程序员面试之葵花宝典

1.面向对象的特征有哪些方面

抽象:抽象就是忽略一个主题中与当前目标 无关的那些方面，以便更充分地注意与当前目标有关的方面。抽象并不打算了解全部问题，而 只是选择其中的一部分， 暂时不 用部分细节。抽象包括两个方面， 一是过程抽象，二是数据抽象。

1) 继承：继承是一种联结类的层次模型，并且允许和鼓励类的重用，它提供了一种明确表述共性的方法。对象的一个新类可以从现有的类中派生，这个过程称为类继承。新类继承了原始类的特性， 新类称为原始类的派生类（子类），而原始类称为新类的基类（父类）。派生类可以从它的基类那里继承方法和实例变量， 并且类可以修改或增加新的方法使之更适合特殊的需要。
2) 封装：封装是把过程和数据包围起来，对数据的访问只能通过已定义的界面。面向对象计算始于这个基本概念，即现实世界可以被描绘成一系列完全自治、封装的对象，这些对象通过一个受保护的接口访问其他对象。
3) 多态性：多态性是指允许不同类的对象对同一消息作出响应。多态性包括参数化多态性和包含多态性。多态性语言具有灵活、抽象、行为共享、代码共享的优势，很好的解决了应用程序函数同名问题。

2、String 是最基本的数据类型吗?

基本数据类型包括 byte、int、char、long、float、double、boolean 和 short。

java.lang.String 类是 final 类型的，因此不可以继承这个类、不能修改这个类。为了提高效率、节省空间，我们应该用 StringBuffer 类。

3、int 和 Integer 有什么区别

Java 提供两种不同的类型：引用类型和原始类型（或内置类型）。int 是 java 的原始数据类型，Integer 是 java 为 int 提供的封装类。Java 为每个原始类型提供了封装类。

原始类型封装类 boolean、Boolean char、Character byte、Byte short、Short int、Integer long、Long float、Float double、Double。

引用类型和原始类型的行为完全不同，并且它们具有不同的语义。引用类型和原始类型具有不同的特征和用法，它们包括：大小和速度问题，这种类型以哪种类型的数据结构存储，当引用类型和原始类型用作某个类的实例数据时所指定的缺省值。对象引用实例变量的缺省值为 null，而原始类型实例变量的缺省值与它们的类型有关。

4、String 和 StringBuffer 的区别

JAVA 平台提供了两个类：String 和 StringBuffer，它们可以储存和操作字符串，即包含多个字符的字符数据。这个 String 类提供了数值不可改变的字符串。而这个 StringBuffer 类提供的字符串进行修改。当你知道字符数据要改变的时候你就可以使用 StringBuffer。典型地，你可以使用 StringBuffers 来动态构造字符数据。

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 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。

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、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。

EJB 包括 Session Bean、Entity Bean、Message Driven Bean，基于 JNDI、RMI、JAT 等技术实现。SessionBean 在 J2EE 应用程序中被用来完成一些服务器端的业务操作，例如访问数据库、调用其他 EJB 组件。EntityBean 被用来代表应用系统中用到的数据。

对于客户机，SessionBean 是一种非持久性对象，它实现某些在服务器上运行的业务逻辑。

对于客户机，EntityBean 是一种持久性对象，它代表一个存储在持久性存储器中的实体的对象视图，或是一个由现有企业应用程序实现的实体。

Session Bean 还可以再细分为 Stateful Session Bean 与 Stateless Session Bean ，这两种的 Session Bean 都可以将系统逻辑放在 method 之中执行，不同的是 Stateful Session Bean 可以记录呼叫者的状态，因此通常来说，一个使用者会有一个相对应的 Stateful Session Bean 的实体。Stateless Session Bean 虽然也是逻辑组件，但是他却不负责记录使用者状态，也就是说当使用者呼叫 Stateless Session Bean 的时候，EJB Container 并不会找寻特定的 Stateless Session Bean 的实体来执行这个 method。换言之，很可能数个使用者在执行某个 Stateless Session Bean 的 methods 时，会是同一个 Bean 的 Instance 在执行。从内存方面来看， Stateful Session Bean 与 Stateless Session Bean 比较， Stateful Session Bean 会消耗 J2EE Server 较多的内存，然而 Stateful Session Bean 的优势却在于他可以维持使用者的状态。

9、Collection 和 Collections 的区别。 Collection 是集合类的上级接口，继承与他的接口主要有 Set 和 List.

Collections 是针对集合类的一个帮助类，他提供一系列静态方法实现对各种集合的搜索、排序、线程安全化等操作。

10、&和&&的区别。 &是位运算符，表示按位与运算，&&是逻辑运算符，表示逻辑与（and）。

11、HashMap 和 Hashtable 的区别。 HashMap 是 Hashtable 的轻量级实现（非线程安全的实现），他们都完成了 Map 接口，主要区别在于 HashMap 允许空（null）键值（key）,由于非线程安全，效率上可能高于 Hashtable。

HashMap 允许将 null 作为一个 entry 的 key 或者 value，而 Hashtable 不允许。

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。 Hashtable 继承自 Dictionary 类，而 HashMap 是 Java1.2 引进的 Map interface 的一个实现。

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。

12、final, finally, finalize 的区别。 final 用于声明属性，方法和类，分别表示属性不可变，方法不可覆盖，类不可继承。finally 是异常处理语句结构的一部分，无论有没有异常发生,总要执行 finally 语句,它为程序提供了一个统一的出口,使程序能正常退出。finalize 是 Object 类的一个方法，在垃圾收集器执行的时候，会调用被回收对象的此方法，可以覆盖此方法提供垃圾收集时的其他资源回收，例如关闭文件等。

13、sleep() 和 wait() 有什么区别? sleep 是线程类（Thread）的方法，导致此线程暂停执行指定时间，给执行机会给其他线程，但是监控状态依然保持，到时后会自动恢复。调用 sleep 不会释放对象锁。wai

t 是 Object 类的方法，对此对象调用 wait 方法导致本线程放弃对象锁，进入等待此对象的等待锁定池，只有针对此对象发出 notify 方法（或 notifyAll）后本线程才进入对象锁定池准备获得对象锁进入运行状态。

14、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。

15、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。

16、同步和异步有何异同，在什么情况下分别使用他们？举例说明。

如果数据将在线程间共享。例如正在写的数据以后可能被另一个线程读到，或者正在读的数据可能已经被另一个线程写过了，那么这些数据就是共享数据，必须进行同步存取。当应用程序在对象上调用了一个需要花费很长时间来执行的方法，并且不希望让程序等待方法的返回时，就应该使用异步编程，在很多情况下采用异步途径往往更有效率。

17、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。

18、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。

19、forward 和 redirect 的区别 forward 是服务器请求资源，服务器直接访问目标地址的 URL，把那个 URL 的响应内容读取过来，然后把这些内容再发给浏览器，浏览器根本不知道服务器发送的内容是从哪儿来的，所以它的地址栏中还是原来的地址。 redirect 就是服务端根据逻辑,发送一个状态码,告诉浏览器重新去请求那个地址，一般来说浏览器会用刚才请求的所有参数重新请求，所以 session,request 参数都可以获取。

20、EJB 与 JAVA BEAN 的区别？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。

21、Static Nested Class 和 Inner Class 的不同。 Static Nested Class 是被声明为静态（static）的内部类，它可以不依赖于外部类实例被实例化。而通常的内部类需要在外部类实例化后才能实例化。

22、JSP 中动态 INCLUDE 与静态 INCLUDE 的区别？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" %>

23、什么时候用 assert。 assertion(断言)在软件开发中是一种常用的调试方式，很多开发语言中都支持这种机制。在实现中，assertion 就是在程序中的一条语句，它对一个 boolean 表达式进行检查，一个正确程序必须保证这个 boolean 表达式的值为 true；如果该值为 false，说明程序已经处于不正确的状态下，系统将给出警告或退出。一般来说，assertion 用于保证程序最基本、关键的正确性。assertion 检查通常在开发和测试时开启。为了提高性能，在软件发布后，assertion 检查通常是关闭的。

24、GC 是什么? 为什么要有 GC? GC 是垃圾收集的意思（Gabage Collection）,内存处理是编程人员容易出现问题的地方，忘记或者错误的内存回收会导致程序或系统的不稳定甚至崩溃，Java 提供的 GC 功能可以自动监测对象是否超过作用域从而达到自动回收内存的目的，Java 语言没有提供释放已分配内存的显示操作方法。

25、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; （s1+1 运算结果是 int 型，需要强制转换类型） short s1 = 1; s1 += 1;（可以正确编译）

26、Math.round(11.5)等於多少? Math.round(-11.5)等於多少? Math.round(11.5)==12 Math.round(-11.5)==-11 round 方法返回与参数最接近的长整数，参数加 1/2 后求其 floor.

