项目5说明文档

数据结构课程设计

——银行业务

作 者 姓 名： 谭欢秘

学 号： 1954185

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 1](#_Toc60205681)

[1.1 背景分析 1](#_Toc60205682)

[1.2 功能分析 1](#_Toc60205683)

[2 设计 2](#_Toc60205684)

[2.1 数据结构设计 2](#_Toc60205685)

[2.2 类结构设计 2](#_Toc60205686)

[2.3 类成员与操作设计 2](#_Toc60205687)

[2.3.1 Queue类成员 2](#_Toc60205688)

[2.3.2 Counter类成员 3](#_Toc60205689)

[2.4 主函数设计 3](#_Toc60205690)

[3 实现 4](#_Toc60205691)

[3.1 Queue类一些非核心功能的实现 4](#_Toc60205692)

[3.1.1 getSize()函数 4](#_Toc60205693)

[3.1.2 isEmpty()函数 4](#_Toc60205694)

[3.1.3 makeEmpty() 5](#_Toc60205695)

[3.1.4 getFront() 5](#_Toc60205696)

[3.1.5 showQueue() 5](#_Toc60205697)

[3.2 Queue类元素进队功能的实现 6](#_Toc60205698)

[3.2.1进队功能流程图 6](#_Toc60205699)

[3.2.2进队功能核心代码 6](#_Toc60205700)

[3.3 Queue类元素出队功能的实现 7](#_Toc60205701)

[3.3.1元素出队功能流程图 7](#_Toc60205702)

[3.3.2 元素出队功能核心代码 7](#_Toc60205703)

[3.4 Counter类中初始化功能的实现 8](#_Toc60205704)

[3.4.1 initialize()功能流程图 8](#_Toc60205705)

[3.4.2 initialize()功能核心代码 8](#_Toc60205706)

[3.5 Queue类中output()功能的实现 9](#_Toc60205707)

[3.5.1 output()功能流程图 9](#_Toc60205708)

[3.5.2 output()功能核心代码 9](#_Toc60205709)

[3.9 主函数的实现 11](#_Toc60205710)

[3.9.1 主函数流程图 11](#_Toc60205711)

[3.9.2 总体系统核心代码 11](#_Toc60205712)

[3.9.3 总体系统截屏示例 11](#_Toc60205713)

[4 测试 12](#_Toc60205714)

[4.1 功能测试 12](#_Toc60205715)

[4.1.1 初始化功能测试 12](#_Toc60205716)

[4.1.2 按顺序输出序号功能测试 13](#_Toc60205717)

[4.2 边界测试 14](#_Toc60205718)

[4.2.1 总人数大小非法 14](#_Toc60205719)

[4.2.2 顾客序号数据非法 15](#_Toc60205720)

[5 总结 15](#_Toc60205721)

[5.1 遇到的错误 15](#_Toc60205722)

[5.1.1 xx函数中 15](#_Toc60205723)

[5.2 改进与优化 16](#_Toc60205724)

[5.3 项目心得 16](#_Toc60205725)

# 1 分析

## 背景分析

本题目中某银行有A，B两个业务窗口，且处理业务的速度不一样，其中A窗口处理速度是B窗口的2倍----即当A窗口每处理完2个顾客是，B窗口处理完1个顾客。给定到达银行的顾客序列，期中编号为奇数的顾客需要到A窗口办理业务，为偶数的顾客则去B窗口。不同窗口同时处理完2个顾客时，A窗口的顾客优先输出。本题目需要按照业务完成的顺序输出顾客序列。

可以看出，先忽略掉A、B窗口之间处理速度和输出优先级的小差异，整体上本题目的特征是满足FIFO（先进先出）的，所以应该采用的数据结构是队列。我们只需声明两个队列A、B，将奇数顾客放入A队列、偶数顾客放入B队列，然后按照两个窗口处理速度进行输出即可。最终的程序将接受顾客总人数、顾客编号的输入，并按照要求输出业务完成的顾客序列。

## 功能分析

首先是需要完成顾客信息初始化操作。在输入顾客总人数、顾客序号时，一边输入、一边将奇数偶数序号分开，分别放入A队列、B队列中。其次便是输出，有些类似于取交集的操作，只不过A中每次固定先让两个元素出队并打印出来、再是B中每次固定让一个元素出队并打印。若A、B中有一个队列已空，则直接让另一个队列中的元素逐一出队并输出。直到两个队列均为空时，结束输出。

综上所述，本项目应该具有输入顾客信息、按柜台处理顺序输出序号的功能。

# 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、删除、修改操作，而链表进行增加、删除等操作十分简便，因此考虑使用链表数据结构。同时，处理的元素相对而言是先进先出、后进后出的，所以采用队列的数据结构。

## 2.2 类结构设计

本项目主要有两个类，队列Queue<typename Type>类与银行柜台Counter类。其中Queue类作为链表类，私有成员里嵌套定义了链表的节点类QueueNode类。

