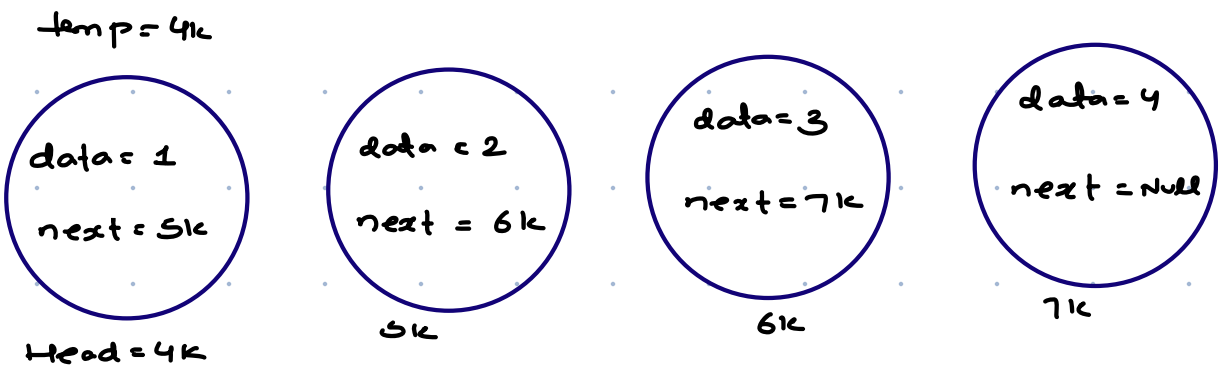
LinkedList

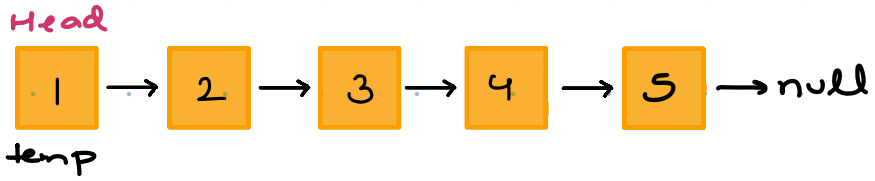
Good Morning

Agenda

01. Search for a value
02. Insertion in LL
03. Deletion in LL
04. Reverse a LL
05. Deep copy of LL (Hard)



Q Given LL, search for a value K



K = 3 ⟶ true

K = 7 ⟶ false

```
boolean search (Node head, int k)
    if (head == Null) return false
    Node temp = head
    while (temp != Null)
        if (temp.data == k) return true;
        temp = temp.next
    return false;
```

TC : O(N)

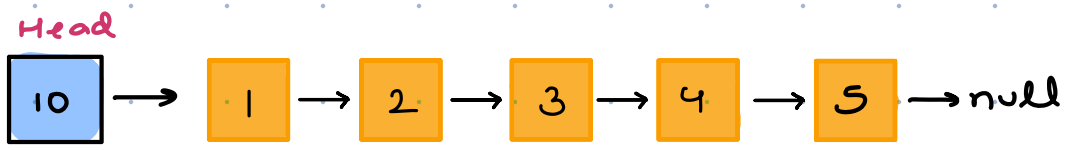
SC : O(1)

