**JAVA面试、笔试题**

**欲想成功，必先自宫！**

目录

[一、 HTML&CSS部分 9](#_Toc427939102)

[1、HTML中定义表格的宽度用80px和80%的区别是什么？ 9](#_Toc427939103)

[2、CSS样式定义优先级顺序是？ 10](#_Toc427939104)

[3、div和span的区别？ 10](#_Toc427939105)

[4、CSS选择器包括？ 10](#_Toc427939106)

[5、用css3语法中，如何实现一个矩形框的圆角效果和50%红色透明效果？，请写出关键脚本 10](#_Toc427939107)

[6、Div与Table的区别 11](#_Toc427939108)

[7、行级标签转块级标签，块级标签转行级标签 11](#_Toc427939109)

[二、Java基础部分 11](#_Toc427939110)

[1、java中有哪些基本类型？ 11](#_Toc427939111)

[2、java为什么能够跨平台运行？ 12](#_Toc427939112)

[3、String是基本数据类型吗？我可不可以写个类继承于String？ 12](#_Toc427939113)

[4、谈谈&和&&的区别？ 12](#_Toc427939114)

[5、Switch语句里面的条件可不可以是byte、long、String？使用时候还应注意什么？ 12](#_Toc427939115)

[6、short s1=1;s1=s1+1;有什么错？short s1 = 1;s1+=1 有什么错？ 12](#_Toc427939116)

[7、char为什么能存贮一个汉字？ 13](#_Toc427939117)

[8、用最效率的办法算出2乘以8等于几？ 13](#_Toc427939118)

[9、final修饰变量时，该变量是对象时，对象的值可不可以改变？ 13](#_Toc427939119)

[10、静态变量和实例变量的区别？ 13](#_Toc427939120)

[11、面向对象的基本特征是什么？ 13](#_Toc427939121)

[12、作用域public,private,protected,以及不写时的区别？ 14](#_Toc427939122)

[13、Overload和Override的区别。 14](#_Toc427939123)

[14、构造器可不可以被重载或重写？ 14](#_Toc427939124)

[15、Java中有没有多继承？ 15](#_Toc427939125)

[16、抽象类和接口的区别？ 15](#_Toc427939126)

[17、java中实现多态的机制是什么？ 15](#_Toc427939127)

[18、int和integer的区别？ 15](#_Toc427939128)

[19、String和StringBuffer的区别？StringBuffer和StringBuilder区别？ 16](#_Toc427939129)

[20、String s=new String(“xyz”);创建了几个String Object? 16](#_Toc427939130)

[21、数组中有没有length()方法，String中有没有length（）方法？ 16](#_Toc427939131)

[22、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? 16](#_Toc427939132)

[23、final, finally, finalize的区别。 17](#_Toc427939133)

[24、‘==’和equals的区别？ 17](#_Toc427939134)

[25、error和exception有什么区别？ 17](#_Toc427939135)

[26、heap和stack有什么区别。 18](#_Toc427939136)

[27、GC是什么? 为什么要有GC? 18](#_Toc427939137)

[28、什么是内部类？分为哪几种？ 18](#_Toc427939138)

[29、为什么需要内部类？ 18](#_Toc427939139)

[30、内部类可以引用它的包含类的成员吗？有没有什么限制？ 19](#_Toc427939140)

[31、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)? 20](#_Toc427939141)

[32、使用java命令查看java虚拟机版本 20](#_Toc427939142)

[33、数字转字符有多少种方式，分别是什么 20](#_Toc427939143)

[34、Java创建对象有几种方式 20](#_Toc427939144)

[35、写一个反序列化实例 20](#_Toc427939145)

[三、JavaScript/JQuery/Ajax部分 22](#_Toc427939146)

[1、请写一段Javascript程序，对以下程序的用户输入日期的有效性进行判断，如果格式错误就提示用户。在程序的恰当位置注明怎样调用你写的程序。日期格式为：年年年年月月日日小时，例如2003010812。 22](#_Toc427939147)

[2、Java和Javascript区别在哪? 23](#_Toc427939148)

[3、列举javaScript的3种主要数据类型，2种复合数据类型和2种特殊数据类型。 24](#_Toc427939149)

[4、谈谈你的JS的理解？ 25](#_Toc427939150)

[5、ajax的优点？ 26](#_Toc427939151)

[6、简述一下ajax调试代码查找错误的方法？ 26](#_Toc427939152)

[7、简述ajax中Js脚本缓存问题该如何解决？ 26](#_Toc427939153)

[8、Ajax应用和传统的web应用有何不同？ 26](#_Toc427939154)

[9、javascript的作用？ 27](#_Toc427939155)

[10、为什么要有jquery？ 27](#_Toc427939156)

[11、jQuery选择器有多少种？ 27](#_Toc427939157)

[12、jquery选择器有哪些优势？ 27](#_Toc427939158)

[13、你是如何使用jquery中的ajax的？ 28](#_Toc427939159)

[14、jquery中的$.get和$.post请求区别？ 28](#_Toc427939160)

[15、jquery中如何操作样式的？ 28](#_Toc427939161)

[16、如何设置和获取HTML和文本的值？ 28](#_Toc427939162)

[17、Jquery能做些什么？ 28](#_Toc427939163)

[18、在ajax中data主要有哪几种？ 29](#_Toc427939164)

[19、jQuery中ajax由几部分组成？ 29](#_Toc427939165)

[四、jsp/servlet部分 29](#_Toc427939166)

[1、Tomcat的优化经验 29](#_Toc427939167)

[2、Tomcat根目录下有哪些文件 29](#_Toc427939168)

[3、什么是TOMCAT,怎样启动停止，配置文件，日志文件的存储。 30](#_Toc427939169)

[4、解释一下什么是servlet;什么是servlet容器; 30](#_Toc427939170)

[5、说一说Servlet的生命周期，执行过程? 31](#_Toc427939171)

[6、实例化servlet有几种方式 31](#_Toc427939172)

[7、HTTP请求的GET与POST方式的区别 31](#_Toc427939173)

[8、请写一个Servlet的基本架构。 32](#_Toc427939174)

[9、forward 和redirect的区别? 32](#_Toc427939175)

[10、servlet中怎么定义forward 和redirect 33](#_Toc427939176)

[11、过滤器有哪些作用？ 33](#_Toc427939177)

[12、JSP的常用指令？ 33](#_Toc427939178)

[13、JSP和Servlet中的请求转发分别如何实现？ 33](#_Toc427939179)

[14、JSP乱码如何解决？ 34](#_Toc427939180)

[15、session 和 application的区别？ 34](#_Toc427939181)

[16、jsp有哪些内置对象?作用分别是什么? 34](#_Toc427939182)

[17、Jsp有哪些动作?作用分别是什么? 35](#_Toc427939183)

[18、JSP中动态INCLUDE与静态INCLUDE的区别？ 35](#_Toc427939184)

[19、JSP和Servlet有哪些相同点和不同点，他们之间的联系是什么？ 35](#_Toc427939185)

[20、页面传递对象的方法？ 36](#_Toc427939186)

[21、Cookied和session区别？ 36](#_Toc427939187)

[五、数据库部分 36](#_Toc427939188)

[1、触发器的作用？ 36](#_Toc427939189)

[2、什么是存储过程？用什么来调用？ 37](#_Toc427939190)

[3、存储过程的优缺点？ 37](#_Toc427939191)

[4、存储过程与函数的区别 37](#_Toc427939192)

[5、索引的作用？和它的优点缺点是什么？ 38](#_Toc427939193)

[6、什么样的字段适合建索引 38](#_Toc427939194)

[7、索引类型有哪些？ 38](#_Toc427939195)

[8、什么是事务？什么是锁？ 39](#_Toc427939196)

[9、什么叫视图？游标是什么？ 39](#_Toc427939197)

[10、视图的优缺点 39](#_Toc427939198)

[11、列举几种表连接方式,有什么区别？ 40](#_Toc427939199)

[12、主键和外键的区别？ 40](#_Toc427939200)

[13、在数据库中查询语句速度很慢，如何优化？ 40](#_Toc427939201)

[14、数据库三范式是什么? 41](#_Toc427939202)

[15、union和union all有什么不同? 41](#_Toc427939203)

[16、Varchar2和varchar有什么区别？ 42](#_Toc427939204)

[17、Oracle和Mysql的区别？ 42](#_Toc427939205)

[18、Oracle语句有多少类型 43](#_Toc427939206)

[19、oracle分页语句 43](#_Toc427939207)

[20、从数据库中随机取50条 44](#_Toc427939208)

[21、order by与group by的区别 44](#_Toc427939209)

[22、commit在哪里会运用 44](#_Toc427939210)

[23、行转列、列换行怎么转 44](#_Toc427939211)

[24、什么是PL/SQL？ 44](#_Toc427939212)

[25、序列的作用 45](#_Toc427939213)

[26、表和视图的关系 45](#_Toc427939214)

[27、oracle基本数据类型 45](#_Toc427939215)

[28、truncate与 delete区别 45](#_Toc427939216)

[29、oracle获取系统时间 46](#_Toc427939217)

[30、oracle怎么去除去重 46](#_Toc427939218)

[31、现在有表: 46](#_Toc427939219)

[32、现有表： 47](#_Toc427939220)

[33、怎样把这样一个表（表名：tmp\_table\_201307） 48](#_Toc427939221)

[34、数据库中有一张表ismg\_icp\_flow，结构如下 49](#_Toc427939222)

[六、Java高级部分 50](#_Toc427939223)

[1、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？ 50](#_Toc427939224)

[2、sleep() 和 wait() 有什么区别? 50](#_Toc427939225)

[3、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? 51](#_Toc427939226)

[4、线程的基本概念 51](#_Toc427939227)

[5、什么是多线程 51](#_Toc427939228)

[6、程序、进程、线程之间的关系 51](#_Toc427939229)

[7、创建线程有几种方式，分别是什么？ 52](#_Toc427939230)

[8、线程的生命周期 53](#_Toc427939231)

[9、线程currentThread()与interrupt()方法的使用 53](#_Toc427939232)

[10、线程状态 53](#_Toc427939233)

[11、什么是java序列化，如何实现java序列化？ 54](#_Toc427939234)

[12、编写一个程序，将d:\java目录下的所有.java文件复制到d:\jad目录下，并将原来文件的扩展名从.java改为.jad。 54](#_Toc427939235)

[13、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？ 57](#_Toc427939236)

[14、字节流与字符流的区别 57](#_Toc427939237)

[15、怎么判断指定路径是否为目录 59](#_Toc427939238)

[16、怎么获取指定路径下的全部文件 59](#_Toc427939239)

[17、Java怎么读取文件和写入文件 60](#_Toc427939240)

[18、java怎么复制文件 62](#_Toc427939241)

[19、用JDBC如何调用存储过程 63](#_Toc427939242)

[20、JDBC中的PreparedStatement相比Statement的好处 65](#_Toc427939243)

[21、写一个用jdbc连接实例。 66](#_Toc427939244)

[22、ArrayList和Vector的区别？ 67](#_Toc427939245)

[23、List、Set和Map的区别？ 68](#_Toc427939246)

[24、Collection 和 Collections的区别。 69](#_Toc427939247)

[25、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? 69](#_Toc427939248)

[26、HashMap与HashTable的区别 69](#_Toc427939249)

[27、Java中有多少种数据结构，分别是什么？ 70](#_Toc427939250)

[28、Arraylist 和linklist 的区别 70](#_Toc427939251)

[29、List遍历方式有多少种 70](#_Toc427939252)

[30、Map怎么遍历 71](#_Toc427939253)

[31、怎么获取Map所有的key，所有的value 71](#_Toc427939254)

[32、获取Class的实例有几种方式 71](#_Toc427939255)

[33、怎么获取类中所有的方法，所有属性 71](#_Toc427939256)

[七、框架部分 71](#_Toc427939257)

[1、谈谈你对Struts2的理解。 71](#_Toc427939258)

[2、谈谈你对Hibernate的理解。 72](#_Toc427939259)

[3、你对Spring的理解。 73](#_Toc427939260)