## 2.3 类成员与操作设计

### 2.3.1 Queue类成员

--私有成员部分：

--公有成员部分：

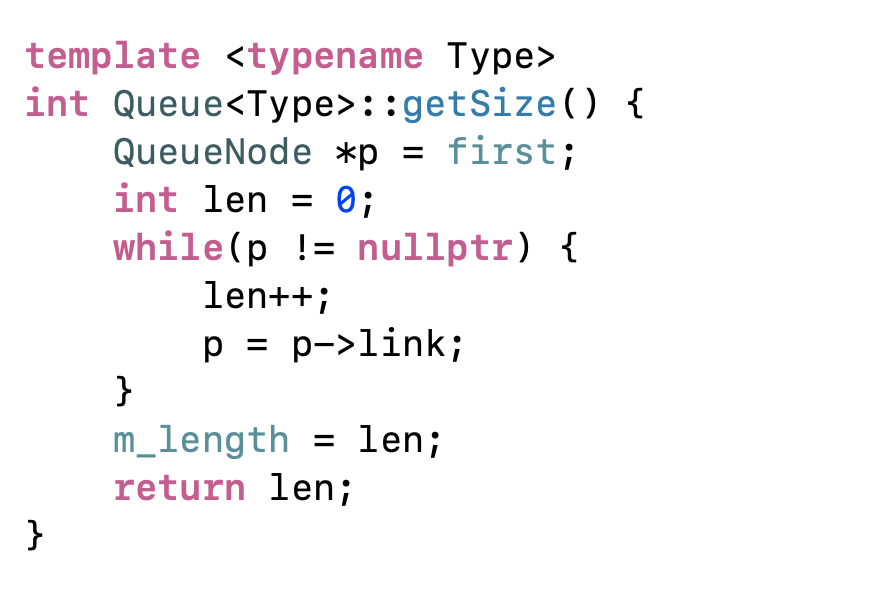
### 2.3.2 Counter类成员

## 2.4 主函数设计

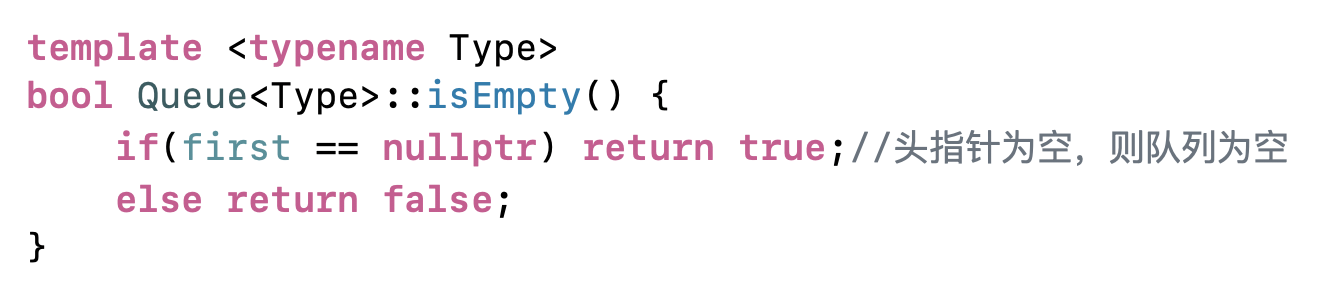
声明一个Counter类对象 bank。首先调用bank.initialize()进行初始化、输入银行顾客人数、序号信息；然后调用bank.output()进行输出处理完成的顾客序号。

3 实现

3.1 Queue类一些非核心功能的实现

3.1.1 getSize()函数

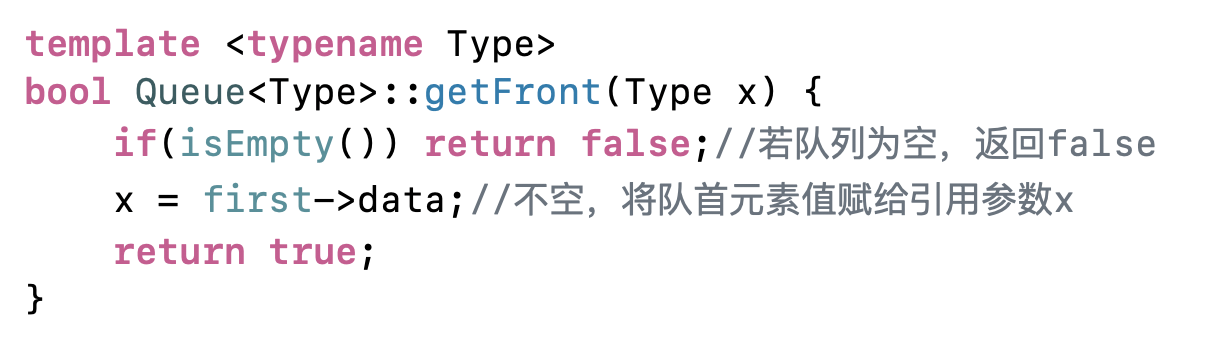
从队首元素开始，让指针p不断后移，每后移一次len++。直到p指向nullptr时，循环结束，将len的值赋给Queue类的私有成员m\_length。每次执行了出队、进队操作后，可以利用该函数进行重新统计队中元素个数；同时将len值设为函数return的值，在保证了Queue类封装性的前提下也让其他类可以读取栈中元素的总个数。

3.1.2 isEmpty()函数

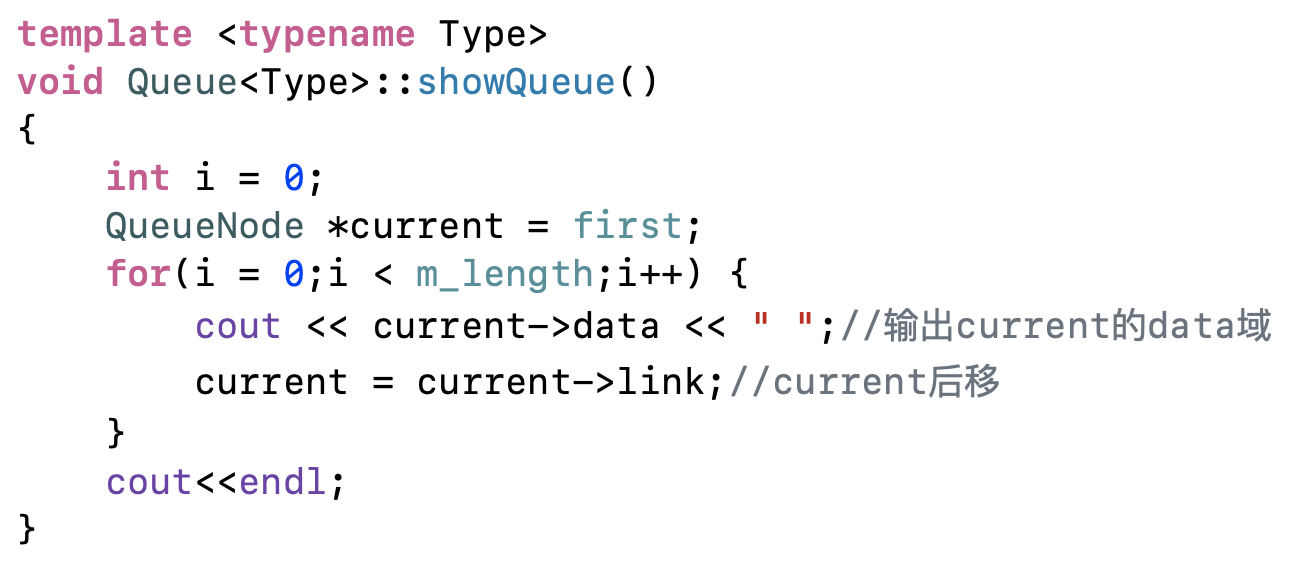
若队首元素指针为空，则表示队空、返回true；否则返回false。

3.1.3 makeEmpty()