* Insert a new Node with Data at K° position

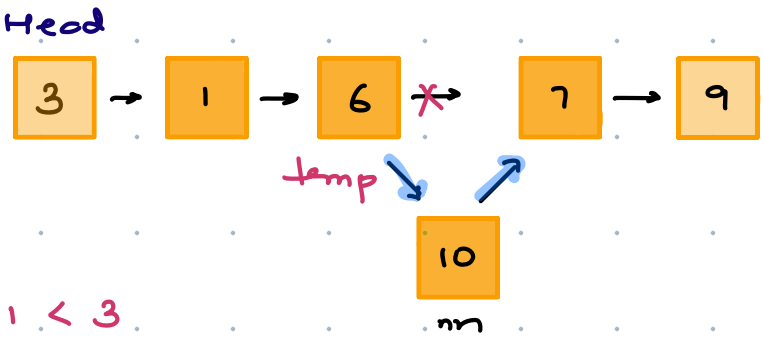
Data = 10

K = 0

} Add in font



```
Node nn = new Node(10);
nn.next = head
head = nn;
```



Data = 10

k = 3

i = 1 < 3

2 < 3

3 < 3 ×

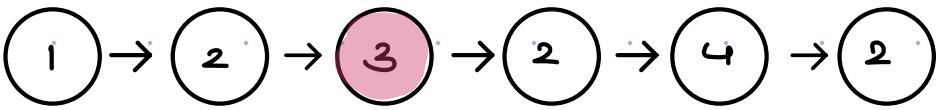
```
Node nn = new Node(10);
Node temp = head
i = 1
while (i++ < k)
    temp = temp.next;
nn.next = temp.next
temp.next = nn;
```

TC : O(N)

SC : O(1)

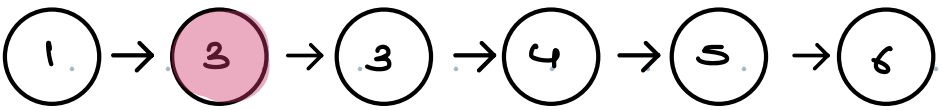
* Deletion in Linkedlist

→ Delete the first occurrance of value x given in Linkedlist. If x is not present, leave LL as it is.



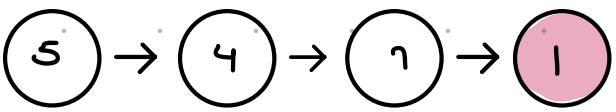
x = 3

Ans = 



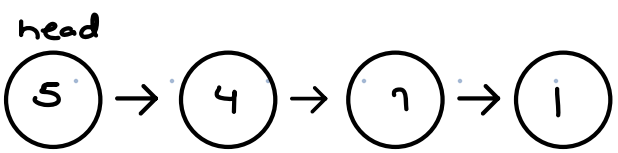
x = 3

Ans = 



x = 1

Ans = 

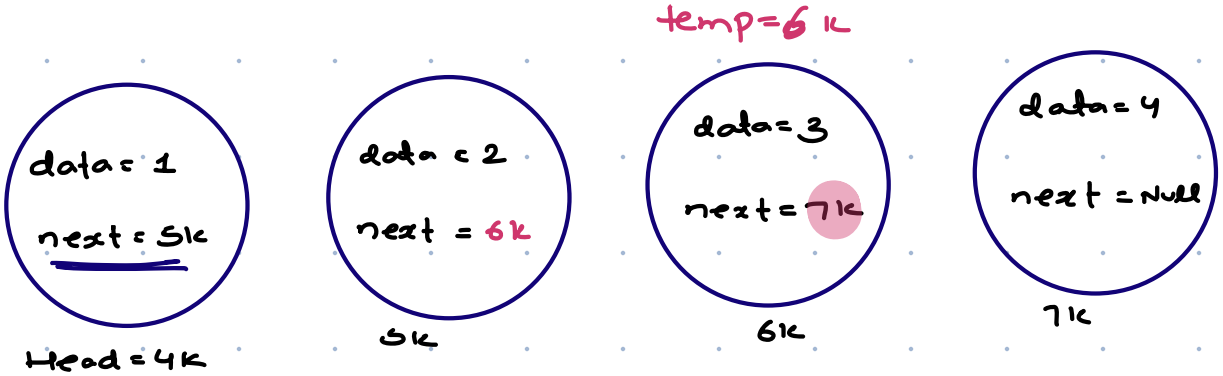


x = 5

Ans 

```
Node deletefirst ( Node head, int x )
{
    if (head == Null) return head;
    if (head.data == x)
    {
        head = head.next;
        return head;
    }
    Node temp = head;
    while (temp != Null && temp.next != Null)
    {
        if (temp.next.data == x)
        {
            temp.next = temp.next.next;
            return head;
        }
        temp = temp.next;
    }
    return head;
}
```

x = 4



OnePlus removes Defects

Scenerio

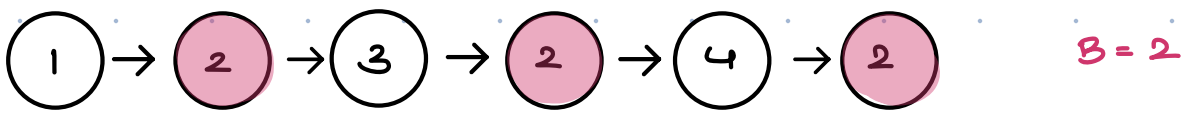
OnePlus has a lineup of **N** mobile phones ready in their manufacturing line. It has detected a defect in one of their phone models during production.

