## 实验2

学号：2022217587 姓名：党存远 专业：物联网工程

|  |  |  |  |
| --- | --- | --- | --- |
| **“数据结构课程实验”验收细则** | | | |
| **成绩等级** | **具体表现** | **教师评分**  **（百分制）** | |
| A（100-90] | 1）能较快完成实验；2）编码正确；3）能完整正确地回答教师提问； | □ |  |
| B（90-80] | 1）能够在规定时间内完成实验；2）编码正确；3）能正确地回答教师提问； | □ |  |
| C（80-70] | 1）能够在规定时间内完成实验； | □ |  |
| D（70-60] | 1）虽完成实验但未能在规定时间内完成验收的； | □ |  |
| E（<60） | 1）未能进行验收的； | □ |  |

**2.1 实验目的**

熟练掌握线性表的链式存储结构。

熟练掌握单链表的有关算法设计。

根据具体问题的需要，设计出合理的表示数据的链式存储结构，并设计相关算法。

**2.2 实验要求**

本次实验中的链表结构指带头结点的单链表；

单链表结构和运算定义，算法的实现以库文件方式实现，不得在测试主程序中直接实现；

比如存储、算法实现放入文件：linkedList.h

实验程序有较好可读性，各运算和变量的命名直观易懂，符合软件工程要求；

程序有适当的注释。

**2.3 实验任务**

编写算法实现下列问题的求解。

<1>尾插法创建单链表，打印创建结果。

<2>头插法创建单链表，打印创建结果。

<3>销毁单链表。

<4>求链表长度。

<5>求单链表中第i个元素（函数），若不存在，报错。

实验测试数据基本要求：

第一组数据：单链表长度n≥10，i分别为5，n，0，n+1，n+2

第二组数据：单链表长度n=0，i分别为0，2

<6>在第i个结点前插入值为x的结点。

实验测试数据基本要求：

第一组数据：单链表长度n≥10，x=100, i分别为5,n,n+1,0,1,n+2

第二组数据：单链表长度n=0，x=100，i=5

<7>链表中查找元素值为x的结点，成功返回结点指针，失败报错。

实验测试数据基本要求：

单链表元素为（1，3，6，10，15，16，17，18，19，20）

x=1，17，20，88

<8>删除单链表中第i个元素结点。

实验测试数据基本要求：

第一组数据：单链表长度n≥10，i分别为5,n,1,n+1,0

第二组数据：单链表长度n=0， i=5

<9>在一个递增有序的单链表L中插入一个值为x的元素，并保持其递增有序特性。

实验测试数据基本要求：

单链表元素为（10,20,30,40,50,60,70,80,90,100）,

x分别为25，85，110和8

<10>将单链表Ｌ中的奇数项和偶数项结点分解开（元素值为奇数、偶数），分别放入新的单链表中，然后原表和新表元素同时输出到屏幕上，以便对照求解结果。

实验测试数据基本要求：

第一组数据：单链表元素为（1,2,3,4,5,6,7,8,9,10,20,30,40,50,60）

第二组数据：单链表元素为（10,20,30,40,50,60,70,80,90,100）

**2.4 运行结果截图及说明**

<1>尾插法创建单链表，打印创建结果：

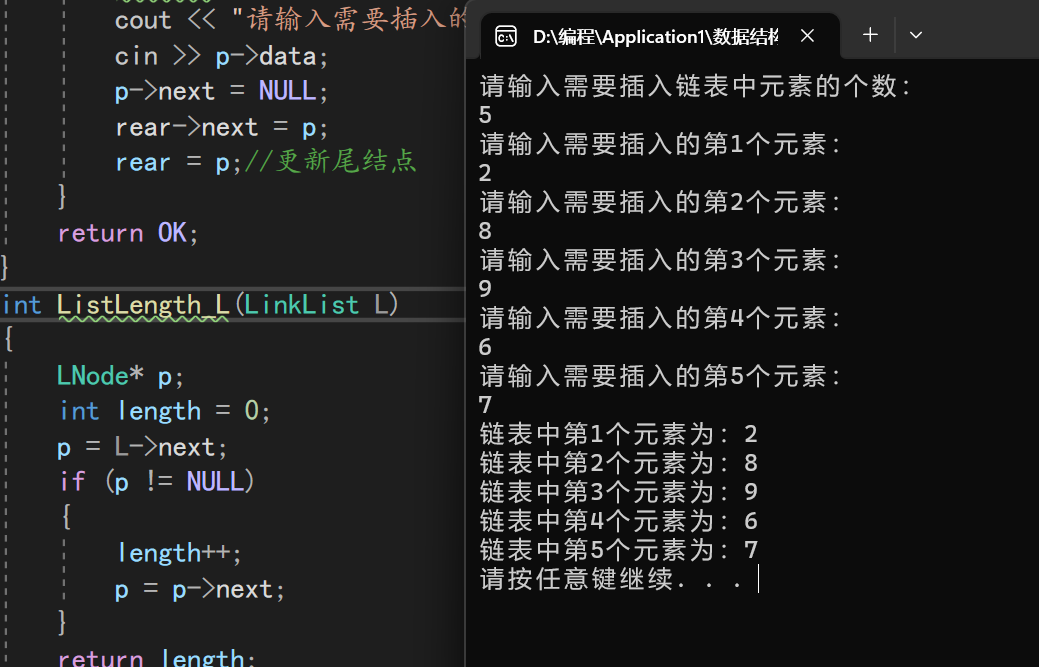
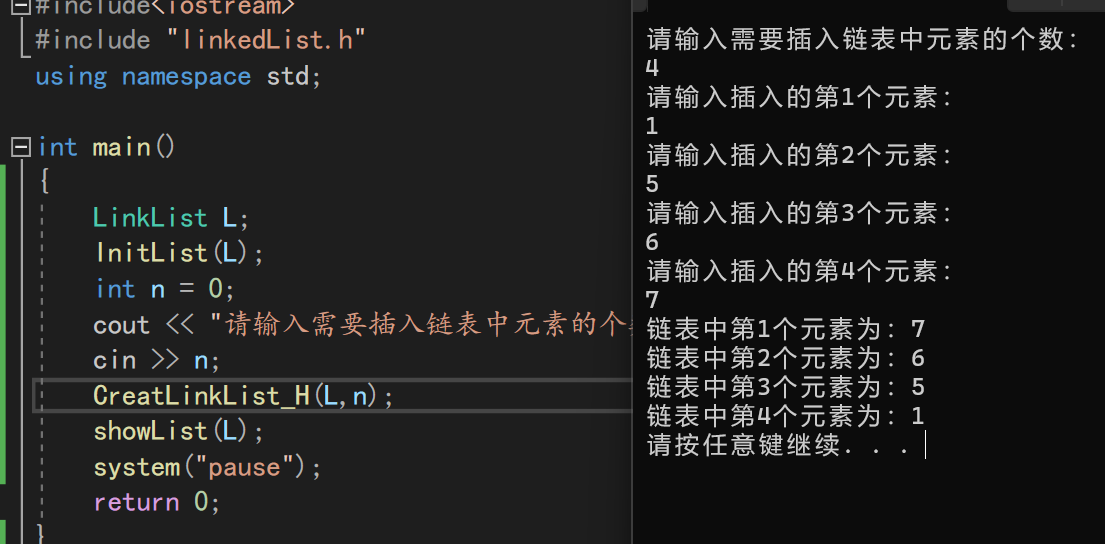