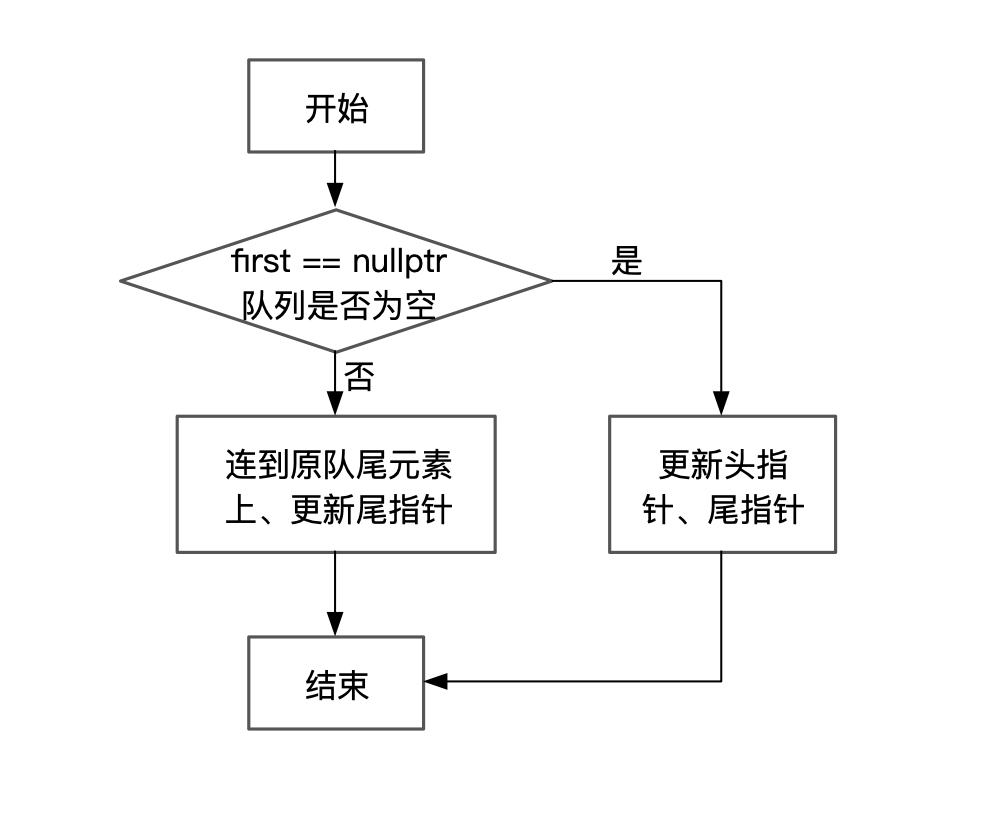
逐一将所有元素deQueue出队即可。

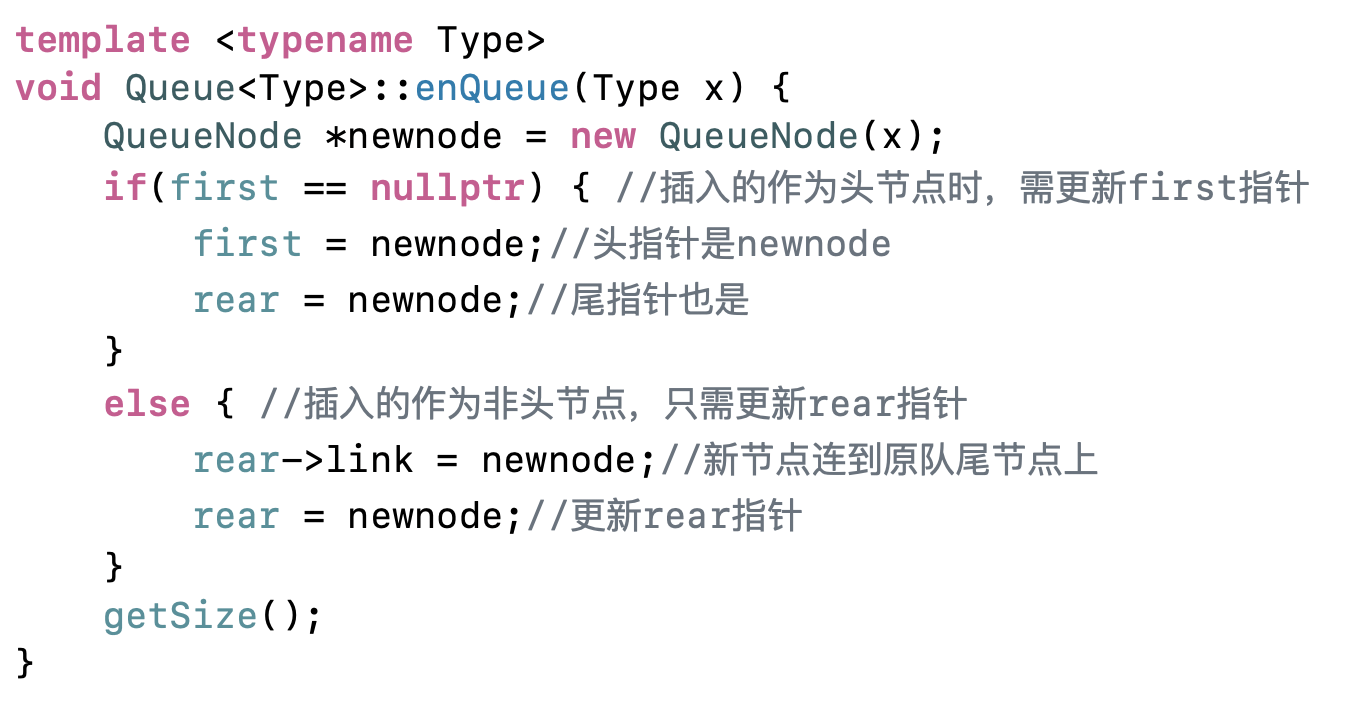
3.1.4 getFront()

