项目说明文档

数据结构课程设计

——两个有序链表序列的交集

作 者 姓 名： 李圣

学 号： 1952723

指 导 教 师： 张颖

学院、 专业： 软件学院 软件工程

同济大学

Tongji Universit

目录

[1 分析 1](#_Toc12039)

[1.1 功能分析 1](#_Toc14284)

[2 设计 1](#_Toc13788)

[2.1 文件结构设计 1](#_Toc10563)

[2.2 成员与函数设计 1](#_Toc8762)

[2.3 系统设计 4](#_Toc10749)

[3 实现 5](#_Toc13867)

[3.1 向后插入（insert）实现 5](#_Toc29931)

[3.1.1 向后插入流程图 5](#_Toc26050)

[3.1.2 向后插入核心代码 6](#_Toc18221)

[3.2 输入（input）实现 6](#_Toc5513)

[3.2.1 输入流程图 6](#_Toc26446)

[3.2.2 输入核心代码 7](#_Toc15719)

[3.3 求交集的实现 8](#_Toc4882)

[3.3.1 求交集流程图 8](#_Toc7072)

[3.3.2 求交集核心代码 9](#_Toc28919)

[3.4 总体系统的实现 10](#_Toc16603)

[3.4.1 总体系统流程图 10](#_Toc19110)

[3.4.2 总体系统核心代码 11](#_Toc17746)

[3.4.3 总体系统截屏示例 11](#_Toc16343)

[4 测试 11](#_Toc14493)

[4.1 鲁棒性测试 11](#_Toc23933)

[4.1.1 正常案例测试 11](#_Toc10421)

[4.1.2 负数测试 12](#_Toc9015)

[4.1.3 升序测试 12](#_Toc16926)

[4.1.4 输入非法测试 13](#_Toc4437)

# 分析

## 功能分析

项目要求使用链表对两个非降序正序列进行寻找交集。考虑到链表使用的广泛性以及普遍性，觉得开发一个自己的带头结点的双向循环类模板MyList置于MyList.h中。因此，项目除了源文件还包括了一个头文件MyList.h。

此外为了熟练使用Linux，该项目使用Linux进行调试开发。

# 设计

## 文件结构设计

考虑到List的使用，该项目包含两个文件，如下：

MyList.h

main.cpp

其中，main.cpp中调用MyList.h

## 成员与函数设计

**MyListNode类：**

template<typename val\_type>

using node = MyListNode;

**私有成员：**

val\_type \_val;

node\* \_prev;

node\* \_next;

void deleteAll(node\* head);//删除后续所有节点

**公有成员：**

MyListNode();

MyListNode(const node& node);

MyListNode(const val\_type& val);

~MyListNode();

node\* prev()const;

node\* next()const;

**MyListIterator类：**

template<typename node\_type, typename val\_type>

using iterator = MyListIterator;

using node = node\_type;

using pointer = node\_type\*;

using reference = val\_type&;

**私有成员：**

pointer \_ptr;

**公有成员：**

MyListIterator(pointer ptr = nullptr);

bool operator==(const iterator& it);

bool operator!=(const iterator& it);

iterator& operator++();

iterator& operator++(int);

iterator& operator--();

iterator& operator--(int);

reference operator\*();

pointer operator->();

**MyList类：**

template<typename val\_type>

using list = MyList;

using node = MyListNode<val\_type>;

using iterator = MyListIterator<node, val\_type>;

**私有成员：**

int \_length;

node\* \_head;

node\* \_tail;

**公有成员：**

//构造

MyList();

MyList(MyList<val\_type>& l2);

//析构

~MyList()

iterator begin();

iterator end();

//清空表，留头节点

void clear();

//长度

int length();

int size();

void insert(iterator position, const val\_type& val);

val\_type remove(iterator position);

void push\_back(const val\_type& val);

val\_type pop\_back();

list& operator= (list& l2);

list& operator= (list&& l2);

