1. 请简述子类不可以继承父类的哪些成员？使用继承有什么好处？多重继 承关系中属性的初始化顺序是怎样的？
2. 什么是方法重载？Override 和 Overload 的区别
3. 简述 this 关键字的作用、super 关键字的作用。
4. 什么情况下需要使用 static 关键字？
5. 请简述什么是多态？实现多态三要素？instanceof 关键词的作用？在 Java 中如何实现 多态？
6. 简述 final 关键字在类、方法和变量上的用途。
7. 如何实现对象的封装？什么是装箱拆箱？
8. 请简述线程的五个状态？线程对象调用 start()方法和调用 run()方法的区 别?
9. 什么是面向对象编程的主要特点？简述类和对象的关系和区别?
10. 集合类的理解 ArrayList 和 LinkedList 的联系和区别
11. 在 Java 中，构造方法的作用是什么？
12. 什么是继承？有什么好处？
13. 简述抽象类和接口的区别。
14. 面向对象设计中的“单一职责原则”是什么意思？
15. 请举例说明什么是异常，Java 中，如何进行异常处理？finally 关键字的作用？
16. 什么是对象的序列化？
17. 解释一下 Java 的垃圾回收机制。
18. 请简述怎么定义枚举类型？写出 java 中的常见包装类（用中文写不给 分）？
19. 请简述写出 I/O 流的四个基类，IO 流分成哪几个流，他们之间有什么区别？

20. 简述 java 的异常机制，什么是异常？ throw 和 throws 的区别

1. ****子类不可以继承父类的哪些成员****：
   1. 父类的私有成员：子类无法直接访问父类的私有成员，只能通过父类提供的公共方法来间接访问。
   2. 父类的构造方法：子类不能继承父类的构造方法，但可以通过super()关键字调用父类的构造方法来进行初始化。
   3. 父类的静态成员：子类可以继承父类的静态成员，但不能重写父类的静态成员。

****使用继承的好处****：

* 代码复用：子类可以继承父类的属性和方法，避免了代码的重复编写。
* 提高代码的可维护性：如果需要修改父类的属性或方法，只需要在父类中进行修改，子类会自动继承修改后的属性或方法。
* 提高代码的可读性：通过继承，可以将相关的类组织在一起，使代码更加清晰易读。

****多重继承关系中属性的初始化顺序****：

* 先初始化父类的属性，再初始化子类的属性。
* 如果有多个父类，按照继承的顺序依次初始化父类的属性。
* 如果父类中有静态属性，先初始化父类的静态属性，再初始化子类的静态属性。

1. ****方法重载****：方法重载是指在同一个类中，允许存在多个同名方法，但这些方法的参数列表不同（参数个数、参数类型或参数顺序不同）。方法重载可以提高代码的可读性和可维护性，使代码更加简洁明了。

****Override 和 Overload 的区别****：

* Override 是指子类重写父类的方法，要求方法名称、参数列表和返回类型（可以是子类型）与父类中被重写的方法完全相同，并且子类方法的访问权限不能低于父类方法的权限。
* Overload 是指在同一个类中，允许存在多个同名方法，但这些方法的参数列表不同（参数个数、参数类型或参数顺序不同）。

1. ****this 关键字的作用****：
   * 在方法内部，可以使用this关键字来引用当前对象。
   * 在构造方法内部，可以使用this关键字来调用其他构造方法。
   * 在成员方法内部，可以使用this关键字来访问当前对象的成员变量和成员方法。

****super 关键字的作用****：

* 在子类中，可以使用super关键字来调用父类的构造方法。
* 在子类中，可以使用super关键字来访问父类的成员变量和成员方法。

****什么情况下需要使用 static 关键字****：

* + 静态变量：如果一个变量需要被多个对象共享，并且在整个程序的生命周期内都保持不变，可以将其声明为静态变量。
  + 静态方法：如果一个方法不需要访问对象的成员变量和成员方法，可以将其声明为静态方法。
  + 静态代码块：如果需要在类加载时执行一些初始化操作，可以将其放在静态代码块中。

****多态****：多态是指同一个行为具有多种不同的表现形式或形态。多态允许不同类的对象对同一消息做出不同的响应。多态主要通过继承和方法重写来实现，使得可以用父类的引用指向子类的对象，在运行时根据实际对象的类型来确定具体执行的方法。

****实现多态三要素****：

* 继承：子类继承父类，并重写父类的方法。
* 方法重写：子类重写父类的方法，实现自己特定的逻辑。
* 父类引用指向子类对象：通过父类的引用指向子类的对象，在运行时根据实际对象的类型来确定具体执行的方法。