首先判断队列是否为空。若不为空，则取头指针first的data域赋给x，通过引用传递给参数。

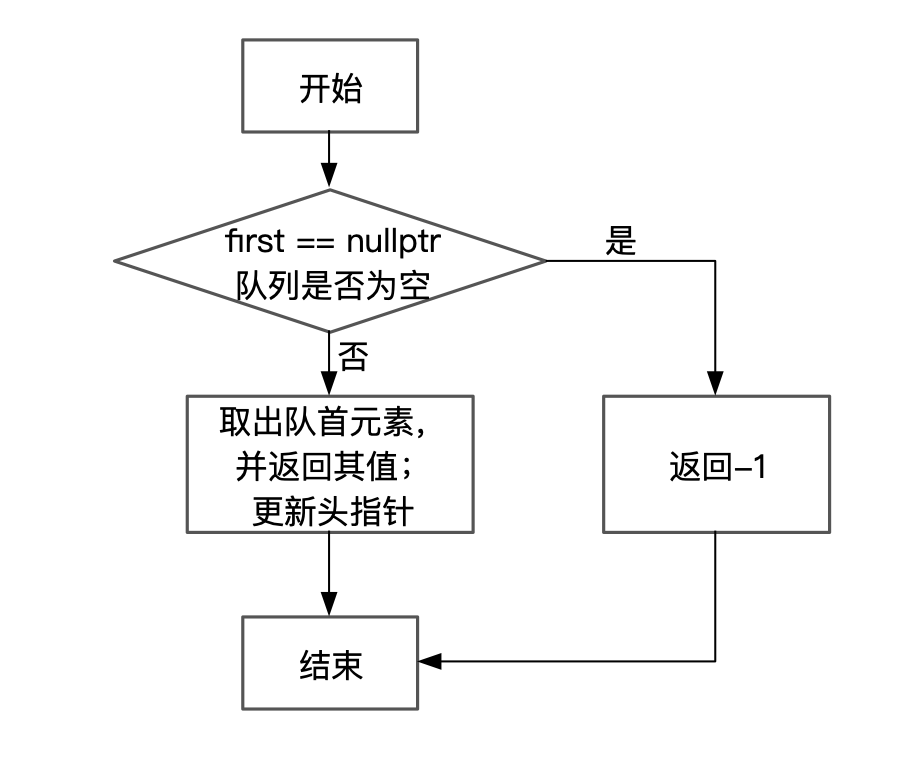
3.1.5 showQueue()

