1. 面向对象的特征有哪些？

抽象、封装、继承、多态

* **抽象**是指从特定的角度出发，从已经存在的一些事物中抽取我们所关注的特性、行为，从而形成一个新的事物的思维过程，是一种从复杂到简洁的思维方式
* **封装是指将对象的实现细节隐藏起来，然后通过公共的方法来向外暴露出该对象的功能**。但封装不仅仅是**private+getter/setter**，使用封装可以对setter进行更深层次的定制，例如你可以对执行方法的对象做规定，也可以对数据走一定的要求，还可以做类型转换等等。使用封装不仅仅安全，更可以简化操作。
* **继承是面向对象实现软件复用的重要手段，当子类继承父类后，子类是一种特殊的父类，能直接或者间接得到父类里的成员。**继承的缺点：①继承是一种强耦合关系，父类变化子类也必须边；②继承破坏了封装，对于父类而言，它的实现细节对子类来说都是透明的。继承链中对象方法的调用的优先级：this.show(O), super.show(O), this.show((super)O), super.show((super)O)。

<https://www.jianshu.com/p/5771df145452>

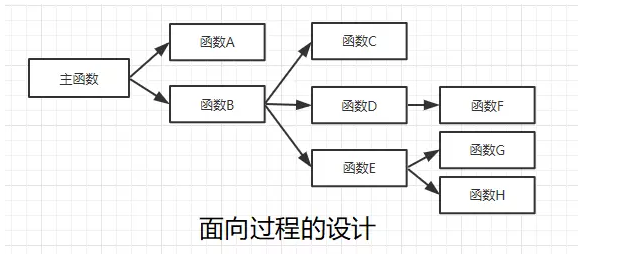
**向上转型：当父类对象引用变量引用子类对象时，被引用对象的类型决定了调用谁的成员方法，引用变量类型决定可调用的方法。如果子类没有覆盖该方法，那么会去父类中寻找。**

* **多态简而言之就是同一个行为具有多个不同表现形式或者形态的能力。**比如：有一杯水，我不知道它是温的、冰的、烫的，但是我一摸就知道了，我摸水杯的这个动作，对于不同温度的睡，就会得到不同的结果，这就是多态。**多态的条件**：①继承②重写③向上转型。多态的好处：当把不同的子类对象都当作父类类型来看，**可以屏蔽不同子类对象之间的实现差异，**从而适应需求的不断变化。

1. 面向对象和面向过程的区别

答：面向过程是一种站在过程的角度思考问题的思想，强调的是功能行为，**功能的执行过程**，即先干啥，后干啥。

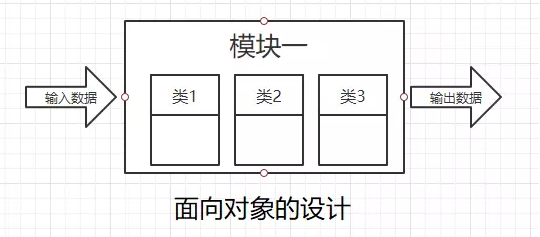
**面向过程的设计**：最小的程序单元是函数，每个函数负责完成某一个功能，用以接受输入数据，函数对输入数据进行处理，然后输出结果数据。整个软件系统由一个个的函数组成，其中作为程序入口的函数被称之为主函数，主函数一次调用其它函数，普通函数之间可以相互调用，进而实现整个系统功能。



面向过程的缺陷：面向过程的设计，采用的是自顶而下的设计方式，在设计阶段就需要考虑每一个模块应该分解成那些子模块。然后将子模块细化为一个个函数。

问题：设计不够直观，与人类的习惯思维不一致；系统软件适应性差，可扩展性差，维护性低。面向过程最大的问题在于随着系统的膨胀，面向过程将无法应付，最终导致系统的崩溃。

**面向对象：**面向对象是一种基于面向过程的新的编程思想，是一种站在对象的角度思考问题的思想，我们把多个功能合理的放到不同的对象中，**强调的是具备某些功能的对象。**



面向对象更加符合我们常规的思维方式，稳定性好，可重用性强，易于开发大型软件产品，由良好的可维护性。

**注意：并不是说面向对象一定优于面向过程的设计。**

1. **JDK和JRE的区别是什么**？

答：JRE---Java运行环境（Java Runtime Environment），它包括Java虚拟机，Java核心库类和支持文件，但并不包含开发工具（JDK）---编译器，调试器和其它工具。

