## 实验5 Object-Oriented Programming:Polymorphism

## 学号: 09017423 姓名：杨彬

### 实验目的

1． What polymorphism is, how it makes programming more convenient, and how it makes systems more extensible and maintainable.

2． To declare and use virtual functions to effect polymorphism.

3． The distinction between abstract and concrete classes.

4． To declare pure virtual functions to create abstract classes.

5． How C++ implements virtual functions and dynamic binding "under the hood."

6． How to use virtual destructors to ensure that all appropriate destructors run on an object.

### 实验内容

**Exp1:** **Description of the Problem**

英文：(Payroll System Modification) Modify the payroll system of Figs. 13.13~13.23 to include private data member birthDate in class Employee. Use class Date from Figs. 11.12~11.13 to represent an employee's birthday. Assume that payroll is processed once per month. Create a vector of Employee references to store the various employee objects. In a loop, calculate the payroll for each Employee (polymorphically), and add a $100.00 bonus to the person's payroll amount if the current month is the month in which the Employee's birthday occurs.

中文：修改图13.13~13.23的工资系统，增加private数据成员birthDate(Date对象), 要求使用图11.12~11.13的Date作为生日类型。假设工资系统每月处理一次，创建一个vector存储Employee指针来存储不同的员工对象，用一个循环计算每个员工的工资时(多态)，遇到当月过生日的员工多发100美元奖金。

**Exp2:**  **Description of the Problem**

(Shape Hierarchy) Implement the Shape hierarchy designed in Exercise 12.7 (which is based on the hierarchy in Fig. 12.3). Each TwoDimensionalShape should contain function getArea to calculate the area of the two-dimensional shape. Each ThreeDimensionalShape should have member functions getArea and getVolume to calculate the surface area and volume of the three-dimensional shape, respectively. Create a program that uses a vector of Shape pointers to objects of each concrete class in the hierarchy. The program should print the object to which each vector element points. Also, in the loop that processes all the shapes in the vector, determine whether each shape is a TwoDimensionalShape or a ThreeDimensionalShape. If a shape is a TwoDimensionalShape, display its area. If a shape is a ThreeDimensionalShape, display its area and volume.

**Exp3: Description of the Problem**

对实验8中的习题12.10进行修改。

创建一个与银行账户相关的类继承层次。银行的所有账户都可以存款和取款。存款能够产生一定的利息，查询和取款交易要缴纳一定的手续费。

要求：基类：Account(参考实验提示)；

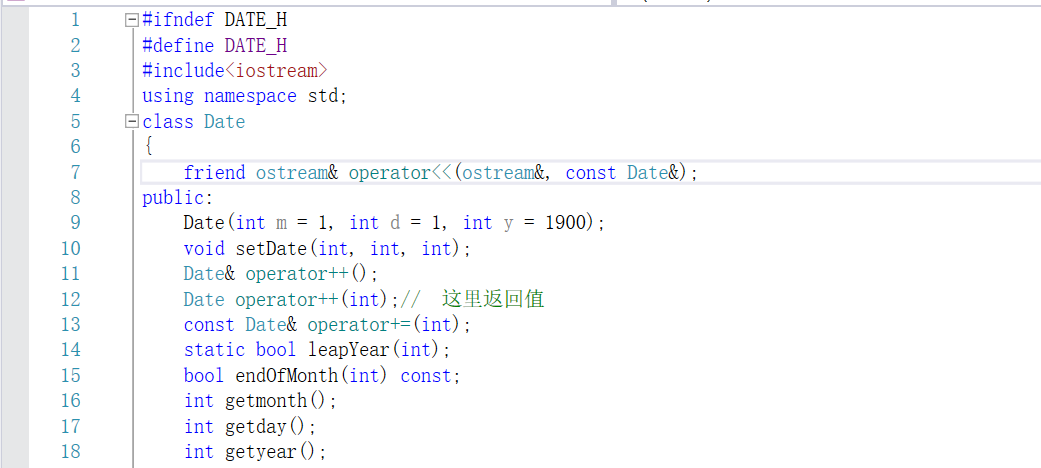
派生类：SavingAccount和CheckingAccount；

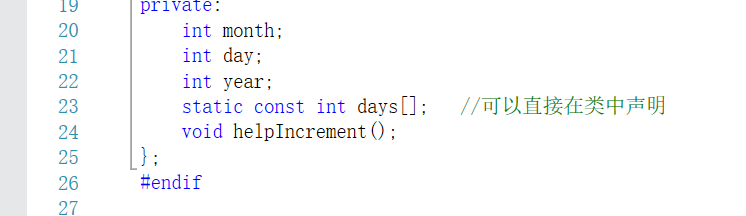
SavingAccount：继承Account的成员函数；构造函数接收两个参数：存款初始值（initialBalance）和利率（rate）；增加一个数据成员：利率(interestRate)，增加public类型的成员函数用于计算利率(calculateInterest())。

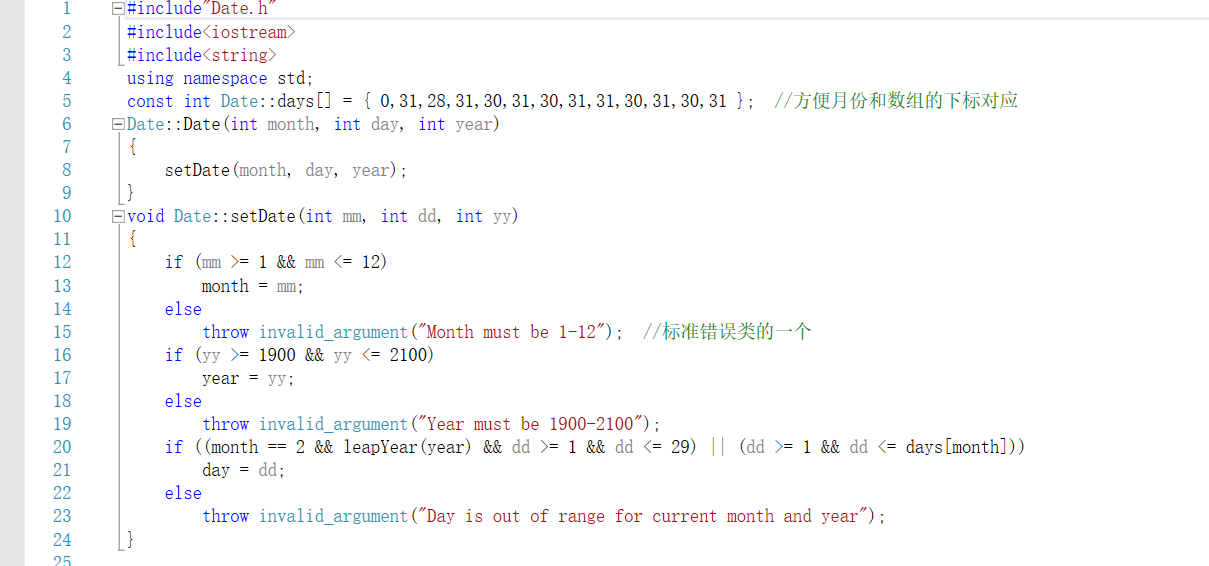
CheckingAccount：构造函数应接收到两个参数，一个是存款初始值（initialBalance），一个是手续费（fee）；增加一个数据成员：手续费（transactionFee）；重新定义成员函数credit()和debit()，以便能够从存款余额中减去手续费，要求成员函数通过调用基类的成员函数来更新存款数目，debit()函数应该在确定取款之后才能扣除手续费。

要求1：创建一个vector存储一组SavingAccount和CheckingAccount对象（多态），处理每一个账户时，判断该账户的类型，如果是SavingAccount，使用其成员函数calculateInterest()计算利率并加入账户，处理完一个账户，调用基类的成员函数getBalance() 打印其新的存款。

