\*\*面向对象编程的SOLID和STUPID原则：\*\*

\*\*SOLID\*\* 是五个面向对象设计原则的缩写，由迈克尔·费瑟斯提出：

1. \*\*单一职责原则（SRP）\*\* - 每个类应该只有一个改变的理由。

2. \*\*开闭原则（OCP）\*\* - 类应该对扩展开放，对修改关闭。

3. \*\*里氏替换原则（LSP）\*\* - 程序中的一个对象可以被它的子类型对象替换而不影响程序的正确性。

4. \*\*接口隔离原则（ISP）\*\* - 客户端不应该依赖于它们不使用的方法。

5. \*\*依赖倒置原则（DIP）\*\* - 高层模块不应该依赖于低层模块，两者都应该依赖于抽象。抽象不应该依赖于细节，细节应该依赖于抽象。

\*\*STUPID\*\* 是一个非官方的简化软件设计规则：

1. \*\*STUPID\*\* - 不要使事情比必要的更复杂。

\*\*Object类。默认方法的实现。\*\*

在Java中，`Object` 是所有类的根类。默认方法包括 `equals()`、`hashCode()`、`toString()`、`getClass()`、`wait()`、`notify()` 和 `notifyAll()`。这些方法通常在继承类中被重写，以提供符合类的特定行为。

\*\*简单和多重继承。Java中继承实现的特点。\*\*

Java支持简单继承，这意味着一个类只能直接继承一个类。然而，接口允许多重继承，因为一个类可以实现多个接口。Java在继承实现的特点包括私有成员的基类不被继承。

\*\*抽象类的概念。abstract修饰符。\*\*

抽象类是不能被直接实例化的类，主要用于定义必须在派生类中实现的通用接口或行为。`abstract` 修饰符用于声明抽象类或方法。

\*\*接口的概念。Java中接口的实现。接口与抽象类的区别。\*\*

接口在Java中是一种引用类型，可以包含只有public方法和变量成员，这些成员默认是 `public static final`。接口用于定义类必须实现的契约。与抽象类的区别在于接口不能包含方法实现（Java 8之前），并且类可以实现多个接口。

\*\*接口方法的default、static和private修饰符。\*\*

- \*\*default\*\*：带有default修饰符的方法可以有方法体，并且在实现接口的类中可以被实现。

- \*\*static\*\*：接口中的静态方法不能在实现接口的类中被重写，并且可以直接通过接口调用。

- \*\*private\*\*：接口中的私有方法仅供接口中的其他方法使用。

\*\*Java中的枚举类型（enum）。实现和使用的特点。\*\*

Java中的 `enum` 是一种特殊的类类型，允许创建有限的常量集合。特点包括字段初始化、方法、实现接口。枚举可以用于表示固定集合，如星期几或月份。

\*\*Java中的记录类型（record）。使用特点。\*\*

记录类型（record）是Java 14中引入的新类型，旨在表示数据。特点包括自动创建 `equals()`、`hashCode()` 和 `toString()` 方法，以及简化的构造函数和字段定义。

\*\*带有static和final修饰符的方法和字段。\*\*

- \*\*static\*\*：静态方法和字段属于类，而不是类的实例。它们可以在不创建类实例的情况下被调用。

- \*\*final\*\*：带有final修饰符的方法不能在派生类中被重写。被标记为final的字段必须在声明时或构造函数中初始化。

\*\*方法的重载和覆盖。\*\*

- \*\*重载（Overloading）\*\*：类中多个方法具有不同的签名（不同的参数数量和/或类型）。

- \*\*覆盖（Overriding）\*\*：派生类中的方法具有与基类中的方法相同的签名，并且可以改变或添加方法的行为。

\*\*异常处理，三种类型的异常。\*\*

Java中有三种类型的异常：

1. \*\*受检异常（Checked Exceptions）\*\*：必须由程序员处理或在方法签名中声明的异常。

2. \*\*非受检异常（Unchecked Exceptions）\*\*：不必处理的异常，例如 `NullPointerException` 或 `ArrayIndexOutOfBoundsException`。

3. \*\*错误（Errors）\*\*：与JVM相关的严重问题，如 `OutOfMemoryError`，通常不处理。

\*\*标准数组和动态数组（ArrayList）。主要区别。\*\*

- \*\*标准数组\*\*：具有固定长度，存储同一类型的元素。通过索引访问元素。

- \*\*ArrayList\*\*：动态数组，可以改变其长度。它实现了 `List` 接口，并提供添加、删除和搜索元素的方法。

\*\*内部类、局部类和匿名类。\*\*

- \*\*内部类\*\*：在其他类内部定义的类，但在方法外部。

- \*\*局部类\*\*：在方法内部定义的类。

- \*\*匿名类\*\*：没有名称的类，在创建实例时定义。它们经常用于实现接口或继承类以提供不同的行为。