两个有序链表序列的交集

作 者 姓 名： 王凌

学 号： 1951504

指 导 教 师： 张颖

学院、 专业： 软件工程

同济大学

Tongji University

目录

[1 分析 1](#_Toc14668)

[1.1 功能要求 1](#_Toc10868)

[1.2功能分析 1](#_Toc1044)

[2 设计 2](#_Toc31836)

[2.1 Node类设计 2](#_Toc7869)

[2.2 LinkList类设计 2](#_Toc1497)

[2.3主函数设计 3](#_Toc22012)

[2.3.1输入部分的设计 3](#_Toc17571)

[2.3.2求交集部分的设计 3](#_Toc19757)

[3实现 3](#_Toc8453)

[3.1链表类LinkList的实现 3](#_Toc23626)

[3.1.1自定义构造函数的实现 3](#_Toc18561)

[3.1.2自定义析构函数的实现 4](#_Toc8824)

[3.1.3成员函数getfirst()，empty()的实现 4](#_Toc3756)

[3.1.4成员函数ListDelete(int)的实现 5](#_Toc6643)

[3.1.5成员函数ListInsertTail(Node\*)的实现 6](#_Toc3559)

[3.1.6成员函数ClearLinkList()的实现 6](#_Toc8290)

[3.2主函数的实现 7](#_Toc20188)

[3.2.1输入部分的实现 7](#_Toc16097)

[3.2.2求交集部分的实现 8](#_Toc13544)

[4 测试 9](#_Toc5952)

[4.1 功能测试 9](#_Toc7902)

[4.1.1 测试1—— 一般情况 9](#_Toc18501)

[4.1.2 测试2—— 交集为空的情况 9](#_Toc26302)

[4.1.3 测试3—— 完全相交的情况 9](#_Toc26869)

[4.1.4 测试4—— 其中一个序列完全属于交集的情况 10](#_Toc93)

[4.1.5 测试5—— 其中一个序列为空的情况 10](#_Toc31535)

[4.2 边界测试 10](#_Toc13838)

[4.2.1 测试1—— 两个链表为空 10](#_Toc30752)

# 1 分析

## 功能要求

输入说明：输入分2行，分别在每行给出由若干个正整数构成的非降序序列，用-1表示序列的结尾（-1不属于这个序列）。数字用空格间隔。

输出说明：在一行中输出两个输入序列的交集序列，数字间用空格分开，结尾不能有多余空格；若新链表为空，输出NULL。

测试用例：

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 输入 | 输出 | 说明 |
| 1 | 1 2 5 -1  2 4 5 8 10 -1 | 2 5 | 一般情况 |
| 2 | 1 3 5 -1  2 4 6 8 10 -1 | NULL | 交集为空的情况 |
| 3 | 1 2 3 4 5 -1  1 2 3 4 5 -1 | 1 2 3 4 5 | 完全相交的情况 |
| 4 | 3 5 7 -1  2 3 4 5 6 7 8 -1 | 3 5 7 | 其中一个序列完全属于交集的情况 |
| 5 | -1  10 100 1000 -1 | NULL | 其中一个序列为空的情况 |