27、String s = new String("xyz");创建了几个 String Object? 两个

28、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=ne
w 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(); } }  } }
```

29、Java 有没有 goto? java 中的保留字，现在没有在 java 中使用。

30、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。

31、EJB 包括（SessionBean,EntityBean）说出他们的生命周期，及如何管理事务的？

SessionBean：Stateless Session Bean 的生命周期是由容器决定的，当客户机发出请求要建立一个 Bean 的实例时，EJB 容器不一定要创建一个新的 Bean 的实例供客户机调用，而是随便找一个现有的实例提供给客户机。当客户机第一次调用一个 Stateful Session Bean 时，容器必须立即在服务器中创建一个新的 Bean 实例，并关联到客户机上，以后此客户机调用 Stateful Session Bean 的方法时容器会把调用分派到与此客户机相关联的 Bean 实例。EntityBean：Entity Beans 能存活相对较长的时间，并且状态是持续的。只要数据库中的数据存在，Entity beans 就一直存活。而不是按照应用程序或者服务进程来说的。即使 EJB 容器崩溃了，Entity beans 也是存活的。Entity Beans 生命周期能够被容器或者 Beans 自己管

理。EJB 通过以下技术管理实务：对象管理组织（OMG）的对象实务服务（OTS），Sun Microsystems 的 Transaction Service（JTS）、Java Transaction API（JTA），开发组（X/Open）的 XA 接口。

32、应用服务器有那些？

BEA WebLogic Server，IBM WebSphere Application Server，Oracle9i Application Server，jBoss，Tomcat

33、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

34、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。

35、List, Set, Map 是否继承自 Collection 接口？ List，Set 是，Map 不是

36、说出数据连接池的工作机制是什么？

J2EE 服务器启动时会建立一定数量的池连接，并一直维持不少于此数目的池连接。客户端程序需要连接时，池驱动程序会返回一个未使用的池连接并将其表记为忙。如果当前没有空闲连接，池驱动程序就新

建一定数量的连接，新建连接的数量有配置参数决定。当使用的池连接调用完成后，池驱动程序将此连接表记为空闲，其他调用就可以使用这个连接。

37、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

38、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。

39、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。

40、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。

41、是否可以继承 String 类?String 类是 final 类故不可以继承。

42、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。

43、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。

44、编程题: 用最有效率的方法算出 2 乘以 8 等於几? 2 << 3

45、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。

46、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 编程语言只有值传递参数。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。

47、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。

48、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(){} private static Singleton instance = new Singleton(); public static Singleton getInstance() { return instance; } } 第二种形式: public class Singleton { private static Singleton instance = null; public static synchronized Singleton getInstance() { 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

49、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。

50、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.Throwable 的子类。

51、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

52、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。

53、你所知道的集合类都有哪些？主要方法？最常用的集合类是 List 和 Map。 List 的具体实现包括 ArrayList 和 Vector，它们是可变大小的列表，比较适合构建、存储和操作任何类型对象的元素列表。 List 适用于按数值索引访问元素的情形。 Map 提供了一个更通用的元素存储方法。 Map 集合类用于存储元素对（称作"键"和"值"），其中每个键映射到一个值。

54、描述一下 JVM 加载 class 文件的原理机制?JVM 中类的装载是由 ClassLoader 和它的子类来实现的,Java ClassLoader 是一个重要的 Java 运行时系统组件。它负责在运行时查找和装入类文件的类。

55、char 型变量中能不能存贮一个中文汉字?为什么?

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

56、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?

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

57、JSP 的内置对象及方法。

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 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 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 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 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 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 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 表示从该页面产生的一个 servlet 实例

58、线程的基本概念、线程的基本状态以及状态之间的关系线程指在程序执行过程中，能够执行程序代码的一个执行单位，每个程序至少都有一个线程，也就是程序本身。Java 中的线程有四种状态分别是：运行、就绪、挂起、结束。

59、JSP 的常用指令<%@page language=”java” contenType=”text/html;charset=gb2312” session=”true” buffer=”64kb” autoFlush=”true” isThreadSafe=”true” info=”text” errorPage=”error.jsp” isErrorPage=”true” isELIgnored=”true” pageEncoding=”gb2312” import=”java.sql.*”%>isErrorPage(是否能使用 Exception 对象)，isELIgnored(是否忽略表达式) <%@include file=”filename”%><%@taglib prefix=”c”uri=”http://……”%>

60、什么情况下调用 doGet()和 doPost()？Jsp 页面中的 form 标签里的 method 属性为 get 时调用 doGet()，为 post 时调用 doPost()。

61、servlet 的生命周期 web 容器加载 servlet，生命周期开始。通过调用 servlet 的 init()方法进行 servlet 的初始化。通过调用 service()方法实现，根据请求的不同调用不同的 do***()方法。结束服务，web 容器调用 servlet 的 destroy()方法。

62、如何现实 servlet 的单线程模式 <%@ page isThreadSafe=”false”%>

63、页面间对象传递的方法 request，session，application，cookie 等

64、JSP 和 Servlet 有哪些相同点和不同点，他们之间的联系是什么？

JSP 是 Servlet 技术的扩展，本质上是 Servlet 的简易方式，更强调应用的外表表达。JSP 编译后是"类 servlet"。Servlet 和 JSP 最主要的不同点在于，Servlet 的应用逻辑是在 Java 文件中，并且完全从表示层中的 HTML 里分离开来。而 JSP 的情况是 Java 和 HTML 可以组合成一个扩展名为.jsp 的文件。JSP 侧重于视图，Servlet 主要用于控制逻辑。

65、四种会话跟踪技术 cookie,url 重写,session,隐藏域

65,jsp 的四种范围

page 否是代表与一个页面相关的对象和属性。一个页面由一个编译好的 Java servlet 类（可以带有任何的 include 指令，但是没有 include 动作）表示。这既包括 servlet 又包括被编译成 servlet 的 JSP 页

面

request 是是代表与 Web 客户机发出的一个请求相关的对象和属性。一个请求可能跨越多个页面，涉及多个 Web 组件（由于 forward 指令和 include 动作的关系）

session 是是代表与用于某个 Web 客户机的一个用户体验相关的对象和属性。一个 Web 会话可以也经常会跨越多个客户机请求

application 是是代表与整个 Web 应用程序相关的对象和属性。这实质上是跨越整个 Web 应用程序，包括多个页面、请求和会话的一个全局作用域

66、Request 对象的主要方法：

setAttribute(String name,Object)：设置名字为 name 的 request 的参数值

getAttribute(String name)：返回由 name 指定的属性值

getAttributeNames()：返回 request 对象所有属性的名字集合，结果是一个枚举的实例

getCookies()：返回客户端的所有 Cookie 对象，结果是一个 Cookie 数组

getCharacterEncoding()：返回请求中的字符编码方式

getContentLength()：返回请求的 Body 的长度

getHeader(String name)：获得 HTTP 协议定义的文件头信息

getHeaders(String name)：返回指定名字的 request Header 的所有值，结果是一个枚举的实例

getHeaderNames()：返回所以 request Header 的名字，结果是一个枚举的实例

getInputStream()：返回请求的输入流，用于获得请求中的数据

getMethod()：获得客户端向服务器端传送数据的方法

getParameter(String name)：获得客户端传送给服务器端的有 name 指定的参数值

getParameterNames()：获得客户端传送给服务器端的所有参数的名字，结果是一个枚举的实例

getParameterValues(String name)：获得有 name 指定的参数的所有值

getProtocol()：获取客户端向服务器端传送数据所依据的协议名称

getQueryString()：获得查询字符串

getRequestURI()：获取发出请求字符串的客户端地址

getRemoteAddr()：获取客户端的 IP 地址

getRemoteHost()：获取客户端的名字

getSession([Boolean create])：返回和请求相关 Session

getServerName()：获取服务器的名字

getServletPath()：获取客户端所请求的脚本文件的路径

getServerPort()：获取服务器的端口号

removeAttribute(String name)：删除请求中的一个属性

67、J2EE 是技术还是平台还是框架？J2EE 本身是一个标准，一个为企业分布式应用的开发提供的标准平台。

J2EE 也是一个框架，包括 JDBC、JNDI、RMI、JMS、EJB、JTA 等技术。

68、我们在 web 应用开发过程中经常遇到输出某种编码的字符，如 iso8859-1 等，如何输出一个某种编码的字符串？

Public String translate (String str) { String tempStr = ""; try { tempStr = new String(str.getBytes("ISO-8859-1"), "GBK"); tempStr = tempStr.trim(); } catch (Exception e) { System.err.println(e.getMessage()); } return tempStr; }

69、简述逻辑操作(&,|,^)与条件操作(&&,||)的区别。区别主要答两点：a.条件操作只能操作布尔型的,而逻辑操作不仅可以操作布尔型,而且可以操作数值型 b.逻辑操作不会产生短路

70、XML 文档定义有几种形式？它们之间有何本质区别？解析 XML 文档有哪几种方式？

a: 两种形式 dtd schema，b: 本质区别:schema 本身是 xml 的，可以被 XML 解析器解析(这也是从 DTD 上发展 schema 的根本目的)，c: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)

71、简述 synchronized 和 java.util.concurrent.locks.Lock 的异同 ？

主要相同点：Lock 能完成 synchronized 所实现的所有功能主要不同点：Lock 有比 synchronized 更精确的线程语义和更好的性能。synchronized 会自动释放锁，而 Lock 一定要求程序员手工释放，并且必须在 finally 从句中释放。

72、EJB 的角色和三个对象

一个完整的基于 EJB 的分布式计算结构由六个角色组成，这六个角色可以由不同的开发商提供，每个角色所作的工作必须遵循 Sun 公司提供的 EJB 规范，以保证彼此之间的兼容性。这六个角色分别是 EJB 组件开发者（Enterprise Bean Provider） 、应用组合者（Application Assembler）、部署者（Deployer）、EJB 服务器提供者（EJB Server Provider）、EJB 容器提供者（EJB Container Provider）、系统管理员（System Administrator）三个对象是 Remote（Local）接口、Home（LocalHome）接口，Bean 类

73、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。

74、EJB 规范规定 EJB 中禁止的操作有哪些？ 1.不能操作线程和线程 API(线程 API 指非线程对象的方法如 notify,wait 等)，2.不能操作 awt，3.不能实现服务器功能，4.不能对静态属生存取，5.不能使用 IO 操作直接存取文件系统，6.不能加载本地库.，7.不能将 this 作为变量和返回，8.不能循环调用。

75、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

76、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。

77、EJB 的激活机制 以 Stateful 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。

78、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

79、客服端调用 EJB 对象的几个基本步骤设置 JNDI 服务工厂以及 JNDI 服务地址系统属性，查找 Home 接口，从 Home 接口调用 Create 方法创建 Remote 接口，通过 Remote 接口调用其业务方法。

80、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

81、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。

82、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.

83、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。

84、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。

85、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

86、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。

87、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

88、CORBA 是什么?用途是什么？ 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。

89、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 Attribute 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。

90、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。

91、Servlet 执行时一般实现哪几个方法？public void init(ServletConfig config)；public ServletConfig getServletConfig()；public String getServletInfo()；public void service(ServletRequest request,ServletResponse response)；public void destroy()

92、常用的设计模式？说明工厂模式。 Java 中的 23 种设计模式：Factory（工厂模式），Builder（建造模式）， Factory Method（工厂方法模式），Prototype（原始模型模式），Singleton（单例模式），Facade（门面模式），Adapter（适配器模式）， Bridge（桥梁模式）， Composite（合成模式），Decorator（装饰模式）， Flyweight（享元模式）， Proxy（代理模式），Command（命令模式）， Interpreter（解释器模式）， Visitor（访问者模式），Iterator（迭代子模式）， Mediator（调停者模式），Memento（备忘录模式），Observer（观察者模式），State（状态模式），Strategy（策略模式），Template Method（模板方法模式）， Chain Of Responsibleity（责任链模式）。工厂模式：工厂模式是一种经常被使用到的模式，根据工厂模式实现的类可以根据提供的数据生成一组类中某一个类的实例，通常这一组类有一个公共的抽象父类并且实现了相同的方法，但是这些方法针对不同的数据进行了不同的操作。首先需要定义一个基类，该类的子类通过不同的方法实现了基类中的方法。然后需要定义一个

工厂类，工厂类可以根据条件生成不同的子类实例。当得到子类的实例后，开发人员可以调用基类中的方法而不必考虑到底返回的是哪一个子类的实例。

93、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。

94、排序都有哪几种方法？请列举。用 JAVA 实现一个快速排序。排序的方法有：插入排序（直接插入排序、希尔排序），交换排序（冒泡排序、快速排序），选择排序（直接选择排序、堆排序），归并排序，分配排序（箱排序、基数排序）
快速排序的伪代码。/ /使用快速排序方法对 a[0 :n- 1]排序，从 a[0 :n- 1]中选择一个元素作为 m i d d l e，该元素为支点，
把余下的元素分割为两段 left 和 r i g h t，使得 l e f t 中的元素都小于等于支点，而 right 中的元素都大于等于支点，递归地使用快速排序方法对 left 进行排序，递归地使用快速排序方法对 right 进行排序，所得结果为 l e f t + m i d d l e + r i g h t。

95、请对以下在 J2EE 中常用的名词进行解释(或简单描述)web 容器：给处于其中的应用程序组件（JSP，SERVLET）提供一个环境，使 JSP,SERVLET 直接更容器中的环境变量接口交互，不必关注其它系统问题。主要有 WEB 服务器来实现。例如：TOMCAT,WEBLOGIC,WEBSPHERE 等。该容器提供的接口严格遵守 J2EE 规范中的 WEB APPLICATION 标准。我们把遵守以上标准的 WEB 服务器就叫做 J2EE 中的 WEB 容器。EJB 容器：Enterprise java bean 容器。更具有行业领域特色。他提供给运行在其中的组件 EJB 各种管理功能。只要满足 J2EE 规范的 EJB 放入该容器，马上就会被容器进行高效率的管理。并且可以通过现成的接口来获得系统级别的服务。例如邮件服务、事务管理。JNDI：（Java Naming & Directory Interface）JAVA 命名目录服务。主要提供的功能是：提供一个目录系统，让其它各地的应用程序在其上面留下自己的索引，从而满足快速查找和定位分布式应用程序的功能。JMS：（Java Message Service）JAVA 消息服务。主要实现各个应用程序之间的通讯。包括点对点和广播。JTA：（Java Transaction API）JAVA 事务服务。提供各种分布式事务服务。应用程序只需调用其提供的接口即可。JAF：（Java Action

FrameWork）JAVA 安全认证框架。提供一些安全控制方面的框架。让开发者通过各种部署和自定义实现自己的个性安全控制策略。RMI/IIOP:（Remote Method Invocation /internet 对象请求中介协议）他们主要用于通过远程调用服务。例如，远程有一台计算机上运行一个程序，它提供股票分析服务，我们可以在本地计算机上实现对其直接调用。当然这是要通过一定的规范才能在异构的系统之间进行通信。RMI 是 JAVA 特有的。

96、JAVA 语言如何进行异常处理，关键字：throws,throw,try,catch,finally 分别代表什么意义？在 try 块中可以抛出异常吗？

Java 通过面向对象的方法进行异常处理，把各种不同的异常进行分类，并提供了良好的接口。在 Java 中，每个异常都是一个对象，它是 Throwable 类或其它子类的实例。当一个方法出现异常后便抛出一个异常对象，该对象中包含有异常信息，调用这个对象的方法可以捕获到这个异常并进行处理。Java 的异常处理是通过 5 个关键词来实现的：try、catch、throw、throws 和 finally。一般情况下是用 try 来执行一段程序，如果出现异常，系统会抛出（throws）一个异常，这时候你可以通过它的类型来捕捉（catch）它，或最后（finally）由缺省处理器来处理。用 try 来指定一块预防所有"异常"的程序。紧跟在 try 程序后面，应包含一个 catch 子句来指定你想要捕捉的"异常"的类型。throw 语句用来明确地抛出一个"异常"。throws 用来标明一个成员函数可能抛出的各种"异常"。Finally 为确保一段代码不管发生什么"异常"都被执行一段代码。可以在一个成员函数调用的外面写一个 try 语句，在这个成员函数内部写另一个 try 语句保护其他代码。每当遇到一个 try 语句，"异常"的框架就放到堆栈上面，直到所有的 try 语句都完成。如果下一级的 try 语句没有对某种"异常"进行处理，堆栈就会展开，直到遇到有处理这种"异常"的 try 语句。

97、一个".java"源文件中是否可以包括多个类（不是内部类）？有什么限制？可以。必须只有一个类名与文件名相同。

98、MVC 的各个部分都有那些技术来实现?如何实现? 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。

99、java 中有几种方法可以实现一个线程？用什么关键字修饰同步方法? stop()和 suspend()方法为何不推荐使用？有两种实现方法，分别是继承 Thread 类与实现 Runnable 接口用 synchronized 关键字修饰同步方法反对使用 stop()，是因为它不安全。它会解除由线程获取的所有锁定，而且如果对象处于一种不连贯状态，那么其他线程能在那种状态下检查和修改它们。结果很难检查出真正的问题所在。suspend()方法容易发生死锁。调用 suspend()的时候，目标线程会停下来，但却仍然持有在这之前获得的锁定。此时，其他任何线程都不能访问锁定的资源，除非被"挂起"的线程恢复运行。对任何线程来说，如果它们想恢复目标线程，同时又试图使用任何一个锁定的资源，就会造成死锁。所以不应该使用 suspend()，而应在自己的 Thread 类中置入一个标志，指出线程应该活动还是挂起。若标志指出线程应该挂起，便用 wait()命其进入等待状态。若标志指出线程应当恢复，则用一个 notify()重新启动线程。

100、java 中有几种类型的流？JDK 为每种类型的流提供了一些抽象类以供继承，请说出他们分别是哪些类？
字节流，字符流。字节流继承于 InputStream \ OutputStream，字符流继承于 InputStreamReader \ OutputStreamWriter。在 java.io 包中还有许多其他的流，主要是为了提高性能和使用方便。

101、java 中会存在内存泄漏吗，请简单描述。会。如：int i,i2; return (i-i2); //when i 为足够大的正数,i2 为足够大的负数。结果会造成溢位，导致错误。

102、java 中实现多态的机制是什么？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。

103、垃圾回收器的基本原理是什么？垃圾回收器可以马上回收内存吗？有什么办法主动通知虚拟机进行垃圾回收？对于 GC 来说，当程序员创建对象时，GC 就开始监控这个对象的地址、大小以及使用情况。

通常，GC 采用有向图的方式记录和管理堆(heap)中的所有对象。通过这种方式确定哪些对象是"可达的"，哪些对象是"不可达的"。当 GC 确定一些对象为"不可达"时，GC 就有责任回收这些内存空间。可以。程序员可以手动执行 System.gc()，通知 GC 运行，但是 Java 语言规范并不保证 GC 一定会执行。

104、静态变量和实例变量的区别？static i = 10; //常量； class A a; a.i =10;//可变

105、什么是 java 序列化，如何实现 java 序列化？

序列化就是一种用来处理对象流的机制，所谓对象流也就是将对象的内容进行流化。可以对流化后的对象进行读写操作，也可将流化后的对象传输于网络之间。序列化是为了解决在对对象流进行读写操作时所引发的问题。序列化的实现：将需要被序列化的类实现 Serializable 接口，该接口没有需要实现的方法，implements Serializable 只是为了标注该对象是可被序列化的，然后使用一个输出流(如：FileOutputStream)来构造一个 ObjectOutputStream(对象流)对象，接着，使用 ObjectOutputStream 对象的 writeObject(Object obj)方法就可以将参数为 obj 的对象写出(即保存其状态)，要恢复的话则用输入流。

106、是否可以从一个 static 方法内部发出对非 static 方法的调用？不可以,如果其中包含对象的 method()；不能保证对象初始化.

107、写 clone()方法时，通常都有一行代码，是什么？Clone 有缺省行为，super.clone();他负责产生正确大小的空间，并逐位复制。

108、在 JAVA 中，如何跳出当前的多重嵌套循环？用 break; return 方法。

109、List、Map、Set 三个接口，存取元素时，各有什么特点？List 以特定次序来持有元素，可有重复元素。Set 无法拥有重复元素,内部排序。Map 保存 key-value 值，value 可多值。

110、J2EE 是什么？J2EE 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。

111、UML 方面 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。

112、说出一些常用的类，包，接口，请各举 5 个常用的类：BufferedReader BufferedWriter FileReader FileWirter String Integer；常用的包：java.lang java.awt java.io java.util java.sql；常用的接口：Remote List Map Document NodeList

113、开发中都用到了那些设计模式?用在什么场合? 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。

114、jsp 有哪些动作?作用分别是什么? 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。

115、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)? 可以继承其他类或完成其他接口，在 swing 编程中常用此方式。

116、应用服务器与 WEB SERVER 的区别？应用服务器：Weblogic、Tomcat、Jboss； WEB SERVER：IIS、 Apache

117、BS 与 CS 的联系与区别。C/S 是 Client/Server 的缩写。服务器通常采用高性能的 PC、工作站或小型机，并采用大型数据库系统，如 Oracle、Sybase、Informix 或 SQL Server。客户端需要安装专用的客

户端软件。B/S 是 Brower/Server 的缩写，客户机上只要安装一个浏览器（Browser），如 Netscape Navigator 或 Internet Explorer，服务器安装 Oracle、Sybase、Informix 或 SQL Server 等数据库。在这种结构下，用户界面完全通过 WWW 浏览器实现，一部分事务逻辑在前端实现，但是主要事务逻辑在服务器端实现。浏览器通过 Web Server 同数据库进行数据交互。C/S 与 B/S 区别： 1．硬件环境不同: C/S 一般建立在专用的网络上, 小范围里的网络环境, 局域网之间再通过专门服务器提供连接和数据交换服务； B/S 建立在广域网之上的, 不必是专门的网络硬件环境,例与电话上网, 租用设备. 信息自己管理. 有比 C/S 更强的适应范围, 一般只要有操作系统和浏览器就行 2．对安全要求不同 ：C/S 一般面向相对固定的用户群, 对信息安全的控制能力很强. 一般高度机密的信息系统采用 C/S 结构适宜. 可以通过 B/S 发布部分可公开信息.B/S 建立在广域网之上, 对安全的控制能力相对弱, 可能面向不可知的用户。3．对程序架构不同 ： C/S 程序可以更加注重流程, 可以对权限多层次校验, 对系统运行速度可以较少考虑. B/S 对安全以及访问速度的多重的考虑, 建立在需要更加优化的基础之上. 比 C/S 有更高的要求 B/S 结构的程序架构是发展的趋势, 从 MS 的.Net 系列的 BizTalk 2000 Exchange 2000 等, 全面支持网络的构件搭建的系统. SUN 和 IBM 推的 JavaBean 构件技术等,使 B/S 更加成熟. 4．软件重用不同: C/S 程序可以不可避免的整体性考虑, 构件的重用性不如在 B/S 要求下的构件的重用性好. B/S 对的多重结构,要求构件相对独立的功能. 能够相对较好的重用.就入买来的餐桌可以再利用,而不是做在墙上的石头桌子 。5．系统维护不同 ：C/S 程序由于整体性, 必须整体考察, 处理出现的问题以及系统升级. 升级难. 可能是再做一个全新的系统， B/S 构件组成,方面构件个别的更换,实现系统的无缝升级. 系统维护开销减到最小.用户从网上自己下载安装就可以实现升级. 6．处理问题不同 ：C/S 程序可以处理用户面固定, 并且在相同区域, 安全要求高需求, 与操作系统相关. 应该都是相同的系统，B/S 建立在广域网上, 面向不同的用户群, 分散地域, 这是 C/S 无法作到的. 与操作系统平台关系最小. 7．用户接口不同： C/S 多是建立的 Window 平台上,表现方法有限,对程序员普遍要求较高，B/S 建立在浏览器上, 有更加丰富和生动的表现方式与用户交流. 并且大部分难度减低,减低开发成本.

8．信息流不同 ： C/S 程序一般是典型的中央集权的机械式处理, 交互性相对低，B/S 信息流向可变化, B-B B-C B-G 等信息、流向的变化, 更像交易中心。

118、LINUX 下线程，GDI 类的解释。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。

119、STRUTS 的应用(如 STRUTS 架构) 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。

120、Jdo 是什么？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。

121、内部类可以引用他包含类的成员吗？有没有什么限制？一个内部类对象可以访问创建它的外部类对象的内容

122、WEB SERVICE 名词解释。JSWDL 开发包的介绍。JAXP、JAXM 的解释。SOAP、UDDI,WSDL 解释。 Web ServiceWeb Service 是基于网络的、分布式的模块化组件，它执行特定的任务，遵守具体的技术规范，这些规范使得 Web Service 能与其他兼容的组件进行互操作。JAXP(Java API for XML Parsing) 定义了在 Java 中使用 DOM, SAX, XSLT 的通用的接口。这样在你的程序中你只要使用这些通用的接口，当你需要改变具体的实现时候也不需要修改代码。JAXM(Java API for XML Messaging) 是为 SOAP 通信提供访问方法和传输机制的 API。WSDL 是一种 XML 格式，用于将网络服务描述为一组端点，这些端点

对包含面向文档信息或面向过程信息的消息进行操作。这种格式首先对操作和消息进行抽象描述，然后将其绑定到具体的网络协议和消息格式上以定义端点。相关的具体端点即组合成为抽象端点（服务）。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。

JAVA 代码查错

abstract class Name { private String name; public abstract boolean isStupidName(String name) {}}大侠们，这有何错误?答案：错。abstract method 必须以分号结尾，且不带花括号。

2.public class Something { void doSomething () { private String s = ""; int l = s.length(); } } 有错吗?答案：错。局部变量前不能放置任何访问修饰符 (private，public，和 protected)。final 可以用来修饰局部变量(final 如同 abstract 和 strictfp，都是非访问修饰符，strictfp 只能修饰 class 和 method 而非 variable)。

3.abstract class Something { private abstract String doSomething ();}这好像没什么错吧?答案：错。abstract 的 methods 不能以 private 修饰。abstract 的 methods 就是让子类 implement(实现)具体细节的，怎么可以用 private 把 abstract method 封锁起来呢? (同理，abstract method 前不能加 final)。

4.public class Something { public int addOne(final int x) { return ++x; }}这个比较明显。
答案：错。int x 被修饰成 final，意味着 x 不能在 addOne method 中被修改。

5.public class Something { public static void main(String[] args) { Other o = new Other(); new Something().addOne(o); } public void addOne(final Other o) { o.i++; }} class Other { public int i;}和上面的很相似，都是关于 final 的问题，这有错吗?
答案：正确。在 addOne method 中，参数 o 被修饰成 final。如果在 addOne method 里我们修改了 o 的

reference，(比如: o = new Other();)，那么如同上例这题也是错的。但这里修改的是 o 的 member va irable(成员变量)，而 o 的 reference 并没有改变。

6.class Something { int i; public void doSomething() { System.out.println("i = " + i); }} 有什么错呢? 看不出来啊。答案: 正确。输出的是"i = 0"。int i 属於 instant variable (实例变量，或叫成员变量)。instant variable 有 default value。int 的 default value 是 0。

7.class Something { final int i; public void doSomething() { System.out.println("i = " + i); }}和上面一题只有一个地方不同，就是多了一个 final。这难道就错了吗?答案: 错。final int i 是个 final 的 instant variable (实例变量，或叫成员变量)。final 的 instant variable 没有 default value，必须在 constructor (构造器)结束之前被赋予一个明确的值。可以修改为"final int i = 0;"。

8.public class Something { public static void main(String[] args) { Something s = new Something(); System.out.println("s.doSomething() returns " + doSomething()); } public String doSomething() { return "Do something ..."; }} 看上去很完美。答案: 错。看上去在 main 里 call doSomething 没有什么问题，毕竟两个 methods 都在同一个 class 里。但仔细看，main 是 static 的。static method 不能直接 call non-static methods。可改成"System.out.println("s.doSomething() returns " + s.doSomething());"。同理，static method 不能访问 non-static instant variable。

9.此处，Something 类的文件名叫 OtherThing.javaclass Something { private static void main(String[] something_to_do) { System.out.println("Do something ..."); }} 这个好像很明显。
答案: 正确。从来没有人说过 Java 的 Class 名字必须和其文件名相同。但 public class 的名字必须和文件名相同。

10. interface A{ int x = 0;} class B{ int x =1;} class C extends B implements A { public void pX(){ System.out.println(x); } public static void main(String[] args) { new C().pX(); }}答案：错误。在编译时会发生错误(错误描述不同的 JVM 有不同的信息，意思就是未明确的 x 调用，两个 x 都匹配

（就象在同时 import java.util 和 java.sql 两个包时直接声明 Date 一样）。对于父类的变量,可以用 super.x 来明确，而接口的属性默认隐含为 public static final.所以可以通过 A.x 来明确。

11.interface Playable { void play();} interface Bounceable { void play();} interface Rollable extends Playable, Bounceable { Ball ball = new Ball("PingPang");} class Ball implements Rollable { private String name; public String getName() { return name;} public Ball(String name) { this.name = name; } public void play() { ball = new Ball("Football"); System.out.println(ball.getName()); }}这个错误不容易发现。答案: 错。"interface Rollable extends Playable, Bounceable"没有问题。interface 可继承多个 interfaces，所以这里没错。问题出在 interface Rollable 里的"Ball ball = new Ball("PingPang");"。任何在 interface 里声明的 interface variable (接口变量，也可称成员变量)，默认为 public static final。也就是说"Ball ball = new Ball("PingPang");"实际上是"public static final Ball ball = new Ball("PingPang");"。在 Ball 类的 Play()方法中，"ball = new Ball("Football");"改变了 ball 的 reference，而这里的 ball 来自 Rollable interface，Rollable interface 里的 ball 是 public static final 的，final 的 object 是不能被改变 reference 的。因此编译器将在"ball = new Ball("Football");"这里显示有错。

JAVA 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