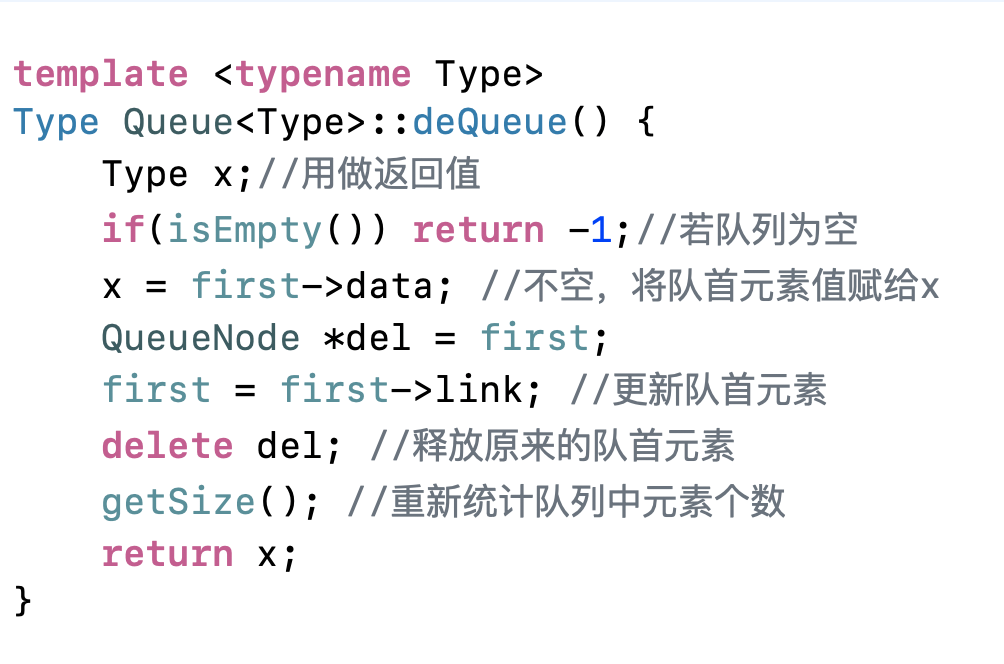
3.2 Queue类元素进队功能的实现

3.2.1进队功能流程图

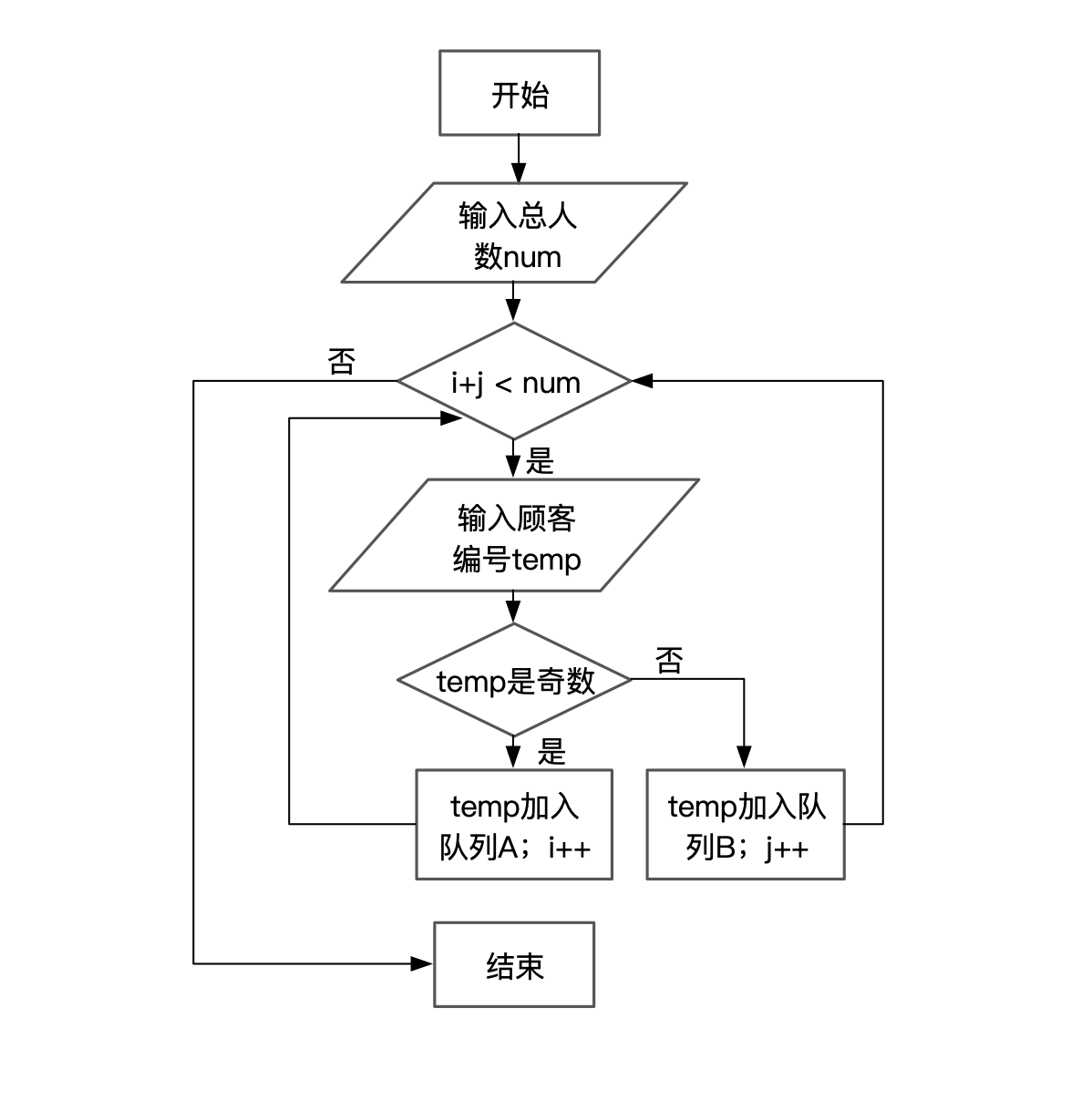
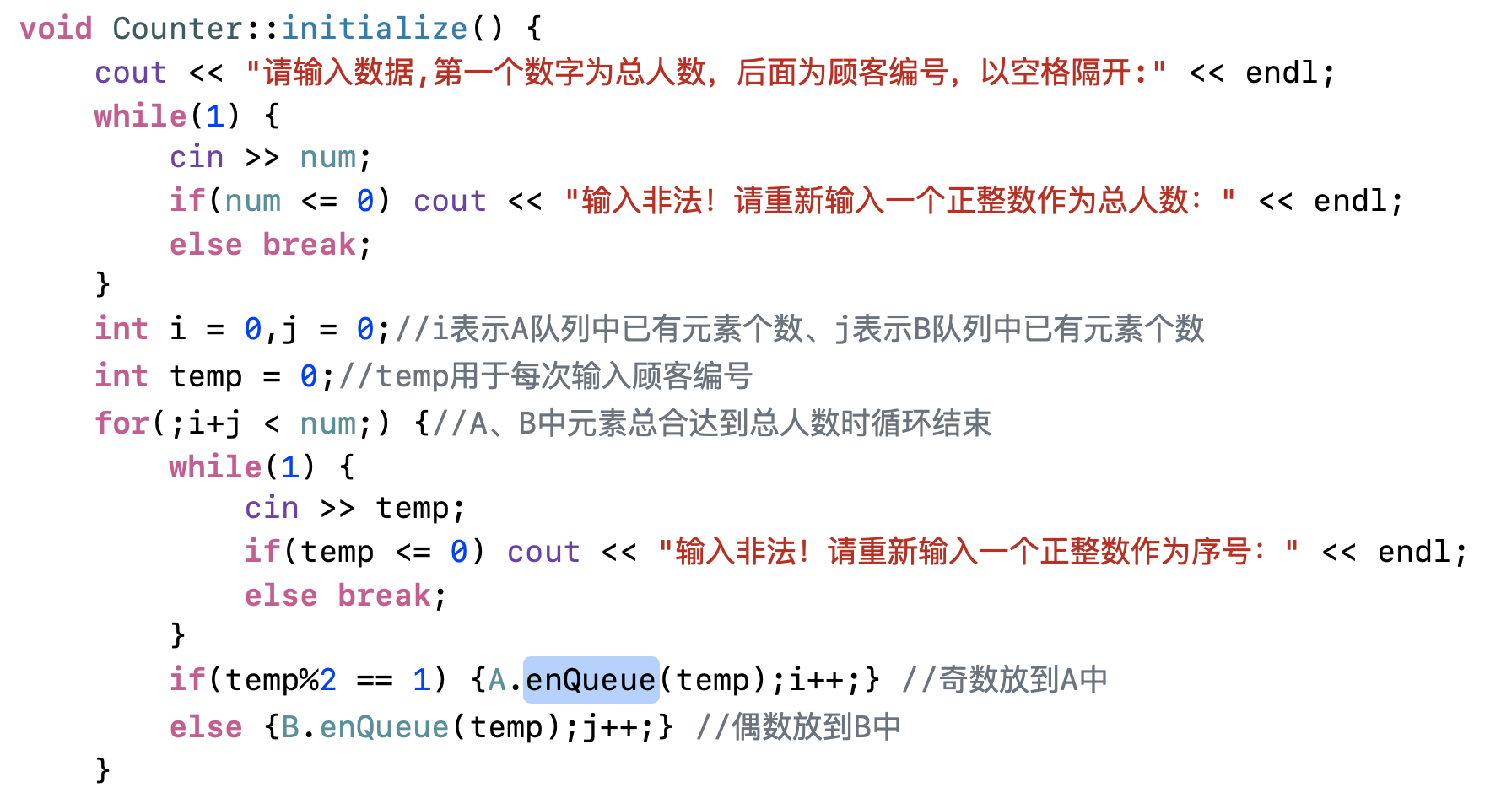
3.2.2进队功能核心代码

