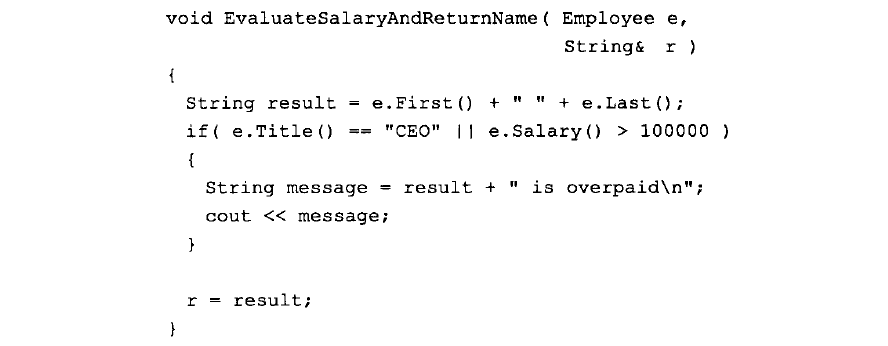
条款10.STL中的Queue将弹出元素分为两步——top,pop；目的之一是为了避免削弱函数的异常安全性

1. 条款14.很多实现中都将类分成接口和具体实现，比如Stack和Stack\_impl，主要是为了进行异常的分别单独处理，将异常处理放在impl中。同时在此例中还有个作用是，将内存管理和对象的构造/析构这两个方面分离开
2. 条款23.如果里面的impl声明为指针，则是由于impl指向的对象存放具体实现的私有成员函数和私有数据成员，为了方便对这些成员更改时，客户代码（使用这个类的代码）不需要重新编译。

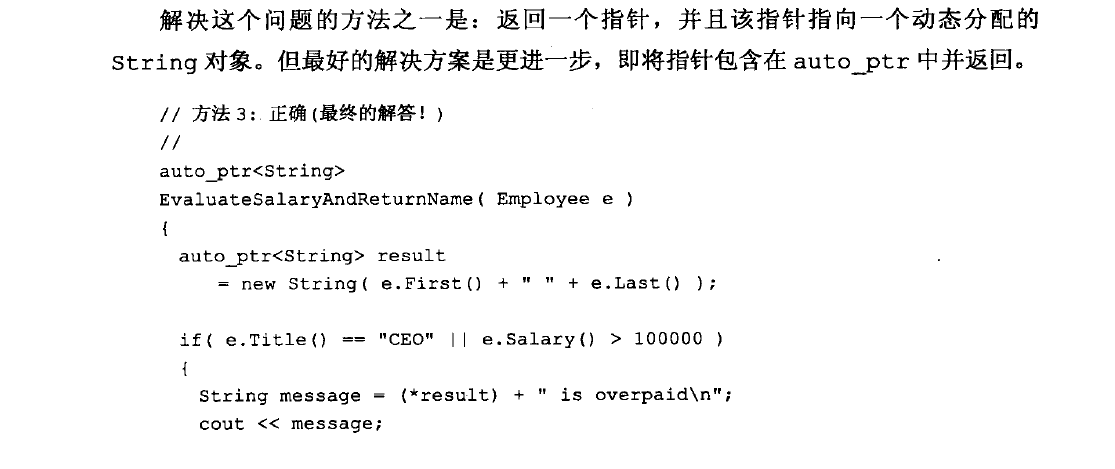
条款16.引入异常的初衷之一就是为了使构造函数和析构函数能够报告失败（因为这些函数不能返回一个值）（注：为什么构造函数没有返回值，一种说法是为了避免语法混乱,如C c = C()，如果有返回值此例不成立，同时也可以说明返回值是对象本身）

条款19.

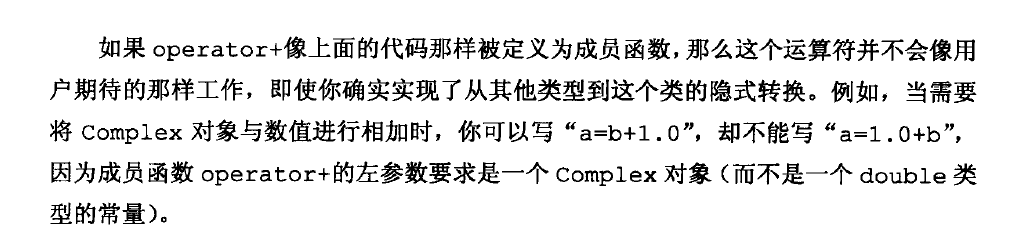


在某些设计中不用返回值返回某个值，而是通过参数传递一个引用来获得这个值，其中一个目的是在通过返回值返回一个对象值时，一般会带来拷贝构造和拷贝赋值两个操作，有可能会出错，但是这个已经出了函数管理的范围，现在是由函数来保证正确的获得这个值

