

实验项目名称	C++基础练习		实验成绩	
实 验 者	张心宇 175064	专业班级	中法计 172	
实验日期			2018 年 9 月 27 日	
<div>一、实验目的</div> <div>1、学习类与对象的定义，掌握类与对象的使用方法。</div> <div>2、学习数据成员与成员函数的访问方式，理解构造函数和析构函数的定义与执行过程，学会构造函数的重载方法。</div> <div>3、掌握数组与指针的定义与使用方法，理解数组与指针的存储分配与表示。</div> <div>4、掌握用指针和引用向函数传递参数。</div> <div>5、掌握静态数据成员和静态成员函数的使用。</div> <div>6、理解友元与友元函数的作用与使用方法。</div> <div>二、实验内容</div> <div>1、下面是一个计算器类的定义，请完成该类成员函数的实class Counter</div> <div>{</div> <div>public:</div> <div>Counter(int number);</div> <div>void increment(); //给原值加1</div> <div>void decrement(); //给原值减1</div> <div>int getValue(); //取得计数器值</div> <div>int print(); //显示计数</div> <div>private:</div> <div>int value;</div> <div>};</div> <div>2、根据注释语句的提示，实现类Date 的成员函数。</div> <div>class Date</div> <div>{</div> <div>public:</div> <div>void printDate();//显示日期</div> <div>void setDay(int d);//设置日的值</div> <div>void setMonth(int m);//设置月的值</div> <div>void setYear(int y);//设置年的值</div> <div>private:</div> <div>int day,month,year;</div> <div>};</div> <div>int main()</div> <div>{</div> <div>Date testDay;</div> <div>testDay.setDay(5);</div> <div>testDay.setMonth(10);</div> <div>testDay.setYear(2014);</div>				

```
testDay.printDate();  
return 0;  
}
```

3、考课本例子，建立一个源程序文件，在此文件中建立一个新的类建的类命名为Rect。

```
class Rect  
{  
public:  
int Area_int();  
double Area_double();  
Rect(double length, double width);  
Rect(int length, int width);  
virtual ~Rect();  
private:  
int nLength;  
int nWidth;  
double dLength;  
double dWidth;  
};
```

【要求】

(1)向Rect 类中添加数据成员及成员函数，并完善成员函数的功能。如设计一个Area_int()函数，计算边长为整型的长方形的面积；设计一个Area_double()函数，计算边长为double 型的长方形的面积。

(2)重载构造函数。一种构造函数用整型变量记录长方形的长和宽，另一种构造函数用double 型记录。

(3)体现对象的构造和析构过程。例如，在构造函数中用cout<<"I am the constructor!"<<endl；在析构函数中输出cout<<"I am the destructor"<<endl。

(4)在main()函数中定义两个Rect 类的对象，一个对象用实例实现(就像定义普通的变量一样)，另一个对象用指针实现(利用关键字new，给指针分配内存空间)。并用不同的参数，以调用不同的构造函数体现构造函数的重载。

4、声明一个Student 类，在该类中包括一个数据成员score（分数）、两个静态数据成员total_score（总分）和count(学生人数)；还包括一个成员函数account()用于设置分数、累计学生的成绩之和、累计学生人数，一个静态成员函数sum()用于返回学生的成绩之和，另一个静态成员函数average()用于求全班成绩的平均值。在main()函数中，输入某班学生的成绩，并调用上述函数求出全班学生的成绩之和和平均分。

5、依据下面的描述设计并实现一个时间类Time，并在main 函数中定义所需要的对象验证所实现代码的正确性。

（1）私有数据成员包含小时（Hour）、分钟（Minute）和秒（Second）；

（2）三个重载构造函数：

a)一个是无参数的构造函数；

b)一个是带参数的构造函数，实现对数据成员的初始化；

c)一个是copy 构造函数，实现用一个对象初始化本对象；

（3）成员函数实现时间对象相加运算：

a) Time Add(Time&);