### 实验代码及结果：

**Exp1:** 

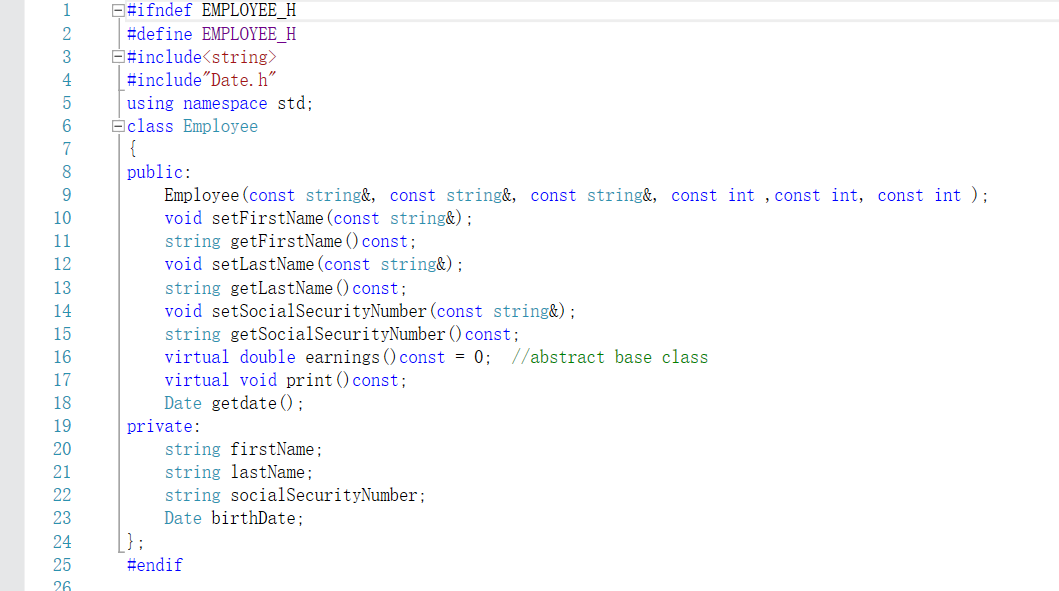






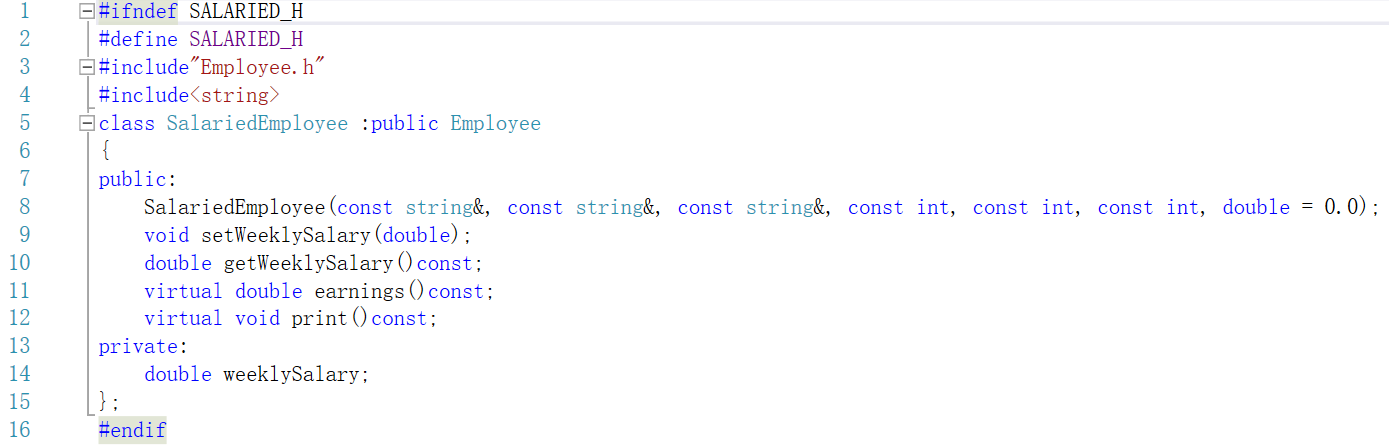








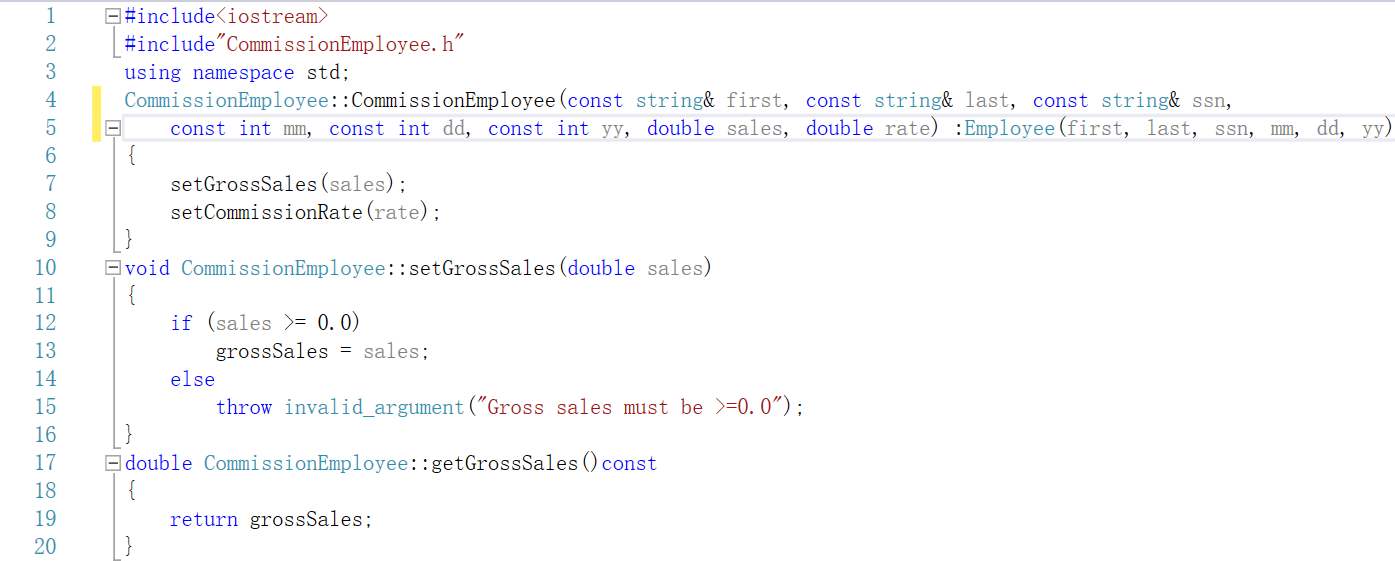


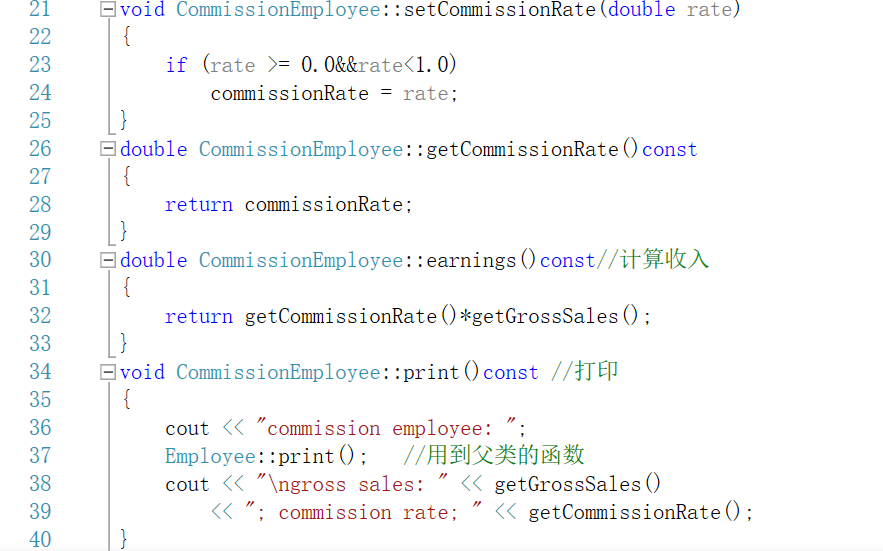


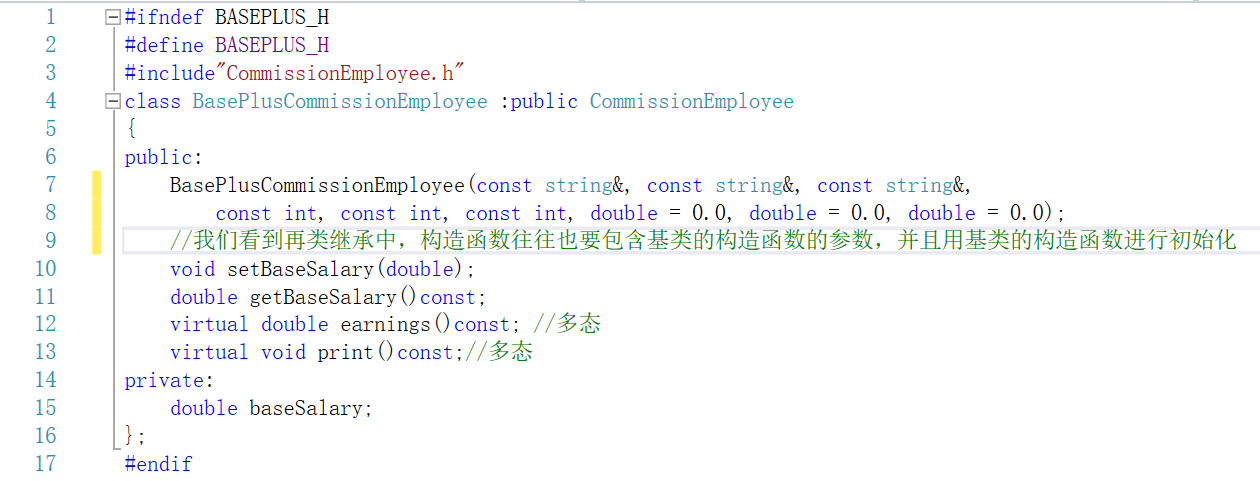








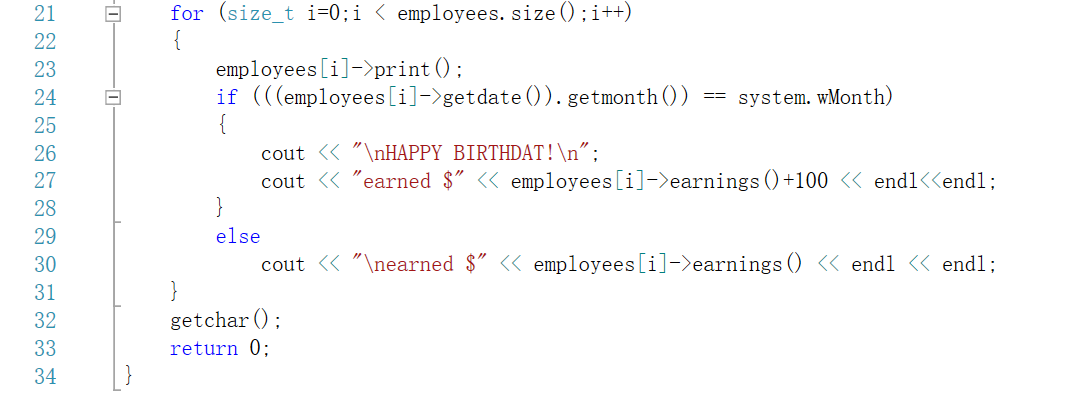


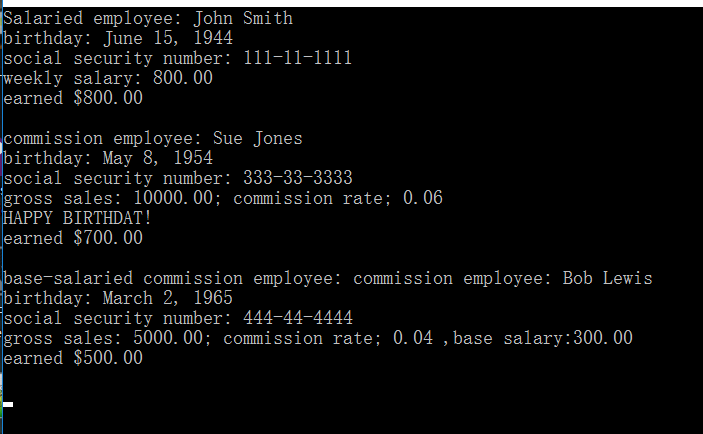


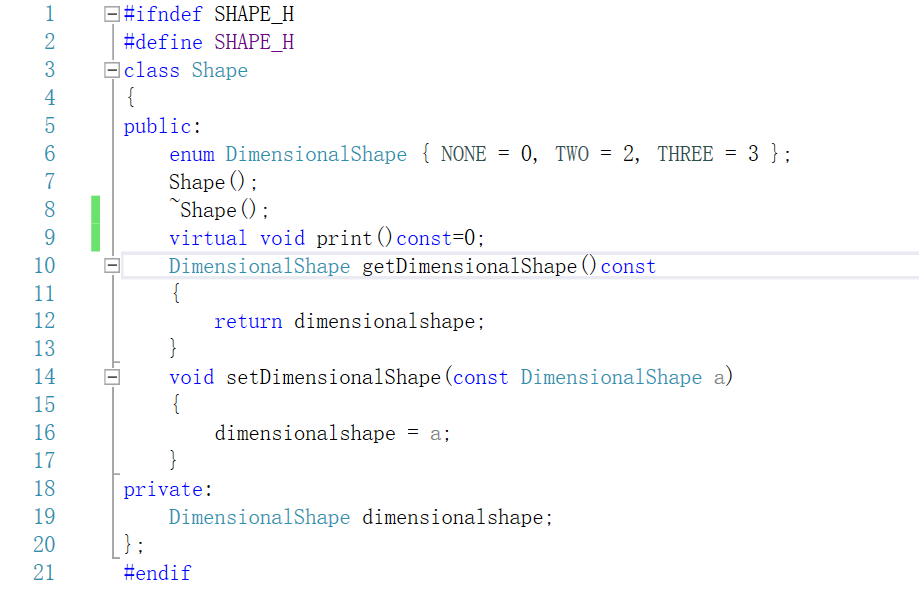


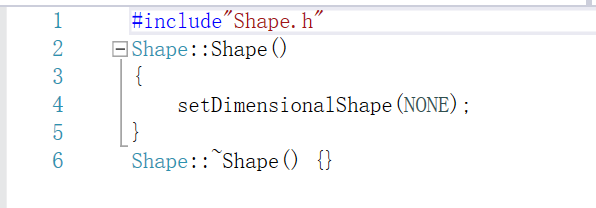


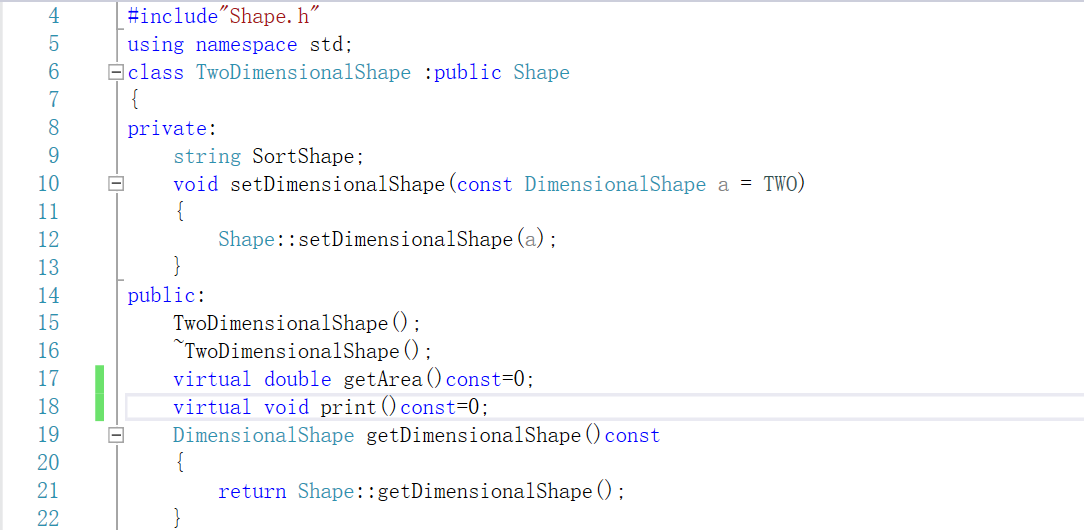