```
import java.util.*;
public class bycomma{ 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;}
 public static void main(String args[]){ String[] s = splitStringByComma("5,8,7,4,3,9,1"); int[] ii = new int[s.length]; for(int i = 0;i<s.length;i++){ ii[i] =Integer.parseInt(s[i]); } Arrays.sort(ii); //a
```

sc for(int i=0;i<s.length;i++){ System.out.println(ii[i]); } //desc for(int i=(s.length-1);i>=0;i--) { System.out.println(ii[i]); } }}

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。

package test.format;import java.text.NumberFormat;import java.util.HashMap;public class SimpleMoneyFormat {

public static final String EMPTY = "";public static final String ZERO = "零";public static final String ONE = "壹";public static final String TWO = "贰";public static final String THREE = "叁";public static final String FOUR = "肆"; public static final String FIVE = "伍";public static final String SIX = "陆";public static final String SEVEN = "柒"; public static final String EIGHT = "捌"; public static final String NINE = "玖"; public static final String TEN = "拾"; public static final String HUNDRED = "佰"; public static final String THOUSAND = "仟"; public static final String TEN_THOUSAND = "万"; public static final String HUNDRED_MILLION = "亿"; public static final String YUAN = "元"; public static final String JIAO = "角"; public static final String FEN = "分"; public static final String DOT = "."; private static SimpleMoneyFormat formatter = null;private HashMap chineseNumberMap = new HashMap(); private HashMap chineseMoneyPattern = new HashMap(); private NumberFormat numberFormat = NumberFormat.getInstance(); private SimpleMoneyFormat() { numberFormat.setMaximumFractionDigits(4); numberFormat.setMinimumFractionDigits(2); numberFormat.setGroupingUsed(false); chineseNumberMap.put("0", ZERO); chineseNumberMap.put("1", ONE); chineseNumberMap.put("2", TWO); chineseNumberMap.put("3", THREE); chineseNumberMap.put("4", FOUR); chineseNumberMap.put("5", FIVE); chineseNumberMap.put("6", SIX); chineseNumberMap.put("7", SEVEN); chineseNumberMap.put("8", EIGHT); chineseNumberMap.put("9", NINE); chineseNumberMap.put(DOT, DOT); chineseMoneyPattern.put("1", TEN); chineseMoneyPattern.put("2", HUNDRED); chineseMoneyPattern.put("3", THOUSAND); chineseMoneyPattern.put("4", TEN_THOUSAND); chineseMoneyPattern.put("5", TEN); chineseMoneyPattern.put("6", HUN