[4、Struts2优缺点 73](#_Toc427939261)

[5、说说struts1与struts2的区别。 75](#_Toc427939262)

[6、struts的核心组件有哪些？ 76](#_Toc427939263)

[7、Strus2的执行过程 76](#_Toc427939264)

[8、为什么要使用struts2？ 76](#_Toc427939265)

[9、openSession和getCurrentSession 77](#_Toc427939266)

[10、拦截器的作用？拦截器和过滤器的区别？ 77](#_Toc427939267)

[11、struts.xml中result的type有哪些类型？ 78](#_Toc427939268)

[12、一般情况下，关系数据模型与对象模型之间有哪些匹配关系? 78](#_Toc427939269)

[13、hibernate 数据的三个状态 78](#_Toc427939270)

[14、Hibernate中load和get的区别？ 79](#_Toc427939271)

[15、Hibernate的工作原理? 79](#_Toc427939272)

[16、hibernate优缺点？ 79](#_Toc427939273)

[17、Hibernate是如何延迟加载的？ 80](#_Toc427939274)

[18、如果优化Hibernate？ 80](#_Toc427939275)

[19、什么是ORM？ 81](#_Toc427939276)

[20、Hibernate的主键生成策略？ 81](#_Toc427939277)

[21、Hibernate的级联操作 81](#_Toc427939278)

[22、Hibernate有哪5个核心接口？ 82](#_Toc427939279)

[23、什么是重量级？什么是轻量级？ 82](#_Toc427939280)

[24、谈谈Spring的IOC和DI 82](#_Toc427939281)

[25、什么是AOP？ 84](#_Toc427939282)

[26、Spring的通知类型有哪些？ 84](#_Toc427939283)

[27、什么是MVC？ 84](#_Toc427939284)

[28、hibernate查询方式有多少种？ 85](#_Toc427939285)

[29、spring中Bean的scope 85](#_Toc427939286)

[30、SSH对应MVC的哪些层 86](#_Toc427939287)

[31、spring注入方式有几种 86](#_Toc427939288)

[八、XML部分 86](#_Toc427939289)

[1、xml有哪些解析技术?区别是什么? 86](#_Toc427939290)

[2、你在项目中用到了xml技术的哪些方面?如何实现的? 87](#_Toc427939291)

[3、编程用JAVA解析XML的方式. 87](#_Toc427939292)

[4、XML文档定义有几种形式？它们之间有何本质区别？ 90](#_Toc427939293)

[5、XML和HTML的区别？ 90](#_Toc427939294)

[6、XML文件和普通文件相比有哪些优点？ 90](#_Toc427939295)

[九、设计模式部分 91](#_Toc427939296)

**[请写出你所知道的设计模式？](#_Toc427939297)** [91](#_Toc427939297)

[十、算法部分 93](#_Toc427939298)

**[1、说明生活中遇到的二叉树，用java实现二叉树](#_Toc427939299)** [93](#_Toc427939299)

[2、第1个人10，第2个比第1个人大2岁，依次递推，请用递归方式计算出第8个人多大？ 100](#_Toc427939300)

[3、排序都有哪几种方法？请列举。用JAVA实现一个快速排序。 101](#_Toc427939301)

[4、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。 103](#_Toc427939302)

[5、从类似如下的文本文件中读取出所有的姓名，并打印出重复的姓名和重复的次数，并按重复次数排序： 104](#_Toc427939303)

[6、写一个Singleton出来。 109](#_Toc427939304)

[7、古典问题：有一对兔子，从出生后第3个月起每个月都生一对兔子，小兔子长到第三个月后每个月又生一对兔子，假如兔子都不死，问每个月的兔子总数为多少？ 112](#_Toc427939305)

[8、简单的说个递归 113](#_Toc427939306)

[9、什么是平衡二叉树 113](#_Toc427939307)

[10、怎么判断二叉树是否有环 114](#_Toc427939308)

# HTML&CSS部分

## 1、HTML中定义表格的宽度用80px和80%的区别是什么？

PX标识像素，%标识整个页面的宽度百分比

## 2、CSS样式定义优先级顺序是？

内联样式最高优先权，然后是内部样式，然后才是外部样式

## 3、div和span的区别？

　　DIV 和 SPAN 元素最大的特点是默认都没有对元素内的对象进行任何格式化渲染。主要用于应用样式表（共同点）。

　　两者最明显的区别在于DIV是块元素，而SPAN是行内元素(也译作内嵌元素)。

　　详解：1.所谓块元素，是以另起一行开始渲染的元素，行内元素则不需另起一行，测试一下下面的代码你会有更形象的理解：

测试<span>紧跟前面的"测试"显示</span><div>这里会另起一行显示</div>

## 4、CSS选择器包括？

1）类别选择器 用“.”来标识

2）标签选择器 用HTML标签来标识

3）ID选择器 用“#”号来标识

4）通配符选择器 用“\*”号来标识

## 5、用css3语法中，如何实现一个矩形框的圆角效果和50%红色透明效果？，请写出关键脚本

<style>

　　div{

width:200px;

height:200px;

border-radius: 30px;

opacity: 0.5;/\* 火狐 \*/

/\* filter:alpha(opacity=50); IE \*/

background-color:red;

}

　　</style>

## 6、Div与Table的区别

1） div大大缩减页面代码，提高页面浏览速度，table代码量繁多，页面浏览效率慢。

2） div结构清晰，可读性非常强，也容易被搜索引擎搜索到，优化了搜索引擎，Table结构复杂，可读性差。

3）div 缩短改版时间。只要简单的修改几个CSS文件就可以改变很多页面。Table要想改变的话，需要一个页面一个页面的去修改。

4）div表现和内容相分离，非常符合w3c标准。

5）table制作效率高于div

6）table浏览器兼容性高于div，我们常用的IE6.0，IE7.0火狐Firefox浏览器对div css设置上非常挑剔。

## 7、行级标签转块级标签，块级标签转行级标签

行级转块级：display:block

块级转行级：float:left

# 二、Java基础部分

## 1、java中有哪些基本类型？

byte、short、int、long、float、double、char、boolean

## 2、java为什么能够跨平台运行？

因为Java程序编译之后的代码不是能被硬件系统直接运行的代码，而是一种“中间码”——字节码。然后不同的硬件平台上安装有不同的Java虚拟机(JVM)，由JVM来把字节码再“翻译”成所对应的硬件平台能够执行的代码。因此对于Java编程者来说，不需要考虑硬件平台是什么。所以Java可以跨平台。

## 3、String是基本数据类型吗？我可不可以写个类继承于String？

不是，Strng是引用类型；String是final的类，是不可以被继承的。

## 4、谈谈&和&&的区别？

　　&和&&都可以用作逻辑与的运算符，表示逻辑与（and），当运算符两边的表达式的结果都为true时，整个运算结果才为true，否则，只要有一方为false，则结果为false。

&&还具有短路的功能，即如果第一个表达式为false，则不再计算第二个表达式。

&还可以用作位运算符，当&操作符两边的表达式不是boolean类型时，&表示按位与操作.。

## 5、Switch语句里面的条件可不可以是byte、long、String？使用时候还应注意什么？

switch里面的条件必须是能隐式的转化成为Int的故long和String不行，byte可以；使用Switch时候还应注意它的穿透，即每个case后要跟break;

## 6、short s1=1;s1=s1+1;有什么错？short s1 = 1;s1+=1 有什么错？

　　对于short s1 = 1; s1 = s1 + 1; 由于s1+1运算时会自动提升表达式的类型，所以结果是int型，再赋值给short类型s1时，编译器将报告需要强制转换类型的错误。

　　对于short s1 = 1; s1 += 1;由于 += 是java语言规定的运算符，java编译器会对它进行特殊处理，因此可以正确编译。

## 7、char为什么能存贮一个汉字？

char型变量是用来存储Unicode编码的字符的，unicode编码字符集中包含了全世界所 有的字体。

## 8、用最效率的办法算出2乘以8等于几？

2<<3 位移运算是最底层的运算，他直接操作的是二进制，故效率最快。

## 9、final修饰变量时，该变量是对象时，对象的值可不可以改变？

final修饰的变量指的是引用不可变，对象的值是可以改变的。

## 10、静态变量和实例变量的区别？

静态变量也称为类变量，归全类共有，它不依赖于某个对象，可通过类名直接访问；而实例变量必须依存于某一实例，只能通过对象才能访问到它。

## 11、面向对象的基本特征是什么？

1)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。

2)继承：子类拥有父类一切非私有的属性和方法。

3)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。

4)多态性：同一种事物的不同种表现形式。

## 12、作用域public,private,protected,以及不写时的区别？

作用域 当前类 同包 子孙类 其他

public √ √ √ √

protected √ √ √ ×

default √ √ × ×

private √ × × ×

不写时默认为default。

## 13、Overload和Override的区别。

（Overload）重载：发生在同一个类之中，方法名相同、参数列表不同，与返回值无关、与final无关、与修饰符无关、与异常无关。

（Override）重写：发生在子类和父类之间，方法名相同、参数列表相同、返回值相同、不能是final的方法、重写的方法不能有比父类方法更为严格的修饰符权限、重写的方法所抛出的异常不能比父类的更大。

如果父类私有的方法，子类拥有方法签名相同的方法，子类不属于重写父类的方法，该方法属于子类的新方法。

## 14、构造器可不可以被重载或重写？

构造器不能被继承，故不能被重写、但可以被重载。

## 15、Java中有没有多继承？

java中没有多继承，但是可以多实现，即一个类实现多个接口。

虽然没有多继承，但是java中接口可以近似的实现多继承，那就是接口；接口和接口之间可以进行多继承。

## 16、抽象类和接口的区别？

1）抽象类继承与object接口不继承object.

2）抽象类有构造器，接口中没有构造器。

3）抽象类中可以有普通成员变量和常量，接口中只能有常量，而且只能是public static final 不写默认。

4）抽象类中可以有抽象方法，也可以由普通的方法，接口中只能有抽象的方法而且修饰符只能是public abstract 不写默认。

5）抽象类中可以有final的方法,接口中不能有final的方法。

6）抽象类只能是单继承，多实现，接口是可以多继承其他接口，但是不能实现接口，和不能继承其他类。

7）抽象类中可以有静态的方法，接口中不可以。

## 17、java中实现多态的机制是什么？

重写、重载、父类的声明指向子类的对象。

## 18、int和integer的区别？

int是java的基本数据类型,integer是1.4版本后提供的基本类型包装类，当两者作为成员变量时，初始值分别为;int是0；integer是null;其中integer提供了一些对整数操作的方法，还定义了integer型数值的最值，其他基本类型也有对应的包装类，基本类型包装类的出现，使得java完全面向对象.

## 19、String和StringBuffer的区别？StringBuffer和StringBuilder区别？

　　String是不可变的，对String类的任何改变都会返回一个新的String 对象。

　　StringBuffer是可变的，对StringBuffer中的内容修改都是当前这个对象。

　　String重写了equals方法和hashCode方法，StringBuffer没有重写equals方法。String是final的类。StringBuffer不是。

String创建的字符串是在常量池中，创建的变量初始化一次，如果再对该字符串改变会产生新的字符串地址值，StringBuffer是在堆中创建对象，当对字符串改变时不会产生新的字符串地址值，如果对字符串进行频繁修改的话建议使用StringBuffer，以节省内存。

StringBuffer和StringBuilder，StringBuffer是线程安全的，StringBulider是线程不安全的。当不考虑并发问题时候，请使用StringBulider。

## 20、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。

## 21、数组中有没有length()方法，String中有没有length（）方法？

数组中没有length()方法,但是有length属性，String中有length()方法

## 22、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?

这道题很有争议，我是通过debug模式分为两种情况进行测试的.

1）finally中没有return时候：

会先执行try里面的，return会执行但是没有真正的return此时去执行了finally里面的，然后再返回来执行return.

