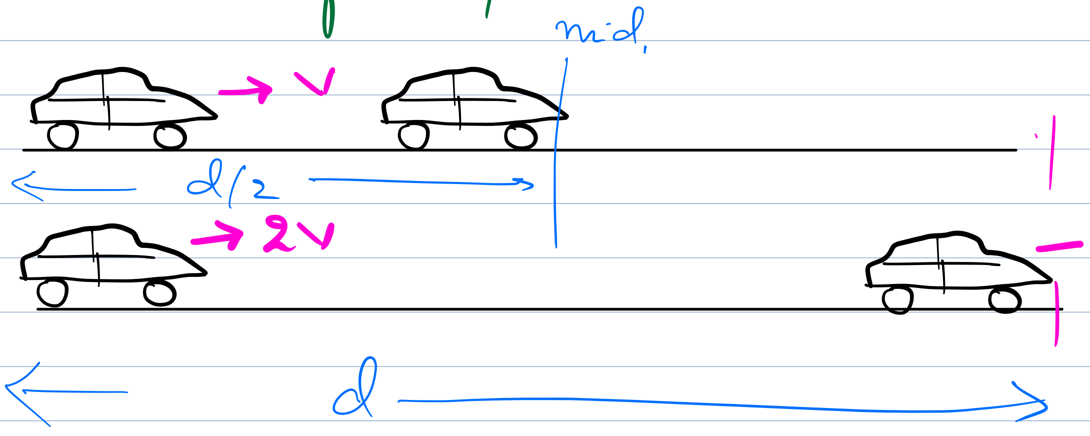
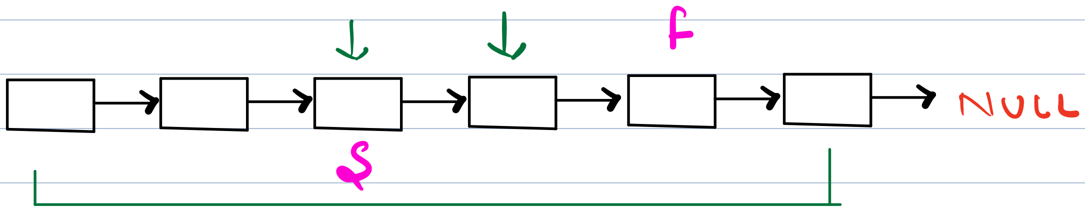
Q// Given a LL. find the middle node.

1) Count the total no. of nodes (N)

$\Rightarrow$ find the N/2 th node from the begining.

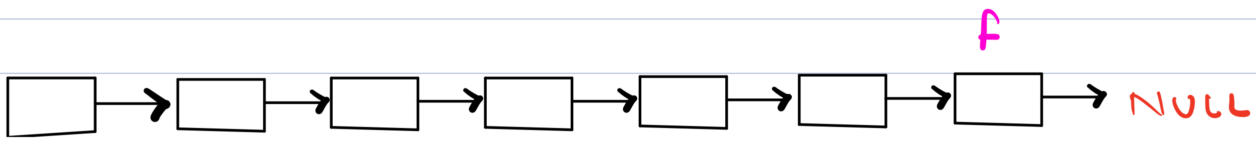
2) Slow & fast pointer.





fast = = NULL $\Rightarrow$ 2nd mid (slow)

fast. next. next = = NULL $\Rightarrow$ 1st mid (slow)



}

fast.next == NULL

Code

```
Node getMid ( head) {
        Node slow = head;
        Node fast = head;
        while ( fast.next != NULL && fast.next.next
                                         != NULL) {
            slow = slow. next;
            fast = fast. next. next;
        }
        return slow;
}
```

T.C. = O(N)

Q Given 2 sorted LL
Amazon
Merge both of them to create
a new sorted LL & return the