JDK---Java开发工具（Java Development Kit）是完整的Java软件开发包，包含JRE，编译器和其它工具（比如JavaDoc,Java调试器），可以让开发者开发、编译、执行Java应用程序。

1. **Java中覆盖和重载是什么意思？**

答：**覆盖（Override）是指子类对父类方法的一种重写**，只能比父类抛出更少的异常，访问权限不能比父类的小，被覆盖的方法不能是private的，否则只是在子类中重新定义一个新方法。

**重载（Overload）表示同一个类中可以有多个名称相同的方法，但这些方法的参数列表各不相同。**

面试官：构成重载的条件有哪些？

答：参数类型不同，参数个数不同，参数顺序不同

面试官：函数的返回值不同可以构成重载吗？为什么？

答：不可以，因为Java中调用函数并不需要强制赋值。

1. Java和C++的区别？

答：①都是面向对象的语言，都支持封装、继承、多态；

②指针：Java不提供指针来直接访问内存，程序更加安全；

③继承：Java的类是单继承的，C++支持多重继承；Java通过一个类实现多个接口来实现C++中的多继承；**Java中类不可以多继承，但是，接口可以多继承**；

④内存：Java有自动内存管理机制，不需要程序员手动释放内存。

1. 抽象类和接口的区别？
   1. 抽象类可以没有抽象方法；**接口中的方法必须是抽象方法**；
   2. 抽象类中可以没有普通的成员变量；接口中的变量必须是**static final类型**的，**必须被初始化，接口中只有常量，没有变量**；
   3. 抽象类只能单继承，接口可以继承多个父接口；
   4. Java 8中接口会有default方法，即方法可以被实现。



**面试官：抽象类和接口如何选择**

答：① 如果要创建不带任何方法定义和成原变量的基类，那么就应该选择接口而不是抽象类；

② 如果知道某个类应该是基类，那么第一个选择的应该是让它成为一个接口，只有在必须要有方法定义和成员变量的时候，才应该选择抽象类。因为抽象类中允许存在一个或多个被具体实现的方法，只要方法没有被全部实现该类仍是抽象类。

1. Static关键字是什么意思？

答：static关键字表明一个成员变量或者成员方法可**以在没有所属的类的实例变量的情况下被访问**。

静态变量（static变量）被所有的对象共享，在内存中只有一个副本，它当且仅当在类初次加载时会被初始化。而非静态变量时对象所拥有的，在创建对象的时候被初始化，存在多个副本，各个对象拥有的副本互不影响。

面试官：Java中是否可以覆盖（override）一个private或者static的方法？

答：Java中static方法不能被覆盖，因为方法覆盖是基于运行时动态绑定的，而**static方法是编译时静态绑定的**。Static方法跟类的任何实例都不相关，所以概念上不适用。

Java中也不可以覆盖private的方法，因为**private修饰的变量和方法只能在当前类中使用**，如果是其它的类继承当前类时不能访问到private变量和方法的，当然也不能覆盖。

1. **JDK中常用的包有哪些？**

答：java.lang java.util java.io java.net java.sql

1. **JDK JRE和JVM的联系和区别？**

答：JDK是Java开发工具包，是Java开发环境的核心组件，并提供编译、调试和运行一个Java程序所需要的所有工具，可执行文件和二进制文件，是一个平台特定的软件。

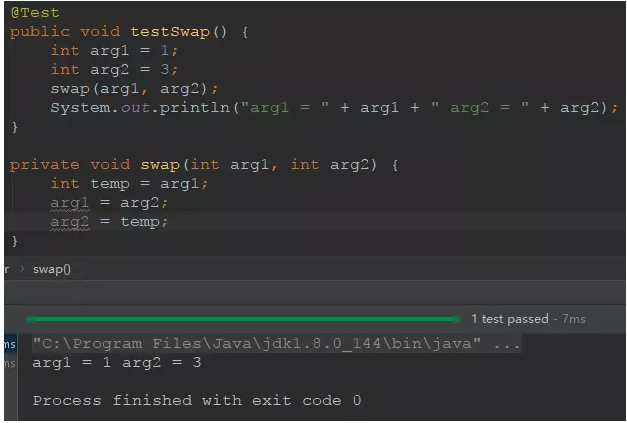
JRE是Java运行时环境，是JVM的实施实现，提供了运行Java程序的平台。JRE包含了JVM，但不包含Java编译器/调试器之类的开发工具；

