第四章习题心得：派生继承，派生继承理解有点难，编写程序时花费时间且所收获成果较少，但结果较好。

基类中的私有成员。无论哪种继承方式,基类中的私有成员不允许派生类继承,即在派生类中是不可直接访问的。

基类中的公有成员。当类的继承方式为公有继承时,基类中的所有公有成员在派生类中仍以公有成员的身份出现,在派生类内部和派生类外部都可以访问这些成员;当类的继承方式为私有继承时,基类中的所有公有成员在派生类中都以私有成员的身份出现，在派生类内可以访问这些成员,但派生类外部不能访问它们;当类的继承方式为保护继承时,基类中的所有公有成员在派生类中都以保护成员的身份出现，在派生类内可以访问这些成员,但派生类外部不能访问它们,而在下一层派生类内可以访问它们。

基类中的保护成员。当类的继承方式为公有继承时,基类中的所有保护成员在派生类中仍以保护成员的身份出现,在派生类内可以访问这些成员,但派生类外部不能访问它们,而在下一层派生类内可以访问它们;当类的继承方式为私有继承时,基类中的所有保护成员在派生类中都以私有成员的身份出现，在派生类内可以访问这些成员,但派生类外部不能访问它们;当类的继承方式为保护继承时,基类中的所有保护成员在派生类中仍以保护成员的身份出现,在派生类内可以访问这些成员,但派生类外部不能访问它们,而在下一层派生类内可以访问它们。

当基类的构造函数没有参数﹐或没有显式定义构造函数时,派生类可以不向基类传递参数,甚至可以不定义构造函数。

当基类含有带参数的构造函数时,派生类必须定义构造函数﹐以提供把参数传递给基类构造函数的途径。

在C++中,派生类构造函数的一般格式为:派生类名(参数总表):基类名(参数表)

派生类新增数据成员的初始化语句

其中基类构造函数的参数,通常来源于派生类构造函数的参数总表,也可以用常数值。

在派生类中可以根据需要定义自己的析构函数﹐用来对派生类中增加的成员进行清理工作。基类的清理工作仍然由基类的析构函数负责。由于析构函数是不带参数的,在派生类中是否要自定义析构函数与它所属基类的析构函数无关。在执行派生类的析构函数时，系统会自动调用基类的析构函数,对基类的对象进行清理。

当一个派生类具有多个基类时,这种派生方法称为多继承。

多重继承的构造函数的执行顺序与单继承构造函数的执行顺序相同,也是遵循先执行基类的构造函数,再执行对象成员的构造函数﹐最后执行派生类构造函数体的原则。处于同一层次的各个基类构造函数的执行顺序,取决于声明派生类时所指定的各个基类的顺序,与派生类构造函数中所定义的成员初始化列表的各项顺序没有关系。析构函数的执行顺序则刚好与构造函数的执行顺序相反。

当引用派生类的成员时,首先在派生类自身的作用域中寻找这个成员,如果没有找到,则到它的基类中寻找。如果一个派生类是从多个基类派生出来的,而这些基类又有一个共同的基类,则在这个派生类中访问这个共同的基类中的成员时,可能会产生二义性。为了解决这种二义性,C++引入了虚基类的概念。

Copyright IMG_2562021-2099 wangzezheng. All rights reserved