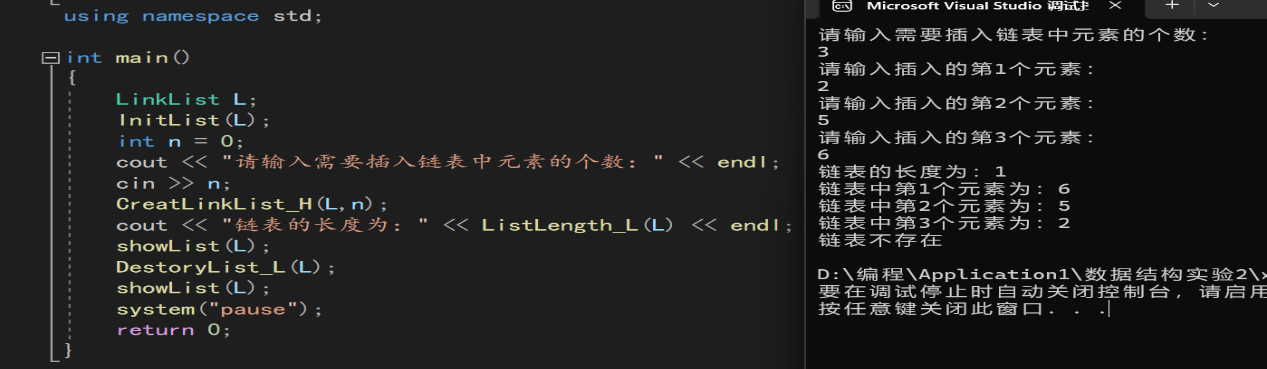
图1：尾插法创建一个单链表并且打印其中的元素

<2>头插法创建单链表，打印创建结果：



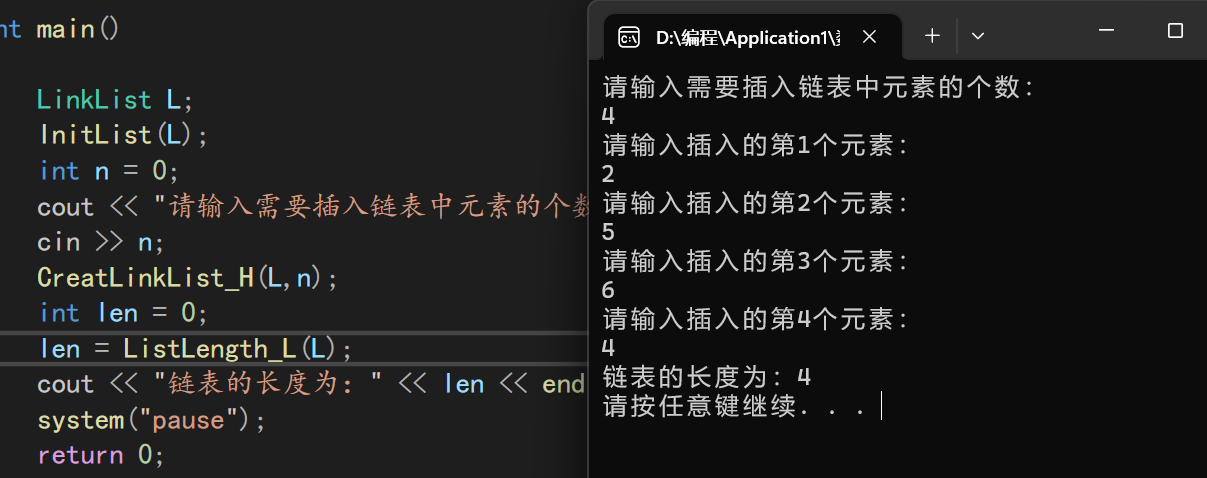
图二：尾插法插入数据

<3>销毁单链表：



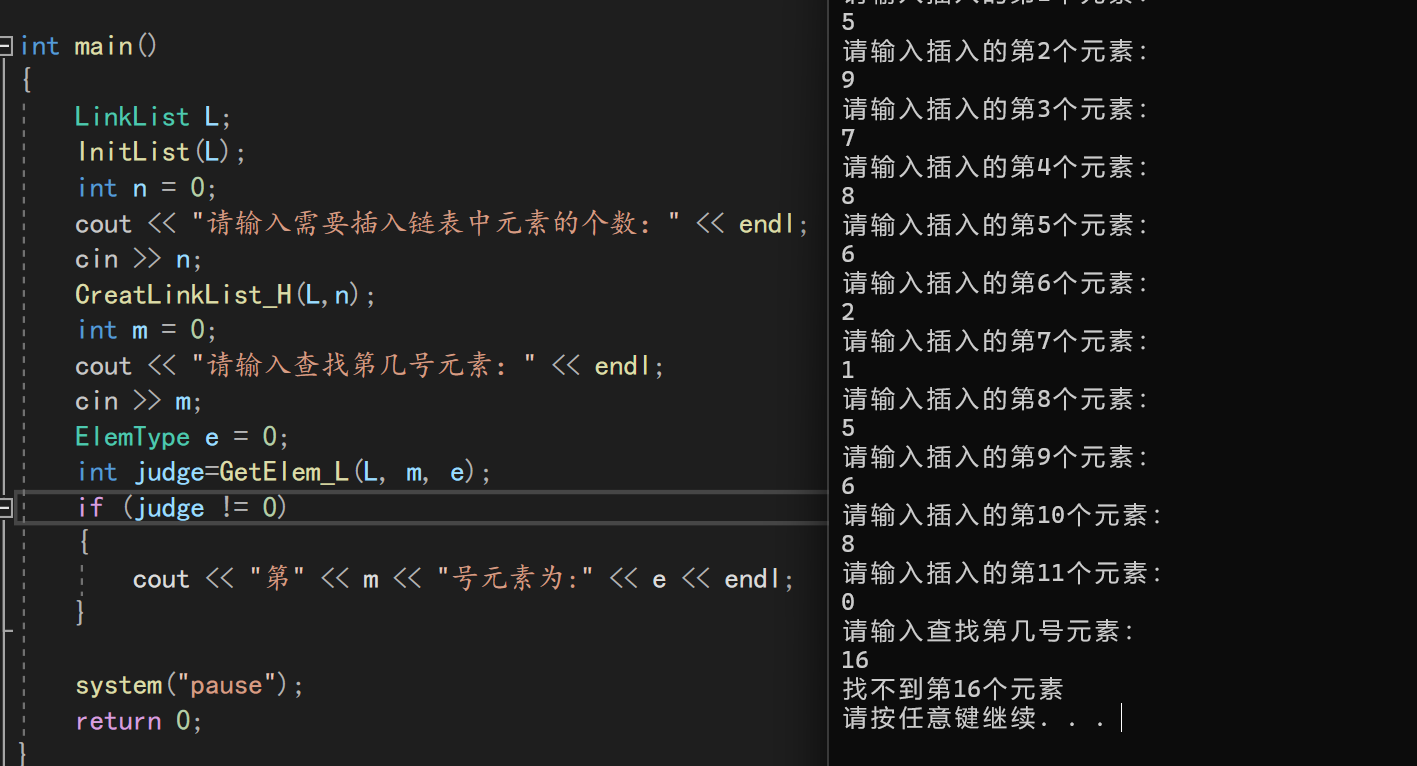
图三：链表的销毁

<4>求链表长度：



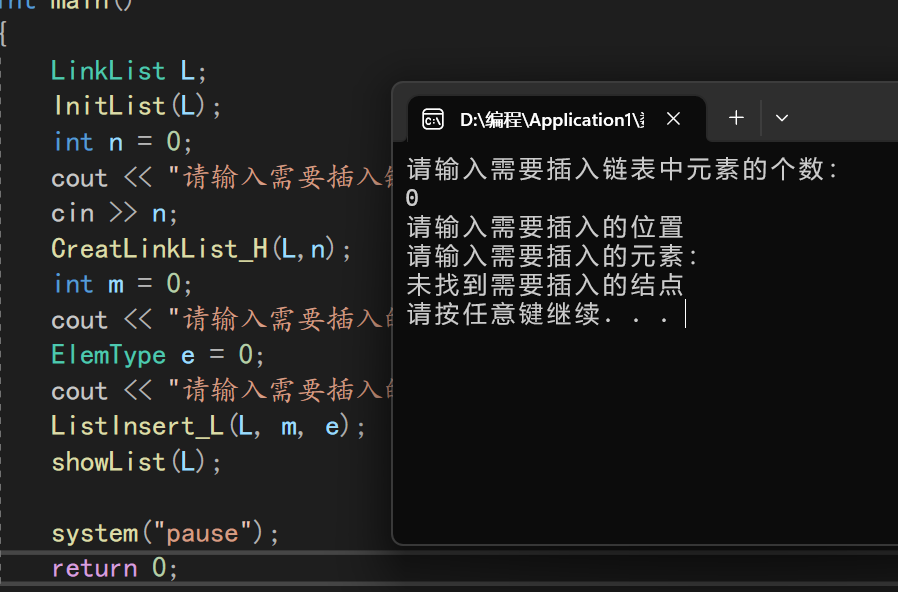
**图四：求链表的表长**

<5>求单链表中第i个元素（函数），若不存在，报错：

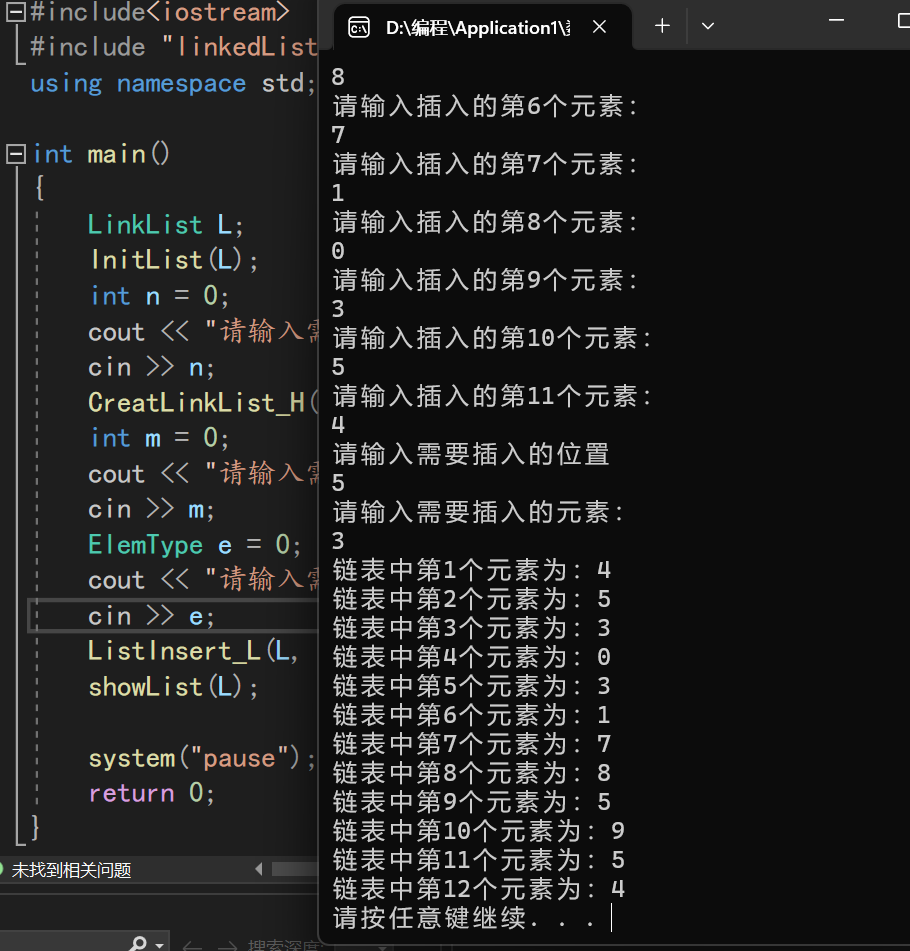
****

**图五：查找结点为i的元素**

<6>在第i个结点前插入值为x的结点：

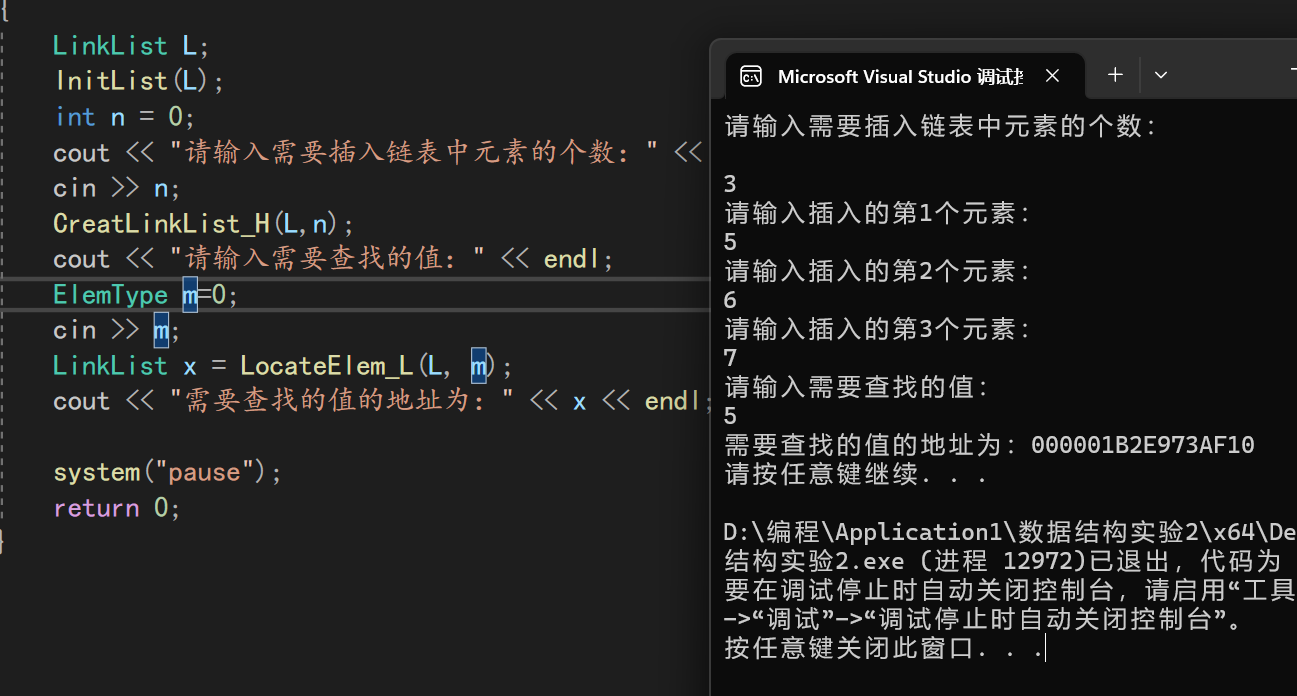
****

**图六：插入结点为x的结点**

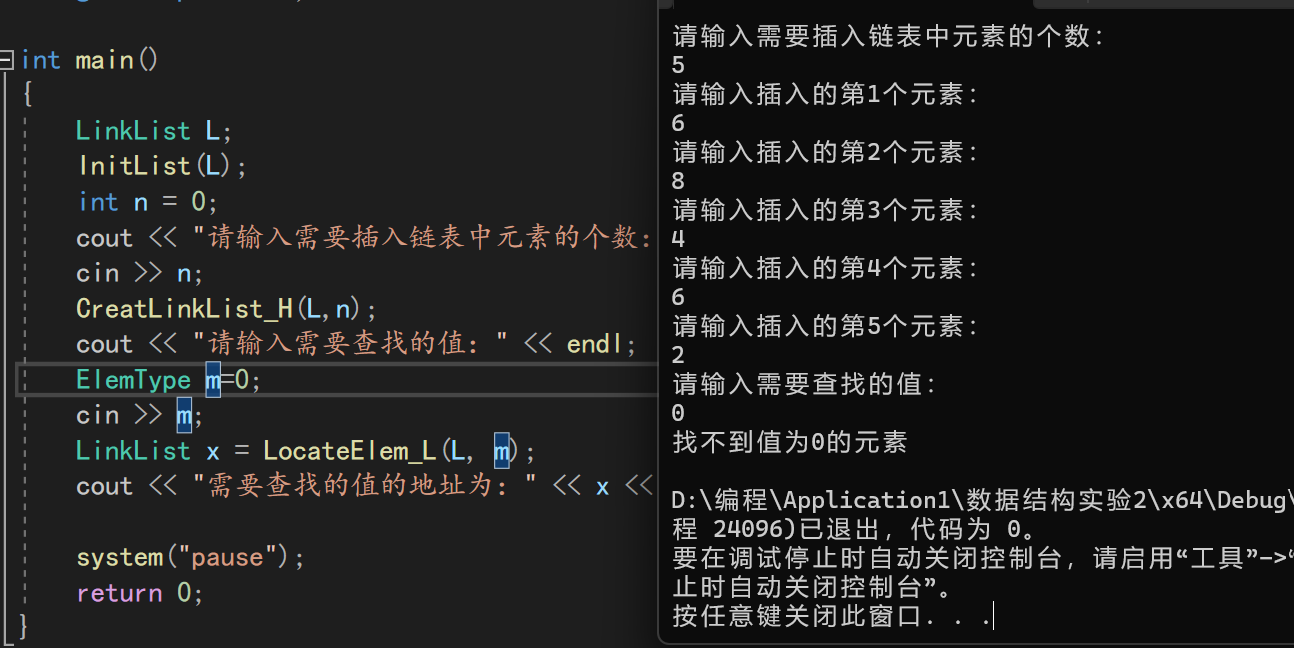