JVM是Java虚拟机，当我们运行一个程序时，JVM负责将字节码转换为特定机器代码，JVM提供了内存管理/垃圾回收和安全机制等。

区别：JDK用于开发，JRE用于运行Java程序；JDK和JRE中都包含了JVM；JVM是Java编程语言的核心并且具有平台独立性。

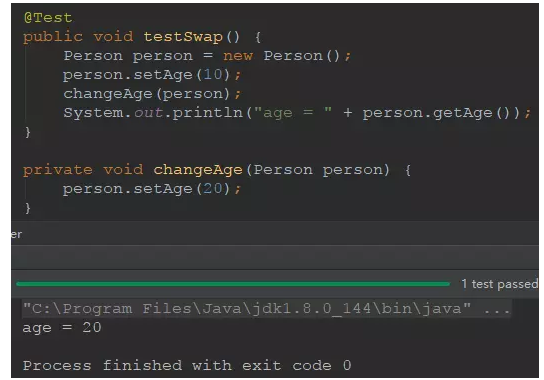
1. **Java是值传递还是引用传递？**

答：值传递是对基本型变量而言的，传递的是该变量的一个副本，改变副本不影响原变量；引用传递一般是对于对象型变量而言的，传递的是该对象地址的一个副本，并不是原对象本身。一般认为，**Java内的传递都是值传递，Java中实例对象的传递是引用传递**。



我们希望在swap（）方法之后交换arg1和arg2的值，但事实上并没有，为什么？

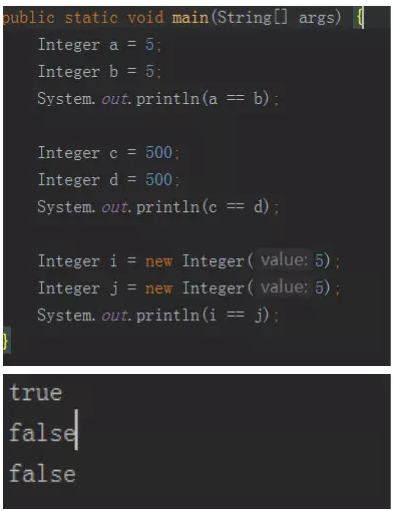
答：Java是值传递的，也就是说，我们在调用一个需要传递参数的函数时，**传递给函数的参数并不是我们传递进去的参数本身，而是它的一个副本**。我们改变的数据其实只是改变了副本的数据而已，并不会对原来的参数有任何的改变。



我们自己定义了一个内部类Person，该类只有一个int类型的age属性，然后又getter/setter，我们希望通过changeAge（）函数来改变Person对象的age属性，为什么这次成功？

答：主函数将person赋值一份到changeAge函数中去，最终还是只改变了changeAge中赋值的那一份参数的值，而原本的参数并没有改变，**但changeAge中的那一份和原本的参数指向了同一个内存区域**。

1. **Integer的缓存机制**



第一个返回true很好理解，a和b指向相同的地址；

第二个返回false是因为**Integer有缓存机制，在JVM启动初期就缓存了-128到127这个区间内的所有数字**；

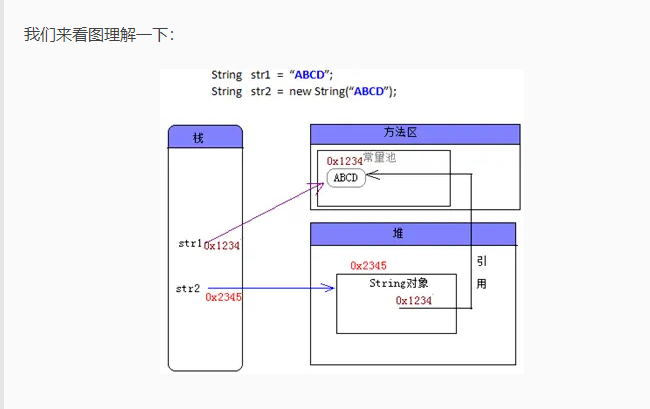
第三个返回false是因为用了new关键字来开辟新的空间，i和j两个对象分别指向**堆区**中的两块内存空间。