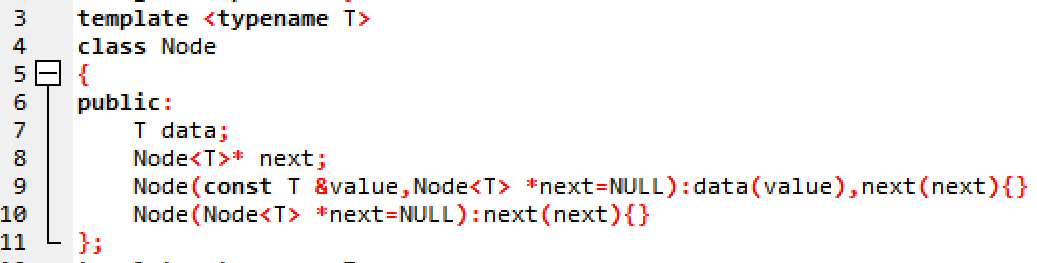
## 1.2功能分析

首先需要构造两条链表分别存储所给的两列信息，然后从链表头开始逐渐对比，并且把相同的元素打印显示，直到走到其中一个链表的尽头。

# 2 设计

## 2.1 Node类设计

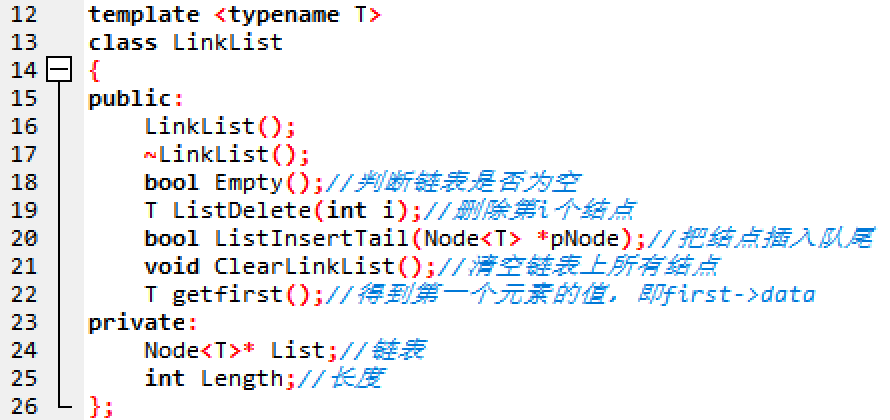
Node类是一个模板类，每一个Node对象为链表上的基础节点。它能够存储一个T类型的数据和一个指向下一个节点的指针变量。



## 2.2 LinkList类设计

有两个私有成员变量List数组和链表长度Length，起到了封装的作用。由于是私有成员变量，而链表上某些结点的值和链表的总长度无法在类外访问，根据题目的需求还需要第一个结点所存储的值，所以又因此设置了getfirst()函数。

每一个成员函数的作用依次是：Empty()函数用来判断链表是否为空；ListDelete(int i)用来删除链表上第i个结点；getfirst()得到第一个元素的值；ListInsertTail(Node<T> \*pNode)用来把结点插入链表尾部；ClearLinkList()用来清空链表上所有结点。



## 2.3主函数设计

### 2.3.1输入部分的设计

首先LinkList类创建两个链表对象listA、listB，然后提示用户输入两行数字，每一行分别以-1结尾。定义second\_bool这个初值为0的布尔型变量，它表示到了读第二个集合的真假，如果读到第一个-1，second\_bool变为1，不做存储操作，直接开始读第二个集合。用while循环读取并存储，每次循环先判断有没有读完。如果没有读完，根据second\_bool的值来判断是添加到listA还是listB的队尾。

### 2.3.2求交集部分的设计

定义初值为0的布尔类型变量Print\_bool来记录有无输出，如果没有输出则说明没有交集，输出NULL。

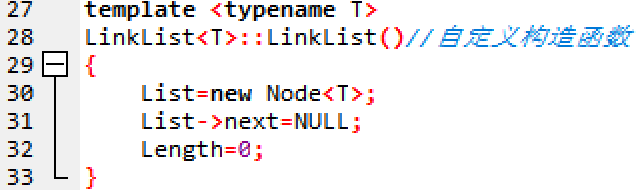
由于两个链表都是非降序，所以我们只需要在两个链表不为空的时候用while循环不断的比较两个链表的头节点的值的大小，如果他们相等，就打印出来，并且删除任意一个链表的头结点，本程序默认删除listA的头结点。如果它们不等，那么就删除较小的结点的头结点。while循环的终止条件是其中一条链表为空。

这应该是时间复杂度最小的做法，我们只需要比较min{listA.length,listB.length}次，是复杂度为O(n)的算法。

# 3实现

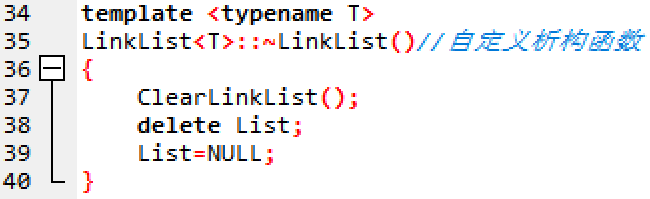
## 3.1链表类LinkList的实现

### 3.1.1自定义构造函数的实现



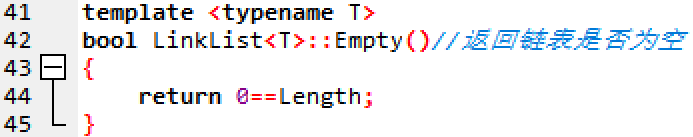
在链表头部生成一个头指针，注意，这个头结点是不存任何数据的，只标记链表的头部。初始化头节点的next为空，链表的长度为空。这样的目的是减少删除节点的分类讨论，详细的说明在2.2.4结点删除成员函数中，此处不再赘述。

