

27

FRIDAY

MAY

2-0-2-2

Week-22 (147-218)

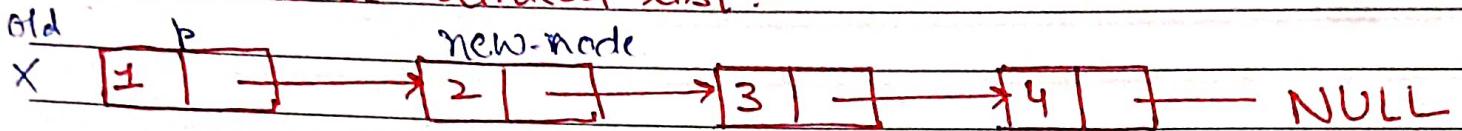
May

2022

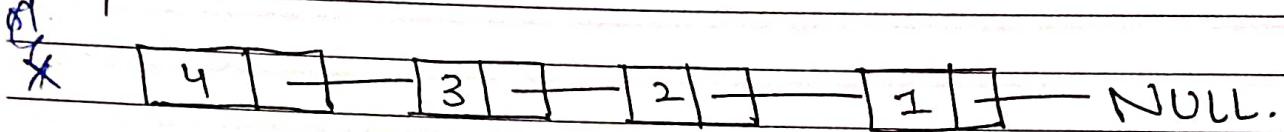
S	M	T	W	F	S	S	M	T	W	F	S
					1	2	3	4	5	6	7
8	9	10	11	12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27	28	29	30	31

LINKED-LIST.

Reverse a linked-list :-



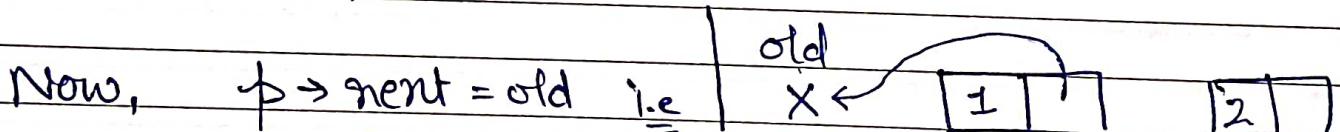
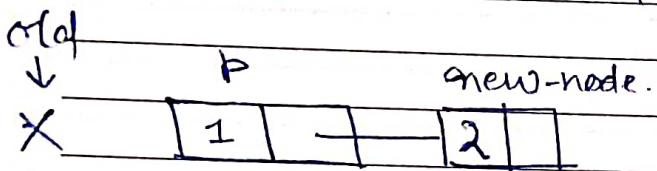
Output:



→ Use three pointers:

Set, old = Null, p = head!

new-node = p->next;



Now, Repeat till p != NULL.

but change/increment the value of old, p, new-node.



old = p, p = new-node;

// Checkout code for better understanding.

