**PALINDROME**

Check if a linked list is a palindrome.( Boolean return type )

**Input Format:**

Add N space separated elements in the list

**Constraints:**

None

**Output Format**

Boolean answer( true or false)

**Sample Input**

5

1 2 3 6 8

**Sample Output**

False

#include<iostream>

using namespace std;

class Node{

public:

int data;

Node \*next;

Node(int d)

{

data=d;

next=NULL;

}

};

void insertAtEnd(Node \*&head,int num)

{

if(head==NULL)

{

head=new Node(num);

return;

}

Node \*tail=head;

while(tail->next!=NULL)

tail=tail->next;

Node \*n=new Node(num);

tail->next=n;

return;

}

void reverse(Node \*&head)

{

Node \*c=head;

Node \*p=NULL;

Node \*n;

while(c!=NULL)

{

n=c->next;

c->next=p;

p=c;

c=n;

}

head=p;

}

void print(Node \*head)

{

Node \*temp=head;

while(temp!=NULL)

{

cout<<temp->data<<" ";

temp=temp->next;

}

}

bool palindrome(Node \*head,int n)

{

if(n%2==0)

{

Node \*temp=head;

int k=0;

while(k!=n/2)

{

k++;

temp=temp->next;

}

//print(temp);

reverse(temp);

while(temp!=NULL)

{

//cout<<head->data<<" "<<temp->data<<" ";

if(head->data!=temp->data)

return false;

head=head->next;

temp=temp->next;

}

return true;

}

if(n%2!=0)

{

Node \*temp=head;

int k=0;

while(k!=n/2)

{

k++;

temp=temp->next;

}

temp=temp->next;

//print(temp);

reverse(temp);

while(temp!=NULL)

{

//cout<<head->data<<" "<<temp->data<<" ";

if(head->data!=temp->data)

return false;

head=head->next;

temp=temp->next;

}

return true;

}

}

int main()

{

Node \*head=NULL;

int n,i,num;

cin>>n;

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

{

cin>>num;

insertAtEnd(head,num);

}

if(palindrome(head,n))

cout<<"True";

else

cout<<"False";

}