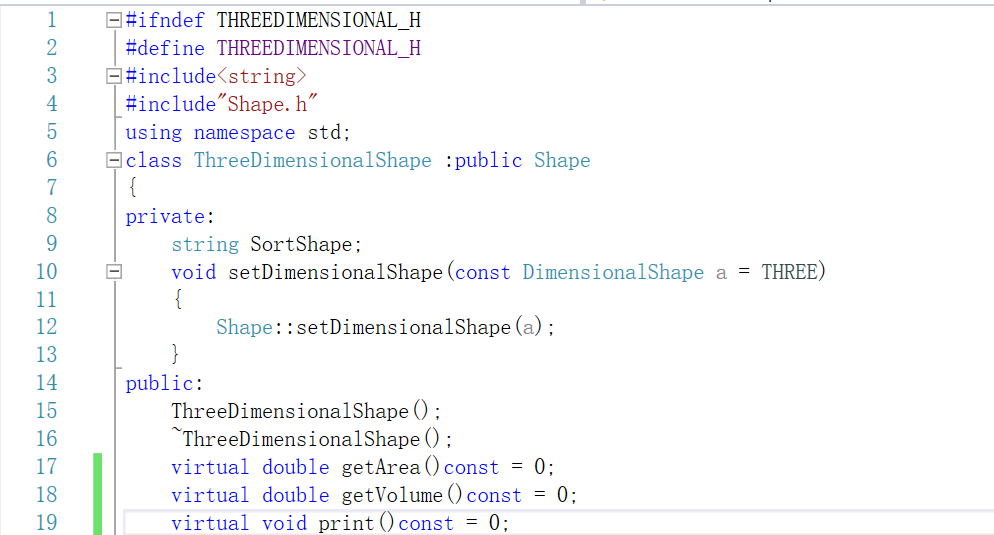


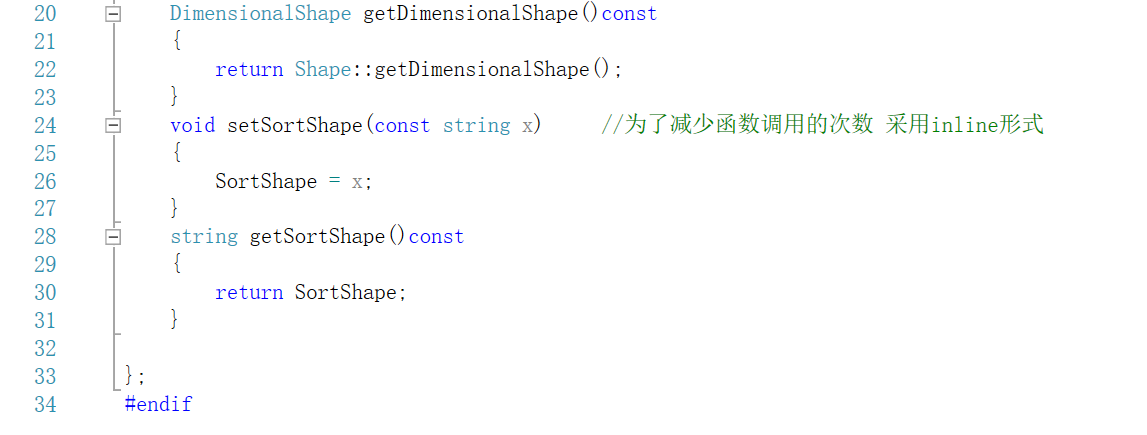
**Exp2:** 

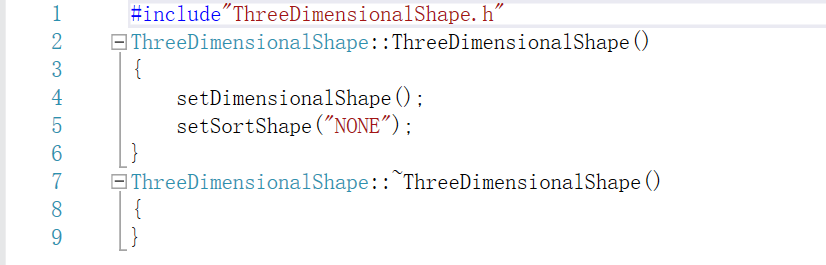


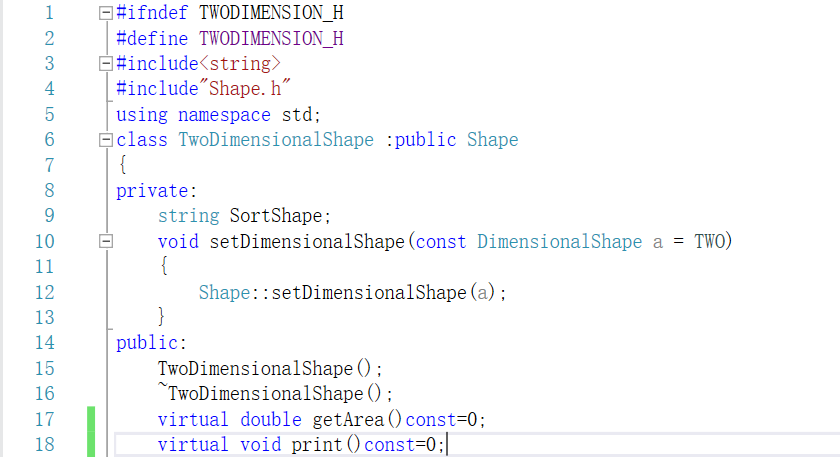




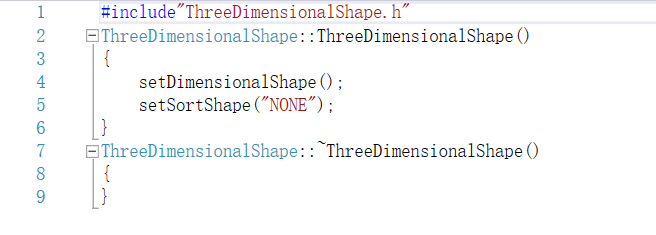


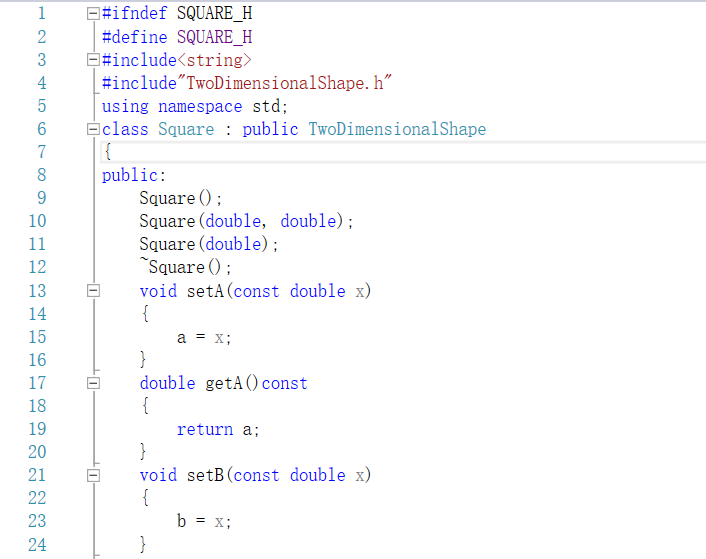


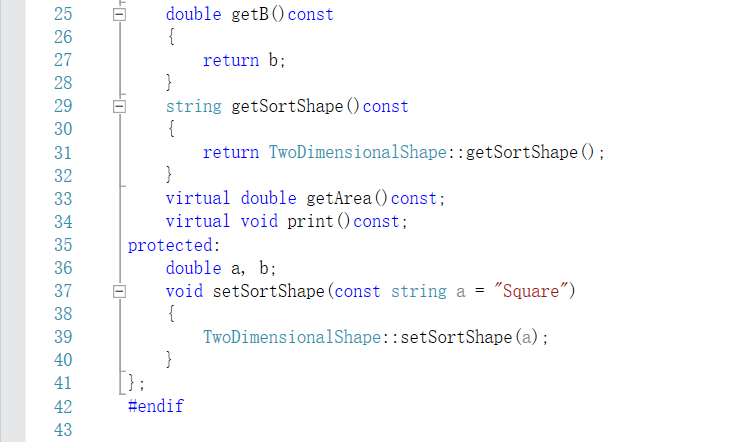


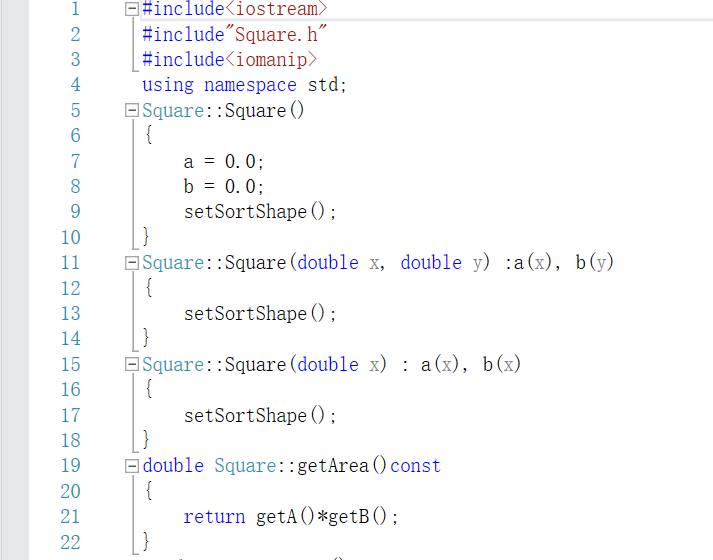




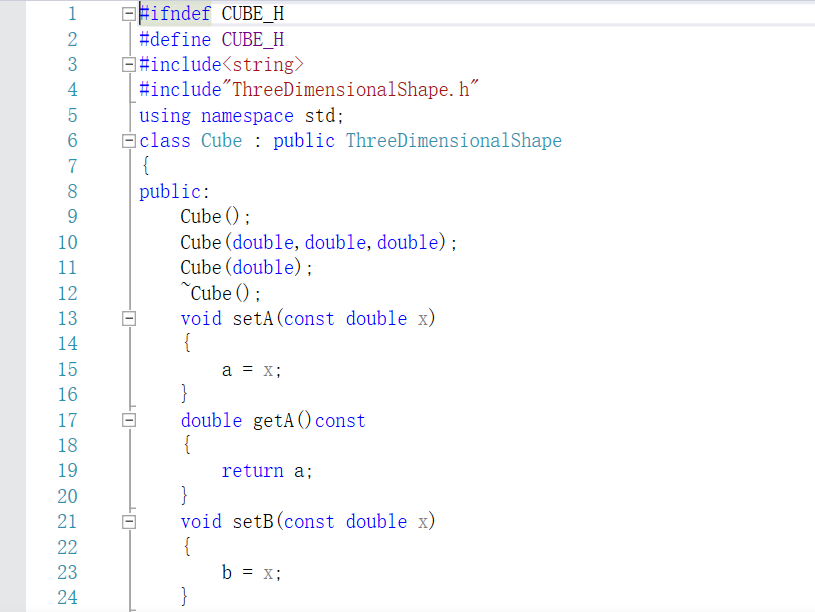


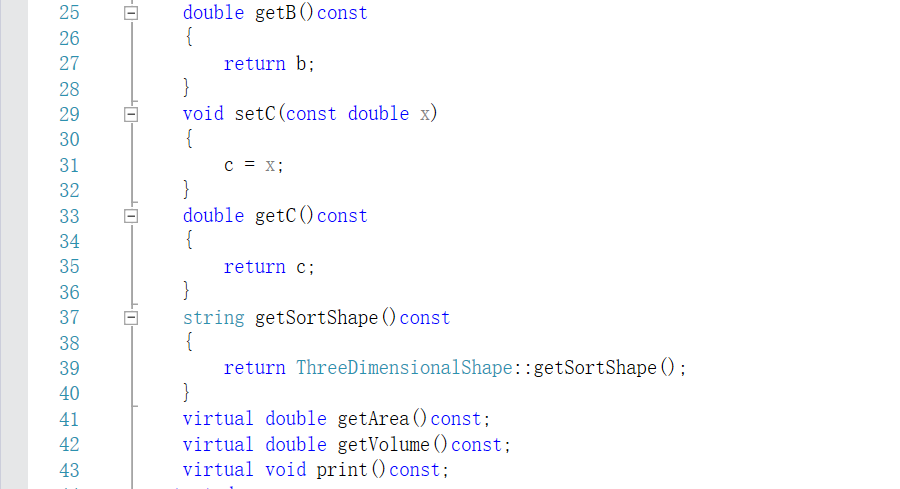


