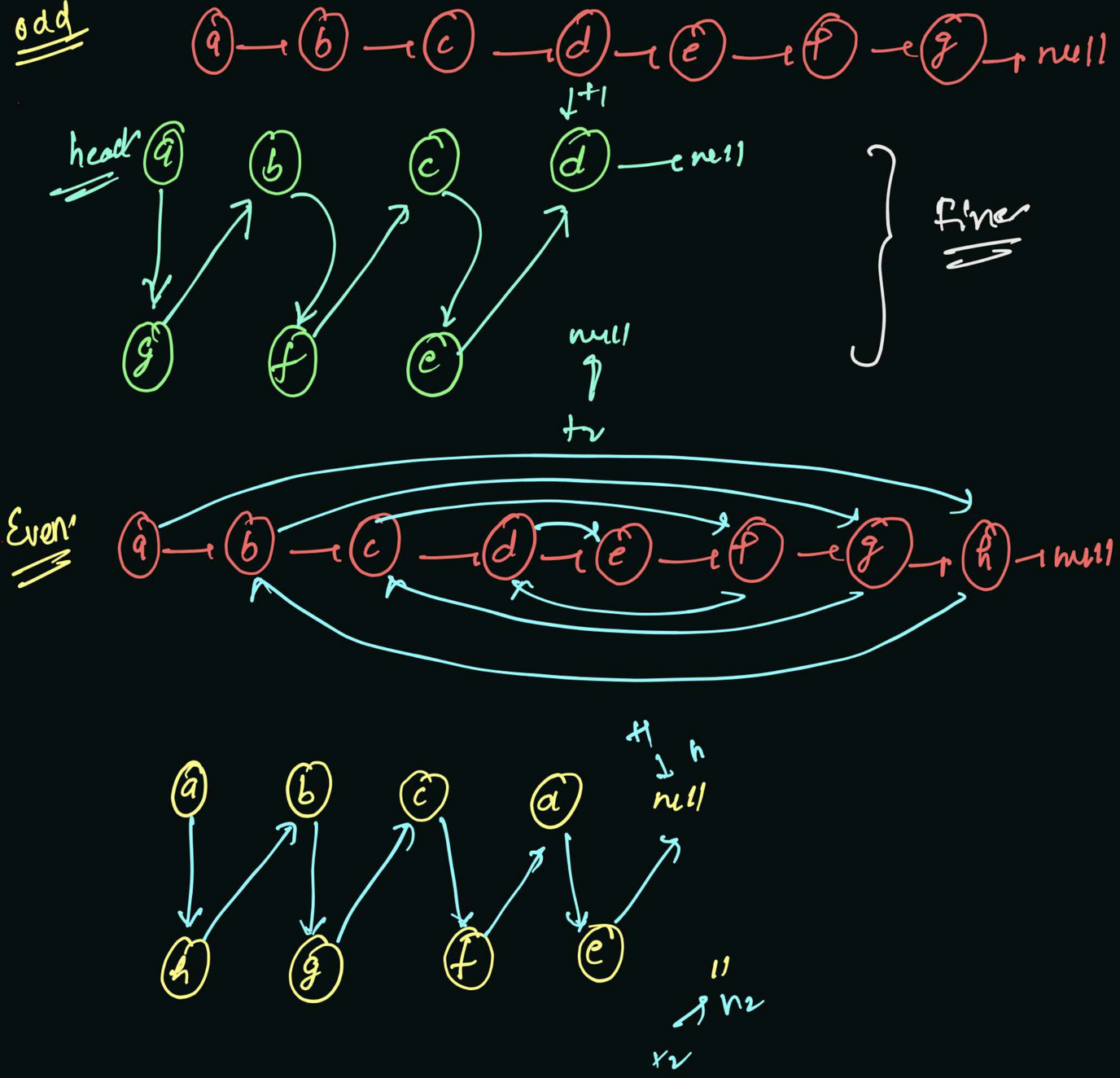






Steps:

