**java面试题集**   
**1.C++或Java中的异常处理机制的简单原理和应用。**   
当JAVA程序违反了JAVA的语义规则时，JAVA虚拟机就会将发生的错误表示为一个异常。违反语义规则包括2种情况。一种是JAVA类库内置的语义检查。例如数组下标越界,会引发IndexOutOfBoundsException;访问null的对象时会引发NullPointerException。另一种情况就是JAVA允许程序员扩展这种语义检查，程序员可以创建自己的异常，并自由选择在何时用throw关键字引发异常。所有的异常都是java.lang.Thowable的子类。   
**2. Java的接口和C++的虚类的相同和不同处。**   
由于Java不支持多继承，而有可能某个类或对象要使用分别在几个类或对象里面的方法或属性，现有的单继承机制就不能满足要求。与继承相比，接口有更高的灵活性，因为接口中没有任何实现代码。当一个类实现了接口以后，该类要实现接口里面所有的方法和属性，并且接口里面的属性在默认状态下面都是public static,所有方法默认情况下是public.一个类可以实现多个接口。   
**3. 垃圾回收的优点和原理。并考虑2种回收机制。**   
Java语言中一个显著的特点就是引入了垃圾回收机制，使c++程序员最头疼的内存管理的问题迎刃而解，它使得Java程序员在编写程序的时候不再需要考虑内存管理。由于有个垃圾回收机制，Java中的对象不再有“作用域”的概念，只有对象的引用才有“作用域”。垃圾回收可以有效的防止内存泄露，有效的使用可以使用的内存。垃圾回收器通常是作为一个单独的低级别的线程运行，不可预知的情况下对内存堆中已经死亡的或者长时间没有使用的对象进行清楚和回收，程序员不能实时的调用垃圾回收器对某个对象或所有对象进行垃圾回收。回收机制有分代复制垃圾回收和标记垃圾回收，增量垃圾回收。   
**4. 请说出你所知道的线程同步的方法。**   
wait():使一个线程处于等待状态，并且释放所持有的对象的lock。   
sleep():使一个正在运行的线程处于睡眠状态，是一个静态方法，调用此方法要捕捉InterruptedException异常。   
notify():唤醒一个处于等待状态的线程，注意的是在调用此方法的时候，并不能确切的唤醒某一个等待状态的线程，而是由JVM确定唤醒哪个线程，而且不是按优先级。   
Allnotity():唤醒所有处入等待状态的线程，注意并不是给所有唤醒线程一个对象的锁，而是让它们竞争。   
**5. 请讲一讲析构函数和虚函数的用法和作用。**   
6. Error与Exception有什么区别？   
Error表示系统级的错误和程序不必处理的异常，   
Exception表示需要捕捉或者需要程序进行处理的异常。   
**7. 在java中一个类被声明为final类型，表示了什么意思？**   
表示该类不能被继承，是顶级类。   
**8. 描述一下你最常用的编程风格。**   
**9. heap和stack有什么区别。**   
栈是一种线形集合，其添加和删除元素的操作应在同一段完成。栈按照后进先出的方式进行处理。   
堆是栈的一个组成元素   
**10. 如果系统要使用超大整数（超过long长度范围），请你设计一个数据结构来存储这种超大型数字以及设计一种算法来实现超大整数加法运算）。**   
public class BigInt()   
{   
int[] ArrOne = new ArrOne[1000];   
String intString="";   
public int[] Arr(String s)   
{   
intString = s;   
for(int i=0;i<ArrOne.leght;i++)   
{   
**11. 如果要设计一个图形系统，请你设计基本的图形元件(Point,Line,Rectangle,Triangle)的简单实现**   
**12，谈谈final, finally, finalize的区别。**   
　　final—修饰符（关键字）如果一个类被声明为final，意味着它不能再派生出新的子类，不能作为父类被继承。因此一个类不能既被声明为 abstract的，又被声明为final的。将变量或方法声明为final，可以保证它们在使用中不被改变。被声明为final的变量必须在声明时给定初值，而在以后的引用中只能读取，不可修改。被声明为final的方法也同样只能使用，不能重载。   
　　finally—再异常处理时提供 finally 块来执行任何清除操作。如果抛出一个异常，那么相匹配的 catch 子句就会执行，然后控制就会进入 finally 块（如果有的话）。   
　　finalize—方法名。Java 技术允许使用 finalize() 方法在垃圾收集器将对象从内存中清除出去之前做必要的清理工作。这个方法是由垃圾收集器在确定这个对象没有被引用时对这个对象调用的。它是在 Object 类中定义的，因此所有的类都继承了它。子类覆盖 finalize() 方法以整理系统资源或者执行其他清理工作。finalize() 方法是在垃圾收集器删除对象之前对这个对象调用的。   
**13，Anonymous Inner Class (匿名内部类) 是否可以extends(继承)其它类，是否可以implements(实现)interface(接口)?**   
　　匿名的内部类是没有名字的内部类。不能extends(继承) 其它类，但一个内部类可以作为一个接口，由另一个内部类实现。   
**14，Static Nested Class 和 Inner Class的不同，说得越多越好(面试题有的很笼统)。**   
　　Nested Class （一般是C++的说法），Inner Class (一般是JAVA的说法)。Java内部类与C++嵌套类最大的不同就在于是否有指向外部的引用上。具体可见http: //www.frontfree.net/articles/services/view.asp?id=704&page=1   
　　注： 静态内部类（Inner Class）意味着1创建一个static内部类的对象，不需要一个外部类对象，2不能从一个static内部类的一个对象访问一个外部类对象   
第四，&和&&的区别。   
　　&是位运算符。&&是布尔逻辑运算符。   
**15，HashMap和Hashtable的区别。**   
　　都属于Map接口的类，实现了将惟一键映射到特定的值上。   
　　HashMap 类没有分类或者排序。它允许一个 null 键和多个 null 值。   
　　Hashtable 类似于 HashMap，但是不允许 null 键和 null 值。它也比 HashMap 慢，因为它是同步的。   
**16，Collection 和 Collections的区别。**   
　　Collections是个java.util下的类，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。   
**17，什么时候用assert。**   
　　断言是一个包含布尔表达式的语句，在执行这个语句时假定该表达式为 true。如果表达式计算为 false，那么系统会报告一个 Assertionerror。它用于调试目的：   
assert(a > 0); // throws an Assertionerror if a <= 0   
断言可以有两种形式：   
assert Expression1 ;   
assert Expression1 : Expression2 ;   
　　Expression1 应该总是产生一个布尔值。   
　　Expression2 可以是得出一个值的任意表达式。这个值用于生成显示更多调试信息的 String 消息。   
　　断言在默认情况下是禁用的。要在编译时启用断言，需要使用 source 1.4 标记：   
　　javac -source 1.4 Test.java   
　　要在运行时启用断言，可使用 -enableassertions 或者 -ea 标记。   
　　要在运行时选择禁用断言，可使用 -da 或者 -disableassertions 标记。   
　　要系统类中启用断言，可使用 -esa 或者 -dsa 标记。还可以在包的基础上启用或者禁用断言。   
　　可以在预计正常情况下不会到达的任何位置上放置断言。断言可以用于验证传递给私有方法的参数。不过，断言不应该用于验证传递给公有方法的参数，因为不管是否启用了断言，公有方法都必须检查其参数。不过，既可以在公有方法中，也可以在非公有方法中利用断言测试后置条件。另外，断言不应该以任何方式改变程序的状态。   
**18，GC是什么? 为什么要有GC? (基础)。**   
　　GC是垃圾收集器。Java 程序员不用担心内存管理，因为垃圾收集器会自动进行管理。要请求垃圾收集，可以调用下面的方法之一：   
System.gc()   
Runtime.getRuntime().gc()   
**19，String s = new String("xyz");创建了几个String Object?**   
　　两个对象，一个是“xyx”,一个是指向“xyx”的引用对象s。   
**20，Math.round(11.5)等於多少? Math.round(-11.5)等於多少?**   
　　Math.round(11.5)返回（long）12，Math.round(-11.5)返回（long）-11;   
**21，short s1 = 1; s1 = s1 + 1;有什么错? short s1 = 1; s1 += 1;有什么错?**   
　　short s1 = 1; s1 = s1 + 1;有错，s1是short型，s1+1是int型,不能显式转化为short型。可修改为s1 =(short)(s1 + 1) 。short s1 = 1; s1 += 1正确。   
