项目说明文档

数据结构课程设计

——两个有序链表的交集

作 者 姓 名： 杨晶

学 号： 1854025

指 导 教 师： 张颖

学院、 专业： 软件学院 软件工程

同济大学

Tongji University

目 录

[1 分析](#_Toc495668153)

[1.1 背景分析](#_Toc495668154)

[1.2 功能分析](#_Toc495668155)

[2 设计](#_Toc495668156)

[2.1 数据结构设计](#_Toc495668157)

[2.2 类结构设计](#_Toc495668158)

[2.3 成员与操作设计](#_Toc495668159)

[2.4 系统设计](#_Toc495668160)

[3 实现](#_Toc495668161)

[3.1 搜索功能的实现](#_Toc495668162)

[3.1.1 搜索功能流程图](#_Toc495668163)

[3.1.2 搜索功能核心代码](#_Toc495668164)

[3.2 打印功能的实现](#_Toc495668166)

[3.2.1 打印功能流程图](#_Toc495668167)

[3.2.2 打印功能核心代码](#_Toc495668168)

[3.3 输入功能的实现](#_Toc495668170)

[3.3.1 输入功能流程图](#_Toc495668171)

[3.3.2 输入功能核心代码](#_Toc495668172)

[3.3.3 输入功能截图示例](#_Toc495668173)

[3.4 求交集功能的实现](#_Toc495668174)

[3.4.1 求交集功能流程图](#_Toc495668175)

[3.4.2 求交集功能核心代码](#_Toc495668176)

[3.4.3 求交集功能截屏示例](#_Toc495668177)

[3.5 总体系统的实现](#_Toc495668182)

[3.5.1 总体系统流程图](#_Toc495668183)

[3.5.2 总体系统核心代码](#_Toc495668184)

[3.5.3 总体系统截屏示例](#_Toc495668185)

[4 测试](#_Toc495668186)

[4.1 功能测试](#_Toc495668187)

[4.2 边界测试](#_Toc495668193)

[4.2.1 某个链表无输入数据](#_Toc495668194)

[4.2.2 交集数据在头/尾节点的情况](#_Toc495668195)

[4.2.3 交集为空](#_Toc495668196)

[4.3 出错测试](#_Toc495668197)

[4.3.1 异常输入](#_Toc495668198)

[4.3.2 不满足非降序正整数序列](#_Toc495668199)

# 1 分析

## 1.1 背景分析

全球知名咨询公司麦肯锡称：“数据，已经渗透到当今每一个行业和业务职能领域，成为重要的生产因素。人们对于海量数据的挖掘和运用，预示着新一波生产率增长和消费者盈余浪潮的到来”。

用数据说话，重视定量分析，也逐渐成为科学研究、企业经营、政府决策等过程着重考虑的问题。越来越多的人们意识到数据分析对经济发展的重要意义。数据分析部门利用数据资料丰富的优势，开展分析研究，透过事物的表面现象深入到事物的内在本质，由感性认识阶段上升到理性认识阶段，实现认识运动的质的飞跃，从而提示事物的现状及其内在联系和发展规律，不仅有利于我们客观全面地认识历史、现状及其发展趋势，促进数据管理水平的提高，而且有利于制定正确的决策和计划，以充分发挥数据分析促进管理、参与决策的重要作用。而在数据处理的过程中，我们常常面对的一种情况是要处理两组数据的交集。

综上所述，是否具有一个求非降序有序正整数链表的交集的能力，是小到团体企业是否能够生产更多的商业价值，规避或者减少风险带来的损失，提高数据质量的关键因素；也是一个国家社会是否能健全国防后备力量，立足于当今大数据时代的决定性指标之一。

因此，为了促进各个企业以及国家社会经济发展，我们可以试着写出一个求非降序有序链表交集的小程序。

## 1.2 功能分析

一个可以成功求出非降序有序链表的交集的程序应该可以实现的功能有，对两行由若干个正整数构成的非降序序列，并且可以进行正常的输入和输出。

其中输入用界定符“-1”来标志一段序列的结束，在输出时同样输出一个非降序有序链表，如果交集为空则输出“NULL”。

# 2 设计

## 2.1 数据结构设计

如上功能分析所述，该系统要求大量的增加和求交集等操作，按理来说用连续存储也可以实现，但考虑到占用连续存储空间时，如果预先进行存储分配，则当两组数据都很长时容易造成一部分空间长期闲置的问题。因此考虑使用链式存储，即链表数据结构。

## 2.2 类结构设计

经典的链表一般包括两个抽象数据类型——链表结点类与链表类，而两个类之间的耦合关系可以采用嵌套、继承等多种关系。

## 2.3 成员与操作设计

**链表结点类（Element）**

1. **class** Element {
2. **private**:
3. **int** data;
4. **public**:
5. Element\* next;
6. **int** getData() **const** { **return** data; }
7. //构造函数
8. Element(**int** a, Element\* link = nullptr) {
9. data = a;
10. next = link;
11. };
12. //在该元素后插入一个元素并返回这个元素
13. Element\* insertAfter(**int** i) {
14. next = **new** Element(i);
15. **return** next;
16. }
17. };

**链表类（List）**

1. **class** List {
2. **private**:
3. Element\* head;
4. //输入运算符重载
5. **friend** istream& operator>>(istream&, List&);
7. **public**:
8. //默认构造函数
9. List() { head = **new** Element(-1); }
10. Element\* getHead() **const** { **return** head; }
11. **bool** Search(**int** i);//搜索函数
12. **void** Display() **const**;//打印函数
13. };

