重写是子类对父类的允许访问的方法的实现过程进行重新编写, **返回值**和**形参**都不能改变。**即外壳不变，核心重写！**

重写的好处在于子类可以根据需要，定义特定于自己的行为。 也就是说子类能够根据需要实现父类的方法。

重写方法不能抛出新的检查异常或者比被重写方法申明更加宽泛的异常。

**方法的重写规则**

* 参数列表与被重写方法的参数列表必须完全相同。
* 返回类型与被重写方法的返回类型可以不相同，但是必须是父类返回值的派生类
* 访问权限不能比父类中被重写的方法的访问权限更低。例如：如果父类的一个方法被声明为 public，那么在子类中重写该方法就不能声明为 protected。
* 父类的成员方法只能被它的子类重写。
* 声明为 final 的方法不能被重写。
* 声明为 static 的方法不能被重写，但是能够被再次声明。
* 子类和父类在同一个包中，那么子类可以重写父类所有方法，除了声明为 private 和 final 的方法。
* 子类和父类不在同一个包中，那么子类只能够重写父类的声明为 public 和 protected 的非 final 方法。
* 重写的方法能够抛出任何非强制异常，无论被重写的方法是否抛出异常。但是，重写的方法不能抛出新的强制性异常，或者比被重写方法声明的更广泛的强制性异常，反之则可以。
* 构造方法不能被重写。
* 如果不能继承一个方法，则不能重写这个方法。
* 当需要在子类中调用父类的被重写方法时，要使用 super 关键字。

方法重写是子类与父类的一种多态性表现。

重载(overloading) 是在一个类里面，方法名字相同，而参数不同。返回类型可以相同也可以不同。

每个重载的方法（或者构造函数）都必须有一个独一无二的参数类型列表。

最常用的地方就是构造器的重载。

**重载规则:**

* 被重载的方法必须**改变参数列表**(参数个数或类型不一样)；
* 被重载的方法可以改变返回类型；
* 被重载的方法可以改变访问修饰符；
* 被重载的方法可以声明新的或更广的检查异常；
* 方法能够在同一个类中或者在一个子类中被重载。
* 无法以返回值类型作为重载函数的区分标准。

方法重载是一个类的多态性表现

方法签名就是：方法名+参数