2）finally中有return时候(其实这种情况不符合编程规范，会报黄线警告)：

会先执行try里面的，return会执行但是没有真正的return此时去执行了finally里面的，然后执行finally里面的return，直接返回。

## 23、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。

内部类要访问局部变量，局部变量必须定义成final类型。

finally是异常处理语句结构的一部分，表示总是执行。

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。JVM不保证此方法总被调用

## 24、‘==’和equals的区别？

‘==’比较的是两个变量的内容和在内存中的地址值是否全部相等，如果要比较两个基本数据类型那必须用’==’

equals如果没有重写，则和’==’的意义一样，如果重写了，则会会按照重写的内容进行比较，javaBean规定当重写equals时候必须重写hashCode，如果不重写会出现对象相同但是hashCode不同，这样会出现问题，eg:HashSet存储元素时候是按照hashCode，如果重写equals不重写hashCode会导致同一个对象，存储了两次。

## 25、error和exception有什么区别？

error表示恢复不是不可能但是很困难的情况下的一种严重问题，例如程序书写错误，虚拟机错误等，exception是一种设计和实现问题，如果程序运行正常，从不会发生的情况。error是可以避免的，exception是不可避免的。

## 26、heap和stack有什么区别。

java的内存分为两类，一类是栈内存，一类是堆内存。栈内存是指程序进入一个方法时，会为这个方法单独分配一块私属存储空间，用于存储这个方法内部的局部变量，当这个方法结束时，分配给这个方法的栈会释放，这个栈中的变量也将随之释放。

堆是与栈作用不同的内存，一般用于存放不放在当前方法栈中的那些数据，例如，使用new创建的对象都放在堆里，所以，它不会随方法的结束而消失。方法中的局部变量使用final修饰后，放在堆中，而不是栈中。

## 27、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。

## 28、什么是内部类？分为哪几种？

内部类是指在一个外部类的内部再定义一个类。内部类作为外部类的一个成员，并且依附于外部类而存在的。内部类可为静态，可用protected和private修饰（而外部类只能使用public和缺省的包访问权限）。

内部类主要有以下几类：成员内部类、局部内部类、静态内部类、匿名内部类。

## 29、为什么需要内部类？

典型的情况是，内部类继承自某个类或实现某个接口，内部类的代码操作创建其的外围类的对象。所以你可以认为内部类提供了某种进入其外围类的窗口。

使用内部类最吸引人的原因是：每个内部类都能独立地继承自一个（接口的）实现，所以无论外围类是否已经继承了某个（接口的）实现，对于内部类都没有影响。如果没有内部类提供的可以继承多个具体的或抽象的类的能力，一些设计与编程问题就很难解决。从这个角度看，内部类使得多重继承的解决方案变得完整。接口解决了部分问题，而内部类有效地实现了“多重继承”。

## 30、内部类可以引用它的包含类的成员吗？有没有什么限制？

完全可以。如果不是静态内部类，那没有什么限制！

如果你把静态嵌套类当作内部类的一种特例，那在这种情况下不可以访问外部类的普通成员变量，而只能访问外部类中的静态成员，例如，下面的代码：

class Outer

{

static int x;

static class Inner

{

void test()

{

syso(x);

}

}

}

答题时，也要能察言观色，揣摩提问者的心思，显然人家希望你说的是静态内部类不能访问外部类的成员，但你一上来就顶牛，这不好，要先顺着人家，让人家满意，然后再说特殊情况，让人家吃惊。

## 31、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)?

可以继承其他类或实现其他接口。不仅是可以，而是必须!

## 32、使用java命令查看java虚拟机版本

java –version

## 33、数字转字符有多少种方式，分别是什么

1）String.valueOf()

2）"" + 数字

3）Integer.toString()

## 34、Java创建对象有几种方式

1）new关键字

2）反射

3）克隆

4）反序列化

## 35、写一个反序列化实例

testObject.java

public class testObject implements Serializable{}

userTest.java

import java.io.File;

import java.io.FileOutputStream;

import java.io.ObjectOutputStream;

public class useTest{

public static void main(String[] args){

FileOutputStream fileOut = null;

ObjectOutputStream out = null;

Try{

fileOut = new FileOutputStream(new File(“c:\\test.dat”));

out = new ObjectOutputStream(fileOut);

out.writeObject(new testObject());

}catch(Exception e){

e.printStackTrace();

}finally{

try{

fileOut.close();

fileOut = null;

out.close();

out = null;

}catch(Exception e){

}

}

}

}

# 三、JavaScript/JQuery/Ajax部分

## 1、请写一段Javascript程序，对以下程序的用户输入日期的有效性进行判断，如果格式错误就提示用户。在程序的恰当位置注明怎样调用你写的程序。日期格式为：年年年年月月日日小时，例如2003010812。

<html>

<head>

<title>NewPage1</title>

</head>

<body>

<form method="POST" action="test.php"> <p>查询日期（yyyymmddhh）：<br>

从 <input type="text" name="DateFrom" size="20"><br>

到 <input type="text" name="DateTo" size="20"></p><p>

<input type="submit" value="提交" name="submit1">

<input type="reset" value="重写" name="reset1">

</p>

</form>

</body>

</html>

答：

<html>

<head>

<script type="text/javascript">

function fun(obj){

if(!isDate(obj.value)){

alert("日期格式错误！");

obj.value="";

}

}

function isDate(str){

var reg = /^(\d{4})(\d{2})(\d{2})(\d{2})$/;

var result = str.match(reg);

if(result == null||parseFloat(result[4])>23)return false;

var t=result[1]+","+parseFloat(result[2])+","+parseFloat(result[3]);

var d = new Date(result[1],result[2]-1,result[3]);

var newStr = d.getFullYear()+","+parseFloat(d.getMonth

()+1)+","+parseFloat(d.getDate());

return t == newStr;

}

</script>

</head>

<form method="POST" action=""> <p>查询日期(2003010812):<br>

从<input type="text" name="DateFrom" size="20" onblur="fun(this)"><br>

到<input type="text" name="DateTo" size="20" onblur="fun(this)"></p><p>

</form>

</html>

## 2、Java和Javascript区别在哪?

Java与JavaScript是目前比较流行的两种语言，单从表面上看，两者名称很相似，于是许多初学者容易将二者混淆，或者直接归为一类，其实不然，虽然两者有着紧密的联系，但确是两个完全不同的语言。接下来，笔者仅就她们的几个主要差别归纳起来。  
一．开发厂商  
众所周知，Java是SUN公司推出的程序设计语言，特别适合于Internet应用程序开发，其前身是Oak语言，而JavaScript则是NetScape公司的产品，是为了扩展NetScape Navigator功能而开发的一种可嵌入Web页面中的解释性语言，其前身是Live Script。由于Java的普及，NetScape和SUN两家公司签订合同后才将其命名为JavaScript。  
二．面向对象与基于对象  
Java是一种真正的纯面向对象编程语言，在Java中，一切都是对象；JavaScript是一种脚本语言，由于她本身提供了非常丰富的内部对象供程序员使用，因而它是基于对象的语言。  
三．开发和运行环境的不同  
若希望利用Java编写程序并使之运行，必须事先在系统内安装相应版本的JDK和JVM，保证代码能够得到编译和运行的环境；而编写JavaScript则相对简单，只需使用某种HTML文档编辑器甚至某种字符编辑器（如Notepad）即可，然后打开浏览器即可运行。  
四．变量的区别  
Java中使用变量在编译之前必须声明其数据类型，因而她采用的是强类型变量；JavaScript则不用在变量前声明类型，而是由解释器在运行时自动检查，所以她是弱类型变量。  
五．标签的不同  
利用Java写出的Applet小程序，在HTML中用<applet>……</applet>来标识；JavaScript程序在HTML中运行，其代码在<Script>……</Script>标签内。  
六．解释与编译  
Java源代码在执行前被编译，因而在网络应用中，必须要求客户端安装有解释平台，也就意味着Java应用不与HTML文档集成（Applet小程序例外）；JavaScript是一种解释性语言，其代码在发往客户端之前不需编译，而是将其嵌入到HTML文档中，一起发送给客户端，由浏览器解释执行。  
另外，JavaScript仅是一种解释性语言，并没有复杂的语法和规则，更不支持如Java里的继承这样的性质，因此也比Java更加容易学习。

## 3、列举javaScript的3种主要数据类型，2种复合数据类型和2种特殊数据类型。

主要数据类型：string, boolean, number

复合数据类型：function, object

## 4、谈谈你的JS的理解？

JavaScript是一种脚本语言，它采用小程序段的方式实现编程。像其它脚本语言一样,JavaScript同样已是一种解释性语言,它提供了一个易的开发过程。它的基本结构形式与C、C++、VB、Delphi十分类似。但它不像这些语言一样，需要先编译，而是在程序运行过程中被逐行地解释。它与HTML标识结合在一起，从而方便用户的使用操作。   
2）基于对象的语言。   
JavaScript是一种基于对象的语言，同时以可以看作一种面向对象的。这意味着它能运用自己已经创建的对象。因此，许多功能可以来自于脚本环境中对象的方法与脚本的相互作用。 3）简单性   
JavaScript的简单性主要体现在：首先它是一种基于Java基本语句和控制流之上的简单而紧凑的设计, 从而对于学习Java是一种非常好的过渡。其次它的变量类型是采用弱类型，并未使用严格的数据类型。   
4）安全性   
JavaScript是一种安全性语言，它不允许访问本地的硬盘，并不能将数据存入到服务器上，不允许对网络文档进行修改和删除，只能通过浏览器实现信息浏览或动态交互。从而有效地防止数据的丢失。   
5）动态性的   
JavaScript是动态的，它可以直接对用户或客户输入做出响应，无须经过Web服务程序。它对用户的反映响应，是采用以事件驱动的方式进行的。所谓事件驱动，就是指在主页(Home Page)中执行了某种操作所产生的动作，就称为“事件”(Event)。比如按下鼠标、移动窗口、选择菜单等都可以视为事件。当事件发生后，可能会引起相应的事件响应。   
6）跨平台性   
JavaScript是依赖于浏览器本身，与操作环境无关，只要能运行浏览器的计算机，并支持JavaScript的浏览器就可正确执行。从而实现了“编写一次,走遍天下”的梦想。实际上JavaScript最杰出之处在于可以用很小的程序做大量的事。无须有高性能的电脑，软件仅需一个字处理软件及一浏览器，无须WEB服务器通道，通过自己的电脑即可完成所有的事情。

## 5、ajax的优点？

使用ajax的最大优点，就是能在不更新整个页面的前提下维护数据。这使得web应用程序更为迅捷地回应用户动作，并避免了在网络上发送那些没有改变过的信息。

## 6、简述一下ajax调试代码查找错误的方法？

这是js调试代码存在已久的问题，简单的我们可以使用浏览器提供的错误提示框，还有可以使用DW CS4提供的纠正错误，或者通过专业的插件，如firebug等

## 7、简述ajax中Js脚本缓存问题该如何解决？

这个问题是大家遇到最常见的问题之一，因为修改了js内容调试的时候并不能显示新写代码的结果，是因为Js为了加速页面执行，当前页面会使用缓存保持当前调用的相同的连接，为了开发时调试方便可以在连接地址后面增加一个随机函数.