**22，sleep() 和 wait() 有什么区别? 搞线程的最爱**   
　　sleep()方法是使线程停止一段时间的方法。在sleep 时间间隔期满后，线程不一定立即恢复执行。这是因为在那个时刻，其它线程可能正在运行而且没有被调度为放弃执行，除非(a)“醒来”的线程具有更高的优先级 (b)正在运行的线程因为其它原因而阻塞。   
　　wait()是线程交互时，如果线程对一个同步对象x 发出一个wait()调用，该线程会暂停执行，被调对象进入等待状态，直到被唤醒或等待时间到。   
**23，Java有没有goto?**   
　　Goto—java中的保留字，现在没有在java中使用。   
**24，数组有没有length()这个方法? String有没有length()这个方法？**   
　　数组没有length()这个方法，有length的属性。   
　　String有有length()这个方法。   
**25，Overload和Override的区别。Overloaded的方法是否可以改变返回值的类型?**   
　　方法的重写Overriding和重载Overloading是Java多态性的不同表现。重写Overriding是父类与子类之间多态性的一种表现，重载Overloading是一个类中多态性的一种表现。如果在子类中定义某方法与其父类有相同的名称和参数，我们说该方法被重写 (Overriding)。子类的对象使用这个方法时，将调用子类中的定义，对它而言，父类中的定义如同被“屏蔽”了。如果在一个类中定义了多个同名的方法，它们或有不同的参数个数或有不同的参数类型，则称为方法的重载(Overloading)。Overloaded的方法是可以改变返回值的类型。   
**26，Set里的元素是不能重复的，那么用什么方法来区分重复与否呢? 是用==还是equals()? 它们有何区别?**   
　　Set里的元素是不能重复的，那么用iterator()方法来区分重复与否。equals()是判读两个Set是否相等。   
　　equals()和==方法决定引用值是否指向同一对象equals()在类中被覆盖，为的是当两个分离的对象的内容和类型相配的话，返回真值。   
**27，给我一个你最常见到的runtime exception。**   
ArithmeticException, ArrayStoreException, BufferOverflowException, BufferUnderflowException, CannotRedoException, CannotUndoException, ClassCastException, CMMException, ConcurrentModificationException, DOMException, EmptyStackException, IllegalArgumentException, IllegalMonitorStateException, IllegalPathStateException, IllegalStateException,   
ImagingOpException, IndexOutOfBoundsException, MissingResourceException, NegativeArraySizeException, NoSuchElementException, NullPointerException, ProfileDataException, ProviderException, RasterFORMatException, SecurityException, SystemException, UndeclaredThrowableException, UnmodifiableSetException, UnsupportedOperationException   
**28，error和exception有什么区别?**   
　　error 表示恢复不是不可能但很困难的情况下的一种严重问题。比如说内存溢出。不可能指望程序能处理这样的情况。   
　　exception 表示一种设计或实现问题。也就是说，它表示如果程序运行正常，从不会发生的情况。   
29，List, Set, Map是否继承自Collection接口?   
List，Set是   
Map不是   
**30，abstract class和interface有什么区别?**   
　　声明方法的存在而不去实现它的类被叫做抽象类（abstract class），它用于要创建一个体现某些基本行为的类，并为该类声明方法，但不能在该类中实现该类的情况。不能创建abstract 类的实例。然而可以创建一个变量，其类型是一个抽象类，并让它指向具体子类的一个实例。不能有抽象构造函数或抽象静态方法。Abstract 类的子类为它们父类中的所有抽象方法提供实现，否则它们也是抽象类为。取而代之，在子类中实现该方法。知道其行为的其它类可以在类中实现这些方法。   
　　接口（interface）是抽象类的变体。在接口中，所有方法都是抽象的。多继承性可通过实现这样的接口而获得。接口中的所有方法都是抽象的，没有一个有程序体。接口只可以定义static final成员变量。接口的实现与子类相似，除了该实现类不能从接口定义中继承行为。当类实现特殊接口时，它定义（即将程序体给予）所有这种接口的方法。然后，它可以在实现了该接口的类的任何对象上调用接口的方法。由于有抽象类，它允许使用接口名作为引用变量的类型。通常的动态联编将生效。引用可以转换到接口类型或从接口类型转换，instanceof 运算符可以用来决定某对象的类是否实现了接口。   
**31，abstract的method是否可同时是static,是否可同时是native，是否可同时是synchronized?**   
　　都不能   
**32，接口是否可继承接口? 抽象类是否可实现(implements)接口? 抽象类是否可继承实体类(concrete class)?**   
　　接口可以继承接口。抽象类可以实现(implements)接口，抽象类是否可继承实体类，但前提是实体类必须有明确的构造函数。   
**33，启动一个线程是用run()还是start()?**   
　　启动一个线程是调用start()方法，使线程所代表的虚拟处理机处于可运行状态，这意味着它可以由JVM调度并执行。这并不意味着线程就会立即运行。run()方法可以产生必须退出的标志来停止一个线程。   
**34，构造器Constructor是否可被override?**   
　　构造器Constructor不能被继承，因此不能重写Overriding，但可以被重载Overloading。   
**35，是否可以继承String类?**   
　　String类是final类故不可以继承。   
**36，当一个线程进入一个对象的一个synchronized方法后，其它线程是否可进入此对象的其它方法?**   
　　不能，一个对象的一个synchronized方法只能由一个线程访问。   
**37，try {}里有一个return语句，那么紧跟在这个try后的finally {}里的code会不会被执行，什么时候被执行，在return前还是后?**   
　　会执行，在return前执行。   
**38，编程题: 用最有效率的方法算出2乘以8等於几?**   
　　有C背景的程序员特别喜欢问这种问题。   
　　2 << 3   
**39，两个对象值相同(x.equals(y) == true)，但却可有不同的hash code，这句话对不对?**   
　　不对，有相同的hash code。   
**40，当一个对象被当作参数传递到一个方法后，此方法可改变这个对象的属性，并可返回变化后的结果，那么这里到底是值传递还是引用传递?**   
　　是值传递。Java 编程语言只由值传递参数。当一个对象实例作为一个参数被传递到方法中时，参数的值就是对该对象的引用。对象的内容可以在被调用的方法中改变，但对象的引用是永远不会改变的。   
**41，swtich是否能作用在byte上，是否能作用在long上，是否能作用在String上?**   
　　switch（expr1）中，expr1是一个整数表达式。因此传递给 switch 和 case 语句的参数应该是 int、 short、 char 或者 byte。long,string 都不能作用于swtich。   
**42，编程题: 写一个Singleton出来。**   
　　Singleton模式主要作用是保证在Java应用程序中，一个类Class只有一个实例存在。   
　　一般Singleton模式通常有几种种形式：   
　　第一种形式：定义一个类，它的构造函数为private的，它有一个static的private的该类变量，在类初始化时实例话，通过一个public的getInstance方法获取对它的引用,继而调用其中的方法。   
public class Singleton {   
　　private Singleton(){}   
　　//在自己内部定义自己一个实例，是不是很奇怪？   
　　//注意这是private 只供内部调用   
　　private static Singleton instance = new Singleton();   
　　//这里提供了一个供外部访问本class的静态方法，可以直接访问　　   
　　public static Singleton getInstance() {   
　　　　return instance; 　　   
　　 }   
}   
　　第二种形式：   
public class Singleton {   
　　private static Singleton instance = null;   
　　public static synchronized Singleton getInstance() {   