```
DRED); chineseMoneyPattern.put("7", THOUSAND); chineseMoneyPattern.put("8", HUNDRED_MILLI
ON); }
 public static SimpleMoneyFormat getInstance() {  if (formatter == null)  formatter = new Simple
MoneyFormat(); return formatter; }
  public String format(String moneyStr) {  checkPrecision(moneyStr);  String result; result = conv
ertToChineseNumber(moneyStr);  result = addUnitsToChineseMoneyString(result);  return result;  }
 public String format(double moneyDouble) { return format(numberFormat.format(moneyDouble));
 }  public String format(int moneyInt) {  return format(numberFormat.format(moneyInt));
  }  public String format(long moneyLong) {  return format(numberFormat.format(moneyLong)); }
 public String format(Number moneyNum) {  return format(numberFormat.format(moneyNum)); }
 private String convertToChineseNumber(String moneyStr) {  String result; StringBuffer cMoneyStri
ngBuffer = new StringBuffer(); for (int i = 0; i < moneyStr.length(); i++) {  cMoneyStringBuffer.
append(chineseNumberMap.get(moneyStr.substring(i, i + 1))); }
   //拾佰仟万亿等都是汉字里面才有的单位，加上它们 int indexOfDot = cMoneyStringBuffer.indexOf
(DOT);  int moneyPatternCursor = 1;  for (int i = indexOfDot - 1; i > 0; i--) {  cMoneyStringBu
ffer.insert(i, chineseMoneyPattern.get(EMPTY + moneyPatternCursor));  moneyPatternCursor = mo
neyPatternCursor == 8 ? 1 : moneyPatternCursor + 1;  }  String fractionPart = cMoneyStringBuff
er.substring(cMoneyStringBuffer.indexOf("."));  cMoneyStringBuffer.delete(cMoneyStringBuffer.indexO
f("."), cMoneyStringBuffer.length()); while (cMoneyStringBuffer.indexOf("零拾") != -1) {
     cMoneyStringBuffer.replace(cMoneyStringBuffer.indexOf("零拾"), cMoneyStringBuffer.indexOf("零
拾") + 2, ZERO);
   } while (cMoneyStringBuffer.indexOf("零佰") != -1) {
     cMoneyStringBuffer.replace(cMoneyStringBuffer.indexOf("零佰"), cMoneyStringBuffer.indexOf("零
佰") + 2, ZERO);
   }while (cMoneyStringBuffer.indexOf("零仟") != -1) {
```