## 8、Ajax应用和传统的web应用有何不同？

在传统的javascript中，如果想得到服务器端数据库或文件上的信息，或者发送客户端信息到服务器，需要建立一个HTML form然后Post或者get提交数据到服务端。用户需要点击submit 来发送或者接受数据信息，然后等待服务器响应请求，页面重写加载，因为服务器每次都要返回一个新的页面，所以传统的web应用有可能会很慢而且用户交互不友好。

使用ajax就可以使javascript通过XMLHttpRequest对象直接与服务器进行交互。通过HTTPRequest，一个web页面可以发送一个请求道web服务器并且接受web服务器返回的信息(不需要加载任何界面)，展示给用户的还是同一个页面，用户感觉不到页面刷新，也看不到Javascript后台进行的发送请求和接受的响应。

## 9、javascript的作用？

表单验证、网页特效、网页游戏

## 10、为什么要有jquery？

1）jQuery是JavaScript的轻量级框架，对JavaScript进行了很好的封装，很多复杂的JavaScript代码不用写了，直接调用就可以，使开发简单、高效。

2）jQuery强大的选择器封装了DOM，操作网页元素更简单了。

3）在大型JavaScript框架中，jQuery对性能的理解最好，大小不超过30KB。

4）完善的ajax有着出色的浏览器兼容性，任何浏览器使用ajax都能兼容。

5）基于jQuery开发的插件目前已经有大约数千个。开发者可使用插件来进行表单确认、图表种类、字段提示、动画、进度条等任务。

## 11、jQuery选择器有多少种？

基本：

$("#myELement") ID选择器   
$("div") 标签选择器  
$(".myClass") 类选择器  
$("\*") 通配符选择器

层级选择器

过滤选择器

子元素选择器

## 12、jquery选择器有哪些优势？

简单的写法(‘#id’)用来代替document.getElementById()。

支持css选择器。

完善的处理机制，就算写错了Id也不会报错。

## 13、你是如何使用jquery中的ajax的？

如果是常规的ajax程序的话，使用load()、$.get()、$.post()，一般我会使用的是$.post()方法，如果需要设定，beforeSend（提交前回调函数），error（失败后处理）,success(成功后处理)，及complete（请求完成后处理）毁掉函数等，这个时候我会使用$.ajax()

## 14、jquery中的$.get和$.post请求区别？

1）$.get方法使用get方法来进行一步请求，$.post是使用post方法来进行请求。

2）get请求会讲参数跟在url后进行传递，而post请求则是作为Http消息的实体.内容发送给web服务器的，这种传递是对用户不可见的。

3）get方式传输的数据大小不能超过2kb而post请求要大的多

4）get方式请求的数据会被浏览器缓存起来，因此有安全问题

## 15、jquery中如何操作样式的？

addClass()来追加样式，removeClass()来删除样式，toggle()来切换样式。

## 16、如何设置和获取HTML和文本的值？

Html()方法，类似于innerHTML属性，可以用来读取或者设置某个元素中的HTML内容，text()类似于innerText属性，可以用来读取或这是某个元素的文本内容，val()可以用来设置和获取元素的值。

## 17、Jquery能做些什么？

1）获取页面元素

2）修改页面的外观

3）修改页面的内容

4）响应页面的操作

5）为页面添加动态效果

6）无需刷新页面，即可从服务器获取信息

7）简化常见的javascript的任务

## 18、在ajax中data主要有哪几种？

html拼接、json数组、form表单经过serialize()序列化的

## 19、jQuery中ajax由几部分组成？

1）请求url

2）请求参数

3）请求类型，get或post

4）回调函数

5）传输类型，html或json等

# 四、jsp/servlet部分

## 1、Tomcat的优化经验

去掉对web.xml的监视，把jsp提前编辑成Servlet。

有富余物理内存的情况，加大tomcat使用的jvm的内存

## 2、Tomcat根目录下有哪些文件

1）config 配置文件存放的路径

2）webapps 项目部署的目录

3）bin tomcat运行需要的脚本与jar包的目录

4）lib 运行项目时所需要的jar包的目录

5）work 部署项目的缓存目录

6）temp 临时文件存放的目录

7）logs 记录日志的目录

## 3、什么是TOMCAT,怎样启动停止，配置文件，日志文件的存储。

tomcat其实是一种web服务器，java编写的web项目可以部署在其上，用户在客户端请求时，都是先将请求发送到tomcat上，tomcat再将请求发送到对应的项目上。

启动tomcat

在Windows下：进入bin目录，双击startup.bat

在Linux下：cd进入bin目录，sh startup.sh

在开发工具eclipse中，右键选择Debug Server或者Run Server

停止tomcat

在Windows下：进入bin目录，双击shutdown.bat

在Linux下：cd进入bin目录，sh shutdown.sh

在开发工具eclipse中，选择服务器stop Server

配置文件在tomcat的config文件夹下

日志文件在logs文件夹下

## 4、解释一下什么是servlet;什么是servlet容器;

在web容器中运行的服务器端java程序，主要用于响应HTTP请求。Servlet一般用于mvc中的控制器部分。

用来管理servlet生命周期的应用程序如(tomcat webloc等)

## 5、说一说Servlet的生命周期，执行过程?

Servlet生命周期分为实例化、初始化、响应请求调用service()方法、消亡阶段调用destroy()方法。

执行过程如下：

1）当浏览器发送一个请求地址，tomcat会接收这个请求

2）tomcat会读取项目中的web.xml中的配置

3）当请求地址符合servlet-mapping标签映射的地址，会进入这个servlet

4）servlet首先会实例化（构造），然后初始化执行init()方法，init()方法至始至终执行一次，servlet对象是单实例

5）根据请求的方式是get或post，在service()方法中调用doget()或dopost()方法，完成此次请求

6）当服务器停止，会调用destroy()方法，销毁实例

## 6、实例化servlet有几种方式

Servlet实例化有两种，如下：

1. 第一次请求时，实例化servlet对象

2）在web.XML文件中的<Servlet></Servlet>之间添加<loadon-startup>1</loadon-startup>，tomcat启动时就会实例化servlet对象

## 7、HTTP请求的GET与POST方式的区别

Form中的get和post方法，在数据传输过程中分别对应了HTTP协议中的GET和POST方法。二者主要区别如下：  
 1）Get是用来从服务器上获得数据，而Post是用来向服务器上传数据；  
 2）Get将表单中数据按照variable=value的形式，添加到action所指向的URL后面，并且两者使用“?”连接，而各个变量之间使用“&”连接；Post是将表单中的数据放在form的数据体中，按照变量和值相对应的方式，传递到action所指向URL；  
 3）Get是不安全的，因为在传输过程，数据被放在请求的URL中；Post的所有操作对用户来说都是不可见的；  
 4）Get传输的数据量小，这主要是因为受URL长度限制；而Post可以传输大量的数据，所以在上传文件只能使用Post；  
 5）Get限制Form表单的数据集必须为ASCII字符，而Post支持整个ISO10646字符集；  
 6）Get是Form的默认方法。

## 8、请写一个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 {

}

}

## 9、forward 和redirect的区别?

forward是容器中控制权的转向，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，并且从浏览器的地址栏中可以看到跳转后的链接地址。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。

总结：forward是程序内部跳转，不会跳出tomcat服务器，redirect可以外部跳转，从一个服务器跳转到另一个服务器。

## 10、servlet中怎么定义forward 和redirect

转发：request.getRequestDispatcher (“demo.jsp"). forward(request, response);

重定向：response.sendRedirect(“demo.jsp");

## 11、过滤器有哪些作用？

可以验证客户是否来自可信的网络，可以对客户提交的数据进行重新编码，可以从系统里获得配置的信息，可以过滤掉客户的某些不应该出现的词汇，可以验证用户是否登录，可以验证客户的浏览器是否支持当前的应用，可以记录系统的日志等等。

## 12、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EL表达式；

<%@include file=”filename”%>

<%@taglib prefix=”c”uri=”http://……”%>

## 13、JSP和Servlet中的请求转发分别如何实现？

JSP中的请求转发可利用forward动作实现：<jsp:forward />；

Serlvet中实现请求转发的方式为：

getServletContext().getRequestDispatcher(path).forward(req,res)。

## 14、JSP乱码如何解决？

　　1）JSP页面乱码

　　<%@ page contentType=”text/html ; charset=utf-8”%>

　　2）表单提交中文时出现乱码

　　request.setCharacterEncoding(“utf-8”);

　　3）数据库连接出现乱码

　　是数据库连接中加入useUnicode=true&characterEncoding=utf-8;

## 15、session 和 application的区别？

1）两者的作用范围不同：

Session对象是用户级的，而Application是应用程序级别的

一个用户一个session对象，每个用户的session对象不同，在用户所访问的网站多个页面之间共享同一个session对象

一个Web应用程序一个application对象，每个Web应用程序的application对象不同，但一个Web应用程序的多个用户之间共享同一个application对象。

两者的生命周期不同：

session对象的生命周期：用户首次访问网站创建，用户离开该网站 (不一定要关闭浏览器) 消亡。

application对象的生命周期：启动Web服务器创建，关闭Web服务器销毁。

## 16、jsp有哪些内置对象?作用分别是什么?

JSP共有以下9种基本内置组件

　　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

## 17、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

## 18、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 ，先将嵌入的jsp页面编译，然后把编译后的内容放入到主页面进行处理，编译两次。

静态INCLUDE用include伪码实现，使用jsp指令引用<%@ include file=included.htm %>，不会检查所含文件的变化，适用于包含静态页面，先将内容先包含到主页面然后在一起编译，只编译一次。

## 19、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。

## 20、页面传递对象的方法？

Request、session、application、cookie等

## 21、Cookied和session区别？

1）cookie数据存放在客户的浏览器上，session数据放在服务器上。

2）cookie不是很安全，别人可以分析存放在本地的COOKIE并进行COOKIE欺骗考虑到安全应当使用session。

3）session会在一定时间内保存在服务器上。当访问增多，会比较占用你服务器的性能考虑到减轻服务器性能方面，应当使用COOKIE。

4）单个cookie保存的数据不能超过4K，很多浏览器都限制一个站点最多保存20个cookie。

# 五、数据库部分

## 1、触发器的作用？

触发器是一中特殊的存储过程，主要是通过事件来触发而被执行的。它可以强化约束，来维护数据的完整性和一致性，可以跟踪数据库内的操作从而不允许未经许可的更新和变化。可以联级运算。如，某表上的触发器上包含对另一个表的数据操作，而该操作又会导致该表触发器被触发。

## 2、什么是存储过程？用什么来调用？

存储过程是一个预编译的SQL语句，优点是允许模块化的设计，就是说只需创建一次，以后在该程序中就可以调用多次。如果某次操作需要执行多次SQL，使用存储过程比单纯SQL语句执行要快。

调用：

1）可以用一个命令对象来调用存储过程。

2）可以供外部程序调用，比如：java程序。

## 3、存储过程的优缺点？

优点：

1）存储过程是预编译过的，执行效率高。

2）存储过程的代码直接存放于数据库中，通过存储过程名直接调用，减少网络通讯。

3）安全性高，执行存储过程需要有一定权限的用户。

4）存储过程可以重复使用，可减少数据库开发人员的工作量。

缺点：移植性差

## 4、存储过程与函数的区别

|  |  |
| --- | --- |
| 存储过程 | 函数 |
| 用于在数据库中完成特定的操作或者任务（如插入、删除等） | 用于特定的数据（如选择） |
| 程序头部声明用procedure | 程序头部声明用function |
| 程序头部声明时不需描述返回类型 | 程序头部声明时要描述返回类型，而且PL/SQL块中至少要包括一个有效的return语句 |
| 可以使用in/out/in out 三种模式的参数 | 可以使用in/out/in out 三种模式的参数 |
| 可作为一个独立的PL/SQL语句来执行 | 不能独立执行，必须作为表达式的一部分调用 |
| 可以通过out/in out 返回零个或多个值 | 通过return语句返回一个值，且改值要与声明部分一致，也可以是通过out类型的参数带出的变量 |
| SQL语句(DML 或SELECT)中不可调用存储过程 | SQL语句(DML 或SELECT)中可以调用函数 |

## 5、索引的作用？和它的优点缺点是什么？

索引就一种特殊的查询表，数据库的搜索可以利用它加速对数据的检索。它很类似与现实生活中书的目录，不需要查询整本书内容就可以找到想要的数据。索引可以是唯一的，创建索引允许指定单个列或者是多个列。缺点是它减慢了数据录入的速度，同时也增加了数据库的尺寸大小。

## 6、什么样的字段适合建索引

唯一、不为空、经常被查询的字段

## 7、索引类型有哪些？

逻辑上：  
Single column 单行索引  
Concatenated 多行索引  
Unique 唯一索引  
NonUnique 非唯一索引  
Function-based函数索引  
Domain 域索引  
物理上：  
Partitioned 分区索引  
NonPartitioned 非分区索引  
B-tree：  
Normal 正常型B树  
Rever Key 反转型B树   
Bitmap 位图索引

