Recursion Concepts & Qns ...

Motivation (भाषण) ...

"आज नही तो कल तुम पक्का interviews फोड़ोगे।"

video 13

#codestorywithMIK

Facebook
Instagram → codestorywithMIK

(Twitter) → CSwithMIK

codestorywith MIK →

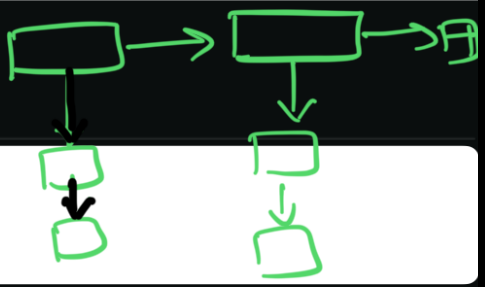
Company :-

Company Tags

Paytm | Flipkart | Amazon | Microsoft | Snapdeal | 24*7 Innovation Labs | Payu | Visa | Goldman Sachs | Qualcomm | Drishti-Soft

Flattening a Linked List

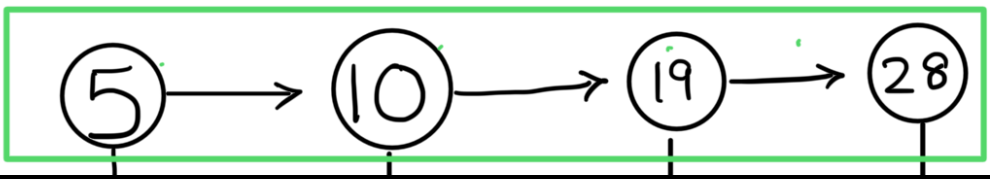
Medium Accuracy: 51.53% Submissions: 135K+ Points: 4



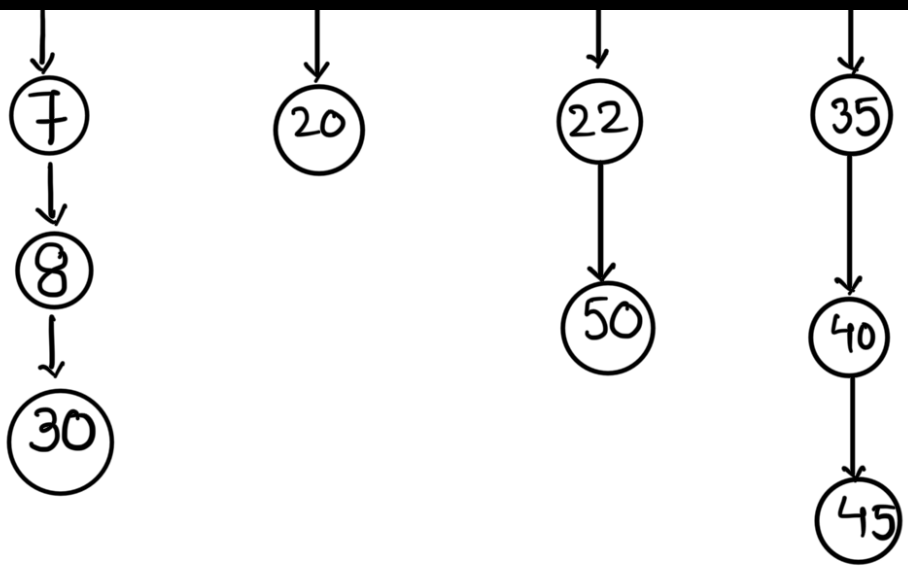
Given a Linked List of size N, where every node represents a sub-linked-list and contains two pointers:
(i) a **next** pointer to the next node.
(ii) a **bottom** pointer to a linked list where this node is head.
Each of the sub-linked-list is in sorted order.
Flatten the Link List such that all the nodes appear in a single level while maintaining the sorted order.

Note: The flattened list will be printed using the bottom pointer instead of the next pointer.
For more clarity have a look at the printList() function in the driver code.

