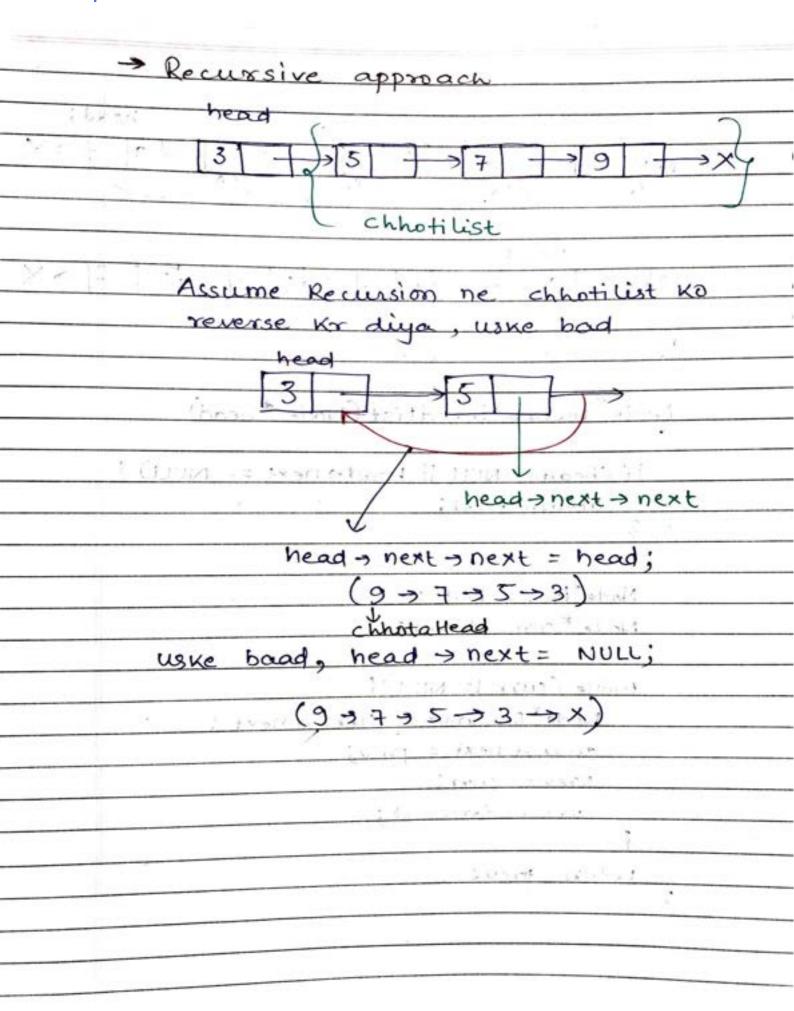
# Linked List

-By Nikhil Kumar

# LINKED LIST

	head;
6 7	NUU 3 7 7 7 9 7 9 7 9 7 9 7 9 7 9 7 9 7 9 7
115	head %= 9 + 7 + 5 + 3 +>
	Program!-
	Node* reverselinked list (Node * head)
	if (head == NULL    head => next == NULL) { return head;
	inead a next and the life id:
	Node * curr = head;
	while (curr 1= NULL)!
	Node forward = curr > next;  curr > next = prev;
	prev = curr;
	curr = forward;
	return prev;

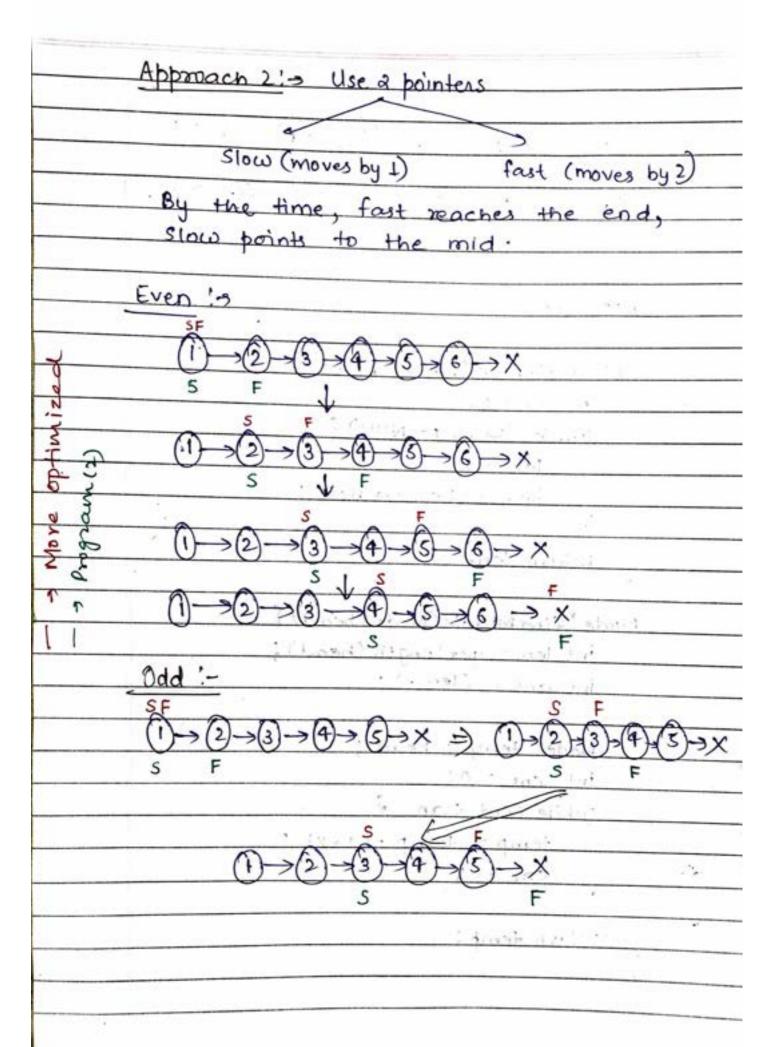


	Program!-
4	
	Node* reverse (Node* head) {
	KI 64 AV
	if (head == NULL   head > next == NULL) {
	return head;
	}
	Node * chhotaHead = reverse (head = next);
	head - next - next = head;
	head + next = NULL;
	return chhotaHead;
	3
	Node* reverse linked list (Node* head) {
	return reverse (head); show show
	inosa a rene "shoul
	Julia sana * shall
	1 (vard trees to sai) as vavay
180259	Librard accutat
	1