June

2022

S	M	T	W	T	F	S	S	M	T	W	T	F	S
1	2	3	4	5	6	7	8	9	10	11			
12	13	14	15	16	17	18	19	20	21	22	23	24	25
26	27	28	29	30

SATURDAY

2-0-2-2

28

MAY

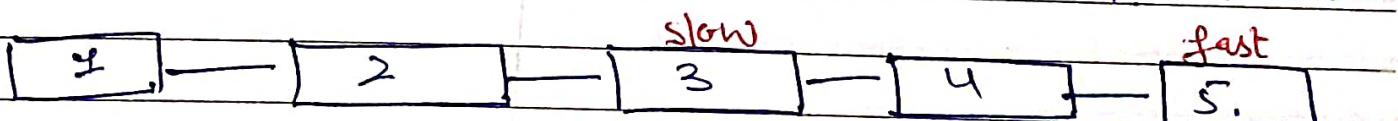
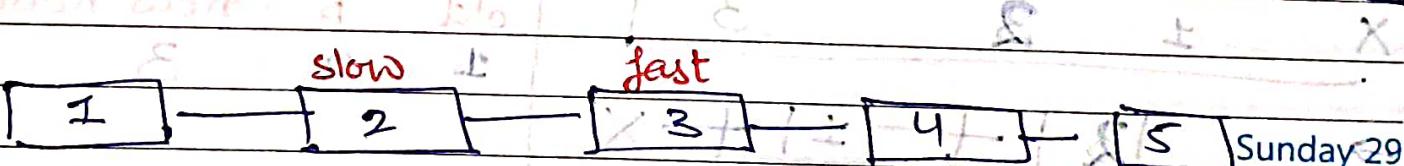
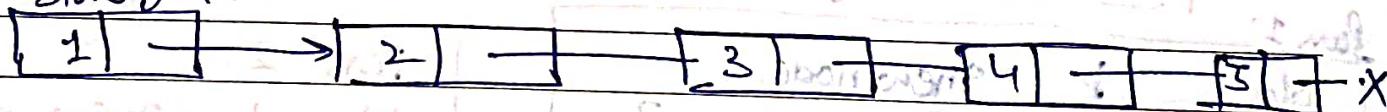
Week-22 (148-217)

Find Middle-node of a linked-list:-

Approach 1: (Brute force) \Rightarrow get length of listi.e. 5, now $(\text{length } 12) + 1^{\text{th}}$ element is the middle.for odd cases, and for even
it depends on the question.

Approach two (hare and tortoise, fast & slow pointers).

In this Approach, we declare 2 pointers slow and fast, such that with each pass fast moves two times and slow moves 1 time.

if $(\text{fast} \rightarrow \text{fast} \rightarrow \text{next}) == (\text{NULL})$ return slow.

That's the basic approach here.

30

MONDAY

MAY

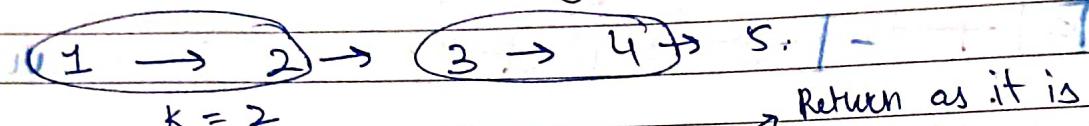
2-0-2-2

Week-23 (150-215)

2022

May							June						
S	M	T	W	F	S	S	S	M	T	W	F	S	
					1	2	3	4	5	6	7		
					8	9	10	11	12	13	14	15	16
					17	18	19	20	21				
					22	23	24	25	26	27	28	29	30
					31								

Reverse Nodes in K-groups:-



2 → 1 → 4 → 3 → 5.
 if they're not in pair.

Return as it is

Recursive Approach:-

1.) Solve for k-nodes.

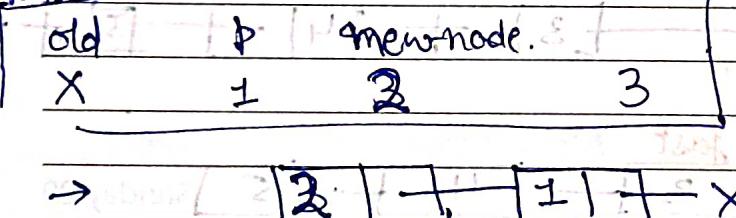
2.) Call recursion for solving next nodes.

Base-Case :- if (length(head) < k) return head;

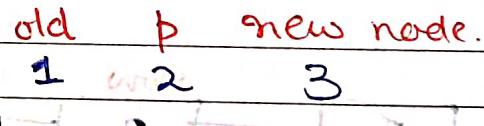
// Cause we don't have to handle this kinda case!

| Solve 1 Case | Reverse with nodes! (using basic Algo).