　　//这个方法比上面有所改进，不用每次都进行生成对象，只是第一次　　　 　   
　　//使用时生成实例，提高了效率！   
　　if (instance==null)   
　　　　instance＝new Singleton();   
return instance; 　　}   
}   
其他形式：   
　　定义一个类，它的构造函数为private的，所有方法为static的。   
　　一般认为第一种形式要更加安全些   
　　Hashtable和HashMap   
　　Hashtable继承自Dictionary类，而HashMap是Java1.2引进的Map interface的一个实现   
　　HashMap允许将null作为一个entry的key或者value，而Hashtable不允许   
　　还有就是，HashMap把Hashtable的contains方法去掉了，改成containsvalue和containsKey。因为contains方法容易让人引起误解。   
　　最大的不同是，Hashtable的方法是Synchronize的，而HashMap不是，在   
多个线程访问Hashtable时，不需要自己为它的方法实现同步，而HashMap   
就必须为之提供外同步。   
Hashtable和HashMap采用的hash/rehash算法都大概一样，所以性能不会有很大的差异。   
**43.描述一下JVM加载class文件的原理机制?**   
**44.试举例说明一个典型的垃圾回收算法？**   
**45.请用java写二叉树算法，实现添加数据形成二叉树功能，并以先序的方式打印出来.**   
**46.请写一个java程序实现线程连接池功能？   
47.给定一个C语言函数，要求实现在java类中进行调用。   
48、编一段代码，实现在控制台输入一组数字后，排序后在控制台输出；   
49、列出某文件夹下的所有文件；   
50、调用系统命令实现删除文件的操作；   
51、实现从文件中一次读出一个字符的操作；   
52、列出一些控制流程的方法；   
53、多线程有哪些状态？   
54、编写了一个服务器端的程序实现在客户端输入字符然后在控制台上显示，直到输入"END"为止，让你写出客户端的程序；**   
**55、作用域public,private,protected,以及不写时的区别**   
答：区别如下：   
作用域 当前类 同一package 子孙类 其他package   
public √ √ √ √   
protected √ √ √ ×   
friendly √ √ × ×   
private √ × × ×   
不写时默认为friendly   
**56、ArrayList和Vector的区别,HashMap和Hashtable的区别**   
答：就ArrayList与Vector主要从二方面来说.   
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   
就HashMap与HashTable主要从三方面来说。   
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   
**57、char型变量中能不能存贮一个中文汉字?为什么?**   
答：是能够定义成为一个中文的，因为java中以unicode编码，一个char占16个字节，所以放一个中文是没问题的   
**58、多线程有几种实现方法,都是什么?同步有几种实现方法,都是什么?**   
答：多线程有两种实现方法，分别是继承Thread类与实现Runnable接口   
同步的实现方面有两种，分别是synchronized,wait与notify   
**59、垃圾回收机制,如何优化程序?**   
希望大家补上，谢谢   
60、float型float f=3.4是否正确?   
答:不正确。精度不准确,应该用强制类型转换，如下所示：float f=(float)3.4   
**61、介绍JAVA中的Collection FrameWork(包括如何写自己的数据结构)?**   
答：Collection FrameWork如下：   
Collection   
├List   
│├LinkedList   
│├ArrayList   
│└Vector   
│　└Stack   
└Set   
Map   
├Hashtable   
├HashMap   
└WeakHashMap   
Collection是最基本的集合接口，一个Collection代表一组Object，即Collection的元素（Elements）   
Map提供key到value的映射   
**62、Java中异常处理机制，事件机制？**   
**11、JAVA中的多形与继承？**   
希望大家补上，谢谢   
**63、抽象类与接口？**   
答：抽象类与接口都用于抽象，但是抽象类(JAVA中)可以有自己的部分实现，而接口则完全是一个标识(同时有多重继承的功能)。   
**编程题：**   
**1．现在输入n个数字，以逗号，分开；   
然后可选择升或者降序排序；   
按提交键就在另一页面显示   
  按什么 排序，结果为，  ，**   
提供reset   
答案（1）  public static String[] splitStringByComma(String source){   
           if(source==null||source.trim().equals(""))   
                   return null;   
           StringTokenizer commaToker =  new StringTokenizer(source,",");   
           String[] result = new String[commaToker.countTokens()];   
           int i=0;   
           while(commaToker.hasMoreTokens()){   
                   result[i] = commaToker.nextToken();   
                   i++;   
           }   
           return result;   
  }   
循环遍历String数组   
Integer.parseInt(String s)变成int类型   
组成int数组   
Arrays.sort(int[] a),   
a数组升序   
降序可以从尾部开始输出   
**2．金额转换，阿拉伯数字的金额转换成中国传统的形式如：   
（￥1011）－>（一千零一拾一元整）输出。**   
**3、继承时候类的执行顺序问题,一般都是选择题,问你将会打印出什么?**   
答:父类：   
package test;   
public class FatherClass   
{   
public FatherClass()   
{   
System.out.println("FatherClass Create");   
}   
}   
子类:   
package test;   
import test.FatherClass;   
public class ChildClass extends FatherClass   
{   
public ChildClass()   
{   
System.out.println("ChildClass Create");   
}   
public static void main(String[] args)   
{   
FatherClass fc = new FatherClass();   
ChildClass cc = new ChildClass();   
}   
}   
输出结果：   
C:>java test.ChildClass   
FatherClass Create   
FatherClass Create   
ChildClass Create   
**4、内部类的实现方式?**   
答：示例代码如下：   
package test;   
public class OuterClass   
{   
private class InterClass   
{   
public InterClass()   
{   
System.out.println("InterClass Create");   
}   
}   
public OuterClass()   
{   
InterClass ic = new InterClass();   
System.out.println("OuterClass Create");   
}   
public static void main(String[] args)   
{   
OuterClass oc = new OuterClass();   
}   
}   
输出结果:   
C:>java test/OuterClass   
InterClass Create   
OuterClass Create   
再一个例题：   
public class OuterClass {   
private double d1 = 1.0;   
//insert code here   
}   
You need to insert an inner class declaration at line 3. Which two inner class declarations are   
valid?(Choose two.)   