	. *
Void reverse (Node shead, Node curr, N	lode Preuly
1/base case	
( curr == NULL) &	
head = prev;	
returni	(
3	(
A forest contract of the contr	(
Node * forward = curr > next;	(
reverse (head, forward, curr);	
Curr > next = prev;	•
3	
Lesi bear Thank Thank I have	
Node * reverse linked list (Node * head) {	
Node* curr = head;	
Node prev = NULL;	
reverse (head, curr, prev);	
return head;	•
	•
	4
	•
	0
	9
	0
	6
	0
	0

https://www.linkedin.com/in/nikhilkumar0609	9/

Ø	Middle of a linked list
	1/p: -> (1) -> (2) -3) -4) -> NULL
	1/p: > 1 2 3 > 4 3 > NULL
	Approach 1:-
	int getlength (Node * head) { int len = 0;
	While (head 1= NULL) {  len++;
	head = head = next;
*	return len;
	Node * find Middle (Node * head) {  int len = getlength (head);
	int ans = (len/2);
(2 to 1)	Node * temp = head;
	int cnt = 0; while (cnt (ans) f
	temp = temp > next;
	3
	? return temp;



```
Program : 5 (2)
Node * get Middle (Node * head) &
  if (head == NULL | head > next == NULL
      return head;
  if (head = next = next == NULL) f
      return head > next;
  Node * slow = head;
  Node * fast = head = next;
 while (fast != NULL) {
   fast = fast = next;
   if (fast 1 = NULL) {
    fast = fast > next;
  Slow = slow or next;
  return slow;
Node * find Middle (Node * head) &
 return get Middle (head);
```

Important

	More Optimized Method:
	Node * find Middle (Node * head) {
	if (head == NULL)
	return head;
	Node slow = head;
	Node * fast = head; > next;
	while (fast 1= NULL && fast > next 1= NULL)
1200 500 500	fast = fast = next = next;
-	Slow = slow + next;
	<b>4</b>
	return slow;
	J. or stipping and an experience
	1.10.400.0.500
	a see a see
	yp: → [1,2,3,4] => 0/p = [3,4]
<del>United S</del>	YP: > [1,2,3,4] => 6/P = [2,3,4]
	5 Changer of all rest to the state of a section
	(Charact ) altering to a party

Reverse list in K-Groups
1/p: 3 + 7 + 8 + >11 + >17 + 2 + >x
7 + 3 + 11 + 8 + 2 + 17 + x
%: > K= 3 8 +>7 +3 +2 +17 +11 +>X
Approach 1:2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
Algorithm ! >
I case solve Krnge  Iterative algo (court < K)
First k Node reverse
head > next = recursion call
return head of reversed Linked list
return Prev;
Cr Josephi Stanie Lead
in the second se

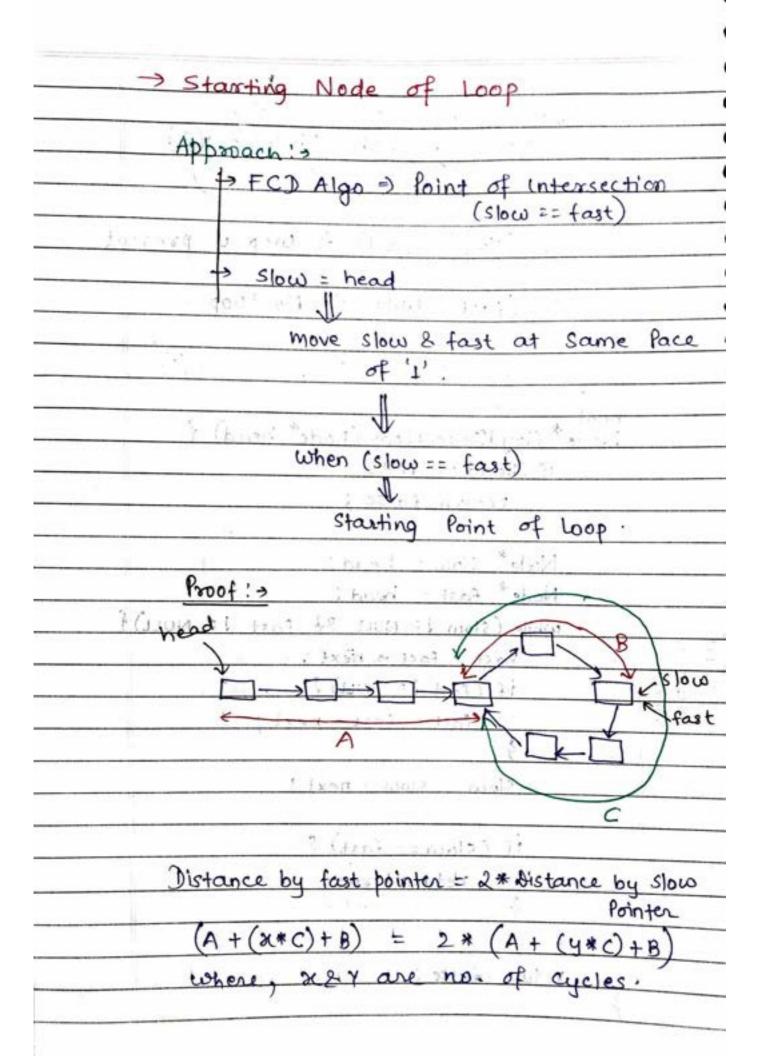
Pa	ngram:-
N	ode* KReverse (Node* head, int K) {
	// base case
	if (head == NULL) {
10 TO 1	return NULL;
	3
- 4-1-11-11-1-1-1-1-1-1-1-1-1-1-1-1-1-1-	// Step 1: reverse first k nodes
	Node next = NULL;
	Node * curr = head;
	Node * prev = NULL;
	int count = 0;
	Separa erlat sted t
(	while (curr 1= NULL && count < K) &
	next = curro next;
	Curr + next = Prev;
	prev = curr;
	Upo rizon = next ; rar = leas (
	Count ++;
JEL 6:	enter the transfer of the set of
	// step 2: > recursion dekh lega aage ka
	if (next != NULL) &
	head - next = KREVErse (next, W;
	}
	// step 3:0 return head of reversed list
	return prev;
4	