3.3 Queue类元素出队功能的实现

3.3.1元素出队功能流程图

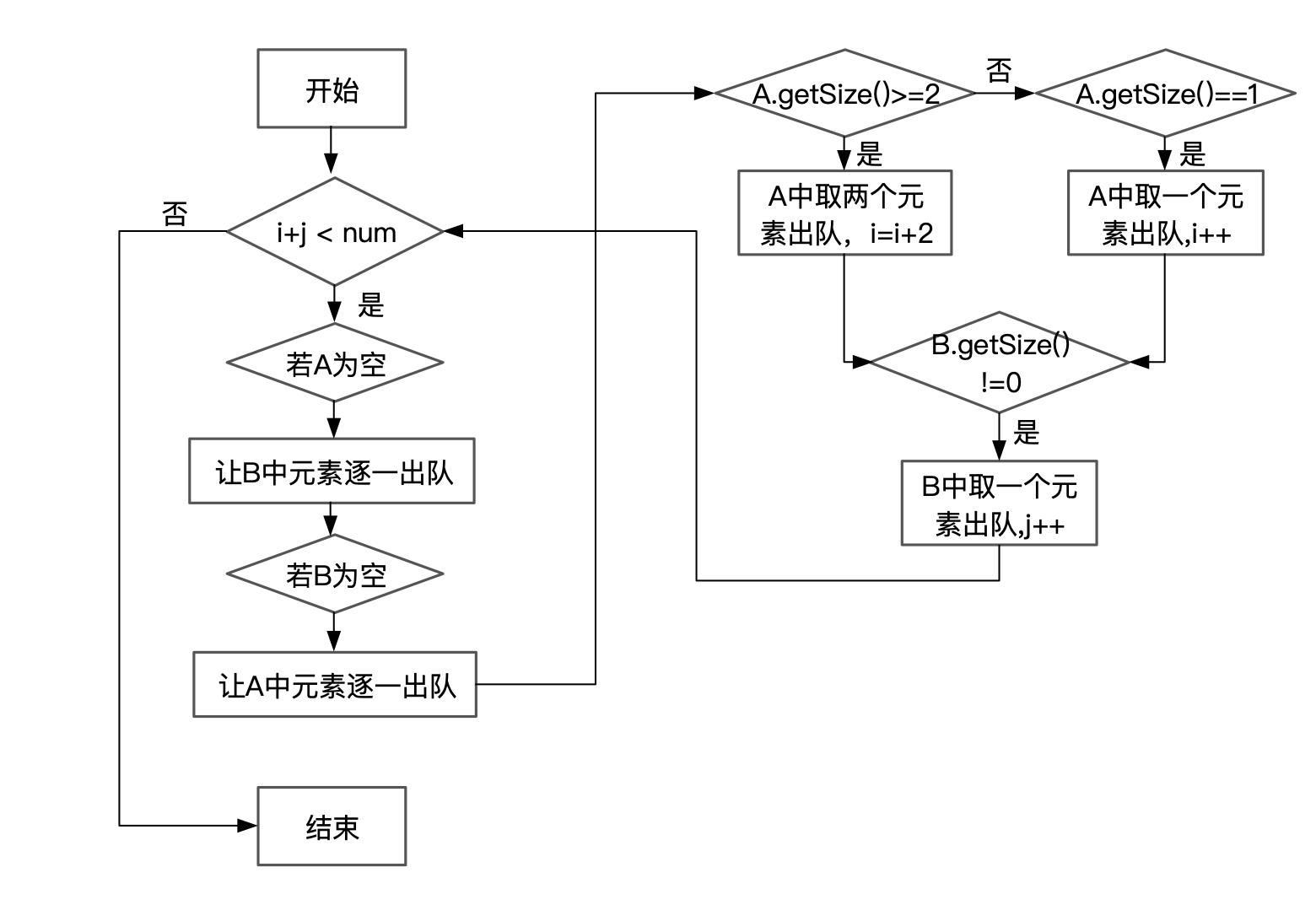
3.3.2 元素出队功能核心代码

3.4 Counter类中初始化功能的实现

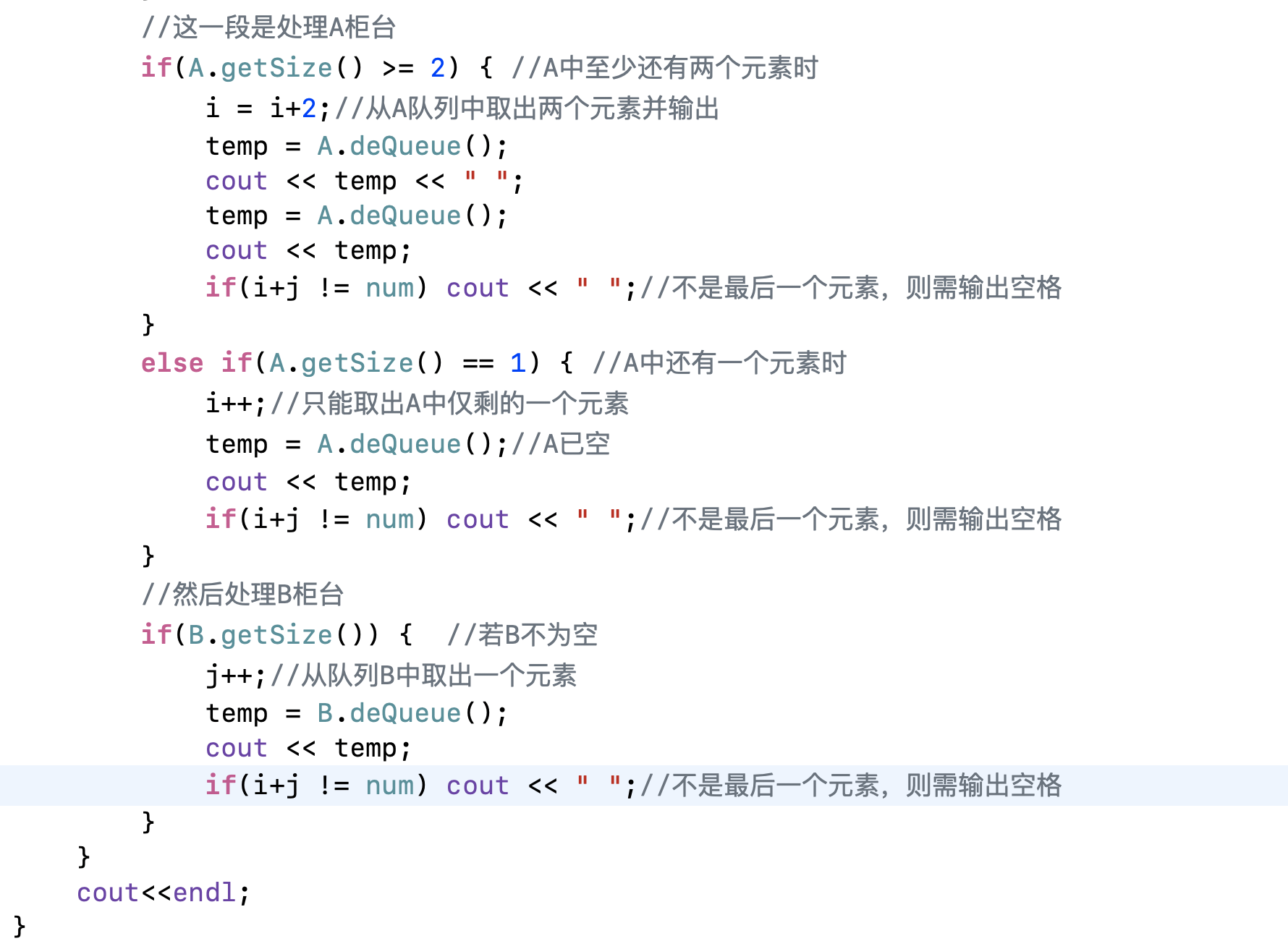
3.4.1 initialize()功能流程图

3.4.2 initialize()功能核心代码

3.5 Queue类中output()功能的实现

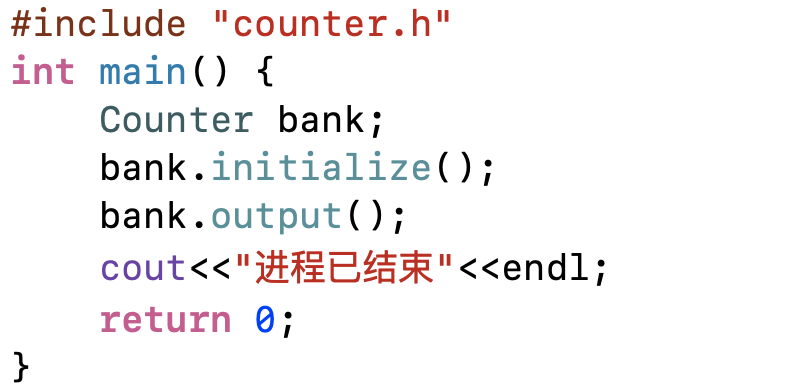
3.5.1 output()功能流程图

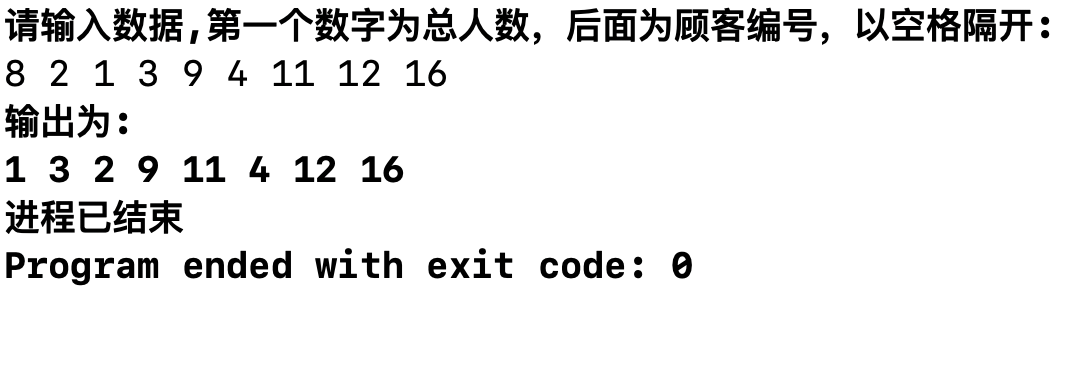
3.5.2 output()功能核心代码

需要注意A、B有一个为空，A中只剩一个元素（不足以输出两个时）的特殊情况。

3.9 主函数的实现

3.9.1 主函数流程图

3.9.2 总体系统核心代码

3.9.3 总体系统截屏示例

4 测试

4.1 功能测试

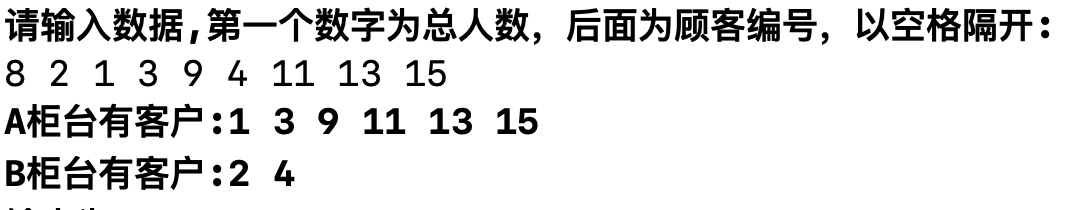
