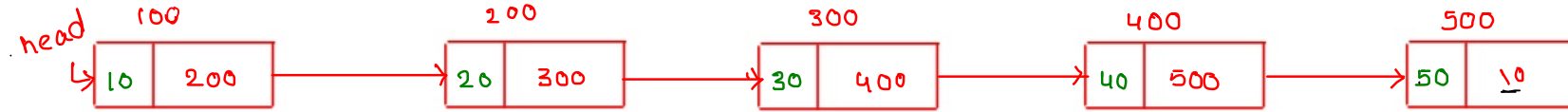


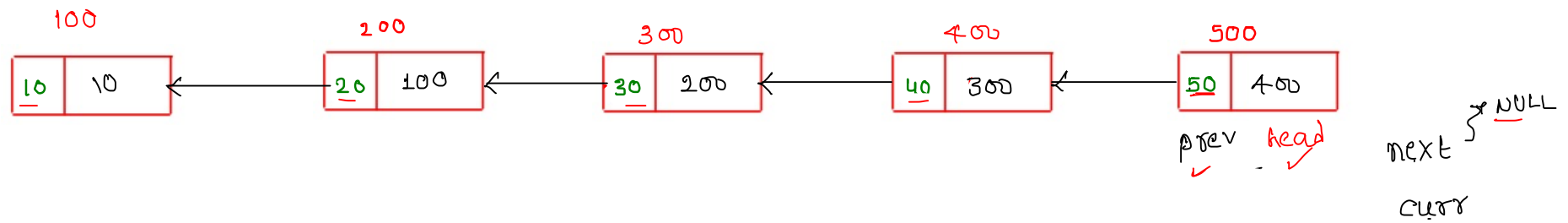
6) Reverse a SLL [ Accolite, Adobe, Amazon, MakeMyTrip, Qualcomm, Samsung, SAP Labs, Snapdeal, Zoho ]

i/p



i/p :- 10 → 20 → 30 → 40 → 50 ✓

o/p :- 50 → 40 → 30 → 20 → 10



Node curr = head, prev = NULL, next = NULL

while (curr != NULL)

1. next = curr.next

2. curr.next = prev

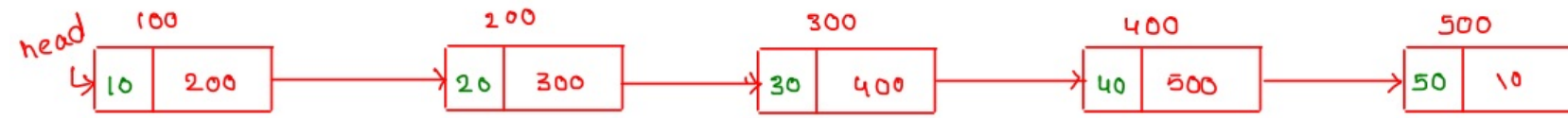
3. prev = curr

4. curr = next


}

head = prev

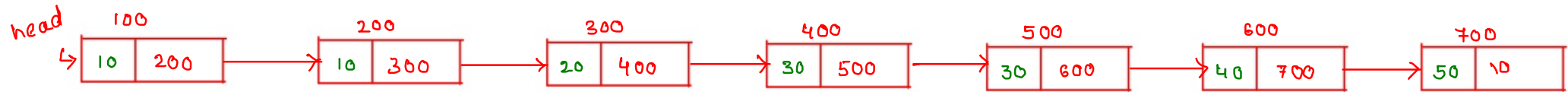
return head;



```
function reverseSLL(head)
{
    ✓ prev=null
    ✓ curr=head
    ✓ next=null
    while(curr!=null)
    {
        1. next=curr.next
        2. curr.next=prev
        3. prev=curr
        4. curr=next
    }
    head=prev
    return head
}
```



## 7) Remove duplicates from sorted list [Adobe, Myntra, Visa]



✓ i/p :- 10 → 10 → 20 → 30 → 30 → 40 → 50

o/p :- 10 → 20 → 30 → 40 → 50

⇒ ONLY

distinct ✓

HW

long-run ✓

```
function removeDuplicates(head)
{
    temp=head, prev=head
    while(temp!=null)
    {
        if(temp.data!=prev.data)
        {
            prev.next=temp
            prev=temp
        }
        temp=temp.next
    }
    if(prev!=temp)
    {
        prev.next=null
    }
    return head
}
```