### 3.1.2自定义析构函数的实现

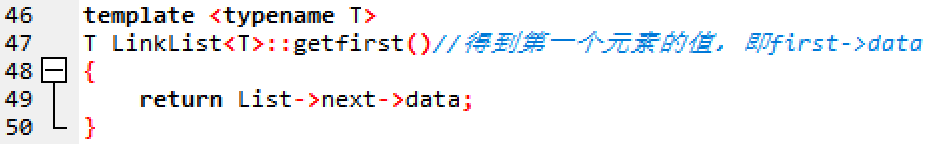


首先调用ClearLinkList()函数，清除除了List表头外的所有结点，再删除List表头结点，将表头置空。这样就删除了链表上的所有元素。

### 3.1.3成员函数getfirst()，empty()的实现

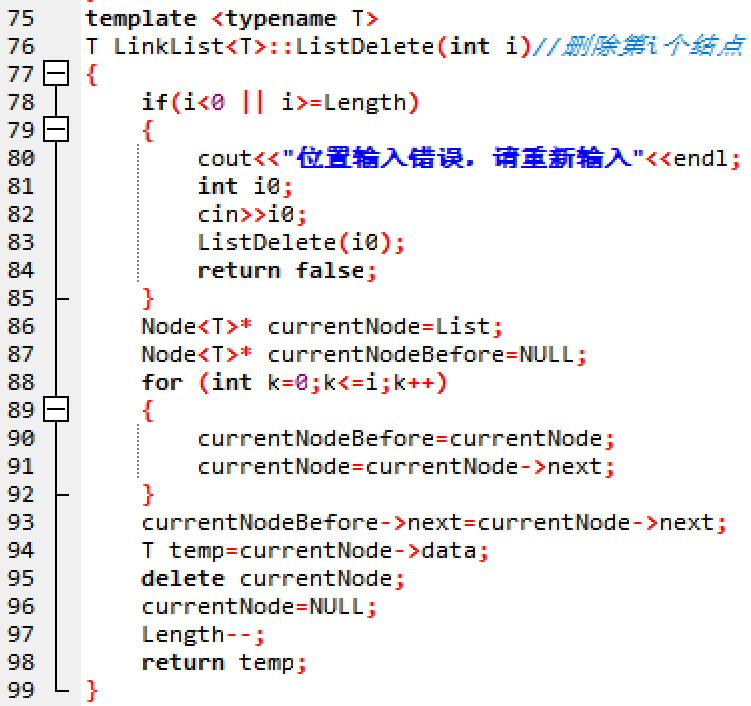


当长度为0时Empty()函数返回1，否则返回0。



返回第一个元素的值。

### 3.1.4成员函数ListDelete(int)的实现

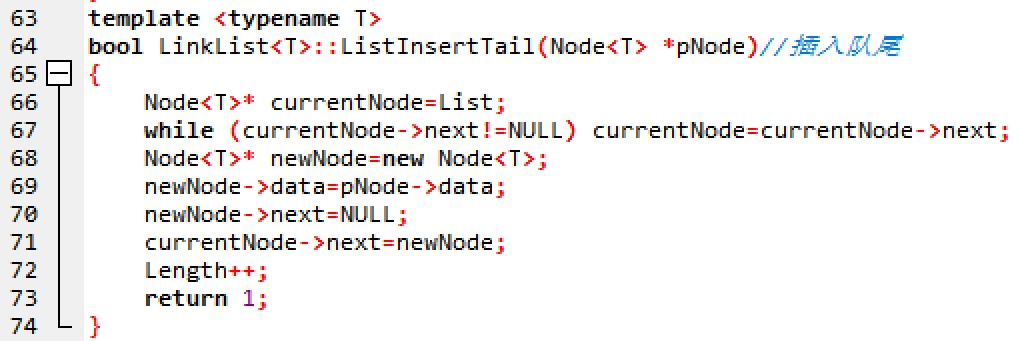


删除第i个结点，若输入i不在长度范围之内的话，报错且递归调用自身重新输入直到符合条件为止。

之后记录两个指针变量，currentNodeBefore变量记录当前结点之前一个结点的地址，currentNode结点记录当前结点地址。循环i+1次找到第i个结点和第i-1个结点。将第i-1个结点的后继（next）改为第i个结点的后继（next）这样原链表就删除了第i个结点。但是我们还需要得到第i个结点的值，并且删除第i个结点的存储空间碎片，所以我们记录第i个结点的数据，之后删除第i个结点，并且令链表总长度减一，这样就消除了删除结点的影响。最后返回所记录的第i个结点的值。