-	(Assume empty ist as Circular)
	Approach 1:3
	Agair M jis node se start kiya, uss node pe dubara aa jaan toh circular linked hit ho
	Program :
	bool is Circularlist (Node * head) {
4 10	Mempty list
	if (head == NULL) { > 1 1 1 1 1 1
	return true;
	Towns and the second of the se
1.00	Node temp = head - next;
$i$ ) $\beta$	while (temp != NULL & temp != head) {
	temp = temp > next;
	3
	i historia Stand, Stand anor
	if (temp == heald) Strate and
	return thise; gover ) winter
	foskski i skutik
	return failse;   books   1
	3 : Such corurss
	i and if from the tiple
	Laborate district district
	1.5311

Detect and Remove Loop
→ Detect wop
Algorithm 1:-  True True True True True True True  (1) $(2)$ $(3)$ $(4)$ $(5)$ $(6)$
3 4 5 6
Yahan Phle se hi true mark kiya hua ho
mtlb loop exist Krta hai
Program:-  bool detect loop (Node * nead) { Sic = 0(n)  if (head == NUL) Tic = 0(n)
return false;
Map < Node*, bool > visited;  Node * temp = head;
while (temp 1=NUL) &  // Cycle is present
of (visited [temp] == toue) &  return true;
visited [temp] = true; temp = temp > next;
return false;
2

```
true/false and red pen is used for the code to return
     the node from where the loop is starting
        Floyd's Cycle detection Algorithm -
                               wop is present
              (slow == fast)
              (fast = NULL) =) No loop .
                   (T.c > O(n), s.c > O(1)
       Program !-
       Node * floyd Detect Loop (Node* head) &
            if (head == NULL)
               return false;
                      NULL;
           Node * Slow = head;
                fast = head;
           while (Slow != NULL && fast != NULL) {
              fast = fast > next;
               if (fast != NULL) }
                  fast = fast = next;
              Slow = Slow > next;
              if (Slow == fast) &
         3 Slow;
   of pro9
                          (A+(300)+B)
       return false jan vale
                  NULL;
```

Note: In this page, blue pen is used to return



# https://www.linkedin.com/in/nikhilkumar0609/ A + B = K times C. Mtlb, A+B ka mtlb hai ki maine Cycle Complete Krdi hai fast. Toh agar main 'B' distance pe hu toh iss cycle to complete Krne ke live 'A' distance Igega. Toh fast pointer aux head se '1' steps aage bothne pe It (slow = fast) ho jaaye ton whi starting por node hogi loop ki.