Pass 1:



Pass 2:-



if (new-node != NULL): basically there are nodes left to solve.

head → next = reverse (new-node, k);

4

June

2022

S	M	T	F	S	S	M	T	W	T	F	S
1	2	3	4	5	6	7	8	9	10	11	
12	13	14	15	16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	•	•	•	•	•

TUESDAY

31

2-0-2-2

MAY

Week-23 (151-214)

As head is going to be the last element of reversed group, so, we are supposed to attach the next nodes to head. So,

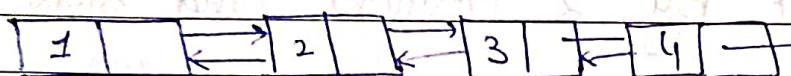
→ unfinished

$\text{head} \rightarrow \text{next} = \text{reverse}(\text{new-node}, k)$

Here we're parsing

new-node as head!

Reverse a Doubly-linked list:-



while ($b \neq \text{NULL}$) { | create new = $b \rightarrow \text{next};$

$\Rightarrow b \rightarrow \text{next} = \text{old};$ | next attach if

$b \rightarrow \text{prev} = \text{new};$ | know b becomes head

$\text{old} = b;$ | now b is old

$b = \text{new};$

$\{$ | initialization

 head = old;

$b = \text{old};$ |

$b = \text{old};$ |

$(\text{last} = \text{null}) \rightarrow \text{last} = b;$

 start source loop, head = old; |

 modified it and attached tail from end with

$b = \text{old};$

 tail attach tail from end attached again

 tail to tail until both ends with old B&G

$b = \text{old};$

01

WEDNESDAY

JUNE

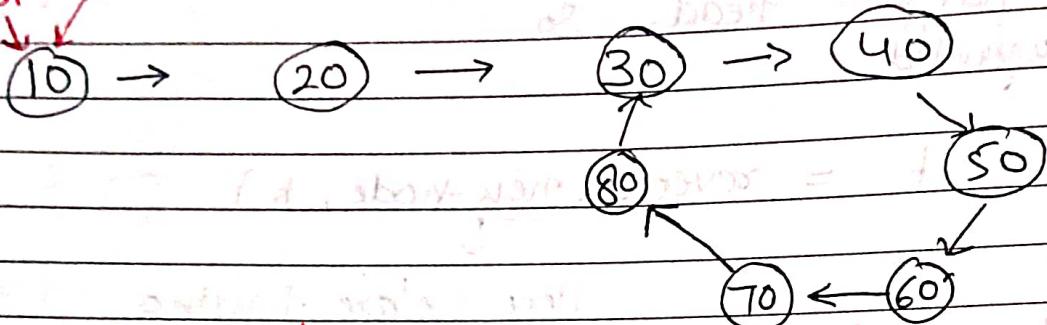
2-0-2-2

Week-23 (152-213)

June 2022											
S	M	T	W	T	F	S	S	M	T	W	F
			1	2	3	4	5	6	7	8	9
			10	11	12	13	14	15	16	17	18
			19	20	21	22	23	24	25	26	27
			28	29	30	•	•	•	•	•	•

Floyd's Algorithm to Detect Loop in a linked-list

fast slow



★ To Detect a Cycle:-

→ Initialize two pointers slow & fast :-

→ Move fast pointer by 2 positions and slow pointer by 1 position.

→ if both pointers meet at some point then a loop exists and if fast pointer meets the end position then no loop exists.

→ To Check the Starting point of Cycle:

→ first do repeat the above steps and when ($slow == fast$).

→ Put $Slow = head$, and move both slow and fast pointer by 1 position.

→ Now where slow and fast pointers will meet, it'll be the starting point of loop.