4.1.1 初始化功能测试

**测试用例1**：

8 2 1 3 9 4 11 13 15

**预期结果1**：

奇数1 3 9 11 13 15进入A队列、偶数2 4进入B队列

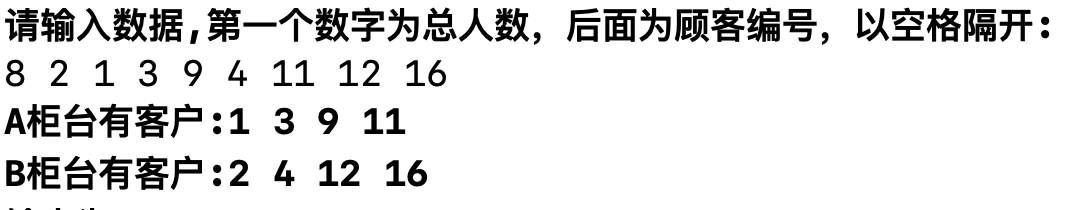
**实验结果1：**

**测试用例2**：

8 2 1 3 9 4 11 12 16

**预期结果2**：

奇数1 3 9 11进入A队列、偶数2 4 12 16进入B队列

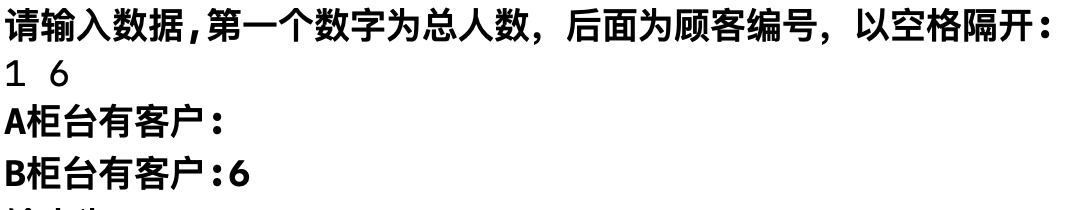
**实验结果2：**

**测试用例3**：

1 6

**预期结果3**：

A队列为空，偶数6进入B队列

**实验结果3：**

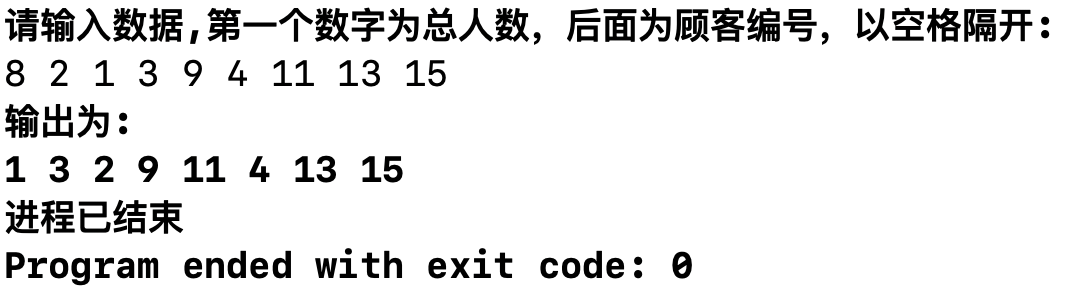
4.1.2 按顺序输出序号功能测试

**测试用例1**：

8 2 1 3 9 4 11 13 15

**预期结果1**：

输出1 3 2 9 11 4 13 15

**实验结果1：**

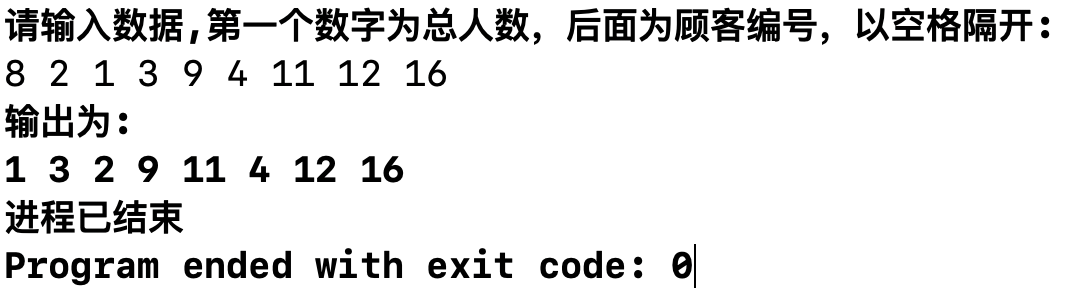
