《面向对象程序设计实验》任务书

# 实验一 面向过程的整型队列编程

**一、实验目的与要求**

1．熟悉C++的编程环境；

2．熟悉C++程序的编辑、编译、链接、调试及运行上机操作；

3．运用面向过程的编程技术实现队列；

4．重点掌握程序调试技术。

**二、实验内容**

整型队列是一种先进先出的存储结构，对其进行的操作通常包括：向队列尾部添加一个整型元素、从队列首部移除一个整型元素等。

整型循环队列类型Queue及其操作函数采用面向过程的结构化程序设计方式定义，其中结构体定义如下：

struct Queue {

int\* const elems; //elems申请内存用于存放队列的元素

const int max; //elems申请的最大元素个数max

int head; //队列头head

int tail; //队列尾tail

};

**需要注意是事项：**

1．请不要为Queque结构体新增或修改成员。

2．队列应实现为循环队列，当队尾指针tail快要追上队首指针head时，即如果满足(tail+1)%max == head，则表示表示队列已满，故队列最多存放max-1个元素；当head == tail时则表示队列为空。

3．编程时应采用VS2019开发，并将其编译模式设置为x86模式。

4．请先实现完成队列操作的所有如下函数，然后编写一个main函数对这些函数进行测试。

1. **void queInit(Queue \*const p, int m);**

功能：初始化p指向的新队列，最多申请m个元素。

要求：对p指向的新队列初始化时，为其elems分配m个整型元素内存，并初始化max为m，以及初始化head = tail = 0。p为空指针或者p指向的队列已经分配内存时应报错并输出错误提示信息。

1. **void queInit(Queue \*const p, const Queue& q);**

功能：用队列q初始化p指向的新队列。

要求：用已经存在的队列q深拷贝构造p指向的新队列，新队列不能和队列q的elems共用同一块内存，新队列的elems需要分配和q为elems分配的同样大小的内存，并且将已经存在q的elems的内容深拷贝至新队列分配的内存；新队列的max、head、tail应设置成和已经存在的队列q相同。p为空指针或者p指向的队列已经分配内存时应报错并输出错误提示信息。

1. **int queSize(const Queue\* const p);**

功能：返回队列申请的最大元素个数max。

要求：p为空指针时应报错并输出错误提示信息。

1. **int queNumber(const Queue\* const p);**

功能：返回p指向的队列的实际元素个数。

要求：p为空指针时应报错并输出错误提示信息。

1. **Queue\* const queEnter(Queue\* const p, int e);**

功能：将e加入队列，并返回p。

要求：队列满或者p为空指针时应报错并输出错误提示信息。

1. **Queue\* const queLeave(Queue\* const p, int& e);**

功能：从队列移出元素到e，并返回p。

要求：队列为空或者p为空指针时应报错并输出错误提示信息。

1. **Queue\* const queAssign(Queue\* const p, const Queue& q);**

功能：用队列 q赋值给p指向的队列并返回p。

要求：p指向的队列如果已经有内存则应先释放以避免内存泄漏，然后分配和队列q为elems分配的同样大小的内存，并将q的elems的元素复制到p指向队列的elems，最后设置其max、head、tail和q的对应值相同。p为空指针时应报错并输出错误提示信息。

1. **void quePrint(const Queue\* const p, char\* s);**

功能：打印p指向的队列至s。

要求：print函数打印队列时从队首打印至队尾，打印的元素之间以一个空格分隔。s为空指针时应报错并输出错误提示信息。

1. **void queClear(Queue\* const p);**

功能：清空并重置p指向的队列。

要求：将p所指队列恢复到初始化后的状态。

1. **void queDestroy(Queue\* const p);**

功能：销毁p指向的队列。

要求：释放p指向的队列动态分配的内存，重置队列到初始化前的状态。