（4）友元函数实现时间对象相加的运算符重载：

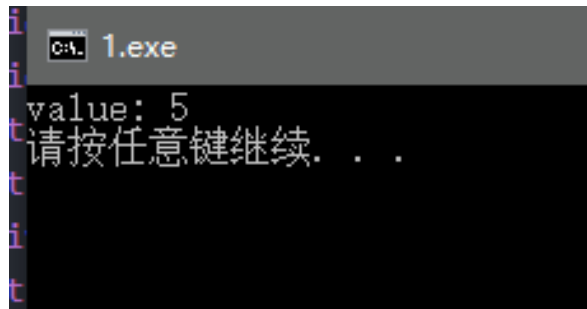
```
friend Time operator + (Time&, Time&);
```

(5) 公有成员函数void Display () ; 输出对象的数据成员;

三、实验源程序及结果截图

1.

```
#include<iostream>
using namespace std;
class Counter
{
public:
Counter(int number);
void increment(); //给原值加 1
void decrement(); //给原值减 1
int getValue(); //取得计数器值
int print(); //显示计数
private:
int value;
};
Counter::Counter(int number):value(number) {}
void Counter::increment() {
    value++;
}
void Counter::decrement() {
    value--;
}
int Counter::getValue() {
    return value;
}
int Counter::print() {
    cout<<"value: "<<getValue()<<endl;
    return 0;
}
int main() { //main 写成了 mian()
    Counter num(4);
    num.increment();
    num.print();
    return 0;
}
```



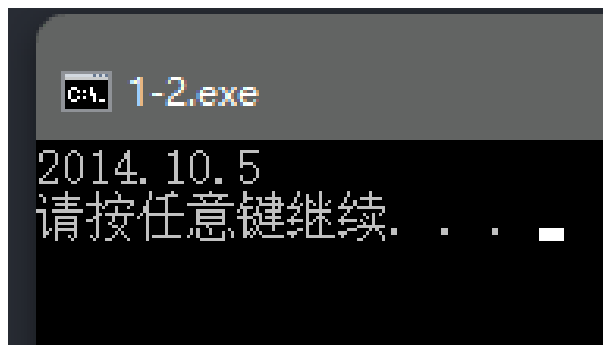
2.

```
#include<iostream>
using namespace std;
class Date
{
```

```

public:
void printDate(); //显示日期
void setDay(int d); //设置日的值
void setMonth(int m); //设置月的值
void setYear(int y); //设置年的值
private:
int day, month, year;
};
void Date::printDate() {
    cout<<year<<'.'<<month<<'.'<<day<<endl;
}
void Date::setDay(int d) {
    day = d;
}
void Date::setMonth(int m) {
    month = m;
}
void Date::setYear(int y) {
    year = y;
}
int main()
{
    Date testDay;
    testDay.setDay(5);
    testDay.setMonth(10);
    testDay.setYear(2014);
    testDay.printDate();
    return 0;
}

```



3.

```

#include<iostream>
using namespace std;
class Rect
{
public:
    int Area_int();
    double Area_double();
    //---2. 重载构造函数
    Rect(double length, double width);
    Rect(int length, int width);
    ~Rect();
private:
    int nLength;

```

```

    int nWidth;
    double dLength;
    double dWidth;
};

//---1. 计算面积
int Rect::Area_int() {
    return nLength*nWidth;
}
double Rect::Area_double() {
    return dLength*dWidth;
}

//---3. 跟踪构造和析构
Rect::Rect(double length, double width):dLength(length),dWidth(width) {
    cout<<"i'm the double constructor!"<<endl;//---3
}
Rect::Rect(int length, int width):nLength(length),nWidth(width) {
    cout<<"i'm the int constructor!"<<endl;//---3
}
Rect::~Rect() {
    cout<<"i'm the destructor!"<<endl;
}

//---4
int main() {
    Rect r1(2,1);
    //另一个对象用指针实现(利用关键字 new, 给指针分配内存空间)    掌握动态内存分配。
    Rect *r2;
    r2 = new Rect(2.0, 1.0);
    delete r2;
    //析构函数什么时候调用, 为什么只运行一次析构函数, 所有对象难道是在共用一个类的资源? .... 因为没有 delete
    // r2->Area_int();
    return 0;
}

4.
#include<iostream>
using namespace std;
class Student{
public:
    void account(double);//设置分数, 累计成绩, 累计学生
    static double sum();//学生成绩和

