Name: Jingjing Tan

StudentID:19548

Lab1

Question1:

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* File Name: LinkedList-Lab1-q2

\* Author: JingjingTan

\* Engineering School, NPU

\* description: Insert element to single linked list by

\* reverse index, such as Head->11->12->13->14->NULL

\* If index=1 and element=345, then new linked

\* list will be Head->11->12->13->345->14->NULL

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <stdio.h>

#include <stdlib.h>

typedef int ElemType;

typedef struct Lnode

{

ElemType data;

struct Lnode \*next;

}LNode;

LNode \*create\_LinkList(void){

int data;

LNode \*head, \*p, \*q;

head=p=(LNode \*)malloc(sizeof(LNode));

head->next=NULL;

do{

scanf("%d",& data);

q= (LNode \*)malloc(sizeof(LNode));

q->data=data;

q->next=p->next;

p->next=q;

p=q ;

}while(getchar()!='\n');

return (head);

}

int getLength(LNode \*L)

{

int size=0;

Lnode \*p;

p=L->next;

while (cur!= NULL)

{

++size;

cur = cur->next;

}

return size;

}

void Insert\_LNode(LNode \*L, int i, ElemType e) {

int j=0;

LNode \*p,\*q;

p=L->next ;

while(p!=NULL && j<i-1) {

p=p->next;

j++;

}

if(p==NULL || j!=i-1)

printf("i is too big OR i is 0!!\n");

else{

q=(LNode\*)malloc(sizeof(LNode));

q->data=e;

q->next=p->next;

p->next=q;

}

}

void printList(LNode\* list){

printf("Head");

while(1){

printf("->");

list = list->next;

if(list->next == NULL) {

printf("%d->NULL\n", list->data);

break;

}

else

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

}

}

int main(void) {

LNode \*a;

a = create\_LinkList();

printList(a);

int elem=123, i=2;

Insert\_LNode(a, i, elem);

printList(a);

return 0;

}

Question 2:

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* File **Na**me: LinkedList-Lab1-q2

\* Author: JingjingTan

\* Engineering School, NPU

\* description: concatenate two singly linked list, like a =

\* head->4->3->1->NULL,

\* , and b= Head->11->12->13->NULL

\* , then new linked list will be

\* Head->4->3->1->11->12->13->NULL

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <stdio.h>

#include <stdlib.h>

typedef int ElemType;

typedef struct Lnode

{

ElemType data;

struct Lnode \*next;

}LNode;

LNode \*create\_LinkList(void){

int data;

LNode \*head, \*p, \*q;

head=p=(LNode \*)malloc(sizeof(LNode));

head->next=NULL;

do{

scanf("%d",& data);

q= (LNode \*)malloc(sizeof(LNode));

q->data=data;

q->next=p->next;

p->next=q;

p=q ;

}while(getchar()!='\n');

return (head);

}

void concat\_linked\_list(LNode \*a,LNode \*b)

{

if( a != NULL && b!= NULL )

{

if (a->next == NULL)

a->next = b->next;

else

concat\_linked\_list(a->next,b);

}

else

{

printf("Either a or b is NULL\n");

}

}

void printList(LNode\* list){

printf("Head");

while(1){

printf("->");

list = list->next;

if(list == NULL){

printf("NULL");

exit(0);

}

else if(list->next == NULL) {

printf("%d->NULL\n", list->data);

break;

}

else

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

}

}

int main(void) {

LNode \*a;

a = create\_LinkList();

printList(a);

LNode \*b;

b = create\_LinkList();

printList(b);

concat\_linked\_list(a, b);

printf("The concatenated linked list is: \n");

printList(a);

return 0;

}

Question3:

(a)

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* File Name: LinkedList-Lab1-q3(a)

\* Author: JingjingTan

\* Engineering School, NPU

\* description: find the maxium value for a given singly

\* linked list, like a = head->8->2->10->1->NULL,

\* then result will be 10.

