同濟大學
TONGJI UNIVERSITY

项目说明文档

银行业务

指导教师：张颖

1751984 王舸飞

1.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窗口的两倍。在办理业务的过程中，编号为奇数的顾客需要到A窗口办理，编号为偶数的需要到B窗口办理。某次顾客人数不超过1000人，要求给出业务完成的顺序序列。

在同时处理完成的情况下，要求优先输出A窗口的编号，输出的编号用空格隔开，且要注意到最后输出的编号后不得有空格。

1.2 功能分析

输入为一串正整数，其中第一个数字代表本次处理的人数。要求实现的功能很简单，即将之后的编号按照优先级调整顺序重新输出。

2.设计

2.1 数据结构设计

在银行处理业务要排队，这很容易让人联想到采用队列这一数据结构。队列先进先出的特点恰好符合本题的相关要求。考虑到本次处理业务有1000名顾客且只用进行一次输入输出，故采用顺序表存储队列，且不采用循环队列只使用普通的队列进行存储。

本题未使用stl库而是采用自己创建的队列头文件完成。

2.2 成员与操作设计

队列类（Quene）

受保护的成员：

int person[1000];//用于存储人员编号

int rear,front;//队列的头尾指针

int state; //因为本题特殊需求而创建，用于判断处理较慢的一个队列是否可以输出

公有操作：

Quene(){ReState();rear=0;front=0;memset(person,0,sizeof(person));};//构造函数，初始值置0

bool IsEmpty(){return front==rear?true:false;}//当首位指针相同时判断队列为空

void EnQuene(int);//将所给的人员编号插入队列中

void DeQuene();//退队函数，退队的同时进行一次输出

void AddState(){state++;}//当前状态加一

void ReState(){state=0;}//状态清零

int GetState(){return state;}//获取当前状态

其他函数：

#define ifeven(i) i%2==0?1:0 //预定义，判断是否为偶数

2.3 系统设计

程序运行后，系统会创建一个整型变量用于存储人数和一个整型数组用于存储编号，之后会创建两个队列qa、qb，分别代表本题中的快队和慢队。

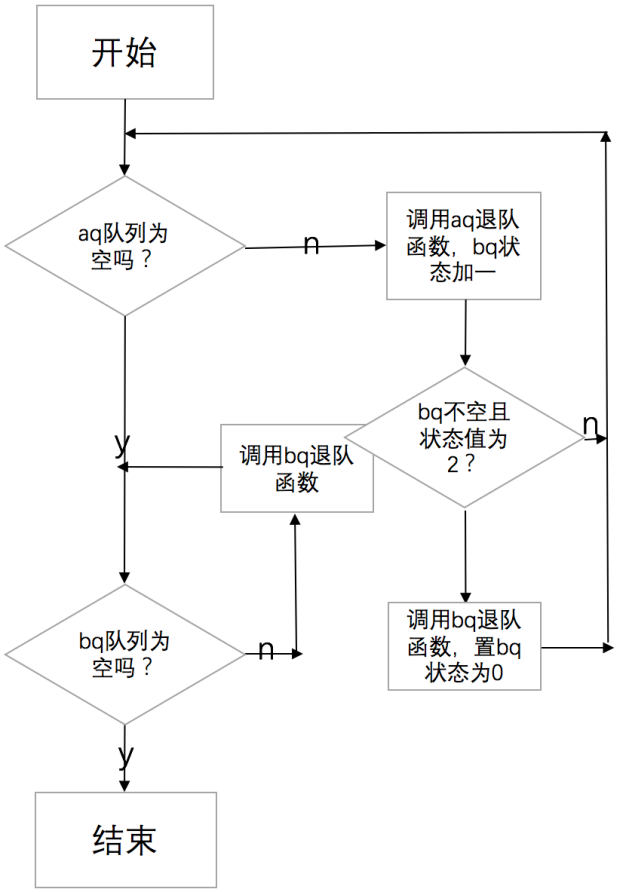
在输入完成后，系统会根据奇偶性将将人员编号插入两队中，通过题目要求执行相应的输出输出结果。