## 系统设计

**函数：**

void cinFail()；

void input(MyList<long long> &list)；

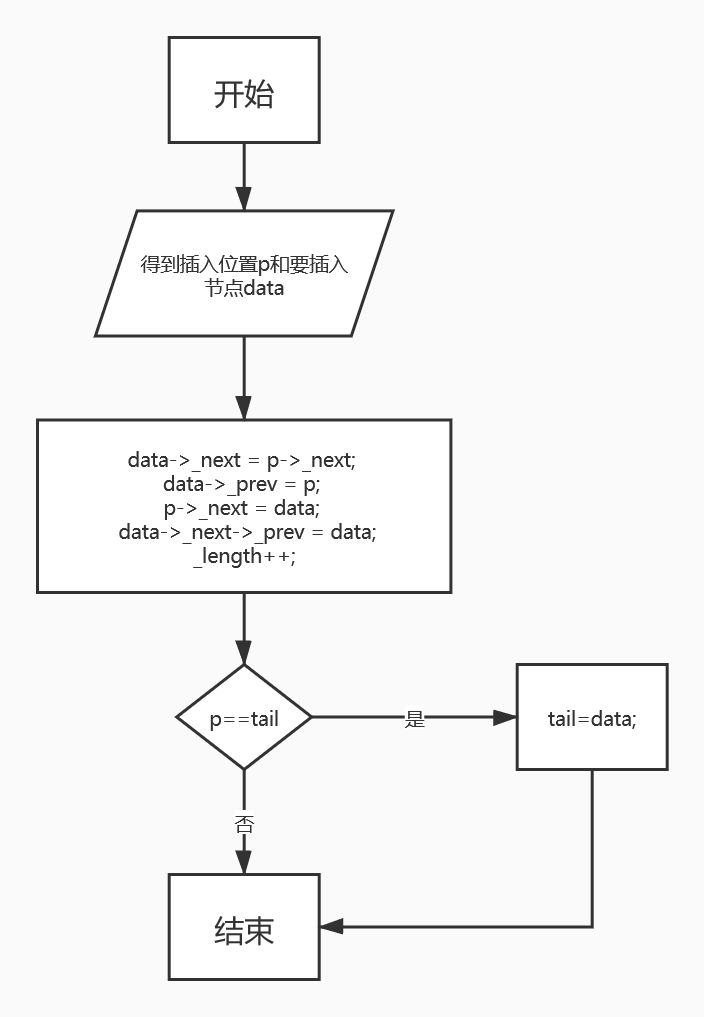
MyList<long long> intersection(MyList<long long> l1, MyList<long long> l2)；

