多态通过分离做什么和怎么做，从另一角度将接口和实现分离开来。

多态能够改善代码的组织结构和可读性，能够创建可扩展的程序

与封装的关系

封装通过合并特征和行为来创建新的数据类型。”实现隐藏”则通过将细节”私有化”把接口和实现分离开来

多态的作用则是消除类型之间的耦合关系。多态方法调用允许一种类型表现出和其他相似类型之间的区别，只要它们都是从同一基类导出而来的。这种区别是根据方法行为的不同而表示出来的，虽然这些方法都可以通过同一个基类来调用

对象既可以作为它自己本身的类型使用，也可以作为它的基类型使用。而这种把对某个对象的引用视为对其基类型的引用的做法被称为向上转型

将一个方法调用同一个方法主体关联起来称作绑定。

若在程序执行前进行绑定，叫做前期绑定

后期绑定，它的含义是在运行时根据对象的类型进行绑定。后期绑定也叫做动态绑定或运行时绑定。

Java中除了static方法和final方法(private方法属于final方法)之外,其他所有的方法都是后期绑定。这意味着通常情况下，我们不必判定是否应该进行后期绑定—它会自动发生

为什么要为某个方法声明为final呢？

第一：它可以防止其他人覆盖该方法

第二：可以有效的”关闭”动态绑定，或者说告诉编译器不需要对其进行动态绑定。这样编译器就可以为final方法调用生成更有效的代码。然而这样做对程序的整体性能不会有什么改观。所以根据设计来决定是否使用final,而不是出于试图提高性能的目的来使用final。