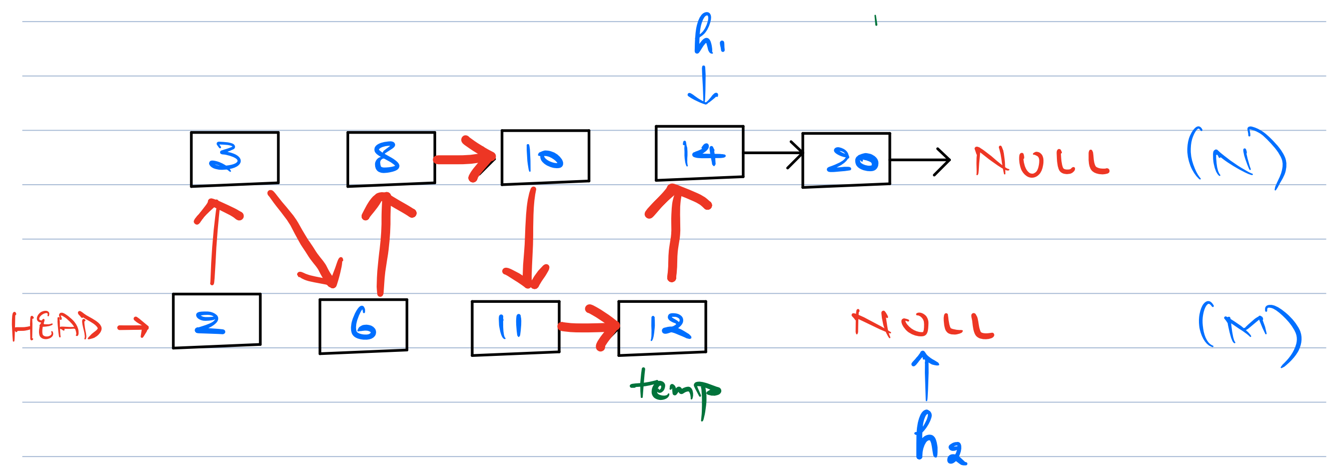
head of the new LL.

S.C. = $\underline{O(1)}$

$$\begin{tikzcd}
\text{head1} \arrow[r] & \boxed{3} \arrow[r] & \boxed{8} \arrow[r] & \boxed{10} \arrow[r] & \boxed{14} \arrow[r] & \text{NULL}
\end{tikzcd}$$

$$\begin{tikzcd}
\text{head2} \arrow[r] & \boxed{1} \arrow[r] & \boxed{2} \arrow[r] & \boxed{9} \arrow[r] & \text{NULL}
\end{tikzcd}$$

$$\begin{tikzcd}
\boxed{1} \arrow[r] & \boxed{2} \arrow[r] & \boxed{3} \arrow[r] & \boxed{8} \arrow[r] & \boxed{9} \arrow[r] & \boxed{10} \arrow[r] & \boxed{14} \arrow[r] & \text{NULL}
\end{tikzcd}$$



Code

```
Node   merge ( h1, h2) {
    if (h1.val < h2.val) {
            Head = h1;
            h1 = h1.next;
    }
    else {
            Head = h2;
            h2 = h2.next;
    }

    temp = Head;
    while ( h1 != NULL && h2 != NULL) {
          if ( h1.val < h2.val ) {
                    temp.next = h1;
                    h1 = h1.next
                    temp = temp.next;
          }
          else {
                 temp.next = h2;
                 h2 = h2.next
                 temp = temp.next;
          }
    }

    if (h1 == NULL) {
            temp.next = h2;
    }
    else {
          temp.next = h1;
    }

    return Head;
}
```

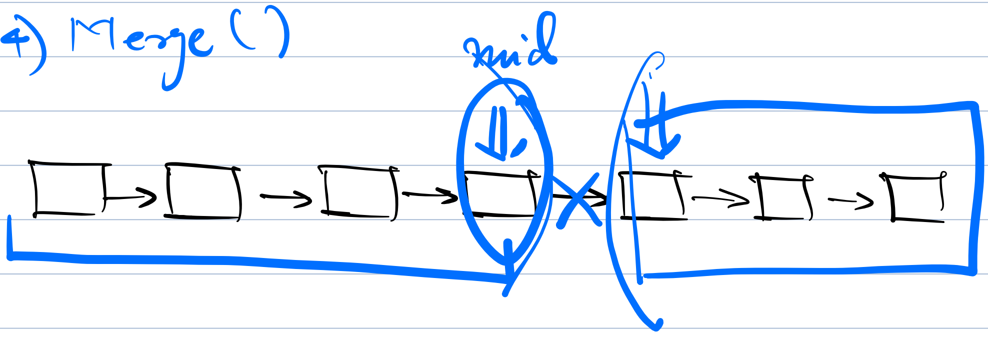
$$T.C. = O(N+M)$$
$$S.C. = \underline{\underline{O(1)}}$$

