#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <assert.h>

struct node {

int data;

struct node \*next;

};

typedef struct node single\_LL;

single\_LL\* start = NULL;

// Find the Length of linked list iterative

int iter\_llist\_length()

{

single\_LL \*temp = start;

int len = 0;

if(start == NULL) { // if list is empty

return 0;

}

else {

while(temp != NULL) {

len++;

temp = temp->next;

}

}

return len;

}

int recur\_llist\_length(single\_LL \*nd) {

static int len = 0;

if(nd == NULL) {

return 0;

}

else {

len = 1 + recur\_llist\_length(nd->next);

}

return len;

}

// Find length of linked list Recursive

/\* Create a single Link List \*/

void create(int no){

single\_LL \*ptr, \*net;

int val, i;

for(i = 1; i <= no; i++) {

scanf("%d",&val);

ptr = (single\_LL\* )malloc(sizeof(single\_LL));

assert(ptr != NULL);

ptr->data = val;

ptr->next = NULL;

if(start == NULL) {

start = ptr;

net = ptr;

}

else {

net->next = ptr;

net =net->next;

}

}

}

void display()

{

single\_LL\* nd;

nd = start;

while(nd != NULL) {

printf("%d->",nd->data);

nd = nd->next;

}

printf("NULL");

}

int main(void) {

// your code goes here

int nd;

single\_LL \*hd;

scanf("%d",&nd);

create(nd);

printf("Display Created Node \n");

display();

// printf("\n Length of Iterative Single List %d", iter\_llist\_length());

printf("\n Length of Recursive Single list %d", recur\_llist\_length(start) );

return 0;

}