#include<stdio.h>

#include<stdlib.h>

void printNode();

void addFirst(int data);

void addLast(int data);

void addPos(int pos,int data);

void delFirst();

void delLast();

void delPos(int pos);

int searchData(int search);

int searchLastOcur(int search);

int search2ndLastOcur(int search);

int searchDataCount(int search);

void link\_list\_at\_end();

int count();

struct Node{

int data;

struct Node\*next;

};

struct Node\*head=NULL;

void main(){

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

node->data=30;

node->next=NULL;

head=node;

addFirst(30);

addLast(20);

addPos(3,30);

searchDataCount(30);

printNode();

printf("%d\n",count());

}

void printNode(){

struct Node\*temp=head;

while(temp!=NULL){

if(temp->next!=NULL){

printf("| %d |->",temp->data);

}else{

printf("| %d |",temp->data);

}

temp=temp->next;

}

printf("\n");

}

void addFirst(int data){

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

node->data=data;

node->next=head;

head=node;

}

void addLast(int data){

struct Node\*temp=head;

while(temp->next!=NULL){

temp=temp->next;

}

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

node->data=data;

temp->next=node;

node->next=NULL;

}

void addPos(int pos,int data){

pos=pos-2;

struct Node\*temp=head;

while(pos){

temp=temp->next;

pos--;

}

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

node->data=data;

node->next=temp->next;

temp->next=node;

}

void delFirst(){

struct Node\*temp=head;

head=head->next;

free(temp);

}

void delLast(){

struct Node\*temp=head;

while(temp->next->next!=NULL){

temp=temp->next;

}

free(temp->next);

temp->next=NULL;

}

void delPos(int pos){

pos=pos-2;

struct Node\*temp=head;

while(pos){

temp=temp->next;

pos--;

}

struct Node\*temp1=temp->next;

temp->next=temp->next->next;

free(temp1);

}

int count(){

int cnt=0;

struct Node\*temp=head;

while(temp!=NULL){

temp=temp->next;

cnt++;

}

return cnt;

}

int searchData(int search){

struct Node\*temp=head;

int index=0;

int flag=1;

while(temp!=NULL){

index++;

if(temp->data==search){

printf("data is present at inexd: %d\n",index);

flag--;

break;

}

temp=temp->next;

}

if(flag){

printf("Data is not present in linkedlist\n");

}

}

int searchLastOcur(int search){

struct Node\*temp=head;

int index=0;

int flag=0;

int lo;

while(temp!=NULL){

index++;

if(temp->data==search){

lo=index;

flag++;

}

temp=temp->next;

}

if(flag!=0){

printf("data last occurrence at inexd: %d\n",lo);

}else{

printf("Data is not present in linkedlist");

}

}

int search2ndLastOcur(int search){

struct Node\*temp=head;

int index=0;

int flag=0;

int oldOcur=0,newOcur=0;

while(temp!=NULL){

index++;

if(temp->data==search){

oldOcur=newOcur;

newOcur=index;

flag++;

}

temp=temp->next;

}

if(flag!=0){

printf("data 2nd last ocurrence at inexd: %d\n",oldOcur);

}else{

printf("Data is not present in linkedlist");

}

}

int searchDataCount(int search){

struct Node\*temp=head;

int cnt=0;

while(temp!=NULL){

if(temp->data==search){

cnt++;

}

temp=temp->next;

}

if(cnt!=0){

printf("%d is present in linked list %d times\n",search,cnt);

}else{

printf("Data is not present in linkedlist\n");

}

}

ganeshu@ganeshu-HP-Laptop-15s-gr0xxx:~/ds$ cat singly-liner2.c

#include<stdio.h>

#include<stdlib.h>

int count();

int count2();

void printNode();

void printNode2();

void addLast();

void addLast2();

void listConcat();

void ConcatFirstN();

void ConcatLastN();

void ConcatListRange();

void LLCopy();

void LLNCopy();

void LLLNCopy();

void LLCopyRange();

void LLCopyAlt();

struct Node{

int data;

struct Node\*next;

};

struct Node\*head=NULL;

struct Node\*head2=NULL;

void main(){

int n1,n2;

head=malloc(sizeof(struct Node));

head->data=10;

head->next=NULL;

printf("number of nodes in list 1 : ");

scanf("%d",&n1);

for(int i=1;i<n1;i++){

if(i==1){

printf("data in 1 list : ");

scanf("%d",&head->data);

}

addLast();

}

head2=malloc(sizeof(struct Node));

head2->data=0;

head2->next=NULL;

char ch;

printf("Fill 2nd list : \n----for yes enter ' y 'and for 'n'----\n");

scanf(" %c",&ch);

if(ch=='y'||ch=='Y'){

printf("number of nodes in list 2 : ");

scanf("%d",&n2);

for(int i=1;i<n2;i++){

if(i==1){

printf("data in 1 list : ");

scanf("%d",&head2->data);

