黑马程序员入学面试题

第一部分　知识点

1. 面试大概流程

(1)　自我介绍　　(2)　几道编程题　　(3)　Java 基础知识提问

2. 面向对象的理解

面向对象是基于面向过程的。在开发过程中，要完成特定的功能就去找相应的对象，如果找不到就创建相应的对象，使用对象，维护完善对象。

3. 面向对象和面向过程的区别

面向过程强调的是功能行为，实现过程中的每一步的具体实现。

面向对象将功能封装成对象，强调的是具备功能的对象，让对象去调用方法。

4. abstract 与哪些关键字不能共存为什么

private：私有的方法是不可见的，无法被重写

final：被 final 修饰的方法是最终方法，无法被重写

static：被 static 修饰的方法，要随类加载到方法区，由于抽象方法没有方法体所以不能加载

5. static 的特点

随着类的加载而加载；优先于对象存在；被所有对象所共享；可以直接被类名所调用。

静态方法只能访问静态成员，非静态方法既可以访问静态也可访问非静态

静态方法中不可以定义 this、super 关键字，因为静态优先于对象存在，所以静态方法中不可以出现 this；

6. 类与类、类与接口、接口与接口的关系

继承，继承或实现，继承

7. final、finally、finalize 的区别

final：可以修饰类、方法和变量，被 final 修饰的类无法被继承，方法无法被重写，变量为常量只能赋值一次

finally：异常处理 try、catch 时使用，可以添加也可以不添加，用于执行一些必须执行的代码，如关闭资源等

finalize：Object 类中的方法，其中定义了对象要被垃圾收集器回收之前要做的相关的清理工作

8. 什么是多态，多态的好处和弊端

多态可以理解为事物存在的多种体现形态。父类的引用指向了自己的子类对象；父类的引用也可以接收自己子类的对象。

好处：提高了代码的扩展性

弊端：父类的引用只能访问父类中有的成员(父类引用无法调用子类中特有的方法)

9. 多态实现的机制

靠的是父类或接口定义的引用变量可以指向子类或具体实现类的实例对象，而程序调用的方法在运行期才动态绑定，就是引用变量所指向的具体实例对象的方法，也就是内存里正在运行的那个对象的方法，而不是引用变量的类型中定义的方法。重写、重载是多态性的不同表现。

10. wait()和 sleep()的区别

(1)所在的类不同：wait 是 Object 类中的方法，sleep 是 Thread 类中的方法

(2)sleep 方法没有释放锁，而 wait 方法释放了锁，使得其他线程可以使用同步控制块或者方法。

(3)Sleep()必须指定时间，wait()可以指定也可以不指定时间。

(4)wait，notify 和 notifyAll 只能在同步方法或者同步代码块里面使用，而 sleep 可以在任何地方使用

(5)sleep 必须捕获异常，而 wait，notify 和 notifyAll 不需要捕获异常

11. 网络编程的三要素

(1)IP：网络中设备的唯一标识

(2)端口 :用于标识设备中接收数据的网络应用程序具体是哪一个,有效端口 0~65535，0~1024 为系统端口及保留端口

(3)传输协议：UDP 和 TCP 协议，信息通讯的规则

12. UDP 和 TCP 的区别

UDP：不需要建立连接，是不可靠协议，将数据及源和目的封装到数据报中，每个数据报限制在 64k 以内，发送数据较少，速度快

TCP：通过三次握手建立连接通路，是可靠协议，可进行大量的数据传输，效率较低

13. 什么是暴力反射

我们知道 java 的特性之一就是封装，将对象的属性和具体实现隐藏起来，只对外提供公共方法访问，private 修饰的内部属性和方法对我们是不可见的。

我们通过正常的方法是无法获取以及修改的 ,可是通过反射却可以强制获取并做一些修改，这就破坏了封装性，这就是所谓的暴力反射

14. 反射获取字节码对象方式、创建对象的两种方式

获取字节码方式三种：

(1)类名.class，例如：System.class

(2)对象.getClass()，例如：new Date().getClass();

(3)Class.forName("类名")，例如：Class.forName("java.util.Date");

创建对象的两种方式：

(1)直接用字节码创建对象，只能调用默认的构造方法：字节码.newInstance();