```
    cMoneyStringBuffer.replace(cMoneyStringBuffer.indexOf("零仟"), cMoneyStringBuffer.indexOf("零仟") + 2, ZERO);
   }while (cMoneyStringBuffer.indexOf("零万") != -1) {
    cMoneyStringBuffer.replace(cMoneyStringBuffer.indexOf("零万"), cMoneyStringBuffer.indexOf("零万") + 2, TEN_THOUSAND);
   } while (cMoneyStringBuffer.indexOf("零亿") != -1) {
    cMoneyStringBuffer.replace(cMoneyStringBuffer.indexOf("零亿"), cMoneyStringBuffer.indexOf("零亿") + 2, HUNDRED_MILLION); } while (cMoneyStringBuffer.indexOf("零零") != -1) {
    cMoneyStringBuffer.replace(cMoneyStringBuffer.indexOf("零零"), cMoneyStringBuffer.indexOf("零零") + 2, ZERO);
   } if (cMoneyStringBuffer.lastIndexOf(ZERO) == cMoneyStringBuffer.length() - 1) cMoneyStringBuffer.delete(cMoneyStringBuffer.length() - 1, cMoneyStringBuffer.length());  cMoneyStringBuffer.append(fractionPart);
  result = cMoneyStringBuffer.toString();  return result; }private String addUnitsToChineseMoneyString(String moneyStr) {  String result; StringBuffer cMoneyStringBuffer = new StringBuffer(moneyStr); int indexOfDot = cMoneyStringBuffer.indexOf(DOT);  cMoneyStringBuffer.replace(indexOfDot, indexOfDot + 1, YUAN); cMoneyStringBuffer.insert(cMoneyStringBuffer.length() - 1, JIAO); cMoneyStringBuffer.insert(cMoneyStringBuffer.length(), FEN); if (cMoneyStringBuffer.indexOf("零角零分") != -1)//没有零头，加整
  cMoneyStringBuffer.replace(cMoneyStringBuffer.indexOf("零角零分"), cMoneyStringBuffer.length(), "整"); else  if (cMoneyStringBuffer.indexOf("零分") != -1)//没有零分，加整  cMoneyStringBuffer.replace(cMoneyStringBuffer.indexOf("零分"), cMoneyStringBuffer.length(), "整");
   else { if(cMoneyStringBuffer.indexOf("零角")!=-1)
      cMoneyStringBuffer.delete(cMoneyStringBuffer.indexOf("零角"),cMoneyStringBuffer.indexOf("零角")+2);
```

tmpBuffer.append("整"); } result = cMoneyStringBuffer.toString(); return result; } private voi d checkPrecision(String moneyStr) {

int fractionDigits = moneyStr.length() - moneyStr.indexOf(DOT) - 1; if (fractionDigits > 2) thro w new RuntimeException("金额" + moneyStr + "的小数位多于两位。"); //精度不能比分低 }public static void main(String args[]) { System.out.println(getInstance().format(new Double(10010001.0 1))); }}

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) { Fa therClass 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 In terClass { 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 v alid?(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

3、 接口和内部类、抽象类的特征答：接口：在一个类里，只有申明没有实现。内部类：是在一个类的内部定义的一个类；抽象类：是以 abstract 定义的，里面至少有一个抽象方法。

4、 文件读写的基本类

答：File Reader 类和 FileWriter 类分别继承自 Reader 类和 Writer 类。FileReader 类用于读取文件，File Writer 类用于将数据写入文件，这两各类在使用前，都必须要调用其构造方法创建相应的对象，然后调用相应的 read()或 write()方法。

6、 线程的基本概念、线程的本状态以及状态之间的关系

•新建 (Born) ：新建的线程处于新建状态•就绪 (Ready) ：在创建线程后，它将处于就绪状态，等待 start() 方法被调用•运行 (Running) ：线程在开始执行时进入运行状态•睡眠 (Sleeping) ：线程的执行可通过使用 sleep() 方法来暂时中止。在睡眠后，线程将进入就绪状态•等待 (Waiting) ：如果调用了 wait() 方法，线程将处于等待状态。用于在两个或多个线程并发运行时。•挂起 (Suspended) ：在临时停止或中断线程的执行时，线程就处于挂起状态。•恢复 (Resume) ：在挂起的线程被恢复执行时，可以说它已被恢复。•阻塞 (Blocked) – 在线程等待一个事件时（例如输入/输出操作），就称其处于阻塞状态。•死亡 (Dead) – 在 run() 方法已完成执行或其 stop() 方法被调用之后，线程就处于死亡状态。 5、 串行化的注意事项以及如何实现串行化答：如果有循环引用是不可以串行化的。对象输出流的 WriteObject 方法和 对象输入流的 ReadObect 方法

7、 线程的同步、如何实现线程的同步答：当两个或多个线程同时访问同一个变量，并且以个线程需要修改这个变量。就要用到线程同步。在 Java 中，同步是通过 synchronized 关键字来定义的。 诺是想同步化某程序段，可以使用 synchronized(object){}方法，其中{}内的程序语句被同步化。

9、 socket 通信（tcp/udp 区别及 JAVA 的实现方式）TCP——传输控制协议，具有极高的可靠性，保证数据包按照顺序准确到达，但其也有着很高的额外负担。UDP——使用者数据元协议，并不能保证数据包

会被成功的送达，也不保证数据包到达的顺序，但其传输速度很快。大多数我们会使用 TCP，偶尔才会动用 UDP，如声音讯号，即使少量遗失，也无　　关紧要。

10、 JAVA 的事件委托机制和垃圾回收机制

java 事件委托机制的概念,一个源产生一个事件并将它送到一个或多个监听器那里。在这种方案中，监听器简单的等待，直到它收到一个事件。一旦事件被接受，监听器将处理这个事件，然后返回。垃圾回收机制 垃圾收集是将分配给对象但不在使用的内存回收或释放的过程。如果一个对象没有指向它的引用或者其赋值为 null,则次对象适合进行垃圾回收

11、 JDBC 调用数据库的基本步骤导入必要的类，装入 JDBC 驱动程序，识别数据源，分配一个 Connection 对象，分配一个 Statement 对象，使用 Statement 执行一个查询，从返回的 ResultSet 对象中检索数据，关闭 ResultSet，关闭 Statement 对象，关闭 Connection 对象

12、 解析 XML 文件的几种方式和区别答：Dom 解析 在内存中创建一个 DOM 树，能随机访问文件内容，也可以修改原文件内容 SAX 解析 线性解析，不能随机访问，也无法修改原文件 Dom 解析要先用 SAX 解析创建 DOM 树

13、 JAVA 的四种基本权限的定义 public private protected 默认

14、 JAVA 的国际化 答：Java 中提供了诺干国际化明感类，来实现国际化的。例如：dateformat timezone 等等。

2、 forward 和 rederect 的区别答：redirect 重定向到另外一个页面，重新开始一个请求 forward 跳转到另外一个页面， 请求不断开

3、 jsp 的常用的命令答：page, include, talib, forward,

1、 什么情况下调用 doget()和什么情况 dopost 答：当表单提交时 method 设置的 是 get 就调用 doget 方法，如果是 post 就调用 dopost 方法。 http get 方法请求一页面，调用 doget() http post 方法请求一页面，调用 dopost()

2、 servlet 的 init()方法和 service()方法的区别答：初始化时调用 init()方法有请求到达时调用 service()方法，service()根据请求的类型，调用 doget()或 depost()等方法

5、 servlet 的配置

<web-app><servlet><servlet-name>Admin</servlet-name><servlet-class>jb-aptech.adminservlet</servlet-class><init-param><param-name>email</param-name><param-value>admin@jb-aptech.com.cn</param-value></init-param> </servlet></web-app>

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

7、 客服端口调用 EJB 对象的几个基本步骤答;设置 JNDI 服务工厂以及 JNDI 服务地址系统属性，查找 Home 接口，从 Home 接口调用 Create 方法创建 Remote 接口通过 Remote 接口调用其业务方法

12、 java 的调试如何进行。答：jdb 是 java 的调试器，类似于 UNIX 系统的调试器 dbx,jdb 使用 Java 调试器应用程序接口来完成对本地或远程的 Java 调试器的调用工作。一般是在要测试的代码段想控制台打印消息。

13、 java 中对象之间的通讯采用什么方法。答：直接调用另一对象方法来进行通讯以及数据的交换。

15、 tcp/ip 在连接是有几次握手？释放是有几次握手？答：建立连接是 2 次,释放是 3 次。

16、 谈谈你对 swing mvc 模式的理解？

答：Swing 号称是完全按照 MVC 的思路来进行设计的。在设计开始前，Swing 的希望能够达到的目标就包括：

模型驱动（Model-Driven）的编程方式。 提供一套单一的 API，但是能够支持多种视感（look-and-feel），为用户提供不同的界面。 严格的说，Swing 中的 MVC 实际上是 MVC 的一个变体：M-VC。 Swing 中只显示的定义了 Model 接口，而在一个 UI 对象中集成了视图和控制器的部分机制。View 和 Control 比较松散的交叉组合在一起，而更多的控制逻辑是在事件监听者部分引入的。 但是，这并没有妨碍在 Swing 中体现 MVC 的精髓。事实上，在 Swing 的开发初期，Swing 确实是按照标准的 MVC 模式来设计的，但是很快的问题就出现了：View 和 Controller 实际上是紧密耦合的，很难作出一个能够适应不同 View 的一般化的 Controller 来，而且，一般也没有很大的必要。

17、 Java 中线程间怎么通讯？什么叫僵死线程？答：线程之间可以通过管道进行通讯。

18、 Java 程序怎么优化？答：提高 JAVA 的性能，一般考虑如下的四个主要方面：
程序设计的方法和模式 （2） JAVA 布署的环境。 （3） JAVA 应用程序的实现 (4) 硬件和操作系统
为了提高 JAVA 程序的性能，需要遵循如下的六个步骤。 a) 明确对性能的具体要求 b) 了解当前程序的性能 c) 找到程序的性能瓶颈 d) 采取适当的措施来提高性能 e) 只进行某一方面的修改来提高性能 f) 返回到步骤 c,继续作类似的工作，一直达到要求的性能为止。

21、 在 java 中如何进行 socket 编程。答：Sockets 有两种主要的操作方式:面向连接的和无连接的。
无连接的操作使用数据报协议.这个模式下的 socket 不需要连接一个目的的 socket,它只是简单地投出数据报.无连接的操作是快速的和高效的,但是数据安全性不佳.面向连接的操作使用 TCP 协议.一个这个模式下的 socket 必须在发送数据之前与目的地的 socket 取得一个连接.一旦连接建立了,sockets 就可以使用一个流接口:打开-读-写-关闭.所有的发送的信息都会在另一端以同样的顺序被接收.面向连接的操作比无连接的操作效率更低,但是数据的安全性更高.
在服务器，使用 ServerSocket 监听指定的端口，端口可以随意指定（由于 1024 以下的端口通常属于保留端口，在一些操作系统中不可以随意使用，所以建议使用大于 1024 的端口），等待客户连接请求，客户连接后，会话产生；在完成会话后，关闭连接。在客户端，使用 Socket 对网络上某一个服务器的某一个

端口发出连接请求，一旦连接成功，打开会话；会话完成后，关闭 Socket。客户端不需要指定打开的端口，通常临时的、动态的分配一个 1024 以上的端口。

22、 用 java 怎样实现多线程？线程有那些状态？答：Java 中实现多线程的方法有两种，一是继承 java.lang 包中的 Thread 类,二是用户自己的类实现 Runnable 接口。初始状态，就绪状态，阻塞状态，运行状态，死亡状态。

23、 编译 java 用那些命令？答：javac 编译命令。Help 命令可以帮助你得到你想要的命令。

24、 同时编译 java 两个类应带什么参数？答：CLASSPATH

动态查询如何实现？表的结构变化后，如果不需要修改程序，如何设计和实现查询？答：讲查询封装进存储过程中，通过调用存储过程实现动态调用；表结构发生变化后修改相应的存储过程即可再不修改程序的情况下实现查询。

2、 如何优化数据库，如何提高数据库的性能？答：优化数据库主要是优化查询语句，通过高性能的查询语句提高数据库的性能。

3、 设计数据库应注意那些问题答：首先应尽量满足三范式的要求，在一定程度上打破 3 范式的要求以提高数据库的性能。

4、 表与表之间的关联关系答：分为 3 种：一对一、一对多、多对多。

5、 主键和外键的区别答：主键在本表中是唯一的、不可唯空的，外键可以重复可以唯空；外键和另一张表的主键关联，不能创建对应表中不存在的外键。

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 InterruptedExceptio n 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。

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。

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．写一个方法，实现字符串的反转，如：输入 abc，输出 cba

public static String reverse(String s){int length=s.length();StringBuffer result=new StringBuffer(length);for(int i=length-1;i>=0;i--)result.append(s.charAt(i));return result.toString();}

3. 数据类型之间的转换　如何将数值型字符转换为数字（Integer，Double）　如何将数字转换为字符　如何去小数点前两位，并四舍五入。

4. 日期和时间　如何取得年月日，小时分秒 Date dat=new Date();dat.getYear();dat.getMonth();dat.getDay();dat.getHours();...

如何取得从 1970 年到现在的毫秒数 long now=dat.getTime();如何获取某个日期是当月的最后一天如何格式化日期

DateFormate df=DateFormate.getInstance();df.Format(dat);

6. 文件和目录（I/O）操作　如何列出某个目录下的所有文件　如何列出某个目录下的所有子目录　判断一个文件或目录是否存在　如何读写文件

9. Java 中访问数据库的步骤，Statement 和 PreparedStatement 之间的区别。PreparedStatement 对象与 Statement 对象的不同点在于它的 SQL 语句是预编译过的，并且可以有占位符使用运行时参数。

10.在 Web 开发中需要处理 HTML 标记时，应做什么样的处理，要筛选那些字符（< > & ‘”）

如果 include2.html 的内容变化了，那么使用包含指令方式浏览器请求 jsp 页面显示的还是以前的内容，但是使用包含动作方式，浏览器请求 jsp 页面显示的就是新的内容。

4. 描述 Cookie 和 Session 的作用，区别和各自的应用范围，Session 工作原理。

Cookie 是在客户端开辟的一块可长期存储用户信息的地方；8. 描述一下你最常用的编程风格。(1) 类名首字母应该大写。字段、方法以及对象（句柄）的首字母应小写。对于所有标识符，其中包含的所有单词都应紧靠在一起，而且大写中间单词的首字母。Java 包（Package）属于一种特殊情况：它们全都是小写字母，即便中间的单词亦是如此。对于域名扩展名称，如 com，org，net 或者 edu 等，全部都应小写（这也是 Java 1.1 和 Java 1.2 的区别之一）。(2) 为了常规用途而创建一个类时，请采取"经典形式"，并包含对下述元素的定义：equals()hashCode()toString()clone()（implement Cloneable）implement Serializable(3) 对于自己创建的每一个类，都考虑置入一个 main()，其中包含了用于测试那个类的代码。为使用一个项目中的类，我们没必要删除测试代码。若进行了任何形式的改动，可方便地返回测试。这些代码也可作为如何使用类的一个示例使用。(4) 应将方法设计成简要的、功能性单元，用它描述和实现一个不连续的类接口部分。理想情况下，方法应简明扼要。若长度很大，可考虑通过某种方式将其分割成较短的几个方法。这样做也便于类内代码的重复使用（有些时候，方法必须非常大，但它们仍应只做同样的一件事情）。(5) 设计一个类时，请设身处地为客户程序员考虑一下（类的使用方法应该是非常明确的）。然后，再设身处地为管理代码的人考虑一下（预计有可能进行哪些形式的修改，想想用什么方法可把它们变得更简单）。(6) 使类尽可能短小精悍，而且只解决一个特定的问题。下面是对类设计的一些建议：■一个复杂的开关语句：考虑采用"多形"机制■数量众多的方法涉及到类型差别极大的操作：考虑用几个类来分别实现

■许多成员变量在特征上有很大的差别：考虑使用几个类(7) 让一切东西都尽可能地"私有"——private。可使库的某一部分"公共化"（一个方法、类或者一个字段等等），就永远不能把它拿出。若强行拿出，就可能破坏其他人现有的代码，使他们不得不重新编写和设计。若只公布自己必须公布的，就可放心大胆地改变其他任何东西。在多线程环境中，隐私是特别重要的一个因素——只有 private 字段才能在非同步使用的情况下受到保护。(8) 谨惕"巨大对象综合症"。对一些习惯于顺序编程思维、且初涉 OOP 领域的新手，往往喜欢先写一个顺序执行的程序，再把它嵌入一个或两个巨大的对象里。根据编程原理，对象表达的应该是应用程序的概念，而非应用程序本身。(9) 若不得已进行一些不太雅观的编程，至少应该把那些代码置于一个类的内部。(10) 任何时候只要发现类与类之间结合得非常紧密，就需要考虑是否采用

内部类，从而改善编码及维护工作（参见第 14 章 14.1.2 小节的“用内部类改进代码”）。(11) 尽可能细致地加上注释，并用 javadoc 注释文档语法生成自己的程序文档。(12) 避免使用“魔术数字”，这些数字很难与代码很好地配合。如以后需要修改它，无疑会成为一场噩梦，因为根本不知道“100”到底是指“数组大小”还是“其他全然不同的东西”。所以，我们应创建一个常数，并为其使用具有说服力的描述性名称，并在整个程序中都采用常数标识符。这样可使程序更易理解以及更易维护。(13) 涉及构建器和异常的时候，通常希望重新丢弃在构建器中捕获的任何异常——如果它造成了那个对象的创建失败。这样一来，调用者就不会以为那个对象已正确地创建，从而盲目地继续。(14) 当客户程序员用完对象以后，若你的类要求进行任何清除工作，可考虑将清除代码置于一个良好定义的方法里，采用类似于 cleanup()这样的名字，明确表明自己的用途。除此以外，可在类内放置一个 boolean（布尔）标记，指出对象是否已被清除。在类的 finalize()方法里，请确定对象已被清除，并已丢弃了从 RuntimeException 继承的一个类（如果还没有的话），从而指出一个编程错误。在采取象这样的方案之前，请确定 finalize()能够在自己的系统中工作（可能需要调用 System.runFinalizersOnExit(true)，从而确保这一行为）。(15) 在一个特定的作用域内，若一个对象必须清除（非由垃圾收集机制处理），请采用下述方法：初始化对象；若成功，则立即进入一个含有 finally 从句的 try 块，开始清除工作。(16) 若在初始化过程中需要覆盖（取消）finalize()，请记住调用 super.finalize()（若 Object 属于我们的直接超类，则无此必要）。在对 finalize()进行覆盖的过程中，对 super.finalize()的调用应属于最后一个行动，而不应是第一个行动，这样可确保在需要基础类组件的时候它们依然有效。(17) 创建大小固定的对象集合时，请将它们传输至一个数组（若准备从一个方法里返回这个集合，更应如此操作）。这样一来，我们就可享受到数组在编译期进行类型检查的好处。此外，为使用它们，数组的接收者也许并不需要将对象“造型”到数组里。(18) 尽量使用 interfaces，不要使用 abstract 类。若已知某样东西准备成为一个基础类，那么第一个选择应是将其变成一个 interface（接口）。只有在不得不使用方法定义或者成员变量的时候，才需要将其变成一个 abstract（抽象）类。接口主要描述了客户希望做什么事情，而一个类则致力于（或允许）具体的实施细节。(19) 在构建器内部，只进行那些将对象设为正确状态所需的工作。尽可能地避免调用其他方法，因为那些方法可能被其他人覆盖或取消，从而在构建过程中产生不可预知的结果（参见第 7 章的详细说明）。(20) 对象不应只是简单地容纳一些数据；它们的行为也应得到良好的定义。(21) 在现成类的基础上创建新类时，请首先选择“新

建”或“创作”。只有自己的设计要求必须继承时，才应考虑这方面的问题。若在本来允许新建的场合使用了继承，则整个设计会变得没有必要地复杂。(22) 用继承及方法覆盖来表示行为间的差异，而用字段表示状态间的区别。一个非常极端的例子是通过对不同类的继承来表示颜色，这是绝对应该避免的：应直接使用一个“颜色”字段。(23) 为避免编程时遇到麻烦，请保证在自己类路径指到的任何地方，每个名字都仅对应一个类。否则，编译器可能先找到同名的另一个类，并报告出错消息。若怀疑自己碰到了类路径问题，请试试在类路径的每一个起点，搜索一下同名的.class 文件。(24) 在 Java 1.1 AWT 中使用事件“适配器”时，特别容易碰到一个陷阱。若覆盖了某个适配器方法，同时拼写方法没有特别讲究，最后的结果就是新添加一个方法，而不是覆盖现成方法。然而，由于这样做是完全合法的，所以不会从编译器或运行期系统获得任何出错提示——只不过代码的工作就变得不正常了。(25) 用合理的设计方案消除“伪功能”。也就是说，假若只需要创建类的一个对象，就不要提前限制自己使用应用程序，并加上一条“只生成其中一个”注释。请考虑将其封装成一个“独生子”的形式。若在主程序里有大量散乱的代码，用于创建自己的对象，请考虑采纳一种创造性的方案，将些代码封装起来。(26) 警惕“分析瘫痪”。请记住，无论如何都要提前了解整个项目的状况，再去考察其中的细节。由于把握了全局，可快速认识自己未知的一些因素，防止在考察细节的时候陷入“死逻辑”中。(27) 警惕“过早优化”。首先让它运行起来，再考虑变得更快——但只有在自己必须这样做、而且经证实在某部分代码中的确存在一个性能瓶颈的时候，才应进行优化。除非用专门的工具分析瓶颈，否则很有可能是在浪费自己的时间。性能提升的隐含代价是自己的代码变得难于理解，而且难于维护。(28) 请记住，阅读代码的时间比写代码的时间多得多。思路清晰的设计可获得易于理解的程序，但注释、细致的解释以及一些示例往往具有不可估量的价值。无论对你自己，还是对后来的人，它们都是相当重要的。如对此仍有怀疑，那么请试想自己试图从联机 Java 文档里找出有用信息时碰到的挫折，这样或许能将你说服。

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++){

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

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。

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。

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

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。

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。

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%>第一页<a href="List.jsp?page=<%=upPage%>">上一页 <% for(int j=1;j<=intPageCount;j++){ if(currentPage!=j){

%> <a href="list.jsp?page=<%=j%>">[<%=j%>] <% }else{ out.println(j); } } %>

<a href="List.jsp?page=<%=nextPage%>">下一页<a href="List.jsp?page=<%=intPageCount%>">最后页

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；

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

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

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; publ ic InsertSort(int num,int mod) { al = new ArrayList(num); Random rand = new Random(); Syste m.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()>=((Intege r)al.get(MaxSize-1)).intValue()) { al.add(MaxSize,tempInt); MaxSize++; System.out.println(al.toStrin g()); } 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.pr intln("The ArrayList Sort After:"); for(int i=0;i<al.size();i++) { System.out.println("al["+i+"]="+al.g et(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.S erializable 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 A BC"而不是"我 ABC+汉的半个"。

答：代码如下： package test; class SplitString { String SplitStr; int SplitByte; public SplitString(Stri ng str,int bytes) { SplitStr=str; SplitByte=bytes; System.out.println("The String is:'"+SplitStr+"';Spl itBytes="+SplitByte); } public void SplitIt() { int loopCount; loopCount=(SplitStr.length()%SplitByt e==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(); } }

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 { } }

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.executeQu
ery("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。

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.createEle
ment("老师");
org.w3c.dom.Element wang = doc.createElement("王"); org.w3c.dom.Element liu = doc.createElem
ent("刘");
wang.appendChild(doc.createTextNode("我是王老师")); root.appendChild(wang); doc.appendChild(ro
ot);
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" en
coding="gb2312"?>
<person> <name>王小明</name> <college>信息学院</college> <telephone>6258113</telephon
e> <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 extend
s HandlerBase {
private Hashtable table = new Hashtable(); private String currentElement = null; private String c
urrentValue = null; public void setTable(Hashtable table) { this.table = table; } public Hashtable
getTable() { return table; }
public void startElement(String tag, AttributeList attrs) throws SAXException { currentElement = t
ag; }
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 (curre
ntElement.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> </HE
AD> <BODY>
<%@ page errorPage="ErrPage.jsp" contentType="text/html;charset=GB2312" %> <%@ page i
mport="java.io.*" %>
<%@ page import="java.util.Hashtable" %> <%@ page import="org.w3c.dom.*" %> <%@ pag