Q Google MS Amazon LinkedIn Meta

Given a linked list

Sort the linked list using merge Sort.

```
Merge Sort ( ) {
    1) Mid
    2) Merge Sort (low, mid);
    3) Merge Sort (mid+1, high);

    4) Merge ()
}
```



Code

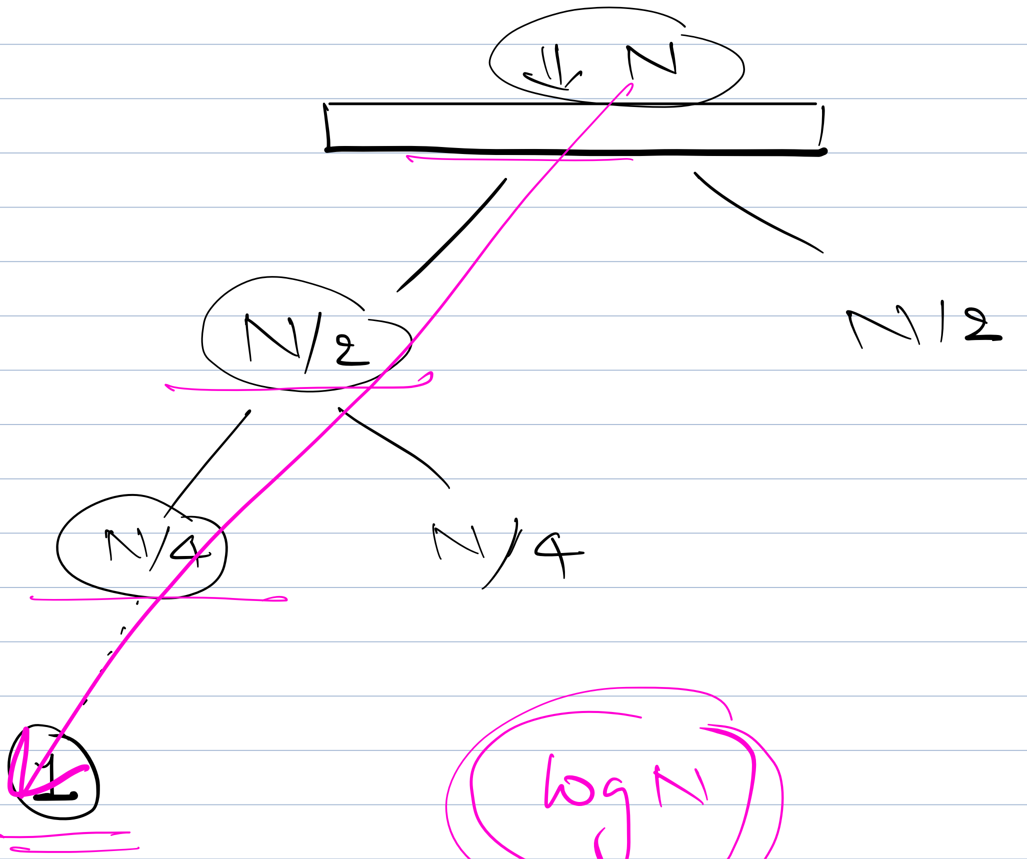
```
Node mergeSort (head) {

Base Case
    if (head==NULL || head.next==NULL) {
                    return head;
    }

    Node mid = getMid (head); O(N)

    Node h2 = mid.next;
    mid.next = NULL;

    Node h1 = MergeSort (head);
    h2 = MergeSort (h2);
```