A. class InnerOne{   
public static double methoda() {return d1;}   
}   
B. public class InnerOne{   
static double methoda() {return d1;}   
}   
C. private class InnerOne{   
double methoda() {return d1;}   
}   
D. static class InnerOne{   
protected double methoda() {return d1;}   
}   
E. abstract class InnerOne{   
public abstract double methoda();   
}   
说明如下：   
一.静态内部类可以有静态成员，而非静态内部类则不能有静态成员。 故 A、B 错   
二.静态内部类的非静态成员可以访问外部类的静态变量，而不可访问外部类的非静态变量；return d1 出错。   
故 D 错   
三.非静态内部类的非静态成员可以访问外部类的非静态变量。 故 C 正确   
四.答案为C、E   
5、Java 的通信编程，编程题(或问答)，用JAVA SOCKET编程，读服务器几个字符，再写入本地显示？   
答:Server端程序:   
package test;   
import java.net.\*;   
import java.io.\*;   
public class Server   
{   
private ServerSocket ss;   
private Socket socket;   
private BufferedReader in;   
private PrintWriter out;   
public Server()   
{   
try   
{   
ss=new ServerSocket(10000);   
while(true)   
{   
socket = ss.accept();   
String RemoteIP = socket.getInetAddress().getHostAddress();   
String RemotePort = ":"+socket.getLocalPort();   
System.out.println("A client come in!IP:"+RemoteIP+RemotePort);   
in = new BufferedReader(new   
InputStreamReader(socket.getInputStream()));   
String line = in.readLine();   
System.out.println("Cleint send is :" + line);   
out = new PrintWriter(socket.getOutputStream(),true);   
out.println("Your Message Received!");   
out.close();   
in.close();   
socket.close();   
}   
}catch (IOException e)   
{   
out.println("wrong");   
}   
}   
public static void main(String[] args)   
{   
new Server();   
}   
};   
Client端程序:   
package test;   
import java.io.\*;   
import java.net.\*;   
public class Client   
{   
Socket socket;   
BufferedReader in;   
PrintWriter out;   
public Client()   
{   
try   
{   
System.out.println("Try to Connect to 127.0.0.1:10000");   
socket = new Socket("127.0.0.1",10000);   
System.out.println("The Server Connected!");   
System.out.println("Please enter some Character:");   
BufferedReader line = new BufferedReader(new   
InputStreamReader(System.in));   
out = new PrintWriter(socket.getOutputStream(),true);   
out.println(line.readLine());   
in = new BufferedReader(new InputStreamReader(socket.getInputStream()));   
System.out.println(in.readLine());   
out.close();   
in.close();   
socket.close();   
}catch(IOException e)   
{   
out.println("Wrong");   
}   
}   
public static void main(String[] args)   
{   
new Client();   
}   
};   
6、用JAVA实现一种排序，JAVA类实现序列化的方法(二种)？ 如在COLLECTION框架中，实现比较要实现什么样的接口？   
答:用插入法进行排序代码如下   
package test;   
import java.util.\*;   
class InsertSort   
{   
ArrayList al;   
public InsertSort(int num,int mod)   
{   
al = new ArrayList(num);   
Random rand = new Random();   
System.out.println("The ArrayList Sort Before:");   
for (int i=0;i<num ;i++ )   
{   
al.add(new Integer(Math.abs(rand.nextInt()) % mod + 1));   
System.out.println("al["+i+"]="+al.get(i));   
}   
}   
public void SortIt()   
{   
Integer tempInt;   
int MaxSize=1;   
for(int i=1;i<al.size();i++)   
{   
tempInt = (Integer)al.remove(i);   
if(tempInt.intValue()>=((Integer)al.get(MaxSize-1)).intValue())   
{   
al.add(MaxSize,tempInt);   
MaxSize++;   
System.out.println(al.toString());   
} else {   
for (int j=0;j<MaxSize ;j++ )   
{   
if   
(((Integer)al.get(j)).intValue()>=tempInt.intValue())   
{   
al.add(j,tempInt);   
MaxSize++;   
System.out.println(al.toString());   
break;   
}   
}   
}   
}   
System.out.println("The ArrayList Sort After:");   
for(int i=0;i<al.size();i++)   
{   
System.out.println("al["+i+"]="+al.get(i));   
}   
}   
public static void main(String[] args)   
{   
InsertSort is = new InsertSort(10,100);   
is.SortIt();   
}   
}   
JAVA类实现序例化的方法是实现java.io.Serializable接口   
Collection框架中实现比较要实现Comparable 接口和 Comparator 接口   
7、编程：编写一个截取字符串的函数，输入为一个字符串和字节数，输出为按字节截取的字符串。 但是要保证汉字不被截半个，如“我ABC”4，应该截为“我AB”，输入“我ABC汉DEF”，**6，应该输出为“我ABC”而不是“我ABC+汉的半个”。**   
答：代码如下：   
package test;   
class SplitString   
{   
String SplitStr;   
int SplitByte;   
public SplitString(String str,int bytes)   
{   
SplitStr=str;   
SplitByte=bytes;   
System.out.println("The String is:′"+SplitStr+"′;SplitBytes="+SplitByte);   
}   
public void SplitIt()   
{   
int loopCount;   
loopCount=(SplitStr.length()%SplitByte==0)?(SplitStr.length()/SplitByte):(SplitStr.length()/Split   
Byte+1);   
System.out.println("Will Split into "+loopCount);   
for (int i=1;i<=loopCount ;i++ )   
{   
if (i==loopCount){   
System.out.println(SplitStr.substring((i-1)\*SplitByte,SplitStr.length()));   
} else {   
System.out.println(SplitStr.substring((i-1)\*SplitByte,(i\*SplitByte)));   
}   
}   
}   
public static void main(String[] args)   
{   
SplitString ss = new SplitString("test中dd文dsaf中男大3443n中国43中国人   
0ewldfls=103",4);   
ss.SplitIt();   
}   
}   
**8、JAVA多线程编程。 用JAVA写一个多线程程序，如写四个线程，二个加1，二个对一个变量减一，输出。**   
希望大家补上，谢谢   
**9、STRING与STRINGBUFFER的区别。**   
答：STRING的长度是不可变的，STRINGBUFFER的长度是可变的。如果你对字符串中的内容经常进行操作，特别是内容要修改时，那么使用StringBuffer，如果最后需要String，那么使用StringBuffer的toString()方法   
**Jsp方面**   
**1、jsp有哪些内置对象?作用分别是什么?**   
答:JSP共有以下9种基本内置组件（可与ASP的6种内部组件相对应）：   
　request 用户端请求，此请求会包含来自GET/POST请求的参数   
response 网页传回用户端的回应   
pageContext 网页的属性是在这里管理   
session 与请求有关的会话期   
application servlet 正在执行的内容   
out 用来传送回应的输出   
config servlet的构架部件   
page JSP网页本身   
exception 针对错误网页，未捕捉的例外   
**2、jsp有哪些动作?作用分别是什么?**   
答:JSP共有以下6种基本动作   
jsp:include：在页面被请求的时候引入一个文件。   
jsp:useBean：寻找或者实例化一个JavaBean。   
jsp:setProperty：设置JavaBean的属性。   
jsp:getProperty：输出某个JavaBean的属性。   
jsp:forward：把请求转到一个新的页面。   
jsp:plugin：根据浏览器类型为Java插件生成OBJECT或EMBED标记   
**3、JSP中动态INCLUDE与静态INCLUDE的区别？**   
答：动态INCLUDE用jsp:include动作实现   
<jsp:include page="included.jsp" flush="true" />它总是会检查所含文件中的变化，适合用于包含动态页面，并且可以带参数   