1. **下述两个方法分别创建了几个String对象？**



答：String str1 =”ABCD”**最多创建一个String对象，最少不创建String对象**。如果常量池中，存在”ABCD”，那么str1直接引用，此时不创建String对象。否则，先在常量池创建“ABCD”内存空间，再引用。

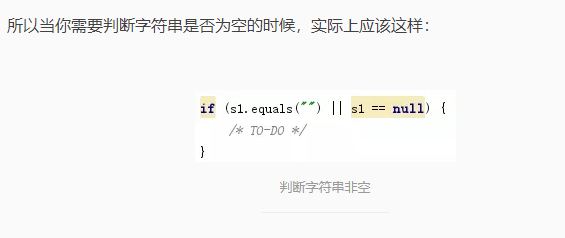
String str2 = new String（”ABCD”）**最多创建两个String对象，至少创建一个String对象**。New关键字绝对会在**堆空间**中创建一块新的内存区域，所以至少创建一个String对象。



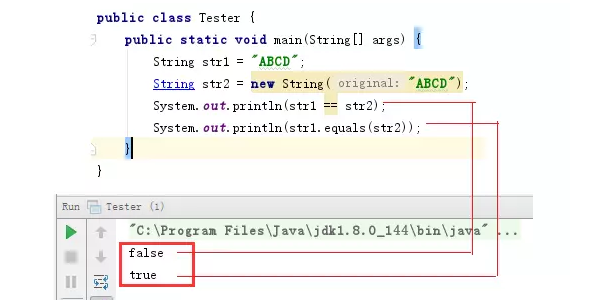
1. 当执行第一句的时候，会在常量池中添加一个新的ABCD字符，str1指向常量池的ABCD
2. 当执行第二句的时候，因为有new操作符，所以会在**堆空间**开辟一块空间用来存储新的String对象，因为此时常量池中已经有ABCD字符，所以堆中的String对象指向常量池中的ABCD，而str2则指向堆空间的String对象。

**String类详解：String 、 StringBuffer 、 StringBuilder**

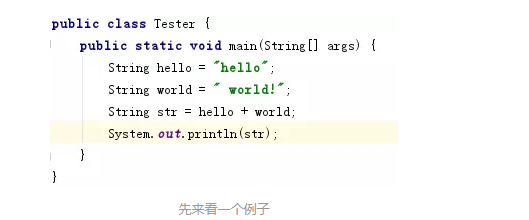
1. **String对象不可变**，即当对象创建完毕后，该对象的内容（字符序列）是不允许改变的，如果内容改变则会创建一个新的String对象，返回到原地址中。String类中每一个看起来会修改String值得方法，实际上都是创建了一个新的String对象，以包含修改后得字符串对象，而最初得String对象则丝毫未动。
2. **String对象的空值**：String str1 = null //没有初始化，没有分配内存空间；String str2= “ “// 分配有内存空间，有内容



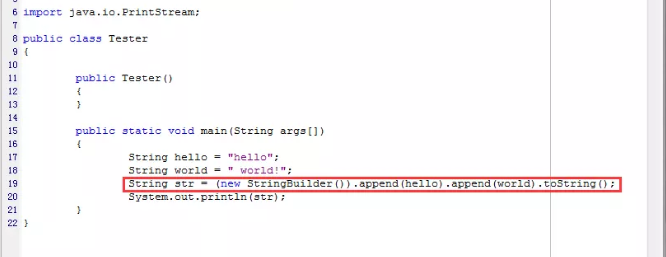
1. **自负串的比较：**使用“==“，只能比较**引用的内存地址**是否相同，使用”equals“方法，则比较的是**字符串的内容**。



1. **“+“号是怎么来连接字符串的：**



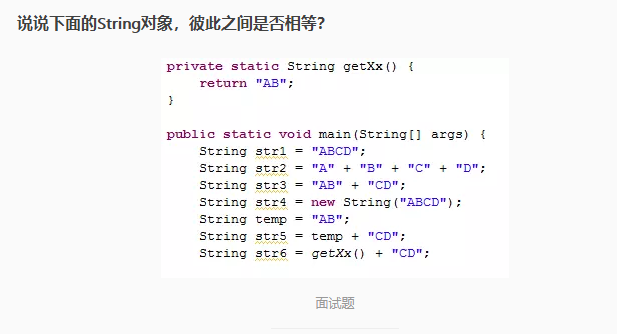
Java反编译程序（Xjad）--将生成的class文件反编译为java：



编译器自动引入了一个**java.lang.StringBuilder**类。在这个例子中，编译器首先是创建了一个**StringBuilder对象**，用以构造最终的String，并为每个字符串**调用了一次StringBuilder的append（）方法**，总计两次。最后调用**toString()**生成结果。这是编译器自动优化的结果。