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <stdio.h>

#include <stdlib.h>

typedef int ElemType;

typedef struct Lnode

{

ElemType data;

struct Lnode \*next;

}LNode;

LNode \*create\_LinkList(void){

int data;

LNode \*head, \*p, \*q;

head=p=(LNode \*)malloc(sizeof(LNode));

head->next=NULL;

do{

scanf("%d",& data);

q= (LNode \*)malloc(sizeof(LNode));

q->data=data;

q->next=p->next;

p->next=q;

p=q ;

}while(getchar()!='\n');

return (head);

}

int max\_int\_linked\_list(LNode \*a)

{

if(a == NULL)

printf("Invalid input!");

int max = a->data;

while(a!=NULL)

{

if(a->data > max){

max = a->data;

}

a=a->next;

}

return max;

}

void printList(LNode\* list){

printf("Head");

while(1){

printf("->");

list = list->next;

if(list == NULL){

printf("NULL");

exit(0);

}

else if(list->next == NULL) {

printf("%d->NULL\n", list->data);

break;

}

else

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

}

}

int main(void) {

LNode \*a;

a = create\_LinkList();

printList(a);

int res=max\_int\_linked\_list(a);

printf("The maxium value of the linked list is: %d",res);

return 0;

}

(b)

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* File **Na**me: LinkedList-Lab1-q3(b)

\* Author: JingjingTan

\* Engineering School, NPU

\* description: find the minium value for a given singly

\* linked list, like a = head->8->2->10->1->NULL,

\* then result will be 10.

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <stdio.h>

#include <stdlib.h>

typedef int ElemType;

typedef struct Lnode

{

ElemType data;

struct Lnode \*next;

}LNode;

LNode \*create\_LinkList(void){

int data;

LNode \*head, \*p, \*q;

head=p=(LNode \*)malloc(sizeof(LNode));

head->next=NULL;

do{

scanf("%d",& data);

q= (LNode \*)malloc(sizeof(LNode));

q->data=data;

q->next=p->next;

p->next=q;

p=q ;

}while(getchar()!='\n');

return (head);

}

int min\_int\_linked\_list(LNode \*a)

{

a=a->next;

if(a == NULL)

printf("Invalid input!");

int min = a->data;

while(a!=NULL)

{

if(a->data < min){

min = a->data;

}

a=a->next;

}

return min;

}

void printList(LNode\* list){

printf("Head");

while(1){

printf("->");

list = list->next;

if(list == NULL){

printf("NULL");

exit(0);

}

else if(list->next == NULL) {

printf("%d->NULL\n", list->data);

break;

}

else

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

}

}

int main(void) {

LNode \*a;

a = create\_LinkList();

printList(a);

int res=min\_int\_linked\_list(a);

printf("The minium value of the linked list is: %d",res);

return 0;

}

(c)

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* File **Na**me: LinkedList-Lab1-q3(c)

\* Author: JingjingTan

\* Engineering School, NPU

\* description: find the maxium value for a given singly

\* linked list of char, like L = head->c->d->a->f->NULL,

\* then result will be f.

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <stdio.h>

#include <stdlib.h>

typedef char ElemType;

typedef struct Lnode

{

ElemType data;

struct Lnode \*next;

}LNode;

LNode \*create\_LinkList(void){

char data;

LNode \*head, \*p, \*q;

head=p=(LNode \*)malloc(sizeof(LNode));

head->next=NULL;

do{

scanf("%c",& data);

q= (LNode \*)malloc(sizeof(LNode));

q->data=data;

q->next=p->next;

p->next=q;

p=q ;

}while(getchar()!='\n');

return (head);

}

char max\_char\_linked\_list(LNode \*a)

{

a=a->next;

if(a == NULL)

printf("Invalid input!");

int max = a->data;

while(a!=NULL)

{

if(a->data > max){

max = a->data;

}

a=a->next;

}

return max;