静态INCLUDE用include伪码实现,定不会检查所含文件的变化，适用于包含静态页面   
<%@ include file="included.htm" %>   
**4、两种跳转方式分别是什么?有什么区别?**   
答：有两种，分别为：   
<jsp:include page="included.jsp" flush="true">   
<jsp:forward page= "nextpage.jsp"/>   
前者页面不会转向include所指的页面，只是显示该页的结果，主页面还是原来的页面。执行完后还会回来，相当于函数调用。并且可以带参数.后者完全转向新页面，不会再回来。相当于go to 语句。   
**Servlet方面**   
**1、说一说Servlet的生命周期?**   
答:servlet有良好的生存期的定义，包括加载和实例化、初始化、处理请求以及服务结束。这个生存期由javax.servlet.Servlet接口的init,service和destroy方法表达。   
**2、Servlet版本间(忘了问的是哪两个版本了)的不同?**   
希望大家补上，谢谢   
**3、JAVA SERVLET API中forward() 与redirect()的区别？**   
答:前者仅是容器中控制权的转向，在客户端浏览器地址栏中不会显示出转向后的地址；后者则是完全的跳转，浏览器将会得到跳转的地址，并重新发送请求链接。这样，从浏览器的地址栏中可以看到跳转后的链接地址。所以，前者更加高效，在前者可以满足需要时，尽量使用forward()方法，并且，这样也有助于隐藏实际的链接。在有些情况下，比如，需要跳转到一个其它服务器上的资源，则必须使用sendRedirect()方法。   
**4、Servlet的基本架构**   
public class ServletName extends HttpServlet {   
public void doPost(HttpServletRequest request, HttpServletResponse response) throws   
ServletException, IOException {   
}   
public void doGet(HttpServletRequest request, HttpServletResponse response) throws   
ServletException, IOException {   
}   
}   
**Jdbc、Jdo方面**   
**1、可能会让你写一段Jdbc连Oracle的程序,并实现数据查询.**   
答:程序如下：   
package hello.ant;   
import java.sql.\*;   
public class jdbc   
{   
String dbUrl="jdbc:oracle:thin:@127.0.0.1:1521:orcl";   
String theUser="admin";   
String thePw="manager";   
Connection c=null;   
Statement conn;   
ResultSet rs=null;   
public jdbc()   
{   
try{   
Class.forName("oracle.jdbc.driver.OracleDriver").newInstance();   
c = DriverManager.getConnection(dbUrl,theUser,thePw);   
conn=c.createStatement();   
}catch(Exception e){   
e.printStackTrace();   
}   
}   
public boolean executeUpdate(String sql)   
{   
try   
{   
conn.executeUpdate(sql);   
return true;   
}   
catch (SQLException e)   
{   
e.printStackTrace();   
return false;   
}   
}   
public ResultSet executeQuery(String sql)   
{   
rs=null;   
try   
{   
rs=conn.executeQuery(sql);   
}   
catch (SQLException e)   
{   
e.printStackTrace();   
}   
return rs;   
}   
public void close()   
{   
try   
{   
conn.close();   
c.close();   
}   
catch (Exception e)   
{   
e.printStackTrace();   
}   
}   
public static void main(String[] args)   
{   
ResultSet rs;   
jdbc conn = new jdbc();   
rs=conn.executeQuery("select \* from test");   
try{   
while (rs.next())   
{   
System.out.println(rs.getString("id"));   
System.out.println(rs.getString("name"));   
}   
}catch(Exception e)   
{   
e.printStackTrace();   
}   
}   
}   
**2、Class.forName的作用?为什么要用?**   
答：调用该访问返回一个以字符串指定类名的类的对象。   
**3、Jdo是什么?**   
答:JDO是Java对象持久化的新的规范，为java data object的简称,也是一个用于存取某种数据仓库中的对象的标准化API。JDO提供了透明的对象存储，因此对开发人员来说，存储数据对象完全不需要额外的代码（如JDBC API的使用）。这些繁琐的例行工作已经转移到JDO产品提供商身上，使开发人员解脱出来，从而集中时间和精力在业务逻辑上。另外，JDO很灵活，因为它可以在任何数据底层上运行。JDBC只是面向关系数据库（RDBMS)JDO更通用，提供到任何数据底层的存储功能，比如关系数据库、文件、XML以及对象数据库（ODBMS）等等，使得应用可移植性更强。   
4、在ORACLE大数据量下的分页解决方法。一般用截取ID方法，还有是三层嵌套方法。   
答:一种分页方法   
<%   
int i=1;   
int numPages=14;   
String pages = request.getParameter("page") ;   
int currentPage = 1;   
currentPage=(pages==null)?(1):{Integer.parseInt(pages)}   
sql = "select count(\*) from tables";   
ResultSet rs = DBLink.executeQuery(sql) ;   
while(rs.next()) i = rs.getInt(1) ;   
int intPageCount=1;   
intPageCount=(i%numPages==0)?(i/numPages):(i/numPages+1);   
int nextPage ;   
int upPage;   
nextPage = currentPage+1;   
if (nextPage>=intPageCount) nextPage=intPageCount;   
upPage = currentPage-1;   
if (upPage<=1) upPage=1;   
rs.close();   
sql="select \* from tables";   
rs=DBLink.executeQuery(sql);   
i=0;   
while((i<numPages\*(currentPage-1))&&rs.next()){i++;}   
%>   
//输出内容   
//输出翻页连接   
合计:<%=currentPage%>/<%=intPageCount%><a href="List.jsp?page=1">第一页</a><a   
href="List.jsp?page=<%=upPage%>">上一页</a>   
<%   
for(int j=1;j<=intPageCount;j++){   
if(currentPage!=j){   
%>   
<a href="list.jsp?page=<%=j%>">[<%=j%>]</a>   
<%   
}else{   
out.println(j);   
}   
}   
%>   
<a href="List.jsp?page=<%=nextPage%>">下一页</a><a href="List.jsp?page=<%=intPageCount%>">最后页   
</a>   
**Xml方面**   
**1、xml有哪些解析技术?区别是什么?**   
答:有DOM,SAX,STAX等   
DOM:处理大型文件时其性能下降的非常厉害。这个问题是由DOM的树结构所造成的，这种结构占用的内存较多，而且DOM必须在解析文件之前把整个文档装入内存,适合对XML的随机访问SAX:不现于DOM,SAX是事件驱动型的XML解析方式。它顺序读取XML文件，不需要一次全部装载整个文件。当遇到像文件开头，文档结束，或者标签开头与标签结束时，它会触发一个事件，用户通过在其回调事件中写入处理代码来处理XML文件，适合对XML的顺序访问   
STAX:Streaming API for XML (StAX)   
**2、你在项目中用到了xml技术的哪些方面?如何实现的?**   
答:用到了数据存贮，信息配置两方面。在做数据交换平台时，将不能数据源的数据组装成XML文件，然后将XML文件压缩打包加密后通过网络传送给接收者，接收解密与解压缩后再同XML文件中还原相关信息进行处理。在做软件配置时，利用XML可以很方便的进行，软件的各种配置参数都存贮在XML文件中。   
**3、用jdom解析xml文件时如何解决中文问题?如何解析?**   
答:看如下代码,用编码方式加以解决   
package test;   
import java.io.\*;   
public class DOMTest   
{   
private String inFile = "c:\people.xml";   
private String outFile = "c:\people.xml";   
public static void main(String args[])   
{   
new DOMTest();   
}   
public DOMTest()   
{   
try   
{   
javax.xml.parsers.DocumentBuilder builder =   
javax.xml.parsers.DocumentBuilderFactory.newInstance().newDocumentBuilder();   
org.w3c.dom.Document doc = builder.newDocument();   
org.w3c.dom.Element root = doc.createElement("老师");   
org.w3c.dom.Element wang = doc.createElement("王");   
org.w3c.dom.Element liu = doc.createElement("刘");   
wang.appendChild(doc.createTextNode("我是王老师"));   
root.appendChild(wang);   
doc.appendChild(root);   
javax.xml.transform.Transformer transformer =   
javax.xml.transform.TransformerFactory.newInstance().newTransformer();   
transformer.setOutputProperty(javax.xml.transform.OutputKeys.ENCODING, "gb2312");   
transformer.setOutputProperty(javax.xml.transform.OutputKeys.INDENT, "yes");   
transformer.transform(new javax.xml.transform.dom.DOMSource(doc),   
new   
javax.xml.transform.stream.StreamResult(outFile));   
}   
catch (Exception e)   
