我们学了类的多态、多态性、运算符重载、虚函数等知识点。所谓多态性，简单说来就是一词多义。多态性是面向程序设计的重要特征之一。多态性有两种表现形式。一种是不同的对象在收到相同的消息时，产生不同的动作。同一对象收到相同的消息却产生不同的函数调用。所谓绑定，就是要函数调用与函数体本身产生关联，在编译时确定，称之为“早期绑定”，在程序运行时确定，称之为“晚期绑定”，C++的多态性在“早期绑定”和“晚期绑定”两方面都有体现。多态性具有编译时多态和运行时多态两种实现方式：编译时多态：也叫静态多态性，属于早期绑定，在编译时就实现了绑定，它是静态联编的；主要通过函数重载完成。运行时多态：也叫动态多态性，属于晚期绑定，在编译时还无法确定绑定对象，只有在运行时才能够实现绑定，它是动态联编的；主要通过虚函数重载完成。目前出现的问题：

系统已定义的运算符不适用于新的自定义数据类型 例如. 复数的运算。C++中绝大部分的运算符允许重载，除了：. 成员访问运算符 .\* 成员访问指针运算符 :： 作用域运算符 Sizeof 长度运算符 ？: 条件运算符。C++只能对已有的C++运算符进行重载，不允许用户自定义新的运算符。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。虚函数是在基类中使用了关键字virtual的成员函数，并在派生类中重新定义的函数。虚函数是实现多态的手段，虚函数声明时指出，在类外定义时不加virtual。派生类中对虚函数的超载或覆盖（override）不再加virtual，但函数头要完全相同，虚函数在所有派生类中均保持虚函数的特征。

Copyright ©2021-2099 Haotian Wang. All rights reserved