}

addLast2();

}

}

LLCopyRange();

printNode2();

}

int count(){

struct Node\*temp=head;

int cnt=0;

while(temp!=NULL){

temp=temp->next;

cnt++;

}

return cnt;

}

int count2(){

struct Node\*temp=head2;

int cnt=0;

while(temp!=NULL){

temp=temp->next;

cnt++;

}

return cnt;

}

void printNode(){

struct Node\*temp=head;

while(temp!=NULL){

if(temp->next==NULL){

printf("| %d |",temp->data);

}else{

printf("| %d |->",temp->data);

}

temp=temp->next;

}

printf("\n");

}

void printNode2(){

struct Node\*temp=head2;

while(temp!=NULL){

if(temp->next==NULL){

printf("| %d |",temp->data);

}else{

printf("| %d |->",temp->data);

}

temp=temp->next;

}

printf("\n");

}

void addLast(){

struct Node\*temp=head;

while(temp->next!=NULL){

temp=temp->next;

}

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

printf("data in 1 list : ");

scanf("%d",&node->data);

node->next=NULL;

temp->next=node;

}

void addLast2(){

struct Node\*temp=head2;

while(temp->next!=NULL){

temp=temp->next;

}

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

printf("data in 2 list : ");

scanf("%d",&node->data);

node->next=NULL;

temp->next=node;

}

void listConcat(){

struct Node\*temp=head;

while(temp->next!=NULL){

temp=temp->next;

}

temp->next=head2;

printNode();

}

void ConcatFirstN(){

int n;

printf("input number of elements : ");

scanf("%d",&n);

struct Node\*temp2=head2;

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

struct Node\*temp=head;

while(temp->next!=NULL){

temp=temp->next;

}

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

node->data=temp2->data;

node->next=NULL;

temp->next=node;

temp2=temp2->next;

}

printNode();

}

void ConcatLastN(){

int n;

printf("Enter last n element to copy : ");

scanf("%d",&n);

int pos=count2()-n;

struct Node\*temp2=head2;

while(pos){

temp2=temp2->next;

pos--;

}

struct Node\*temp=head;

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

struct Node\*temp=head;

while(temp->next!=NULL){

temp=temp->next;

}

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

node->data=temp2->data;

node->next=NULL;

temp->next=node;

temp2=temp2->next;

}

printNode();

}

void ConcatListRange(){

struct Node\*temp1=head;

printf("Enter starting range:");

int sr,er;

scanf("%d",&sr);

printf("Enter ending range :");

scanf("%d",&er);

int pos=sr-1;

while(pos){

temp1=temp1->next;

pos--;

}

int lim=er-sr;

for(int i=1;i<=lim+1;i++){

struct Node\*temp2=head2;

while(temp2->next!=NULL){

temp2=temp2->next;

}

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

node->data=temp1->data;

temp2->next=node;

temp1=temp1->next;

}

printNode2();

}

void LLCopy(){

struct Node\*temp1=head;

struct Node\*temp2=head2;

head2->data=head->data;

temp1=temp1->next;

for(int i=1;i<count();i++){

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

node->data=temp1->data;

temp2->next=node;

temp2=temp2->next;

temp1=temp1->next;

}

printNode2();

}

void LLNCopy(){

struct Node\*temp1=head;

struct Node\*temp2=head2;

head2->data=head->data;

temp1=temp1->next;

int n;

printf("Number of elements copy :");

scanf("%d",&n);

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

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

node->data=temp1->data;

temp2->next=node;

temp2=temp2->next;

temp1=temp1->next;

}

printNode2();

}

void LLLNCopy(){

printf("Last N number copy list : ");

int n;

scanf("%d",&n);

int pos=count()-n;

struct Node\*temp1=head;

while(pos){

temp1=temp1->next;

pos--;

}

struct Node\*temp2=head2;

head2->data=temp1->data;

temp1=temp1->next;

while(temp1!=NULL){

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

node->data=temp1->data;

temp2->next=node;

temp2=temp2->next;

temp1=temp1->next;

}

printNode2();

}

void LLCopyRange(){

int ul,ll;

printf("input starting range : ");

scanf("%d",&ul);

printf("input ending range : ");

scanf("%d",&ll);

int pos=ul-1;

struct Node\*temp1=head;

struct Node\*temp2=head2;

while(pos){

temp1=temp1->next;

pos--;

}

temp2->data=temp1->data;

temp1=temp1->next;

for(int i=ul;i<ll;i++){

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

node->data=temp1->data;

temp2->next=node;

temp2=temp2->next;

temp1=temp1->next;

}

}

void LLCopyAlt(){

}