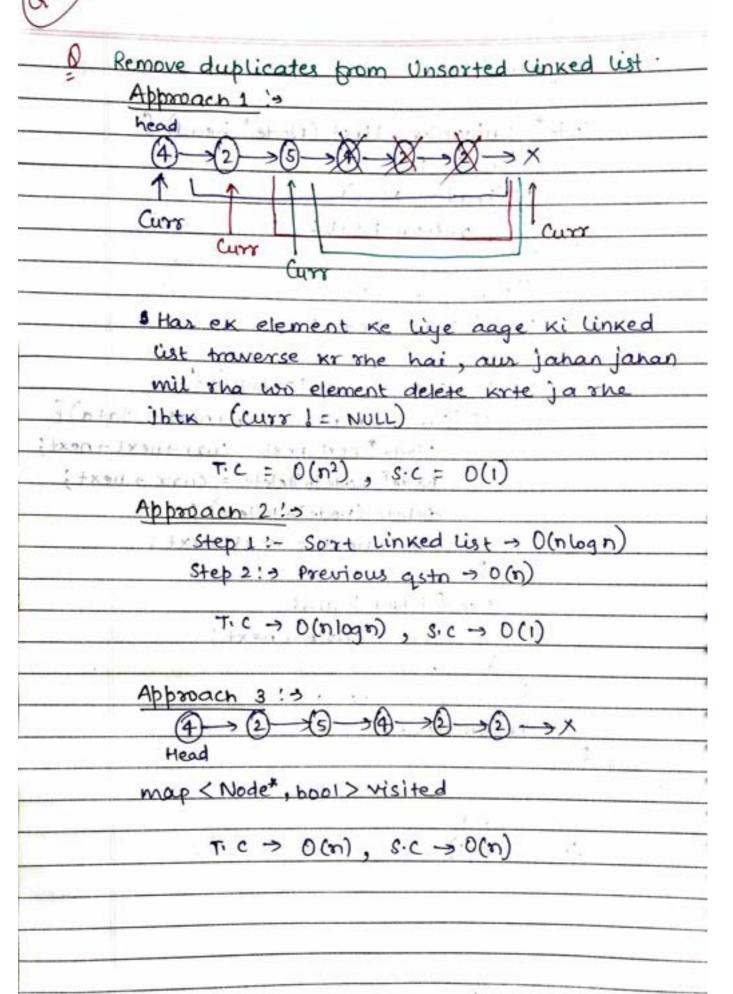
Program:  Node* get Starting Node (Node* head) if  if (head == NULL)  return NULL;  Node* intersection = floyd Detect Loop (head);  Node* slow = head;  while (slow 1 = intersection) i  slow = slow > next;  intersection = intersection > next;  return slow;  Remove (yele from LL  NULL  Agas Starting Node ke ex phile wala  node ke pointer ko NULL pointer bna du toh Loop remove ho jayega:	if (intersection == NULL) } Jb loop nhi ho toh return NULL; starting node kahan se hoga.
Node* intersection = floyd Detect Loop (head);  Node* slow = head;  While (slow != intersection) i  Slow = slow > next;  intersection = intersection > next;  intersection = intersection > next;  intersection;  Remove Cycle from !!  Approach :> starting Node  Noull  Agas Starting Node Ke ex phle wala  node Ke pointer Ko NULL pointer bna	Program !-!
Node" slow = head;  While (slow != intersection) i  Slow = slow > next;  intersection = intersection > next;  ?  Remove Cycle from L.L  Approach !> starting Node  head  NULL  Agas Starting Node Ke ex phle wala  node ke pointer Ko NULL pointer bya	Node intersection = floyd Detect Loop (head);
While (slow) = intersection) i  Slow = slow > next;  intersection = intersection > next;  return slow;  Remove Cycle from LL  Approach; > charting Node  NULL  Agas Starting Node Ke ex phie wala  node Ke pointer Ko NULL pointer bya	Node" slow = head;
return slow;  Remove Cycle from L.L  Starting Node  NULL  Agas Starting Node Ke ek phile wala  node ke pointer ko Null pointer bya	While (slow 1 = intersection) }
Remove Cycle from LL  Approach:  head  NULL  Agas Starting Node Ke ex phle wala  node Ke pointer Ko NULL pointer bya	intersection = intersection -> next;
Approach :> starting Node  NULL  Agas Starting Node Ke ek phie wala  node ke pointer ko NULL pointer bya	3 to 3 teturn slow;
node ke pointer ko NULL pointer bna	head head
	node ke pointer ko NULL pointer bna

```
If (start Ot loop == NULL)
       return NULL;
 Program 12
Void remove Loop (Node* head) {
     if (head == NULL)
        return NULL;
    Node * start Of Loop = get Starting Node (head);
    Node * temp = start Of Loop;
    while (temp > next != Start & Loop) &
         temp = temp = next;
    temp - next = NULL;
    return head;
```

8	Remove Duplicates from Sorted Linked list
	yp: - (1-2) -2 -3 -3 -3 -3 -×
	%p: → (1) → (2) → (3) → (4) → ×
:il.es	Program :-
	Approach is
	(1)—(2)—(3)—>×
	Now, 11= 2 = Curr = Curr = next
	(1) > (2) × > (2) -> (3) -> x
	Now, 2 == 2 next_next =) Curr = next = Curr = next = next
	And delete curronext  curronext = next_next
•	Kyuki agar delete ke phle Curronext Ko change krnge toh jo Node delete
	payenge.

	Program!-
	Node * uniqueSoxted list (Node * head) [
	//empty list
	if (head == NULL)
	return NULL;
	// non-empty list
	Node* curr = head;
140	AT THE REST OF THE PARTY OF THE
	while (curr > next != NULL) &.
	if (curr > data == curr > next > data) {
	Node * next next - Curry angul a next
	Node* node To Delete = curr > next;
	delate (note to belete = curr + next;
	delete (node To Delete);  curronext = next_next;
	2 carranext = next_next;
	Step 2: a Paculous and a section)
	else { // Not equal
	curr = curr > next;
	3
	8 457 5434
	return head;
_	3
	man of Norde . the 15 misted
	1, C = 0(n)
	S.C = (001) . D.S . (000 0 0 5 0