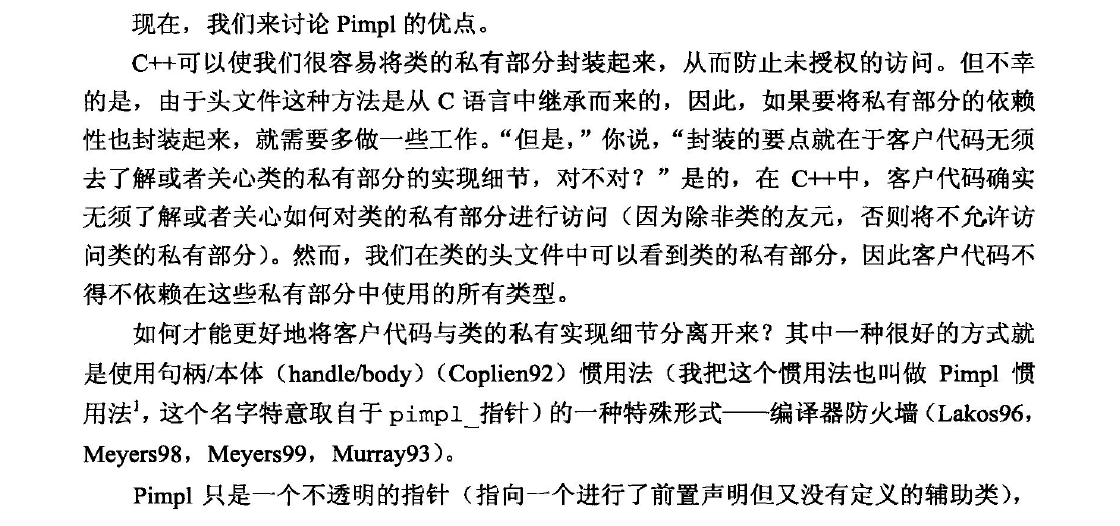
（注：一个更好的解法，到达commit-or-rollback的语义，利用智能指针）