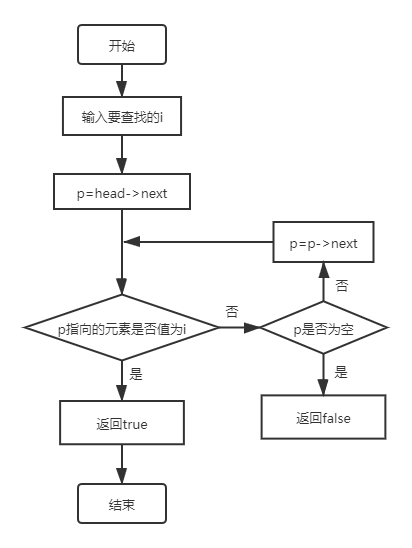
## 2.4 系统设计

在测试函数testLoop中，用户首先通过重载输入函数输入两个链表的数据，再通过我们设计的函数进行交集运算返回交集链表，调用交集链表的打印函数最终打印出结果。

# 3 实现

## 3.1链表类搜索功能的实现

### 3.1.1 搜索功能流程图

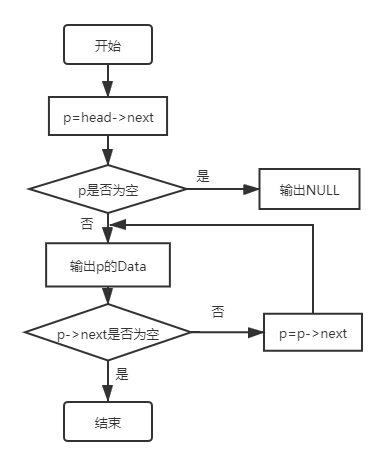


### 3.1.2 搜索功能核心代码

1. //寻找序列中值为i的元素
2. **bool** List::Search(**int** i) {
3. Element\* p = head;
4. **if** (i < 0) **return** **false**;
5. p = p->next;
6. **while** (p != nullptr) {
7. **if** (p->getData() == i)
8. **return** **true**;
9. **else** p = p->next;
10. }
11. **return** **false**;
12. }

## 3.2 打印功能的实现

### 3.2.1 打印功能流程图

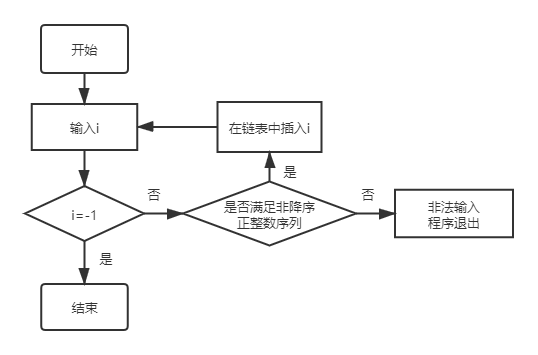


### 3.2.2 打印功能核心代码

1. //打印函数
2. **void** List::Display() **const** {
3. Element\* p = head;
4. **if** (p->next == nullptr) {
5. cout << "NULL";
6. **return**;
7. }
8. **while** (p->next != nullptr) {
9. cout << p->next->getData() << " ";
10. p = p->next;
11. }
12. cout << endl;
13. **return**;
14. }

