项目说明文档

数据结构课程设计

——银行业务问题

作 者 姓 名： 王加炜

学 号： 2150265

指 导 教 师： 张颖

学院、 专业： 软件学院 软件工程

同济大学

Tongji University

# 分析

## 1.1背景分析

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0000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功能分析

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。

# 设计

## 2.1数据结构设计

如上功能分析所述，由于顾客进队和出队顺序符合先进先出的规则，所以很容易想到用队列这一逻辑结构来实现。

## 2.2类结构设计

由于顾客编号是有序的，所以采用数组存储顾客的信息。队列中用数组指针的方式存储了顾客的所有信息。

## 2.3成员与操作设计

### 2.3.1全局变量

#define defaultsize 1000

typedef map<int, int> mip;

mip priority;

### 2.3.2PQueue类

**公有函数：**

PQueue(int sz=defaultsize,int num=0);//构造函数

~PQueue();//析构函数

void makeempty();//清空队列

bool insert(int x);//插入到队尾

bool remove(int& x);//删除队头

bool isempty();//判断是否为空

bool isfull();//判断是否为满

void paixu();//核心函数

void input();//输入函数

void output();//输出函数

**保护函数：**

int count;//顾客总数

int\* elements;//存放顾客的数组

int maxlen;//顾客最大数目

void adjust(int current);//队列调整

## 2.4类设计

初始化一个队列，然后将顾客编号信息进行输入，通过paixu函数算出每个顾客所需的排队时间，按照递增顺序排列，最后按照队列的先进先出顺序将顾客编号进行输出。

# 三．实现

## **3.1顾客编号输入**

### 3.1.1文字说明

该函数负责对于顾客总数和顾客编号的输入，输入同时调用insert函数将该元素入队。

### 3.1.2 代码实现

void PQueue::input()

{

int mark;

int num;

cout << "请输入一行正整数(第一个数字N为顾客总数，后面的数字是N位顾客的编号)：" <<endl;

cin >> num;

for (int i = 0; i < num; i++)

{

cin >> mark;