(2)获取构造方法 Constructor，然后调用构造方法创建，可以通过参数不同调用不同的构造方式

15. 怎么理解反射，反射的应用

反射就是把 Java 类中的各种成分映射成相应的 Java 类。

一般情况下我们要解决某个问题，先找到相关的类，创建该类的对象，然后通过该对象调用对应的方法来解决问题。

反射是一个正好相反的过程，开始可能并没有类可以解决这个问题，而我们却先用一个当时可能并不存在的方法解决了这个问题，后来才有的这个类。

这其实就是框架的原理，现有的框架再有的解决问题的类。框架描述了整体，制订了功

能方法的规范，具体的实现之后按照这个规范编写。这些都需要靠反射来完成。

使用框架有良好的扩展性，某部分功能的优化不需要涉及程序整体，只需要修改特定的部分就好了，然后通过配置文件，获取对应的类名，就可以了。

16. 对匿名内部类的理解

匿名内部类其实是内部类的简写形式。

内部类是定义在类中的类，就好比我们人类，我们人类都有心脏，而心脏又有自己独特组成，可以把心脏也抽象成一个类。

这个心脏类就是人类的内部类。如果要研究某一种心脏疾病，需要一个实例时，我们不需要知道这个患病的心脏到底是谁的，那找到的这个就是匿名的。

匿名内部类必须要继承一个类或实现一个接口，在使用时直接用父类或接口的名称创建了一个子类对象并实现其中的方法，匿名内部类其实是一个匿名的子类对象。

传递参数的时候，需要的是抽象类或者接口的时候，其实需要的是子类或实现类对象，这时可以通过匿名内部类创建。

17. IO 体系

字节流 InputStream/OutputStream

|--FileInputStream/FileOutputStream：文件字节流，用于文件的读写操作

|--BufferedInputStream/BufferedOutputStream：加缓冲区的字节流，用于提高效率

字符流 Reader/Writer

|--FileReader/FileWriter：文件字符流，用于文本文件的读写操作

|--BufferedReader/BufferedWrite：加缓冲区的字符流，用于提高效率

转换流 InputStreamReader/OutputStreamWriter：是字节流和字符流之间的桥梁

配置文件 Properties

18. 集合体系

|--Collection

|--List：元素是有序的，元素允许重复，因为该集合体系都具有索引

|--ArrayList：底层数据结构是数组，查询速度快，增删操作较慢，线程不同步

|--LinkedList：底层数据结构是链表，查询效率较低，增删操作快，线程不同步

|--Vector：功能同 ArrayList 类似，底层也是数组，不同是线程是同步的，效率较低

|--Set：元素是无序的，元素不允许重复，底层用到了 Map

|--HashSet：底层 hash 表，存储的对象最好复写 hashCode 和 equals 方法，保证元素不会重复

|--TreeSet：底层二叉树，存储的对象具备比较性，有两种方法

|--Map：数据是以键值对的形式存储的，有的元素存在映射关系就可以使用该集合，元素不允许重复

|--HashMap：允许有 null 键或值，线程不同步

|--TreeMap

|--HashTable：类似 HashMap，不允许有 null 键或值，线程同步

19. 线程的创建方式，进程线程的理解

两种：继承 Thread 或实现 Runnable 接口

20. 进程与线程

进程是一个正在执行中的程序，每一个程序都至少有一个执行顺序，该顺序是一个路径，或者叫一个控制单元。

线程是进程中的一个独立的控制单元，线程在控制着进程的执行，是进程的执行路径。

21. 局部变量和成员变量区别

成员变量：作用于整个类中，随对象存储在堆内存中，生命周期跟对象一样

局部变量：作用于方法或语句中，方法或语句结束则生命周期结束，存放在栈内存中。

22. 同步函数与同步代码块的区别

它们的作用都是封装多条操作共享数据的语句，只能让一个线程都执行完，在执行过程中，其他线程不可参与进来。

同步代码块：位置比较灵活，封装了操作共享数据的语句，多个线程中只有持有锁的才可以操作共享数据，需要指定一个对象作为锁，锁可以是任意对象，但必须是同一对象。

同步方法：声明方法时加 synchronized 关键字修饰，同步函数使用的锁是 this，持有锁的线程调用这个方法时其他线程无法调用。静态方法的锁是类的字节码文件。

23. 数组和集合的区别

A：长度区别

数组的长度固定

集合长度可变

B：内容不同

数组存储的是同一种类型的元素

而集合可以存储不同类型的元素

C：元素的数据类型问题

数组可以存储基本数据类型，也可以存储引用数据类型

集合只能存储引用类型

24. StringBuffer 和 StringBuilder 的区别

StringBuffer 是线程安全的，StringBuilder 是线程不安全的，所以效率比起来 StringBuilder 要比 StringBuffer 快。

一般单线程的程序使用 StringBuilder 比较好，多线程的情况下可以自己加锁，也可以直接使用 StringBuffer

25. String 和 StringBuffer 区别

String 对象一经创建就不会发生变化了 即便是赋新值也不是在原对象的基础上改变，而是创建一个新的字符串对象，将引用指向这个新的对象，会造成空间的浪费，效率较低

StringBuffer 只创建一个对象，是在同一个实例的基础之上追加，效率更高，当需要用

到 String 的时候 toString 就可以了

第二部分 知识点

26. 说明 Java 中 String str=null 与 String str=""的区别？

String str = null 表示声明了一个 String 对象的引用 str ,但是没有为其分配内存空间。

String str = "" 表示创建了一个长度等于 0 的空字符串，并在内存中为其分配了内存空间。

String str = new String("tw"); str 在内存中有两个对象，1.堆内存有一个 new Sting 2.常量池中有一个字符串。

27. 使用运算符"= ="和方法 equals() 进行比较对象的区别？

使用"= ="算符比较基本数据类型时，比较的是值相等；使用"=="算符比较对象时，比较的是内存地址和内容。

使用 equals()方法 比较对象时，比较的是对象的内容，与地址无关，如果没有重写 equals()方法就直接调用的是 Object 的 equals()方法。

"=="比较的是外在和内涵；equals()比较的是外在。用博大精深的中文讲就是：一个相同，一个相似。

28. 接口和抽象类有何区别？

（1）成员特点：

构造方法：接口没有构造方法；抽象类有构造方法。

成员变量：接口中只有常量；抽象类中可以是常量，也可以是变量。

成员方法：接口中只有抽象方法；抽象类中既有抽象方法，也有非抽象方法。

（2）关系特点：

类与类：类与类之间只有单继承，可以有多层继承。

类与接口：类与接口之间是实现关系，可以单实现，也可以多实现。

接口与接口：接口与接口之间是继承关系，可以单继承，也可以多继承。

（3）设计理念：

接口对应的设计模式是简单工厂设计模式，它被实现，这个接口中定义的是整个体现的扩展内容，体现：like a。

抽象类对象的设计模式是模板设计模式，它被继承，这个类中定义的是整体体现的共性内容，显现 is a。

（4）抽象关键字 abstract 和哪些不可以共存？final， private , static

（5）必须实现抽象类或接口的所有抽象方法才可以实例化。

（6）继承 extends，实现 implements

抽象类与接口的异同？		
	抽象类 abstract class，不能实例化	接口 interface，当抽象类方法都是抽象时 ,可以表示为接口 ,接口是公开的。
成员变量	变量/常量	常量：public static final
成员方法	抽象/非抽象 ,没有方法体 ,可以包含静态方法，public/protected 修饰	抽象，没有方法体，没有静态方法，public abstract
构造方法	有	没有
关系区别	类与类：继承，单继承 一般的应用里，最顶级的是接口，然后是抽象类实现接口，最后才到具体类实现。	类与接口：实现，单/多实现 接口与接口：继承，单/多继承 实现接口需要实现接口定义的所有方法
设计理念	被继承体现的是 is a 的关系， 抽象类中定义的是继承体系的共性功能。	被实现体现的是 like a 的关系 ,接口中定义的是该继承体系的扩展功能。

29. 简述方法的重写与重载

重写是指重新实现基类中的方法。在运行过程中 ,如果将子类创建的对象赋值给子类的引用或父类的引用 ,则无论通过哪个类型的引用 ,真正调用的方法都将是在子类中重写的方法。 如果希望调用父类中的方法，则需要通过父类创建类的实例，然后通过该实例才能访问父类定义的方法。重写方法时实现多态的一个重要表现。

在 Java 中 ,一个类的内部可以定义多个同名的方法 ,但是它们的方法参数要有所不同。重写的是与父类同名的方法 ,与返回值有关。重载是同名方法 ,与返回值无关 ,与参数有关，如：个数、类型、排列。

重写注意事项：

（1）父类中的私有方法不可以被重写

（2）子类方法访问权限一定要大于父类的访问权限

（3）静态的方法只能被静态的方法重写，这个其实不能算对象的关系。

30. final、finally、finallize 有何区别？

final 表示一个修饰符，如果用它来修饰一个类，则该类是不能继承的；如果用它来修饰一个变量 ,则该变量一旦赋值之后就不能再修改；如果用它来修饰一个方法 ,则该方法不能够被重写。

finally 用于异常处理，它用来修饰一个代码块，即使前面的代码处理异常，该代码块中的代码也会执行。它通常用于释放资源。finally 代码块之前有调用 System.exit（ ）方法的话，finally 里面的代码就不会执行。

finallize 表示 Object 类中定义的一个方法，他可以重写，用于回收资源。

31. 编程计算 3 乘 8 等于几，什么方法效率更高？

位运算的方法效率更高。编程运算其实质是二进制的运算，运用位运算的方法移动后补位即可得结果，比一般的 3*8 更高效。

32. 不借用第三方变量，互换两个变量。

n=11，m=3

（2）n=n^m;

m=n^m;//(n^m)^m 一个数异或另一个数两次，还是得原来那个数

n=n^m//n^(n^m)