head = Merge (h1, h2); $O(N)$
return head;
}

$T(N) = 2T(N/2) + N$

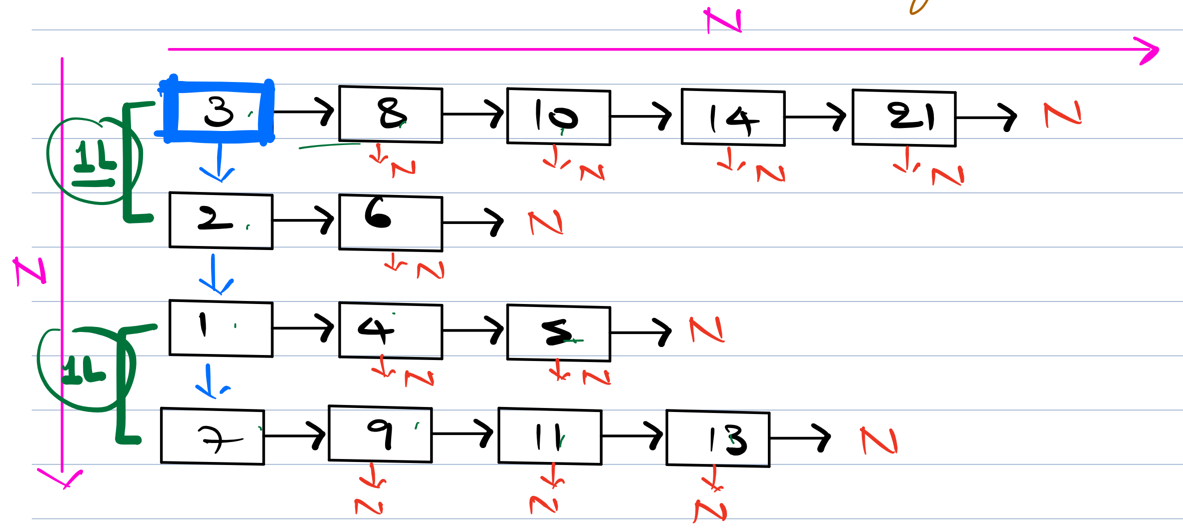
$= O(N \log N)$

S.C. = $O(1)$ ✗



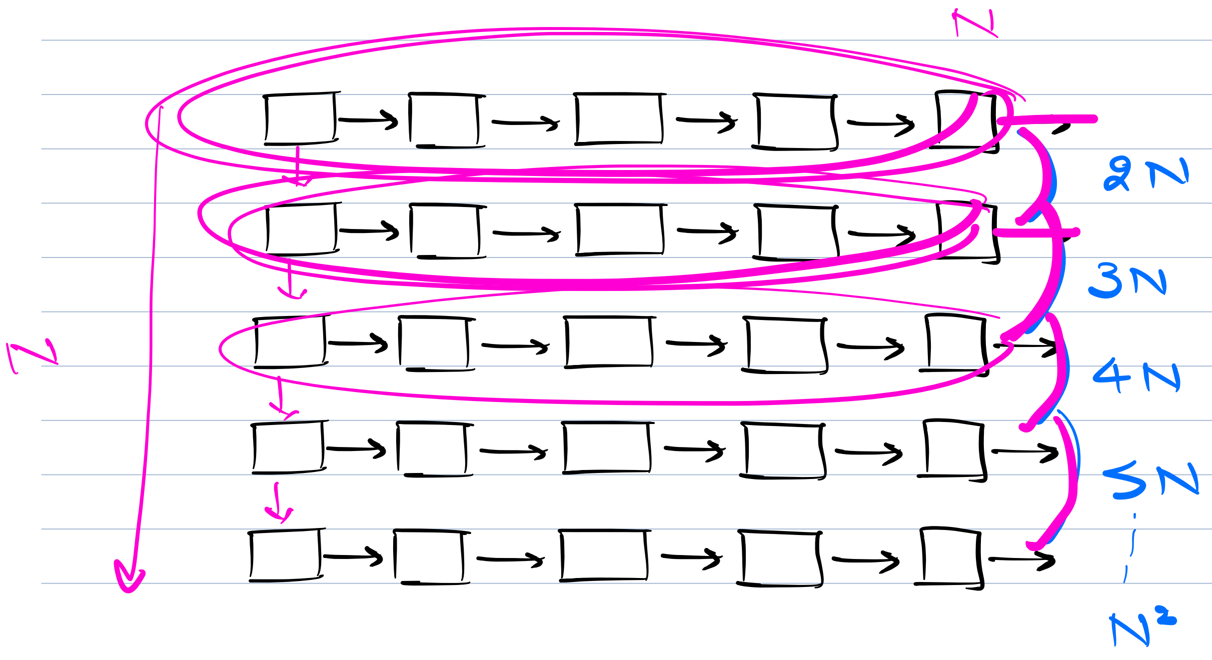
Q
Google

Given a 2D list. (sorted horizontally)
