/\* Program to Reverse elements in Single Linked List\*/

#include<iostream>

#include<stdlib.h>

using namespace std;

struct node

{

int data;

struct node \*link;

};

struct node \*root=NULL;

void append() // Function to insert nodes

{

struct node \*temp;

temp=(struct node \*)malloc(sizeof(struct node));

cout<<"Enter Node Data"<<endl;

cin>>temp->data;

temp->link=NULL;

if(root==NULL)

root=temp;

else

{

struct node \*p;

p=root;

while(p->link!=NULL)

{

p=p->link;

}

p->link=temp;

}

}

int length() // Function to find length of list

{

struct node \*temp;

int count=0;

temp=root;

if(temp==NULL)

cout<<"List is Empty"<<endl;

else

{

while(temp!=NULL)

{

count++;

temp=temp->link;

}

}

return count;

}

void display() // Function to display list

{

struct node \*temp;

temp=root;

if(temp==NULL)

cout<<"List is Empty"<<endl;

else

{

while(temp!=NULL)

{

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

temp=temp->link;

}

cout<<"NULL"<<endl;

}

}

void reverselist() //Function to reverse list

{

int i,j,temp,len;

len=length();

i=0; j=len-1;

struct node \*p, \*q; // Using two pointers

p=q=root;

while(i<j) //reversal test condition

{

int k=0;

while(k<j)

{

q=q->link; //Points q to the last node

k++;

}

temp=p->data; //swapping logic

p->data=q->data;

q->data=temp;

i++; j--; //increment i and decerement j

p=p->link; //move pointer p to next node

q=root; //move pointer q to root

}

}

int main()

{

int n;

cout<<"Enter Number of Nodes"<<endl;

cin>>n;

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

append();

cout<<"Displaying nodes before reversal"<<endl;

display();

cout<<"Reversing list....."<<endl<<endl<<endl;

reverselist();

cout<<"Displaying list after reversal"<<endl<<endl;

display();

exit(0);

}