July

2022

S	M	T	W	F	S	S	M	T	W	F	S
1	2	3	4	5	6	7	8	9			
10	11	12	13	14	15	16	17	18	19	20	21
22	23	24	25	26	27	28	29	30	31	•	•

THURSDAY

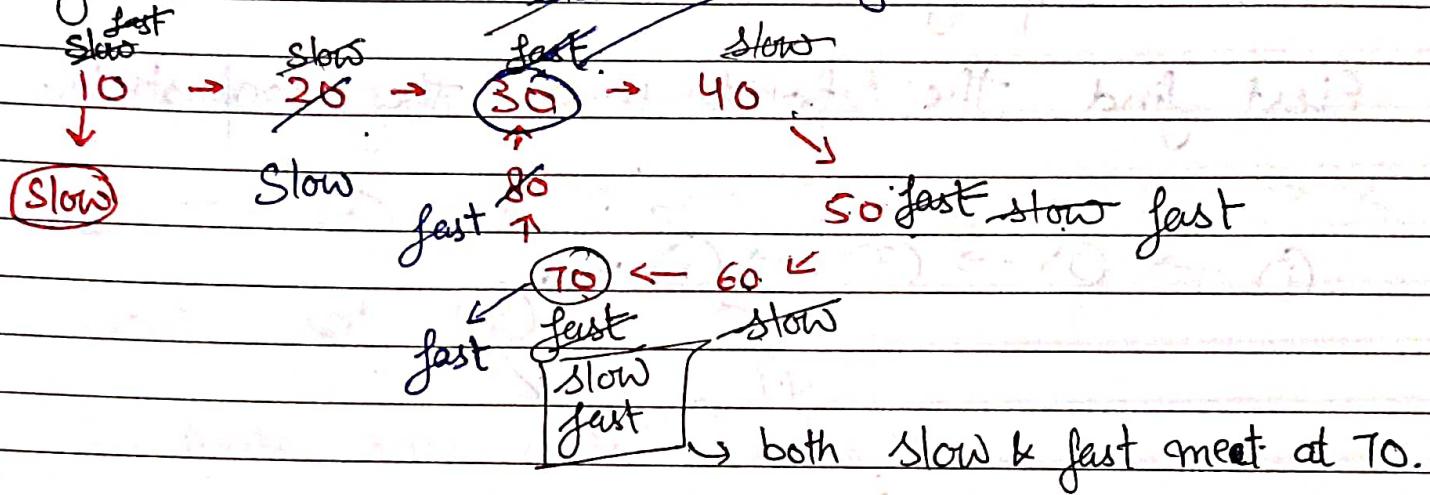
02

2-0-2-2

JUNE

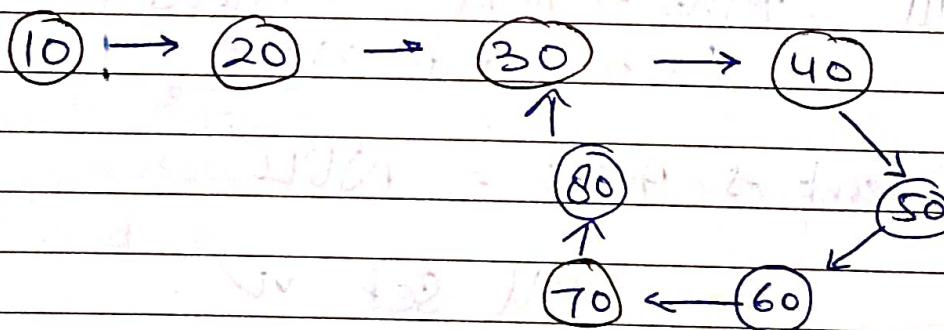
Week-23 (153-212)

Dry-Run :-



Now, Slow = head

We can see that Slow equals fast at 30th Node.
Hence, we can say that the loop begins at 30.



03

FRIDAY

JUNE

2-0-2-2

Week-23 (154-211)

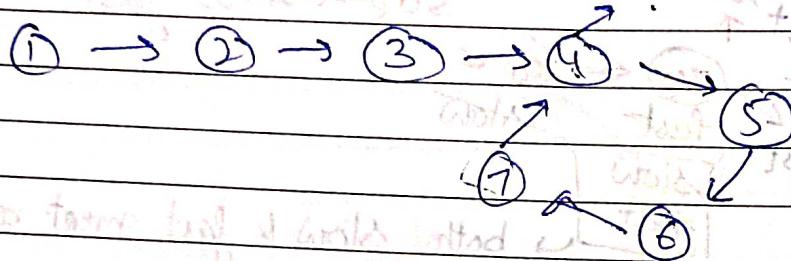
June

2022

S	M	T	W	F	S	S	M	T	W	F	S
1	2	3	4	5	6	7	8	9	10	11	
12	13	14	15	16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	•	•	•	•	•

Remove Loop from a linked-list :-

First find the point where the loop starts.