## 8、什么是事务？什么是锁？

事务就是被绑定在一起作为一个逻辑工作单元的SQL语句分组，如果任何一个语句操作失败那么整个操作就被失败，以后操作就会回滚到操作前状态，或者是上有个节点。为了确保要么执行，要么不执行，就可以使用事务。要将有组语句作为事务考虑，就需要通过ACID测试，即原子性，一致性，隔离性和持久性。

锁：在所以的DBMS中，锁是实现事务的关键，锁可以保证事务的完整性和并发性。与现实生活中锁一样，它可以使某些数据的拥有者，在某段时间内不能使用某些数据或数据结构。当然锁还分级别的。

## 9、什么叫视图？游标是什么？

视图：是一种虚拟的表，具有和物理表相同的功能。可以对视图进行增，改，查，操作，试图通常是有一个表或者多个表的行或列的子集。对视图的修改不影响基本表。它使得我们获取数据更容易，相比多表查询。

  游标：是对查询出来的结果集作为一个单元来有效的处理。游标可以定在该单元中的特定行，从结果集的当前行检索一行或多行。可以对结果集当前行做修改。一般不使用游标，但是需要逐条处理数据的时候，游标显得十分重要。

## 10、视图的优缺点

优点：

1）对数据库的访问，因为视图可以有选择性的选取数据库里的一部分。   
2）用户通过简单的查询可以从复杂查询中得到结果。   
3）维护数据的独立性，试图可从多个表检索数据。   
4）对于相同的数据可产生不同的视图。

缺点：

性能：查询视图时，必须把视图的查询转化成对基本表的查询，如果这个视图是由一个复杂的多表查询所定义，那么，即使是视图的一个简单查询，也把它变成一个复杂的结合体，需要花费一定的时间。

## 11、列举几种表连接方式,有什么区别？

内连接、自连接、外连接（左、右、全）、交叉连接

内连接：只有两个元素表相匹配的才能在结果集中显示。

外连接：

左外连接:左边为驱动表，驱动表的数据全部显示，匹配表的不匹配的不会显示。

右外连接:右边为驱动表，驱动表的数据全部显示，匹配表的不匹配的不会显示。

全外连接：连接的表中不匹配的数据全部会显示出来。

交叉连接： 笛卡尔效应，显示的结果是链接表数的乘积。

## 12、主键和外键的区别？

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。

## 13、在数据库中查询语句速度很慢，如何优化？

1.建索引

2.减少表之间的关联

3.优化sql，尽量让sql很快定位数据，不要让sql做全表查询，应该走索引,把数据 量大的表排在前面

4.简化查询字段，没用的字段不要，已经对返回结果的控制，尽量返回少量数据

5.尽量用PreparedStatement来查询，不要用Statement

## 14、数据库三范式是什么?

第一范式（1NF）：字段具有原子性,不可再分。所有关系型数据库系统都满足第一范式。

数据库表中的字段都是单一属性的，不可再分。例如，姓名字段，其中的姓和名必须作为一个整体，无法区分哪部分是姓，哪部分是名，如果要区分出姓和名，必须设计成两个独立的字段。

第二范式（2NF）：是在第一范式（1NF）的基础上建立起来的，即满足第二范式（2NF）必须先满足第一范式（1NF）。

要求数据库表中的每个实例或行必须可以被惟一地区分。通常需要为表加上一个列，以存储各个实例的惟一标识。这个惟一属性列被称为主关键字或主键。

第二范式（2NF）要求实体的属性完全依赖于主关键字。所谓完全依赖是指不能存在仅依赖主关键字一部分的属性，如果存在，那么这个属性和主关键字的这一部分应该分离出来形成一个新的实体，新实体与原实体之间是一对多的关系。为实现区分通常需要为表加上一个列，以存储各个实例的惟一标识。简而言之，第二范式就是非主属性非部分依赖于主关键字。

第三范式（3NF）：必须先满足第二范式（2NF）。简而言之，第三范式（3NF）要求一个数据库表中不包含已在其它表中已包含的非主关键字信息。

所以第三范式具有如下特征：

         1，每一列只有一个值

         2，每一行都能区分。

         3，每一个表都不包含其他表已经包含的非主关键字信息。

例如，帖子表中只能出现发帖人的id，而不能出现发帖人的id，还同时出现发帖人姓名，否则，只要出现同一发帖人id的所有记录，它们中的姓名部分都必须严格保持一致，这就是数据冗余。

## 15、union和union all有什么不同?

UNION在进行表链接后会筛选掉重复的记录，所以在表链接后会对所产生的结果集进行排序运算，删除重复的记录再返回结果。实际大部分应用中是不会产生重复的记录，最常见的是过程表与历史表UNION。

UNION ALL只是简单的将两个结果合并后就返回。这样，如果返回的两个结果集中有重复的数据，那么返回的结果集就会包含重复的数据了。

从效率上说，UNION ALL 要比UNION快很多，所以，如果可以确认合并的两个结果集中不包含重复的数据的话，那么就使用UNION ALL。

## 16、Varchar2和varchar有什么区别？

Char的长度是固定的，而varchar2的长度是可以变化的，比如，存储字符串“abc”对于char(20)，表示你存储的字符将占20个字节，包含17个空，而同样的varchar2（20）只占了3个字节，20只是最大值，当你存储的字符小于20时，按实际长度存储。

char的效率要被varchar2的效率高。

目前varchar是varchar2的同义词，工业标准的varchar类型可以存储空字符串，但是oracle不能这样做，尽管它保留以后这样做的权利。Oracle自己开发了一个数据类型varchar2,这个类型不是一个标准的varchar，他将在数据库中varchar列可以存储空字符串的特性改为存储null值，如果你想有向后兼容的能力，oracle建议使用varchar2而不是varchar

## 17、Oracle和Mysql的区别？

1）库函数不同。

2）Oracle是用表空间来管理的，Mysql不是。

3）显示当前所有的表、用户、改变连接用户、显示当前连接用户、执行外部脚本的语句的不同。

4）分页查询时候时候，mysql用limt oracle用rownum

5）sql的语法的不同。

## 18、Oracle语句有多少类型

Oracle语句分三类：DDL、DML、DCL。

 DDL（Data Definition Language）数据定义语言，包括：

Create语句：可以创建数据库和数据库的一些对象。

Drop语句：可以删除数据表、索引、触发程序、条件约束以及数据表的权限等。

Alter语句：修改数据表定义及属性。

Truncate语句：删除表中的所有记录,包括所有空间分配的记录被删除。

DML（Data Manipulation Language）数据操控语言，包括：

Insert语句：向数据表张插入一条记录。

 Delete语句：删除数据表中的一条或多条记录，也可以删除数据表中的所有记录，但是它的操作对象仍是记录。

Update语句：用于修改已存在表中的记录的内容。

Select语句：用于查询已存在表中的记录的内容。

DCL（Data Control Language）数据库控制语言，包括：

Grant语句：允许对象的创建者给某用户或某组或所有用户（PUBLIC）某些特定的权限。

Revoke语句：可以废除某用户或某组或所有用户访问权限

## 19、oracle分页语句

使用rownum，两种如下：

第一种：

select \* from (select t.\*,rownum row\_num from mytable t) b where b.row\_num between 1 and 10

第二种：

select \* from ( select a.\*, rownum rn from mytable a where rownum <= 10 ) where rn >= 1

使用rowid，如下：

select \* from scott.emp where rowid in (select rd from (select rowid as rd ,rownum as rn from scott.emp ) where rn<=6 and rn>3)

## 20、从数据库中随机取50条

select \* from (select \* from t\_example order by dbms\_random.random) where rownum <= 50

## 21、order by与group by的区别

order by 排序查询、asc升序、desc降序

group by 分组查询、having 只能用于group by子句、作用于组内，having条件子句可以直接跟函数表达式。使用group by 子句的查询语句需要使用聚合函数。

## 22、commit在哪里会运用

oracle的commit就是DML语句提交数据（这里是释放锁不是锁表），在未提交前你前面的操作更新的都是内存，没有更新到物理文件中。

执行commit从用户角度讲就是更新到物理文件了，事实上commit时还没有写date file，而是记录了redo log file，要从内存写到data物理文件，需要触发检查点，由DBWR这个后台进程来写，这里内容有点多的，如果不深究的话你就理解成commit即为从内存更新到物理文件。

## 23、行转列、列换行怎么转

1）使用decode函数

2）使用case when语句

## 24、什么是PL/SQL？

[PL/SQL](http://baike.baidu.com/view/470171.htm" \t "_blank)是一种程序语言，叫做过程化SQL语言（Procedural Language/SQL）。PL/SQL是[Oracle数据库](http://baike.baidu.com/view/1685727.htm" \t "_blank)对SQL语句的扩展。在普通SQL语句的使用上增加了编程语言的特点，所以PL/SQL把数据操作和查询语句组织在PL/SQL代码的过程性单元中，通过逻辑判断、循环等操作实现复杂的功能或者计算。PL/SQL 只有 Oracle 数据库有。 MySQL 目前不支持 PL/SQL 的。

## 25、序列的作用

Oracle使用序列来生成唯一编号，用来处理一个表中自增字段。 Oracle序列是原子对象，并且是一致的。也就是说，一旦您访问一个序列号，Oracle将在处理下一个请求之前自动递增下一个编号，从而确保不会出现重复值。

## 26、表和视图的关系

视图其实就是一条查询sql语句，用于显示一个或多个表或其他视图中的相关数据。

表就是关系数据库中实际存储数据用的。

## 27、oracle基本数据类型

1）字符串类型

char、nchar、varchar、varchar2、nvarchar2

2）数字类型

number、integer

3）浮点类型

binary\_float、binary\_double、float

4）日期类型

date、 timestamp

5）LOB类型

blob、clob、nclob、bfile

## 28、truncate与 delete区别

TRUNCATE TABLE 在功能上与不带 WHERE 子句的 DELETE 语句相同：二者均删除表中的全部行。但 TRUNCATE TABLE 比 DELETE 速度快，且使用的系统和事务日志资源少。 DELETE 语句每次删除一行，并在事务日志中为所删除的每行记录一项。  
TRUNCATE TABLE 通过释放存储表数据所用的数据页来删除数据，并且只在事务日志中记录页的释放。   
TRUNCATE,DELETE,DROP放在一起比较：  
TRUNCATE TABLE：删除内容、释放空间但不删除定义。  
DELETE TABLE:删除内容不删除定义，不释放空间。  
DROP TABLE：删除内容和定义，释放空间。

## 29、oracle获取系统时间

select to\_char(sysdate, 'yyyy-MM-dd HH24:mi:ss') from dual;

## 30、oracle怎么去除去重

使用distinct关键字

## 31、现在有表:

A(id ,name,regdate)

B(id,groupid)

C(id,name2)

写出下面的SQL语句

A)统计A表中每个月注册用户数

select count(\*),to\_char(regdate,'yyyymm') from A group by to\_char(regdate,'yyyymm');

B)统计A表中有姓名相同的用户数

select count(\*) from (select name from A group by name having count(\*) >1);

C)如果表A中有姓名相同的用户,把相同的查出,写入表C中

nsert into C(name2) select name from A group by name having count(\*) >1;

D)A中ID有多个相同的数据,A中姓名相同的ID只保留注册时间最大的数据

delete from E where e.regdate < (select max(regdate) from a X where E.id = X.id);

## 32、现有表：

Student(S#,SN,SD)学生表

Course(C#,CN,T#)课程表

SC（S#,C#,score）成绩表

1.查询选了课程‘税收’的学生的学号和名字

答：

select SN,SD from Student where S# in(  
select S# from Course C , SC where C.C#=SC.C# and CN=’税收基础’);

2.查询选课数量大于5的学生的学号和名字

答：

select SN,SD from Student where S# in (

select S# from SC group by S# having count(distinct C#) > 5);

)

3.建立一个学生表students，包括name,age,head\_teacher,id,score(姓名，年龄，班主任，学号，成绩)

Create table students

(

Id number(9) not null primary key,

Name varchar2(40) not null,

Age int check(age between 0 and 100),

Head\_teacher vachar2(40),

Score float

);

