银行业务

作 者 姓 名： 李翠琪

学 号： 1751022

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](#_Toc25184029)

[1.1 项目内容 1](#_Toc25184030)

[1.2 功能要求 1](#_Toc25184031)

[2 设计与实现 2](#_Toc25184032)

[2.1 Node类设计与实现 2](#_Toc25184033)

[2.2 LinkedList类设计与实现 2](#_Toc25184034)

[2.3 主函数设计与实现 3](#_Toc25184035)

[3 测试 5](#_Toc25184036)

[3.1 功能测试 5](#_Toc25184037)

[3.1.1 测试1 5](#_Toc25184038)

[3.1.2 测试2 5](#_Toc25184039)

[3.1.3 测试3 5](#_Toc25184040)

[3.1.4 测试4 6](#_Toc25184041)

[3.1.5 测试5 6](#_Toc25184042)

[3.1.6 测试6 6](#_Toc25184043)

# 1 分析

## 1.1 项目内容

设某银行有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，请按照业务完成的顺序输出顾客序列。假定不考虑顾客信后到达的时间间隔，并且当不同窗口同时处理完2个顾客时，A窗口的顾客优先输出。

## 1.2 功能要求

1. 输入说明：输入为一行正整数，其中第一数字N（N<=1000）为顾客总数，后面跟着N位顾客的编号。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。数字间以空格分隔。
2. 输出说明：按照业务处理完成的顺序输出顾客的编号。数字键以空格分隔，但是最后一个编号不能有多余的空格。
3. 测试用例：

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 输入 | 输出 | 说明 |
| 1 | 8 2 1 3 9 4 11 13 15 | 1 3 2 9 11 4 13 15 | 正常测试，A窗口人多 |
| 2 | 8 2 1 3 9 4 11 12 16 | 1 3 2 9 11 4 12 16 | 正常测试，B窗口人多 |
| 3 | 1 6 | 6 | 最小N |

# 2 设计与实现

## 2.1 Node类设计与实现

Node类为LinkedList的基础节点，是一个模板类。它能够存储一个T类型的值以及一个指向下一Node节点的next成员变量。

template <class T> class Node  
{  
public:  
  
 Node(const T & value, Node<T> \*next= nullptr) : value(value), next(next) {}  
 Node(Node<T> \*next= nullptr) : next(next) {}  
  
 T value;  
 Node<T> \*next;  
};

## 2.2 LinkedList类设计与实现

模板类，有两个Node类型的成员变量first以及last，分别指向链表头和末尾。这个类当作一个队列,其中有enqueue(),dequeue(),empty()函数。

empty()：只要简单的判断first是否为空即可简单的判断队列是否为空。

enqueue()：因为是队列，所以这个函数负责将数据加入链表的末尾。添加的过程是：如果头尾指针都为空的话，就让头尾指针指向这个元素，否则就在尾指针之后添加这个元素，并且让尾指针指向它。（而且如果first->next为空，得让它指向last，不这样的话first只是一个单独的节点，和后续节点没有产生联系）。

dequeue()：如果队列为空，返回。否则返回队列头元素（first的元素），并且让first指向它的下一个元素，如果first为空则也让last为空。

template<class T>class LinkedList  
{  
public:  
 LinkedList():first(nullptr),last(nullptr)  
 {  
  
 }  
 bool empty()  
 {  
 return first == nullptr;  
 }  
 void enqueue(T t)  
 {  
 if(first== nullptr && last== nullptr)  
 {  
 first = new Node<T>(t, nullptr);  
 last = new Node<T>(t, nullptr);  
 }  
 else  
 {  
 Node<T> \*temp = new Node<T>(t, nullptr);  
 last->next = temp;  
 last=last->next;  
 if(first->next== nullptr)  
 {  
 first->next = last;  
 }  
 }  
 }  
  
 T dequeue()  
 {  
 if (first == nullptr)  
 {  
 return nullptr;  
 }  
 T t = first->value;  
 first = first->next;  
  
 if(first== nullptr)  
 {  
 last = nullptr;  
 }  
 return t;  
 }  
  
 Node<T> \* first;  
 Node<T> \* last;  
};

