// RotateListTest.cpp : 定义控制台应用程序的入口点。

//

/\*

第一周 第六题

题目：6) 给定一个链表，旋转链表，将链表每个节点向右移动 k 个位置，

其中 k 是非负数。

示例 1:

输入: 1->2->3->4->5->NULL, k = 2

输出: 4->5->1->2->3->NULL

解释:

向右旋转 1 步: 5->1->2->3->4->NULL

向右旋转 2 步: 4->5->1->2->3->NULL

示例 2:

输入: 0->1->2->NULL, k = 4

输出: 2->0->1->NULL

解释:

向右旋转 1 步: 2->0->1->NULL

向右旋转 2 步: 1->2->0->NULL

向右旋转 3 步: 0->1->2->NULL

向右旋转 4 步: 2->0->1->NULL

主要思路：

将链表中的数向右移k位。

先计算出链表的个数len

然后将k模len。

再将里面的两部分如上图1,2,3与4,5

反过来合在一起就可以了。

编译环境：Visual Studio 2015

作者：杨政权

\*/

//具体代码

#include "stdafx.h"

#include <iostream>

using namespace std;

//定义链表

struct ListNode {

int val;

ListNode \*next;

ListNode(int x):val(x),next(NULL){}

};

//创建一个链表

ListNode\* createList(int\* arr, int length) {

ListNode\* pHead = NULL;

ListNode\* pTemp=NULL, \*pNode;

for (int i = 0; i < length; i++) {

pNode = (ListNode\*)malloc(sizeof(ListNode)); //申请内存空间

pNode->val= arr[i];

pNode->next = NULL;

if (NULL == pHead)

pHead = pNode;

else

pTemp->next = pNode; //pNode 指向下一个节点

pTemp = pNode;

}

return pHead;

}

//主要算法

class RotateList {

public:

ListNode \*rotateRight(ListNode \*head, int k) {

if (head == NULL) //为空直接返回

return NULL;

int len = 1;

ListNode\* p1 = head;

while (p1->next != NULL)

{

len++;

//p1指向下一个节点

p1 = p1->next;

}

k = k%len;

if (k == 0)

{

//返回head

return head;

}

int step1 = len - k - 1;

p1 = head;

while (step1--)

p1 = p1->next;

ListNode\* p2 = p1->next;

ListNode\* p3 = p2;

while (p3->next != NULL)

p3 = p3->next;

p3->next = head;

p1->next = NULL;

return p2;

}

};

//打印链表

void printList(ListNode\* pHead) {

ListNode\* pNode = pHead;

while (pNode) {

cout << pNode->val << " ";

pNode = pNode->next;

}

cout << endl;

}

int main()

{

RotateList Rotate1,Rotate2;

int arr1[] = { 1, 2 ,3,4,5};

int arr2[] = { 0,1, 2 };

//测试1

cout << "测试1" << endl;

cout << "输入1，2，3，4，5 k=2" << endl;

cout << "输出:" << endl;

int length1 = sizeof(arr1) / sizeof(arr1[0]);

ListNode\* pHead1 = createList(arr1, length1);

pHead1 = Rotate1.rotateRight(pHead1, 2);

printList(pHead1);

cout << endl << endl;

//测试2

cout << "测试2" << endl;

cout << "输入 0，1，2 k=4" << endl;

cout << "输出：" << endl;

int length2 = sizeof(arr2) / sizeof(arr2[0]);

ListNode\* pHead2 = createList(arr2, length2);

pHead2 = Rotate2.rotateRight(pHead2, 4);