4.对上表插入一条记录，姓名：张三，年龄：18，班主任：李四，学号：22

Insert into student(id,name,age,head\_teacher) values(‘22’,’张三’,’18’,’李四’)；

5.对上表中的age+name创建一个索引，并说明它的作用和使用方法

Create index student\_index on students(age,name);

## 33、怎样把这样一个表（表名：tmp\_table\_201307）

year month amount

2012 1 1.1

2012 2 1.2

2012 3 1.3

2012 4 1.4

2012 4 1.6

2013 1 2.1

2013 2 2.2

2013 2 2.2

2013 3 2.3

　2013 4 2.4

　　查成这样一个结果

　　year m1 m2 m3 m4

　　2012 1.1 1.2 1.3 3

　　2013 2.1 4.4 2.3 2.4

　　Select

　　year,

　　Sum(case when month = 1 then amount else 0 end) as m1,

　　Sum(case when month = 2 then amount else 0 end) as m2,

　　Sum(case when month = 3 then amount else 0 end) as m3,

　　Sum(case when month = 4 then amount else 0 end) as m4

　　From tmp\_table\_201307 a

　　Group by year

　　Order by 1;

## 34、数据库中有一张表ismg\_icp\_flow，结构如下

SQL> desc ismg\_icp\_flow

Name Null Type

————————————————————————————————

ICPNO NOT NULL VARCHAR2(6)

SERVICE\_ID NOT NULL VARCHAR2(10)

STAT\_MIN NOT NULL DATETIME

MT\_RECV\_OK NOT NULL NUMBER(10)

请写出一条SQL语句同时满足以下3个条件：

计算MT\_RECV\_OK的总和（显示为total），并以此对结果集进行升序排序

以ICPNO和SERVICE\_ID两个字段进行分组

所得出数据的STAT\_MIN必须为大于2003年1月1号0时0分并且小于2003 年2月1号0时0分

期望的输出结果如下：

ICPNO SERVICE\_ID TOTAL

———— —————— ————————————

901001 7700 271965

901001 7800 3857795

答：

select ICPNO,SERVICE\_ID,sum(MT\_RECV\_OK) TOTAL

from ismg\_icp\_flow

where STAT\_MIN between to\_date('2003-1-1','yyyy-mm-dd')

and to\_date('2003-2-1','yyyy-mm-dd')

group by ICPNO,SERVICE\_ID

order by TOTAL;

# 六、Java高级部分

## 1、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？

实现线程有两种方式：1.继承Thread类，重写run方法，在调用start方法。

实现Runnable接口，重写run方法。在传给Thread构造器，调用时调用Thread的start方法。

用synchronized关键字修饰同步方法 。

不使用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。

## 2、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。 wait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。

## 3、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?

分几种情况：

1）其他方法前是否加了synchronized关键字，如果没加，则能。

2）如果这个方法内部调用了wait，则可以进入其他synchronized方法。

3）如果其他个方法都加了synchronized关键字，并且内部没有调用wait，则不能。

4）如果其他方法是static，它用的同步锁是当前类的字节码，与非静态的方法不能同步，因为非静态的方法用的是this。

## 4、线程的基本概念

一个程序中可以有多条执行线索同时执行，一个线程就是程序中的一条执行线索，每个线程上都关联有要执行的代码，即可以有多段程序代码同时运行，每个程序至少都有一个线程，即main方法执行的那个线程。如果只是一个cpu，它怎么能够同时执行多段程序呢？这是从宏观上来看的，cpu一会执行a线索，一会执行b线索，切换时间很快，给人的感觉是a,b在同时执行，好比大家在同一个办公室上网，只有一条链接到外部网线，其实，这条网线一会为a传数据，一会为b传数据，由于切换时间很短暂，所以，大家感觉都在同时上网。

## 5、什么是多线程

线程是程序执行流的最小单元，相对独立、可调度的执行单元，是系统独立调度和分派CPU的基本单位。在单个程序中同时运行多个线程完成不同的工作，称为多线程。

## 6、程序、进程、线程之间的关系

程序是一段静态的代码，是应用软件执行的蓝本。

进程是程序一次动态执行的过程，它对应了从代码加载、执行完毕的一个完整过程，这也是进程开始到消亡的过程。

线程是进程中独立、可调度的执行单元，是执行中最小单位。

一个程序一般是一个进程，但可以一个程序中有多个进程。

一个进程中可以有多个线程，但只有一个主线程。

Java应用程序中默认的主线程是main方法，如果main方法中创建了其他线程，JVM就会执行其他的线程。

## 7、创建线程有几种方式，分别是什么？

创建线程有三种方式：

1）是继承Thread类，创建格式如下：

Thread thread = new Thread();

2）是实现Runnable接口，创建格式如下：

Thread thread = new Thread(new Runnable());

其实Thread类实现了Runnable接口

3）通过线程池方式，获取线程

 package com.myjava.thread;  
import java.util.concurrent.ExecutorService;  
import java.util.concurrent.Executors;  
public class ThreadPool {  
    private  static int POOL\_NUM = 10;  
    public static void main(String[] agrs){  
        ExecutorService executorService = Executors.newFixedThreadPool(5);  
        for (int i = 0; i < POOL\_NUM; i++) {  
            RunnableThread thread = new RunnableThread();  
            executorService.execute(thread);  
        }  
    }  
}  
class RunnableThread implements  Runnable{  
    private   int THREAD\_NUM = 10;  
    public void run() {  
        for (int i = 0; i <THREAD\_NUM; i++) {  
            System.out.println("线程"+Thread.currentThread()+i);  
        }  
          
    }  
}

## 8、线程的生命周期

创建--运行--中断--死亡

创建：线程构造

运行：调用start()方法，进入run()方法

中断：sleep()、wait()

死亡：执行完run()方法或强制run()方法结束，线程死亡

## 9、线程currentThread()与interrupt()方法的使用

currentThread()方法是获取当前线程

interrupt()唤醒休眠线程，休眠线程发生InterruptedException异常

## 10、线程状态

1）新建状态（New）：新创建了一个线程对象。

2）就绪状态（Runnable）：线程对象创建后，其他线程调用了该对象的start()方法。该状态的线程位于可运行线程池中，变得可运行，等待获取CPU的使用权。

3）运行状态（Running）：就绪状态的线程获取了CPU，执行程序代码。

4）阻塞状态（Blocked）：阻塞状态是线程因为某种原因放弃CPU使用权，暂时停止运行。

5）死亡状态（Dead）：线程执行完了或者因异常退出了run()方法，该线程结束生命周期。

## 11、什么是java序列化，如何实现java序列化？

通俗的说，就是可以将内存中Java对象可以写在硬盘上(序列化到硬盘上)，反序列化就是讲硬盘的内容读取到内存中去；java是通过实现Serializable接口，实现的序列化，Serializable接口里面没有任何的方法，只是个标示接口。

## 12、编写一个程序，将d:\java目录下的所有.java文件复制到d:\jad目录下，并将原来文件的扩展名从.java改为.jad。

答：listFiles方法接受一个FileFilter对象，这个FileFilter对象就是过虑的策略对象，不同的人提供不同的FileFilter实现，即提供了不同的过滤策略。

import java.io.File;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.FilenameFilter;

import java.io.IOException;

import java.io.InputStream;

import java.io.OutputStream;

public class Jad2Java {

public static void main(String[] args) throws Exception {

File srcDir = new File("java");

if(!(srcDir.exists() && srcDir.isDirectory()))

throw new Exception("目录不存在");

File[] files = srcDir.listFiles(

new FilenameFilter(){

public boolean accept(File dir, String name) {

return name.endsWith(".java");

}

}

);

System.out.println(files.length);

File destDir = new File("jad");

if(!destDir.exists()) destDir.mkdir();

for(File f :files){

FileInputStream fis = new FileInputStream(f);

String destFileName = f.getName().replaceAll("\\.java$", ".jad");

FileOutputStream fos = new FileOutputStream(new File(destDir,destFileName));

copy(fis,fos);

fis.close();

fos.close();

}

}

private static void copy(InputStream ips,OutputStream ops) throws Exception{

int len = 0;

byte[] buf = new byte[1024];

while((len = ips.read(buf)) != -1){

ops.write(buf,0,len);

}

}

}

由本题总结的思想及策略模式的解析：

1.

class jad2java{

1. 得到某个目录下的所有的java文件集合

1.1 得到目录 File srcDir = new File("d:\\java");

1.2 得到目录下的所有java文件：File[] files = srcDir.listFiles(new MyFileFilter());

1.3 只想得到.java的文件： class MyFileFilter implememyts FileFilter{

public boolean accept(File pathname){

return pathname.getName().endsWith(".java")

}

}

2.将每个文件复制到另外一个目录，并改扩展名

2.1 得到目标目录，如果目标目录不存在，则创建之

2.2 根据源文件名得到目标文件名，注意要用正则表达式，注意.的转义。

2.3 根据表示目录的File和目标文件名的字符串，得到表示目标文件的File。

//要在硬盘中准确地创建出一个文件，需要知道文件名和文件的目录。

2.4 将源文件的流拷贝成目标文件流，拷贝方法独立成为一个方法，方法的参数采用抽象流的形式。

//方法接受的参数类型尽量面向父类，越抽象越好，这样适应面更宽广。

}

分析listFiles方法内部的策略模式实现原理

File[] listFiles(FileFilter filter){

File[] files = listFiles();

//Arraylist acceptedFilesList = new ArrayList();

File[] acceptedFiles = new File[files.length];

int pos = 0;

for(File file: files){

boolean accepted = filter.accept(file);

if(accepted){

//acceptedFilesList.add(file);

acceptedFiles[pos++] = file;

}

}

Arrays.copyOf(acceptedFiles,pos);

//return (File[])accpetedFilesList.toArray();

}

## 13、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

## 14、字节流与字符流的区别

把一片二进制数据数据逐一输出到某个设备中，或者从某个设备中逐一读取一片二进制数据，不管输入输出设备是什么，我们要用统一的方式来完成这些操作，用一种抽象的方式进行描述，这个抽象描述方式起名为IO流，对应的抽象类为OutputStream和InputStream ，不同的实现类就代表不同的输入和输出设备，它们都是针对字节进行操作的。

在应用中，经常要完全是字符的一段文本输出去或读进来，用字节流可以吗？计算机中的一切最终都是二进制的字节形式存在。对于“中国”这些字符，首先要得到其对应的字节，然后将字节写入到输出流。读取时，首先读到的是字节，可是我们要把它显示为字符，我们需要将字节转换成字符。由于这样的需求很广泛，人家专门提供了字符流的包装类。

底层设备永远只接受字节数据，有时候要写字符串到底层设备，需要将字符串转成字节再进行写入。字符流是字节流的包装，字符流则是直接接受字符串，它内部将串转成字节，再写入底层设备，这为我们向IO设别写入或读取字符串提供了一点点方便。

字符向字节转换时，要注意编码的问题，因为字符串转成字节数组，

其实是转成该字符的某种编码的字节形式，读取也是反之的道理。

讲解字节流与字符流关系的代码案例：

import java.io.BufferedReader;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.InputStreamReader;

import java.io.PrintWriter;

public class IOTest {

public static void main(String[] args) throws Exception {

String str = "中国人";

/\*FileOutputStream fos = new FileOutputStream("1.txt");

fos.write(str.getBytes("UTF-8"));

fos.close();\*/

/\*FileWriter fw = new FileWriter("1.txt");

fw.write(str);

fw.close();\*/

PrintWriter pw = new PrintWriter("1.txt","utf-8");

pw.write(str);

pw.close();

/\*FileReader fr = new FileReader("1.txt");

char[] buf = new char[1024];

int len = fr.read(buf);

String myStr = new String(buf,0,len);

System.out.println(myStr);\*/

/\*FileInputStream fr = new FileInputStream("1.txt");

byte[] buf = new byte[1024];

int len = fr.read(buf);

String myStr = new String(buf,0,len,"UTF-8");

System.out.println(myStr);\*/

BufferedReader br = new BufferedReader(

new InputStreamReader(

new FileInputStream("1.txt"),"UTF-8"

)

);