* Using Floyd's algo. find the starting point of loop.

Now take a variable, let's say temp. and initialise it with 'slow->next';

Now till $\text{temp} \rightarrow \text{next} != \text{Slow}$
 $\text{temp} = \text{temp} \rightarrow \text{next}$

Now, $\text{temp} \rightarrow \text{next} = \text{NULL}$

All set ✓

July

2022

S	M	T	F	S	S	M	T	F	S
1	2	3	4	5	6	7	8	9	
10	11	12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27	28	29

Week-23 (155-210)

SATURDAY

12-0-2-2

04

JUNE

Sort 0's 1's and 2's.

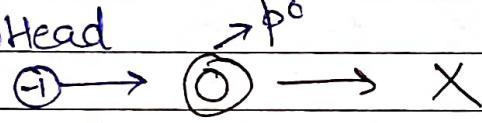
Suppose, given a linked-list:-



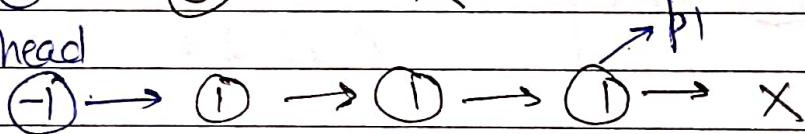
Now sort them

(i) Make three dummy Nodes :-

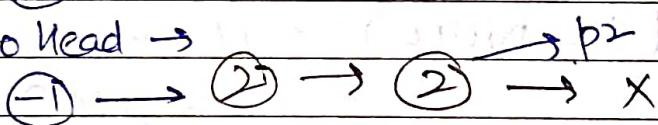
ZeroHead



Onehead



TwoHead

Iterate through the linked-list and
if ($p \rightarrow \text{data} = 0$) put it next to zerohead
and so on.

Sunday 05

Now join these three nodes but first
remove that dummy nodes using.Zhead = Zhead \rightarrow next;Onehead = Onehead \rightarrow next;Twohead = Twohead \rightarrow next;

06

MONDAY

JUNE 2022 2-0-2-2

Week-24 (157-208)

June							2022				
S	M	T	W	F	S	S	M	T	W	F	S
1	2	3	4	5	6	7	8	9	10	11	
12	13	14	15	16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	•	•	•	•	•

Now Join these three linked-lists
using:-

if (zerohead != NULL)

if (onehead != NULL)

$\{ p_0 \rightarrow next = onehead; \}$

$p_1 \rightarrow next = twohead;$

else $p_0 \rightarrow next = twohead;$

else

{

if (onehead != NULL)

$\{ zerohead = onehead;$

$p_1 \rightarrow next = twohead;$

else $zerohead = twohead;$

}

return zerohead;

2.

July 2022						
S	M	T	W	F	S	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

Week-24 (158-207)

TUESDAY

2-0-2-2

07

JUNE

Add two numbers represented by linked-lists.

List 1: $6 \rightarrow 3 \rightarrow X$

List 2: $7 \rightarrow X$

$$\begin{array}{r}
 & & 1 \\
 & & 6 \leftarrow 3 \\
 & + & 7 \\
 \hline
 & 7 & 0
 \end{array}$$

Reverse both the linked lists.

Carry = 0. $3 \rightarrow 6 \rightarrow X$

$7 \rightarrow X$

while ($L_1 \neq L_2$)

Sum = $L_1 \rightarrow \text{data} + L_2 \rightarrow \text{data}$;

Carry = sum / 10;

Sum = sum % 10;

ans $\rightarrow \text{next} = \text{new node}(\text{sum})$;

} Main Thing

Baki khud karo.

