多态性有两个方面

1. 方法的多态性

1，方法的重载

2，方法的覆写

1. 对象的多态性——父子实例对象之间的转换处理

**！！能调用哪些方法取决于引用，具体调用哪个方法取决于实例对象**

1. 对象向上转型： 父类 父类引用 =子类实例 （父类引用指向子类对象）

**使用前提：①存在继承关系 ②子类覆写了父类的方法**

例子1

Animal有run()，Animal的子类有Dog，Cat，Person，这三个子类都覆写了run()。

Animal a；

a可以指向Dog对象，Cat对象，Person对象，可以调用这三个对象各自的run()

a指向Dog对象，那么a.run()调用的就是Dog的run()

例子2：

public void fun(Animal a)———该方法可以接受Dog对象，Cat对象，Person对象

对象向上转型的好处：减少代码量，做出可维护的设计，实现接受或返回参数的统一性

1. 对象向下转型： 子类 子类引用=（子类） 父类实例——强制转换

正确例子：

父类Man有方法print()，子类SuperMan()，扩展了方法fly()

Man man=new SuperMan();——向上转型

man.print();

SuperMan super=(SuperMan)man;——向下转型

super.fly();

但是向下转向并不是一件安全的事，**因为对象向下转型之前一定要经历过向上转型**

错误例子:

Man per=new Man();

per.print();

SuperMan man=(SuperMan)per;————抛出ClassCastException，因为引用与对象不兼容（子类引用不能指向父类实例对象）

Man.fly();

为了防止向下转型抛出异常，我们需要instanceof，在向下转型之前用instanceof判断一次——具体细节看instaceof

所以对错误例子进行修改

Man per=new Man(); Per.print();

if(per instanceof SuperMan) {SuperMan man=(SuperMan)per;}

1. 接口的多态

接口 接口引用=实现了接口的实例(接口引用可以任何一个实现了接口的类的实例对象)

例子：

接口Comparable

类Dog，类Cat都实现了Comparable接口

Comparable c；

C可以指向Dog对象，也可以指向Cat对象