{   
System.out.println (e.getMessage());   
}   
}   
}   
**4、编程用JAVA解析XML的方式.**   
答:用SAX方式解析XML，XML文件如下：   
<?xml version="1.0" encoding="gb2312"?>   
<person>   
<name>王小明</name>   
<college>信息学院</college>   
<telephone>6258113</telephone>   
<notes>男,1955年生,博士，95年调入海南大学</notes>   
</person>   
事件回调类SAXHandler.java   
import java.io.\*;   
import java.util.Hashtable;   
import org.xml.sax.\*;   
public class SAXHandler extends HandlerBase   
{   
private Hashtable table = new Hashtable();   
private String currentElement = null;   
private String currentValue = null;   
public void setTable(Hashtable table)   
{   
this.table = table;   
}   
public Hashtable getTable()   
{   
return table;   
}   
public void startElement(String tag, AttributeList attrs)   
throws SAXException   
{   
currentElement = tag;   
}   
public void characters(char[] ch, int start, int length)   
throws SAXException   
{   
currentValue = new String(ch, start, length);   
}   
public void endElement(String name) throws SAXException   
{   
if (currentElement.equals(name))   
table.put(currentElement, currentValue);   
}   
}   
JSP内容显示源码,SaxXml.jsp:   
<HTML>   
<HEAD>   
<TITLE>剖析XML文件people.xml</TITLE>   
</HEAD>   
<BODY>   
<%@ page errorPage="ErrPage.jsp"   
contentType="text/html;charset=GB2312" %>   
<%@ page import="java.io.\*" %>   
<%@ page import="java.util.Hashtable" %>   
<%@ page import="org.w3c.dom.\*" %>   
<%@ page import="org.xml.sax.\*" %>   
<%@ page import="javax.xml.parsers.SAXParserFactory" %>   
<%@ page import="javax.xml.parsers.SAXParser" %>   
<%@ page import="SAXHandler" %>   
<%   
File file = new File("c:\people.xml");   
FileReader reader = new FileReader(file);   
Parser parser;   
SAXParserFactory spf = SAXParserFactory.newInstance();   
SAXParser sp = spf.newSAXParser();   
SAXHandler handler = new SAXHandler();   
sp.parse(new InputSource(reader), handler);   
Hashtable hashTable = handler.getTable();   
out.println("<TABLE BORDER=2><CAPTION>教师信息表</CAPTION>");   
out.println("<TR><TD>姓名</TD>" + "<TD>" +   
(String)hashTable.get(new String("name")) + "</TD></TR>");   
out.println("<TR><TD>学院</TD>" + "<TD>" +   
(String)hashTable.get(new String("college"))+"</TD></TR>");   
out.println("<TR><TD>电话</TD>" + "<TD>" +   
(String)hashTable.get(new String("telephone")) + "</TD></TR>");   
out.println("<TR><TD>备注</TD>" + "<TD>" +   
(String)hashTable.get(new String("notes")) + "</TD></TR>");   
out.println("</TABLE>");   
%>   
</BODY>   
</HTML>   
**EJB方面**   
**1、EJB2.0有哪些内容?分别用在什么场合? EJB2.0和EJB1.1的区别?**   
答：规范内容包括Bean提供者，应用程序装配者，EJB容器，EJB配置工具，EJB服务提供者，系统管理员。这里面，EJB容器是EJB之所以能够运行的核心。EJB容器管理着EJB的创建，撤消，激活，去活，与数据库的连接等等重要的核心工作。JSP,Servlet,EJB,JNDI,JDBC,JMS.....   
**2、EJB与JAVA BEAN的区别？**   
答:Java Bean 是可复用的组件，对Java Bean并没有严格的规范，理论上讲，任何一个Java类都可以是一个Bean。但通常情况下，由于Java Bean是被容器所创建（如Tomcat)的，所以Java Bean应具有一个无参的构造器，另外，通常Java Bean还要实现Serializable接口用于实现Bean的持久性。Java Bean实际上相当于微软COM模型中的本地进程内COM组件，它是不能被跨进程访问的。Enterprise Java Bean 相当于DCOM，即分布式组件。它是基于Java的远程方法调用（RMI）技术的，所以EJB可以被远程访问（跨进程、跨计算机）。但EJB必须被布署在诸如Webspere、WebLogic这样的容器中，EJB客户从不直接访问真正的EJB组件，而是通过其容器访问。EJB容器是EJB组件的代理，EJB组件由容器所创建和管理。客户通过容器来访问真正的EJB组件。   
**3、EJB的基本架构**   
答:一个EJB包括三个部分:   
Remote Interface 接口的代码   
package Beans;   
import javax.ejb.EJBObject;   
import java.rmi.RemoteException;   
public interface Add extends EJBObject   
{   
//some method declare   
}   
Home Interface 接口的代码   
package Beans;   
import java.rmi.RemoteException;   
import jaax.ejb.CreateException;   
import javax.ejb.EJBHome;   
public interface AddHome extends EJBHome   
{   
//some method declare   
}   
EJB类的代码   
package Beans;   
import java.rmi.RemoteException;   
import javax.ejb.SessionBean;   
import javx.ejb.SessionContext;   
public class AddBean Implements SessionBean   
{   
//some method declare   
}   
**J2EE,MVC方面**   
**1、MVC的各个部分都有那些技术来实现?如何实现?**   
答:MVC是Model－View－Controller的简写。"Model" 代表的是应用的业务逻辑（通过JavaBean，EJB组件实现）， "View" 是应用的表示面（由JSP页面产生），"Controller" 是提供应用的处理过程控制（一般是一个Servlet），通过这种设计模型把应用逻辑，处理过程和显示逻辑分成不同的组件实现。这些组件可以进行交互和重用。

**2、应用服务器与WEB SERVER的区别？**   
希望大家补上，谢谢   
**3、J2EE是什么？**   
答:Je22是Sun公司提出的多层(multi-diered),分布式(distributed),基于组件(component-base)的企业级应用模型(enterpriese application model).在这样的一个应用系统中，可按照功能划分为不同的组件，这些组件又可在不同计算机上，并且处于相应的层次(tier)中。所属层次包括客户层(clietn tier)组件,web层和组件,Business层和组件,企业信息系统(EIS)层。   
**4、WEB SERVICE名词解释。JSWDL开发包的介绍。JAXP、JAXM的解释。SOAP、UDDI,WSDL解释。**   
答：Web Service描述语言WSDL   
SOAP即简单对象访问协议(Simple Object Access Protocol)，它是用于交换XML编码信息的轻量级协议。   
UDDI 的目的是为电子商务建立标准；UDDI是一套基于Web的、分布式的、为Web Service提供的、信息注册中心的实现标准规范，同时也包含一组使企业能将自身提供的Web Service注册，以使别的企业能够发现的访问协议的实现标准。   
**5、BS与CS的联系与区别。**   
希望大家补上，谢谢   
**6、STRUTS的应用(如STRUTS架构)**   
答：Struts是采用Java Servlet/JavaServer Pages技术，开发Web应用程序的开放源码的framework。 采用Struts能开发出基于MVC(Model-View-Controller)设计模式的应用构架。 Struts有如下的主要功能：   
一.包含一个controller servlet，能将用户的请求发送到相应的Action对象。   
二.JSP自由tag库，并且在controller servlet中提供关联支持，帮助开发员创建交互式表单应用。   
三.提供了一系列实用对象：XML处理、通过Java reflection APIs自动处理JavaBeans属性、国际化的提示和消息。   
**设计模式方面**   
**1、开发中都用到了那些设计模式?用在什么场合?**   
答：每个模式都描述了一个在我们的环境中不断出现的问题，然后描述了该问题的解决方案的核心。通过这种方式，你可以无数次地使用那些已有的解决方案，无需在重复相同的工作。主要用到了MVC的设计模式。用来开发JSP/Servlet或者J2EE的相关应用。简单工厂模式等。   
**2、UML方面**   
答：标准建模语言UML。用例图,静态图(包括类图、对象图和包图),行为图,交互图(顺序图,合作图),实现图,   
**JavaScript方面**   
**1、如何校验数字型?**   
var re=/^d{1,8}$|.d{1,2}$/;   
var str=document.form1.all(i).value;   
var r=str.match(re);   
if (r==null)   
{   
sign=-4;   
break;   
}   
else{   
document.form1.all(i).value=parseFloat(str);   
}   
**CORBA方面**   
**1、CORBA是什么?用途是什么?**   
答：CORBA 标准是公共对象请求代理结构(Common Object Request Broker Architecture)，由对象管理组织 (Object Management Group，缩写为 OMG)标准化。它的组成是接口定义语言(IDL), 语言绑定(binding:也译为联编)和允许应用程序间互操作的协议。 其目的为：   