Now if L_1 is not equal to NULL

Sum = $L_1 \rightarrow \text{data} + \text{Carry}$;

Carry = sum / 10;

Sum = sum % 10;

Same goes for L_2 .

if still carry remains; ans $\rightarrow \text{next} = \text{carry}$;
reverse the ans; after removing
dummy node.

It is possible to fail in many ways while to succeed is possible only in one way. - Aristotle

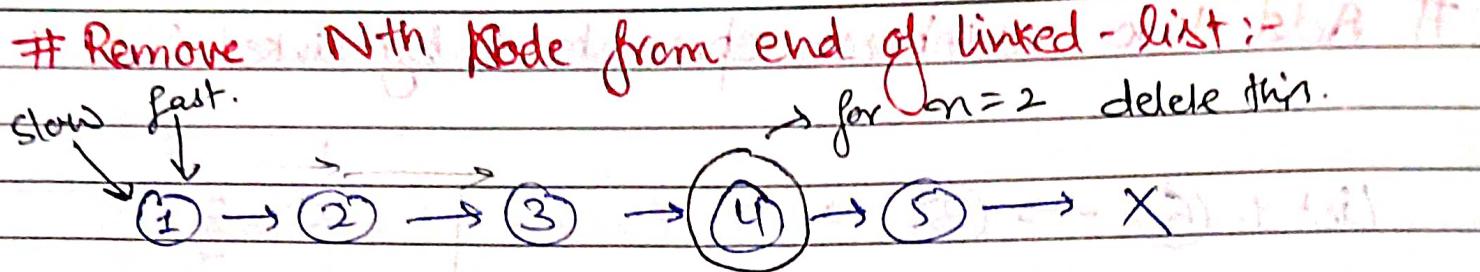
08

WEDNESDAY

JUNE 2-0-2-2

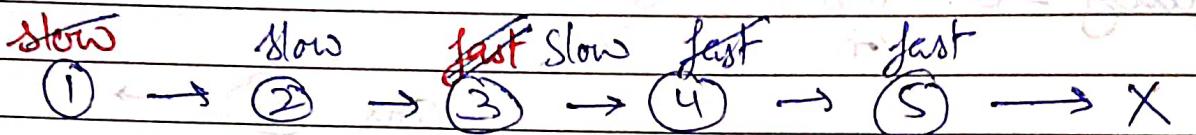
Week-24 (159-206)

June 2022											
S	M	T	W	F	S	S	M	T	W	F	S
					1	2	3	4	5	6	7
					8	9	10	11			
					12	13	14	15	16	17	18
					19	20	21	22	23	24	25
					26	27	28	29	30	*	*



Put two pointers slow & fast on head

for (i < n) {
 fast = fast->next;
 if (i == n - 1) {
 // do something
 }



while (fast->next)

Slow->next = Slow->next->next;

Slow = Slow->next;

fast = fast->next;

slow = slow->next;

4. linked list deletion = free & NULL

Slow->next = Slow->next->next;

Slow = Slow->next;

fast = fast->next;

fast->next = NULL

fast = fast->next;

Slow->next = Slow->next->next;

Slow = Slow->next;

fast = fast->next;

Try not to become a man of success, but rather try to become a man of value. — Albert Einstein