a la la la la la la la



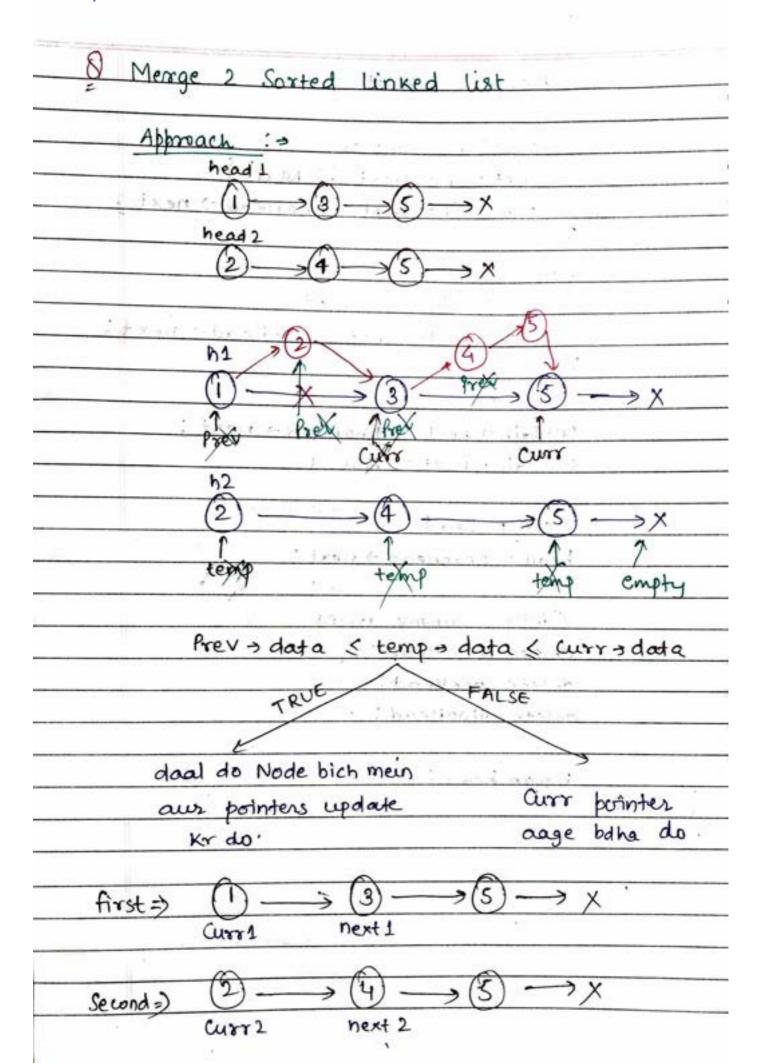
	linked list of
Q	Snrt 1 Os, 1s, 2s.
	Approacht:
	$(1) \longrightarrow (0) \longrightarrow (1) \longrightarrow (2) \longrightarrow \times$
	head
	Algo ! -> Count 0, 1, 2
	(a)
	No. Count
	0 -> 1
	. → 2
	2 -> 2 .
	ØØØ×
	$0 \longrightarrow 1 \longrightarrow 0 \longrightarrow 2 \longrightarrow \times$
	Shord same
	$T(c \rightarrow 0(\tilde{n}), S(c \rightarrow 0(1))$
	# Oh to Committee to the committee of the
	# Phle Saare no. Ko Count Kr live fir data ko replace Kr de rhe.
	aud so replace he at the.
	West (Some County 1 27)
	· il thebegan
	\$ 2 to English States
	Post trade in the sale
	Connect to the
	V Constantings
	Street good good &
	Logic grade

```
Program:-
Node * Sortlist (Node * head) &
   int zeroCount = 0;
   int one Count = 0;
   int two Count = 0;
   Node + temp = head;
  while (temp ! = NULL) {
    if (temp -> data == 0)
       zeroCount ++;
      else if (temp = data ==1)
         one Count ++ }
      else if (temp + data == 2)
        two Count ++;
      temp = temp = next;
   temp = head;
   while (temp != NULL) &
       if (zero Count != 0) {
     temp > data = 0;
     zero Count -- ;
      elseif (oneCount != 0) {
      temp > data = 1;
        one Count -- ; }
      else if (two Count 1=0}
          temp > data =2;
          twoCount -- ; }
      temp = temp > next;
   return head;
3
```

$(1)$ $\rightarrow (2)$	$\rightarrow$ (i) $\rightarrow$ (2) $\rightarrow$ X
Head 1	1 1 curr
Currs	<u> </u>
	Ruk Jac
zero -> (6)	Water States of the Company of the C
One -	0
Two -> (2)>	(2)
117 may 1	Charles Williams
3 alag-alag linked	list by gya, ab Tec
Ko menget Kr dena no	in the Thomas
7	Unit aget * shout
(b)-(1)-(1)-x	2)-(2)-> X
1 : he set	and about
Head	31. 14.675
-> Create dummy no	des des
ZeroHead > 0> 0	2 Zero lace
iller terrolling	
one Head so - 1).	<b>→</b> ( ' )
	- two Tail
two Head 10 -> 2	+wo Tail
Merge:	1
27.17, 22.30	NULL
Literate Lie Trans Dillionie A	
T. ( → O(n), S. ( →	0(1)
· handa	commence of the second

```
Program :-
       void insert At Tail (Node* Stail, Node* curr) &
         tail = next = curr;
       Node * sortlist (Node * head) }
          Node * zeroHead = new Node (-1);
           Node * zero Tail = zeroHead;
           Node * one Head = new Node (-1);
     Node one Tail = one Head;
          Node + two Head = new Node (-1);
           Node* two Tail = two Head;
          Node * curr = head;
          11 create separate list of Os, 1s, 2s
          while (curr != NULL) &
             int value = curro data:
              if (value == 0) {
                 insert AtTail (zeroTail, curr);
        else if (value == 1) &
                 insert At Tail (one Tail, curr);
              else if (value == 2) }
                insert At Tail (two Tail, curr);
             Curr = Curronext;
```

	//merrge 3 sublist		
	1/1s list not empty		
	if (one Head → next 1= NULL) {		
	zeroTail = next = one Head = next;		
	3		
	else &		
	1/1s list is empty		
	zeroTail - next = twoHead - next;		
_	<b>4</b>		
X	the first of the second of the		
	One Tail > next = trootlead > next;		
	two Tail > next = NULL;		
	54		
X	1/setup head		
	head = zeroHead > next;		
- 1	A LAND DOOR		
	//delete dummy nodes:		
7.1	delete zemtead;		
	delete oneHead;		
-	delete two Head;		
	· · · · · · · · · · · · · · · · · · ·		
	steturn head; man do so are ab lest		
	and proper white the		
8	sales again to all		
-	X 5 - 17/6 - 10 1 2 - 1 1 6 1 1		
-	t tear trees		
-			



1.41	Program :
-	Node <int>* solve (Node <int)* <int)*="" first,="" node="" second)="" td="" {<=""></int)*></int>
	//if only one element is present in first list
	if (first > next == NULL) &
	first = next = Second;
	return first;
	·
	Node (int)* curr 1 = first;
	Node <int>* next 1 = curr1 → next;</int>
	Node <int> curr = second;</int>
	Node (int)* next 2 = curr2 > next;
Course	Node Charles and Andrewall States Charles Charles
	while (next 1 != NULL 28 curry != NULL) }
	i house accided
	if ((cum 2 > data >= cum + data) &
	(currendata <= next1 = data)) &
	curry - next = cutr 2;
	next 2 = curr2 = next;
	Curre > next = next 1;
	Solve (second, first) i
	"updating pointer ?
	curr 1 = curr 2;
	Curre = next 2;
	j
	,

	the second of th
	else & //go one step ahead in first lis
	curr 1 = next 1:
Labor 193 Chris	next1 = next1 > next;
STATE OF STATE OF	AT ANY AND ADDRESS OF THE ANY ADDRESS OF THE ANY ADDRESS OF THE AD
	If (next 1 == NULL) &
	Curry - next = curry;
	return first;
	,
	9
78.5	return first;
9	A few cases a significant frequency and
· 1 ×	Service Const. Committees and an extension
Nodesin	ty sort Two lists (Node < int) first, Node < int) second
S Commit	(first = NULL)
	return second;
Ai data) Sil	(second == NULL)
4 Cathric	(return first; (res)
del rado es la	(first > data <= second > data) &
	solve (first, second);
	1 18 2 17 5 5 mm = 5 3 x 5 x
- 3	
	else france de la company
	else § · · · · · · · · · · · · · · · · · ·
	Solve (second, first);
	Solve (second, first);