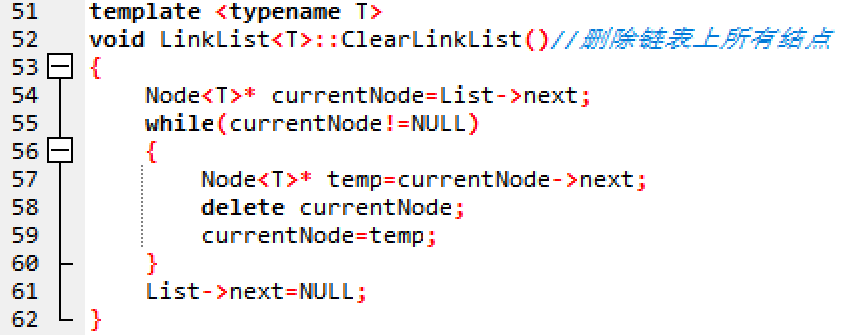
在张老师上课的PPT中链表中删除结点要分两种情况考虑，第一种是表头删除，第二种是表中和表尾。那为什么我只有一种情况呢？主要是因为我的链表设置了一个空表头List结点，这个结点不计数据，只记录表头位置，那么之后记录数据的结点相当于是表中或者是表尾了，所以不需要分情况讨论，节省了代码量与时间。

### 3.1.5成员函数ListInsertTail(Node\*)的实现



先用while循环从表头一直循环到表尾，先将该结点的数据存储，并将后继next改为空，之后再将原来最后一个结点的后继next改为要插入队尾的这个结点。之后链表总长度加一，返回1，说明结点插入链表队尾成功。