**测试用例2**：

8 2 1 3 9 4 11 12 16

**预期结果2**：

1 3 2 9 11 4 12 16

**实验结果2：**

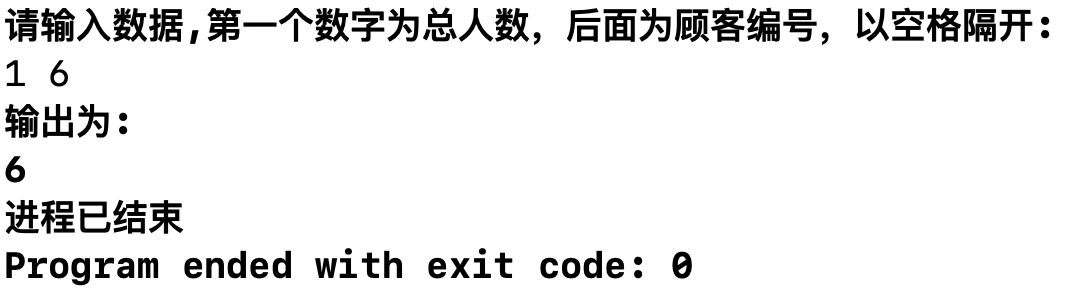


**测试用例3**：

1 6

**预期结果3**：

6

**实验结果3：**

4.2 边界测试

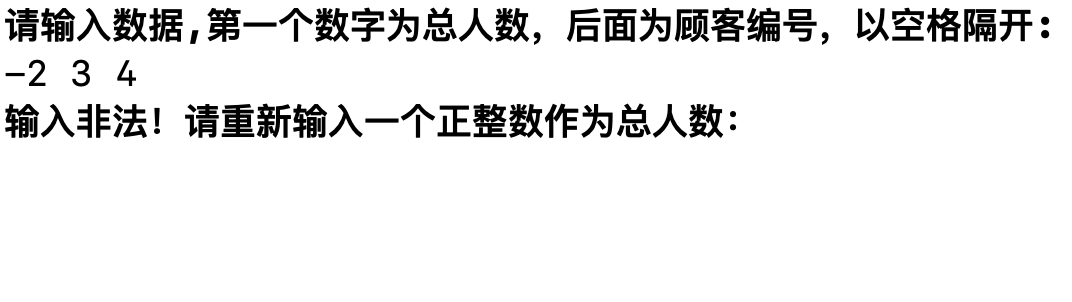
4.2.1 总人数大小非法

**测试用例：**初始输入总人数数据非法，输入非正整数-2

**预期结果：**若输入非正整数，立即给出错误提示并要求重新进行输入，程序运行正

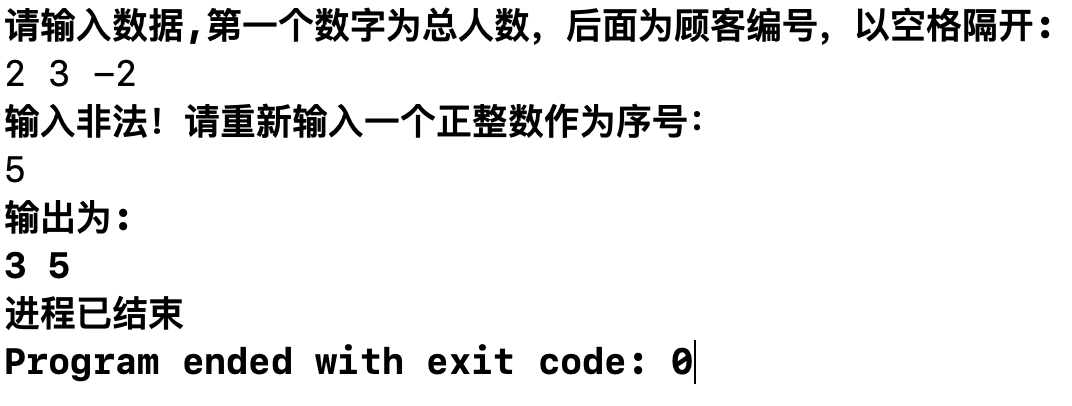
常不崩溃。

**实验结果：**

****4.2.2 顾客序号数据非法

**测试用例：**顾客序号输入非正整数

**预期结果：**程序提示输入不合法，要求重新输入直至合法，期间正常运行，不崩溃。

**实验结果：**

5 总结

5.1 遇到的错误

5.1.1 xx函数中

5.2 改进与优化

5.3 项目心得