## 3.3 链表输入功能的实现

### 3.3.1 输入功能流程图

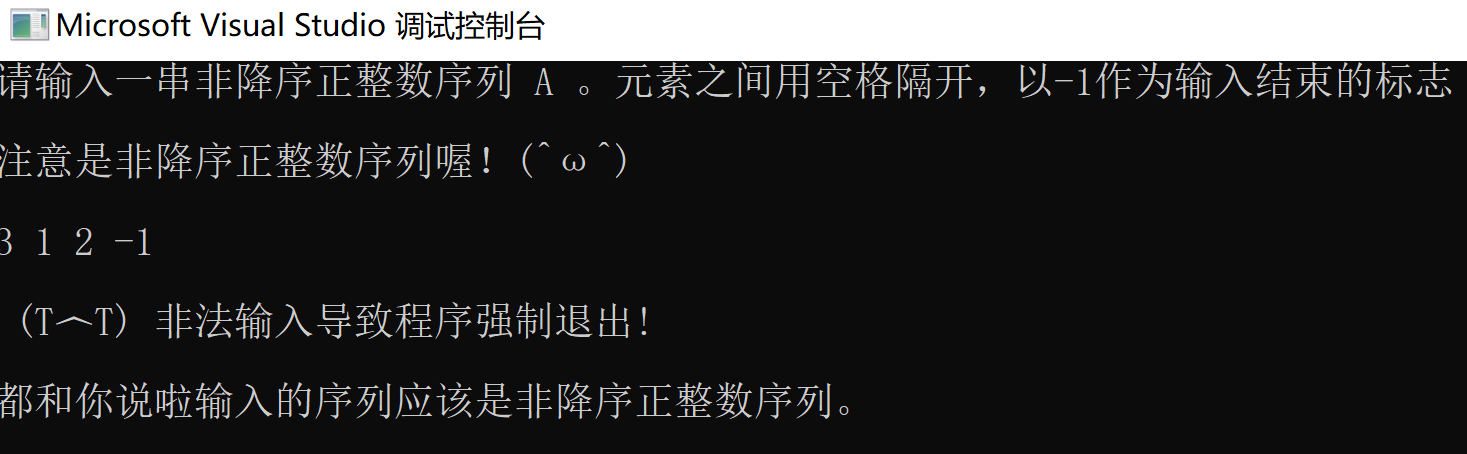


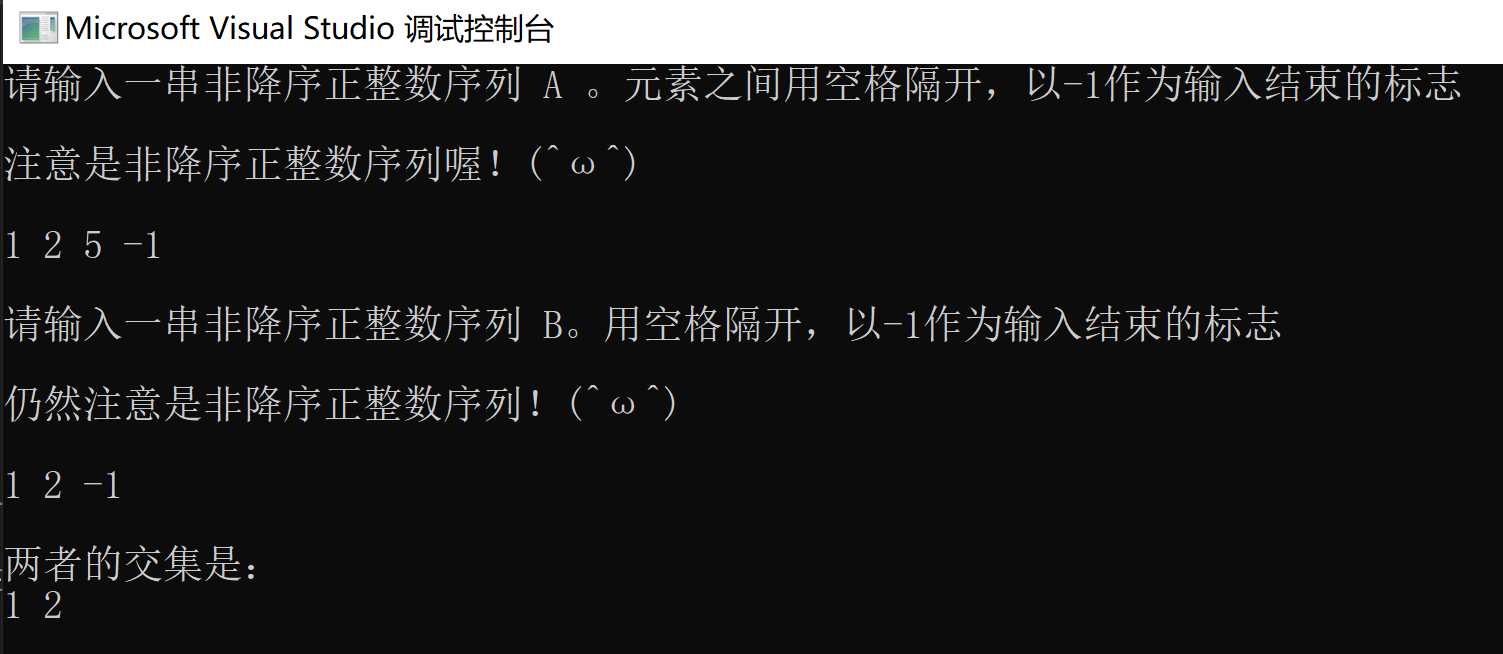
### 3.3.2 输入功能核心代码

1. istream& operator>>(istream& in, List& x) {
2. Element\* rear = x.getHead();
3. **while** (1) {
4. **int** i;
5. in >> i;
6. **if** (i == -1) **break**;
7. // 非降序，正，整数
8. **else** **if** (in.fail()||i != (**int**)i || i < 0 || i < rear->getData()) {
9. cout << endl << " (T︿T) 非法输入导致程序强制退出!"
10. << endl << endl << "都和你说啦输入的序列应该是非降序正整数序列。"
11. <<endl << endl;
12. system("pause");
13. exit(-1);
14. }
15. rear = rear->insertAfter(i);
16. }
17. **return** in;
18. }

