**Reverse a doubly linked list**

his challenge is part of a tutorial track by[MyCodeSchool](http://www.youtube.com/mycodeschool)

You’re given the pointer to the head node of a doubly linked list. Reverse the order of the nodes in the list. The head node might be NULL to indicate that the list is empty. Change the next and prev pointers of all the nodes so that the direction of the list is reversed. Return a reference to the head node of the reversed list.

**Function Description**

Complete the reverse function in the editor below. It should return a reference to the head of your reversed list.

reverse has the following parameter(s):

* head: a reference to the head of a DoublyLinkedList

**Input Format**

The first line contains an integer t, the number of test cases.

Each test case is of the following format:

* The first line contains an integer n, the number of elements in the linked list.
* The next n lines contain an integer each denoting an element of the linked list.

**Constraints**

* 1<= t< =10
* 1<= n <=1000
* DoublyLinkedListNode.data<=1000

**Output Format**

Return a reference to the head of your reversed list. The provided code will print the reverse array as a one line of space-separated integers for each test case.

**Sample Input**

1

4

1

2

3

4

**Sample Output**

4 3 2 1

**Explanation**

The initial doubly linked list is: 1🡨>2🡨>3🡨>4->NULL

The reversed doubly linked list is:4<🡪3🡨>2🡨>1->NULL

using System.CodeDom.Compiler;

using System.Collections.Generic;

using System.Collections;

using System.ComponentModel;

using System.Diagnostics.CodeAnalysis;

using System.Globalization;

using System.IO;

using System.Linq;

using System.Reflection;

using System.Runtime.Serialization;

using System.Text.RegularExpressions;

using System.Text;

using System;

class Solution {

class DoublyLinkedListNode {

public int data;

public DoublyLinkedListNode next;

public DoublyLinkedListNode prev;

public DoublyLinkedListNode(int nodeData) {

this.data = nodeData;

this.next = null;

this.prev = null;

}

}

class DoublyLinkedList {

public DoublyLinkedListNode head;

public DoublyLinkedListNode tail;

public DoublyLinkedList() {

this.head = null;

this.tail = null;

}

public void InsertNode(int nodeData) {

DoublyLinkedListNode node = new DoublyLinkedListNode(nodeData);

if (this.head == null) {

this.head = node;

} else {

this.tail.next = node;

node.prev = this.tail;

}

this.tail = node;

}

}

static void PrintDoublyLinkedList(DoublyLinkedListNode node, string sep, TextWriter textWriter) {

while (node != null) {

textWriter.Write(node.data);

node = node.next;

if (node != null) {

textWriter.Write(sep);

}

}

}

// Complete the reverse function below.

/\*

\* For your reference:

\*

\* DoublyLinkedListNode {

\* int data;

\* DoublyLinkedListNode next;

\* DoublyLinkedListNode prev;

\* }

\*

\*/

static DoublyLinkedListNode reverse(DoublyLinkedListNode head) {

while(head!=null)

{

DoublyLinkedListNode right=head.next;

DoublyLinkedListNode left=head.prev;

head.next=left;

head.prev=right;

if(head.prev==null)return head;

head=head.prev;

}

return head;

}

static void Main(string[] args) {

TextWriter textWriter = new StreamWriter(@System.Environment.GetEnvironmentVariable("OUTPUT\_PATH"), true);

int t = Convert.ToInt32(Console.ReadLine());

for (int tItr = 0; tItr < t; tItr++) {

DoublyLinkedList llist = new DoublyLinkedList();

int llistCount = Convert.ToInt32(Console.ReadLine());

for (int i = 0; i < llistCount; i++) {

int llistItem = Convert.ToInt32(Console.ReadLine());

llist.InsertNode(llistItem);

}

DoublyLinkedListNode llist1 = reverse(llist.head);

PrintDoublyLinkedList(llist1, " ", textWriter);

textWriter.WriteLine();

}

textWriter.Flush();

textWriter.Close();

}

}

You have earned 5.00 points!

These points will also count towards your progress in the Problem Solving Badge.

**100%**

**Congratulations**

You solved this challenge. Would you like to challenge your friends?

[Next Challenge](https://www.hackerrank.com/challenges/find-the-merge-point-of-two-joined-linked-lists?h_l=interview&playlist_slugs%5B%5D=interview-preparation-kit&playlist_slugs%5B%5D=linked-lists&h_r=next-challenge&h_v=zen)

* **Test case 0**
* **Test case 1**
* **Test case 2**
* **Test case 3**
* **Test case 4**
* **Test case 5**
* **Test case 6**
* **Test case 7**

Compiler Message

**Success**

Input (stdin)

Download

* **1**
* **4**
* **1**
* **2**
* **3**
* **4**