They have decided to recall all phones of the defective model from their manufacturing line. Your task is to help **OnePlus** remove all defective phones from their production lineup efficiently.

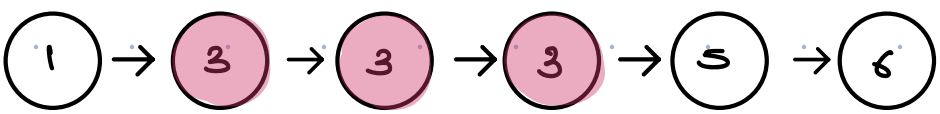
Problem

You are given a **linked list A of N** nodes where each node represents a specific **model type of a OnePlus mobile phone** in the manufacturing line. Each node contains an integer representing the model number of the phone. You will also be given an integer **B** which represents the model number of the defective phone that needs to be removed.

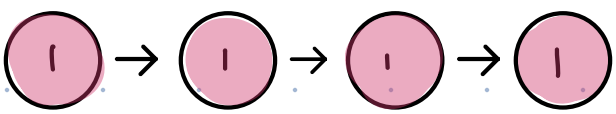
Your goal is to remove all **nodes** (phones) from the linked list that have the model number **B** and return the modified linked list representing the updated manufacturing line.



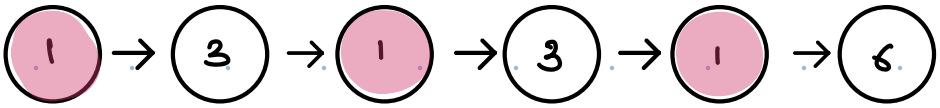
Ans = 1 → 3 → 4



Ans = 1 → 5 → 6

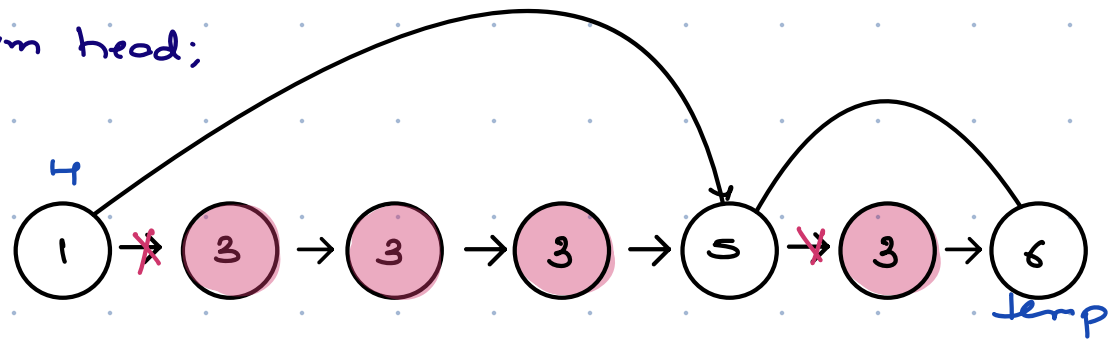


Ans = Null

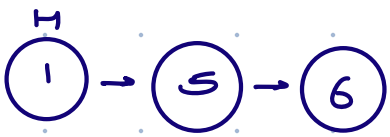


Ans = 3 → 3 → 6

```
while (head != Null && head.data == B)
    head = head.next;

temp = head
while (temp != Null && temp.next != Null)
    if (temp.next.data == B)
        temp.next = temp.next.next;
    else
        temp = temp.next;
return head;
```