### 3.3.3 输入功能截图示例

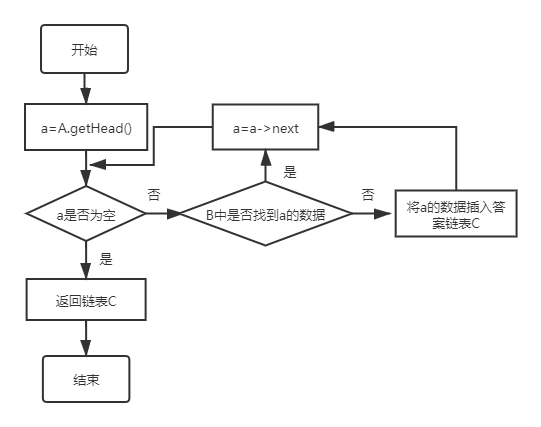
下图为调试情况，exe文件中遇非法输入直接退出。





## 3.4 求交集功能的实现

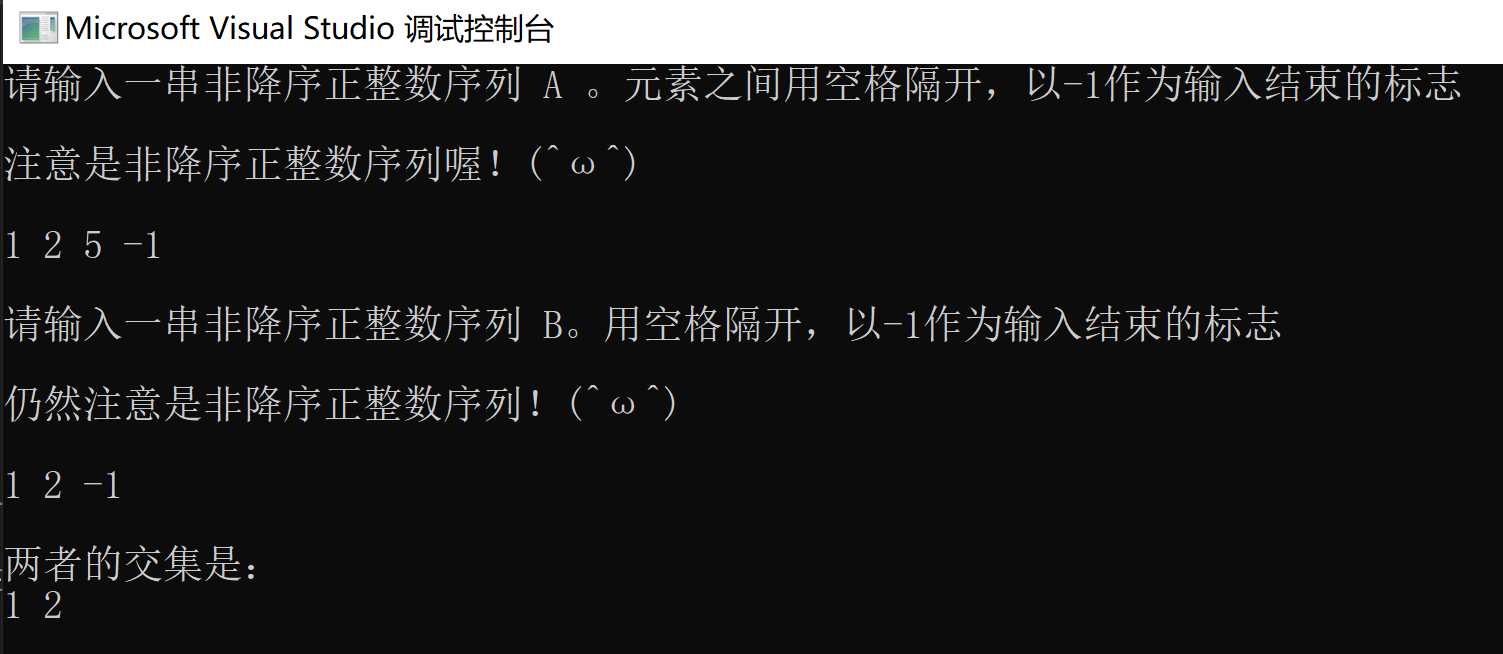
### 3.4.1 求交集功能流程图

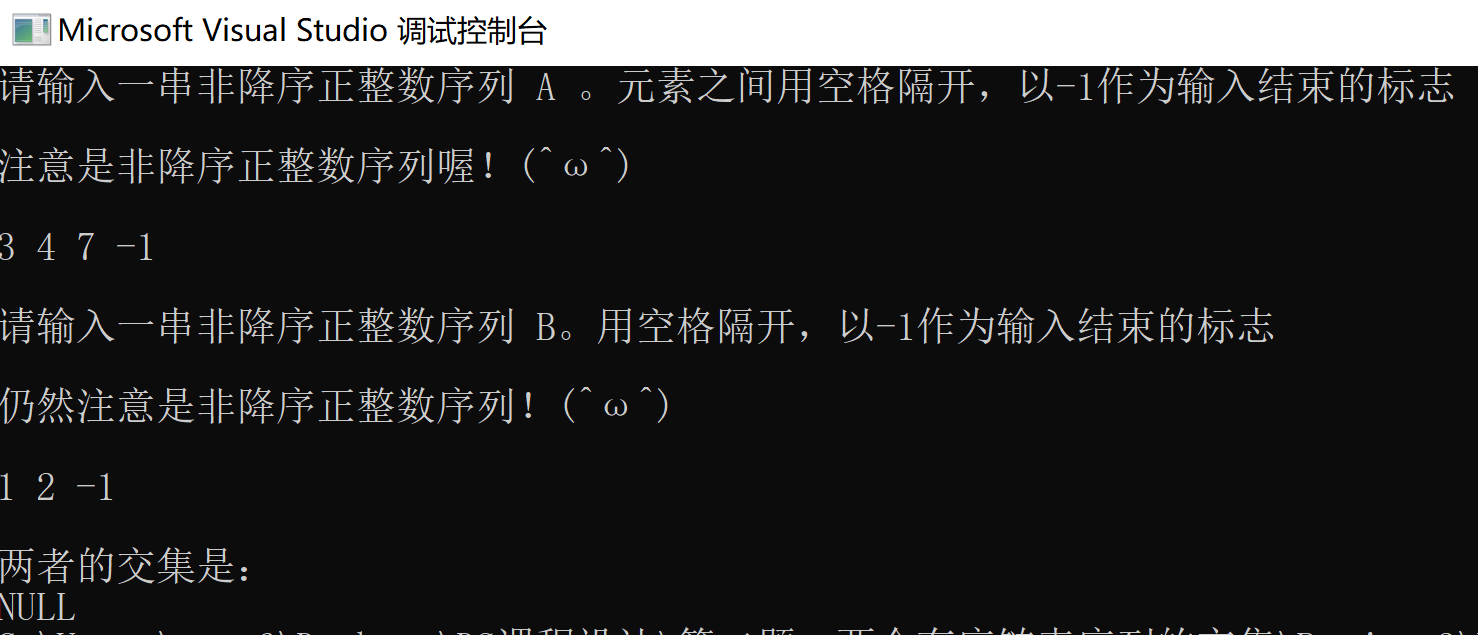


### 3.4.2 求交集功能核心代码

1. //两个链表求交集
2. List Intersection(List& A, List& B) {
3. Element\* a = A.getHead();
4. List C; //结果存储在C中
5. Element\* c = C.getHead();
7. **while** (a->next != nullptr) {
8. a = a->next;
9. **if** (B.Search(a->getData()))
10. c = c->insertAfter(a->getData());
11. }
13. **return** C;
14. }