小编强力推荐第二种方法，这是提升逼格的强劲手段。当面试官问你这个问题，你跟他拓展到位移运算，你的背后就会出现万丈光芒，那时你就会神一样存在在面试官的心里。

33. 传统 for 和高级 for 有什么区别呢？

对集合进行遍历。

只能获取集合元素，但是不能对集合进行操作。

迭代器除了遍历，还可以进行 remove 集合中元素的动作。

如果是用 ListIterator，还可以在遍历过程中进行增删改查的动作。

高级 for 有一个局限性，必须要有被遍历的目标。

建议在遍历数组的时候，还是希望是用传统 for。因为传统 for 可以定义角标。

1.增强 for 循环和 iterator 遍历的效果是一样的，也就说增强 for 循环的内部也就是调用 iteratoer 实现的(可以查看编译后的文件)，但是增强 for 循环 有些缺点，例如不能在增强循环里动态的删除集合内容。不能获取下标等。

2.ArrayList 由于使用数组实现，因此下标明确，最好使用普通循环。

3.而对于 LinkedList 由于获取一个元素，要从头开始向后找，因此建议使用 增强 for 循环，也就是 iterator。

34. Collection 和 Colections 的区别是什么？

Collection 是一个单列集合的顶层接口，它是 List、Set、Queue 等接口的父接口。在这个接口中，定义了集合类中通用的方法。

Collections 是针对集合操作的工具类，有对集合进行排序和二分查找的静态的方法。

35. 面向对象和面向过程的理解

面向对象时相对面向过程而言的。比如说：人坐地铁去上班。人面向地铁这个对象，地铁拥有运输的功能，人通过地铁调用这个功能，只重结果，不重于过程。面向过程就如我们开门，锁孔对齐，门轴转动，门才开了，相对于结果，侧重于过程。

曾经跟朋友聊对面向对象，我说就像搭积木，找到需要的，一个一个往上累；她说这就是 JAVA 方便之处。

（1）面向过程是以函数为基础，完成各种操作，强调的是过程，例如：C 语言；

（2）面向对象以对象为基础，完成各种操作，强调的是对象和结果。

（3）面向对象是基于面向过程的。

面向对象的特点是：

（1）是一种个符合人们思考习惯的思想
（2）把复杂的事情简单化
（3）把我们从执行者变成了指挥者
举例：老师您口渴想喝茶，如果你自己去拿杯子，放茶叶，
接水这些是面向过程的；但这时您恰好知道我是个打水小能手，
又正好我很尊敬您，一见如故、一见钟情啥的，很愿意帮您去泡茶。
您调用我帮您泡茶，使用我的拿杯子、放茶叶、接水等方法。
您由执行者上升到了指挥者，让泡茶这件事更简单化。

36. 线程之间通信的理解

其实，Java 提供了 3 个非常重要的方法来巧妙地解决线程间的通信问题。
这 3 个方法分别是：wait()、notify()和 notifyAll()。
它们都是 Object 类的最终方法，因此每一个类都默认拥有它们。
虽然所有的类都默认拥有这 3 个方法，但是只有在 synchronized 关键字作用的范围内，并且是同一个同步问题中搭配使用这 3 个方法时才有实际的意义。
这些方法在 Object 类中声明的语法格式如下所示：

```
final void wait() hrows InterruptedException
final void notify()
final void notifyAll()
```

调用 wait()方法可以使调用该方法的线程释放共享资源的锁，然后从运行态退出，进入等待队列，直到被再次唤醒。
调用 notify()方法可以唤醒等待队列中第一个等待同一共享资源的线程，并使该线程退出等待队列，进入可运行态。
调用 notifyAll()方法可以使所有正在等待队列中等待同一共享资源的线程从等待状态退出，进入可运行状态，此时，优先级最高的那个线程最先执行。
显然，利用这些方法就不必再循环检测共享资源的状态，而是在需要的时候直接唤醒等待队列中的线程就可以了。这样不但节省了宝贵的 CPU 资源，也提高了程序的效率。
由于 wait()方法在声明的时候被声明为抛出 InterruptedException 异常，因此，在调用 wait()方法时，需要将它放入 try…catch 代码块中。此外，使用该方法时还需要把它放到一个同步代码段中，否则会出现如下异常：
"java.lang.IllegalMonitorStateException: current thread not owner"

37. 线程的状态

线程的五种状态：
1）新建(new) 用 new 语句创建的线程对处于新建状态，此时它和其它 Java 对象一样，仅仅在 Heap 中被分配了内存。当一个线程处于新建状态时，它仅仅是一个空的线程对象，系统不为它分配资源。 Thread t = new Thread(new Runner());

2）就绪(Runnable) 程序通过线程对象调用启动方法 start()后，系统会为这个线程分配它运行时所需的除处理器之外的所有系统资源。这时，它处在随时可以运行的状态，在随后的任意时刻，只要它获得处理器即会进入运行状态。 t.start()

3）运行(Running) 处于这个状态的线程占用 CPU，执行程序代码。在并发环境中，如果计算机只有一个 CPU,那么任何时刻只会有一个线程处于这个状态。如果计算机中有多个 CPU，那么同一时刻可以让几个线程占用不同的 CPU,使它们都处于运行状态，只有处于就绪状态的线程才有机会转到运行状态。

4）阻塞(Blocked) 阻塞状态是指线程因为某些原因放弃 CPU，暂时停止运行。当线程处于阻塞状态时，Java 虚拟机不会给线程分配 CPU，直到线程重新进入就绪状态，它才有机会转到运行状态。

阻塞状态可分为以下 3 种：

位于对象等待池中的阻塞状态(Blocked in object's wait pool):当线程处于运行状态时，如果执行了某个对象的 wait()方法，Java 虚拟机就会把线程放到这个对象的等待池中。

位于对象锁池中的阻塞状态(Blocked in object's lock pool):当线程处于运行状态，试图获得某个对象的同步锁时，如果该对象的同步锁已经被其他线程占用，Java 虚拟机就会把这个线程放到这个对象的锁池中。

其他阻塞状态(Otherwise Blocked)：当前线程执行了 sleep()方法，或者调用了其他线程的 join()方法，或者发出了 I/O 请求时，就会进入这个状态。当一个线程执行 System.out.println()或者 System.in.read()方法时是，就会发出一个 I/O 请求，该线程放弃 CPU，进入阻塞状态，直到 I/O 处理完毕，该线程才会恢复执行。