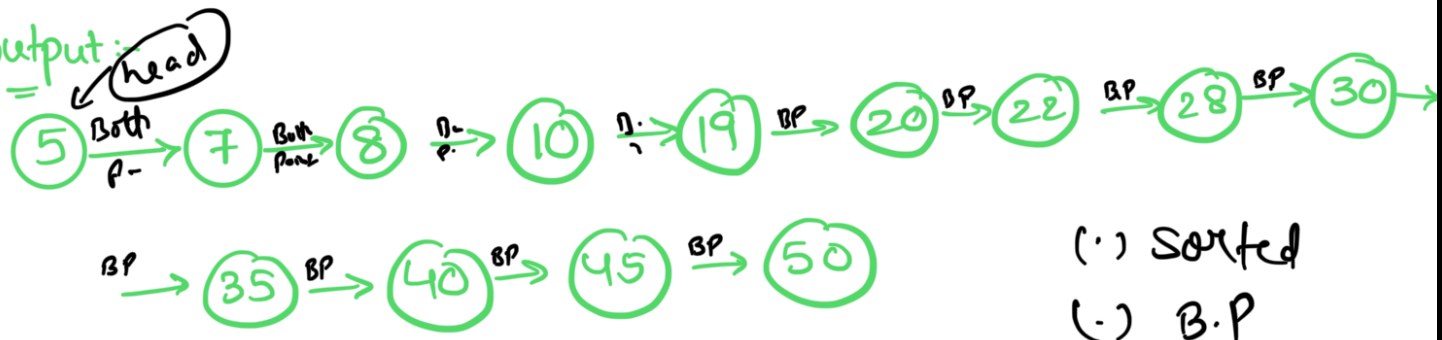
Example :-



n = 4



Output :