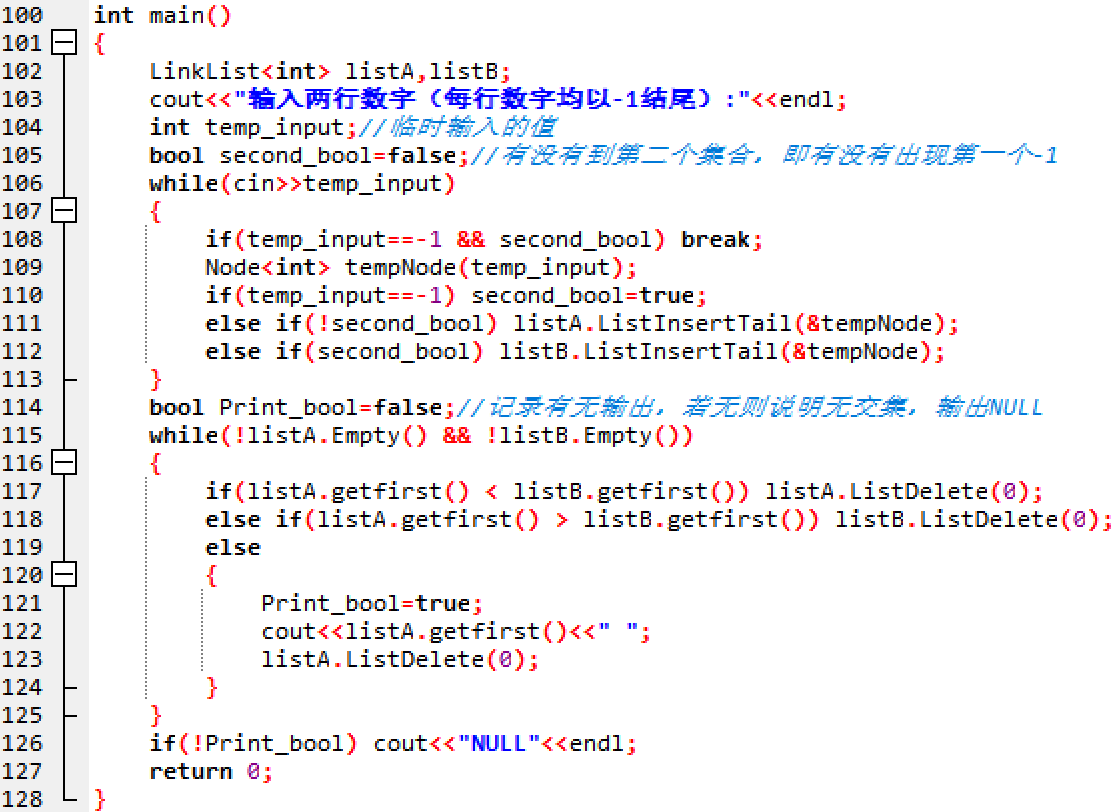
### 3.1.6成员函数ClearLinkList()的实现



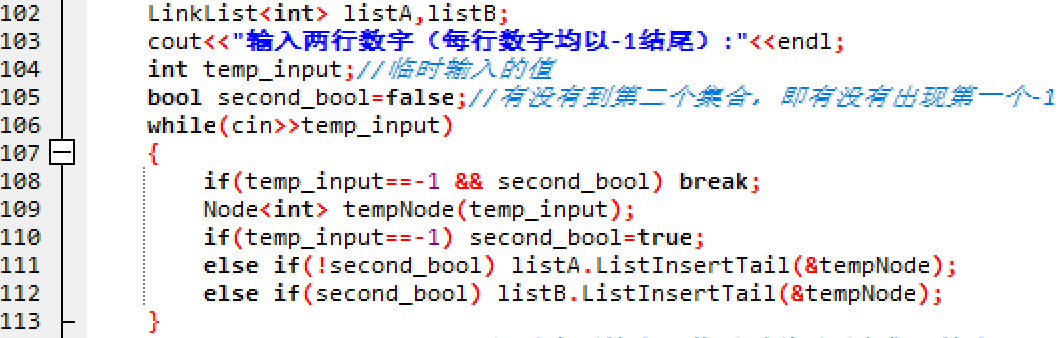
用while循环一直删除表头List的后继结点一直到表头的后继结点为空。最后剩下表头结点List。

这个函数是要搭配自定义析构函数一起用的，析构函数清空链表上所有含有数据的结点之后，再删除List表头结点，清空链表类的所有存储碎片。

## 3.2主函数的实现

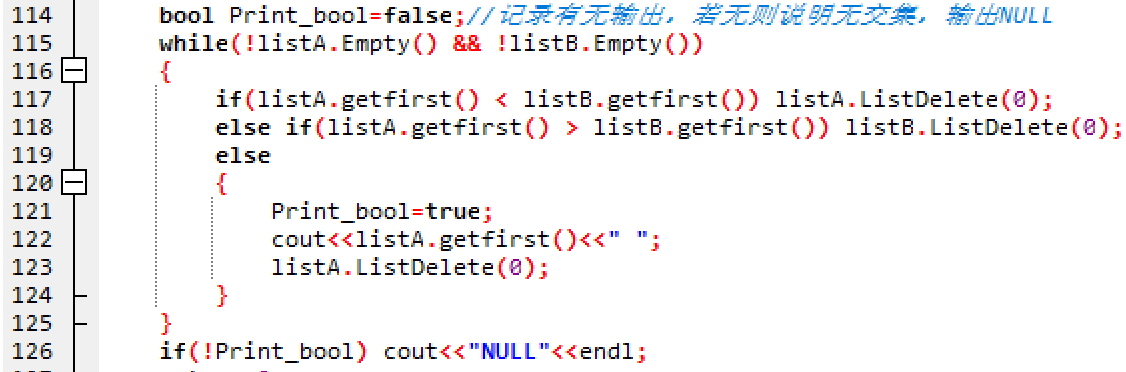


### 3.2.1输入部分的实现



首先LinkList类创建两个链表对象listA、listB，然后提示用户输入两行数字，每一行分别以-1结尾。定义second\_bool这个初值为0的布尔型变量，它表示到了读第二个集合的真假，如果读到第一个-1，second\_bool变为1，不做存储操作，直接开始读第二个集合。用while循环读取并存储，每次循环先判断有没有读完。如果没有读完，根据second\_bool的值来判断是添加到listA还是listB的队尾。

### 3.2.2求交集部分的实现



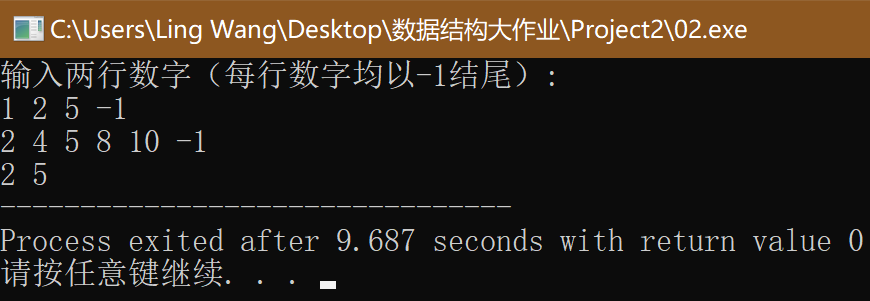
定义初值为0的布尔类型变量Print\_bool来记录有无输出，如果没有输出则说明没有交集，输出NULL。