0	Check falindrome in a Linked List
	Abbmach 1'-
	Approach 1:-
	step1:- create an arrow.
	step2: - copy UL content into array.
	Step3: - Write logic to check palindrome in
	$T: C \rightarrow O(N)$ , $S: C \rightarrow O(N)$
	1 C > O(N), S C > O(N)
	Program :-
	Marie transporti man elementi
	for copying il content into array.
	bool is Palindrome (Node * head)
	• §
	vector (int) arr;
	Nodet temp = head ;
	while (temp != NULL) {
	arr. Dush back (temp > data);
	temper temp - next in and
	F (JIDH = 1 chood) offited
	return checklalindrome (arr);
	· }
7	Step'31 Ka function
	itsed to show Jo hmilikh skte hai
	the second of the second
	Flance with birt and wall
	Filesonial Account of the Santon April 199

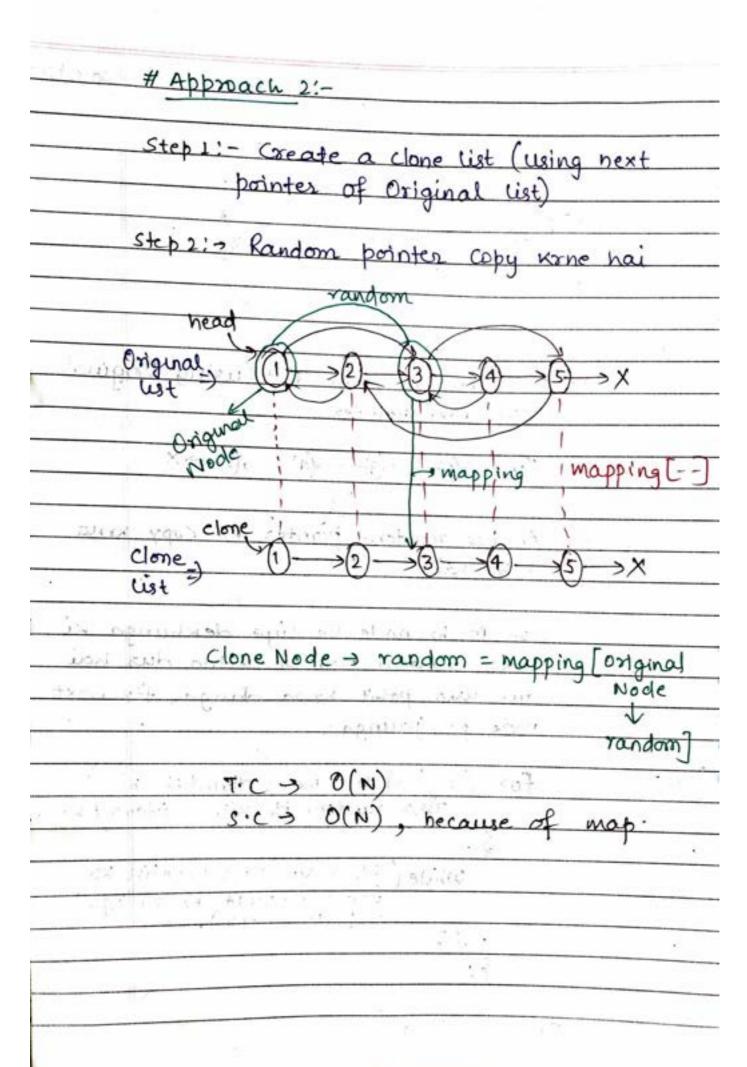
	1 T.C
Step 1:0 find middle	O(N)
Step 2:3 reverse L.L after it	0(N)
Step 3: > compare both halves	O(N)
Step 4:3 repeat step 1.	(n) O
SCOTIS TEPERA STEP 1.	
T· c > O(N) , S· c - O(1)	
, , , , , , , , , , , , , , ,	
Program !-	
bool is Palindrome (Node * head) {	+
if (head == NULL   head > nex	Nun )
return true;	£ == (VOLL)
// step 1 :> find middle	
Node * middle = get Mid Chec	٠,١٠
// Step 2 :> reverse 1.1 a	
Node* temp = midolle = nex	•
middle = next = reverse (	2000
11 Step 3 :> Compare both	The second secon
Node head = head;	
Node head 2 = middle > next	
while (head 1 = NULL) {	
if (heads > data = head	2 -> data)
return false;	\$
head1 = head1 = next;	
head2 = head2 = next	Care - Land Control of the Control o
4	
// step 4 1-> repeat step 2	· (Optional)
temp = middle = next;	
middle -> next = reverse  // Yahan the aa gya toh true he return true;	(temp);

```
first = [2,9,3], second = [5,6,4)
0/p = [7,0,8].
```

