## 重载、重写、多态

重写（override）：就是重写了一个方法，以实现不同的功能。一般是用于子类在继承父类时，重新实现父类中的方法。

规则：

1. 重写的参数列表必须与被重写的方法相同，否则是重载。
2. 重写方法的访问修饰符必须大于等于被重写的方法。若小于的话子类方法如果是private的话，父类的引用就不能指向子类对象的方法了。

3）重写方法的返回值必须和被重写的方法一致。

4）重写方法所抛出的异常必须小于等于被重写方法。

5）被重写的方法不能为private，否则在其子类中只是新定义了一个方法，并没有对其进行重写。

6）静态方法不能被重写（编译出错）。

7）私有方法和静态方法都是静态解析的。

重载（overload）：一般是用于在一个类内实现若干重载方法，这些方法的名称相同而参数形式不同。

规则：

1. 在使用重载时只能通过相同的方法名、不同的参数形式实现。可以是不同的参数类型、不同的参数个数、不同的参数顺序。
2. 不能通过访问权限、返回类型、抛出的异常进行重载。如果返回类型不同但是参数列表完全一样会报错，方法重复定义。行重xx.getXX();没有对返回值的限制
3. 方法的异常类型和数目不会对重载造成影响。

多态的概念比较复杂，有多种意义的多态，一个有趣但不严谨的说法是：继承是子类使用父类的方法，而多态则是父类使用子类的方法。父类的引用可以指向子类对象的方法。  
一般，我们使用多态是为了避免在父类里大量重载引起代码臃肿且难于维护。

如果使用多态的话，最终所选取的方法是按照以下顺序来的：

This.show(O)->super.show(o);->this.show(super(o));->super.show(super(o));

如果父类指向的子类对象，重写了父类的方法，则调用子类的方法。

重写和重载时JAVA多态的不同表现，重写是父类与子类之间多态性的表现，重载是一个类之间多态的表现。

Jvm会具体到哪个类的那个方法，带方法的参数和返回值。

但具体是那个对象，要在运行时确定。

Human man = new Man();

Human 是静态类型。

Man 是实际类型。

public class Shape

{

public static void main(String[] args){

Triangle tri = new Triangle();

System.out.println("Triangle is a type of shape? " + tri.isShape());// 继承

Shape shape = new Triangle();

System.out.println("My shape has " + shape.getSides() + " sides."); // 多态

Rectangle Rec = new Rectangle();

Shape shape2 = Rec;

System.out.println("My shape has " + shape2.getSides(Rec) + " sides."); //重载

}

public boolean isShape(){

return true;

}

public int getSides(){

return 0 ;

}

public int getSides(Triangle tri){ //重载

return 3 ;

}

public int getSides(Rectangle rec){ //重载

return 4 ;

}

}

class Triangle extends Shape

{

public int getSides() { //重写,实现多态

return 3;

}

}

class Rectangle extends Shape

{

public int getSides(int i) { //重载

return i;

}

}