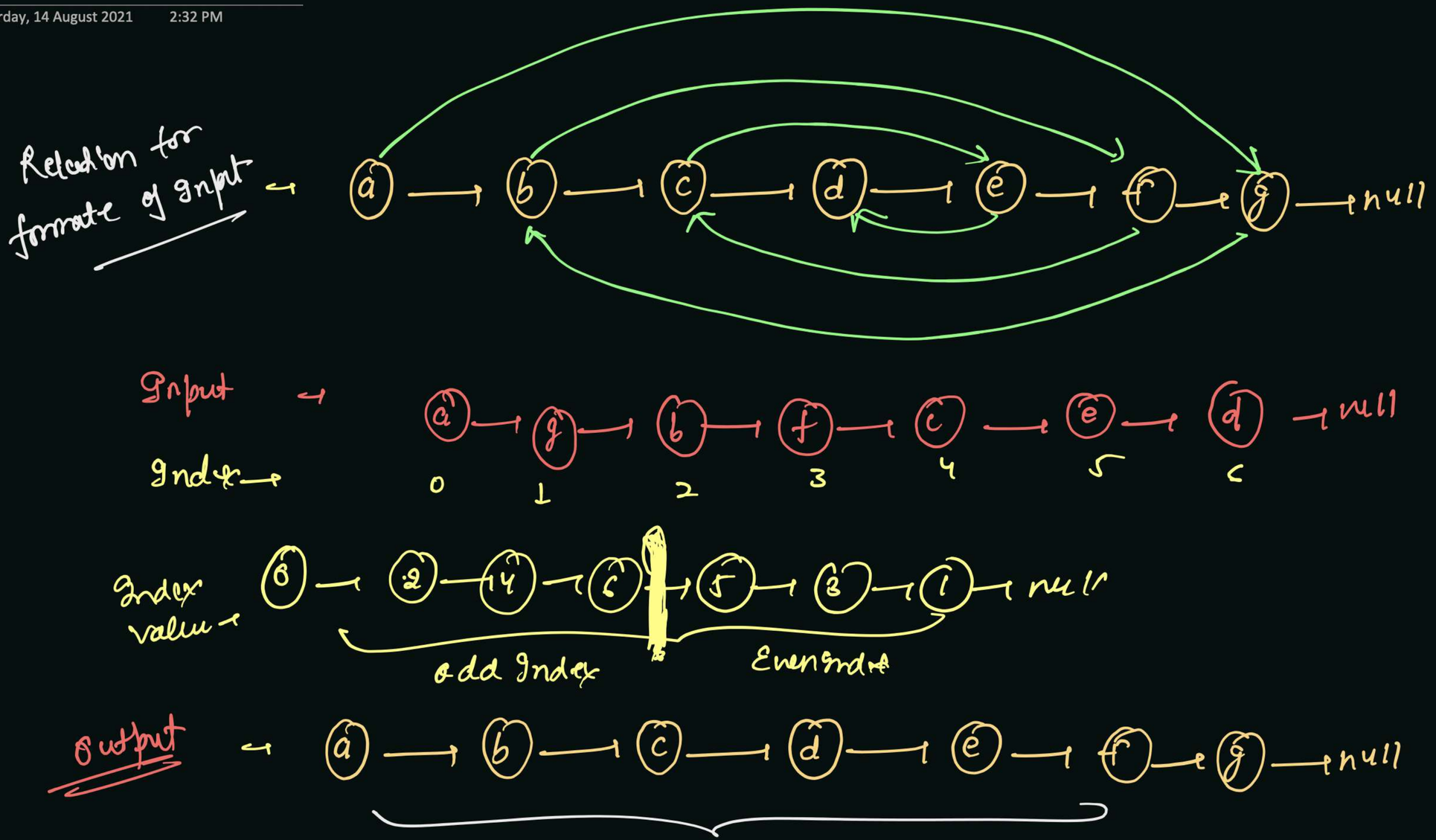
1. Find mid → middle
2. Reverse second half
   - head2
   - mid.next = Null
   - Reverse head2
3. Make a zig zag connection.
   - t1.next = t2
   - t2.next = n1
   - t1 = n1
   - t2 = n2