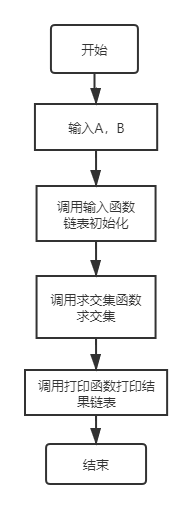
### 3.4.3 求交集功能截屏示例





## 3.5 总体系统的实现

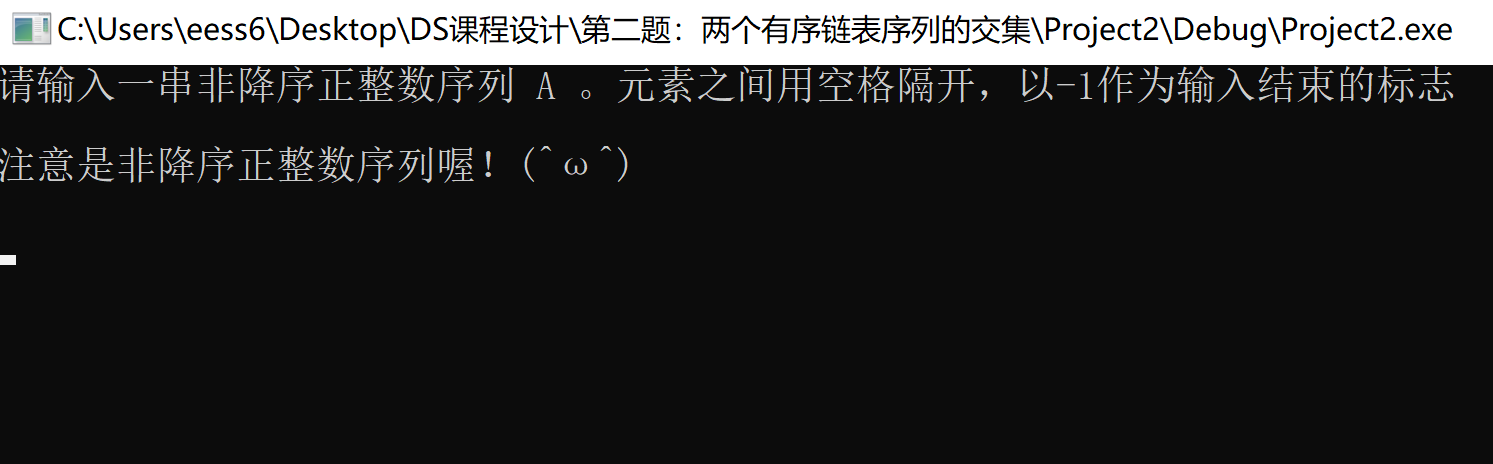
### 3.5.1 总体系统流程图

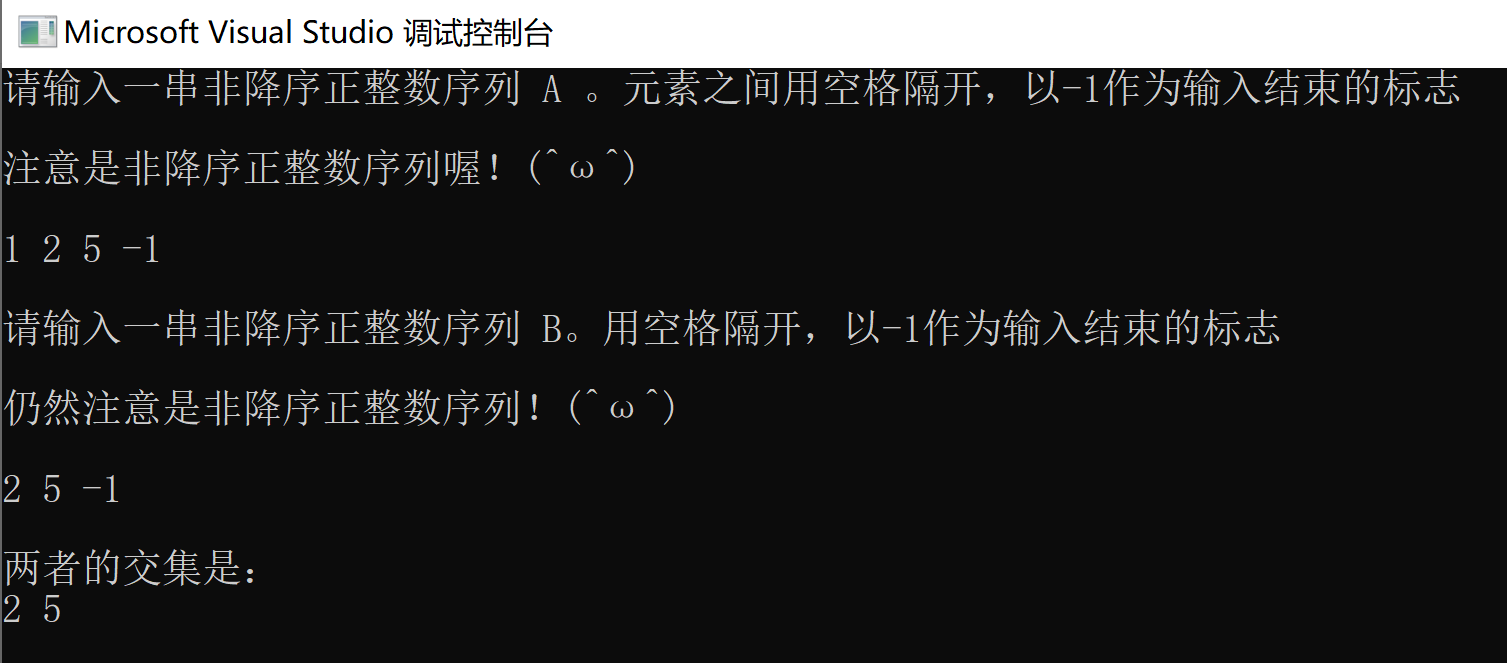


### 3.5.2 总体系统核心代码

1. **void** testLoop() {
2. List A;
3. List B;
4. cout << "请输入一串非降序正整数序列 A 。" <<
5. "元素之间用空格隔开，以-1作为输入结束的标志" << endl << endl;
6. cout << "注意是非降序正整数序列喔！(^ω^)" << endl << endl;
7. cin >> A;
8. cout << endl << "请输入一串非降序正整数序列 B。" <<
9. "用空格隔开，以-1作为输入结束的标志" << endl << endl;
10. cout << "仍然注意是非降序正整数序列！(^ω^)" << endl << endl;
11. cin >> B;
13. cout << endl << "两者的交集是：" << endl;
14. Intersection(A, B).Display();
15. }
17. **int** main() {
18. testLoop();
19. system("pause");
20. **return** 0;
21. }

