***Delete Head of Circular Linked List***

**Deleting head node from Singly Circular Linked List**

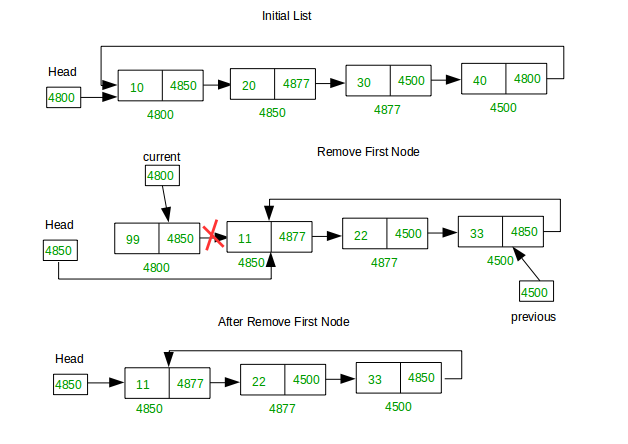
**Examples**:

**Input** : 99->11->22->33->44->55->66

**Output** : 11->22->33->44->55->66

**Input** : 11->22->33->44->55->66

**Output** : 22->33->44->55->66



*Deleting First Node from Circular Linked List*

**Naive Approach:**

C++Java

import java.util.\*;

import java.io.\*;

import java.lang.\*;

class Node{

int data;

Node next;

Node(int d){

data=d;

next=null;

}

}

class Test {

public static void main(String args[])

{

Node head=new Node(10);

head.next=new Node(20);

head.next.next=new Node(30);

head.next.next.next=new Node(40);

head.next.next.next.next=head;

head=delHead(head);

printlist(head);

}

public static void printlist(Node head){

if(head==null)return;

Node r=head;

do{

System.out.print(r.data+" ");

r=r.next;

}while(r!=head);

}

static Node delHead(Node head){

if(head==null)return null;

if(head.next==head)

return null;

Node curr=head;

while(curr.next!=head)

curr=curr.next;

curr.next=head.next;

return (curr.next);

}

}

**Output:**

20 30 40

**Efficient Method:**

C++Java

import java.util.\*;

import java.io.\*;

import java.lang.\*;

class Node{

int data;

Node next;

Node(int d){

data=d;

next=null;

}

}

class Test {

public static void main(String args[])

{

Node head=new Node(10);

head.next=new Node(20);

head.next.next=new Node(30);

head.next.next.next=new Node(40);

head.next.next.next.next=head;

head=delHead(head);

printlist(head);

}

public static void printlist(Node head){

if(head==null)return;

Node r=head;

do{

System.out.print(r.data+" ");

r=r.next;

}while(r!=head);

}

static Node delHead(Node head){

if(head==null)return null;

if(head.next==head)

return null;

head.data=head.next.data;

head.next=head.next.next;

return head;

}

}

**Output:**

20 30 40