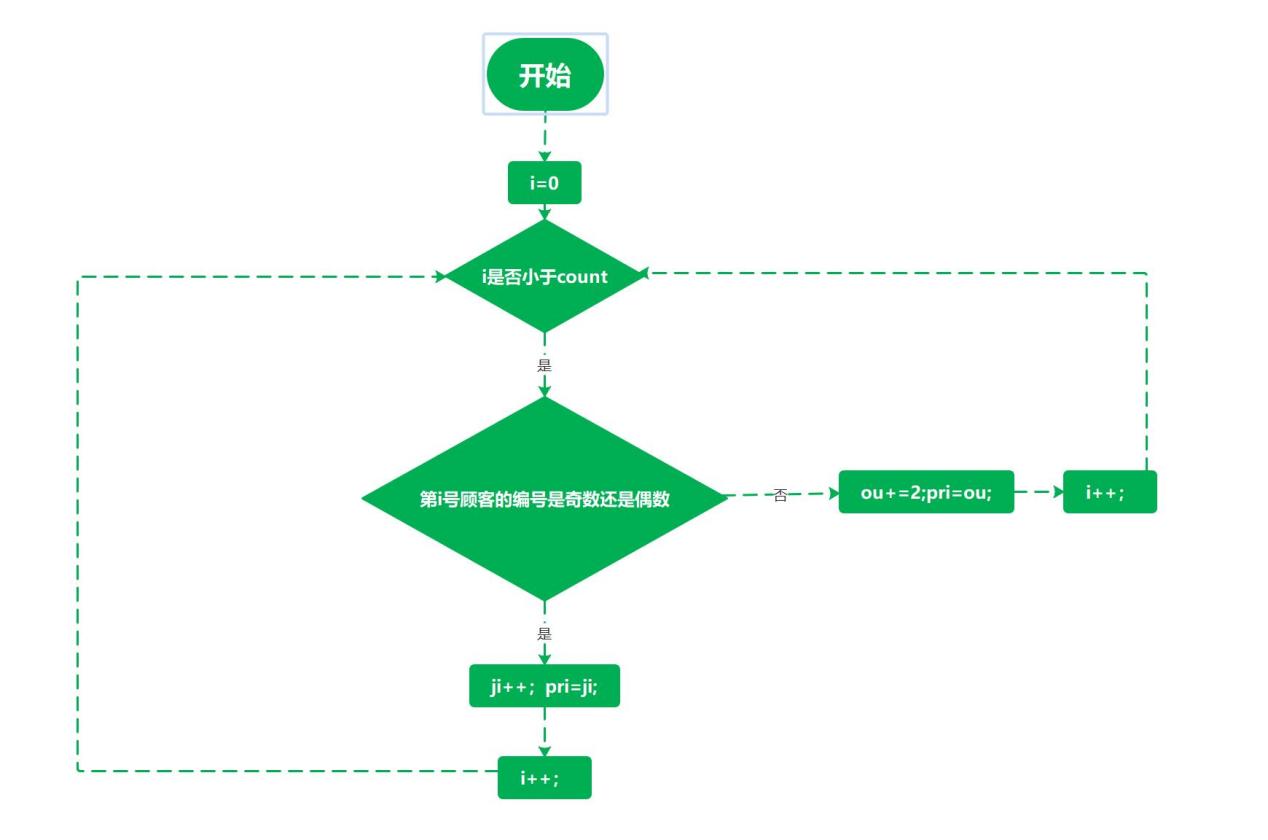
insert(mark);//将该元素插入到队列中去

}

}

## **3.2按照顾客编号给顾客等待时长排序**

### 3.2.1流程图



### 3.2.2文字说明

Paixu函数是整个设计功能实现的关键函数。

通过原队列顺序对每个元素进行遍历，根据该元素是奇数还是偶数计算出该元素的priority中所对应的值，即排队所花费的时间。最后调用从该元素位置截至的队列调整函数，按照排队所用时长的大小升序排列所有元素。

### 3.2.3代码实现

void PQueue::paixu()

{

int pri;

int ji = 0;

int ou = 0;

for (int i = 0; i < count; i++)

{

if (elements[i]% 2 == 1)//若为奇数，说明他通过窗口需要1分钟

{

ji++;

pri = ji;

}

else//若为偶数，说明他通过窗口需要2分钟

{

ou += 2;

pri = ou;