String myStr = br.readLine();

br.close();

System.out.println(myStr);

}

总结：很简单，字符流的底层就是字节流。而字符流主要是读取文本文件内容的，可以一个字符一个字符的读取，也可以一行一行的读取文本文件内容。而字节流读取单位为byte.byte作为计算机存储最基本单位，可以用字节流来读取很多其他格式的文件，比如图片视频等等。基于B/S和C/S的文件传输都可以采用字节流的形式。

## 15、怎么判断指定路径是否为目录

File f = new File(fileName); //构造文件File类

f.isDirectory(); //判断是否为目录

## 16、怎么获取指定路径下的全部文件

File f = new File(filePath); //构造文件File类

String[] fileName = f.list(); //获取目录下的文件名

File[] files = f.listFiles(); //获取目录下的文件

## 17、Java怎么读取文件和写入文件

读取文件：

public class FileRead {

/\*\*

\* 1、找到指定的文件

\* 2、根据文件创建文件的输入流

\* 3、创建字节数组

\* 4、读取内容，放到字节数组里面

\* 5、关闭输入流

\* @param args

\*/

public static void main(String[] args) {

File file = new File("E:" + File.separator + "hello.txt"); //构建指定文件

InputStream in = null;

try {

in = new FileInputStream(file); //根据文件创建文件的输入流

byte[] data = new byte[1024]; //创建字节数组

in.read(data); //读取内容，放到字节数组里面

System.out.println(new String(data));

} catch (FileNotFoundException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

}finally{

try {

in.close(); //关闭输入流

} catch (IOException e) {

e.printStackTrace();

}

}

}

}

写入文件：

public class FileWriter {

/\*\*

\* 文件的输出流，用来写入文件内容

\* 1、找到指定的文件

\* 2、根据文件创建文件的输出流

\* 3、把内容转换成字节数组

\* 4、向文件写入内容

\* 5、关闭输出流

\* @param args

\*/

public static void main(String[] args) {

File file = new File("E:" + File.separator + "hello.txt"); //构建指定文件

OutputStream out = null;

try {

out = new FileOutputStream(file); 根据文件创建文件的输出流

String message = "黄晓明与bady结婚了，扬子和黄圣依有女儿了。";

byte[] mesByte = message.getBytes(); //把内容转换成字节数组

out.write(mesByte); //向文件写入内容

} catch (FileNotFoundException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

}finally{

try {

out.close(); //关闭输出流

} catch (IOException e) {

e.printStackTrace();

}

}

}

}

## 18、java怎么复制文件

public class FileCopy {

/\*\*

\* 实现思路

\* 1、构建源文件与目标文件

\* 2、源文件创建输入流，目标文件创建输出流

\* 3、创建字节数组

\* 4、使用循环，源文件读取一部分内容，目标文件写入一部分内容，直到写完所有内容

\* 5、关闭源文件输入流，目标文件输出流

\* @param args

\*/

public static void main(String[] args) {

//构建源文件

File file = new File("E:" + File.separator + "helloworld.txt");

//构建目标文件

File fileCopy = new File("D:" + File.separator + "helloworld.txt");

InputStream in = null;

OutputStream out = null;

try{

//目标文件不存在就创建

if(!(fileCopy.exists())) {

fileCopy.createNewFile();

}

//源文件创建输入流

in = new FileInputStream(file);

//目标文件创建输出流

out = new FileOutputStream(fileCopy, true);

//创建字节数组

byte[] temp = new byte[1024];

int length = 0;

//源文件读取一部分内容

while((length = in.read(temp)) != -1) {

//目标文件写入一部分内容

out.write(temp, 0, length);

}

}catch(IOException e) {

e.printStackTrace();

}finally {

try {

in.close(); //关闭源文件输入流

out.close(); //关闭目标文件输出流

}catch(IOException e) {

e.printStackTrace();

}

}

}

}

## 19、用JDBC如何调用存储过程

代码如下：

package com.huawei.interview.lym;

import java.sql.CallableStatement;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

import java.sql.Types;

public class JdbcTest {

/\*\*

\* @param args

\*/

public static void main(String[] args) {

// TODO Auto-generated method stub

Connection cn = null;

CallableStatement cstmt = null;

try {

//这里最好不要这么干，因为驱动名写死在程序中了

Class.forName("com.mysql.jdbc.Driver");

//实际项目中，这里应用DataSource数据，如果用框架，

//这个数据源不需要我们编码创建，我们只需Datasource ds = context.lookup()

//cn = ds.getConnection();

cn = DriverManager.getConnection("jdbc:mysql:///test","root","root");

cstmt = cn.prepareCall("{call insert\_Student(?,?,?)}");

cstmt.registerOutParameter(3,Types.INTEGER);

cstmt.setString(1, "wangwu");

cstmt.setInt(2, 25);

cstmt.execute();

//get第几个，不同的数据库不一样，建议不写

System.out.println(cstmt.getString(3));

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

finally

{

/\*try{cstmt.close();}catch(Exception e){}

try{cn.close();}catch(Exception e){}\*/

try {

if(cstmt != null)

cstmt.close();

if(cn != null)

cn.close();

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

## 20、JDBC中的PreparedStatement相比Statement的好处

1）提高性能：在使用preparedStatement对象执行sql时候，命令被数据库编译和解析，然后被放到命令缓冲区，然后每当执行同一个preparedStatement时候，他就被再解析一次，但不会在编译，在缓冲区中可以发现预编译的命令，并且可以重新使用。

如果你要写Insert update delete 最好使用preparedStatement，在有大量用户的企业级应用软件中，经常会执行相同的sql,使用preparedStatement会增加整体的性能。

2）安全性：PreparedStatement可以防止sql注入。

## 21、写一个用jdbc连接实例。

package com.seecen.stream;

import java.sql.\*;

public class TestJDBC {

/\*\*

\* 1、实例话驱动类

\* 2、建立到数据库的连接

\* 3、将数据发送到数据库中

\* 4、执行语句（select语句）

\* 5、关闭

\* @param args

\*/

public static void main(String[] args) {

ResultSet rs = null;

Statement stmt = null;

Connection conn = null;

try {

Class.forName("oracle.jdbc.driver.OracleDriver");

conn = DriverManager.getConnection("jdbc:oracle:thin:@192.168.0.1:1521:yuewei", "scott", "tiger");

stmt = conn.createStatement();

rs = stmt.executeQuery("select \* from dept");

while(rs.next()) {

System.out.println(rs.getString("deptno"));

}

} catch (ClassNotFoundException e) {

e.printStackTrace();

} catch (SQLException e) {

e.printStackTrace();

} finally {

try {

if(rs != null) {

rs.close();

rs = null;

}

if(stmt != null) {

stmt.close();

stmt = null;

}

if(conn != null) {

conn.close();

conn = null;

}

} catch (SQLException e) {

e.printStackTrace();

}

}

}

}

## 22、ArrayList和Vector的区别？

这两个类都实现了List接口（List接口继承了Collection接口），他们都是有序集合，即存储在这两个集合中的元素的位置都是有顺序的，相当于一种动态的数组，我们以后可以按位置索引号取出某个元素，，并且其中的数据是允许重复的，这是HashSet之类的集合的最大不同处，HashSet之类的集合不可以按索引号去检索其中的元素，也不允许有重复的元素（本来题目问的与hashset没有任何关系，但为了说清楚ArrayList与Vector的功能，我们使用对比方式，更有利于说明问题）。

接着才说ArrayList与Vector的区别，这主要包括两个方面：.

（1）同步性：

Vector是线程安全的，也就是说是它的方法之间是线程同步的，而ArrayList是线程序不安全的，它的方法之间是线程不同步的。如果只有一个线程会访问到集合，那最好是使用ArrayList，因为它不考虑线程安全，效率会高些；如果有多个线程会访问到集合，那最好是使用Vector，因为不需要我们自己再去考虑和编写线程安全的代码。

备注：对于Vector&ArrayList、Hashtable&HashMap，要记住线程安全的问题，记住Vector与Hashtable是旧的，是java一诞生就提供了的，它们是线程安全的，ArrayList与HashMap是java2时才提供的，它们是线程不安全的。所以，我们讲课时先讲老的。

（2）数据增长：

ArrayList与Vector都有一个初始的容量大小，当存储进它们里面的元素的个数超过了容量时，就需要增加ArrayList与Vector的存储空间，每次要增加存储空间时，不是只增加一个存储单元，而是增加多个存储单元，每次增加的存储单元的个数在内存空间利用与程序效率之间要取得一定的平衡。Vector默认增长为原来两倍，而ArrayList的增长策略在文档中没有明确规定（从源代码看到的是增长为原来的1.5倍）。ArrayList与Vector都可以设置初始的空间大小，Vector还可以设置增长的空间大小，而ArrayList没有提供设置增长空间的方法。

总结：即Vector增长原来的一倍，ArrayList增加原来的0.5倍。

## 23、List、Set和Map的区别？

1）List和Set是Collection的子接口,map不是。

2）List的底层是数组的方式实现，Set是散列表的方式实现，map是键值对的方式。

3）list是有序可重复的，Set是无序不可重复的，map是有序，key不重复，value可重复

4）list和Set可直接使用itertator来进行遍历，map只能通过先遍历Key在遍历value.

## 24、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。

## 25、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，元素重复与否是使用equals()方法进行判断的。

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。

## 26、HashMap与HashTable的区别

1）继承不同

public class Hashtable extends Dictionary implements Map  
public class HashMap extends AbstractMap implements Map

2）Hashtable 中的方法是同步的，而HashMap中的方法在缺省情况下是非同步的。在多线程并发的环境下，可以直接使用Hashtable，但是要使用HashMap的话就要自己增加同步处理了。

3）Hashtable中，key和value都不允许出现null值，在HashMap中，null可以作为键，这样的键只有一个；可以有一个或多个键所对应的值为null。当get()方法返回null值时，即可以表示 HashMap中没有该键，也可以表示该键所对应的值为null。因此，在HashMap中不能由get()方法来判断HashMap中是否存在某个键， 而应该用containsKey()方法来判断。

4）两个遍历方式的内部实现上不同。

Hashtable、HashMap都使用了 Iterator。而由于历史原因，Hashtable还使用了Enumeration的方式 。

5）哈希值的使用不同，HashTable直接使用对象的hashCode。而HashMap重新计算hash值。

6）Hashtable和HashMap它们两个内部实现方式的数组的初始大小和扩容的方式。HashTable中hash数组默认大小是11，增加的方式是 old\*2+1。HashMap中hash数组的默认大小是16，而且一定是2的指数

## 27、Java中有多少种数据结构，分别是什么？

List：是列表，有下标值，存储元素可以重复，遍历元素是有序的。

Set：是散列集，无下标值，存储元素不可重复，遍历元素时无序的。

Map：是以键值对存储，一个key一个value，key不可以重复，value可以重复。

数组：指定类型，固定长度，元素存储地址是连续的。

树：元素以树形结构存储，只有一个根节点。

栈：元素是先进后出，后进先出。

向量：动态数组，可以存储任何类型元素，动态长度，元素存储地址是连续的。

队列：元素存储是排列有序的，一定保证先进的先出，后进的后出。

## 28、Arraylist 和linklist 的区别

相同点：  
ArrayList和Linklist都是接口List的实现类，里面的数据都是有序可重复的。

区别：

ArrayList: 采用的是数组形式保存对象的，访问速度更快，而Linklist的插入和删除元素的速度更快

## 29、List遍历方式有多少种

1. 下标遍历
2. Iterator遍历
3. Foreach遍历（最快）

## 30、Map怎么遍历

先调用[keySet](mk:@MSITStore:E:\\API\\JDK_API_1_6_zh_CN.CHM::/java/util/Map.html" \l "keySet())()方法获取所有的key，在遍历key获取所有的元素

## 31、怎么获取Map所有的key，所有的value

Map调用[keySet](mk:@MSITStore:E:\\API\\JDK_API_1_6_zh_CN.CHM::/java/util/Map.html" \l "keySet())()方法获取所有的key值，是一个Set集合