Flatten it to a single list (sorted)



1 → 2 → 3 → 4 → 5 → 6 → 7 → 8 → 9 → 10
11 → 13 → 14 → 21

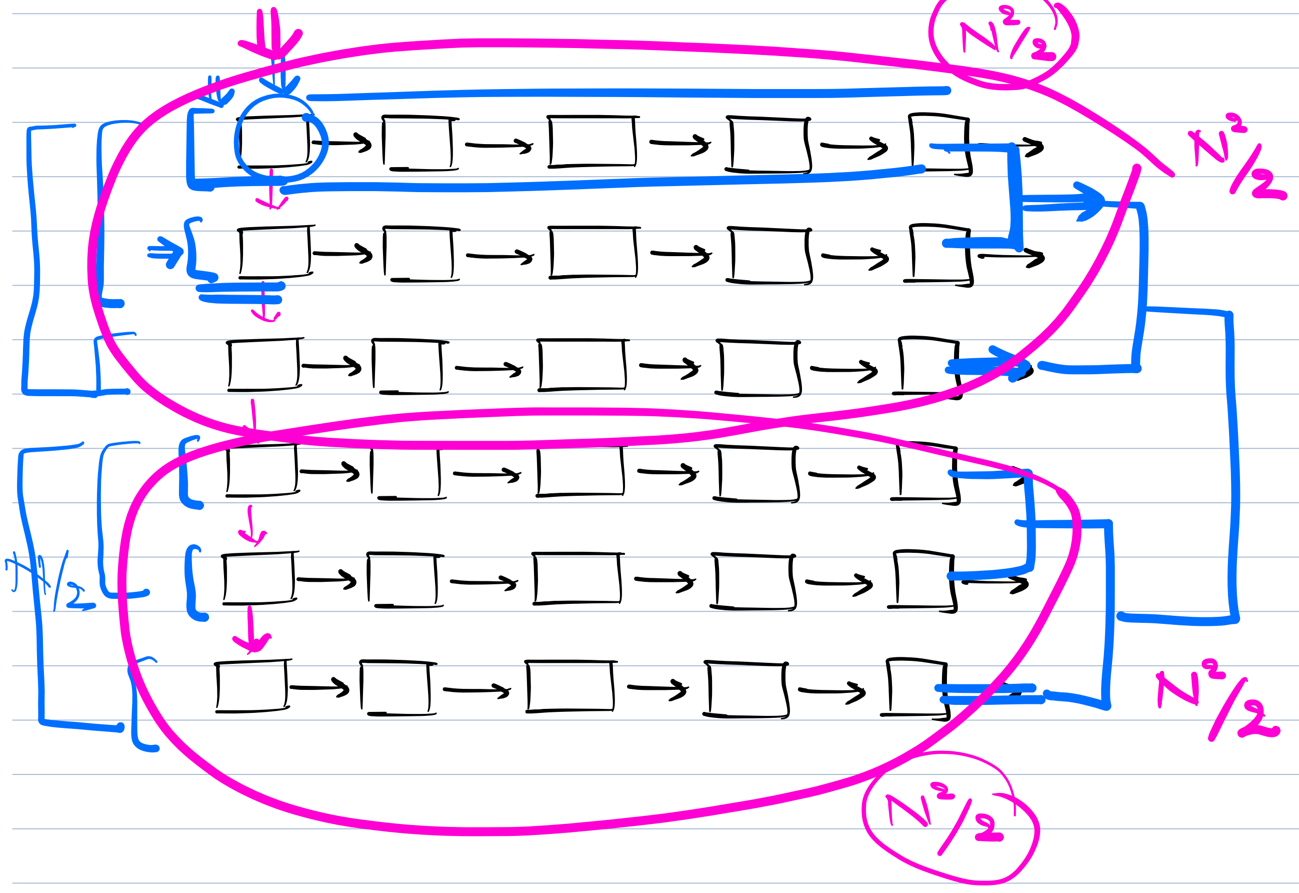
```
Node {
    int data;
    Node next;
    Node down;
}
```