5）死亡(Dead) 当线程退出 run()方法时，就进入死亡状态，该线程结束生命周期。线程有可能是正常执行完 run()方法而退出，也有可能是遇到异常而退出。不管线程是正常结束还是异常结束，都不会对其他线程造成影响。

38. 数组和集合

1）数组是静态的，一个数组实例具有固定的大小，一旦创建了就无法改变容量了。而集合是可以动态扩展容量，可以根据需要动态改变大小，集合提供更多的成员方法，能满足更多的需求。

2）数组要声明元素的类型，集合类的元素类型却是 object。

数组转成集合注意事项：

1.不能使用集合的增删方法。因为数组长度不能改变，其它方法可以使用；

2.如果数组中的元素是引用数据类型，转成集合时，数组元素直接作为集合的元素；如果数组中的元素是基本数据类型，转成集合时，会将数组作为集合中的元素。

39. String、StringBuffer 和 StringBuilder

1）在执行速度方面的比较：StringBuilder > StringBuffer

2）StringBuffer 与 StringBuilder,每当我们用它们对字符串做操作时，实际上是在一个对象上操作的，不像 String 一样创建一些对象进行操作，所以速度就快了。

3）StringBuilder：线程非安全的，是线程不同步的（JDK1.5 后出现）

StringBuffer：线程安全的，是线程同步的

4.什么时候使用字符串缓冲区：数据多，个数无所谓确定，类型无所谓确定，只要最后都转变成字符串，就用字符串缓冲区。

对于三者使用的总结：

1）如果要操作少量的数据用 String

2）单线程操作字符串缓冲区 下操作大量数据 StringBuilder

3）多线程操作字符串缓冲区 下操作大量数据 StringBuffer（多线程中通常不建议使用 StringBuffer，最好使用 StringBuilder 然后自己加同步）

40. 集合

集合名称阅读技巧：

JDK1.2 出现的集合框架中常用的子类对象

前缀名是结构名，后缀名是体系名。

ArrayList：数组结构，看到 Array 就知道查询速度快，看到 List 就知道可以又重复元素，可以增删改查

LinkedList：链表结构，增删快

HashSet：哈希结构，查询速度更快，不保证有序，不可以重复，必须覆盖 hashCode 和 equals 方法

LinkedHashSet：链表加哈希结构，可以实现有序

TreSet :二叉树结构 ,可以排序。有两种方法 :1.自然排序 compable ,2.比较器 comparator

Java 提供了只包含一个 compareTo()方法的 Comparable 接口。这个方法可以个给两个对象排序。具体来说，它返回负数，0，正数来表明输入对象小于，等于，大于已经存在的对象。

Java 提供了包含 compare()和 equals()两个方法的 Comparator 接口。compare()方法用来给两个输入参数排序，返回负数，0，正数表明第一个参数是小于，等于，大于第二个参数。equals()方法需要一个对象作为参数，它用来决定输入参数是否和 comparator 相等。只有当输入参数也是一个 comparator 并且和当前 comparator 的排序结果是相同的时候，这个方法才返回 true.

ArrayList 和 Vector 都是使用数组方式存储数据，此数组元素数大于实际存储的数据以便增加和插入元素，允许直接按序号索引元素. 但是插入元素要涉及数组元素移动等内存操作，所以索引数据快而插入数据慢，

Vector 由于使用了 synchronized 方法（线程安全），通常性能上较 ArrayList 差，Vector 属于遗留容器，现在已经不推荐使用，

LinkedList 使用双向链表实现存储（将内存中零散的内存单元通过附加的引用关联起来，形成一个可以按序号索引的线性结构，这种链式存储方式与数组的连续存储方式相比，其实对内存的利用率更高），按序号索引数据需要进行前向或后向遍历，但是插入数据时只需要记录本项的前后项即可，所以插入速度较快。

由于 ArrayList 和 LinkedListed 都是非线程安全的，如果需要多个线程操作同一个容器，可以通过工具类 Collections 中的 synchronizedList 方法将其转换成线程安全的容器后再使用（这其实是装潢模式最好的例子，将已有对象传入另一个类的构造器中创建新的对象来增加新功能）。

Map 集

Map（k，v） k:此映射所维护的键的类型 v：映射值的类型

Map 集合：双例集合，一次存一对，称为键值对。要保证键的唯一性。

Map 集合没有迭代器，必须先将 map 集合转成 set 集合，在使用迭代器，就可以取出结合中的元素；或者使用 map 集合特有的 entrySet 方法，也可以取出 map 集合中虽有元素。

Map 集合常见子类：

1.HashTable：哈希表结构，同步的，不允许 null 键，null 值；

2.HashMap：哈希表结构，不同步，允许 null 键，null 值；

3.TreeMap：二叉树结构，不同步，可以对 map 集合众多键进行排序。

41. 为什么需要配置 path，什么时候需要 classpath?