}

void printList(LNode\* list){

printf("Head");

while(1){

printf("->");

list = list->next;

if(list == NULL){

printf("NULL");

exit(0);

}

else if(list->next == NULL) {

printf("%c->NULL\n", list->data);

break;

}

else

printf("%c",list->data);

}

}

int main(void) {

LNode \*a;

a = create\_LinkList();

printList(a);

int res=max\_char\_linked\_list(a);

printf("The maxium value of the linked list is: %c",res);

return 0;

}

(d)

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* File **Na**me: LinkedList-Lab1-q3(d)

\* Author: JingjingTan

\* Engineering School, NPU

\* description: find the minium value for a given singly

\* linked list of char, like L = head->c->d->a->f->NULL,

\* then result will be a.

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <stdio.h>

#include <stdlib.h>

typedef char ElemType;

typedef struct Lnode

{

ElemType data;

struct Lnode \*next;

}LNode;

LNode \*create\_LinkList(void){

char data;

LNode \*head, \*p, \*q;

head=p=(LNode \*)malloc(sizeof(LNode));

head->next=NULL;

do{

scanf("%c",& data);

q= (LNode \*)malloc(sizeof(LNode));

q->data=data;

q->next=p->next;

p->next=q;

p=q ;

}while(getchar()!='\n');

return (head);

}

char min\_char\_linked\_list(LNode \*a)

{

a=a->next;

if(a == NULL)

printf("Invalid input!");

int min = a->data;

while(a!=NULL)

{

if(a->data < min){

min = a->data;

}

a=a->next;

}

return min;

}

void printList(LNode\* list){

printf("Head");

while(1){

printf("->");

list = list->next;

if(list == NULL){

printf("NULL");

exit(0);

}

else if(list->next == NULL) {

printf("%c->NULL\n", list->data);

break;

}

else

printf("%c",list->data);

}

}

int main(void) {

LNode \*a;

a = create\_LinkList();

printList(a);

int res=min\_char\_linked\_list(a);

printf("The minium value of the linked list is: %c",res);

return 0;

}

Question 4:

(a)

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* File **Na**me: LinkedList-Lab1-q4(a)

\* Author: JingjingTan

\* Engineering School, NPU

\* description: find the maxium and minium value for a given singly

\* linked list of int, like L=Head->19->3->6->8->2->32->9->11->NULL,

\* then result will be like L=Head->19->3->6->8->32->2->9->11->NULL.

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <stdio.h>

#include <stdlib.h>

typedef int ElemType;

typedef struct Lnode

{

ElemType data;

struct Lnode \*next;

}LNode;

LNode \*create\_LinkList(void){

int data;

LNode \*head, \*p, \*q;

head=p=(LNode \*)malloc(sizeof(LNode));

head->next=NULL;

do{

scanf("%d",& data);

q= (LNode \*)malloc(sizeof(LNode));

q->data=data;

q->next=p->next;

p->next=q;

p=q ;

}while(getchar()!='\n');

return (head);

}

int max\_int\_linked\_list(LNode \*a)

{

a=a->next;

if(a == NULL)

printf("Invalid input!");

int max = a->data;

while(a!=NULL)

{

if(a->data > max){

max = a->data;

}

a=a->next;

}

return max;

}

int min\_int\_linked\_list(LNode \*a)

{

a=a->next;

if(a == NULL)

printf("Invalid input!");

int min = a->data;

while(a!=NULL)

{

if(a->data < min){

min = a->data;

}

a=a->next;

}

return min;

}

void printList(LNode\* list){

printf("Head");

while(1){

printf("->");

list = list->next;

if(list == NULL){

printf("NULL");

exit(0);

}

else if(list->next == NULL) {

printf("%d->NULL\n", list->data);

break;

}

else

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

}

}

void swap(LNode \*L, int x, int y){

if (x==y)

return;

LNode \*prev\_x,\*cur\_x=L;

while(cur\_x && cur\_x->data != x)

{

prev\_x = cur\_x;

cur\_x = cur\_x->next;