用不同的程序设计语言书写   
在不同的进程中运行   
为不同的操作系统开发   
**LINUX方面**   
**1、LINUX下线程，GDI类的解释。**   
答：LINUX实现的就是基于核心轻量级进程的"一对一"线程模型，一个线程实体对应一个核心轻量级进程，而线程之间的管理在核外函数库中实现。   
GDI类为图像设备编程接口类库。   
**JAVA华为面试题**   
**JAVA方面**   
1 面向对象的特征有哪些方面     
2 String是最基本的数据类型吗?   
3 int 和 Integer 有什么区别   
4 String 和StringBuffer的区别   
5运行时异常与一般异常有何异同？   
异常表示程序运行过程中可能出现的非正常状态，运行时异常表示虚拟机的通常操作中可能遇到的异常，是一种常见运行错误。java编译器要求方法必须声明抛出可能发生的非运行时异常，但是并不要求必须声明抛出未被捕获的运行时异常。   
6 说出一些常用的类，包,接口，请各举5个   
7 说出ArrayList,Vector, LinkedList的存储性能和特性   
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，它们都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，而LinkedList使用双向链表实现存储，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。   
8设计4个线程，其中两个线程每次对j增加1，另外两个线程对j每次减少1。写出程序。   
以下程序使用内部类实现线程，对j增减的时候没有考虑顺序问题。   
public class ThreadTest1{   
         private int j;   
         public static void main(String args[]){   
                  ThreadTest1 tt=new ThreadTest1();   
                   Inc inc=tt.new Inc();   
                   Dec dec=tt.new Dec();   
                   for(int i=0;i<2;i++){   
                            Thread t=new Thread(inc);   
                            t.start();   
                            t=new Thread(dec);   
                            t.start();   
                   }   
         }   
         private synchronized void inc(){   
                   j++;   
                  System.out.println(Thread.currentThread().getName()+"-inc:"+j);   
         }   
         private synchronized void dec(){   
                   j--;   
                  System.out.println(Thread.currentThread().getName()+"-dec:"+j);   
         }   
         class Inc implements Runnable{   
                   public void run(){   
                            for(int i=0;i<100;i++){   
                                     inc();   
                            }   
                   }   
         }   
         class Dec implements Runnable{   
                   public void run(){   
                            for(int i=0;i<100;i++){   
                                     dec();   
                            }   
                   }   
         }   
}   
9．   JSP的内置对象及方法。   
request request表示HttpServletRequest对象。它包含了有关浏览器请求的信息，并且提供了几个用于获取cookie, header, 和session数据的有用的方法。   
response response表示HttpServletResponse对象，并提供了几个用于设置送回 浏览器的响应的方法（如cookies,头信息等）   
out out 对象是javax.jsp.JspWriter的一个实例，并提供了几个方法使你能用于向浏览器回送输出结果。   
pageContext pageContext表示一个javax.servlet.jsp.PageContext对象。它是用于方便存取各种范围的名字空间、servlet相关的对象的API，并且包装了通用的servlet相关功能的方法。   
session session表示一个请求的javax.servlet.http.HttpSession对象。Session可以存贮用户的状态信息   
application applicaton 表示一个javax.servle.ServletContext对象。这有助于查找有关servlet引擎和servlet环境的信息   
config config表示一个javax.servlet.ServletConfig对象。该对象用于存取servlet实例的初始化参数。   
page page表示从该页面产生的一个servlet实例   
10.用socket通讯写出客户端和服务器端的通讯，要求客户发送数据后能够回显相同的数据。   
参见课程中socket通讯例子。   
11说出Servlet的生命周期，并说出Servlet和CGI的区别。   
Servlet被服务器实例化后，容器运行其init方法，请求到达时运行其service方法，service方法自动派遣运行与请求对应的doXXX方法（doGet，doPost）等，当服务器决定将实例销毁的时候调用其destroy方法。   
与cgi的区别在于servlet处于服务器进程中，它通过多线程方式运行其service方法，一个实例可以服务于多个请求，并且其实例一般不会销毁，而CGI对每个请求都产生新的进程，服务完成后就销毁，所以效率上低于servlet。   
12.EJB是基于哪些技术实现的?并说出SessionBean和EntityBean的区别，StatefulBean和StatelessBean的区别。   
13．EJB包括（SessionBean,EntityBean）说出他们的生命周期，及如何管理事务的？   
14．说出数据连接池的工作机制是什么?   
15同步和异步有和异同，在什么情况下分别使用他们？举例说明。   
16应用服务器有那些？   
17你所知道的集合类都有哪些？主要方法？   
18给你一个:驱动程序A,数据源名称为B,用户名称为C,密码为D,数据库表为T，请用JDBC检索出表T的所有数据。   
19．说出在JSP页面里是怎么分页的?   