答：path路径：是操作系统的环境变量，目的是希望在任意目录下都能执行该命令操作起来更加方便快捷；

classpath 是给类提供路径：想在其他目录下执行 java 程序时需要配置 classpath 路径。

42. IO 流

输入流：InputStream　输出流：OutputStream

字节输入流：FileInputStream　字节输出流：FileOutoutStream

字节输入流缓冲区：BufferedInputStream　字节输出流缓冲区：BufferedOutputStream

字符输入流：InputStreamReader　字符输出流：OutputStreamWriter

转换流：FileReader FileWriter

字符输入流缓冲区：BufferedReader　字符输出流缓冲区：BufferedWriter

43. OSI 参考模型

一共 7 层：1.物理层；2.数据化链路层；3.网络层 ip；4.传输层 tcp\udp；5.会话层；6.表示层；7.应用层 http

物理层：定义物理设备标准，作用：传输比特流。这一层的数据叫做比特；

数据化链路层：接受物理层的数据对 MAC 地址（网卡地址）进行封装与解封装，工作设备：交换机，这一层的数据叫做帧；

网络层：对下面接受的数据进行 IP 地址的封装与解封装，工作设备：路由器，这一层的数据叫做数据包；

传输层：定义一些数据协议和端口号，对下面接收的数据进行分段和传输，到达目的后进行重组，这一层的数据叫做段；

会话层：通过传输层建立数据传输的通路；

表示层：对接收的解释进行加密和解密，封装和解封装；

应用层：终端应用，FTP（各种文件下载）。

TCP(传输控制协议)：传输效率低，可靠性高，用于操作可靠性要求高，数据量小的数据；

UDP（用户数据协议）：传输效率高，可靠性低，用于操作可靠性要求低，数据量大的数据。

44. abstract 与哪些关键字不能共存为什么

final：1.被 final 修饰的类不可以有子类；2.被 final 修饰的方法不可以被覆盖；

private：1.成员方法私有化，子类不能访问；2.private 私有化的，不可以被覆盖

static：被 static 修饰的方法，可以直接通过类名方式调用，但调用抽象方法没有意义

45. static 的特点

1.被静态修饰的成员，可以直接通过类名方式调用；2.静态成员优先与对象存在；3.静态成员随着累的加载而出现

注意事项：

1.静态只能访问静态；2.静态方法中不能出现 this 和 super 关键字；3.主函数是静态函数

46. wait()和 sleep()的区别

1. wait()是 object 类中的方法，可以指定时间，也可以不指定时间；sleep()是 Thread 类中的方法，必须指定时间；

2. wait()释放资源，释放锁；sleep()释放资源，不释放锁。

3、sleep 必须捕获异常，wait 不需要捕获异常

47. 网络编程的三要素

(1)IP：网络中设备的标识

(2)端口：用于标识设备中接收数据的网络应用程序具体是哪一个，有效端口 0~65535，0~1024 为系统端口及保留端口

(3)传输协议：UDP 和 TCP 协议，信息通讯的规则

23.什么是多态，多态的好处和弊端

多态可以理解为事物存在的多种体现形态。父类的引用指向了自己的子类对象；父类的引用也可以接收自己子类的对象。

好处：提高了代码的扩展性

弊端：父类的引用只能访问父类中有的成员(父类引用无法调用子类中特有的方法)

24.什么是暴力反射

我们知道 java 的特性之一就是封装，将对象的属性和具体实现隐藏起来，只对外提供公共方法访问，private 修饰的内部属性和方法对我们是不可见的。

我们通过正常的方法是无法获取以及修改的 ,可是通过反射却可以强制获取并做一些修改，这就破坏了封装性，这就是所谓的暴力反射

25.反射获取字节码对象方式、创建对象的两种方式

获取字节码方式三种：

(1)类名.class，例如：System.class

(2)对象.getClass()，例如：new Date().getClass();

(3)Class.forName("类名")，例如：Class.forName("java.util.Date");

创建对象的两种方式：

(1)直接用字节码创建对象，只能调用默认的构造方法：字节码.newInstance();

(2)获取构造方法 Constructor，然后调用构造方法创建，可以通过参数不同调用不同的构造方式

26.怎么理解反射，反射的应用

反射就是把 Java 类中的各种成分映射成相应的 Java 类。

一般情况下我们要解决某个问题，先找到相关的类，创建该类的对象，然后通过该对象调用对应的方法来解决问题。

反射是一个正好相反的过程，开始可能并没有类可以解决这个问题，却先用一个当时可能并不存在的方法解决了这个问题，后来才有的这个类。

这其实就是框架的原理，先有的框架再有解决问题的类。框架描述了整体，制订了功能方法的规范。这些都需要靠反射来完成。

使用框架有良好的扩展性，某部分功能的优化不需要涉及程序整体，只需要修改特定的部分就好了，然后通过配置文件，获取对应的类名，就可以了。

27.同步函数与同步代码块

它们的作用都是封装多条操作共享数据的语句，只能让一个线程都执行完，在执行过程中，其他线程不可参与进来。

同步代码块：位置比较灵活，多个线程中只有持有锁的才可以操作共享数据，需要指定一个对象作为锁

同步方法：声明方法时加 synchronized 关键字修饰，同步函数使用的锁是 this，持有锁的线程调用这个方法时其他线程无法调用。

第三部分 面试题

48. collection 和 collections 的区别。

collections 是个 java.util 下的类，是操作集合的工具类，可以对集合进行查找、排序、安全化，它包含有各种有关集合操作的静态方法。

collection 是个 java.util 下的接口，它是各种集合结构的父接口。

49. list, set, map 是否继承自 collection 接口? list，set 是 map 不是。

50. arraylist 和 vector 的区别。

一.同步性:vector 是线程安全的，也就是说是同步的，而 arraylist 是线程序不安全的，

不是同步的

二.数据增长:当需要增长时,vector 默认增长为原来一培，而 arraylist 却是原来的一半

51. 4. hashmap 和 hashtable 的区别

一.历史原因:hashtable 是基于陈旧的 dictionary 类的，hashmap 是 java 1.2 引进的 map 接口的一个实现

二.同步性:hashtable 是线程安全的 ,也就是说是同步的 ,而 hashmap 是线程序不安全的，不是同步的

三.值：只有 hashmap 可以让你将空值作为一个表的条目的 key 或 value

第四部分 面试题

52. 1．写一个卖票的程序。

1) 写一个类，该类实现了 Runnable 接口。有一个私有类型的 int 型的参数：tickets。票的总数，为 100，完成 run 方法，输出结果的格式如下：
2) 当前窗口为:窗口 a ，剩余的票数为:19，其中窗口 a 为线程的名称。
3) 开启四个卖票窗口(起四个线程)，同时执行卖票的程序。

```
public class SellTicketDemo {

    public static void main(String[] args) {
        SellTicket st = new SellTicket(new Ticket());
        new Thread(st,"窗口1").start();
        new Thread(st,"窗口2").start();
        new Thread(st,"窗口3").start();
        new Thread(st,"窗口4").start();
    }

}

class SellTicket implements Runnable {
    private Ticket t;

    public SellTicket(Ticket t) {
        this.t = t;
    }

    @Override
    public void run() {
        while (t.getTickets() > 0) {
            t.sellTicket(new Random().nextInt(5) + 1);
        }
    }
```

```
}
//车票类
class Ticket {
    private int tickets = 100;//车票的整数是100张

    public int getTickets() {//获取车票的余数
        return tickets;
    }

    public synchronized void sellTicket(int num) {//售票
        if (tickets >= num) {
            try {
                Thread.sleep(100);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
            tickets -= num;
            System.out.println(Thread.currentThread().getName() + "卖出" + num
                    + "张票,余票为：" + tickets);
        } else {
            try {
                Thread.sleep(100);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
            System.out.println("余票不足");
        }
    }
}
```

53. 用代码实现 List 和 map 存储、取出数据 使用泛型

54. 将用户输入在控制台上的指定路径下所有的.txt 文件复制到 c 盘下随意目录（你可以自己指定路径）并在复制后将.txt 后缀名更改成.java 后缀名

```
public class Test1 {
    public static void main(String[] args) throws IOException {
        // 封装目录
        File srcfile = new File( “d:\\test” );
        File desfile = new File("d:\\demo");
        if (!desfile.exists()) {
            desfile.mkdir();
```

```
        }
        File[] fileArray = srcfile.listFiles(new FilenameFilter() {

            @Override
            public boolean accept(File file, String name) {
                return new File(file, name).isFile() && name.endsWith(".txt");
            }

        });
        for (File file : fileArray) {
            // d:\\demo\\a.mp3
            File newfile = new File(desfile, file.getName());
            copyFile(file, newfile);
        }
        // 在目的地下改名
        File[] files = desfile.listFiles();
        for (File file : files) {
            File newfile = new File(desfile, file.getName().replace(".txt",".java"));
            file.renameTo(newfile);
        }

    }
    //复制文件
    private static void copyFile(File file, File newfile) throws IOException {
        //封装数据源，高效流
        BufferedInputStream bis = new BufferedInputStream(new FileInputStream(
                file));
        //封装目的地，高效流
        BufferedOutputStream bos = new BufferedOutputStream(
                new FileOutputStream(newfile));
        //一次复制一个字节数组
        byte[] bys = new byte[1024];
        int len = 0;
        while ((len = bis.read(bys)) != -1) {
            bos.write(bys, 0, len);
            bos.flush();
        }
        //释放资源
        bos.close();
        bis.close();
```