}

LNode \*prev\_y=NULL, \*cur\_y = L;

while(cur\_y && cur\_y->data !=y){

prev\_y = cur\_y;

cur\_y = cur\_y->next;

}

if(cur\_x == NULL || cur\_y == NULL)

return;

if(prev\_x != NULL)

prev\_x->next = cur\_y;

else

L = cur\_y;

if(prev\_y!= NULL)

prev\_y->next = cur\_x;

else

L = cur\_x;

LNode \*temp=cur\_y->next;

cur\_y->next = cur\_x->next;

cur\_x->next = temp;

}

int main(void) {

LNode \*a;

a = create\_LinkList();

printList(a);

int max=max\_int\_linked\_list(a);

printf("The maxium value of the linked list is: %d\n",max);

int min=min\_int\_linked\_list(a);

printf("The minium value of the linked list is: %d\n",min);

printf("Now let's swap the position of the max and min: \n");

swap(a, min, max);

printList(a);

return 0;

}

(b)

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* File **Na**me: LinkedList-Lab1-q4(b)

\* Author: JingjingTan

\* Engineering School, NPU

\* description: find the maxium and minium value for a given singly

\* linked list of char, like L = Head->f->g->v->x->e->a->d->h->NULL,

\* then result will be like L=Head->f->g->v->a->e->x->d->h->NULL.

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <stdio.h>

#include <stdlib.h>

typedef char ElemType;

typedef struct Lnode

{

ElemType data;

struct Lnode \*next;

}LNode;

LNode \*create\_LinkList(void){

char data;

LNode \*head, \*p, \*q;

head=p=(LNode \*)malloc(sizeof(LNode));

head->next=NULL;

do{

scanf("%c",& data);

q= (LNode \*)malloc(sizeof(LNode));

q->data=data;

q->next=p->next;

p->next=q;

p=q ;

}while(getchar()!='\n');

return (head);

}

char max\_char\_linked\_list(LNode \*a)

{

a=a->next;

if(a == NULL)

printf("Invalid input!");

int max = a->data;

while(a!=NULL)

{

if(a->data > max){

max = a->data;

}

a=a->next;

}

return max;

}

char min\_char\_linked\_list(LNode \*a)

{

a=a->next;

if(a == NULL)

printf("Invalid input!");

int min = a->data;

while(a!=NULL)

{

if(a->data < min){

min = a->data;

}

a=a->next;

}

return min;

}

void printList(LNode\* list){

printf("Head");

while(1){

printf("->");

list = list->next;

if(list == NULL){

printf("NULL");

exit(0);

}

else if(list->next == NULL) {

printf("%c->NULL\n", list->data);

break;

}

else

printf("%c",list->data);

}

}

void swap(LNode \*L, char x, char y){

if (x==y)

return;

LNode \*prev\_x,\*cur\_x=L;

while(cur\_x && cur\_x->data != x)

{

prev\_x = cur\_x;

cur\_x = cur\_x->next;

}

LNode \*prev\_y=NULL, \*cur\_y = L;

while(cur\_y && cur\_y->data !=y){

prev\_y = cur\_y;

cur\_y = cur\_y->next;

}

if(cur\_x == NULL || cur\_y == NULL)

return;

if(prev\_x != NULL)

prev\_x->next = cur\_y;

else

L = cur\_y;

if(prev\_y!= NULL)

prev\_y->next = cur\_x;

else

L = cur\_x;

LNode \*temp=cur\_y->next;

cur\_y->next = cur\_x->next;

cur\_x->next = temp;

}

int main(void) {

LNode \*a;

a = create\_LinkList();

printList(a);

char max=max\_char\_linked\_list(a);

printf("The maxium value of the linked list is: %c\n",max);

char min=min\_char\_linked\_list(a);

printf("The minium value of the linked list is: %c\n",min);

printf("Now let's swap the position of the max and min: \n");

swap(a, min, max);

printList(a);

return 0;

}