# 9) Add two numbers represented by SLL [ Accolite, Amazon, Flipkart, Snapdeal, Microsoft, Qualcomm ]

$$\begin{array}{r}
 \text{L} \quad 1 \quad 1 \quad 0 \quad 0 \quad \text{R} \\
 n_1 = 9 \quad 9 \quad 3 \quad 7 \\
 + \\
 n_2 = \quad \quad 9 \quad 4 \quad 2 \\
 \hline
 1 \quad 0 \quad 8 \quad 7 \quad 9
 \end{array}$$

1 → 0 → 8 → 7 → 9

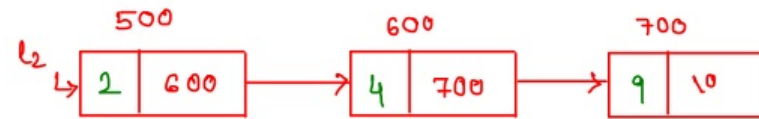
L ← R

9 → 9 → 3 → 7

9 → 4 → 2

} reverse

BUT



9 → 7 → 8 → 0 → 1

```

function addTwoLinkedLists(l1, l2)
{
    ✓ dummy=null
    ✓ temp=dummy
    ✓ carry=0;
    while(l1!=null || l2!=null || carry==1)
    {
        sum=0
        if(l1!=null)
        {
            sum=sum+l1.data
            l1=l1.next
        }
        ✓ if(l2!=null)
        {
            sum=sum+l2.data
            l2=l2.next
        }
        sum=sum+carry
        carry=sum/10
        create a node with the value (sum%10) [ random]
        temp.next=random
        temp=temp.next
    }
    return dummy.next
}

```

$c > 0$   $carry=1$

①  $c > 0$  ✓

$c = 0 \neq 1$   
 $s = 7 \neq 0$

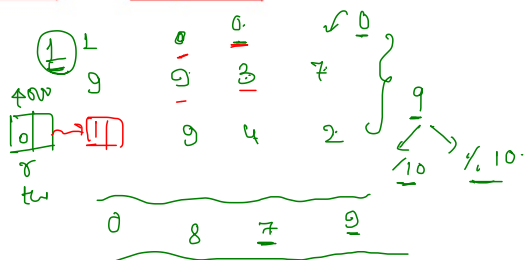
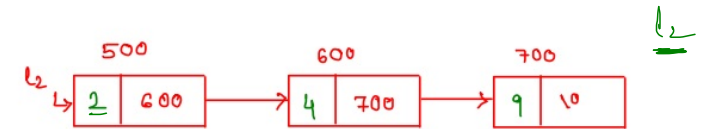
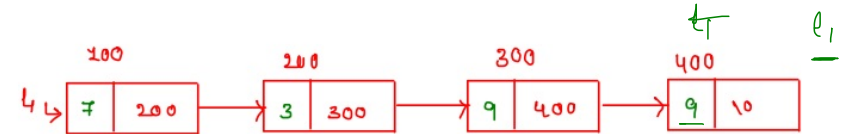
②  $\uparrow$

dummy

temp

9 → 7 → 8 → 10 → 1

9  
9  
18



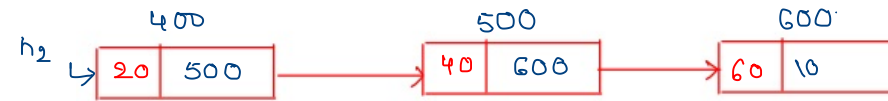
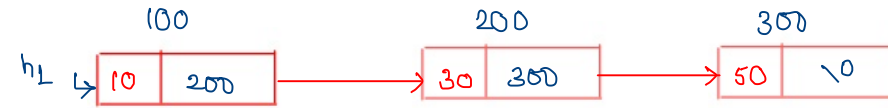
10 18 7.10  
 1 8