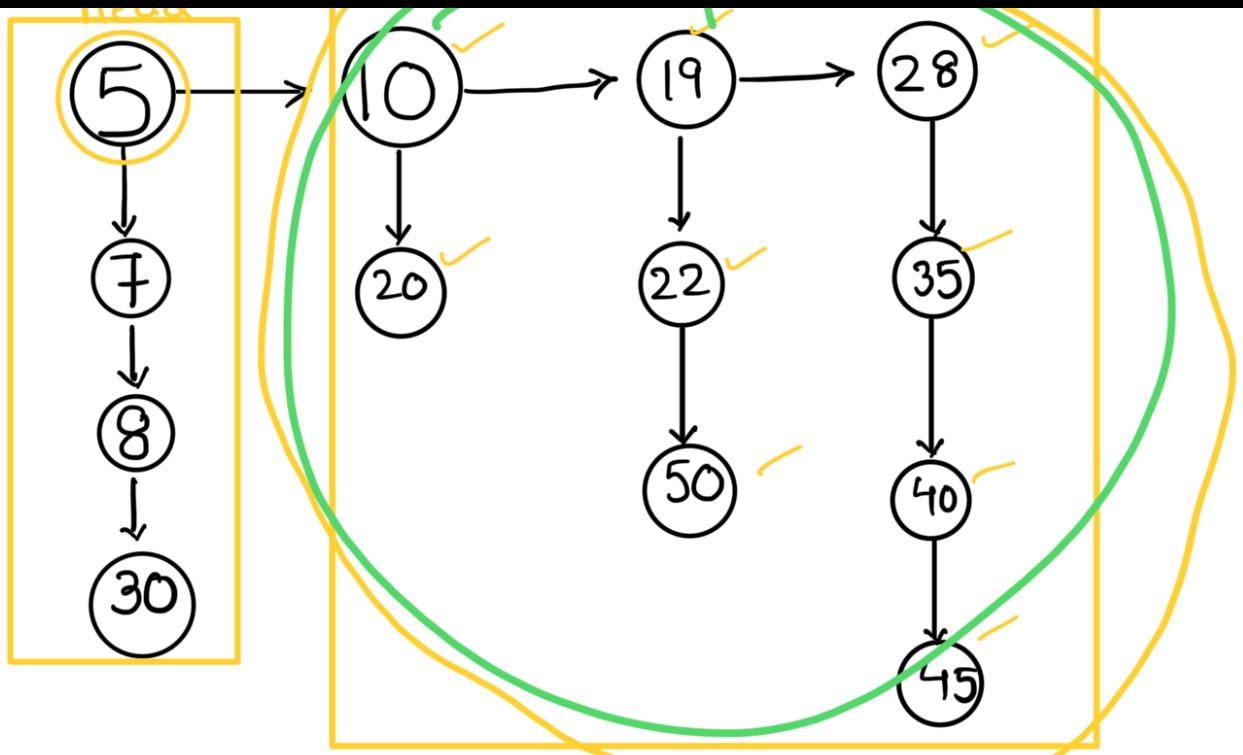
(·) Sorted

(-) B.P

*Category "Flattening"

Best for visualising "TRUST in Recursion" ...

"Beauty Of Recursion"

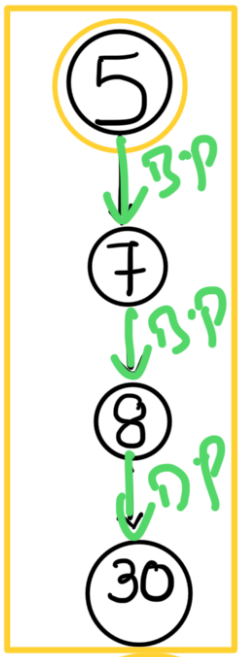


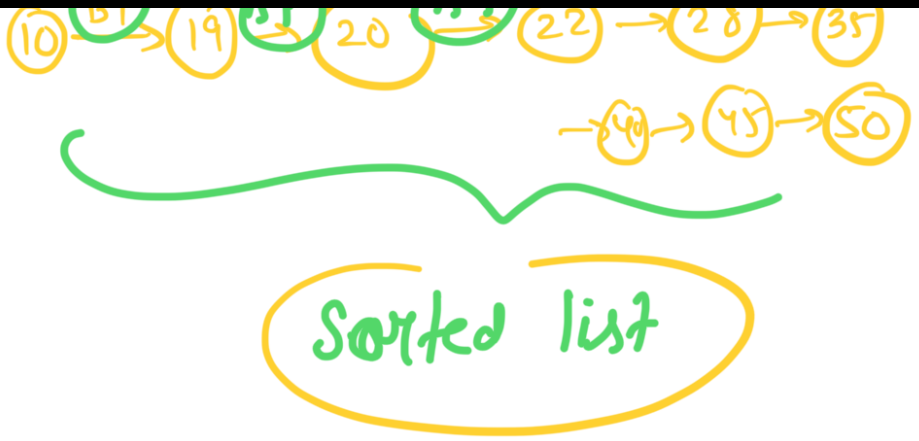
flatten(head->next)

```
Node * flatten (head) {
    if (head == NULL) {
        return NULL;
    }

    head2 = flatten (head->next);
                    //TRUST

    return mergeTwoSortedList (head, head2);

}
```