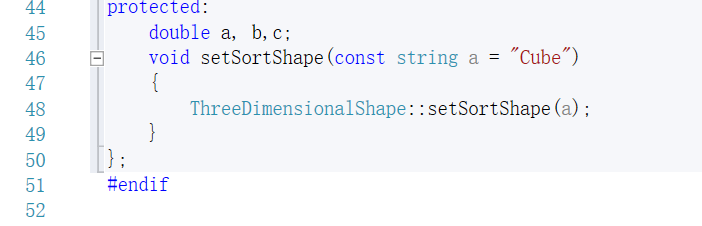


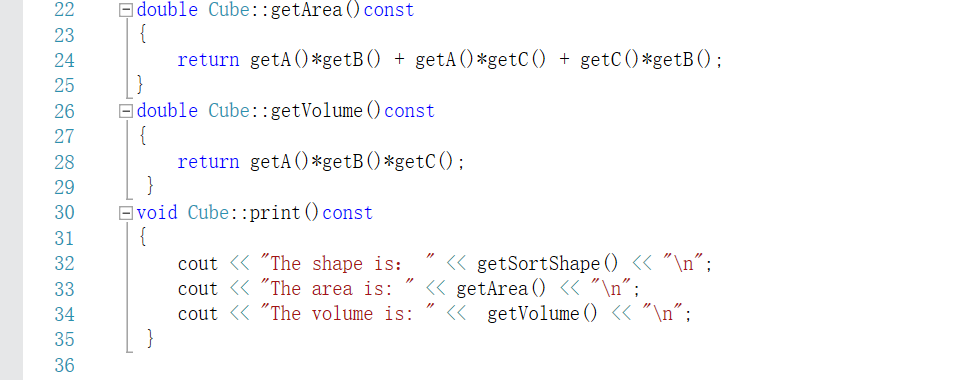


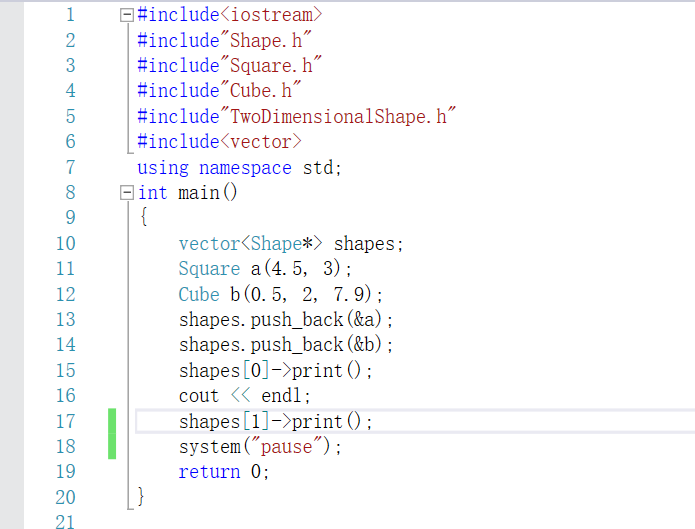


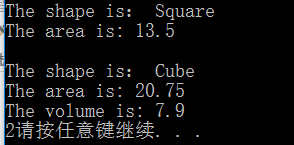


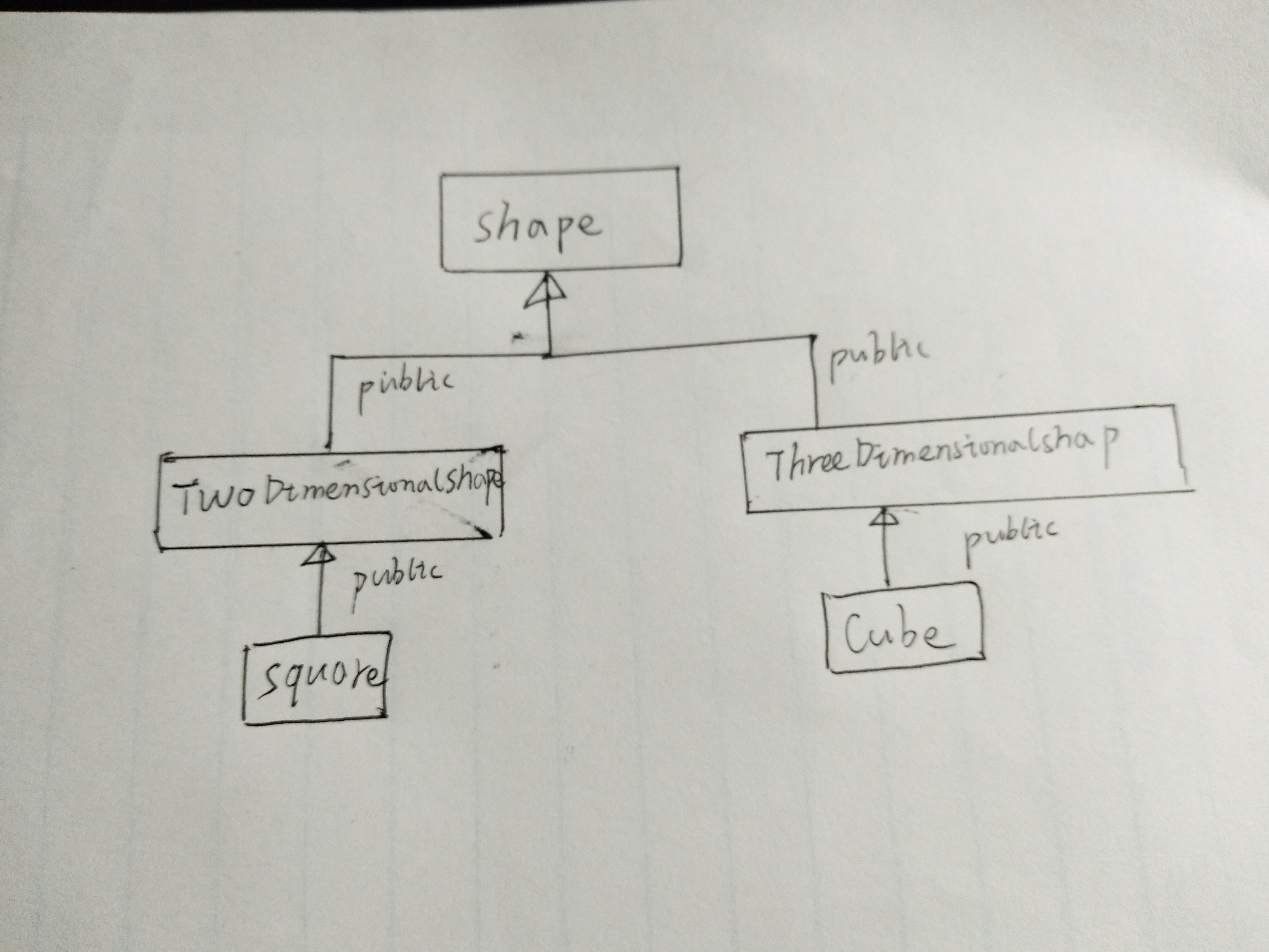




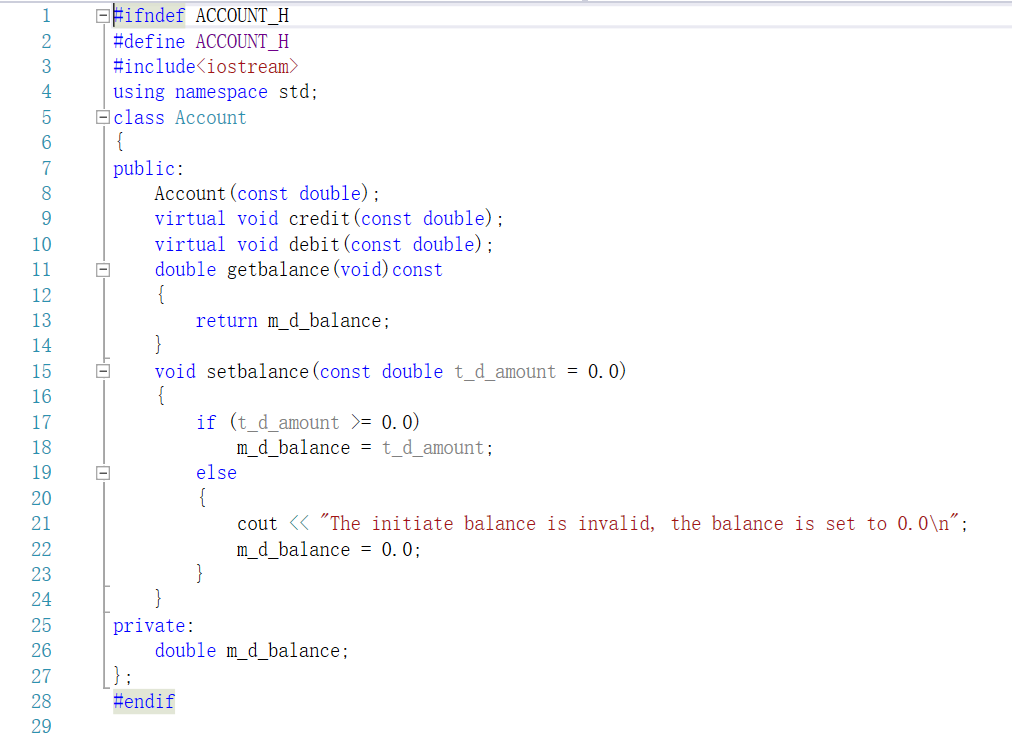


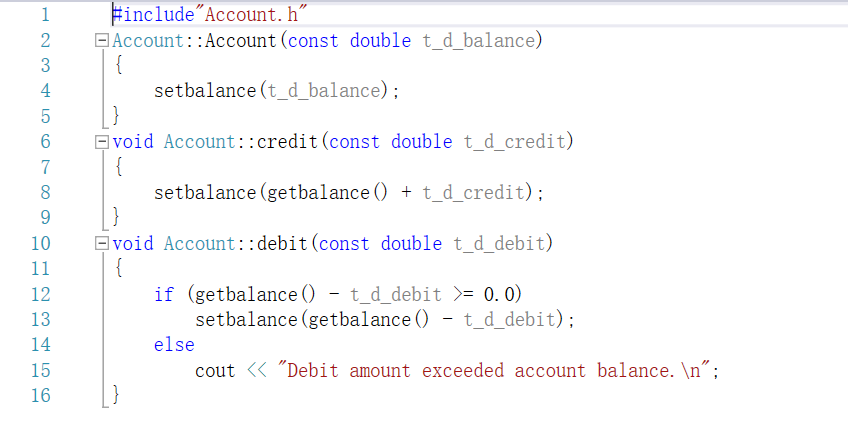


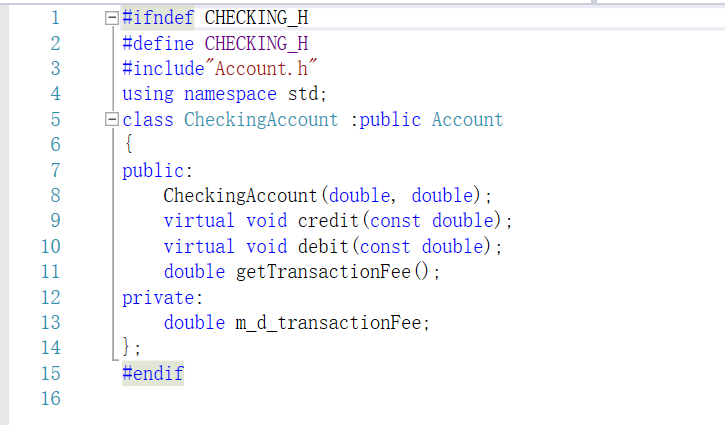


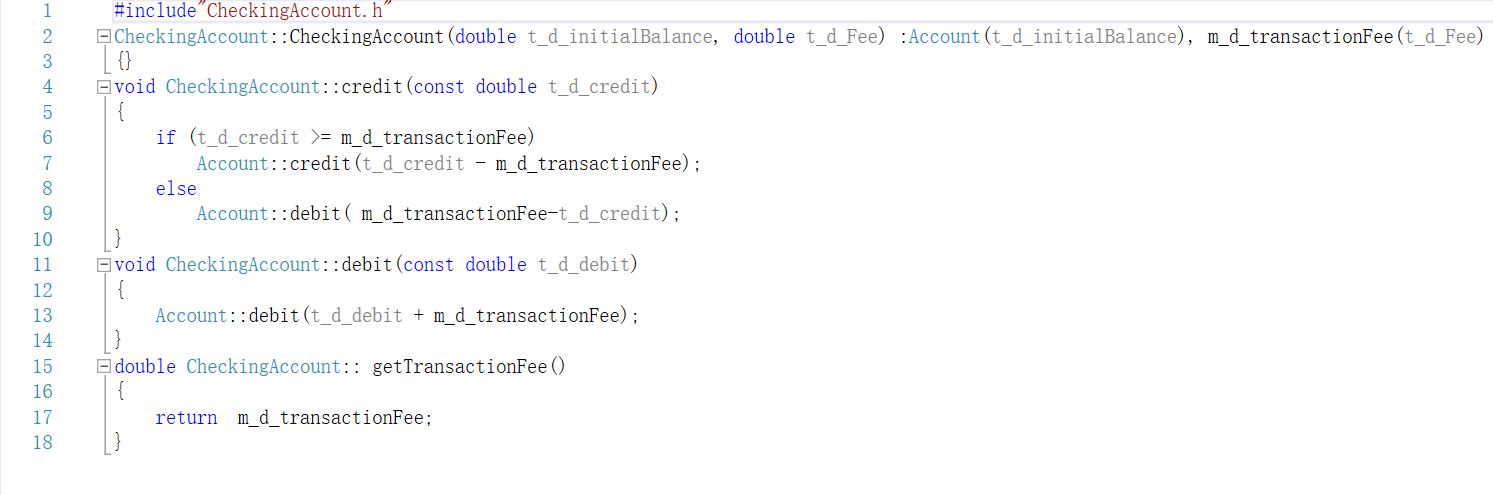
**类图：**

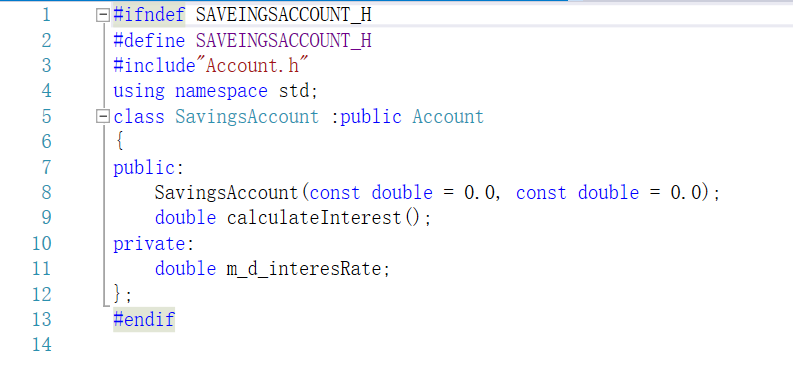
