## 虚函数

**虚函数的使用条件**所有虚函数必须有定义，不管是否用到。如果派生类中没有重新定义虚函数，那么派生类的虚函数会直接继承基类。

**多态性**：具有继承关系的多个类型统称为多态类型。

**动态绑定**：将静态类型的指针或引用绑定到动态类型的对象上。

**虚函数覆盖**：派生类重写基类的虚函数。派生类的函数的形参类型必须和被覆盖的函数一模一样，不然就是重载了那个函数。虚函数在派生类中仍然是虚函数，在派生类的派生类中仍旧可以覆盖。

**强制覆盖**：在派生类中定义虚函数时将override修饰符放在函数体前面，表示这个虚函数必须要在基类中定义过，强调覆盖而不是重载。

**拒绝覆盖**：在基类中定义，使派生类无法重载这个虚函数。

**虚函数中的默认实参**：请勿使用。

**虚函数的回避机制**：使用作用域运算符强迫执行虚函数的某个特定版本：double undiscounted = baseP->Quote::net\_price(42);，可以在派生类中调用基类的虚函数。

## 抽象基类

定义：含有（或未经覆盖直接继承的）纯虚函数的类是抽象基类，抽象基类不能定义对象。

**纯虚函数**：virtual ReturnType Function( ) = 0;

**重构**：重新设计类的体系以便将操作和/或数据从一个类移动到另一个类中。重构基类不用改动派生类的代码，但是必须重新编译。

## 访问控制

**可访问性**：

**受保护的成员**：用protected关键字来声明，对于类的用户不可访问，对于派生类的成员和友元是可访问的。派生类的成员或者友元只能通过派生对象来访问基类中的受保护成员。

**公有继承**：派生类的派生列表中的基类由public访问说明符修饰，能访问protected成员。派生访问说明符对于派生类的成员（及友元）能否访问直接基类的成员没有什么影响。主要是控制派生类用户对基类的成员的访问权限。