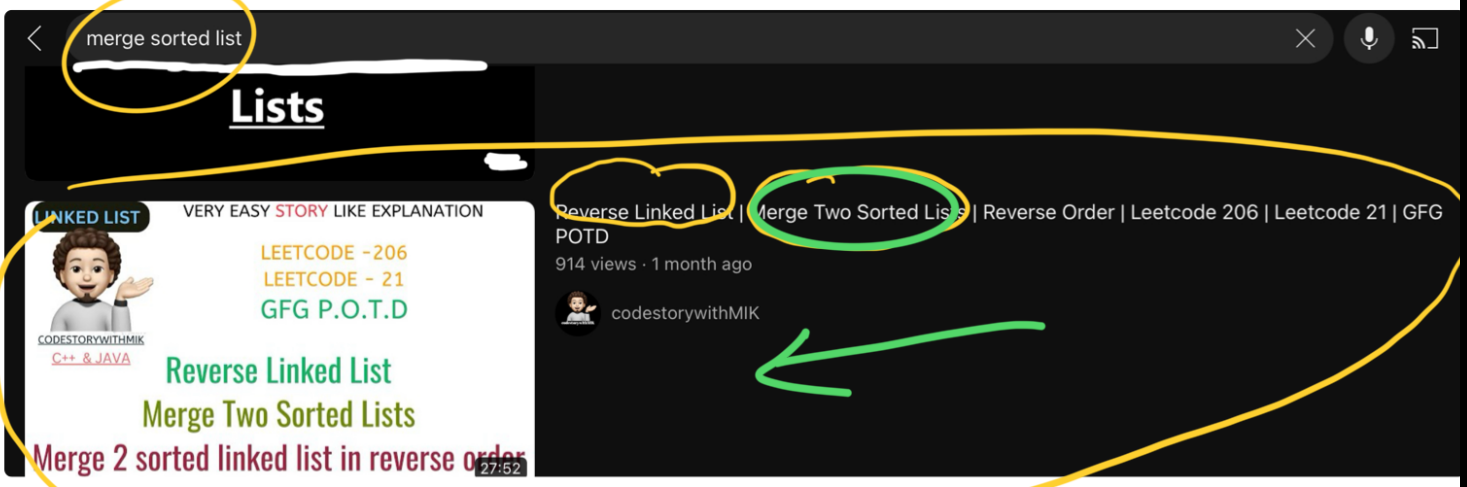
head

head2

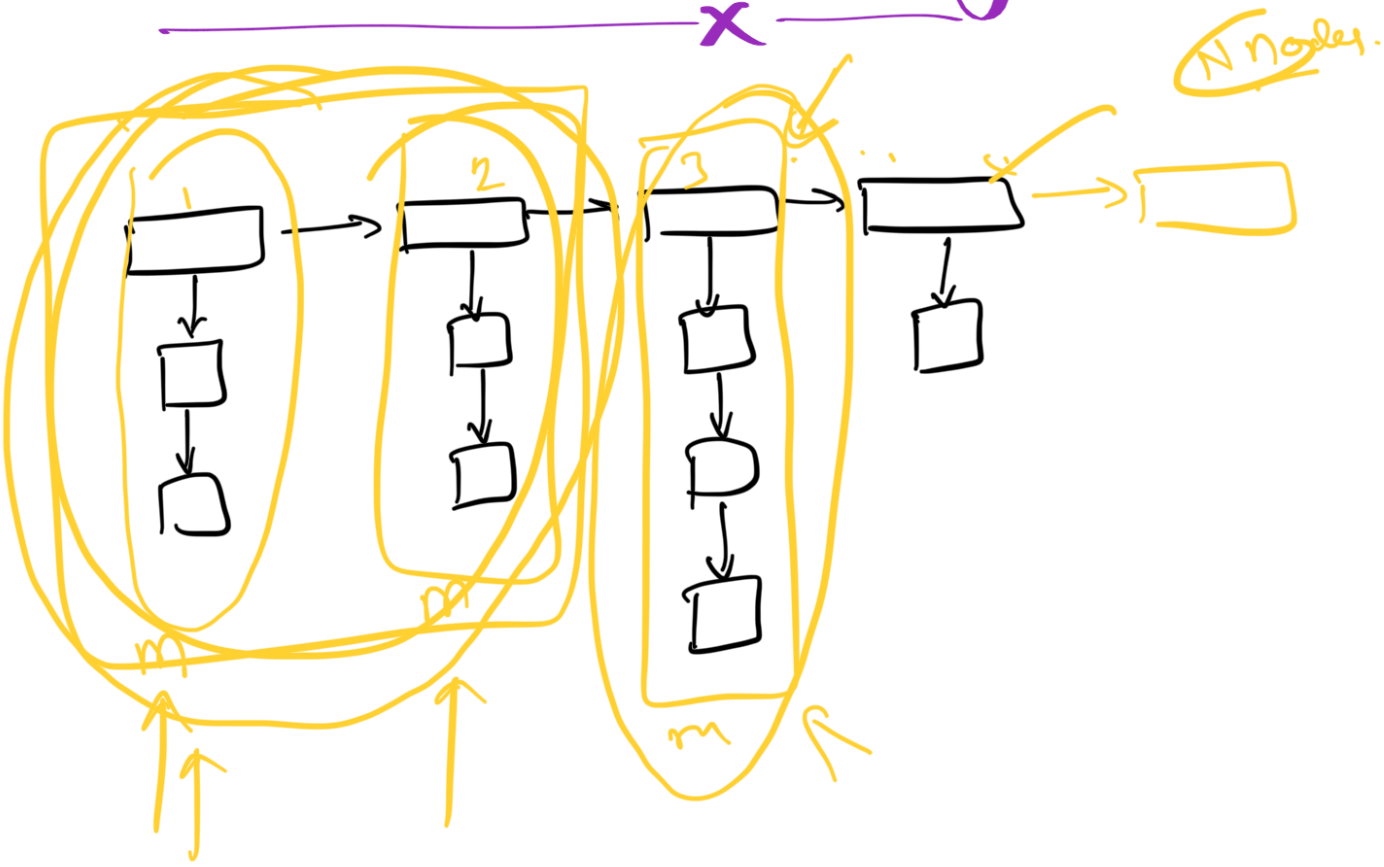

5
7
8
30
Sorted List


Sorted list

Merge


merge sorted list
Lists
LINKED LIST
VERY EASY STORY LIKE EXPLANATION
CODESTORYWITHMIK
C++ & JAVA
LEETCODE -206
LEETCODE - 21
GFG P.O.T.D
Reverse Linked List
Merge Two Sorted Lists
Merge 2 sorted linked list in reverse order
27:52
Reverse Linked List | Merge Two Sorted Lists | Reverse Order | Leetcode 206 | Leetcode 21 | GFG POTD
914 views · 1 month ago
codestorywithMIK