}

priority[elements[i]] = pri;//pri指的是完成排队需要的时间，pri越短说明越早出队

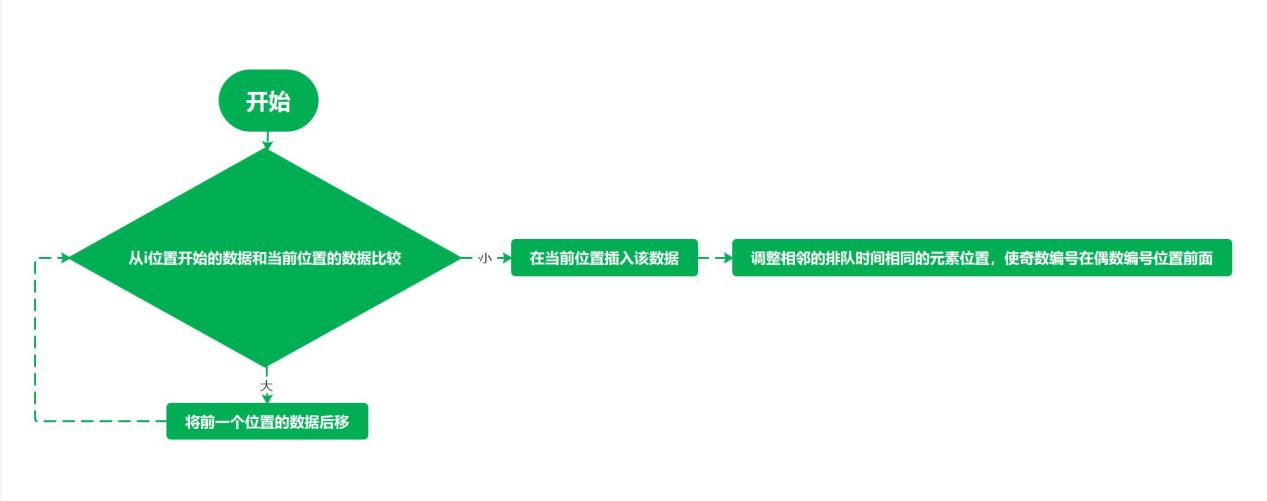
adjust(i);//调整队列顺序

}

}

## **3.3将顾客排队时长升序排列**

### 3.3.1流程图



### 3.3.2文字说明

该函数的作用是从数组下标0开始到current结束，按照每个顾客排队所花费的时间进行升序排列。

需要特别注意的是，当偶数编号的顾客和奇数编号的顾客排队时间相同的时候，奇数编号的顾客要优先出队，所以对于这种情况要单独考虑。如果检索到偶数编号在奇数编号前面的情况，只需要将两者交换一下位置即可。

### 3.3.3代码实现

void PQueue::adjust(int current)

{

int i;

int x = priority[elements[current]];//x表示当前新插入元素排队需要的时间

int y = elements[current];//y表示当前插入的元素

for (i = current-1; i>=0; i--)//从插入之前的最后一个元素开始向前遍历

{

if (x < priority[elements[i]])

{

elements[i+1 ] = elements[i];

}

else

{

break;

}

}

elements[i+1] = y;//至此，由排队时间从小到大的排序完成

if (i>=0 && priority[elements[i]]==priority[elements[i+1]])//特殊情况：偶数编号的顾客和奇数编号的顾客排队时间相同时，根据题意，奇数编号优先出队

{

if (elements[i + 1] % 2 == 1)//交换两个元素的位置

{

int temp;

temp = elements[i];

elements[i] = elements[i + 1];

elements[i + 1] = temp;

}

}

}

## 3.4输出顾客编号

### 3.4.1文字说明

该函数负责按照顾客办理好银行业务所需时间的先后顺序进行输出。

### 3.4.2 代码实现

void PQueue::output()

{

int p;

cout << "按业务处理完成先后顺序的顾客编号是："<<endl;

while (!isempty())

{

remove(p);

if (count > 0)

cout << p << " ";

else

cout << p << endl;

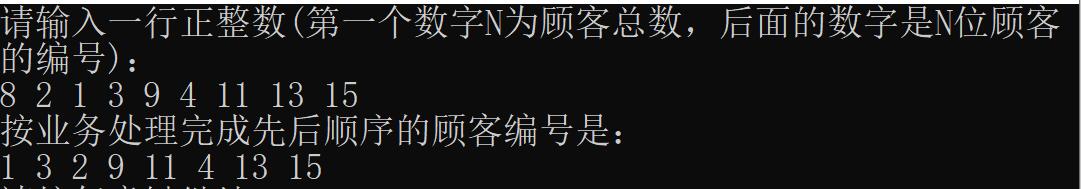
}

}

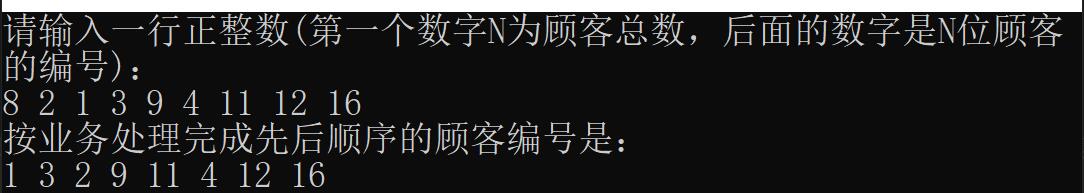
# 四．测试

## 4.1功能测试及边界测试

### 4.1.1正常测试，A窗口人多



### 4.1.2正常测试，B窗口人多



### 4.1.3只有一个顾客

