|  |  |
| --- | --- |
| 1. 继承的构造函数 2. 继承的构造函数的特点 | Derived(parms):base(args){}  Using声明  Public:  Using Disc\_quote::Disc\_quote; |
| 自己的理解：  就是在基类中有许多构造函数，在派生类中若想使用而不用显示调用，直接using 基类：：基类构造函数  则编译器都生成一个与之对应的派生类构造函数，无需显示定义派生类的构造函数，可以使用派生类的构造函数，并且调用父类的相应构造函数，生成的派生类构造函数的个数与参数与基类的一致   1. 继承的构造函数 2. 这些定义的构造函数不是默认的构造函数，是新定义的构造函数 3. 类能够重用其直接基类定义的构造函数。 4. 这些构造函数并非以常规方式继承而来的，为了方便，姑且陈为：继承的   1.一个类只继承其直接基类的构造函数  2.类不能继承默认、拷贝和移动构造函数  3.如果派生类没有直接定义这些构造函数，则编译器将为派生类合成它们   1. 使用using声明语句对基类的构造函数声明，则再派生类中让编译器产生代码。 2. 对于基类的每个构造函数（除了上述情况2外）编译器都生成一个与之对应的派生类构造函数，形参列表完全相同的构造函数 3. 但代码中却可以实现直接基类的拷贝构造函数，间接基类不行 4. derived(parms) : base(args){ } 5. tips:如果派生类含有自己的数据成员，则这些成员将被默认初始化，因为上面的构造函数并不会执行派生类的成员初始化 6. 继承的构造函数的特点 7. using声明不能指定explicit或constexpr，若基类的构造函数是explicit或constexpr的则继承的构造函数也拥有相同的属性 8. 若一个基类构造函数含有默认实参，则不会继承这个构造函数，而是生成多个继承的构造函数，其中每个构造函数分别省略掉一个含有默认实参的形参。   如：基类的构造函数有两个形参，第二个形参含有默认实参的形参  则：派生类获得两个构造函数：  一个构造函数接受两个形参（没有默认实参）  另外一个构造函数只接受一个形参，对应与基类中最左侧的没有默认值的那个形参  基类的定义部分不变！  问：若两个参数都有默认实参怎么办。还是生成两个   1. 除了两个例外，using都会继承这些构造函数 2. 派生类定义了与派生类相同的参数列表，则这些构造函数不会被继承 替换继承而来的构造函数 3. 是默认、拷贝和移动构造函数不会被继承。因为在派生类中这些函数按照正常规则合成。继承的构造函数不会被作为用户定义的构造函数来使用.则像a一样会替换继承而来的构造函数。即使派生类只含有继承的构造函数，它也有一个合成的默认构造函数 | |