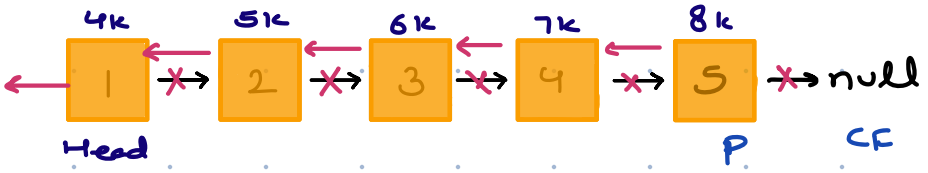


B=3



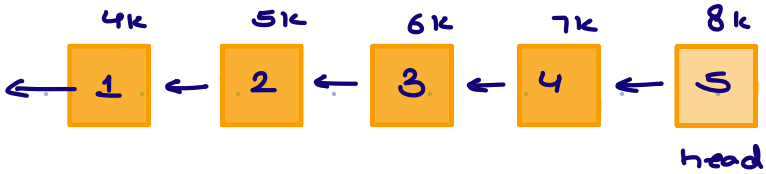
8:20 → 8:30 AM

Reverse a LinkedList

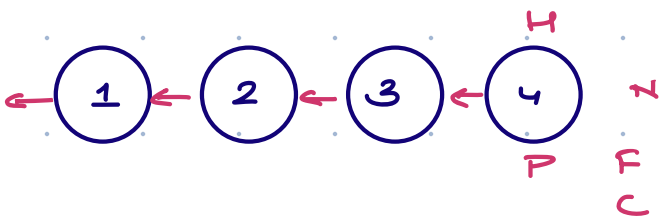


Node forw = curr.next

curr.next = prev




```
Node prev= Null
Node curr = head
while (curr != Null)
{
    Node forw = curr.next;
    curr.next = prev
    prev = curr;
    curr= forw;

}
return head = prev;
```



TC : O(N)

SC : O(1)

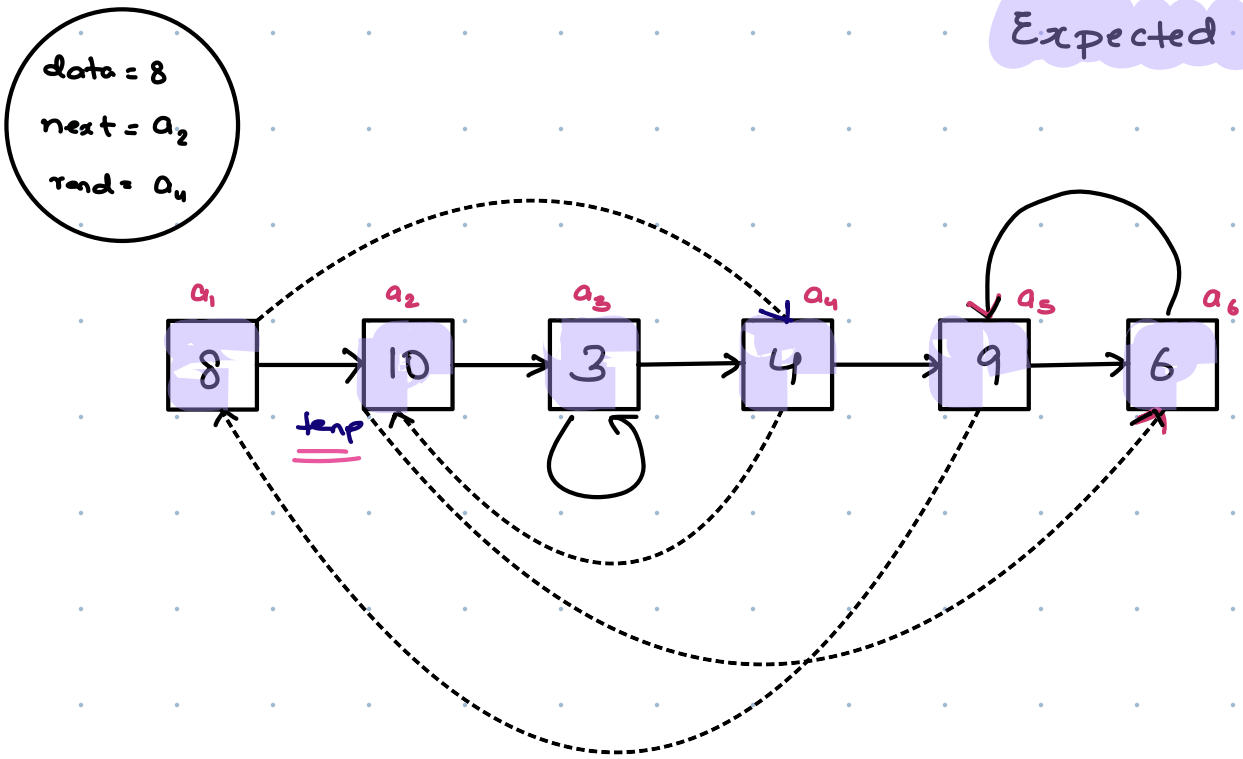
Clone Linkedlist

```
class Node {
    int data;
    Node next;  // pointing to next node
    Node rand;  // pointing to any node in LL
}
```