}
}

55. throw 和 throws 关键字的区别：

(1) throw 用于抛出异常对象，后面跟的是异常对象；throw 用在方法体内。

(2) throws 用于抛出异常类，后面跟的异常类名，可以跟多个，用逗号隔开。throws 用在方法上。

(3) 异常处理方式：抛出、捕捉 try-catch-finally

56. 编译时期异常和运行时异常的区别

1. 编译被检查的异常在函数内被抛出，函数必须要声明，否编译失败。
   声明的原因：是需要调用者对该异常进行处理。
2. 运行时异常如果在函数内被抛出，在函数上不需要声明。

不声明的原因：不需要调用者处理，运行时异常发生，已经无法再让程序继续运行，所以，不让调用处理的，直接让程序停止，由调用者对代码进行修正。

57. 定义异常处理时，什么时候定义 try，什么时候定义 throws 呢？

功能内部如果出现异常，如果内部可以处理，就用 try；
如果功能内部处理不了，就必须声明出来，让调用者处理。

58. 常见的包、类、接口

(1) 常用的类：BufferedReader BufferedWriter FileReader FileWirter String Integer java.util.Date，System，Class，List，HashMap

(2) 常用的包：java.lang、java.io、java.util、java.sql 、javax.servlet、org.apache.strtuts.action、org.hibernate

(3) 常用的接口：List、Map、HttpSession、Servlet、HttpServletRequest、Comparable

59. 常见异常

常见异常	
IndexOutOfBoundsException	角标越界异常
NullPointerException	空指针异常
ConcurrentModificationException	并发修改异常
ClassCastException	类型转换异常
UnsupportedOperationException	不支持操作异常
NullPointerException	没有元素异常
IllegalArgumentException	非法参数异常
IllegalAccessException	非法访问异常

二.简答题(每题 5 分)

1. 简述 JRE 与 JDK 的区别?

答案:

JRE: Java Runtime Environment; Java 运行环境，面向用户

JDK: Java Development Kits; Java 开发工具包，面向开发者，JDK 中包含有 JRE.

2. &和&&有什么区别?

答案:
&: 与， 不管左右两边结果如何，都会运行；
&&：短路与，左边为 False，则右边不会运行。
3. 什么是方法的重载?
答案:
在同一个类中，允许一个以上的同名函数，只要他们的参数类型或者参数列表不同即可。
4.说说 continue 和 break 的区别?
答案:
continue: 只用于 loop 语句，跳出本次循环继续下次循环。
break: 用于 switch 语句和 loop 语句，结束循环语句。
三.编程题 2 题(每题 10 分)
1. 设计一个方法,取名叫 getCount 用来计算出 1-100 之间有多少能被 3 整除,要求有返回值,并把结果打印在控制台上.

```
class ThreeDemo
{
	public	static void main(String[] args)
	{
		int	c = getCount();
		System.out.println("个数是："+c);
	}
	public	static int getCount()
	{
		int	x = 0;
		for(int	i = 1;i<=100;i++)
		{
			int	r = i % 3;
			if(r==0)
			{
				x++;
			}
		}
		return	x;
	}
}
```

2. 设计一个方法,用来打印出下面的图形,取名为 printJuXing.把图形打印在控制台上,不要返回值.

```
*****
*****
```

```
*****
*****
class printJuXing
{
	public	static void main(String[] args)
	{
		printRec(4,5);
	}
	public	static void printRec(int x,int y)
	{
		for(int	i=1;i<=x;i++)
		{
			for(int	j=1;j<=y;j++)
			{
				System.out.print("*");
			}
			System.out.println();
		}
	}
}
```

1、说明 Java 中 String　str=null 与 String　str=""的区别？

String str = null 表示声明了一个 String 对象的引用 str，但是没有为其分配内存空间。

String str = "" 表示创建了一个长度等于 0 的空字符串 ,并在内存中为其分配了内存空间。

2、使用运算符"=="和方法 equals() 进行比较对象的区别？

使用"=="算符比较基本数据类型时，比较的是值相等；使用"=="算符比较对象时，比较的是内存地址和内容。

使用 equals()方法 比较对象时 ,比较的是对象的内容 ,与地址无关 ,如果没有重写 equals()方法就直接调用的是 Object 的 equals()方法。

小编的理解："=="比较的是外在和内涵；equals()比较的是外在。用博大精深的中文讲就是：一个相同，一个相似。

3、接口和抽象类有何区别？

（1）成员特点：

构造方法：接口没有构造方法；抽象类有构造方法。

成员变量：接口中只有常量；抽象类中可以是常量，也可以是变量。

成员方法：接口中只有抽象方法；抽象类中既有抽象方法，也有非抽象方法。

（2）关系特点：
类与类：类与类之间只有单继承，可以有多层继承。
类与接口：类与接口之间是实现关系，可以单实现，也可以多实现。
接口与接口：接口与接口之间是继承关系，可以单继承，也可以多继承。
（3）设计理念：
接口对应的设计模式是简单工厂设计模式，它被实现，这个接口中定义的是整个体现的扩展内容，体现：like a。
抽象类对象的设计模式是模板设计模式，它被继承，这个类中定义的是整体体现的共性内容，显现 is a。

4、简述方法的重写与重载

重写是指重新实现基类中的方法。在运行过程中，如果将子类创建的对象赋值给子类的引用或父类的引用，则无论通过哪个类型的引用，
真正调用的方法都将是在子类中重写的方法。 如果希望调用父类中的方法，则需要通过父类创建类的实例，然后通过该实例才能访问父类定义的方法。重写方法时实现多态的一个重要表现。

在 Java 中，一个类的内部可以定义多个同名的方法，但是它们的方法参数要有所不同。重写的是与父类同名的方法，与返回值有关。重载是同名方法，与返回值无关，与参数有关，如：个数、类型、排列。

重写注意事项：
（1） 父类中的私有方法不可以被重写
（2） 子类方法访问权限一定要大于父类的访问权限
（3） 静态的方法只能被静态的方法重写，这个其实不能算对象的关系。