### 3.5.3 总体系统截屏示例





# 4 测试

## 4.1 功能测试

**测试用例**：

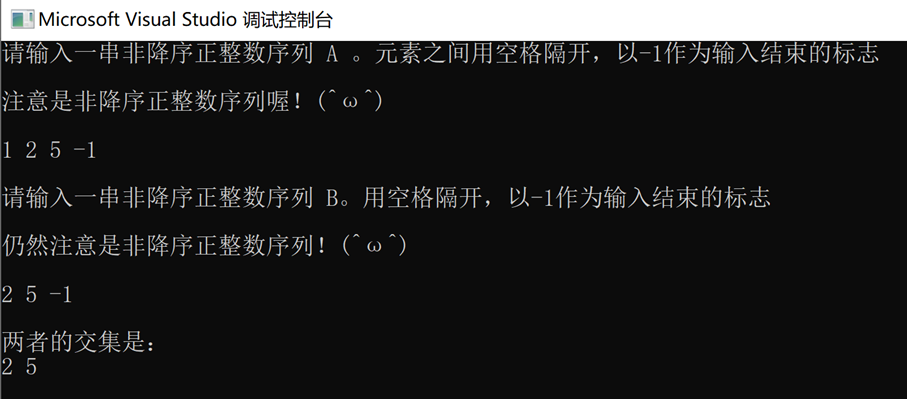
A：1 2 5

B：2 5

**预期结果**：

2 5

**实验结果**



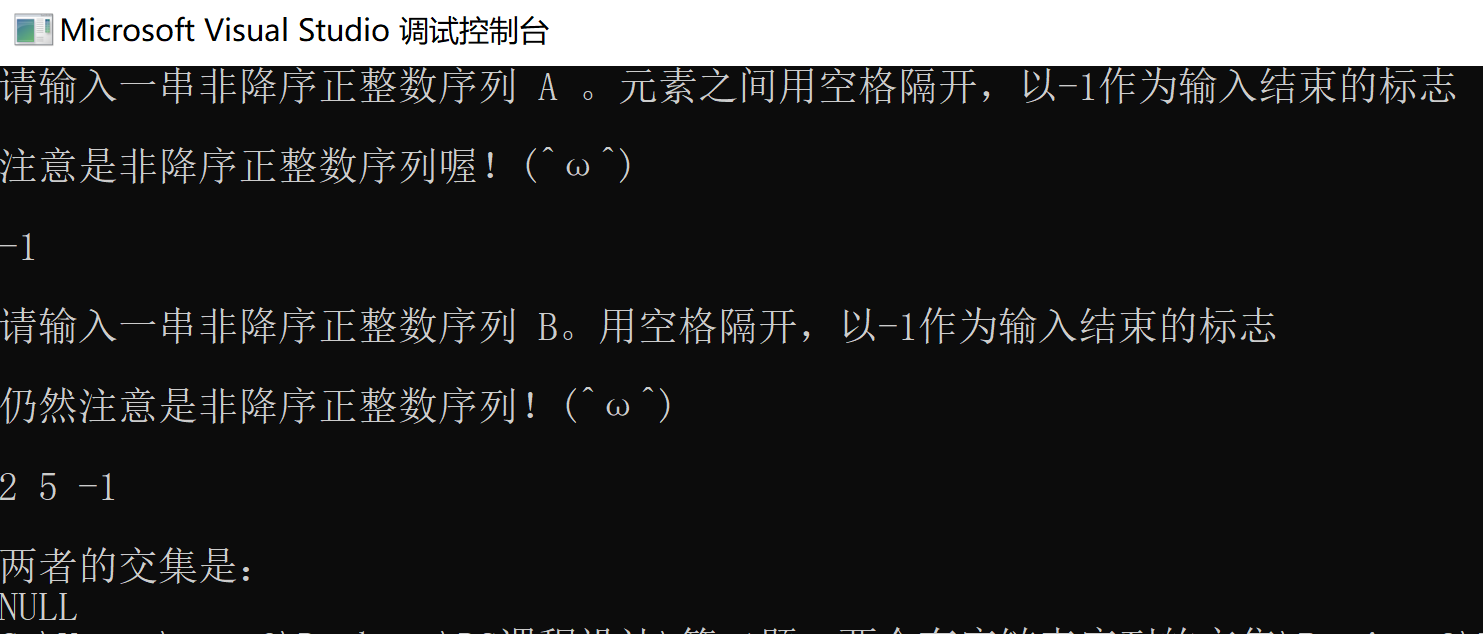
## 4.2 边界测试

### 4.2.1 某个链表无输入数据

**测试用例：**初始无输入数据

**预期结果：输出NULL**。