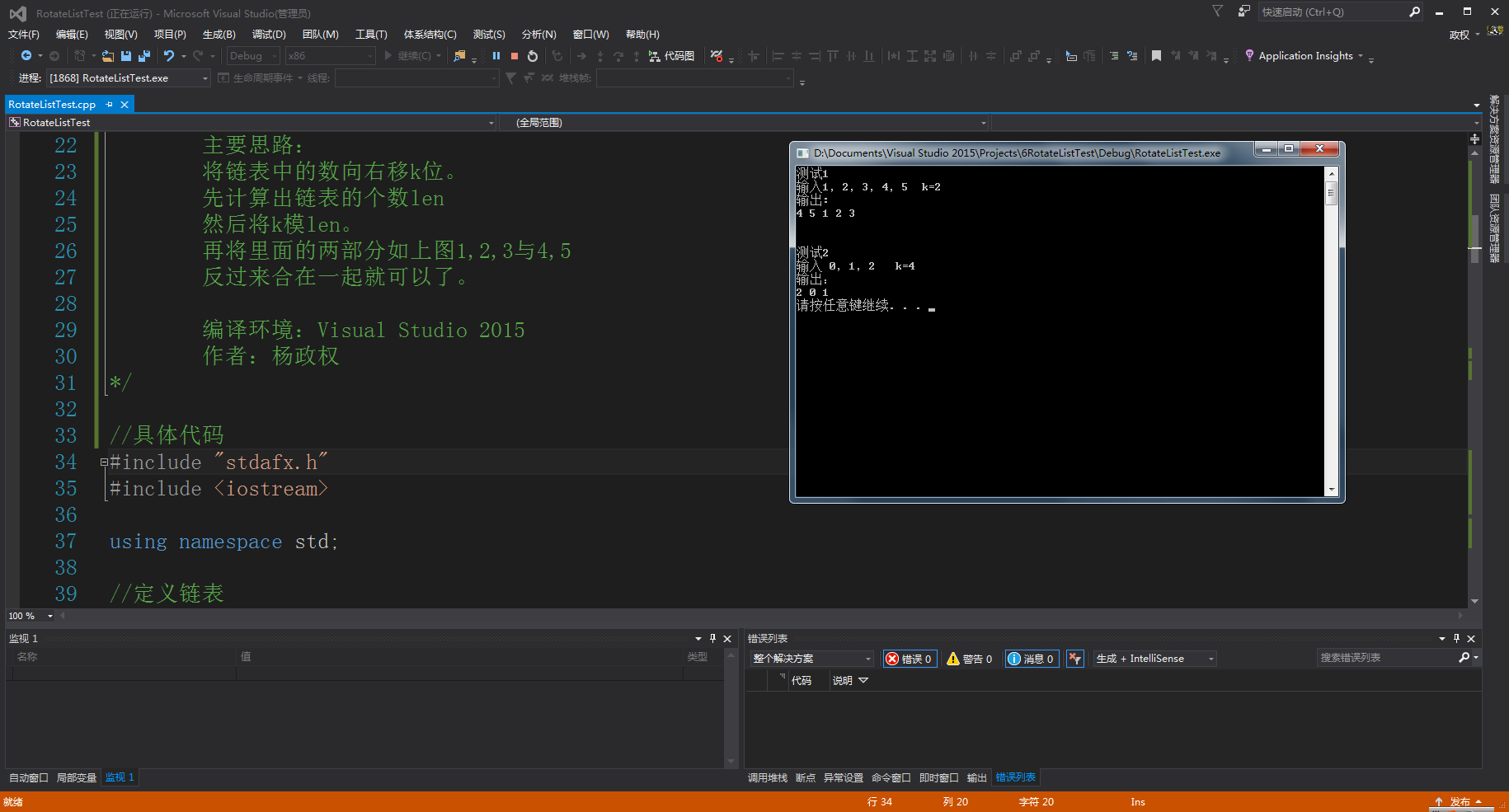
printList(pHead2);

system("pause");

return 0;

}

//程序运行截图：



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

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\* 实验总结：

\*\*\* 这道题是旋转链表

\*\*\* 主要是要熟练掌握链表的基本操作

\*\*\* 再找到这道题的解决思路就可以了

\*\*\* 难度中等

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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