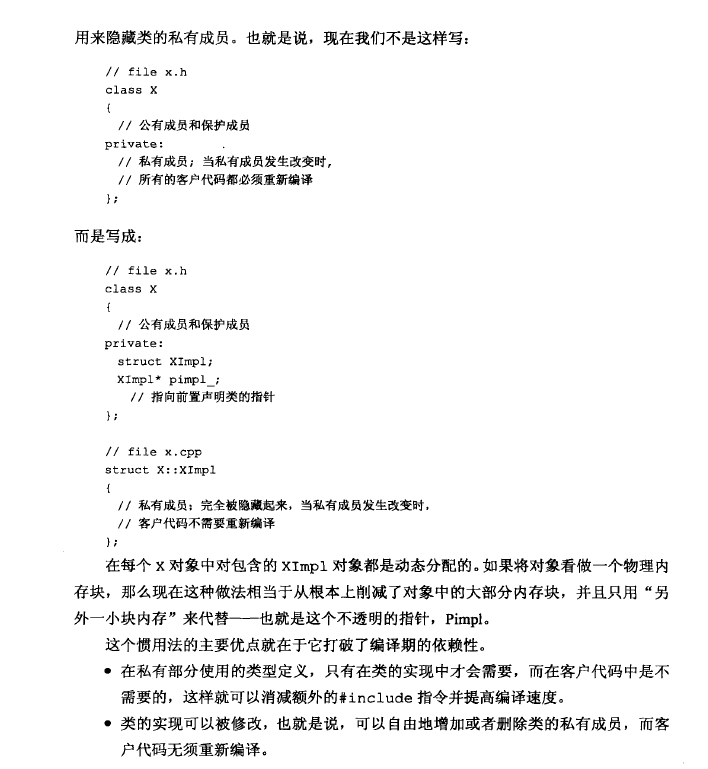


条款20：应该尽量使用+=而不是使用+，因为后者是用前者实现的，同时用+会产生一个临时变量。Operator+不应该申明为成员函数。

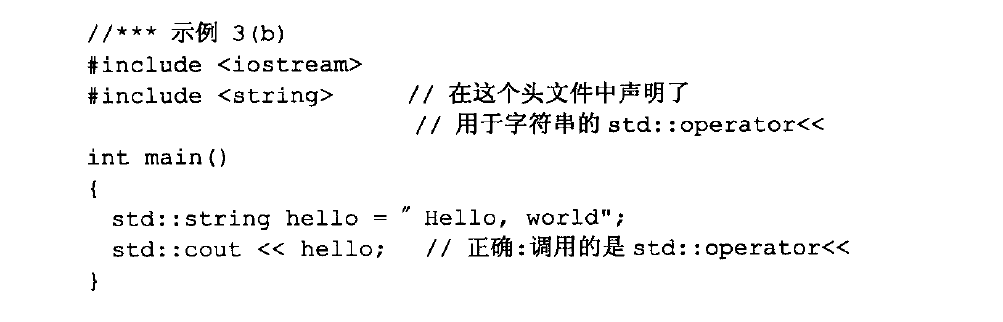


条款27.类中使用pimpl的好处，隐藏实现细节，降低编译依赖





### 条款32.

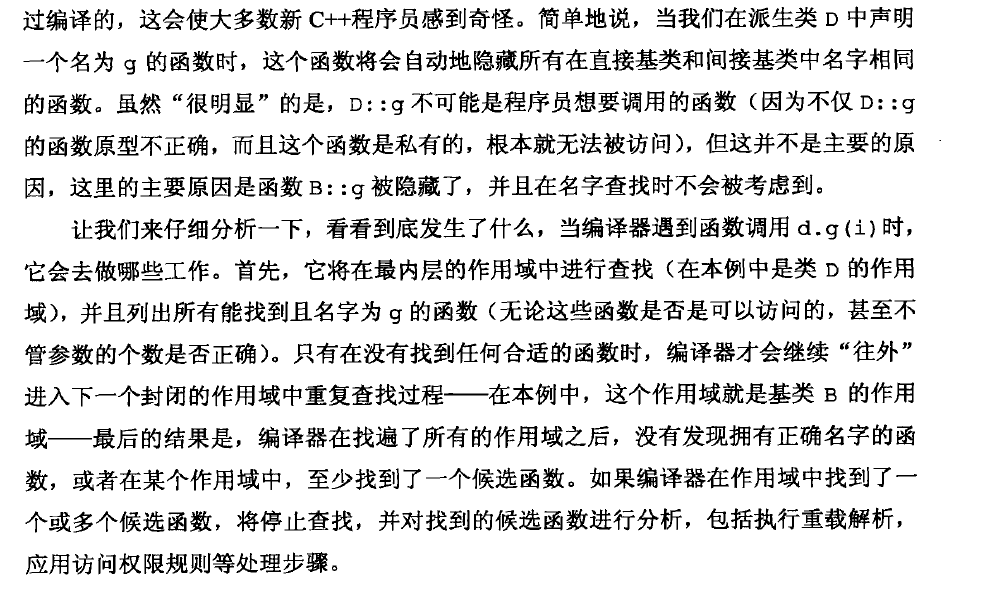


对于上述的std::cout<<hello如何找到string类中重载的<<，与Koenig查找规则有关，而不需要指明<<所在的命名空间std，函数参数类型所在的命名空间也会参与检索，上面string所在的std也会被查找。

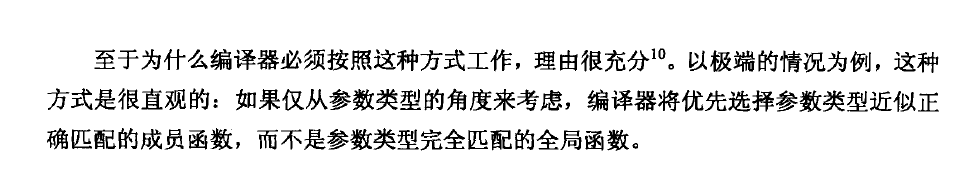
### 条款34.

子类中同名的函数会隐藏父类的所有同名函数（即所有重载的所有函数）

查找过程，先找函数，在判断权限是否可访问



查找理由：



条款43

Const 若只是修饰函数值传递的参数，那么它不具有重载的功能