**实验结果：**



### 4.2.2 交集数据在头/尾节点

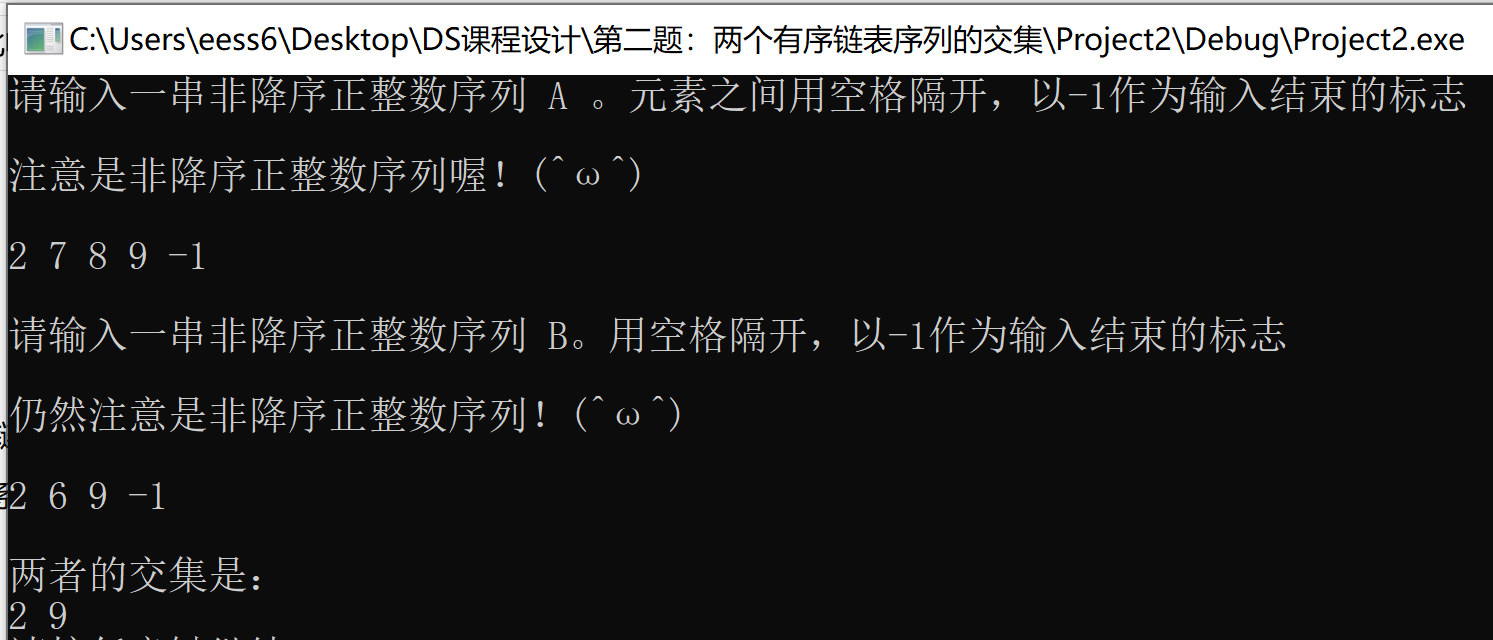
**测试用例：**

A：2 7 8 9 -1

B：2 6 9 -1

**预期结果：**2 9

**实验结果：**



### 4.2.3 交集为空

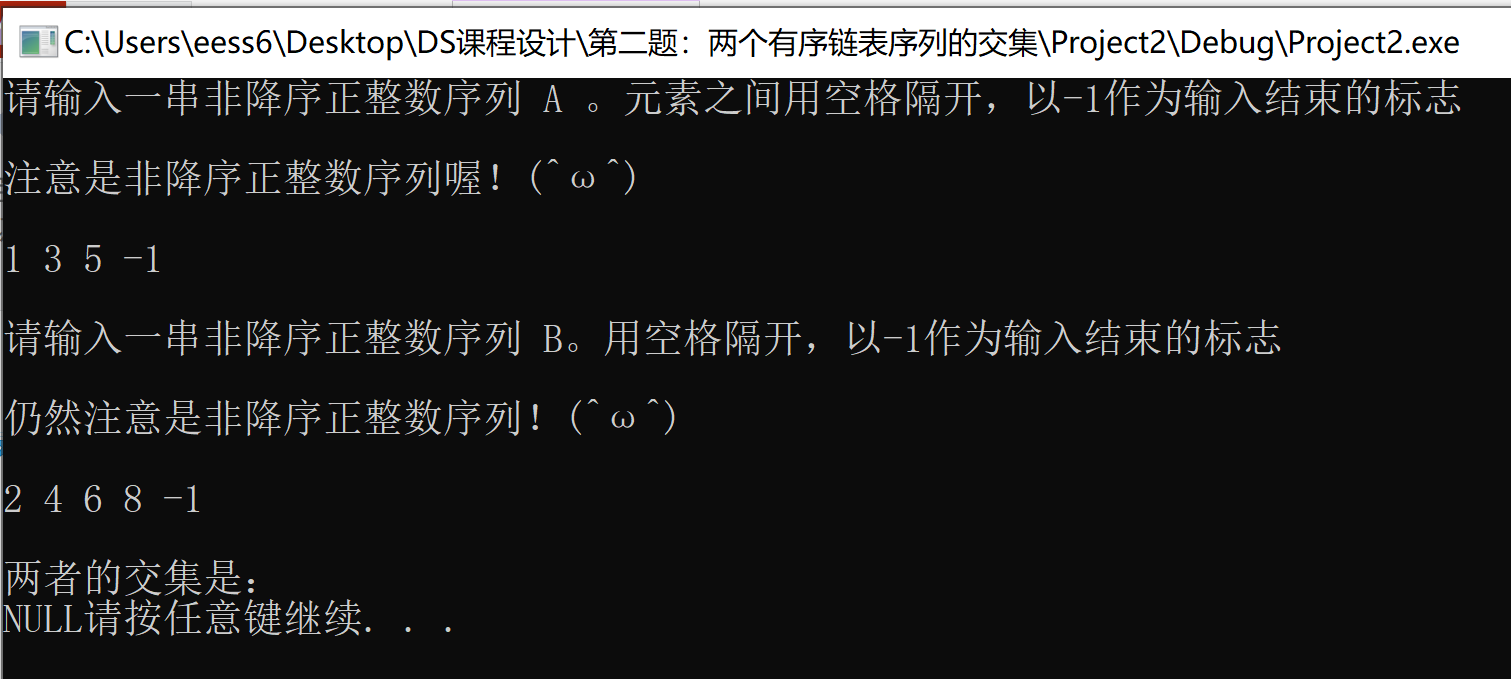
**测试用例：**

A：1 3 5 -1

B：2 4 6 8 -1