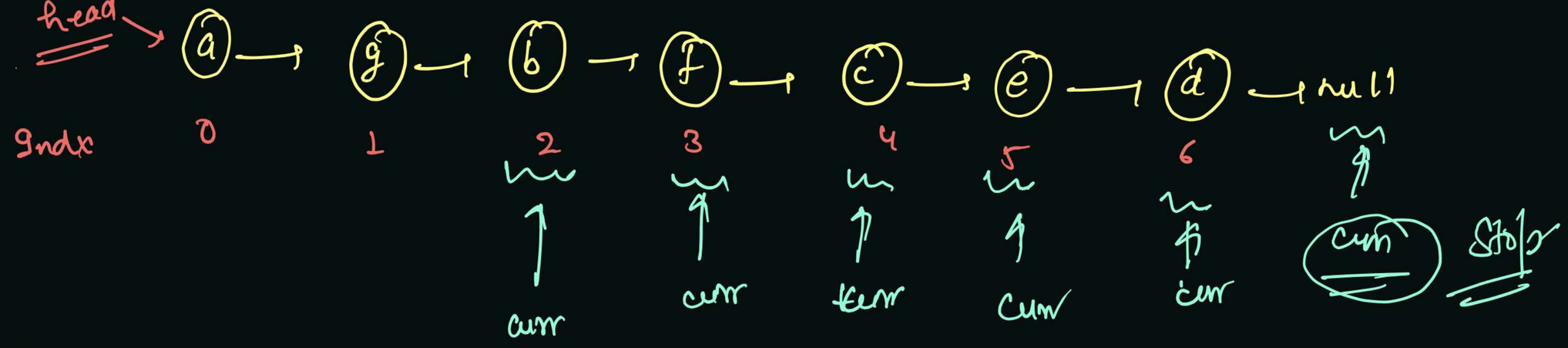


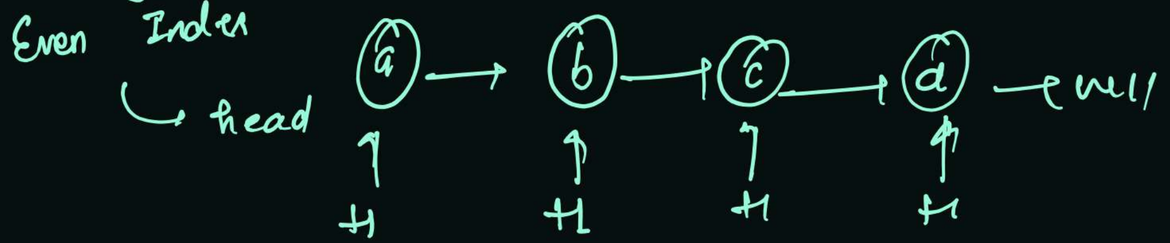

```java
public void reorderList(ListNode head) {
    // step 1. find middle node
    ListNode mid = getMid1(head);

    // step 2. Reverse second half
    ListNode head2 = mid.next;
    mid.next = null;
    head2 = reverseList(head2);
    // step 3. make a zigzag connection
    ListNode t1 = head, t2 = head2;

    while(t1 != null && t2 != null) {
        ListNode n1 = t1.next;
        ListNode n2 = t2.next;

        t1.next = t2;
        t2.next = n1;
        t1 = n1;
        t2 = n2;
    }
```