****

**图七：插入位置为i的元素**

<7>链表中查找元素值为x的结点，成功返回结点指针，失败报错：

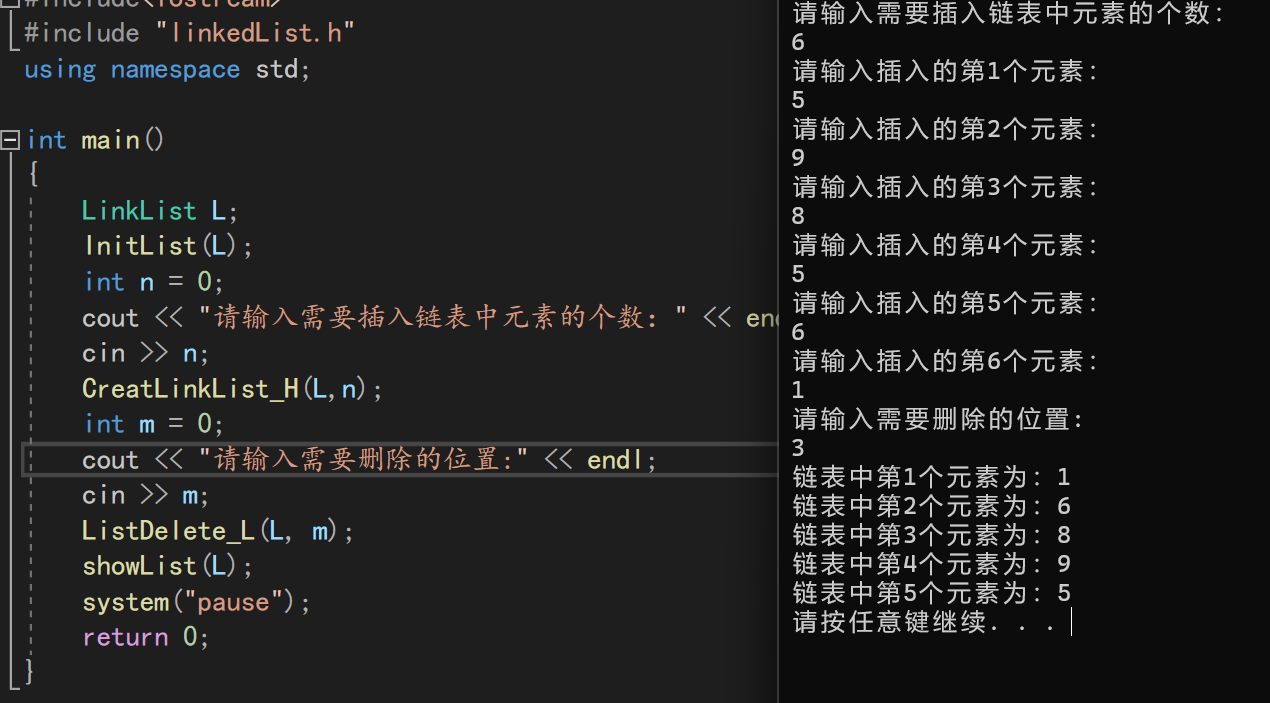
****

**图八：成功查找并且返回地址**

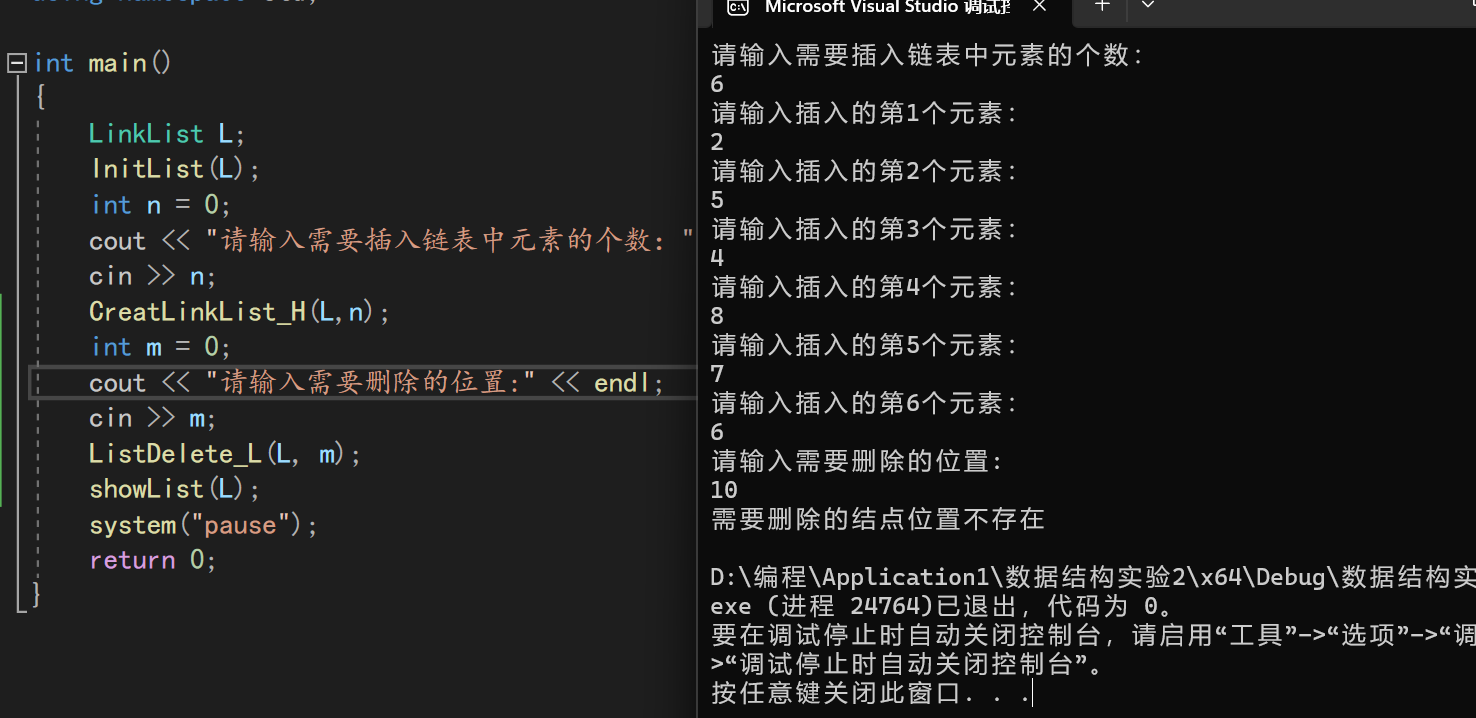
****

**图九：查找失败报错后退出程序**

<8>删除单链表中第i个元素结点：

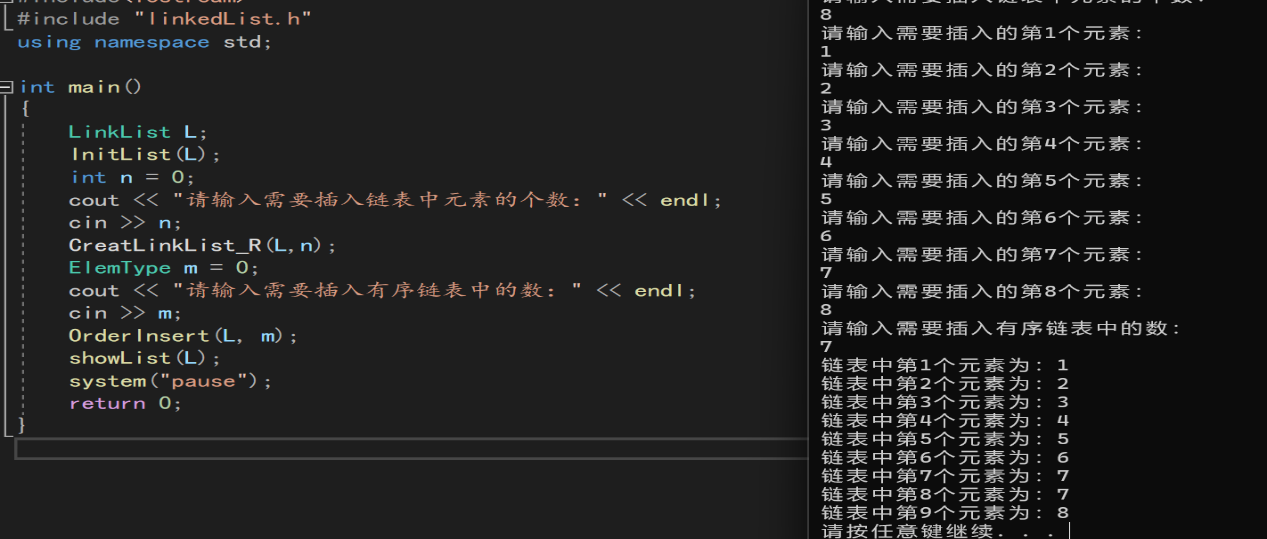
****

**图十：成功删除第i个结点并且打印结果**

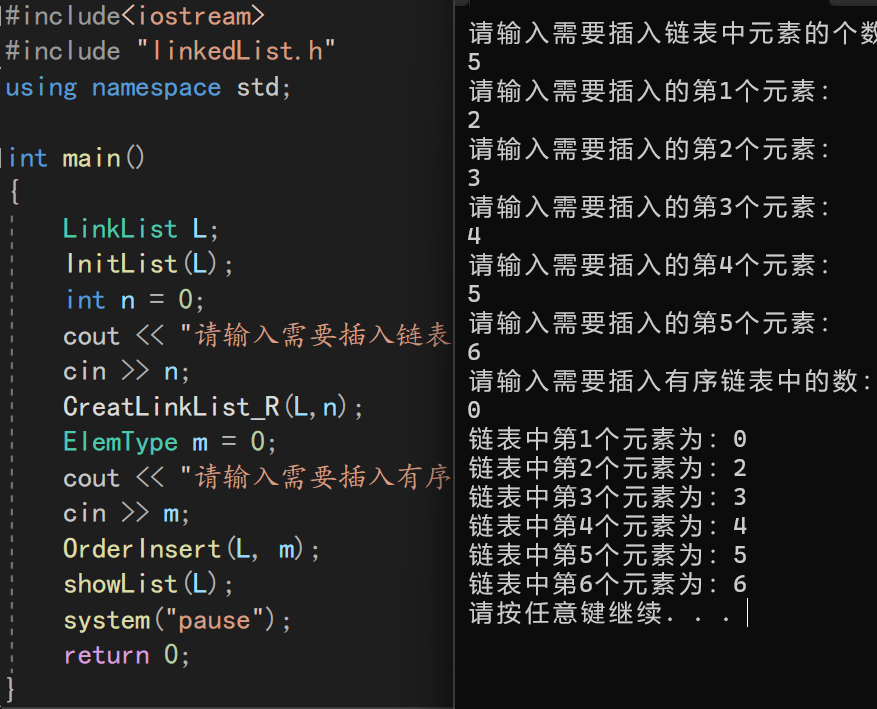
****

**图十一：未能成功删除结点，报错并且退出程序**

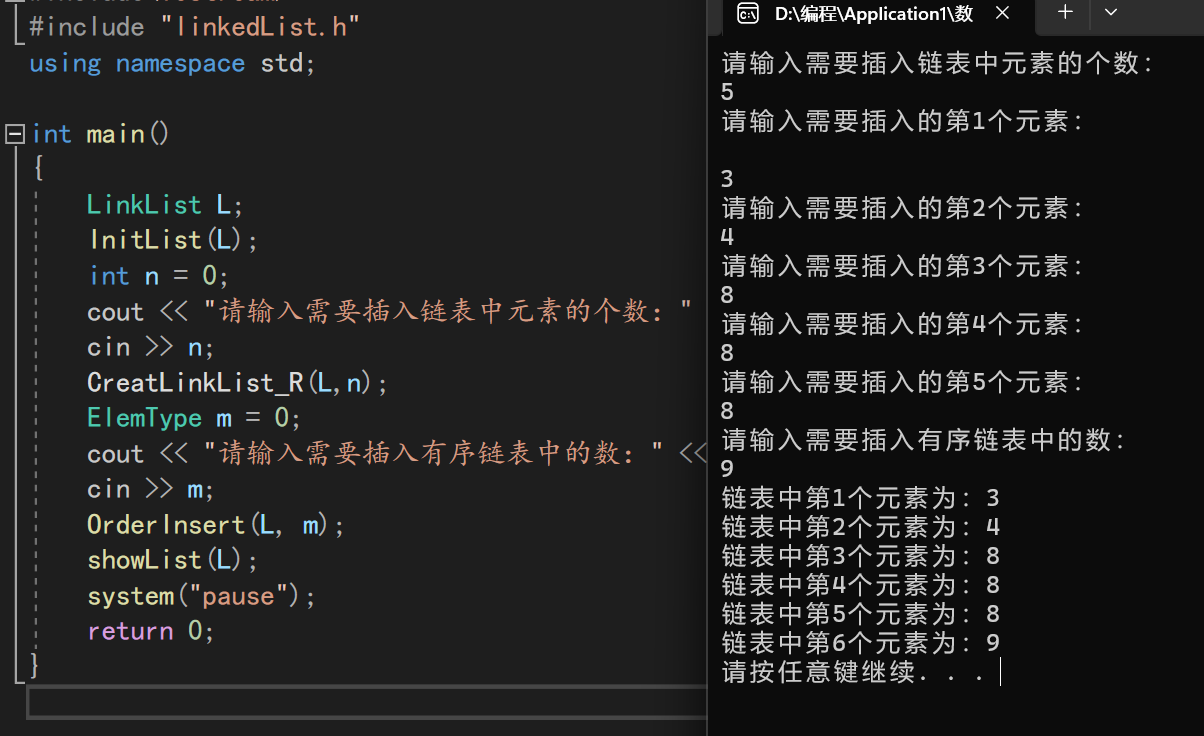
<9>在一个递增有序的单链表L中插入一个值为x的元素，并保持其递增有序特性：