**Exp3:**

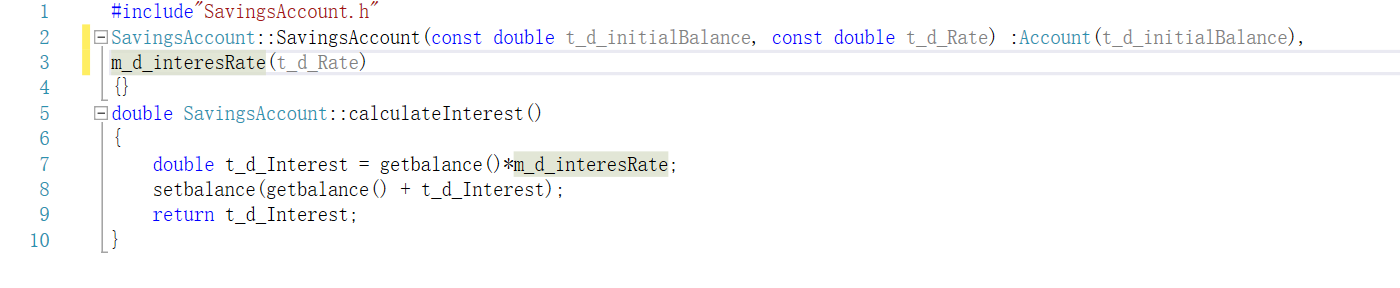


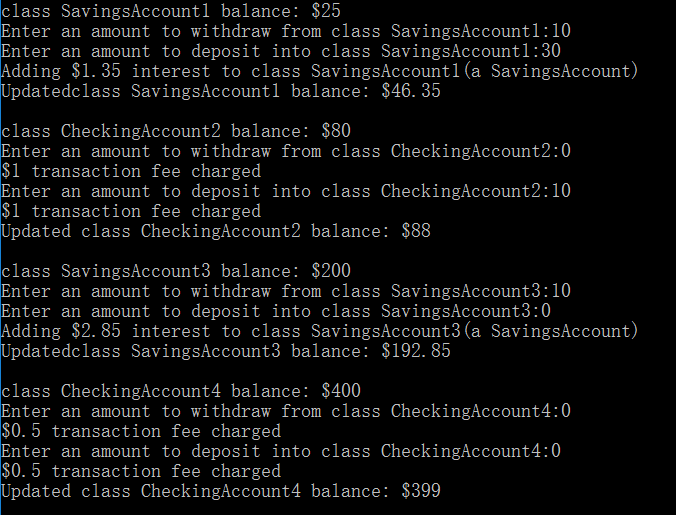
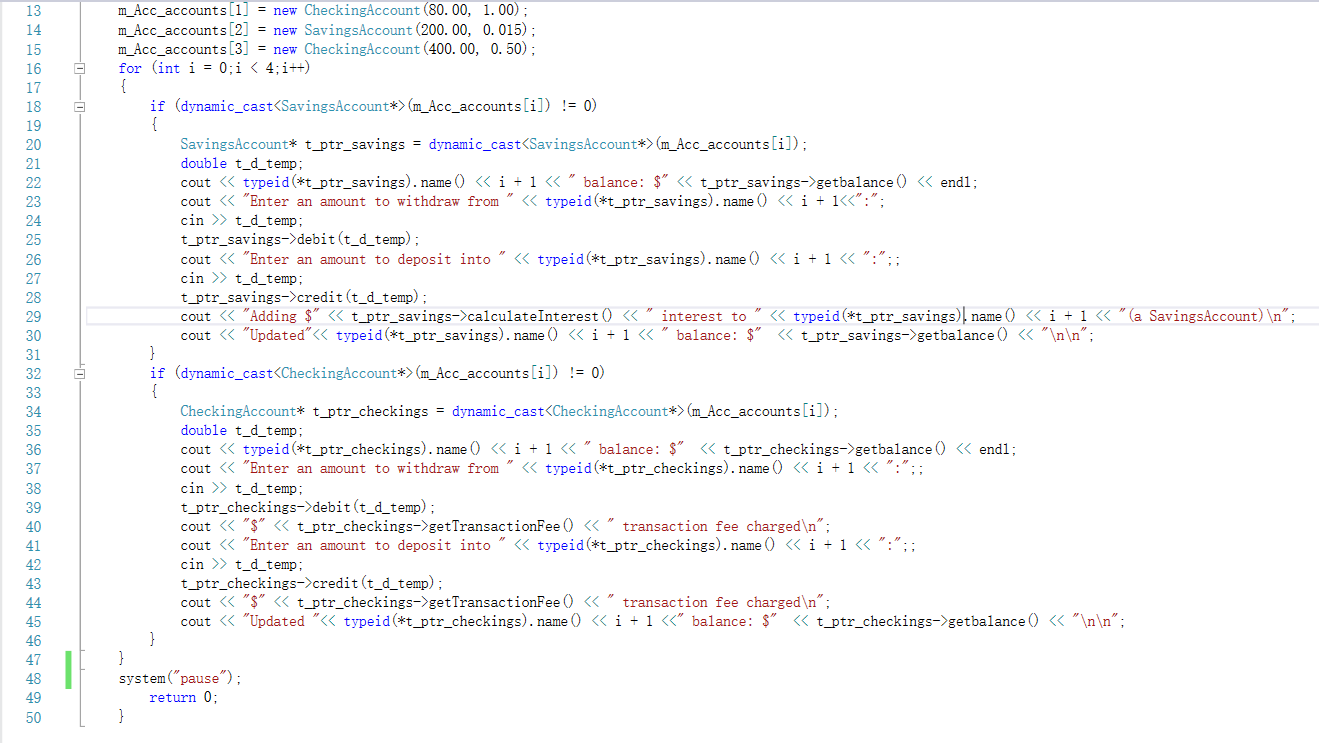






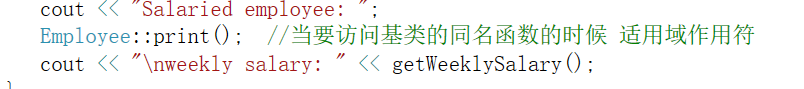


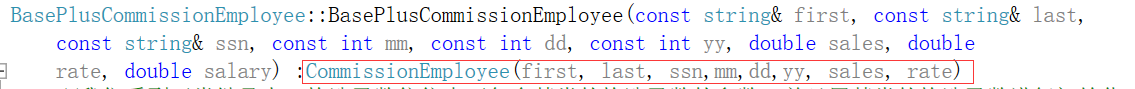




### 遇到的问题及解决过程

**Exp1:**1在vector向量的使用上出现了小错误，一开始在测试函数中我写了下面的句子因为受到平时定义数组的方式的影响，很容易这么写，结果导致之后赋值时候的错误，经过查阅网络关于vector的使用方法，我改成解决了问题

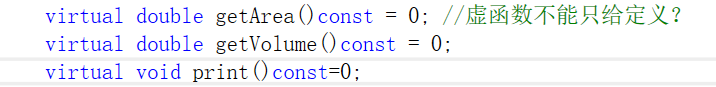
2在类继承的时候如果基类和派生类的函数重名了，那么就必须加上域的作用符。在派生类的print()函数的时候我没有考虑到这个问题，导致了错误。正确应该为：