(X4/ Explanation =) 342+465 = 807 Add two numbers represented by linked list. Algorithm! > Step1: > Reverse both UL Step 2: > Add them from left Step3: > Teverse ans. Program: Node \* veverse (Node head) { Vold insert At Tail (struct Node & head, struct Node & tail, int vai) { Struct Node add (struct Node first, struct Node second) & int carry = 0; Node anstead = NULL; Node anstail = NULLis = mass while (first 1= NULL || second != NULL || carry != 0) { Mode ans - of detury this end; if (first != NULL) vall = fixst > data; ORE TREETISE (CHIS) .. int val 2 = 0; if (second 1= NULL) val2 = Second → data: int sum = carry + val1 + val2; int digit = Sum 1/0;

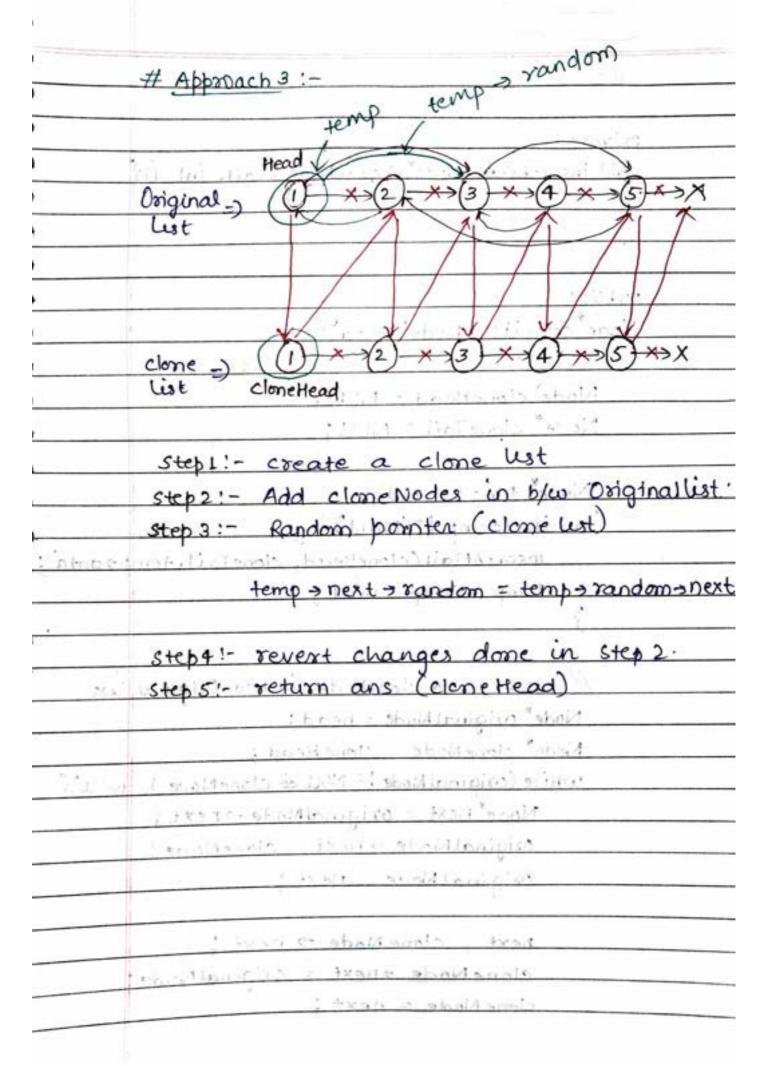
11 create node and	add in ans Lil
insert AtTail (ans Head	, anstail, digit);
the most and the	
carry = sum/10;	
if (first != NULL)	
first = first =	next;
if (Second 1= NULL)	
second = secon	d - next
f return anstread;	extract to a
роьйс:	;
Stauct Node add Two Lists (Struct	Node first, struct Node
- the same served through the server to be	Second) {
# Step1 → reverse int	second) {
# Step 1 -> reverse inp first = reverse (first);	* second) {
# Step1 → reverse int	* Second) {
# Step 1 -> reverse into first = reverse (first);  Second = reverse (second	* Second) {
# Step 1 -> reverse inp first = reverse (first); Second = reverse (second	Second) {
# Step 1 -> reverse into first = reverse (first);  Second = reverse (second	* Second) {  Digitalization  t, Second);
# Step 1 -> reverse interest first = reverse (first);  Second = reverse (second  # Step 2 -> add 2 L.L.  Node ans = add (first)	* Second) {  Digitalization  t, Second);
# Step 1 -> reverse interest first = reverse (first);  Second = reverse (second  # Step 2 -> add 2 L.L.  Node ans = add (first)	t, Second);
// Step 1 -> reverse inp first = reverse (first); Second = reverse (second // Step 2 -> add 2 L.L. Node ans = add (first) // Step 3 -> reverse and	second) {  but li  t, second);
// Step 1 -> reverse inp first = reverse (first); Second = reverse (second // step 2 -> add 2 L.L. Node ans = add (first) // step 3 -> reverse and ans = reverse (ans);	second) {  but lil  t, second);
// Step 1 -> reverse inp first = reverse (first); Second = reverse (second // Step 2 -> add 2 L.L. Node ans = add (first) // Step 3 -> reverse and ans = reverse (ans);	second) {  but li  t, second);
// Step 1 + reverse inp first = reverse (first); Second = reverse (second // step 2 + add 2 LiL Node ans = add (first) // step 3 + reverse and ans = reverse (ans);	second) {  Second) {  Second) ;  t, Second) ;  had

. (1)	Approach 1:-
	head
	$(1)$ $\rightarrow (2)$ $\rightarrow (3)$ $\rightarrow (4)$ $\rightarrow (5)$ $\rightarrow X$
	Steps: - Create a clone list (using Origina
	list next pointer).
	(1) -> (2) -> (3) -> (3) -> (3) -> (4) -> (5) -> (4)
	0 10 10 10
_	
-	Step 2: - Ab bss random pointer Ko Copy Krna
	bcha hai.
_	1603
	Toh M hr node ke lige dekhunga k
	uska random pointer kitna dur ha
	aus usko point Krwa dunga, fir ne
	node pe jaunga.
21	
	For Original Lust mein dhund she ki ) uska random ptr. Kitna distance p
	uska random pto Kitna distance PI
	S.
	O(n2) While (Jotk main clone watelist Ko
	iske random blo Ko shi igh
	ske random ptr. ko shi jgh nhi lga deta).
	, 7 9.