**图十二：在链表中间插入一个数使其递增有序**

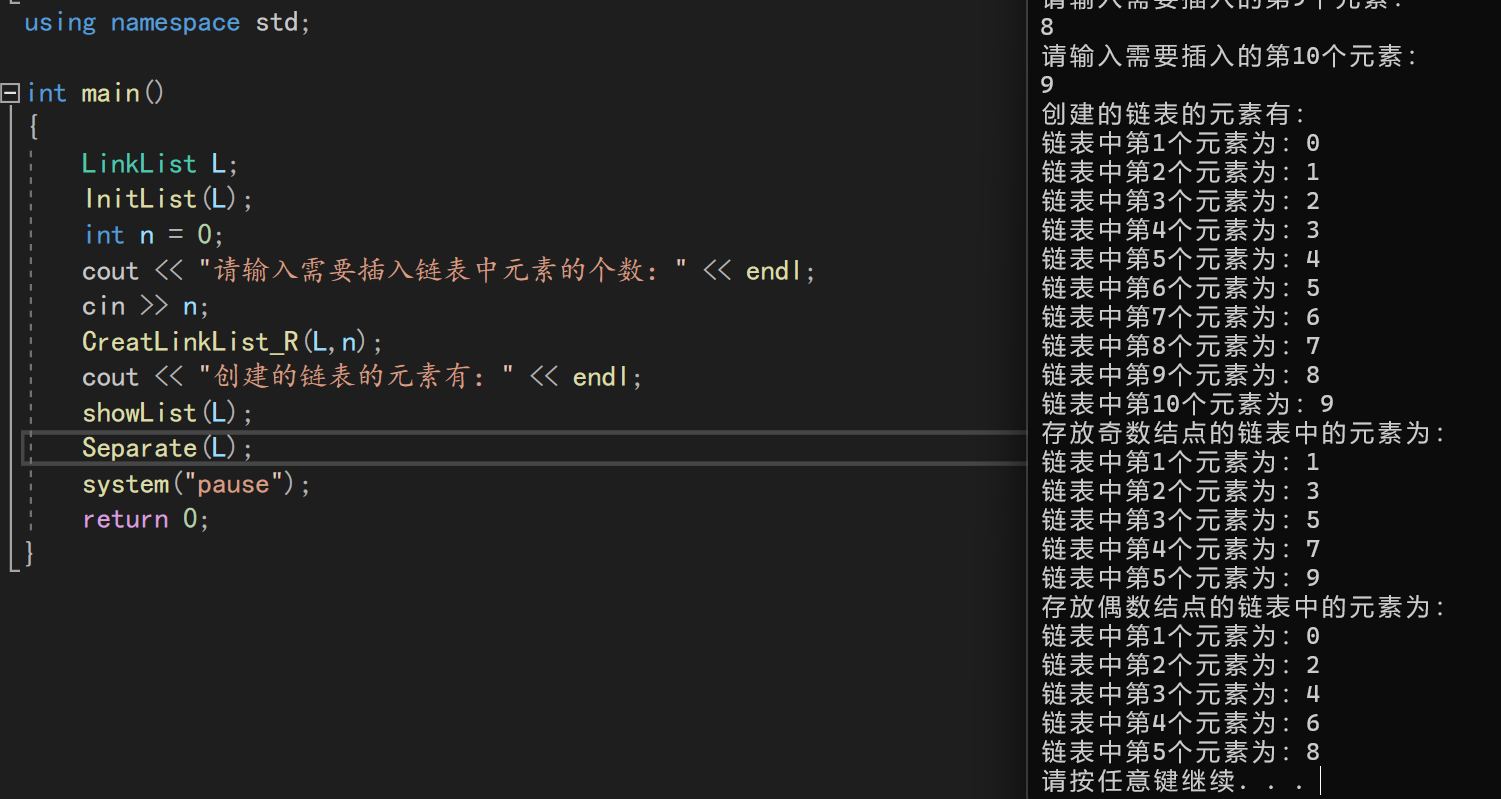
****

**图十三：在链表表头插入一个数使其地仍旧递增有序**

****

**图十四：在链表表尾插入一个数使其仍递增有序**

<10>将单链表Ｌ中的奇数项和偶数项结点分解开（元素值为奇数、偶数），分别放入新的单链表中，然后原表和新表元素同时输出到屏幕上，以便对照求解结果：

****

**图十五：将链表中的奇数结点与偶数结点分别放置在两个链表中**

**2.5 附源代码**

<1>尾插法创建单链表，打印创建结果：

int main()

{

LinkList L;

InitList(L);

int n = 0;

cout << "请输入需要插入链表中元素的个数：" << endl;

cin >> n;

CreatLinkList\_R(L,n);

showList(L);

system("pause");

return 0;

}

Status CreatLinkList\_R(LinkList& L, int n)//n表示需要插入多少个元素