3在类继承中，写派生类的构造函数的时候，忘了调用基类的构造函数。导致在测试的时候，使用print()函数（打印相关信息的时候出错）将代码改成：

4在使用虚函数的时候，类外定义的时候不需要在写上virtual否则会编译错误，将virtual去掉就解决了问题。

**Exp2:**1 在使用虚函数的时候出现了编译错误，因为在基类中只声明虚函数，而没有给出定义。虚函数不能只有声明而没有定义，后来我在基类虚函数的声明之后加上了空的定义，解决了问题。

2在1的基础上，我参考了教材，除了设置空定义，我还将基类的虚函数设置成了纯虚函数。（加上const=0）但是运行的时候出现了下面的错误

经过查找基类的头文件，我发现基类shape声明了3各纯虚函数但是我再派生类中只给出了两个纯虚函数的定义这就导致了派生类本身也是一个抽象类，不能被初始化，将基类的前两个纯虚函数去掉，改为放在下一个层次的派生类中即就解决了问题

2在类继承中数据的访问权限的问题。

在建立派生类的时候，适用了基类的私有数据，这是不允许的。因此编译器报错。我在基类添加了相关的get函数通过get函数来访问基类的数据就解决了问题。

**Exp3:**无

### 实验体会

**Exp1:** 1在类的继承和类的包含的时候，生和死的问题十分重要，其往往遵守尊老爱幼的原则，同时我们在定义派生类的构造函数的时候，往往也要调用基类的构造函数，和包含的类的构造函数，这样才能完整地实现对派生类的构造。同时在构造函数中最好使用初始化参数列表来进行初始化。

2在类继承中往往需要考虑因为继承方式的不同，以及继承前基类成员属性的不同，导致的派生类的权限问题。派生类不能直接访问基类的私有成员，必须通过基类的函数来访问。

3虚函数，不需要类外定义。在实际使用的时候，虚函数通过指针或者是引用的方法来实现多态。如果是直接用“.”运算符的话无法达到多态的效果。

**Exp2:**一在类的多态中关于虚函数使用的心得体会：1普通虚函数不能只有声明而没有定义。2虚函数分为两种纯虚函数和普通虚函数。纯虚函数（const=0）属于接口的继承，它只有声明没有定义，含有纯虚函数的类是抽象类不能被实例化，纯虚函数一般适用于基类是一个抽象的概念，没有必要也不应该被实例化（比如动物、形状）而且纯虚函数应该是在整个类继承体系中，所有相关的的基类和派生类所共有的属性或者函数。如果不是这样，很容易犯2的错误。纯虚函数是接口继承，派生类必须要对纯虚函数进行定义。而普通虚函数，含有普通虚函数的类可以被实例化，（即使其实现为空实现）。3为了方便阅读应该在所有的虚函数的声明前加上virtual 4通过指针或者引用来实现多态。

2在类继承中有一个很重要的问题，就是类的继承方式和继承后派生类对基类数据的权限。

特别是由于基类的私有成员，不能被派生类访问，因此这时候get 和set函数的价值就体现出来了，并且这也启发我们要合理使用protected这一数据属性。

**Exp3:** 通过这个实验，我知道了通过dynamic\_cast运算符将基类的指针转化成派生类的指针，并且用typied()得到类的名称，这对于在多态中对类的判断很有用。 