Time Complexity :-



$O(m + m) = O(2m)$

$O(2m + m) = O(3m)$

$O(2m + 3m + 4m + \dots + Nm)$

$O(m\ (2 + 3 + 4 + \dots\ N))$

$O(m * (N * (N+1)))$

$\frac{1}{2}\ O(m * (N*N + N))$

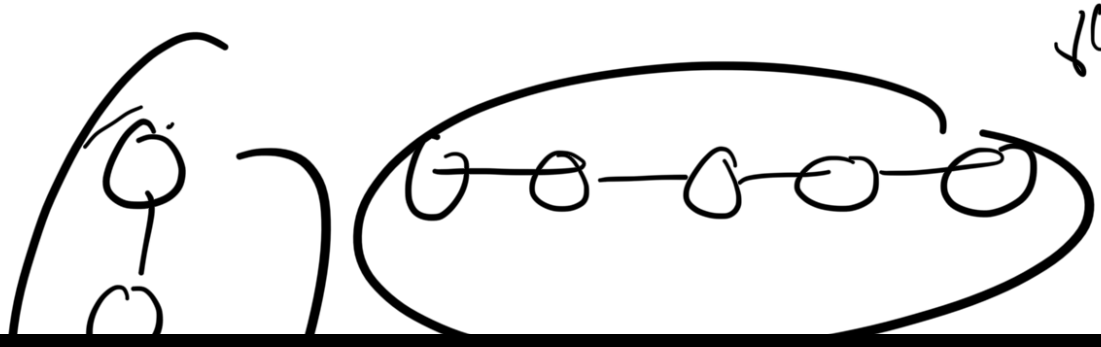
$\not\frac{1}{2}\ O(m*N*N + m*N)$

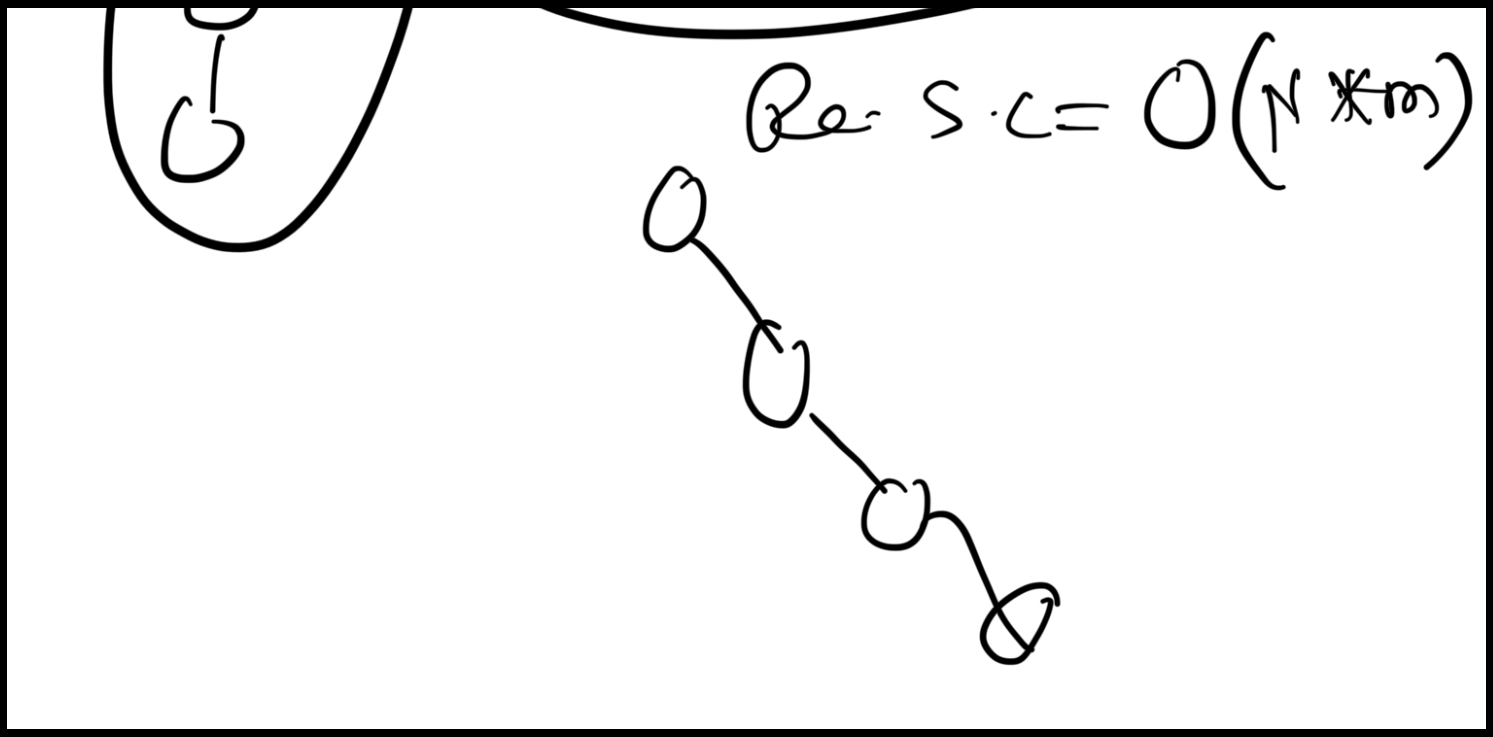
T.C. $\simeq O(m*N*N)$.

Space Complexity :-

Auxiliary Space = $O(1)$

N * m = total no of nodes in flatten linked.




Re- S.C= O(N*m)