{

L = new LNode;

L->next == NULL;

LinkList rear;

rear = L;

LNode \* p;

for (int i = 0; i < n; i++)

{

p = new LNode;

cout << "请输入需要插入的第"<<i+1<<"个元素：" << endl;

cin >> p->data;

p->next = NULL;

rear->next = p;

rear = p;//更新尾结点

}

return OK;

}

void showList(LinkList L)

{

LNode\* p;

p = L->next;

int i = 1;

while(p!=NULL)

{

cout << "链表中第" << i << "个元素为：" << p->data << endl;

p = p->next;

i++;

}

}

<2>头插法创建单链表，打印创建结果：

int main()

{

LinkList L;

InitList(L);

int n = 0;

cout << "请输入需要插入链表中元素的个数：" << endl;

cin >> n;

CreatLinkList\_H(L,n);

showList(L);

system("pause");

return 0;

}

Status CreatLinkList\_H(LinkList& L, int n)

{

if (L == NULL)

{

return ERROR;

}

LNode\* p;

for (int i = 0; i < n; i++)

{

p = new LNode;

cout << "请输入插入的第" << i + 1 << "个元素：" << endl;

cin >> p->data;

p->next = L->next;

L->next = p;

}

return OK;

}

void showList(LinkList L)

{

LNode\* p;

p = L->next;

int i = 1;

while(p!=NULL)

{

cout << "链表中第" << i << "个元素为：" << p->data << endl;

p = p->next;

i++;

}

}

<3>销毁单链表：

LinkList L;

InitList(L);

int n = 0;

cout << "请输入需要插入链表中元素的个数：" << endl;

cin >> n;

CreatLinkList\_H(L,n);

cout << "链表的长度为：" << ListLength\_L(L) << endl;

showList(L);

DestoryList\_L(L);

showList(L);

system("pause");

return 0;

Status DestoryList\_L(LinkList& L)

{

LNode\* p;

if (L == NULL)

{

cout << "链表不存在" << endl;

}

while (L!= NULL)