```

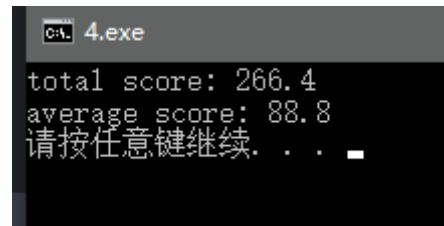


```

    static double average(); //全班平群成绩
private:
    double score; //个人你成绩
    static double total_score; //总成绩
    static int count; //人数
};
void Student::account(double s) {
    score = s;
    total_score += score;
    count++;
}
double Student::sum() { //静态成员函数可以在类内定义吗
    return total_score;
}
double Student::average() {
    return total_score/count;
}
//初始化静态数据成员
double Student::total_score = 0.0;
int Student::count = 0;

int main() {
    Student s1, s2, s3;
    s1.account(99.9);
    s2.account(88.8);
    s3.account(77.7);
    cout<<"total score: "<<Student::sum()<<endl
        <<"average score: "<<Student::average()<<endl;
    return 0;
}

```



```

5.
#include<iostream>
using namespace std;
class Time{
public:
    Time() {} ;
    Time(int, int, int);
    Time(const Time &);
    Time add(Time &);
    friend Time operator +(Time &, Time &); //重载+
    void display(); //输出对象的数据成员
private:
    int hour;

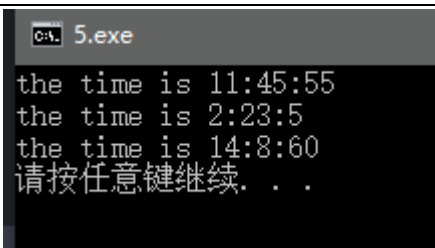
```

```

    int minute;
    int second;
};
Time::Time(int h, int m, int s) {
    hour = h;
    minute = m;
    second = s;
}
Time::Time(const Time &p) {
    hour = p.hour;
    minute = p.minute;
    second = p.second;
}
Time Time::add(Time &p) {
    //使用满 60 的进位的机制实现成员函数
    Time value;int add_minute=0;int add_hour=0;//记录进位
    value.second = this->second+p.second;
    if(value.second>60) { //实现进位机制
        value.second -= 60;
        add_minute = 1;
    }

    value.minute = this->minute+p.minute+add_minute;
    if(value.minute>60) {
        value.minute -=60;
        add_hour = 1;
    }
    value.hour = this->hour+p.hour+add_hour;
    return value;
}
Time operator +(Time &p1, Time &p2) {
    //通过调用 add() 成员函数实现 +的重载。
    return p1.add(p2);
}
void Time::display() {
    cout<<"the time is "<<hour<<' :'<<minute<<' :'<<second<<endl;
}
int main() {
    Time t1(11, 45, 55), t2(2, 23, 5), t3;
    t1.display();t2.display();
    t3=t1+t2;
    t3.display();
    return 0;
}

```



```
5.exe
the time is 11:45:55
the time is 2:23:5
the time is 14:8:60
请按任意键继续. . .
```

四、实验的分析与思考

1. 友元函数使用类的对象作为参数，以此访问到数据成员。
2. 运算符的重载。
3. 对象生命周期结束时，自行调用析构函数。