```
e import="org.xml.sax.*" %>
<%@ page import="javax.xml.parsers.SAXParserFactory" %> <%@ page import="javax.xml.parse
rs.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.newS
AXParser();
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</CAPTI
ON>"); 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(ne
w String("college"))+"</TD></TR>"); out.println("<TR><TD>电话</TD>" + "<TD>" + (String)ha
shTable.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>"); %> </BO
DY> </HTML>
```

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 }

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。

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。

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).val
ue;
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 Obj
ect 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

JAVA 华为面试题 JAVA 方面
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 LinkedLi
st 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(de
c);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 implem
ents Runnable{
public void run(){for(int i=0;i<100;i++){dec();}}}}
```

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。

你知道 Object 类中有那些方法？（随意说几个即可）

1`把一个对象作为一个方法的参数，在该方法中对该对象的属性做了修改，请问在外部调用该方法后，传入方法的对象是否发生了变化？例如：假设 stephen 是一个类 int change value(Stephen Stephen) Stephen a=Spublic static void main(stringc( ) args)Stephen Stephen=new Stephen( )Int; P=change

value (Stephen);请问对象 stephen 变化了吗？

许天岭面试题

在 Jdbc 进行数据库调用时，你经常采用什么方式执行 sql 语句？为什么不用其他方式（比较一下即可）

int 类型在 java 中有多少位？（如果面试题目中有这样的问题，不是公司太牛就是公司太差）

你用过线程吗？请启动 4 个线程对一个静态变量进行加 1 操作。

线程是如何启动的？

每个类实力化时都调用父类的构造函数吗？如果是，那么都调用 object 类的构造函数吗？

你懂得 Ftp 协议吗？如果不懂请问我告诉你 Ftp 协议命令格式及数据包的解析方法，你能用多长时间用 java 基本 apI 搞定一个 ftp 客户端程序（是这样的问题主要看你个人学习能力，一般也就是一人五天的工作量，不必要害怕，一般他不会给你五天做的，就是想看一下你的自信心及对工作的理解能力）

你知道 java 与 C 的通信?吗你会用那些协议进行通信？（其实也就是问 socret 通信）

请问 java 中的网络通信有那些方式，有什么区别？

String a="”For limit I=0;I<100000;I++)A=a+"A”把字符串成"A”连接 100000 次，上面方法不够好，请优化上面代码？（采用 stringBuffer 进行优化）

EJB 的调用过程，请叙述一下。

对于 EJB 的面试，业界基本上是假的，咱们学的东西是够应付，但如果你能说的很有条理，你的档次就高了

如果遇到英文试题，也就是平时经常见的调试信息不用害怕

你在 jsp 中打印是如何实现的？还要说你用系统的打印方法，也就是说，在 JSP 中若有很多内容，而我只需要打印其中一个表格，是如何实现的？

你用 java script 做过树型菜单吗？（这样的问题你应该说没有做国，但是会用，当然你要是真做过也很好，那么将来你的就是做 JSP 界面的高手）

WEB 服务器启动时，系统需要做一些初始化的工作，这些工作该怎么处理，在 structs 下又该怎样处理（不要只会用 structs，而忘记了传统方式，外面还有很多项目没有人会用 structs）

对 structs，相信大家都很熟悉，但不要忘记传统的开发模式。

你写过 tag 吗？

你做过在 jsp 页面上下载一个文本文件吗？请描述你的方法？

你在数据库编程过程中，面临的数据量有多大？如果有一个项目中每天有三张结构完全相同的表，一个 365 天天天如此，每张表记录在 100 万条以上，现需要分页查询，根据这样的项目，采用你用过的分页原理，行吗？（这是考的是性能，一般的分页方式不行，遇到这样的题，你可以说，你需要了解更详细的业务，认真的研究一下，是可以的，当然，如果你认为你的方法可以，可以对这样的问题进行交流，等等。这样的题，说不好也行，不影响你的面试，主要是看一下你对问题的态度）

