感想与心得

在这一章中的我理解到了多态性的内容

多态性可以简单地概括为“一个接口，多种方法”，它是面向对象编程领域的核心概念。 多态性指相同对象收到不同消息或不同对象收到相同消息时产生不同的实现动作。 C++支持两种多态性：编译时多态性，运行时多态性。

由习题中的内容得：

C++语言对运算符重载制定了以下一些规则:  
(1) C++中绝大部分的运算符允许重载，不能重载的运算符只有少数几个。  
(2) C++语言中只能对已有的C++运算符进行重载，不允许用户自己定义新的运算符。(3)运算符重载是针对新类型数据的实际需要，对原有运算符进行适当的改造完成的。一般来讲，重载的功能应当与原有的功能相类似(如用“+”实现加法，用“一”实现减法)。  
(4)重载不能改变运算符的操作对象(即操作数)的个数。  
(5)重载不能改变运算符原有的优先级。  
(6)重载不能改变运算符原有的结合特性。  
(7)运算符重载函数的参数至少应有一个是类对象(或类对象的引用)。  
(8)运算符重载函数可以是普通函数，也可以是类的成员函数，还可以是类的友元函数。  
(9)一般而言,用于类对象的运算符必须重载，但是赋值运算符“一”例外，不必用户进行重载。但在某些情况下，例如数据成员中包含指向动态分配内存的指针成员时，使用系统提供的对象赋值运算符函数就不能满足程序要求，在赋值时可能出现的错误。在这种情况下，就需要用户自己编写赋值运算符重载函数。习题中对静态联编、动态联编，编译时的多态性和运行中的多态性的区别以及实现方法的不同，运算符重载的规则，友元运算符重载函数和成员运算符函数的不同，虚函数的定义，虚函数与函数重载的相同点和不同点进行大致了解，进一步强调本节课重点，加强对概念的理解。C++语言中允许在重载运算符中改变运算符原来的功能；不能用友元函数重载“=”“[]”和->.。C++规定构造函数不能是虚函数，而析构函数可以是虚构函数。C++固定虚函数在派生类中重新定义时，其函数类型，包括函数类型、函数名，参数个数、参数类型顺序，都必须与基类中的原型完全相同。如果在抽象类的派生类中没有重新说明纯虚函数，则该函数在派生类中仍然为纯虚函数。