1. **可变的字符串**：StringBuilder/StringBuffer：当对象创建完毕之后，该对象的内容可以发生改变，当内容发生改变的时候，对象保持不变。 如：append()方法。**如果字符串操作比较简单，那就可以依赖编译器，也就是String类型，它会为你合理的构造最终的字符串结果；但是如果你还使用循环，多次改变字符串的内容，那就更加适合StringBuilder对象。**
2. **StringBuilder/StringBuffer:** StringBuffer线程安全（该对象方法中所有的方法都是用了**synchronized**修饰符），因此开销很大。有没有synchronized修饰符，是两者唯一的区别。

**面试题：**



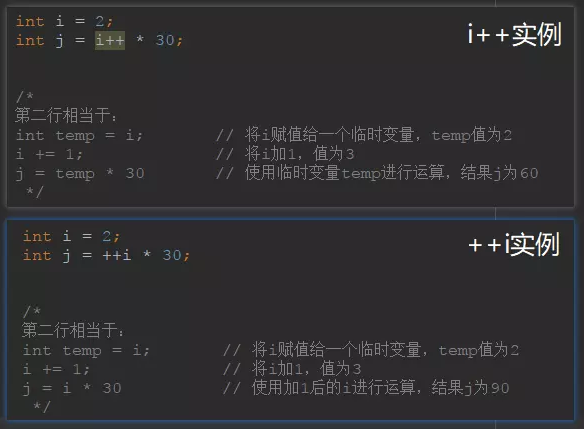
**str1和str2/str3相等，和另外几个都不想等**

**知识点：**

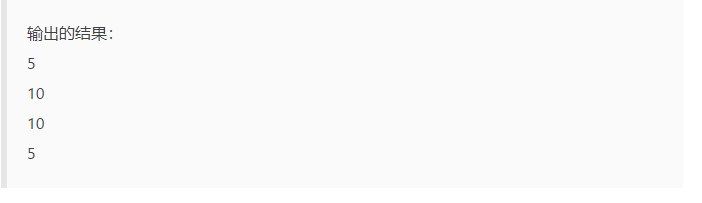
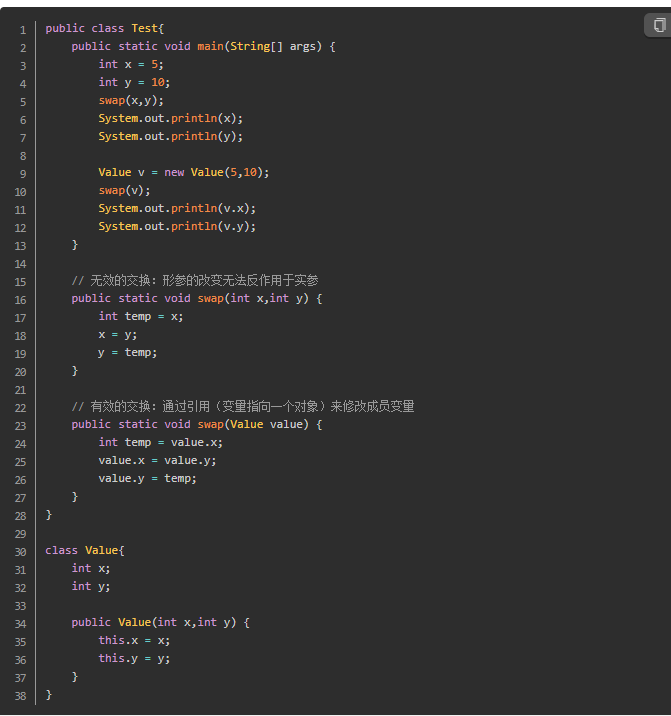
* 单独使用“”引号创建的字符串都是直接量，编译期就已经确定存储到常量池中；
* 使用只包含直接量的字符串连接符如“aa”+“bb“创建的也是直接量编译期就能确定，已经确定存储到常量池中（str2和str3）
* 使用new String（“”）创建的对象会存储到**堆内存**中，是运行期才运行的；
* 使用包含String直接量（无final修饰符）的字符串表达式（如“aa”+str1）创建的对象是运行期才创建的，存储在堆内存中；
* 通过变量/调用方法取连接字符串，都只能是在运行时期才能确定变量的值和方法的返回值，不存在编译优化操作。