5、final、finally、finallize 有何区别？

final 表示一个修饰符，如果用它来修饰一个类，则该类是不能继承的；如果用它来修饰一个变量，则该变量一旦赋值之后就不能再修改；如果用它来修饰一个方法，则该方法不能够被重写。

finally 用于异常处理，它用来修饰一个代码块，即使前面的代码处理异常，该代码块中的代码也会执行。它通常用于释放资源。

finally 代码块之前有调用 System.exit（）方法的话，finally 里面的方块就不会执行。

finallize 表示 Object 类中定义的一个方法，他可以重写，用于回收资源。

6、编程计算 3 乘 8 等于几，什么方法效率更高？
位运算的方法效率更高。编程运算其实质是二进制的运算，运用位运算的方法移动后补位即可得结果，比一般的 3*8 更高效。

7、不借用第三方变量，互换两个变量。

n=11，m=3

（1）n=n+m//如果 n 和 m 的值非常大，容易超出 int 范围

m=n-m

n=n-m

（2）n=n^m;

m=n^m;//(n^m)^m 一个数异或另一个数两次，还是得原来那个数

n=n^m//n^(n^m)

小编强力推荐第二种方法，这是提升逼格的强劲手段。当面试官问你这个问题，你跟他拓展到位移运算，你的背后就会出现万丈光芒，那时你就会神一样存在在面试官的心里。

8、传统 for 和高级 for 有什么区别呢？

对集合进行遍历。

只能获取集合元素，但是不能对集合进行操作。

迭代器除了遍历，还可以进行 remove 集合中元素的动作。

如果是用 ListIterator，还可以在遍历过程中进行增删改查的动作。

高级 for 有一个局限性，必须要有被遍历的目标。

建议在遍历数组的时候，还是希望是用传统 for。因为传统 for 可以定义角标。

1.增强 for 循环和 iterator 遍历的效果是一样的，也就说增强 for 循环的内部也就是调用 iteratoer 实现的(可以查看编译后的文件)，但是增强 for 循环 有些缺点，例如不能在增强循环里动态的删除集合内容。不能获取下标等。

2.ArrayList 由于使用数组实现，因此下标明确，最好使用普通循环。

3.而对于 LinkedList 由于获取一个元素 ,要从头开始向后找 ,因此建议使用 增强 for 循环，也就是 iterator。

9、Collection 和 Colections 的区别是什么？

Collection 是一个接口，它是 List、Set、Queue 等接口的父接口。在这个接口中，定义了集合类中通用的方法。

Collections 是针对集合类的一个工具类，它提供一系列静态方法实现对各种集合的查找、替换、排序、线程安全化等操作。

10、面向对象和面向过程的理解

面向对象时相对面向过程而言的。比如说：人坐地铁去上班。人面向地铁这个对象，地铁拥

有运输的功能，人通过地铁调用这个功能，只重结果，不重于过程。面向过程就如我们开门，锁孔对齐，门轴转动，门才开了，相对于结果，侧重于过程。

曾经跟朋友聊对面向对象，我说就像搭积木，找到需要的，一个一个往上累；她说这就是 JAVA 方便之处。

（1）面向过程是以函数为基础，完成各种操作，强调的是过程，例如：C 语言；

（2）面向对象以对象为基础，完成各种操作，强调的是对象和结果。

（3）面向对象是基于面向过程的。

面向对象的特点是：

（1）是一种个符合人们思考习惯的思想

（2）把复杂的事情简单化

（3）把我们从执行者变成了指挥者

举例：老师您口渴想喝茶，如果你自己去拿杯子，放茶叶，
接水这些是面向过程的；但这时您恰好知道我是个打水小能手，
又正好我很尊敬您，一见如故、一见钟情啥的，很愿意帮您去泡茶。
您调用我帮您泡茶，使用我的拿杯子、放茶叶、接水等方法。
您由执行者上升到了指挥者，让泡茶这件事更简单化。

1，实现十个多线程

2，定义一个数组，求最小值最大值与和

3，复制文件夹

4，复制文件

5，逢七的数字去掉，一百以内

1、用反射的方法修改一个被私有修饰的成员变量。

```java
public class Test4 {
    public void setProperty(Object obj, String propertyName, Object value) {
        // 根据对象获取字节码文件对象
        Class c = obj.getClass();
        Field field = null;
        try {
            // 获取该对象的propertyName成员变量
            field = c.getDeclaredField(propertyName);
        } catch (NoSuchFieldException e) {
            e.printStackTrace();
        } catch (SecurityException e) {
            e.printStackTrace();
        }
        // 取消访问检查
        field.setAccessible(true);
        try {
            // 给对象的成员变量赋值为指定的值
            field.set(obj, value);
        } catch (IllegalArgumentException e) {
            e.printStackTrace();
        } catch (IllegalAccessException e) {
            e.printStackTrace();
        }
    }
}
```

2、Map 的存取方式，用代码实现

3、一共有四个售票口同时卖 100 张票，请模拟售票过程输出时要显示是哪个窗口卖第几张票。

4、已知文件 a.txt 文件中的内容为“bcdeadferwplkou”，请编写程序读取该文件内容，并按照自然顺序排序后输出到 b.txt 文件中。即 b.txt 中的文件内容应为“abcd……”这样的顺序。

```java
public class Test3 {
    public static void main(String[] args) throws IOException {
        //封装目录
        BufferedReader br = new BufferedReader(new FileReader("a.txt"));
        BufferedWriter bw = new BufferedWriter(new FileWriter("b.txt"));

        //读取文件内容
        String line = br.readLine();
        //字符串转成字符数组
        char[] strArray = line.toCharArray();
        //对字符数组进行排序
        Arrays.sort(strArray);

        //把排序后的字符串写到指定的文件
        bw.write(new String(strArray));
        bw.newLine();//换行
```

```
            bw.flush();//刷新缓冲区

            //释放资源
            bw.close();
            br.close();
        }
}
```

首先大家要弄清楚面试的知识范围，把知识点都列出来整理好。面试问的都是一些基础和细节。

其次，最好把整理好的知识点背下来，通过自问自答来测试。因为我不喜欢死记硬背，面试前虽然都有看知识点，但没有去背。结果面试时一紧张，很多东西想不起来，或者答漏，结果可想而知。

最后，做的代码题目也是基础了，有三道，只要前面的走流程时是自己做的，应该不会有问题，注意时间就行，万一太紧张没做好，可以跳过先做其他题目，等到其他都做完在来思考、检查。做题的时候把老师忘掉就行。

1：请解释面向对象是什么?并举例说明?

答：　1、面向过程是以函数为基础，完成各种操作；

　　2、面向对象是以对象为基础。强调的是对象、结果；

　　3、以上两者都是一种思想。

　　面向对象是一种更加符合人们思考习惯；他可以将复杂问题简单化；将程序员从执行者变为指挥者。

2：类与对象的关系?

答：类就是成员变量和成员方法的一中集合，是一个抽象的概念。对象就是类的具体存在和体现。

3：类有哪些内容组成?