## 2.3 主函数设计与实现

首先，先用一个数组存储用户输入的N个正整数，并且创建两个LinkedList链表队列，代表了A，B两个柜台。然后，根据编号为奇数还是偶数让他们分别进入两个队列（即代表了他们在A，B两个柜台上排队）。

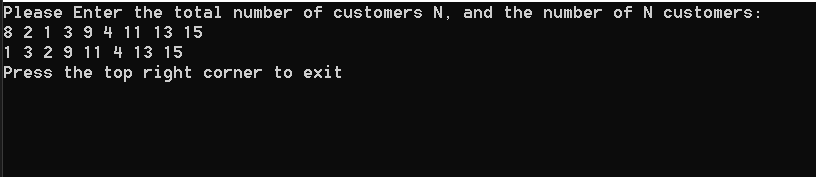
然后进入循环，循环中，如果队列A不为空，则打印此队列的头节点，并且让头节点指向下一个节点，重复一次，判断A队列是否为空，不为空则打印当前的子节点。接着判断B队列是否为空，不为空打印它的头节点，并且让他指向他的下一个节点。（其实这就是在模拟题目中的情形：当A窗口每处理完2个顾客是，B窗口处理完1个顾客。并且当不同窗口同时处理完2个顾客时，A窗口的顾客优先输出。）然后继续进入循环判断条件，如果两个队列都为空则退出循环，否则继续循环。值得注意的是,如果其中一个队列为空的话,那么就会把另一个队列的所有元素都输出了。

int main()  
{  
 cout << "Please Enter the total number of customers N, and the number of N customers:";  
 int N;  
 cin >> N;  
 vector<int> customers;  
  
 LinkedList<int> listA, listB;  
   
 for (int i = 0; i < N; ++i)  
 {  
 int customer;  
 cin >> customer;  
 customers.push\_back(customer);  
 }  
 for (int i = 0; i <N ; ++i)  
 {  
 if(customers[i]%2==1)//去1  
 {  
 listA.enqueue(customers[i]);  
 }  
 else  
 {  
 listB.enqueue(customers[i]);  
 }  
 }  
   
 while(!listA.empty() || !listB.empty())  
 {  
 if(!listA.empty())  
 {  
 cout << listA.first->value<<" ";  
 listA.first = listA.first->next;  
 }  
 if(!listA.empty())  
 {  
 cout << listA.first->value<<" ";  
 listA.first = listA.first->next;  
 }  
 if(!listB.empty())  
 {  
 cout << listB.first->value<<" ";  
 listB.first = listB.first->next;  
 }  
 }  
 return 0;  
}

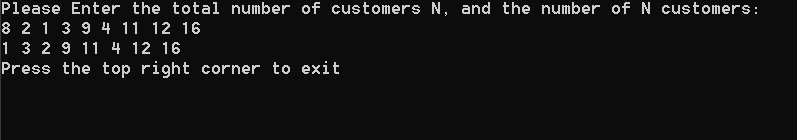
# 3 测试

## 3.1 功能测试

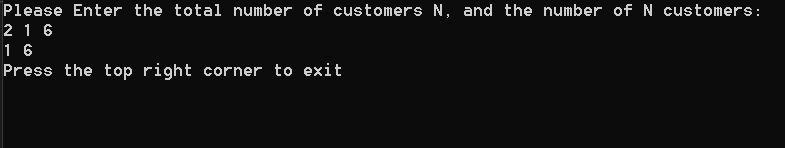
### 3.1.1 测试1



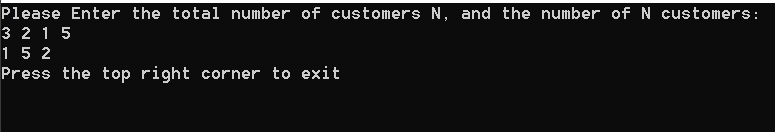
### 3.1.2 测试2



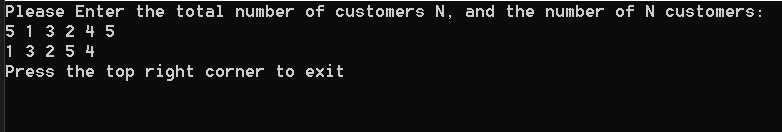
### 3.1.3 测试3



### 3.1.4 测试4



### 3.1.5 测试5



### 3.1.6 测试6

