实验二

#include<string.h>

#include<ctype.h>

#include<malloc.h> /\* malloc()等 \*/

#include<limits.h> /\* INT\_MAX等 \*/

#include<stdio.h> /\* EOF(=^Z或F6),NULL \*/

#include<stdlib.h> /\* atoi() \*/

#include<io.h> /\* eof() \*/

#include<math.h> /\* floor(),ceil(),abs() \*/

#include<process.h> /\* exit() \*/

#define TRUE 1

#define FALSE 0

#define OK 1

#define ERROR 0

#define INFEASIBLE -1

typedef int ElemType;

typedef int Status;

typedef int Boolean;

typedef struct myNode

{

ElemType data;

struct myNode\* next;

} Node;

typedef Node\* LinkList;

Status InitList(LinkList \*L)

{

\*L = (LinkList)malloc(sizeof(Node));

if(!(\*L))

exit(OVERFLOW);

(\*L)->next = NULL;

return OK;

}

Status ListInsert(LinkList L,int i, int e)

{

LinkList p,p1;

int j=1;

p = L->next;

while (p && j<i-1)

{

p=p->next;

++j;

}

p1 = (LinkList)malloc(sizeof(Node));

if(!p1)

exit(OVERFLOW);

p1->data = e;

p1->next = p->next;

p->next = p1;

return OK;

}

Status ListDelete(LinkList L,int i,int \*e)

{

LinkList p,p1;

int j=1;

p = L->next;

while(p && j<i-1)

{

j++;

p = p->next;

}

if(!(p->next) || j>i-1)

return ERROR;

p1 = p->next;

p->next = p1->next;

\*e = p1->data;

free(p1);

return OK;

}

Status Listout(LinkList L)

{

LinkList p;

p = L->next;

while(p)

{

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

p = p->next;

}

return OK;

}

void CreatList(LinkList \*L, int n)

{

InitList(L);

LinkList p1;

int i;

for(i=n, p1=\*L; i>0; i--)

{

LinkList p;

int a;

scanf("%d", &a);

//ListInsert(\*L, i, a);

p = (LinkList)malloc(sizeof(Node));

p->data = a;

p1->next= p;

p1 = p1->next;

//\*L = p1

//p->next = (\*L)->next;

//(\*L)->next = p;

}

p1->next = NULL;

}

LinkList Reverse(LinkList L)

{

LinkList Temp, Prev;

Prev = NULL;

while (L)

{

Temp = L->next;

L->next = Prev;

Prev = L;

L = Temp;

}

return Prev;

}

int main()

{

LinkList L;

//int i,e;

//i=2,e=11;

CreatList(&L, 6);

Reverse(L);

//ListInsert(L,i,e);

Listout(L);

return 0;

}

总结：本次实验要求掌握定义链表的存储结构和实现逆序建立链表，插入、删除节点，难点在于实现链表的逆转，结合所学易知通过指针变换指向可以实现操作。