1. **i++和++i到底有什么不同？**

解析：不管是i++还是++i，都是先将变量加1，然后才继续计算的。二者之间真正的区别是：**++i是将变量的值加1后，使用增值后的变量进行运算的；i++是首先将变量赋值给一个临时变量，接下来对变量加1，然后使用那个临时变量进行运算。**

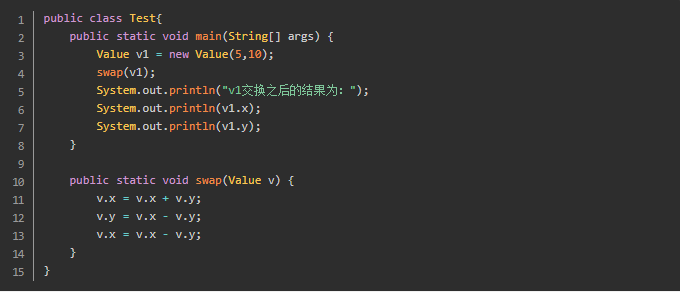


1. **交换变量的三种方式**
   * 1. 通过三个变量



②通过相加的方式

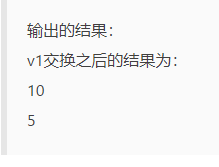
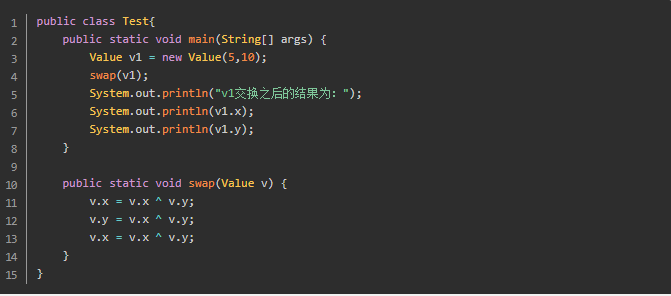
注：**不推荐** 因为当数值很大的时候，16进制的求和运算可能造成数据的溢出。



③通过异或的方式

注：**异或的方法更可取，异或不存在数据溢出。**

位异或运算符（^）有这样的一个性质：两个整型的数据x和y，有**（x^y^y）==x** 这说明如果一个变量x异或两外一个变量y两次，结果为x。



**15.Java对象的初始化顺序？**

答：不考虑静态成员的初始化，调用一个对象的构造函数时，程序**首先调用父类的构造函数**（可以通过super关键字指定父类的构造函数，否则默认调用无参的构造函数，并且需要在子类的构造函数的第一行调用），**之后静态成员变量的初始化函数和静态初始化块则按照在代码当中的顺序执行**，成员变量如果没有指定值的话则赋予默认值，即基本数据类型0或false等，对象则为null；**最后调用自身构造函数。**



**16.true、false与null是关键字吗？**

答：不是。true、false是布尔型的字面常量，null是引用类型的字面常量

面试官：那goto与const呢？

答：是。goto与const（用来声明一个变量的值是无法改变的）均是Java语言保留的关键字，即没有任何语法应用。

**17.exception和error有什么区别？**

答：exception和error都是Throwable的子类。**exception用于用户程序可以捕获的异常情况；error定义了不期望被用户程序捕获的异常**。exception表示一种设计的问题，也就是说只要程序正常运行，从不会发生的情况；error表示回复不是不可能但是很困难的情况下的一种严重问题，比如内存溢出，不可能指望程序处理这样的情况。

**18.throw和throws有什么区别？**

答：:throw关键字用来在程序中明确的抛出异常，相反，throws语句用来表明方法不能处理的异常。每一个方法都必须要指定哪些异常不能处理，所以方法的调用者才能够确保处理可能发生的异常，多个异常要用逗号隔开。