**Chapter 13 类继承 Class Inheritance**

1. 根本：类似于函数库，便于完整地代码再利用。基本的就是在现有的class中加入新的功能和资料，增量式修改

Base Class & Derived Class

1. 结构
2. //RatedPlayer derives from the TableTennisPlayer base class

//下面那行就是class的声明部分，由于是由一个object衍生出的并且刚好是逐值复制，所以可以使用copy constructor

Class RatedPlayer : public TableTennisPlayer

{

… //然后写在block里的内容是除了基类以外增加的private和public的内容

};

1. Base public ∈ derived public
2. Base private ∈ a complete package of derived class
3. Derived class需要自己的constructor
4. 派生类无法直接存取基类的private成员，必须通过基类的成员函数
5. 必须在构造派生类之前建立一个完整的基类
6. 构造函数：先构造基类，再构造派生类

析构函数：先析构派生类，再析构基类

1. 参数的传递

实参传递到派生类的形参后直接将需要的参数传到基类的形参

Eg. 以下是一个构造函数

Overdraft :: Overdraft (const char \*s, long an, double bal, double ml, double r) : BandAccount (s, an,bal)

如果没有明确呼叫基类的构造函数/没有构造函数的初始化字符串，程序会自动呼叫默认构造函数

1. 派生类与基类的特殊使用关系：
2. 派生类可以直接使用基类的成员函数
3. 基类的指针可以指向派生类的对象

基类的reference可以直接指向派生类的对象，无需明确的类型转换

但是基类的reference/pointer只能引用派生类中原来基类的成员函数

1. C++类继承的三种模式
2. Public
3. Is-a关系（is-a-kind-of）

所有在基类中能做到的事情在派生类中都能做到

1. 非is-like-a

不是比喻，不能移除基类中的属性

1. 非is-implemented-as-a

不能由array类别作为基类衍生其他类别

1. 非uses-a

比如电脑可以和打印机同时使用，但是从电脑类派生出打印机类没有意义

1. Protected
2. Private
3. 同名异式polymorphic
4. 两种实践 polymorphic public inheritance的方法
5. 在派生类中重新定义基类的成员函数
6. 使用虚方法
7. 虚拟成员函数virtual member function

Eg. 基类中有一个函数Brass :: ViewAcct ()，派生类中也有一个函数BrassPlus :: ViewAcct ()。

1. 用对象呼叫成员函数

会根据对象的类型来决定要使用基类/派生类的ViewAcct

1. 用reference或者指针呼叫成员函数

（1）！+virtual：根据指针的类型/引用的类型选择使用基类/派生类的ViewAcct

（2）+virtual：根据reference所参考的对象/指针指向的对象的类型选择~

1. 在基类中宣告一个虚拟析构函数，以确保当派生对象结束时，呼叫正确的顺序的析构函数
2. 编写任何一个class里的成员函数时，如果不是在class的block内写的务必加上“：：”
3. 需要虚拟析构函数

任何一个基类中务必定义一个virtual destructor

1. Binding
2. 静态联编 static binding/early binding：在编译时进行，非虚方法

动态联编 dynamic binding/late binding：在程序执行是进行，虚方法

1. 预设静态联编：效率&概念模式（只要有需要的函数保留虚拟标签即可的思想）
2. 指针和引用
3. 一般的指针/引用需要规定的形态所指向的东西相对应,但是基类指针指向派生类指针的时可以不用那么严格，会进行自动转换
4. 不用明确转换：向上转型upcasting：衍生类的reference或pointer转化成基类的

需要明确转换：向下转型downcasting：基类的reference或pointer转换成派生类的

1. 关于虚拟函数的几点注意事项
2. 构造函数

不能是virtual

1. 析构函数

一般都是虚构函数（在基类中）

1. 友元函数

不是成员函数

1. 重新定义成员函数

后定义的会遮蔽之前定义的

1. Protected存取

基类中的protected对继承类属于public，对外界属于private

使derived class可以存取非公开的内部函数

1. 抽象基类 Abstract base class

两个类的共有部分

对应的那两个类成为具体类别 concrete class

1. 纯虚拟函数pure virtual function：在函数声明的时候后面加=0

Eg: Virtual double Area () const = 0; //a pure virtual function

对于名副其实的抽象基础类，至少要有一个纯虚拟函数

1. 在基类中不提供定义，但在派生类中必须提供定义
2. 继承和动态内存分配

基类使用动态内存分配

1. 派生类未使用动态内存分配

什么事都没有

1. 派生类使用了动态内存分配

在哪个类里new的在哪里delete