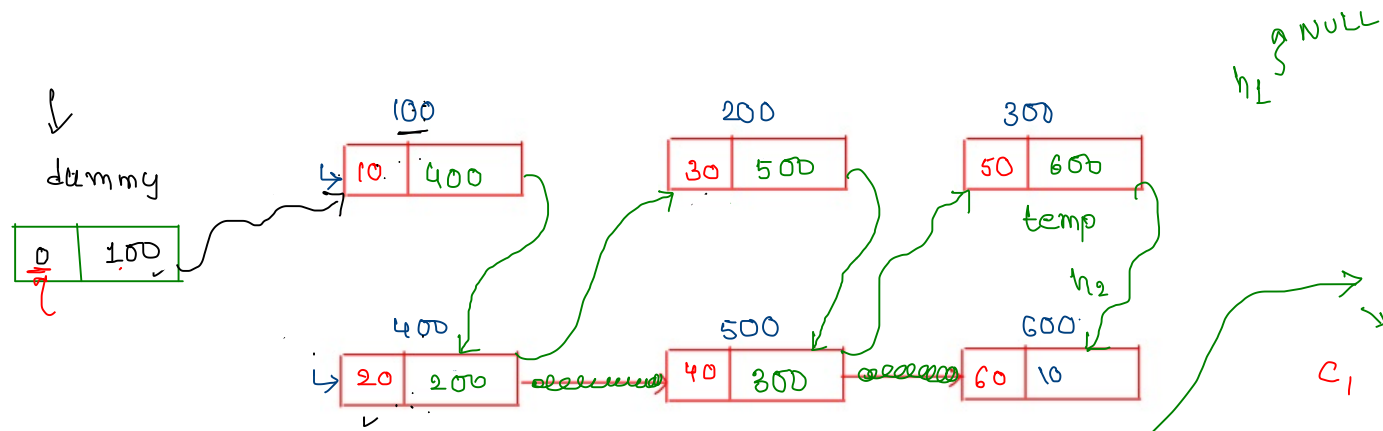


2ptr:- merge 2 sorted arrays

## Merge two sorted Linked Lists

ofp:- 10 → 20 → 30 → 40 → 50 → 60





1) dummy node ✓

2) temp = dummy

$c_1$  if ( $h_1 \cdot data \leq h_2 \cdot data$ )  
 {

temp.next =  $h_1$

$h_1 = h_1 \cdot next$

}

$c_2$  else  
 {

temp.next =  $h_2$

$h_2 = h_2 \cdot next$

}

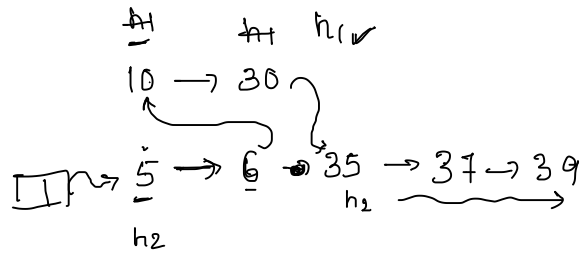
✓ temp = temp.next

return dummy.next

```

if ( $h_1 == NULL$ )
{
    temp.next =  $h_2$ ;
    break;
}
if ( $h_2 == NULL$ )
{
    temp.next =  $h_1$ ;
    break;
}

```



while(true)

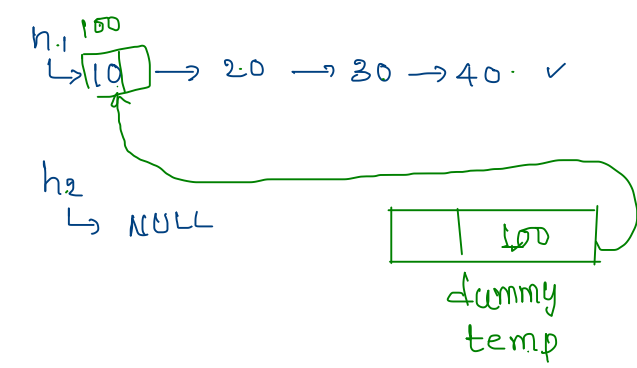
- 1) dummy node ✓
- 2) temp = dummy

```

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

```

if



O/p  
✓✓

```

c1 if(h1.data <= h2.data)
{
    temp.next = h1;
    h1 = h1.next;
}
c2 else
{
    temp.next = h2;
    h2 = h2.next;
}
temp = temp.next;

```

}  
return dummy.next;

```
Node merge(Node h1, Node h2)
{
    Node dummy=new Node(0);
    Node temp=dummy; ✓ some value
    while(true) ∞ loop
    {
        if(h1==null)
        {
            temp.next=h2;
            break; ←
        }
        if(h2==null)
        {
            temp.next=h1;
            break; ←
        }
        C1 if(h1.data<=h2.data)
        {
            temp.next=h1;
            h1=h1.next;
        }
        C2 else
        {
            temp.next=h2;
            h2=h2.next;
        }
        temp=temp.next; ✓
    }
    return dummy.next; ✓
}
```

Ex:-

```
if(Expression1)
{
    variable = Expression2;
}
else
{
    variable = Expression3;
}
```

a, b, c

if ( a > b )

{

c = 10;

}

else

{

c = 12;

}

c ✓

⇒ c = (a > b) ? 10 : 12 ✓

num1 = 10;

num2 = 20;

res = (num1 > num2) ? (num1 + num2) : (num1 - num2)  
T/F T F

res = (10 > 20) ? — : 10 - 20  
F

res = -10