答：成员变量 -- 事物的属性；成员方法 -- 事物的行为，一些功能；构造函数 -- 对数据的初始化

4：局部变量和成员变量的区别?

答：　A:定义位置不同 -- 成员变量在类中；局部变量在方法中，或者方法的形式参数

　　B：初始化值的区别：成员变量：有默认初始化值；局部变量没有初始化值，想使用必须先赋值；

C：存储位置不同：成员变量在堆中，随着对象的存在而存在，对象消失他也消失；局部变量存储在栈中，随着方法的调用而存在，随着方法的调用完毕而消失。

5：构造方法的特点及注意事项?

答： 1、如果没有给出构造函数，那么系统就默认给一个无参构造器

2、假如你给了构造函数，系统就不再提供构造函数

3、构造方法重载就是构造方法接收不同的参数

6：this 关键字是什么?在什么时候使用?

答：当前类的对象引用，谁调用代表谁，那个对象调用了 this 所在的函数，this 就代表那个对象。通过 this 可以调用成员方法和成员变量；用于解决局部变量隐藏成员变量值的问题。

super()：调用父类的构造函数

this()：调用本类的构造函数

7：给成员变量赋值的方式有几种?分别怎么实现?

答：A：给静态的成员变量赋值

（1）可以通过类中的方法给成员变量赋值(私有成员变量的也可以) public void setXXX(){...}

（2）可以通过构造方法给成员变量赋值(私有成员变量的也可以)public XXX(){...}

（3）可以通过代码块完成成员变量的初始化值(私有成员变量的也可以){xxx = ...;}

（4）可以通过类名.成员变量赋值 XXX.name = ...;

B：给非静态的成员变量赋值

（1）可以通过类的方法赋值 public void setXXX(){...}

（2）可以通过对象调用 xxx.name = ...;

（3）可以通过代码块完成成员变量的初始化值{xxx = ...;}

8：static 关键字是什么?有什么特点?在什么时候使用?

答：他可以修饰成员变量和成员方法。被他修饰的内容，随着类的加载而加载，优先于对象存在，可以被类名直接调用，如果类中的某个内容被该类的所有对象所共有，那么，该内容就该被 static 修饰，没有被 static 修饰的其实就是属于对象的特有属性，特殊描述。

1. 面向对象

   面向对象是基于面向过程的，更符合人们的思考习惯，面向过程强调的是功能行为，实现过程的每一步的具体实现；面向对象强调的是对象，将功能封装成对象，通过对象去调用方法。

2. 创建多线程的两种方式

   继承 thread 类，重写 run()方法

   实现 runnable 接口，重写 run()方法，将 runnable 接口的子类对象作为实参传递给

thread 类的构造函数

3. Set List 的区别
   List 有序，可重复，元素有索引；set 无序，唯一，treeset 可以实现排序。
4. do while 和 while 的区别
   前者先执行再判断条件，至少会执行一次；后者先判断条件再执行。
5. break 和 continue 的作用
   break 中断：跳出循环，可以带标签
   continue 继续：continue　退出本次循环
   return：结束函数
6. 抽象类和接口的区别
   接口是一个特殊的抽象类，interface 修饰，抽象类 abstract 修饰，两者都不能实例化。
   前者成员变量可以是变量或常量，后者有 final 修饰只能常量
   前者成员方法可以是抽象方法或非抽象方法，可以是静态方法，后者的成员方法都是抽象方法。
   前者有构造函数，后者没有。
   抽象类被继承体现的是所属关系，抽象类中定义的是继承体系中的共性功能；接口被实现体现的是 like a 关系，接口中定义的是继承体系中的扩展功能。
7. 反射获取字节码的方式
   类名.class
   对象.getClass()
   Class.forName()
8. TreeSet 保证元素唯一性的依据，说说 hashCode()和 equals()方法，同步的锁
9. IO 流都有哪类，复制一个 Mp3 文件用哪些类，哪些方法

面向对象的思想特点

一、面向对象的思想特点

A：更符合人们的思考习惯的思想(能懒就懒)

B：把复杂的事情简单化

C：让程序员从执行者变为了指挥者

面试:跟我谈谈什么是面向对象

1 面向对象是基于面向过程的

2 告诉他什么是面向过程什么是面向对象

3 在说面向对象的思想特点

4 举例说明

1.写一个延迟加载的单例设计模式。

2.用缓冲的字符流复制一个文件。

3.有一个 ArrayList<Integer> list=new ArrayList<Integer>();将一个 String 类型的字符串放进集合中。
4.创建一个带泛型的 Map 集合实例，然后添加一堆元素进去，最后遍历输出。

1、如何理解字符串一旦初始化就不可以被改变。

```
String s = "hello";
s += "world";
System.out.println(s);
```

不可以被改变说的是常量池中的值,并不是引用 s(s 是可以重新被赋值的)

2、

```
String s1 = new String("abc");
String s2 = "abc" ;
```

s1、s2 有区别吗？
第一种方式 :第一种方式在内存中存在两个"abc"字符串,一个是在常量池中,一个是在堆内存当中.
第二种方式：在内存中只存在一个"abc"在常量池中.

一个 IO 和字符串排序结合的、一个 TCP 的、一个反射的

把以下 IP 存入一个 txt 文件，编写程序把这些 IP 按数值大小，从小到大排序并打印出来。
61.54.231.245
61.54.231.9
61.54.231.246
61.54.231.48
61.53.231.249

List 和 Set 有什么异同点？
相同点：它们都继承 Collection
不同点：List：有序（存入和取出的顺序一致）,元素都有索引(角标),元素可以重复
Set：元素不能重复，无序

成员变量和局部变量的区别	
位置的不同	成员变量直接定义在类中 局部变量定义在方法中，参数上，语句中
有效范围的不同	成员变量在这个类中有效。 局部变量只在自己所属的大括号内有效，大括号结束，局部变量失去作用域。
存储的不同	成员变量存在于堆内存中，随着对象的产生而存在，消失而消失 局部变量存在于栈内存中，随着所属区域的运行而存在，结束而释放
成员变量和静态变量的区别	
所属不同	成员变量所属于对象。所以也称为实例变量。 静态变量所属于类。所以也称为类变量。
存储不同	成员变量存在于堆内存中。 静态变量存在于方法区中。
生命周期不同	成员变量随着对象创建而存在。随着对象被回收而消失。 静态变量随着类的加载而存在。随着类的消失而消失。
调用方式不同	成员变量只能被对象所调用 。 静态变量可以被对象调用，也可以被类名调用。 所以，成员变量可以称为对象的特有数据，静态变量称为对象的共享数据。

final 特点：

1：这个关键字是一个修饰符，可以修饰类，方法，变量。

2：被 final 修饰的类是一个最终类，不可以被继承。

3：被 final 修饰的方法是一个最终方法，不可以被覆盖。

4：被 final 修饰的变量是一个常量，只能赋值一次。