你用 java 调用过的存储过程吗？当一个存储过程有返回记录集时，该怎样在 java 中返回？

应该对 oracle 有所了解，对一些数据库的名词，应该知道词的解释。

分页一 前提 希望最新的纪录在开头给你的表建立查询： 表：mytable

查询：create or replace view as mytable_view from mytable order by id desc 其中，最好使用序列号 create sequence mytable_sequence 来自动增加你的纪录 id 号 二 源程序 <%String sConn="你的连接"

Class.forName("oracle.jdbc.driver.OracleDriver"); Connection conn=DriverManager.getConnection(sConn,"你的用户名","密码");

Statement stmt=conn.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,ResultSet.CONCUR_UPDATABLE);

Statement stmtcount=conn.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,ResultSet.CONCUR_UPDATABLE);

ResultSet rs=stmt.executeQuery("select * from mytable_view"); String sqlcount="select count(*) from mytable_view"; ResultSet rscount=stmtcount.executeQuery(sqlcount); int pageSize=你的每页显示纪录数； int rowCount=0; //总的记录数 while (rscount int pageCount; //总的页数 int currPage; //当前页数

String strPage; strPage=request.getParameter("page"); if (strPage==null){ currPage=1; } else {

currPage=Integer.parseInt(strPage); if (currPage<1) currPage=1; } pageCount=(rowCount+pageSize-1)/pageSize; if (currPage>pageCount) currPage=pageCount; int thepage=(currPage-1)*pageSize;

int n=0; rs.absolute(thepage+1); while (n<(pageSize)&&!rs %> <%rs.close(); rscount.close(); stmt.close(); stmtcount.close(); conn.close(); %> //下面是 第几页等

<form name="sinfo" method="post" action="sbinfo_index.jsp?condition=<%=condition%>&type=<%=type%>" onSubmit="return testform(this)"> 第<%=currPage%>页 共<%=pageCount%>页 共<%=rowCount%>条

<%if(currPage>1){%><a href="sbinfo_index.jsp?condition=<%=condition%>&type=<%=type%>">首页<%}%>

<%if(currPage>1){%><a href="sbinfo_index.jsp?page=<%=currPage-1%>&condition=<%=condition%>&type=<%=type%>">上一页<%}%> <%if(currPage<pageCount){%><a href="sbinfo_index.jsp?page=<%=currPage+1%>&condition=<%=condition%>&type=<%=type%>">下一页<%}%>

<%if(pageCount>1){%><a href="sbinfo_index.jsp?page=<%=pageCount%>&condition=<%=condition%>&type=<%=type%>">尾页<%}%> 跳到<input type="text" name="page" size="4" style="font-size:9px">页

<input type="submit" name="submit" size="4" value="GO" style="font-size:9px"> </form> 希望大家喜欢！！！！！！

托普集团程序员面试试一、选择题(每题 1 分，共 20 分)1. 下列那种语言是面向对象的（C）

A. C B. PASCAL C. C++ D. FORTRAN77

2. 在 Windows9x 下，可以进入 MS-D0S 方式。当在 DOS 提示符下键入（ B ）命令后，系统将退出

MS-DOS 方式，返回到 WIndows 方式。 A. CLOSE B. EXIT C. QUIT D. RETURN

3．下面哪些是面向对象的基本特性：(ABC)A 多态 B 继承 C 封装 D 接口

4．在 C++中经常要进行异常处理，下面哪些是异常处理常用到的关键词：(ABC)

A try B catch C throw D break E contiue

5．数据库技术中的"脏数据',是指（C）的数据。A.错误 B.回返 C.未提交 D.未提交的随后又被撤消

6．TCP/IP 是一种（A,B）A.标准 B.协议 C.语言 D.算法

7． 下面有关计算机操作系统的叙述中，不正确的是(B) A 操作系统属于系统软件 B 操作系统只负责管理内存储器，而不管理外存储器 C UNIX 是一种操作系统 D 计算机的处理器、内存等硬件资源也由操作系统管理

8．微机上操作系统的作用是(D) A 解释执行源程序 B 编译源程序

C 进行编码转换 D 控制和管理系统资源

9．下列存储器中存取速度最快的是(A) A 内存 B 硬盘 C 光盘 D 软盘

10．在计算机中，—个字节是由多少个二进制位组成的(B) A. 4 B. 8 C. 16 D. 24

11. 存储 16×16 点阵的一个汉字信息，需要的字节数为(A)A 32 B 64 C 128 D 256

12. 以下选项中合法的字符常量是（BC）A."B" B. '\010' C. 68 D. D

13. 假定 x 和 y 为 double 型，则表达式 x=2,y=x+3/2 的值是（D）A. 3.500000 B. 3 C. 2.000000 D. 3.000000

14. 以下合法的赋值语句是（BCD）//In C++ ,choice D also is correct, but in C language, D is wrong.

A. x=y=100 B. d--; C. x+y; D. c=int(a+b);

15. 设正 x、y 均为整型变量，且 x=10 y=3，则以下语句 pprintf("%d,%d\n",x--,--y); 的输出结果是（D）

A.10,3 B. 9,3 C. 9,2 D.10,2

16. x、y、z 被定义为 int 型变量，若从键盘给 x、y、z 输入数据，正确的输入语句是（B）

A .INPUT x、y、z; B. scanf("%d%d%d",&x,&y,&z);C. scanf("%d%d%d",x,y,z); D. read("%d%d%d",&x,&y,&z);

17.以下数组定义中不正确的是（D）A) int a[2][3]; B) int b[][3]={0,1,2,3};C) int c[100][100]={0}; D) int d[3][]={{1,2},{1,2,3},{1,2,3,4}};

18. 以下程序的输出结果是（A）main(){ int a[4][4]={{1,3,5},{2,4,6},{3,5,7}};

printf("%d%d%d%d\n",a[0][3],a[1][2],a[2][1],a[3][0];

}A) 0650 B) 1470 C) 5430 D) 输出值不定

19 以下程序的输出结果是（B）main(){char st[20]= "hello\0\t\\\";printf(%d %d \n",strlen(st),sizeof(st));

}A) 9 9 B) 5 20 C) 13 20 D) 20 20

20. 当调用 Windows API 函数 InvalidateRect,将会产生什么消息（A）A:WM_PAINT B:WM_CREATE C:WM_NCHITTEST D:WM_SETFOCUS

二、填空题(每题 3 分，共 30 分)

1．请列举当前一些当前流行的数据库引擎,SQL SERVER,ORACLE,BDE,Microsoft Jet。

2． 为了将当前盘当前目录中的所有文本文件（扩展名为.TXT）的内容打印输出，正确的单条 DOS 命令为 COPY *.TXT PRN。

3． 计算机网络分为局域网和广域网，因特网属于广域网。

4. 设 y 是 int 型变量，请写出判断 y 为奇效的关系表达 y%2！=0。

5. 设有以下程序:main(){ int n1,n2;scanf("%d",&n2);while(n2!=0){ n1=n2%10;n2=n2/10;printf("%d",n1);}}

程序运行后，如果从键盘上输入 1298；则输出结果为 8921。

6．以下程序运行后的输出结果是:9876 876

main(){ char s[ ]="9876",*p;for (p=s ; p<s+2 ; p++) printf("%s\n", p);}

7．以下函数的功能是：求 x 的 y 次方，请填空。double fun(double x, int y){ int i;double z;for(i=1, z=x; i<y;i++) z=z* x ;return z;}

8．以下程序段打开文件后，先利用 fseek 函数将文件位置指针定位在文件末尾，然后调用 ftell 函数返回当前文件位置指针的具体位置，从而确定文件长度，请填空。FILE *myf; long f1;myf= fopen ("test.

t","rb");

fseek(myf,0,SEEK_END); f1=ftell(myf);fclose(myf);printf("%d\n",f1);

9. 以下程序输出的最后一个值是 120。

int ff(int n){ static int f=l;f=f*n;return f;}main(){ int i;for(I=1;I<=5;I++ printf("%d\n",ff(i));)

10. 以下程序运行后的输出结果是 52 main(){ int i=10, j=0;do{ j=j+i; i--;while(i>2);printf("%d\n", j);}

三、判断题(每题 2 分，共 20 分)

1：动态链结库不能静态调用。 错误

2：UDP 是面向无连接的网络连接 正确

3：ASP 是一种数据库引擎 错误

4：队列是先进后出。 错误

5：Weblogic 是分布式应用服务器。 正确

6：TCP,UDP 都是传输层的协议。 正确

7: 两个线程不能共存于同一地址空间 错误

8: JAVA 是一种跨平台的开发工具 正确

9. 在 WINDOWS 操作系统中对外设是以文件的方式进行管理 正确

10. 虚拟内存实际是创建在硬盘上的 正确

四、问答题(每题 10 分，共 30 分)

1. 写出从数据库表 Custom 中查询 No、Name、Num1、Num2 并将 Name 以姓名显示、计算出的和以总和显示的 SQL。SELECT No , Name AS '姓名' , Num1 , Num2, （Num1+Num2） AS '总和' FROM Custom

何为"事务处理"，谈谈你对它的理解。事务处理是指一个单元的工作，这些工作要么全做，要么全部不做。作为一个逻辑单元，必须具备四个属性：自动性、一致性、独立性和持久性。自动性是指事务必须是一

个自动的单元工作，要么执行全部数据的修改，要么全部数据的修改都不执行。一致性是指当事务完成时，必须使所有数据都具有一致的状态。在关系型数据库中，所有的规则必须应用到事务的修改上，以便维护所有数据的完整性。所有的内部数据结构，在事务结束之后，必须保证正确。独立性是指并行事务的修改必须与其他并行事务的修改相互独立。一个事务看到的数据要么是另外一个事务修改这些事务之前的状态，要么是第二个事务已经修改完成的数据，但是这个事务不能看到正在修改的数据。

3. 常用的数据结构有哪些？请枚举一些。（不少于 5 个）链表、堆栈、二叉树、队列、图、堆，集合。

4. 什么是 OOP？什么是类？请对比类和对象实例之间的关系。OOP 是 Object_oriented Programming(面向对象编程)的缩写。这主要是为了区别于以前的面向过程的程序设计！指的是用对象的观点来组织与构建系统，它综合了功能抽象和数据抽象，这样可以减少数据之间的耦合性和代码的出错几率。使用面向对象编程技术可以使得软件开发者按照现实世界里人们思考问题的模式编写代码,可以让软件开发者更好地利用代码直接表达现实中存在的对象,将问题空间直接映射到解空间!类：即 class 在面向对象的程序设计中，专门用"类"来表示用户定义的抽象数据类型（user_defined abstract type）。它将具有相同状态、操作和访问机制的多个对象进行了抽象。类具有继承、数据隐藏和多态三种主要特性。利用类的这三种特性可以更好地表示现实世界中事物。类是同一类对象实例的共性的抽象，对象是类的实例化。对象通常作为计算机模拟思维，表示真实世界的抽象，一个对象就像一个软件模块，可以为用户提供一系列的服务---可以改变对象的状态、测试、传递消息等。类定义了对象的实现细节或数据结构。类是静态的，对象是动态的，对象可以看作是运行中的类。类负责产生对象，可以将类当成生产对象的工厂（Object factory）.

