实验三 类与对象

实验目的

1、 掌握类的概念、类的定义格式、类与结构的关系、类的成员属性和类的封装性；

2、 掌握类对象的定义；

3、 理解类的成员的访问控制的含义，公有、私有和保护成员的区别；

4、 掌握构造函数和析构函数的含义与作用、定义方式和实现，能够根据要求正确定义和重载构造函数。能够根据给定的要求定义类并实现类的成员函数；

5、 掌握静态数据成员和常数据成员的使用。

实验内容

1. 设计一个整型数组类 intArray

下面是一个整型数组类 intArray 的声明，请给出该类所有成员的类外定义。

```
//整型数组类
class intArray
{
private:
    int* element;                    //指向动态数组的指针
    int arraysize;                       //数组的大小
public:
    intArray(int size);                  //构造函数
    intArray(const intArray& x);     //复制构造函数
    ~intArray();                     //析构函数
    bool set(int i, int elem);       //设置第i个数组元素的值，设置成功返回true，
                                         //失败返回false
    bool get(int i, int& elem);          //获取第i个数组元素的值，获取成功返回
true，
                                         //失败返回false
    int length() const;                  //获取数组的长度
```

```
    void resize(int size);              //重置数组
    void print();                       //输出数组
};
```

2. 设计一个整型链表类 intList

下面是一个整型链表类 intList 的声明，请给出该类所有成员的类外定义。

```
struct Node
{
    Node* next;
    int data;
};
//整型链表类
class intList
{
private:
    Node* pFirst;
public:
    intList();                              //构造函数
    ~intList();                             //析构函数
    bool insert(int i, int elem);       //向链表的第i个位置插入一个元素，插入成功返回
                                            //true，失败返回false
    bool remove(int i, int& elem);      //删除量表第i个位置插入一个元素，删除成功返回
                                            //true，失败返回false
    int find(int elem) const;       //查找值为elem的元素，返回该元素在链表中的位置
    int length() const;                     //返回链表长度
    void printList();                   //输出链表
};
```

3. 设计一个整型堆栈类 intStack

下面是一个整型堆栈类 intStack 的声明，请给出该类所有成员的类外定义。

```
//整型堆栈类
class intStack
{
private:
    int* data;                      //指向动态数组的指针
    int  top;                   //栈顶指针
    int size;                       //堆栈的容量
public:
```

```
    intStack(int size = 10);   //构造函数
    ~intStack();               //析构函数
    bool push(int elem);       //入栈操作
    bool pop(int& elem);       //出栈操作
    int length() const;            //获取栈中元素的个数
};
```

4. 类和对象的使用

下面是主函数 main 的定义。

```
#include <iostream>
#include "intArray.h"
#include "intList.h"
#include "intStack.h"
using namespace std;
int main()
{
    cout << "Hello int World!\n";
    //intArray
    intArray arr(10);
    for (int i = 0; i < 10; i++)
        if (!arr.set(i, i))
        {
            cout << "array Set fail!" << endl;
            //exit(1);
        }
    arr.print();
    //intList
    intArray arr2(arr);
    intList list;
    for (int i = 0; i < 10; i++)
    {
        int elem = -1;
        if (!arr2.get(i, elem))
        {
            cout << "array Get fail!" << endl;
            //exit(1);
        }
        if (!list.insert(i, elem))
        {
            cout << "list Insert fail!" << endl;
            //exit(1);
```

```
        }
    }
    list.printList();
    int elem;
    if (list.remove(4, elem))
        cout << "Successfully delete the fourth element!" << endl;
    else
    {
        cout << "list Delete fail!" << endl;
        //exit(1);
    }
    list.printList();
    int loc = list.find(5);
    if (loc >= 0)
        cout << "Find! The position of the fifth element is " << loc << endl;
    else
    {
        cout << "list Find fail!" << endl;
        //exit(1);
    }
    //intStack
    intStack stack(10);
    for (int i = 0; i < 10; i++)
    {
        int elem = -1;
        if (!arr2.get(i, elem))
        {
            cout << "array Get fail!" << endl;
            //exit(1);
        }
        if (!stack.push(elem))
        {
            cout << "stack Push fail!" << endl;
            //exit(1);
        }
    }
    cout << "The longth of stack is " << stack.length() << endl;
    for (int i = 0; i < 5; i++)
    {
        int elem = -1;
        if (!stack.pop(elem))
        {
            cout << "stack Pop fail!" << endl;
            //exit(1);
```

```
        }
        else
            cout << elem << " is poped." << endl;
    }
    cout << "The longth of stack is " << stack.length() << endl;
    return 0;
}
```