《面向对象程序设计实验》任务书

# 实验三 面向对象的整型栈编程

**一、实验目的与要求**

1．掌握多态函数的定义方法；

2．掌握引用与深浅拷贝构造、赋值等概念；

3．理解双队列模拟栈的基本操作，掌握继承与聚合对象的函数调用方法；

4．实现支持移动语义的栈。

**二、实验内容**

整型栈是一种先进后出的存储结构，对其进行的操作通常包括：向栈顶压入一个整型元素、从栈顶弹出一个整型元素等。

整型栈类STACK采用之前定义的两个QUEUE类模拟一个栈，其操作函数采用面向对象的C++语言定义。

请将完成上述操作的所有如下函数采用C++语言编程， 然后写一个main函数对栈的所有操作函数进行测试。

**请不要添加定义任何新的数据成员和成员函数。**

class 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

public:

QUEUE(int m);

QUEUE(const QUEUE& q);

QUEUE(QUEUE&& q) noexcept;

virtual int size () const;

virtual int number () const;

virtual QUEUE& enter(int e);

virtual QUEUE& enter(short n, ...);

virtual QUEUE& leave(int& e);

virtual QUEUE& leave(int& n, int\* buf);

virtual QUEUE& operator=(const QUEUE& q);

virtual QUEUE& operator=(QUEUE&& q) noexcept;

virtual QUEUE& queCat(const QUEUE& q);

virtual void print(char \*s) const;

virtual void clear();

virtual ~QUEUE();

};

class STACK : public QUEUE {

QUEUE q;

public:

STACK(int m); //初始化栈：**最多存放2m-2个元素**

STACK(const STACK& s); //用栈s深拷贝初始化栈

STACK(STACK&& s) noexcept; //用栈s移动拷贝初始化栈

int size()const; //返回栈的容量即2m

int number () const; //返回栈的实际元素个数

STACK& enter(int e); //将e入栈，并返回当前栈

STACK& enter(short n, ...); //批量入栈

STACK& leave(int& e); //出栈到e，并返回当前栈

STACK& leave(int& n, int\* buf); //批量出栈

STACK& operator=(const STACK& s); //深拷贝赋值并返回被赋值栈

STACK& operator=(STACK&& s) noexcept; //移动赋值并返回被赋值栈

void print(char \*s)const; //从栈底到栈顶打印栈元素

void clear(); //清空栈

~STACK(); //销毁栈

};

编程时应采用VS2019开发，并将其编译模式设置为x86模式，其它需要注意的事项说明如下：

(1) 在用STACK(int m)对栈初始化时， 为其基类和成员q的elems分配m个整型元素内存，并初始化基类和成员q的max为m，以及初始化对应的head=tail=0。

(2) 对于STACK(const STACK& s)深拷贝构造函数，在用已经存在的对象s深拷贝构造新对象时，新对象不能共用s的基类和成员q为elems分配的内存，新对象要为其基类和成员q的elems分配和s为其基类和成员q的elems分配的同样大小的内存，并且将s相应的elems的内容深拷贝至新对象为对应elems分配的内存；新对象应设置其基类和成员q的max、head、tail和s的对应值相同。

(3) 对于STACK(STACK&& s)移动拷贝构造函数，在用已经存在的对象s移动构造新对象时，新对象接受使用s为其基类和成员q的对应elems分配的内存，并且新对象的max、head、tail应和s的基类和成员q的对应值相同；s的基类和成员q的elems设置为空指针以表示内存被移走，同时其对应的max、head、tail均应置为0。

(4) 对于STACK& operator=(const STACK& s)深拷贝赋值函数，在用等号右边的对象s深拷贝赋值等号左边的对象s时，等号左边对象的基类和成员q不能共用s的基类和成员q为elems分配的内存，若等号左边的对象为其基类和成员q的elems分配了内存，则应先释放掉以避免内存泄漏，然后为其elems分配和s为其基类和成员q的elems分配的同样大小的内存，并且将s对应两个elems的内容拷贝至等号左边对象对应两个elems的内存；等号左边对象中的max、head、tail应设置成和s中基类和成员q的对应值相同。

(5) 对于STACK& operator=(STACK&& s)移动赋值，在用等号右边的对象s移动赋值给等号左边的对象时，等号左边的对象如果已经为其基类和成员q中的elems分配了内存，则应先释放以避免内存泄漏，然后接受使用s的基类和成员q为elems分配的内存，并且等号左边对象中的max、head、tail应和s中基类和成员q中的对应值相同；s中基类和成员q的elems设置为空指针以表示内存被移走，同时其对应的max、head、tail均应设置为0。

(6) 栈空弹出元素或栈满压入元素均应报错并输出错误提示信息，并且保持其内部状态不

(7) 打印栈时从栈底打印到栈顶，打印的元素之间以空格分隔。

(8) 基类QUEUE除构造函数外，其它成员函数都需要声明为虚函数。