**SA思考题7**

**1、试述软件设计中的原则。**

**答：软件设计的七大原则：开闭原则、依赖倒置原则、单一职责原则、接口隔离原则、迪米特原则、里氏替换原则、合成（组合）/聚合复用原则。**

**（1）开闭原则**

**定义：一个软件实体如类、模块函数应该对扩展开放，对修改关闭。**强调的是**用抽象构建框架，用实现扩展细节。**以**提高软件系统的可复用性及可维护性**，帮助我们实现稳定灵活的系统架构。实现开闭原则的核心思想是面向抽象编程。

**代码场景：**web应用通常是有过滤器的，通常有很多个，每个都是一类功能的集合，一般我们只是通过继承 Filter 新增实现类的方式去添加新的过滤器来丰富我们的功能，而不是在原来的过滤器上做修改，将新的功能添加进去。这种就是对扩展开放，对修改关闭。

**（2）依赖倒置原则**

**定义：高层模块不应该依赖低层模块，二者都应该依赖其抽象。抽象不应该依赖细节；细节应该依赖抽象。针对接口编程，不要针对实现编程。**通过抽象（包括使用接口或者抽象类），可以使各个类或者模块的实现彼此独立，互不影响，从而实现模块间的耦合性。简单来说就是程序应该依赖于接口而不是实现类。

**优点：可以减少类间的耦合性、提高系统稳定性、提高代码可读性和维护性，可降低修改程序所造成的风险。**

**（3）单一职责原则**

**定义：不要存在多于一个导致类变更的原因。**例如某个类有了两个方法，每个方法负责一个职能，假如方法一的职责需要发生变化，那么就需要修改这个类文件，这种改变就可能导致原本运行正常的方法二发生故障。对于这种两个职责经常发生变化的方法，需要分别建立相应的 java 文件分别负责相应的职能，避免一方暂时改变另一方不变的情况下，相互影响。通常情况下我们对类单一职责要求较低，但是接口和方法尽量是保证单一职责。根据业务要求所写的类或者接口方法最好能满足需求即可，不需要对其进行可能会用到的扩展，如果真的有，那就再写一个方法、类、接口。

**在java代码中的体现：一个类/接口/方法只负责一项职责。**

**优点：降低类的复杂度、提高类的可读性、提高系统的可维护性、降低变更引起的风险。**

**（4）接口隔离原则**

**定义：用多个专门的接口，而不使用单一的总接口，客户端不应该依赖它不需要的接口。一个类对一个类的依赖应该建立在最小的接口上。建立单一的接口，不要建立庞大臃肿的接口。尽量细化接口，接口中的方法尽量少。注意适度原则，一定要适度。**

某个类可能只需要某个接口中的一个方法，但是该接口却有n多个他不需要的方法，如果实现这个接口那么就要实现该接口的所有定义的方法，这样是不对的，此时应该接口的方法独立为一个小的接口，或者新建一个包含该方法的接口。适度原则指的是，接口中的方法不能太少，如果太少的话那么就会产生多个接口，那么就会有很多的实现类导致整个系统类结构体系很庞大，系统可维护性变差。

**优点：符合高内聚低耦合的设计思想，从而使得类具有良好的可读性、可扩展性和可维护性。**

**（5）迪米特原则**

**定义：一个对象应该对其他对象保持最少的了解，又叫最少知道原则。尽量降低类与类之间的耦合。高内聚低耦合的体现。强调只和朋友交流，不和陌生人说话。保持神秘。**

**“朋友”：出现在成员变量、方法输入、输出参数中的类称为朋友关系类，而出现在方法体内部的类不属于朋友类**，所以这类实体类也是我们要避免的。

从类上面来说也需要进行自我的保护，也即类里面尽量减少对外公开的非 public 的方法、和非静态的 public 变量、成员变量 private 化，利用包权限和 protected 保护自己。

假如有这样一个场景，有一个方法放在A类也行，放在B类也行，那么如何去做？原则是：如果该方法放在本类中既不增加类间关系，也对本类不产生负面影响，那就可以放在本类中。对外部的类的引用越少越好。

**优点：降低类之间的耦合。**

**（6）里氏替换原则**

**定义：如果对每一个类型为 T1 的对象 o1，都有类型为 T2 的对象 o2，使得以 T1 定义的所有程序 P 在所有的对象 o1 都替换成 o2 时，程序 P 的行为没有发生变化，那么类型 T2 是类型 T1 的子类型。通过这个定义我们可以引申出来一个定义继承，T1 是 T2 的父类。**里式替换原则是继承复用的基石，当子类可以替换父类，并且软件功能不受影响时，父类才能真正的被复用，子类也可以增加自己新的功能，里式替换原则是对开闭原则的一个补充，它是对实现抽象化的具体步骤规范。里式替换原则所表达的含义就是反对子类重写父类方法的这一含义。

**定义扩展：一个软件实体如果适用一个父类的话，那么一定适用于其子类，所有引用父类的地方必须能透明的使用其子类的对象，子类对象能够替换父类对象，而程序逻辑不变。**

**引申意义：子类可以扩展父类的功能，但不能改变父类原有的功能。**

含义1：子类可以实现父类的抽象方法，但不能覆盖父类的非抽象方法。

含义2：子类可以增加自己特有的方法。

含义3：当子类的方法**重载**父类的方法时，方法的前置条件（即方法的输入/入参）要比父类方法的输入参数更宽松。

含义4：当子类的方法实现父类的方法时（重写/重载或实现抽象方法），方法的后置条件（即方法的输出/返回值）要比父类严格或相等。

**优点：**

**1.约束继承泛滥，开闭原则的一种体现。**

**2.加强程序的健壮性，同时变更时也可以做到非常好的兼容性，提高程序的可维护性、扩展性。降低需求变更时引入的风险。**

**（7）合成（组合）/聚合复用原则**

**定义：尽量使用对象组合/聚合，而不是继承关系达到软件复用的目的。**

聚合 has-A 和组合 contains-A

**优点：可以使系统更加灵活，降低类与类之间的耦合度，一个类的变化对其他类造成的影响相对较少。**

何时使用合成/聚合、继承

聚合 has-A 关系

组合contains-A 包含关系

继承 is-A 父子关系