5. 有一组数字（3，10，6，8，98，22），请编程排序（升降序皆可），语言不限，算法不限，但须注明是何种算法。//下面使用简单的冒泡法进行排序!

```
#include "iostream.h"  template<class type>  class CBubble{
private: type *pArray; int size;public:CBubble(type a[],int sizeArray);void sort();void display();};
template <class type> CBubble<type>::CBubble(type a[],int sizeArray)
```

```
{ pArray=a; size=sizeArray/sizeof(type);}
template<class type>void CBubble<type>::sort(){  type temp;  for(int i=0;i<size-1;i++) for(int j=
0;j<size-1-i;j++) if(pArray[j]>pArray[j+1])//升序{temp=pArray[j+1];pArray[j+1]=pArray[j];pArray[j]
=temp;}}
template<class type>void CBubble<type>::display(){for(int i=0;i<size;i++)cout<<pArray[i]<<endl;}
void main(void){int a[]={3,10,6,8,98,22};CBubble<int> intData(a,sizeof(a));cout<<"The original da
ta are :"<<endl;intData.display();intData.sort();cout<<"After sorting ,the data are:"<<endl;intData.
display();
}
SQLhttp://www.jactiongroup.net/reference/html/index.html  //书
http://blog.csdn.net/hbuzhang/archive/2004/12/07/207202.aspx  //书
connection  connconn.setAuto(false)//表示手动提交 conn.commit//  提交 conn.rollback();//事务回滚
-内联接 use pubsselect a.au_fname, a.au_lname, p.pub_name  from authors a inner join publisher
s p on a.city = p.city order by p.pub_name asc, a.au_lname asc,  a.au_fname asc
--左外联接 use pubs  select a.au_fname, a.au_lname, p.pub_name  from authors a left join publis
hers p
on a.city = p.city  order by p.pub_name asc,  a.au_lname asc,  a.au_fname asc
-使用子查询 USE pubs  GO  SELECT distinct pub_name  FROM publishers  WHERE pub_id IN  (SEL
ECT pub_idFROM titlesWHERE type = 'business')  GO
--如果平均价格少于 $30，WHILE 循环就将价格加倍，然后选择最高价。
--如果最高价少于或等于 $50，WHILE 循环重新启动并再次将价格加倍。
--该循环不断地将价格加倍直到最高价格超过 $50  USE pubs  GO
WHILE (SELECT AVG(price) FROM titles) < $30
BEGIN
   UPDATE titles
```

```
    SET price = price * 2
  SELECT MAX(price) FROM titles
  IF (SELECT MAX(price) FROM titles) > $50
    BREAK
  ELSE
    CONTINUE
END
---如果平均价格少于 $30，WHILE 循环就将价格加倍，然后选择最高价。
--如果最高价少于或等于 $50，WHILE 循环重新启动并再次将价格加倍。
--该循环不断地将价格加倍直到最高价格超过 $50
USE pubs
GO
WHILE (SELECT AVG(price) FROM titles) < $30
BEGIN
  UPDATE titles
    SET price = price * 2
  SELECT MAX(price) FROM titles
  IF (SELECT MAX(price) FROM titles) > $50
    BREAK
  ELSE
    CONTINUE
END
CREATE PROCEDURE au_info
  @lastname varchar(40),
  @firstname varchar(20)
```

```
AS
SELECT au_lname, au_fname, title, pub_name
  FROM authors a INNER JOIN titleauthor ta
    ON a.au_id = ta.au_id INNER JOIN titles t
    ON t.title_id = ta.title_id INNER JOIN publishers p
    ON t.pub_id = p.pub_id
  WHERE  au_fname = @firstname
    AND au_lname = @lastname
GO
EXECUTE au_info 'Dull', 'Ann'--或者
EXECUTE au_info @lastname = 'Dull', @firstname = 'Ann'--创建存储过程CREATE PROCEDURE title
s_sum @TITLE varchar(40),@SUM money OUTPUT
AS
SELECT @SUM = SUM(price)
FROM titles
WHERE title LIKE @TITLE
GO
DECLARE @TOTALCOST money
EXECUTE titles_sum 'The%', @TOTALCOST OUTPUT
select @TOTALCOST
go
CREATE PROCEDURE Oakland_authors
AS
SELECT au_fname, au_lname, address, city, zip
FROM authors
```

```
WHERE city = 'Oakland'
and state = 'CA'
ORDER BY au_lname, au_fname
GO
--sp_helptext Oakland_authors
ALTER PROCEDURE Oakland_authors
AS
SELECT au_fname, au_lname, address, city, zip
FROM authors
WHERE state = 'CA'
ORDER BY au_lname, au_fname
GO
--sp_helptext Oakland_authors
--提交事务后，所有书籍支付的版税增加 10%。
begin transaction MyTransaction
update roysched
set royalty = royalty * 1.10
commit transaction MyTransaction
--rollback transaction MyTransaction
select royalty from roysched
--select @@trancount
--1.创建试验实验表
create table temptrigger
( id_temp varchar(2) not null primary key,
  temp_name varchar(10) null,
```

```
  temp_age  int  null)go
insert  temptrigger  values('01','张三','10')
insert  temptrigger  values('02','李四','11')
insert  temptrigger  values('03','王五','12')
insert  temptrigger  values('04','赵六','11')
select  *  from  temptrigger   go
--2.创建 insert , update 触发器
create  trigger  temptrigger_modify
on  temptrigger
for  insert,update
as
begin
  if  (select  temp_age  from  inserted)  >  15
    begin
      rollback  transaction
      print  '年龄不能超过 15 岁！'
    end
end
--insert  temptrigger  values('04','大朋','17')
--insert  temptrigger  values('05','大朋','17')
--insert  temptrigger  values('05','大朋','14')
--update  temptrigger  set  temp_age='18'  where  id_temp  =  '01'
--update  temptrigger  set  temp_age='9'  where  id_temp  =  '01'
-3.创建 delete  触发器--drop  trigger  temptrigger_delete
create  trigger  temptrigger_delete
```

```
on temptrigger
for delete
as
begin
  print @@rowcount
  if @@rowcount > 1
  begin
    rollback transaction
    print '一次删除记录不能多于 1 条'
  end
end
--delete from temptrigger
--delete from temptrigger where id_temp='01'
--创建聚集索引 create clustered index clindx_titleid on roysched(title_id)--sp_help roysched
--创建非聚集索引 create nonclustered index unclindx_titleid on roysched(title_id)--sp_help roysched
--查看索引统计 dbcc show_statistics(roysched,titleidind)
--更新索引统计 update statistics authors
--重建索引 dbcc dbreindex('roysched',unclindx_titleid)
--删除索引 drop index roysched.unclindx_titleid-sp_help roysched
1--创建 ssn(社会保险号)的基于 varchar 的自定义数据类型。
--用于存储 11 位社会保险号（999-99-999）的列。该列不能
--为 null。use pubs exec sp_addtype ssn , 'varchar(11)' , 'NOT NULL'
--查看创建的数据类型--sp_help ssn
--使用创建的数据类型 create table mytable( myid varchar(2) primary key, myssn ssn)
4-删除创建的数据类型--drop table mytable--exec sp_droptype ssn
```

•批是包含一个或多个 Transact-SQL 语句的组，从应用程序一次性地发送到 Microsoft SQL Server 执行。批作为一个整体执行，以 GO 命令结束。批处理是客户端作为一个单元发出的一个或多个 SQL 语句的集合。每个批处理编译为一个执行计划。

触发器•触发器是在对表进行插入、更新或删除操作时自动执行的存储过程•触发器通常用于强制业务规则•触发器可以确保数据的完整性和一致性

事务是用户定义的一个操作序列，这些操作要么全做要么全不做，是一个不可分割的工作单位(构成单一逻辑工作单元的操作集合)如果某一事务成功，则在该事务中进行的所有数据更改均会提交，成为数据库中的永久组成部分。

如果事务遇到错误且必须取消或回滚，则所有数据更改均被清除

•锁 是在多用户环境中对数据访问的限制封锁就是事务 T 在对某个数据对象（如表、记录等）操作之前，先向系统发出请求，对其加锁。加锁后事务 T 就对该数据对象有了一定的控制，在事务 T 释放它的锁之前，其它的事务不能更新此数据对象。（锁蕴含的基本概念是用户需要对表的排它访问）•从程序员的角度看：分为乐观锁和悲观锁。乐观锁：完全依靠数据库来管理锁的工作。悲观锁：程序员自己管理数据或对象上的锁处理。

子查询：一个 SELECT 语句嵌套在另一个 SELECT 语句中。

—索引—是一个数据库对象，它是某个表中一列或若干列值的集合和相应的指向表中物理标识这些值的数据页的逻辑指针清单,然后根据指定的排序次序排列这些指针 —优点提高查询执行的速度。 强制实施数据的唯一性。 提高表之间联接的速度。 缺点 存储索引要占用磁盘空间。数据修改需要更长的时间，因为索引也要更新。

•视图•是一种虚拟表，通常是作为来自一个或多个表 的行或列的子集创建的。•视图本质上讲，就是保存在数据库中的 select 查询•视图并不是数据库中存储的数据值的集合。•对最终用户的好处– 结果更容易理解– 获得数据更容易

•对开发人员的好处– 限制数据检索更容易– 维护应用程序更方便

存储过程•使用一个名称存储的预编译 T-SQL 语句和流程控制语句的集合•由数据库开发人员或数据库管理员编写

•用来执行管理任务或应用复杂的业务规则　优点•执行速度更快•首次运行时，进行优化和编译得到执行计划并将该计划存储在系统表中，以后直接运行。•实现多个程序共享应用程序逻辑•组件式编程•能够屏蔽数据库的结构，实现更高的安全性

•减少网络流通量

数据库设计和建模必要性•好的数据库结构有利于：-节省数据的存储空间-能够保证数据的完整性-方便进行数据库应用系统的开发•设计不好的数据库结构将导致-数据冗余、存储空间浪费-内存空间浪费

不管数据库的大小和复杂程度如何，可以用下列基本步骤来设计数据库：–收集信息–标识对象–设计数据模型–标识每个对象 存储的信息类型–标识对象之间的关系

•数据模型是一种标识实体类型及其实体间联系的模型。典型的数据模型由网状模型、层次模型和关系模型。什么是规范化从关系数据库的表中，除去冗余数据的过程称为规范化。—精简数据库的结构—从表中删除冗余的列—标识所有依赖于其它数据的数据

三级范式第一范式的定义：如果一个表中没有重复组（即行与列的交叉点上只有一个值，而不是一组值），则这个表属于第一范式（常记成 1NF）。简而言之："每一字段只存储一个值"。例如:职工号，姓名，电话号码组成一个表（一个人可能有一个办公室电话 和一个家里电话号码） 第二范式的定义：如果一个表属于 1NF，任何属性只依赖于关键字，则这个表属于第二范式（常记成 2NF ）。简而言之：必须先符合 1NF 的条件，且每一行都能被唯一的识别。 将 1NF 转换成 2NF 的方法是添加主键。学号,课程名,成绩第三范式的定义：如果一个表属于 2NF，且不包含传递依赖性，则这个表是第三范式（常记成 3NF）。满足 3NF 的表中不包含传递依赖。简而言之：没有一个非关键属性依赖于另一个非关键属性。学号，课程号，成绩，学分学号，姓名，所在系，系名称，系地址

什么是类与对象？

所谓对象就是真实世界中的实体，对象与实体是一一对应的，也就是说现实世界中每一个实体都是一个对象，它是一种具体的概念。

类是具备某些共同特征的实体的集合，它是一种抽象的概念，用程序设计的语言来说，类是一种抽象的数据类型，它是对所具有相同特征实体的抽象。

属性与方法？

不同对象具有相同特点，就可能抽象为一定的类，那么这些特点基本上可以分为两类，一类是描述对象静态状态的，就是对象的属性，在程序设计中，可以称之为变量；另一类是描述对象的动作，就是对象的方法，在程序设计中我们称之为函数。属性和方法是一个对象所具备的两大基本要素，也是我们后面编程工作的核心。

什么是封装？

只要有足够的方法，就没必要直接去操作对象属性，只要调用这些方法就可以实现要完成的任务，这种现象称为封装，它通过对象方法对其属性的操作把对象属性封装在一个对象内部，对象与外界打交道全部通过其自身的方法来实现，有效的把对象属性隐藏在对象内部。

编写 java 文件的注意事项？

在记事本中编写 java 文件，在保存时一定要把文件名和扩展名用双引号括起来，否则将默认保存为文本文件，如果要保存的 java 文件名为 Program1.java,则在保存时在文件名文本框中一定要输入”Program1.java”。

如何编译 java 程序？

单击开始|运行命令，在命令行上输入 cmd，按回车键（在 window98 中输入 command，按回车键），即可打开一个命令窗口，将目录转换到编写 java 源程序所在的目录，输入 javac filename.java

如何执行 java 程序？

同样在命令窗口中输入 java filename，

基本数据类型？

Java 的数据类型可以划分为 4 大类：整数，浮点数，字符型，布尔型。其中整数可以划分为：byte,short,int,long.浮点数可以划分为 float,double.