****instanceof 关键词的作用****：instanceof关键词用于判断一个对象是否是某个类的实例。如果是，则返回true，否则返回false。

****在 Java 中如何实现多态****：

* 定义父类和子类，子类继承父类，并重写父类的方法。
* 创建父类的引用，指向子类的对象。
* 通过父类的引用调用子类重写的方法，实现多态。

****final 关键字在类、方法和变量上的用途****：

* + 类：使用final关键字修饰的类不能被继承，即该类是最终的类。
  + 方法：使用final关键字修饰的方法不能被重写，即该方法是最终的方法。
  + 变量：使用final关键字修饰的变量不能被修改，即该变量是常量。

****对象的封装****：对象的封装是指将对象的属性和方法封装在一起，对外提供公共的访问接口，隐藏内部的实现细节。通过封装，可以提高代码的安全性和可维护性，使代码更加简洁明了。

****装箱拆箱****：装箱是将基本数据类型转换为对应的包装类对象，拆箱是将包装类对象转换为对应的基本数据类型。在 Java 中，自动装箱和自动拆箱是由编译器自动完成的，不需要手动进行转换。

1. ****线程的五个状态****：
   * 新建（New）：线程对象被创建，但还未启动。
   * 就绪（Runnable）：线程准备好运行，等待获取 CPU 资源。
   * 运行（Running）：线程正在执行。
   * 阻塞（Blocked）：线程因等待资源等原因而暂停执行。
   * 死亡（Dead）：线程执行完毕或因异常而终止。

****线程对象调用 start()方法和调用 run()方法的区别****：

* start()方法：启动线程，使线程进入就绪状态，等待获取 CPU 资源。
* run()方法：线程的执行体，当线程获取到 CPU 资源后，会自动执行run()方法中的代码。

1. ****面向对象编程的主要特点****：
   * 封装性：将对象的属性和方法封装在一起，对外提供公共的访问接口，隐藏内部的实现细节。
   * 继承性：子类可以继承父类的属性和方法，避免了代码的重复编写。
   * 多态性：同一个行为具有多种不同的表现形式或形态，允许不同类的对象对同一消息做出不同的响应。
   * 抽象性：将对象的共同特征提取出来，形成抽象类或接口，提高代码的可读性和可维护性。

****类和对象的关系和区别****：

* 关系：类是对象的模板，对象是类的实例。
* 区别：类是抽象的，不占用内存空间，而对象是具体的，占用内存空间。

****集合类的理解 ArrayList 和 LinkedList 的联系和区别****：

* + 联系：ArrayList和LinkedList都是List接口的实现类，都可以存储有序的元素。
  + 区别：
    - ArrayList是基于数组实现的，查询速度快，但插入和删除速度慢。
    - LinkedList是基于链表实现的，查询速度慢，但插入和删除速度快。

****在 Java 中，构造方法的作用是什么****：

* + 构造方法用于创建对象，并对对象进行初始化。
  + 构造方法的名称与类名相同，没有返回值类型。
  + 构造方法可以有参数，也可以没有参数。

****继承****：继承是指子类继承父类的属性和方法，从而实现代码的复用和扩展。

****继承的好处****：

* 代码复用：子类可以继承父类的属性和方法，避免了代码的重复编写。
* 提高代码的可维护性：如果需要修改父类的属性或方法，只需要在父类中进行修改，子类会自动继承修改后的属性或方法。
* 提高代码的可读性：通过继承，可以将相关的类组织在一起，使代码更加清晰易读。

****抽象类和接口的区别****：

* + 抽象类可以包含非抽象方法和属性，而接口中只能包含抽象方法（Java 8 之后可以包含默认方法和静态方法）。
  + 一个类可以实现多个接口，但只能继承一个抽象类（单继承）。
  + 抽象类中的方法可以有不同的访问权限，而接口中的方法默认都是 public。
  + 抽象类可以有构造方法，接口不能有构造方法。

****面向对象设计中的“单一职责原则”是什么意思****：

* + “单一职责原则”是指一个类应该只负责一项职责，而不应该负责多项职责。
  + 如果一个类负责多项职责，当其中一项职责发生变化时，可能会影响到其他职责的实现，从而导致代码的维护性和可读性降低。

****异常****：异常是指在程序运行过程中发生的错误或异常情况。

****Java 中进行异常处理的方式****：

* 使用try-catch语句捕获并处理异常。
* 使用throws关键字声明可能抛出的异常。
* 使用finally