在main函数中调用这几个函数以实现功能。

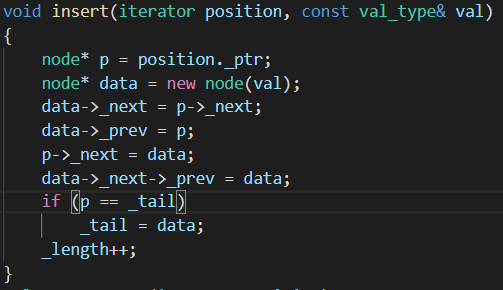
# 实现

## 向后插入（insert）实现

### 向后插入流程图

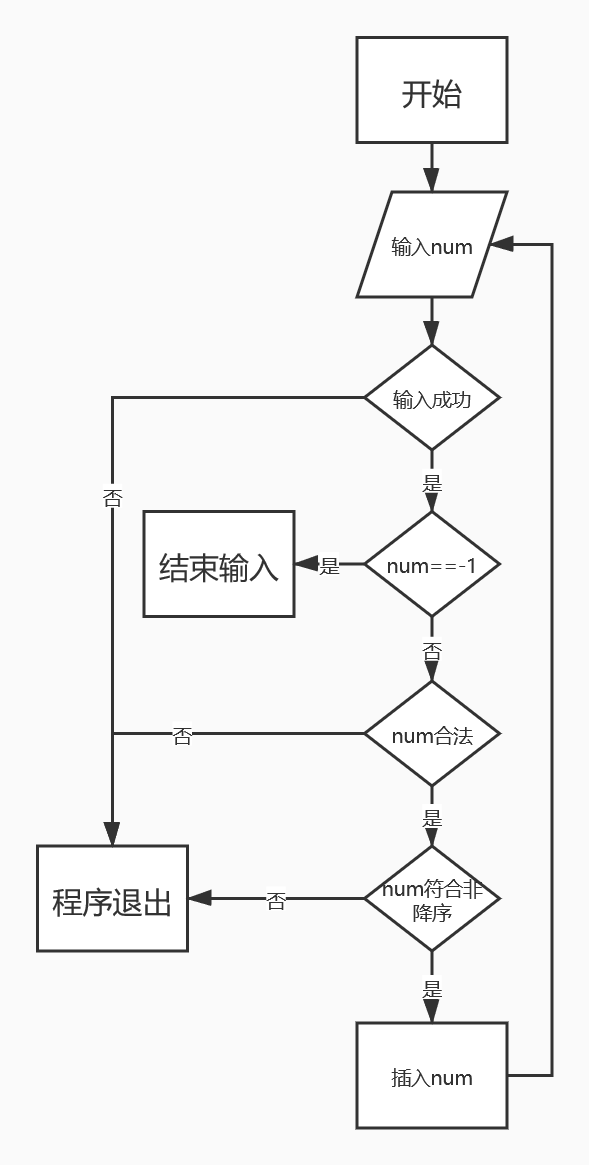


### 向后插入核心代码



## 输入（input）实现

### 输入流程图

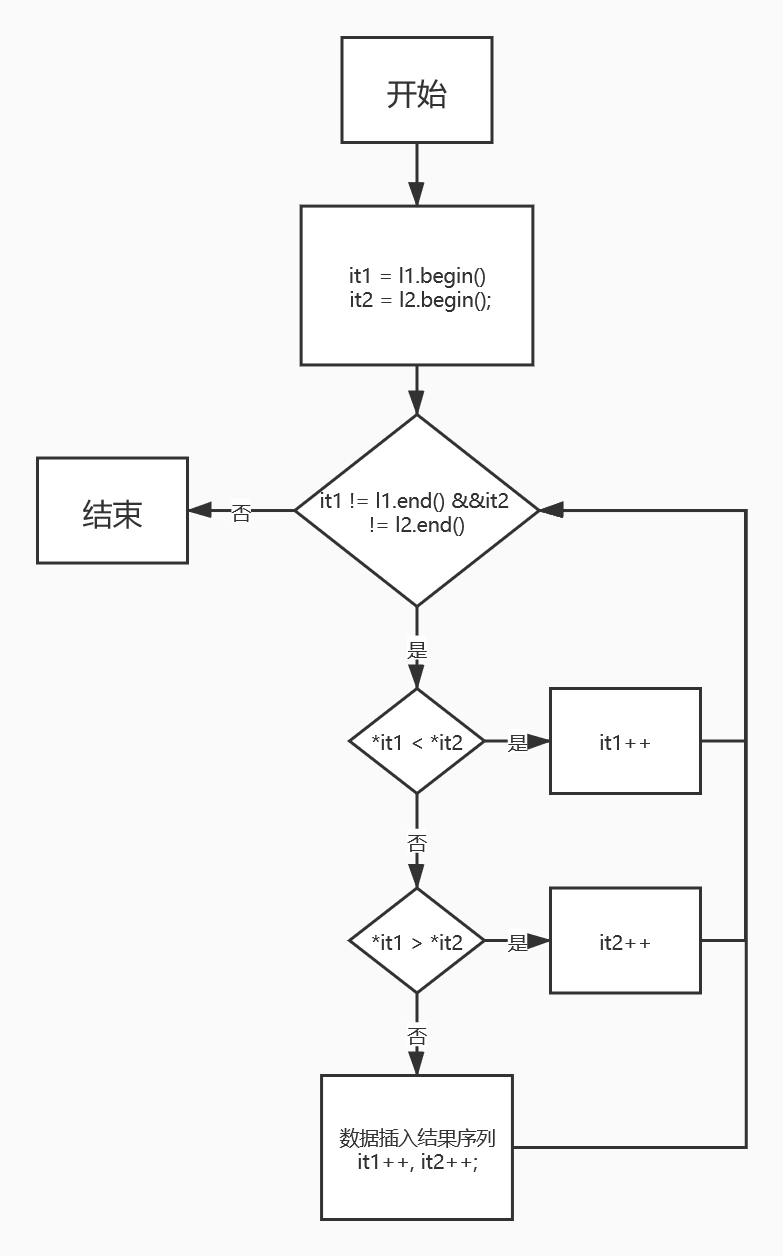


### 输入核心代码

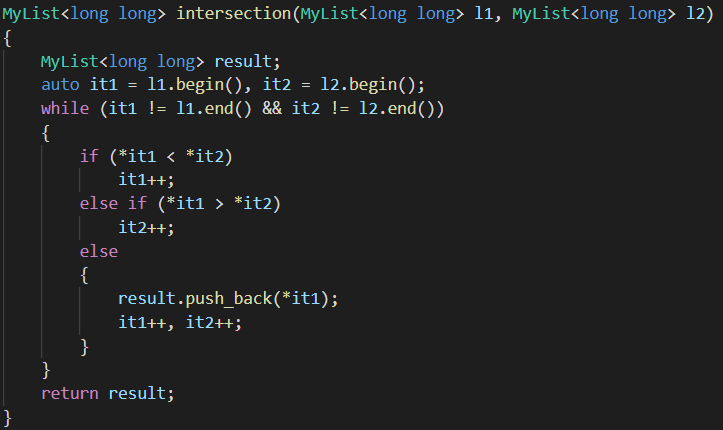


## 求交集的实现

### 求交集流程图

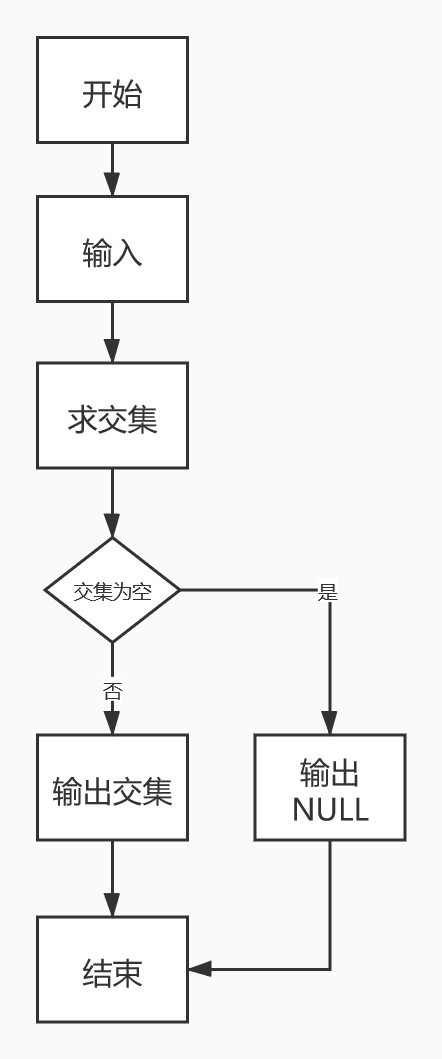


### 求交集核心代码

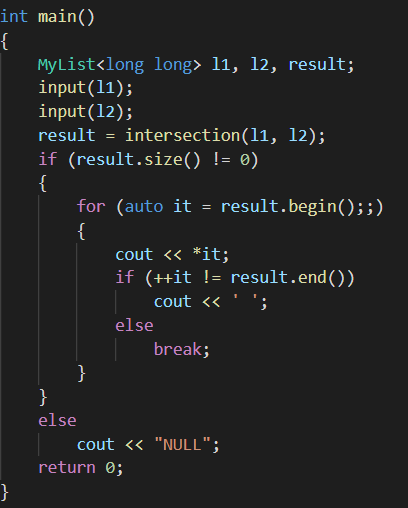


## 总体系统的实现

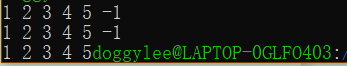
### 总体系统流程图



### 总体系统核心代码



### 总体系统截屏示例



# 测试

## 鲁棒性测试

### 正常案例测试

**测试用例：**一般情况

**预期结果：**对应的正确结果。

**实验结果：**



**测试用例：**交集为空的情况

**预期结果：**对应的正确结果。

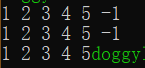
**实验结果：**



**测试用例：**完全相交的情况

**预期结果：**对应的正确结果。

**实验结果：**



**测试用例：**其中一个序列完全属于交集的情况

**预期结果：**对应的正确结果。

**实验结果：**



**测试用例：**其中一个序列为空的情况

**预期结果：**对应的正确结果。

**实验结果：**



### 负数测试

**测试用例：**输入非-1的负数。

**预期结果：**程序给出错误信息，并直接退出程序。

**实验结果：**

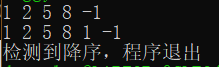


### 升序测试

**测试用例：**序列不是非降序。

**预期结果：**程序给出错误信息，并直接退出程序。

**实验结果：**



### 输入非法测试

**测试用例：**输入非法或超出long long界限。

**预期结果：**程序给出错误信息，并直接退出程序。

**实验结果：**