3.实现

3.1 队列输出的实现

3.1.1 队列输出流程图

注：在调用退队函数后，检查a、b是否均为空，否则输出空格。



3.1.2 队列输出核心代码

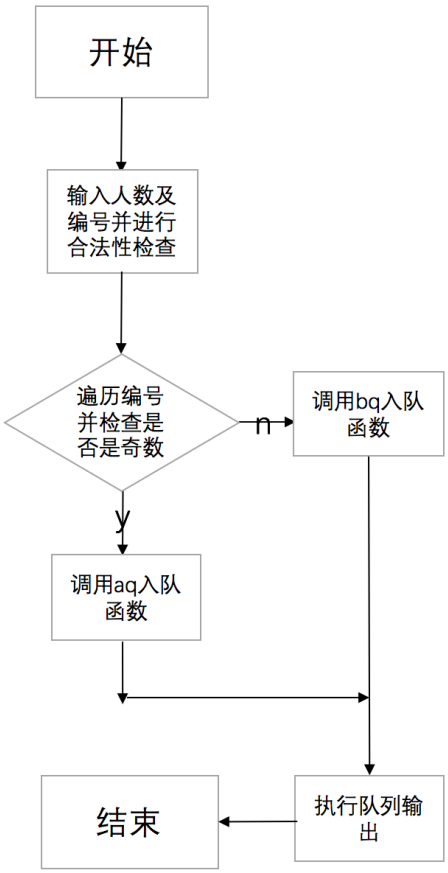
```
while(!aq.IsEmpty()){

    aq.DeQuene();
    if(!aq.IsEmpty()||!bq.IsEmpty())cout<<" ";
    bq.AddState();
    if(bq.GetState()==2&&!bq.IsEmpty()){
        bq.DeQuene();
        if(!aq.IsEmpty()||!bq.IsEmpty())cout<<" ";
        bq.ReState();}
}

while(!bq.IsEmpty()){bq.DeQuene();if(!bq.IsEmpty())cout<<" ";}
```

3.1 总体功能的实现

3.1.1 总体功能流程图



3.1.2 总体功能核心代码

```
int main() {int total,nums[1000];
    Quene aq,bq;
    memset(nums,0,sizeof(nums));          //设初值
    cout<<"请输入办理业务的顾客人数: "<<endl;
    cin>>total;   //输入人数
    while(total<=0){cout<<"输入的人数有误，请重修输入: "<<endl;cin>>total;}
    for(int i=0;i<total;i++)cin>>nums[i]; //输入编号

    for(int i=0;nums[i]!=0;i++)   //先排好队
    {if(ifeven(nums[i]))bq.EnQuene(nums[i]);
    else aq.EnQuene(nums[i]);
    }
```

4.测试

4.1 功能测试

4.1.1 正常情况测试（A窗口人多）

测试用例：8 2 1 3 9 4 11 13 15

预期结果：1 3 2 9 11 4 13 15

实验结果：

```
请输入办理业务的顾客人数:
8 2 1 3 9 4 11 13 15
1 3 2 9 11 4 13 15Program ended with exit code: 0
```

4.1.1 正常情况测试（ B窗口人多）

测试用例：8 2 1 3 9 4 11 12 16

预期结果：1 3 2 9 11 4 12 16

实验结果：

```
请输入办理业务的顾客人数:
8 2 1 3 9 4 11 12 16
1 3 2 9 11 4 12 16Program ended with exit code: 0
```

4.1.3 最小N

测试用例：1 6

预期结果：6

实验结果：

```
请输入办理业务的顾客人数：
1 6
6Program ended with exit code: 0
```

4.1.4 较大数列测试

测试用例：30 1 3 5 7 9 12 34 55 43 32 11 33 67 88 76 75 74 80 99 13 53 54 60 31 36 37 97 98 96 87

预期结果：1 3 12 5 7 34 9 55 32 43 11 88 33 67 76 75 99 74 13 53 80 31 37 54 97 87 60 36 98 96

实验结果：

```
请输入办理业务的顾客人数：
30 1 3 5 7 9 12 34 55 43 32 11 33 67 88 76 75 74 80 99 13 53
54 60 31 36 37 97 98 96 87
1 3 12 5 7 34 9 55 32 43 11 88 33 67 76 75 99 74 13 53 80 31
37 54 97 87 60 36 98 96Program ended with exit code: 0
```

4.1.5 错误人数输入

测试用例：

预期结果：程序给出错误提示并要求重新输入

实验结果：

```
请输入办理业务的顾客人数：
-1
输入的人数有误，请重修输入：
```