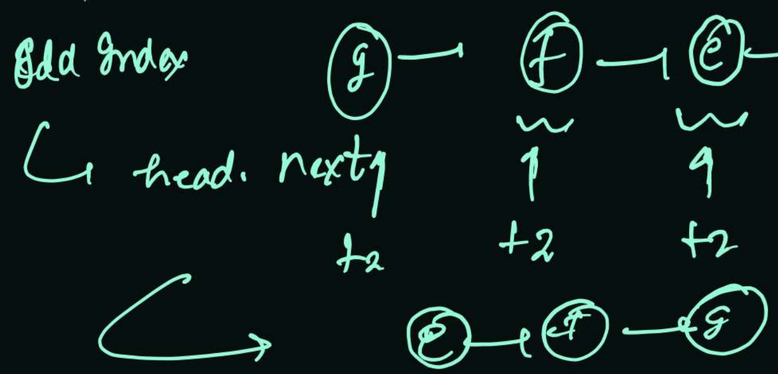
Unfold of a linked list

Saturday, 14 August 2021 2:32 PM









gndx = ~~0 1 2 3 4 5 6~~ 7

```
while( curr != null)
if( gndx%2 == 0) {
   t1.next = curr;
   t1 = curr;
} else {
   t2.next = curr;
   t2 = curr;
}
curr = curr.next
gndx++;
```

t1.next = null

t2.next = null

Reverse odd Index

head-

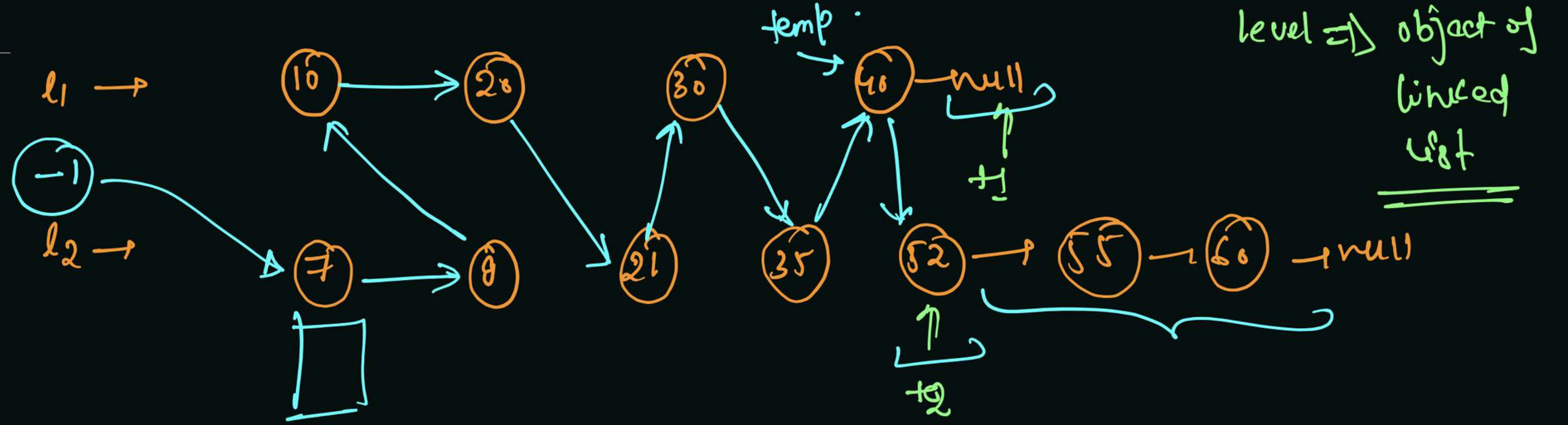
t1.next = odd.