	Program!-
	Private:
	void insertAtTail (Node & shead, Node & stail, intd) &
the year	3
-1000	A Contract of the second of th
	public:
	Node copy List (Node head) &
	//create a clone list (step 1)
	Node clone Head = NULL;
	Node * clone Tail = NULL;
	Node* temp = head;
1	while (temp 1= NULL) {
	insert At Tail (clone Head, clone Tail, temp > do
	temp = temp -> next;
-	3
	//step2:- create a map
	unordered-map < Node*, Node* > old To New Node
	Node original Node = head;
	Node clone Node = clone Head.
	while (original Node!=NULL && clone Node!= NULL)
	while (original Node!=NULL & cloneNode!=NULL)  Old To New Node [original Node] = clone Node
	original Node - next.
	cloneNode = clone Node -> next:
	}

// Se	t random pointer
orig	inalNode = head;
clon	e Node = clone Head;
the state of the s	A trends to me the hard of the
whi	le (original Node != NULL) {
	Clone Node > random = oldToNewNode [original Node
	→ random];
	original Node = original Node > next;
ž .	cloneNode = cloneNode > next;
ret	orn cloneHead;
	A first Continues for all
	There's that boat
	Fright Clare Center to a
Supperinter Contracts	de amende tierale, en
	table end was a worst
	V.
	A STATE AND A SERVICE OF THE SECOND S
tel sidmidal els d'obs	and the state of t
	the set of settlements the sector
	-Manala : Abablayela *abell
Morning telephonets	Security Control of the Control of t
	rate riginal sharker techto
2.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1	extract at a the thirty of
	shadanali sanakani.



Program!-	
Private:	
vold insertAti	iail (Node* Shead, Node* Stail, int d) {
3	F12
public:	
Node copy Li	st (Node* head) {
	create a clone list
Node* cla	oneHead = NULL;
Node* cl	one Tail = NULL;
Jest	payer's to the state of the sta
	emp = head;
while (	temp 1= NULD &
	rtAtTail (cloneHead, cloneTail, temp > data
	= temp = next;
	sk regionale services
	- Add cloneNodes in b/w Original list
	riginal Node = head;
Node* c	lone Node = clone Head;
	original Node != NULL & clone Node != NULL) {
	le*next = originalNode → next;
	ginal Node > next = clone Node;
0න්	ginal Node = next;
	t = clone Node > next;
	ne Node + next = original Node;
clor	e Node = next;
· <b>š</b>	

THE WILL	
	// Step 3: - random pointer copy
	temp = head;
	while (temp 1= NULL) &
	if (temp = next 1= NULL) { - tobal }
	temponext o random = temporandom
Harris	? temp = random = next : temp = random
	temp = temp > next > next;
	3
	1 186.1 (486.5)
	1/step4: revert changes done in step 2
-	original Node = head;
	clone Node = clone Head; 1 9 1911 9hold
	1 (1991 - 1995) N
	while (original Node 1= NULL & clone Node 1= NULL) &
	original Node > next = clone Node > next;
	original Node = original Node -> next;
	If Conginal Nede 1= NULUE in "about
	cloneNode + next = original Node > next;
	clone Node = clone Node = next;
	Finish advois > ptop a tast it
	i has a tasa a dunat
	// step 5 in return ains
	return clone Head;
	)
	I pulping I trans a great to gate
	Litable against
	A desire explicit supplies
	The state of the s

D Merge Sort in Linked list
The art of
Node* findMid (Node* head) {
Node slow = head;
Node fast = head = next;
while (fast 1 = NULL && fast = next != NULL)
Slow = Slow > next;
fast = fast > next > next;
return slow;
Node * merge (Node ! lett , Node * right) {
if (left == NULL)
return right;
if (right == NULL)
return left;
Node ans = new Node (-1);
Nodet temp = ans;
//merge 2 sorted L.L
while (left 1= NULL & right 1= NULL) }
if (left > data < right > data) f
temp > next = left;
temp = left;
left = left > next;
3
else & temp = next = right;
temp= right;
right = right = next;
, }

```
while (text 1= NULL) &
      temp > next = right;
      temp = right;
    right = right > next;
      tebt
   cohile (right 1= NUL) {
      temp = next = left;
     temp = left;
    left = left = next;
   ans = ans > next;
   return ans;
Node merge Sort (Node + head) {
   // base case
   if (head == NULL | head = next == NULL)
      return head;
   Il break il into two halves, after finding Mid
   Node * mld = find Mid (head);
   Node left = head;
   Node* right = mid > next;
   mid = next = NULL;
   Precursive calls to Sort both halves
   lett = mergeSort (lett);
   right = merge Sort (right);
   // merge both left and right halves
   Node * result = merge (left, right);
   return result;
```

Delete Node in a linked list (leatcode 237)	
THE TRACE OF A WINES USE	
NOTE: > you will not be given access to the	
head of lust, instead only the node	
to be deleted directly.	
oc deleta arrects	
Program:	
void deleteNode (Node* node) {	
*node = *node -> next;	
}	
2.3.46.000	
(or)	
node = val = node = next = val;	
node s'next = node snext + next;	_
(JJUN TE FROME KINDER   JAMES - TEACH IN	_
- Check and - c	_
hint of the contract of the co	_
Money to the third the and the	_
Throat that "at all	-
A Kara teles instanting a Marcol	_
Title & Chickens	
	_
2 Charles Carry and the start	
The tribute of tribu	
To pay the control of the second of the seco	_
TO THE STATE OF TH	_