由于两个链表都是非降序，所以我们只需要在两个链表不为空的时候用while循环不断的比较两个链表的头节点的值的大小，如果他们相等，就打印出来，并且删除任意一个链表的头结点，本程序默认删除listA的头结点。如果它们不等，那么就删除较小的结点的头结点。

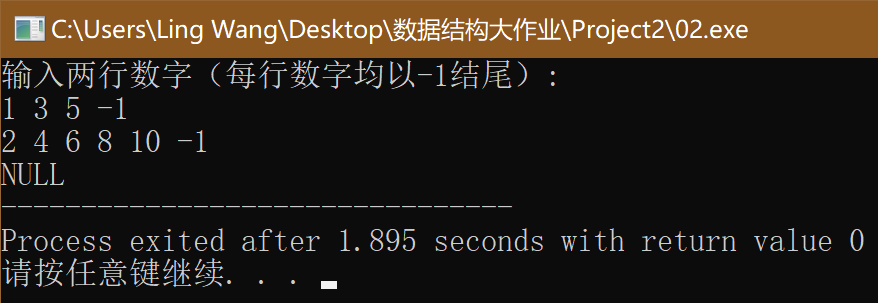
# 4 测试

## 4.1 功能测试

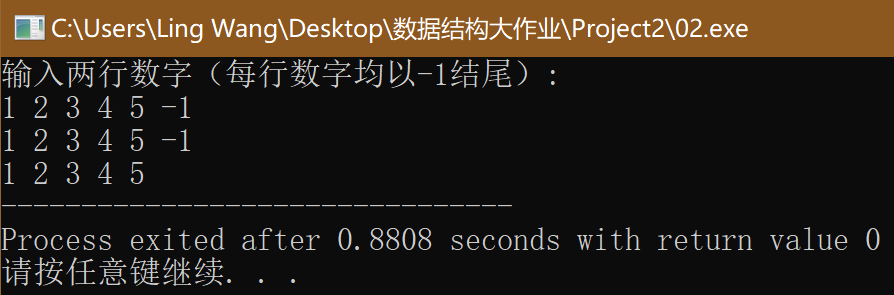
### 4.1.1 测试1—— 一般情况



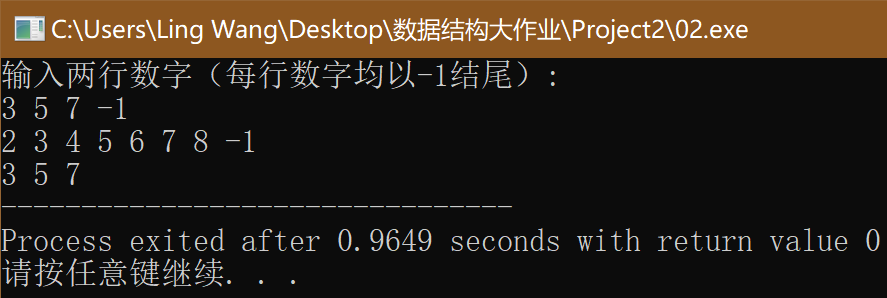
### 4.1.2 测试2—— 交集为空的情况



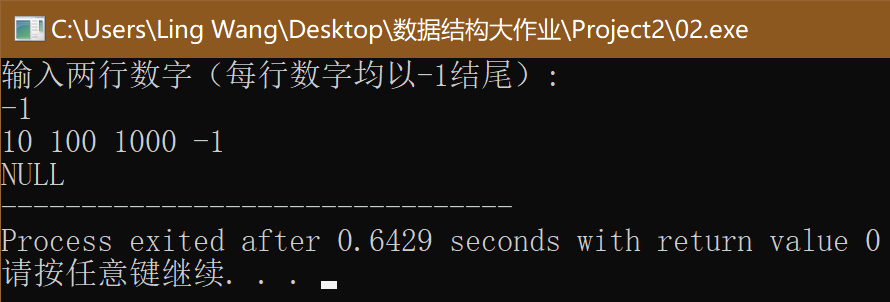
### 4.1.3 测试3—— 完全相交的情况



### 4.1.4 测试4—— 其中一个序列完全属于交集的情况



### 4.1.5 测试5—— 其中一个序列为空的情况



## 4.2 边界测试

### 4.2.1 测试1—— 两个链表为空

