**Reverse a Doubly Linked List**

Given a doubly linked list of n elements. The task is to **reverse**the doubly linked list.

**Example 1:**

**Input:**

LinkedList: 3 <--> 4 <--> 5

**Output:** 5 4 3

**Example 2:**

**Input:**

LinkedList: 75 <--> 122 <--> 59 <--> 196

**Output:** 196 59 122 75

**Expected Time Complexity:**O(n).  
**Expected Auxiliary Space:**O(1).

**Constraints:**  
1 <= number of nodes <= 103  
0 <= value of nodes <= 103

**Company Tags**

[**D-E-Shaw**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=D-E-Shaw) [**Adobe**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Adobe)

//{ Driver Code Starts

//Initial Template for Java

import java.util.\*;

import java.lang.\*;

import java.io.\*;

class Node

{

int data;

Node next, prev;

Node(int data)

{

this.data = data;

this.next = null;

this.prev = null;

}

}

class CodingMaxima

{

public static void main(String args[])throws IOException

{

BufferedReader read = new BufferedReader(new InputStreamReader(System.in));

int t = Integer.parseInt(read.readLine());

while(t-- > 0)

{

int n = Integer.parseInt(read.readLine());

String str[] = read.readLine().trim().split(" ");

Node head = null, tail = null;

for(int i = 0; i < n; i++)

{

int data = Integer.parseInt(str[i]);

if (head == null)

{

head = new Node(data);

tail = head;

}

else

{

tail.next = new Node(data);

tail.next.prev=tail;

tail = tail.next;

}

}

head=reverseDLL(head);

if(verify(head))

displayList(head);

else

System.out.print("Your pointers are not correctly connected");

System.out.println();

}

}

public static void displayList(Node head)

{

//Head to Tail

while(head.next != null)

{

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

head = head.next;

}

System.out.print(head.data);

}

// } Driver Code Ends

public static Node reverseDLL(Node head)

{

Node ptr1=null;

Node ptr2=head;

while(ptr2!=null){

ptr2.prev=ptr2.next;

ptr2.next=ptr1;

ptr1=ptr2;

ptr2=ptr2.prev;

}

head=ptr1;

return head;

}

//{ Driver Code Starts.

public static boolean verify(Node head)

{

int fl=0;

int bl=0;

Node temp=head;

while(temp.next!=null)

{

temp=temp.next;

fl++;

}

while(temp.prev!=null)

{

temp=temp.prev;

bl++;

}

return fl==bl;

}

}

// } Driver Code Ends