**预期结果：输出NULL**。

**实验结果：**



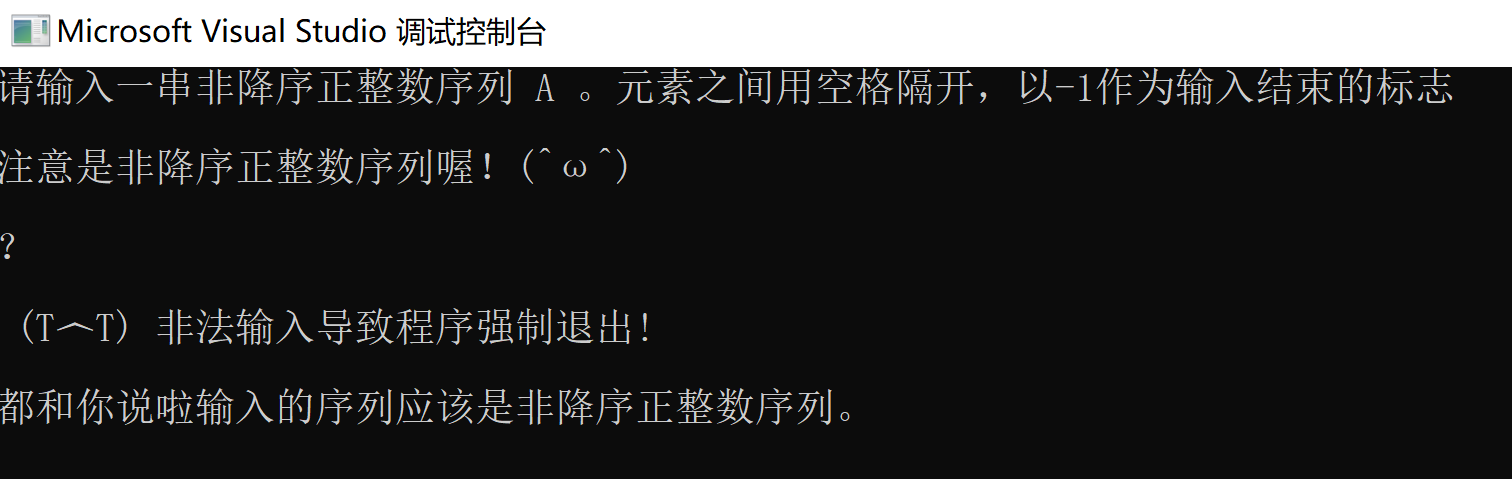
## 4.3 出错测试

### 4.3.1 异常输入

**测试用例：**输入非数字符号，如；。？

**预期结果：**程序直接启动退出步骤。

**实验结果：**

****

### 4.3.2 不满足非降序正整数序列

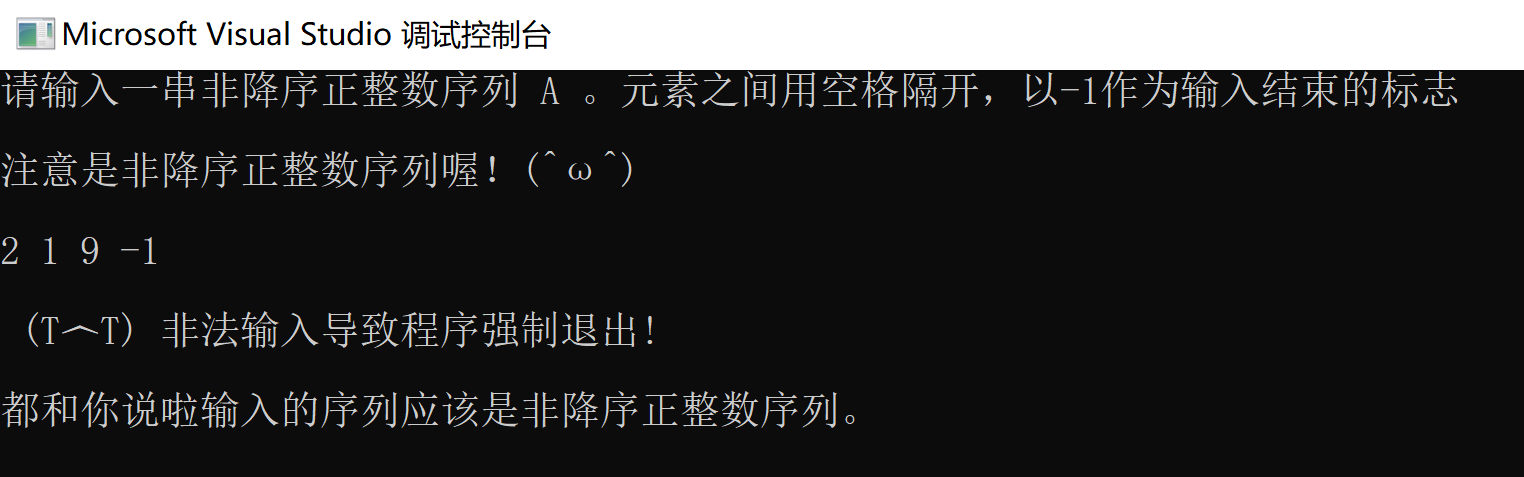
**测试用例：**

A：2 1 -1

B：-1

**预期结果：**程序直接启动退出步骤。

**实验结果：**

****