Note :- Rand is not null.

Given a LinkedList, create & return clone of it.

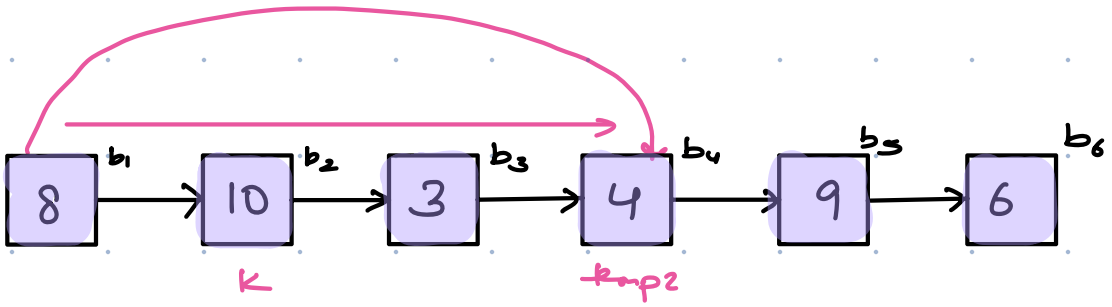
Expected SC : O(1)



Brute force →

01. Create a clone of LL just by using next ptr
02. For a node, go & get random ptr.
03. Iterate on cloned LL & search for random ptr value.

04. Create random ptr connection

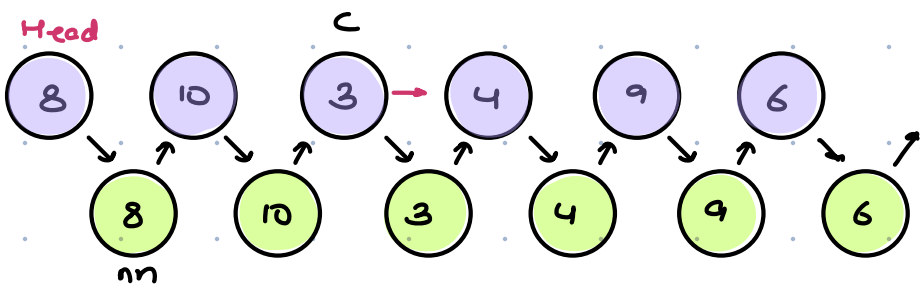


TC : O(N²)

SC : O(1)

