AOP设计思想:hook 是在runtime中动态创建子类的基础上实现的。所有的 swizzling 操作都发生在子类，这样做的好处是你不需要去更改对象本身的类，也就是，当你在 remove aspects 的时候，如果发现当前对象的 aspect 都被移除了，那么，你可以将 isa 指针重新指回对象本身的类，从而消除了该对象的 swizzling ,同时也不会影响到其他该类的不同对象)这样对原来替换的类或者对象没有任何影响而且可以在子类基础上新增或者删除aspect。

销毁Aspct需注意:

如果当前Student有2个实例，stu1和stu2，并且他们都同时hook了相同的方法study( )，stu2在执行完aspect\_remove，把stu2的study( )方法还原了。这里会把stu1的study( )方法也还原了。因为remove方法这个操作是对整个类的所有实例都生效的。

要想每个实例还原各自的方法，不影响其他实例，上述这段代码删除即可。因为在执行 remove 操作的时候，其实和这个对象相关的数据结构都已经被清除了，即使不去恢复 stu2 的study( ) 的执行，在进入 \_\_ASPECTS\_ARE\_BEING\_CALLED\_\_，由于这个没有响应的 aspects ，其实会直接跳到原来的处理逻辑，并不会有其他附加影响。