Index

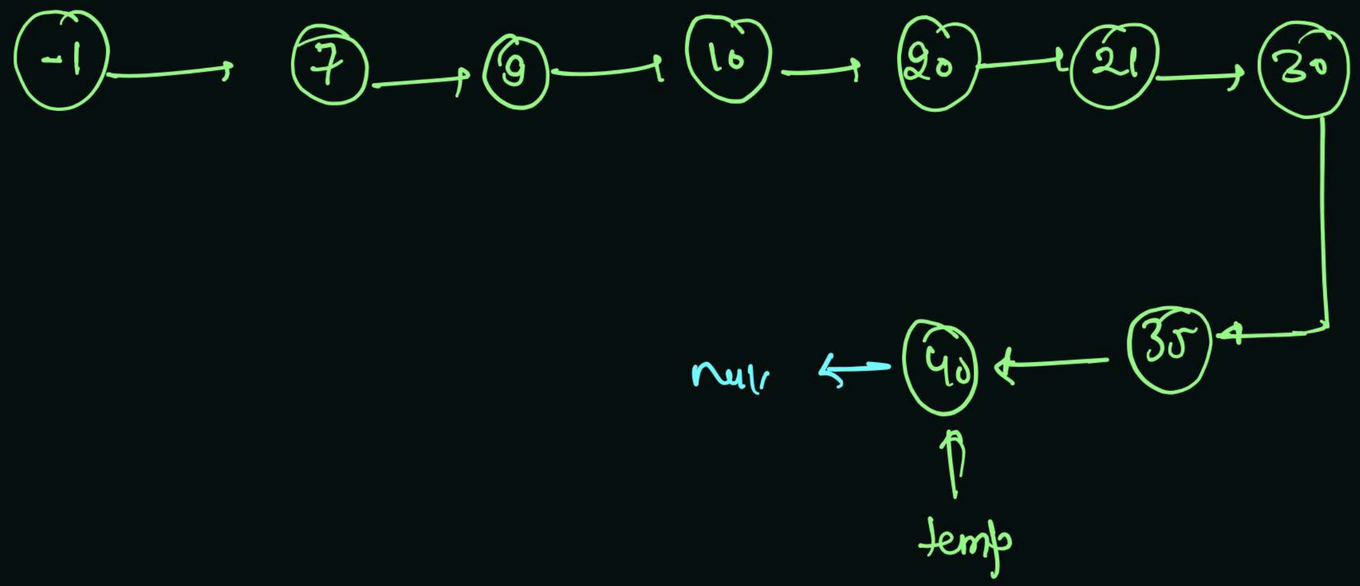
head.

Merge Two Sorted Linked List

Saturday, 14 August 2021 2:32 PM



dummy head

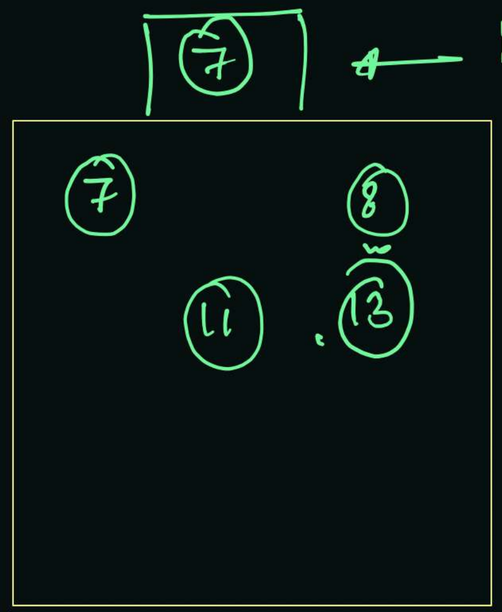



```
if(t1.val < t2.val) {
    temp next = t1;
    temp = t1;
    t1 = t1.next
} else {
    temp.next = t2;
    temp = t2;
    t2 = t2.next;
}
```

temp.next = t2.

