### Check if Linked List is Palindrome

Given a singly linked list of size **N** of integers. The task is to check if the given linked list is palindrome or not.

**Example 1:**

**Input:**

N = 3

value[] = {1,2,1}

**Output:** 1

**Explanation:** The given linked list is

1 2 1 , which is a palindrome and

Hence, the output is 1.

**Example 2:**

**Input:**

N = 4

value[] = {1,2,3,4}

**Output:** 0

**Explanation:** The given linked list

is 1 2 3 4 , which is not a palindrome

and Hence, the output is 0.

Java code

//{ Driver Code Starts

import java.util.\*;

class Node

{

int data;

Node next;

Node(int d)

{

data = d;

next = null;

}

}

class Is\_LinkedList\_Palindrom

{

Node head;

Node tail;

/\* Function to print linked list \*/

void printList(Node head)

{

Node temp = head;

while (temp != null)

{

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

temp = temp.next;

}

System.out.println();

}

/\* Inserts a new Node at front of the list. \*/

public void addToTheLast(Node node)

{

if (head == null)

{

head = node;

tail = node;

} else

{

tail.next = node;

tail = node;

}

}

public static void main(String args[])

{

Scanner sc = new Scanner(System.in);

int t = sc.nextInt();

while(t>0)

{

int n = sc.nextInt();

//int k = sc.nextInt();

Is\_LinkedList\_Palindrom llist = new Is\_LinkedList\_Palindrom();

//int n=Integer.parseInt(br.readLine());

int a1=sc.nextInt();

Node head= new Node(a1);

Node tail = head;

for (int i = 1; i < n; i++)

{

int a = sc.nextInt();

tail.next = new Node(a);

tail = tail.next;

}

Solution g = new Solution();

if(g.isPalindrome(head) == true)

System.out.println(1);

else

System.out.println(0);

t--;

}

}

}

// } Driver Code Ends

class Solution

{

//Function to check whether the list is palindrome.

boolean isPalindrome(Node head)

{

//Your code here

Node current=head;

int n=0;

while(current!=null){

n=n\*10+current.data;

current=current.next;

}

int temp=n;

int rev=0;

while(n!=0){

rev=rev\*10+n%10;

n=n/10;

}

return rev==temp;

}

}