{

p = L;

L = L->next;

delete p;

}

return OK;

}

<4>求链表长度：

LinkList L;

InitList(L);

int n = 0;

cout << "请输入需要插入链表中元素的个数：" << endl;

cin >> n;

CreatLinkList\_H(L,n);

cout << "链表的长度为：" << ListLength\_L(L) << endl;

showList(L);

DestoryList\_L(L);

showList(L);

system("pause");

return 0;

int ListLength\_L(LinkList L)

{

LNode\* p;

int length = 0;

p = L->next;

while (p != NULL)

{

length++;

p = p->next;

}

return length;

}

<5>求单链表中第i个元素（函数），若不存在，报错：

LinkList L;

InitList(L);

int n = 0;

cout << "请输入需要插入链表中元素的个数：" << endl;

cin >> n;

CreatLinkList\_H(L,n);

int m = 0;

cout << "请输入查找第几号元素：" << endl;

cin >> m;

ElemType e = 0;

int judge=GetElem\_L(L, m, e);

if (judge != 0)

{

cout << "第" << m << "号元素为:" << e << endl;

}

system("pause");

return 0;

Status GetElem\_L(LinkList L, int i, ElemType& e)

{

LNode\* p;

if (L == NULL)

{

cout << "此表为空";

return ERROR;

}

p = L->next;

int j = 1;

while (p != NULL && j < i)

{

p = p->next;

j++;

}

if (p == NULL || j > i)

{

cout << "找不到第" << i << "个元素" << endl;

return ERROR;

}

e = p->data;

return OK;

}

<6>在第i个结点前插入值为x的结点：

LinkList L;

InitList(L);

int n = 0;

cout << "请输入需要插入链表中元素的个数：" << endl;

cin >> n;

CreatLinkList\_H(L,n);

int m = 0;

cout << "请输入需要插入的位置" << endl;

cin >> m;

ElemType e = 0;

cout << "请输入需要插入的元素：" << endl;

cin >> e;

ListInsert\_L(L, m, e);

showList(L);

system("pause");

return 0;

Status ListInsert\_L(LinkList& L, int i, ElemType e)

{

LNode\* p;

p = L;

int j = 0;

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

{

p = p->next;

j++;

}

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

{

cout << "未找到需要插入的结点" << endl;

return ERROR;

}

LNode\* s;

s = new LNode;

s->data = e;

s->next = p->next;

p->next = s;

return OK;

}

<7>链表中查找元素值为x的结点，成功返回结点指针，失败报错：

LinkList L;

InitList(L);

int n = 0;

cout << "请输入需要插入链表中元素的个数：" << endl;

cin >> n;

CreatLinkList\_H(L,n);

cout << "请输入需要查找的值：" << endl;

ElemType m=0;

cin >> m;

LinkList x = LocateElem\_L(L, m);

cout << "需要查找的值的地址为：" << x << endl;

system("pause");

return 0;

LinkList LocateElem\_L(LinkList L, ElemType e)

{

LNode\* p;

p = L->next;

while (p != NULL && p->data != e)

{

p = p->next;

}

if (p == NULL)

{

cout << "找不到值为" << e << "的元素" << endl;

exit(0);

}

return p;

}

<8>删除单链表中第i个元素结点：

LinkList L;

InitList(L);

int n = 0;

cout << "请输入需要插入链表中元素的个数：" << endl;

cin >> n;

CreatLinkList\_H(L,n);

int m = 0;

cout << "请输入需要删除的位置:" << endl;

cin >> m;

ListDelete\_L(L, m);

showList(L);

system("pause");

return 0;

Status ListDelete\_L(LinkList& L, int i)

{

LNode\* p;

p = L;

int j = 0;

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

{

p = p->next;

j++;

}

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

{

cout << "需要删除的结点位置不存在" << endl;

exit(0);

}

LNode\* q;

q = p->next;//q指向被删除的位置

if (q != NULL)

{

p->next = q->next;

delete q;

}

return OK;

}

<9>在一个递增有序的单链表L中插入一个值为x的元素，并保持其递增有序特性：

LinkList L;

InitList(L);

int n = 0;

cout << "请输入需要插入链表中元素的个数：" << endl;

cin >> n;

CreatLinkList\_R(L,n);

ElemType m = 0;

cout << "请输入需要插入有序链表中的数：" << endl;

cin >> m;

OrderInsert(L, m);

showList(L);

system("pause");

return 0;

Status OrderInsert(LinkList& L, ElemType e)

{

LNode\* p;

p = L->next;

if (p == NULL)

{

cout << "链表为空" << endl;

exit(0);

}

int j = 1;//先从第一个结点开始

while (p != NULL && p->data < e)//找到第一个比e大的位置插入元素

{

p = p->next;

j++;

}

ListInsert\_L(L, j, e);

return OK;

}

<10>将单链表Ｌ中的奇数项和偶数项结点分解开（元素值为奇数、偶数），分别放入新的单链表中，然后原表和新表元素同时输出到屏幕上，以便对照求解结果：

LinkList L;

InitList(L);

int n = 0;

cout << "请输入需要插入链表中元素的个数：" << endl;

cin >> n;

CreatLinkList\_R(L,n);

cout << "创建的链表的元素有：" << endl;

showList(L);

Separate(L);

system("pause");

return 0;

Status Separate(LinkList& L)

{

LinkList L1;

LinkList L2;

InitList(L1);

InitList(L2);

LNode\* p;

p = L->next;

int l1 = 1, l2 = 1;

while (p != NULL)

{

if (p->data % 2 == 1)

{

ListInsert\_L(L1, l1, p->data);

l1++;

}

else if (p->data % 2 == 0)

{

ListInsert\_L(L2, l2, p->data);

l2++;

}

p = p->next;

}

cout << "存放奇数结点的链表中的元素为：" << endl;

showList(L1);

cout << "存放偶数结点的链表中的元素为：" << endl;

showList(L2);

return OK;

}

**2.6 调试过程中出现的bug及总结**

1、在使用尾插法创建链表时忘记首先需要对链表初始化，导致程序报错。

2、尾插法与头插法都需要申请新空间进行插入操作。

3、在求单链表长度和打印显示单链表中元素的时候，由于参数传递时使用了引用导致程序报错。

4、在递增有序的单链表中插入元素使其仍递增有序的程序中，一开始分了三种情况：头部，中间，尾部，分别写了算法进行实现但是实际上可以当成一类处理，同时在使用插入操作时未能明确插入的位置导致插入出错。

5、在将奇数结点与偶数结点分开的操作中，对于新声明的两个链表未进行初始化操作导致程序报错。

希望能通过多多练习提高设计算法的能力。