页面需要保存以下参数：   
总行数：根据sql语句得到总行数   
每页显示行数：设定值   
当前页数：请求参数   
页面根据当前页数和每页行数计算出当前页第一行行数，定位结果集到此行，对结果集取出每页显示行数的行即可。   
**数据库方面：**   
**1.          存储过程和函数的区别**   
存储过程是用户定义的一系列sql语句的集合，涉及特定表或其它对象的任务，用户可以调用存储过程，而函数通常是数据库已定义的方法，它接收参数并返回某种类型的值并且不涉及特定用户表。   
**2.          事务是什么？**   
事务是作为一个逻辑单元执行的一系列操作，一个逻辑工作单元必须有四个属性，称为 ACID（原子性、一致性、隔离性和持久性）属性，只有这样才能成为一个事务：   
原子性   
事务必须是原子工作单元；对于其数据修改，要么全都执行，要么全都不执行。   
一致性   
事务在完成时，必须使所有的数据都保持一致状态。在相关数据库中，所有规则都必须应用于事务的修改，以保持所有数据的完整性。事务结束时，所有的内部数据结构（如 B 树索引或双向链表）都必须是正确的。   
隔离性   
由并发事务所作的修改必须与任何其它并发事务所作的修改隔离。事务查看数据时数据所处的状态，要么是另一并发事务修改它之前的状态，要么是另一事务修改它之后的状态，事务不会查看中间状态的数据。这称为可串行性，因为它能够重新装载起始数据，并且重播一系列事务，以使数据结束时的状态与原始事务执行的状态相同。   
持久性   
事务完成之后，它对于系统的影响是永久性的。该修改即使出现系统故障也将一直保持。   
**3.          游标的作用？如何知道游标已经到了最后？**   
游标用于定位结果集的行，通过判断全局变量@@FETCH\_STATUS可以判断是否到了最后，通常此变量不等于0表示出错或到了最后。   
**4.          触发器分为事前触发和事后触发，这两种触发有和区别。语句级触发和行级触发有何区别。**   
事前触发器运行于触发事件发生之前，而事后触发器运行于触发事件发生之后。通常事前触发器可以获取事件之前和新的字段值。   
语句级触发器可以在语句执行前或后执行，而行级触发在触发器所影响的每一行触发一次。   
中远面试题   
   1、面向对象的三个基本特征   
   2、方法重载和方法重写的概念和区别   
   3、接口和内部类、抽象类的特性   
   4、文件读写的基本类   
   \*\*5、串行化的注意事项以及如何实现串行化   
   6、线程的基本概念、线程的基本状态以及状态之间的关系   
   7、线程的同步、如何实现线程的同步   
   8、几种常用的数据结构及内部实现原理。   
   9、Socket通信(TCP、UDP区别及Java实现方式)   
  \*\*10、Java的事件委托机制和垃圾回收机制   
  11、JDBC调用数据库的基本步骤   
  \*\*12、解析XML文件的几种方式和区别   
  13、Java四种基本权限的定义   
  14、Java的国际化   
二、JSP   
   1、至少要能说出7个隐含对象以及他们的区别   
  \*\* 2、forward 和redirect的区别   
   3、JSP的常用指令   
三、servlet   
   1、什么情况下调用doGet()和doPost()？   
   2、servlet的init()方法和service()方法的区别   
   3、servlet的生命周期   
   4、如何现实servlet的单线程模式   
   5、servlet的配置   
   6、四种会话跟踪技术   
四、EJB   
   \*\*1、EJB容器提供的服务   
         主要提供声明周期管理、代码产生、持续性管理、安全、事务管理、锁和并发行管理等服务。   
   2、EJB的角色和三个对象   
         EJB角色主要包括Bean开发者 应用组装者 部署者 系统管理员 EJB容器提供者 EJB服务器提供者   
         三个对象是Remote（Local）接口、Home（LocalHome）接口，Bean类   
   2、EJB的几种类型   
         会话（Session）Bean ，实体（Entity）Bean 消息驱动的（Message Driven）Bean   
         会话Bean又可分为有状态（Stateful）和无状态（Stateless）两种   
         实体Bean可分为Bean管理的持续性（BMP）和容器管理的持续性（CMP）两种   
   3、bean 实例的生命周期   
         对于Stateless Session Bean、Entity Bean、Message Driven Bean一般存在缓冲池管理，而对于Entity Bean和Statefull Session Bean存在Cache管理，通常包含创建实例，设置上下文、创建EJB Object（create）、业务方法调用、remove等过程，对于存在缓冲池管理的Bean，在create之后实例并不从内存清除，而是采用缓冲池调度机制不断重用实例，而对于存在Cache管理的Bean则通过激活和去激活机制保持Bean的状态并限制内存中实例数量。   
   4、激活机制   
         以Statefull Session Bean 为例：其Cache大小决定了内存中可以同时存在的Bean实例的数量，根据MRU或NRU算法，实例在激活和去激活状态之间迁移，激活机制是当客户端调用某个EJB实例业务方法时，如果对应EJB Object发现自己没有绑定对应的Bean实例则从其去激活Bean存储中（通过序列化机制存储实例）回复（激活）此实例。状态变迁前会调用对应的ejbActive和ejbPassivate方法。   
   5、remote接口和home接口主要作用   
         remote接口定义了业务方法，用于EJB客户端调用业务方法   
         home接口是EJB工厂用于创建和移除查找EJB实例   
   6、客服端调用EJB对象的几个基本步骤   
一、  设置JNDI服务工厂以及JNDI服务地址系统属性   
二、  查找Home接口   
三、  从Home接口调用Create方法创建Remote接口   
四、  通过Remote接口调用其业务方法   
五、数据库   
   1、存储过程的编写   
   2、基本的SQL语句   
六、weblogic   
1、   如何给weblogic指定大小的内存?   
在启动Weblogic的脚本中（位于所在Domian对应服务器目录下的startServerName），增加set MEM\_ARGS=-Xms32m -Xmx200m，可以调整最小内存为32M，最大200M   
2、   如何设定的weblogic的热启动模式(开发模式)与产品发布模式?   
可以在管理控制台中修改对应服务器的启动模式为开发或产品模式之一。或者修改服务的启动文件或者commenv文件，增加set PRODUCTION\_MODE=true。   
3、   如何启动时不需输入用户名与密码?   
修改服务启动文件，增加 WLS\_USER和WLS\_PW项。也可以在boot.properties文件中增加加密过的用户名和密码.   
4、   在weblogic管理制台中对一个应用域(或者说是一个网站,Domain)进行jms及ejb或连接池等相关信息进行配置后,实际保存在什么文件中?   
保存在此Domain的config.xml文件中，它是服务器的核心配置文件。   
5、   说说weblogic中一个Domain的缺省目录结构?比如要将一个简单的helloWorld.jsp放入何目录下,然的在浏览器上就可打入http://主机:端口号//helloword.jsp就可以看到运行结果了? 又比如这其中用到了一个自己写的javaBean该如何办?   
Domain目录\服务器目录\applications，将应用目录放在此目录下将可以作为应用访问，如果是Web应用，应用目录需要满足Web应用目录要求，jsp文件可以直接放在应用目录中，Javabean需要放在应用目录的WEB-INF目录的classes目录中，设置服务器的缺省应用将可以实现在浏览器上无需输入应用名。   
6、   如何查看在weblogic中已经发布的EJB?   
可以使用管理控制台，在它的Deployment中可以查看所有已发布的EJB   
7、   如何在weblogic中进行ssl配置与客户端的认证配置或说说j2ee(标准)进行ssl的配置   
缺省安装中使用DemoIdentity.jks和DemoTrust.jks  KeyStore实现SSL，需要配置服务器使用Enable SSL，配置其端口，在产品模式下需要从CA获取私有密钥和数字证书，创建identity和trust keystore，装载获得的密钥和数字证书。可以配置此SSL连接是单向还是双向的。   
   8、在weblogic中发布ejb需涉及到哪些配置文件   
不同类型的EJB涉及的配置文件不同，都涉及到的配置文件包括ejb-jar.xml,weblogic-ejb-jar.xmlCMP实体Bean一般还需要weblogic-cmp-rdbms-jar.xml   
   9、EJB需直接实现它的业务接口或Home接口吗,请简述理由.   
远程接口和Home接口不需要直接实现，他们的实现代码是由服务器产生的，程序运行中对应实现类会作为对应接口类型的实例被使用。   
  10、说说在weblogic中开发消息Bean时的persistent与non-persisten的差别   
persistent方式的MDB可以保证消息传递的可靠性,也就是如果EJB容器出现问题而JMS服务器依然会将消息在此MDB可用的时候发送过来，而non－persistent方式的消息将被丢弃。   
  11、说说你所熟悉或听说过的j2ee中的几种常用模式?及对设计模式的一些看法   
       Session Facade Pattern：使用SessionBean访问EntityBean   
Message Facade Pattern：实现异步调用   
EJB Command Pattern：使用Command JavaBeans取代SessionBean，实现轻量级访问   
Data Transfer Object Factory：通过DTO Factory简化EntityBean数据提供特性   
Generic Attr, ibute Access：通过AttibuteAccess接口简化EntityBean数据提供特性   
Business Interface：通过远程（本地）接口和Bean类实现相同接口规范业务逻辑一致性   
ＥＪＢ架构的设计好坏将直接影响系统的性能、可扩展性、可维护性、组件可重用性及开发效率。项目越复杂，项目队伍越庞大则越能体现良好设计的重要性。