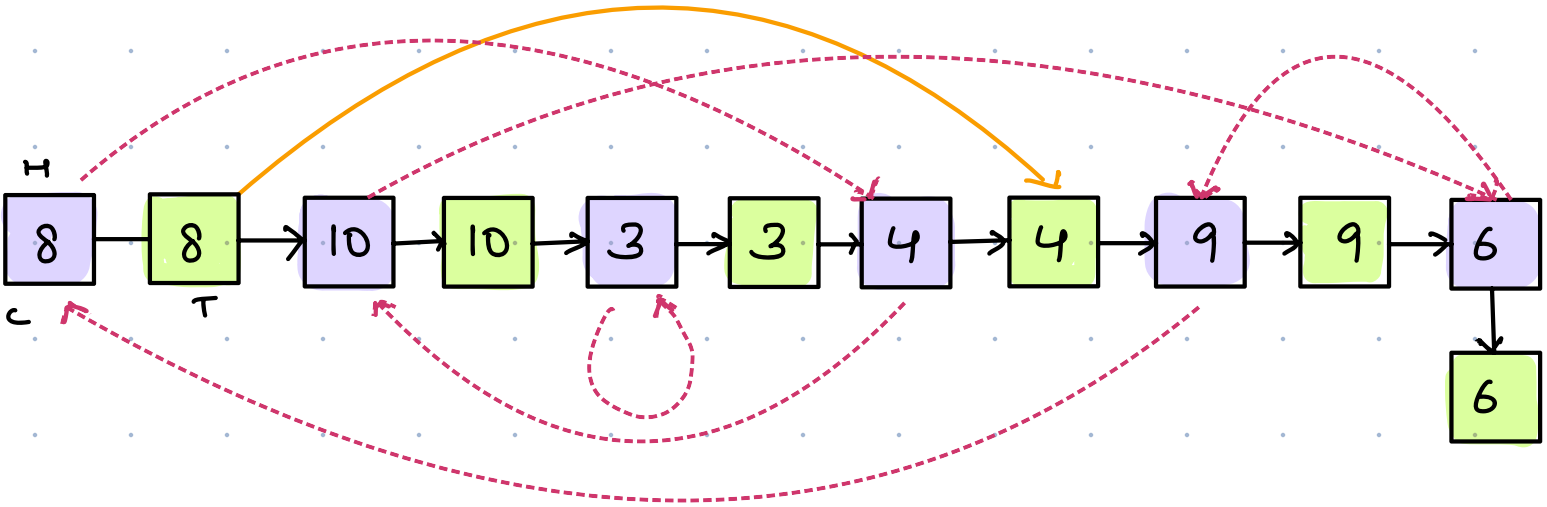
* **Optimal Idea**

01. Interleave all the new nodes in between old nodes



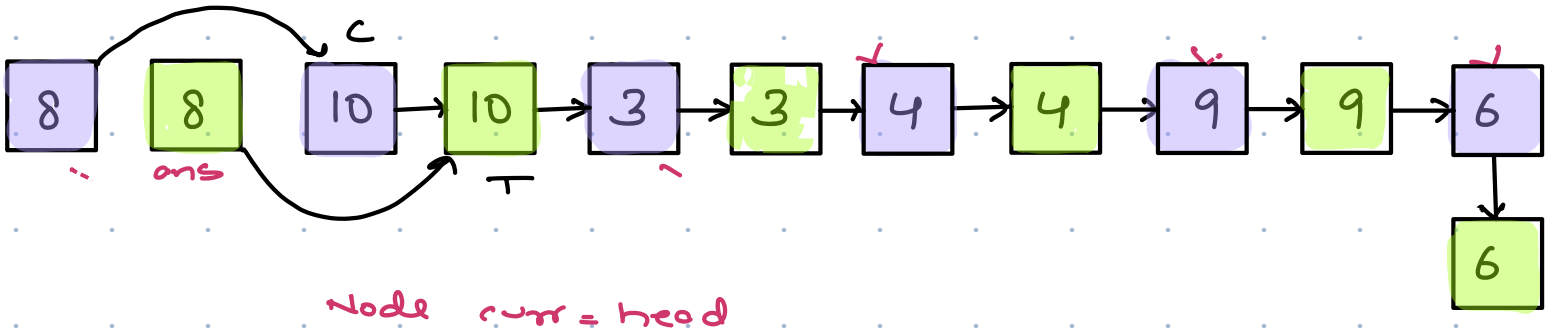

```
Node curr = head
while (curr != Null)
{
    Node nn = new Node (curr.data);
    nn.next = curr.next;
    curr.next = nn;
    curr = nn.next;
}
```

02. Populate random pointer in green LL




```
Node  curr = head

Node  temp = head.next;

while (curr != Null)
    temp.random = curr.random.next;

    curr = curr.next.next

    if (temp.next != Null)  temp = temp.next.next
```

Q3 Unlink green LL from purple LL




```
Node curr = head
Node temp = head.next ;
Node ans = temp
while (curr != Null)
    curr.next = temp.next
    if (temp.next != null) temp.next = temp.next.next
    curr = curr.next
    temp = temp.next ;
return ans ;
```

Tc : O(N)
Sc : O(1)