**Doubly linked list Insertion at given position**

Given a doubly-linked list, a position **p,** and an integer **x.** The task is to add a new node with value **x** at the position just after **pth** node in the doubly linked list.

**Example 1:**

**Input:**

LinkedList: 2<->4<->5

p = 2, x = 6

**Output:** 2 4 5 6

**Explanation:** p = 2, and x = 6. So, 6 is

inserted after p, i.e, at position 3

(0-based indexing).

**Example 2:**

**Input:**

LinkedList: 1<->2<->3<->4

p = 0, x = 44

**Output:** 1 44 2 3 4

**Explanation:** p = 0, and x = 44 . So, 44

is inserted after p, i.e, at position 1

(0-based indexing).

**Expected Time Complexity** : O(N)  
**Expected Auxilliary Space** : O(1)

**Constraints:**  
1 <= N <= 104  
0 <= p < N

//{ Driver Code Starts

import java.util.\*;

class Node

{

int data;

Node next;

Node prev;

Node(int data)

{

this.data = data;

next = prev = null;

}

}

class DLinkedList

{

Node newNode(Node head, int data)

{

Node n = new Node(data);

if(head == null)

{

head = n;

return head;

}

head.next = n;

n.prev = head;

head = n;

return head;

}

void printList(Node node)

{

Node temp = node;

while(temp.next != null)

{

temp = temp.next;

}

while(temp.prev != null)

{

temp = temp.prev;

}

while(temp != null)

{

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

temp = temp.next;

}

System.out.println();

}

public static void main(String args[])

{

Scanner sc = new Scanner(System.in);

DLinkedList DLL = new DLinkedList();

int t = sc.nextInt();

while(t>0)

{

Node temp;

Node head = null;

Node root = null;

int n = sc.nextInt();

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

{

int x = sc.nextInt();

head = DLL.newNode(head,x);

if(root == null) root = head;

}

head = root;

int pos = sc.nextInt();

int data = sc.nextInt();

CodingMaxima g = new CodingMaxima ();

g.addNode(head,pos,data);

DLL.printList(head);

while(head.next != null)

{

temp = head;

head = head.next;

}

t--;

}

}

}

// } Driver Code Ends

class CodingMaxima

{

//Function to insert a new node at given position in doubly linked list.

void addNode(Node head, int pos, int data)

{

Node newNode=new Node(data);

if(head==null)

return ;

Node curr=head;

while(curr!=null && pos>0){

curr=curr.next;

pos--;

}

newNode.prev=curr;

newNode.next=curr.next;

if(curr.next!=null){

curr.next.prev=newNode;

}

curr.next=newNode;

}

}