July

2022

S	M	T	W	F	S	S	M	T	W	F	S
1	2	3	4	5	6	7	8	9			
10	11	12	13	14	15	16	17	18	19	20	21
22	23	24	25	26	27	28	29	30	31	•	•

THURSDAY

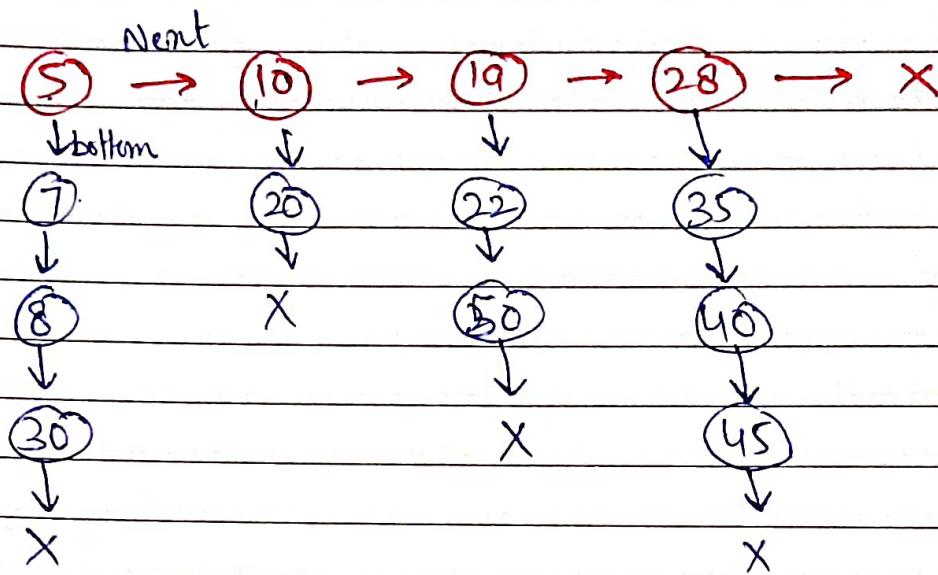
09

2-0-2-2

JUNE

Week-24 (160-205)

Flatenning a linked-list:-



first use recursion to merge two nodes into one.

Take ~~two~~ one dummy node

ans = NULL;

(5) basically
we're doing
this.
↓
(7)
↓
(8)
↓
(10)
↓
(20)
↓
(30)
↓
X

if ($a \rightarrow \text{data} < b \rightarrow \text{data}$)
 {
 ans = a;
 ans \rightarrow bottom = ($a \rightarrow \text{bottom}, b$);
 }
else {
 result = b;
 ans \rightarrow bottom = ($a, b \rightarrow \text{bottom}$);
 }
return result;

10

FRIDAY

JUNE

2-0-2-2

Week-24 (161-204)

June						2022					
S	M	T	W	F	S	S	M	T	W	F	S
					1	2	3	4	5	6	7
12	13	14	15	16	17	18	19	20	21	22	23
26	27	28	29	30	*	*	*	*	*	*	*

Now merge from last node & persist it
merge all blah, blah.

$x \leftarrow (89) \leftarrow (91) \leftarrow (91) \leftarrow (2)$

J J most

Digitized by srujanika@gmail.com

66 67 68 69

(34) *Amphibolite* *metavolcanic* *metabasic* *metamorphic* *metasedimentary*

1996-1997 学年 第一学期

1988-1990
1990-1992
1992-1994
1994-1996
1996-1998
1998-2000
2000-2002
2002-2004
2004-2006
2006-2008
2008-2010
2010-2012
2012-2014
2014-2016
2016-2018
2018-2020
2020-2022
2022-2024
2024-2026
2026-2028
2028-2030
2030-2032
2032-2034
2034-2036
2036-2038
2038-2040
2040-2042
2042-2044
2044-2046
2046-2048
2048-2050
2050-2052
2052-2054
2054-2056
2056-2058
2058-2060
2060-2062
2062-2064
2064-2066
2066-2068
2068-2070
2070-2072
2072-2074
2074-2076
2076-2078
2078-2080
2080-2082
2082-2084
2084-2086
2086-2088
2088-2090
2090-2092
2092-2094
2094-2096
2096-2098
2098-20100

• 15

www.oceanus.org

Page 10 of 10

Re: ~~incorrect~~ ~~old~~ article

Page 1

ANSWER

all rights reserved

...and the last time I saw you

1. $\frac{1}{2} \times \frac{1}{2} = \frac{1}{4}$
2. $\frac{1}{2} \times \frac{1}{2} = \frac{1}{4}$

Page 10 of 10

10. *Leucosia* (L.) *leucostoma* (L.) *var.* *leucostoma*

中華人民共和國

Digitized by srujanika@gmail.com