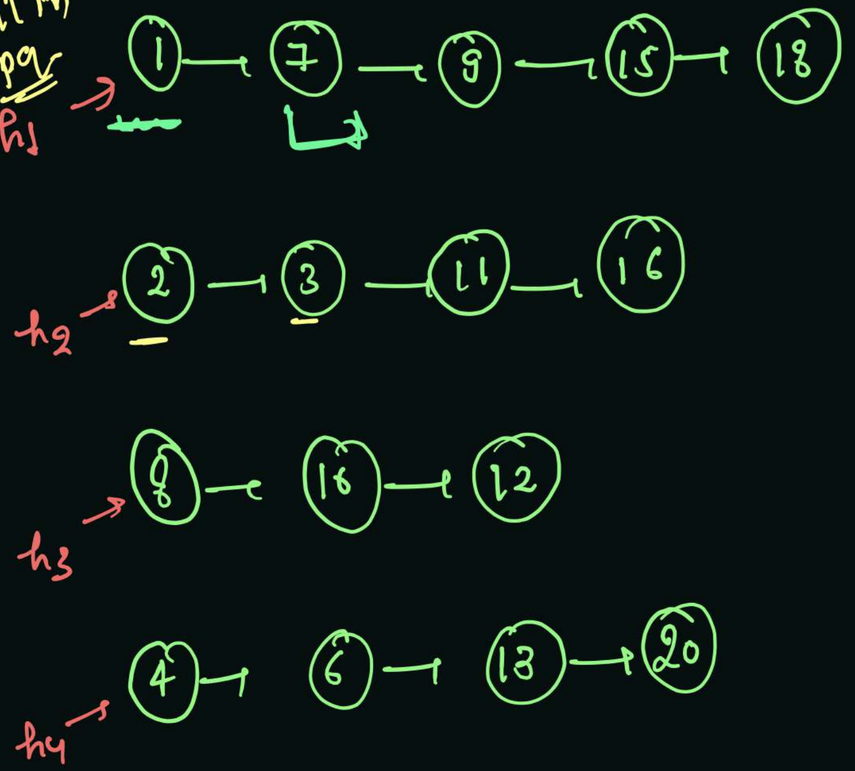
Merge K Sorted Linked List (Priority Queue)

Saturday, 14 August 2021 2:32 PM



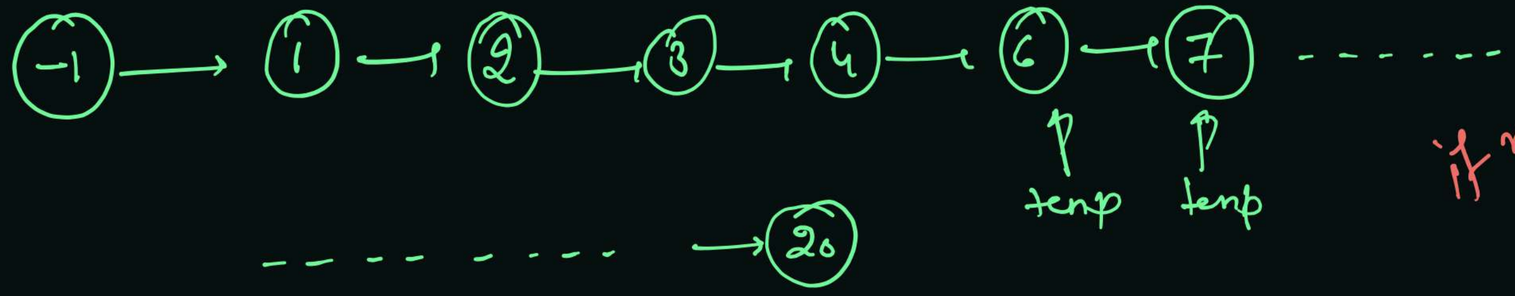
ListNode [ ]
array
of
head pointers
of
first Node.

fill in pq



time complexity = ??
Max Element in Pq = k
add and Remove in pq for single Ele = log k
Total n Elem = n log k

dummy Node



if rem.node.next != null

```
rem = pq.remove.
temp.next = rem.node.
temp = rem;
[pq.push(rem.node.next)]
```