$2N + 3N + 4N \ldots\ldots \quad N^2$

$N\left(\underbrace{2+3+4 \ldots\ldots N}_{O(N^2)}\right)$

$\underline{T.C. = O(N^3)}$



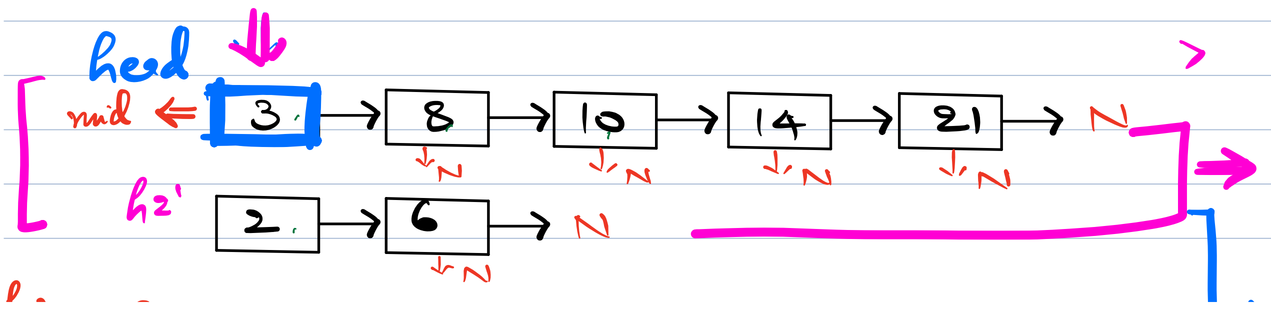
$\frac{N^2}{2} + \frac{N^2}{2} = \underline{\underline{N^2}}$

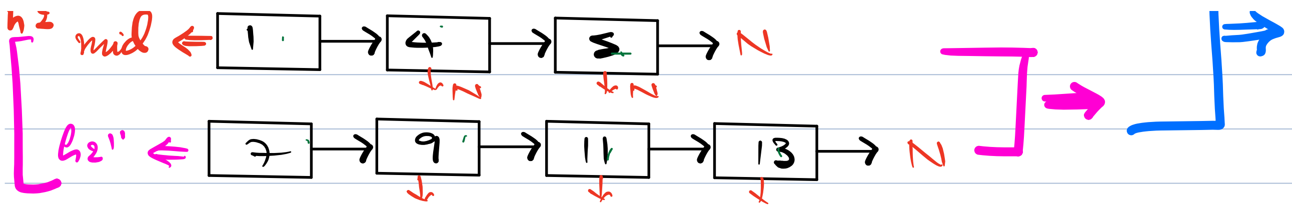
$T(N) = 2T\left(N/2\right) + O(N^2)$

$\hookrightarrow$ H.W.

Code

```
Node merge2List ( head) {
    if (head == NULL || head. down==NULL)
            return head;
    }

    Node mid = getMid (head)
                 down instead of
                      next

    h2 = mid. down
    mid. down = NULL

    head = merge2DList (head)
    h2 =   merge 2D list (h2);

    head = merge (head, h2);

    return head;

}
```




h2
mid
1
4
5
N
N
N
h2'
7
9
11
13
N