Map调用[values](mk:@MSITStore:E:\\API\\JDK_API_1_6_zh_CN.CHM::/java/util/Map.html" \l "values())()方法获取所有的value值，是一个List集合

## 32、获取Class的实例有几种方式

Class<?> demo1=Class.forName("Reflect.Demo"); //使用Class类

Class<?> demo2=new Demo().getClass(); //通过对象

Class<?> demo3=Demo.class; //通过类

## 33、怎么获取类中所有的方法，所有属性

获取所有方法：

Class<?> demo = Class.forName("Reflect.Demo");

[Method](mk:@MSITStore:E:\\API\\JDK_API_1_6_zh_CN.CHM::/java/lang/reflect/Method.html" \o "java.lang.reflect 中的类)[] methods = Demo. [getDeclaredMethods](mk:@MSITStore:E:\\API\\JDK_API_1_6_zh_CN.CHM::/java/lang/Class.html" \l "getDeclaredMethods())();

获取所有属性：

Class<?> demo = Class.forName("Reflect.Demo");

[Field](mk:@MSITStore:E:\\API\\JDK_API_1_6_zh_CN.CHM::/java/lang/reflect/Field.html" \o "java.lang.reflect 中的类)[] fields = demo .[getDeclaredFields](mk:@MSITStore:E:\\API\\JDK_API_1_6_zh_CN.CHM::/java/lang/Class.html" \l "getDeclaredFields())();

# 七、框架部分

## 1、谈谈你对Struts2的理解。

1)struts2是一个MVC框架，MVC是一种开发模式，把业务逻辑代码与视图代码分离，通过控制器连接业务逻辑与视图。MVC将应用程序分成了视图、模型、[控制器](https://www.baidu.com/s?wd=%E6%8E%A7%E5%88%B6%E5%99%A8&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YdnADzuWmznHmznA7-PHIh0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6K1TL0qnfK1TL0z5HD0IgF_5y9YIZ0lQzqlpA-bmyt8mh7GuZR8mvqVQL7dugPYpyq8Q1R1P104rjnLnf" \t "_blank)三部分，使代码结构层次清晰、降低耦合度、代码重用性高。

2）结合Struts2处理请求的工作流程加以说明：

客户端发送一个请求到服务器，tomcat会接收这个请求， tomcat会读取项目中的web.xml中的配置，判断请求是否符合Struts2过滤器StrutsPrepareAndExecuteFilter过滤的路径，如果符合会把这个请求交给Struts2处理，StrutsPrepareAndExecuteFilter会分析请求路径，根据Struts.xml中的配置，请求路径匹配package标签的namespace属性加上action标签的name属性，跳转到对应的action类，默认执行execute方法，如果使用动态方法调用，会执行action类中的对应方法，方法执行完成后会返回一个字符串，这个字符串对应Struts.xml中action标签下的result标签name属性根据result标签的配置跳转到对应的jsp页面，在jsp页面中呈现数据，返回给客户端。

3）结合Struts2优点说明：

a、实现MVC模式，结构清晰,使开发者只关注业务逻辑的实现.

b、有丰富的tag可以用 ,Struts的标记库(Taglib)，如能灵活动用，则能大大提高开发效率

c、页面导航使系统的脉络更加清晰。通过一个配置文件，即可把握整个系统各部分之间的联系，这对于后期的维护有着莫大的好处。尤其是当另一批开发者接手这个项目时，这种优势体现得更加明显。

d、提供Exception处理机制 .

e、数据库链接池管理

f、支持I18N

## 2、谈谈你对Hibernate的理解。

1）hibernate是ORM框架，ORM是对象关系映射,类—>表，属性—>表中的列,对象—>表中的每一条数据，是为了解决面向对象与关系数据库之间互不匹配现象的技术。使我们编程的思想更面向对象了，不用去考虑关系型数据库。

2）hibernate工作流程是：configuration类读取并解析hibernate.cfg.xml配置文件，读取解析映射信息，创建sessionFactory，打开session，创建事务，持久化操作，关闭session，整个应用停止关闭sessionFactory。

3）结合hibernate优点说明：

a、程序更加面向对象，提高开发效率

b、提高了生产率，不用写SQL语句

c、hibernate使用的是hql，支持方言配置，方便数据库移植

d、对jdbc代码进行封装，编程更简便了

e、hibernate是个轻量级框架，对代码无侵入性

## 3、你对Spring的理解。

1）Sping是一个轻量级框架，设计原则是非侵入性的。Sping核心是IOC容器，IOC是一种编程思想，是一种架构艺术，是用来管理控制对象的生命周期和对象之间的关系，通过配置文件进行注入，很好的实现了对象与对象之间解耦。

2）IOC工作原理： IOC实现了工厂模式，通过读取application.xml配置文件中的<bean>标签的类，注入到IOC容器中，通过构造或set方法注入，产生BeanFactory，BeanFactory通过getBean方法获取对象。

3）Spring还提供了另外一种重要编程思想AOP，AOP称为面向切面编程，可以动态的将主线业务逻辑代码与实现功能代码分离，为了更清晰的逻辑，可以让你的业务逻辑去关注自己本身的业务，而不去想一些其他的事情，将日志记录，性能统计，安全控制，事务处理，异常处理等代码从业务逻辑代码中划分出来，通过对这些行为的分离，我们希望可以将它们独立到非指导业务逻辑的方法中，进而改变这些行为的时候不影响业务逻辑的代码。

4）Spring提供了很多第三方框架的整合，如：hibernate、struts、mybatis、web service等，使用IOC管理所有的Java bean，这样可以让框架与框架之间偶尔度降低，方便项目的管理，提高开发效率。

## 4、Struts2优缺点

优点：

1）实现MVC模式，结构清晰,使开发者只关注业务逻辑的实现.

2）有丰富的tag可以用 ,Struts的标记库(Taglib)，如能灵活动用，则能大大提高开发效率

3）页面导航使系统的脉络更加清晰。通过一个配置文件，即可把握整个系统各部分之间的联系，这对于后期的维护有着莫大的好处。尤其是当另一批开发者接手这个项目时，这种优势体现得更加明显。

4）提供Exception处理机制 .

5）数据库链接池管理

6）支持I18N

缺点

1）转到展示层时，需要配置forward，每一次转到展示层，相信大多数都是直接转到jsp，而涉及到转向，需要配置forward，如果有十个展示层的jsp，需要配置十次struts，而且还不包括有时候目录、文件变更，需要重新修改forward，注意，每次修改配置之后，要求重新部署整个项目，而tomcate这样的服务器，还必须重新启动服务器，如果业务变更复杂频繁的系统，这样的操作简单不可想象。现在就是这样，几十上百个人同时在线使用我们的系统，大家可以想象一下，我的烦恼有多大。  
 2）Struts 的Action必需是thread－safe方式，它仅仅允许一个实例去处理所有的请求。所以action用到的所有的资源都必需统一同步，这个就引起了线程安全的问题。  
 3）测试不方便. Struts的每个Action都同W[eb](https://www.baidu.com/s?wd=EB&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YLuyPbnWR3uWT1PW61njNW0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6K1TL0qnfK1TL0z5HD0IgF_5y9YIZ0lQzqlpA-bmyt8mh7GuZR8mvqVQL7dugPYpyq8Q1RLPHfsP1nkrf" \t "_blank)层耦合在一起，这样它的测试依赖于Web容器，单元测试也很难实现。不过有一个Junit的扩展工具Struts TestCase可以实现它的单元测试。  
 4）类型的转换. Struts的FormBean把所有的数据都作为String类型，它可以使用工具Commons-Beanutils进行类型转化。但它的转化都是在[Class](https://www.baidu.com/s?wd=Class&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YLuyPbnWR3uWT1PW61njNW0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6K1TL0qnfK1TL0z5HD0IgF_5y9YIZ0lQzqlpA-bmyt8mh7GuZR8mvqVQL7dugPYpyq8Q1RLPHfsP1nkrf" \t "_blank)级别，而且转化的类型是不可配置的。类型转化时的错误信息返回给用户也是非常困难的。  
 5）对Servlet的依赖性过强. Struts处理Action时必需要依赖ServletRequest 和ServletResponse，所有它摆脱不了Servlet容器。  
 6）前端表达式语言方面.Struts集成了JSTL，所以它主要使用JSTL的表达式语言来获取数据。可是JSTL的表达式语言在Collection和索引属性方面处理显得很弱。  
 7）对Action执行的控制困难. Struts创建一个Action，如果想控制它的执行顺序将会非常困难。甚至你要重新去写Servlet来实现你的这个功能需求。  
 8）对Action 执行前和后的处理. Struts处理Action的时候是基于[class](https://www.baidu.com/s?wd=Class&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YLuyPbnWR3uWT1PW61njNW0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6K1TL0qnfK1TL0z5HD0IgF_5y9YIZ0lQzqlpA-bmyt8mh7GuZR8mvqVQL7dugPYpyq8Q1RLPHfsP1nkrf" \t "_blank)的hierarchies，很难在action处理前和后进行操作。  
 9）对事件支持不够. 在struts中，实际是一个表单Form对应一个Action类(或DispatchAction)，换一句话说：在Struts中实际是一个表单只能对应一个事件，struts这种事件方式称为application event，application event和component event相比是一种粗粒度的事件。

## 5、说说struts1与struts2的区别。

1）都是MVC的WEB框架,

2） struts1的老牌框架，应用很广泛，有很好的群众基础，使用它开发风险很小，成本更低！struts2虽然基于这个框架，但是应用群众并多，相对不成熟，未知的风险和变化很多，开发人员相对不好招，使用它开发项目的风险系数更大，用人成本更高！

3）struts2毕竟是站在前辈的基础设计出来，它会改善和完善struts1中的一些缺陷，struts1中一些悬而未决问题在struts2得到了解决。

4）struts1的前端控制器是一个Servlet，名称为ActionServlet，struts2的前端控制器是一个filter，在struts2.0中叫FilterDispatcher，在struts2.1中叫StrutsPrepareAndExecuteFilter。

5）struts1的action需要继承Action类，struts2的action可以不继承任何类；struts1对同一个路径的所有请求共享一个Action实例，struts2对同一个路径的每个请求分别使用一个独立Action实例对象，所有对于struts2的Action不用考虑线程安全问题。

6）在struts1中使用formbean封装请求参数，在struts2中直接使用action的属性来封装请求参数。

7）struts1中的多个业务方法放在一个Action中时（即继承DispatchAction时），要么都校验，要么都不校验；对于struts2，可以指定只对某个方法进行校验，当一个Action继承了ActionSupport且在这个类中只编写了validateXxx()方法，那么则只对Xxx()方法进行校验。

8）一个请求来了的执行流程进行分析，struts2是自动支持分模块开发，并可以不同模块设置不同的url前缀，这是通过package的namespace来实现的；struts2是支持多种类型的视图；struts2的视图地址可以是动态的，即视图的名称是支持变量方式的，举例，论坛发帖失败后回来还要传递boardid。视图内容显示方面：它的标签用ognl，要el强大很多，在国际化方面支持分模块管理，两个模块用到同样的key，对应不同的消息。

9）与Struts1不同，Struts2对用户的每一次请求都会创建一个Action，所以Struts2中的Action是线程安全的。

10）给我印象最深刻的是：struts配置文件中的redirect视图的url不能接受参数，而struts2配置文件中的redirect视图可以接受参数。

## 6、struts的核心组件有哪些？

1）FilterDispatcher,struts2的核心组件，整个struts2的调度中心，它对请求进行过滤并决定struts2是否出来该请求。

2）Strtus.xml：struts2的应用配置文件，它负责配置系统中用到的action

3）Action:strtus2的动作执行单元实际处理用户的请求，封装业务所需的数据

4）Result：action运行后要转向下一个资源，可以是视图也可以说其他的action

5）Struts标签：用于页面上遍历后台传过来的数据

## 7、Strus2的执行过程

1）客户端发送一个请求到服务器，tomcat会接收这个请求

2）tomcat会读取项目中的web.xml中的配置

3）判断请求是否符合Struts2过滤器StrutsPrepareAndExecuteFilter过滤的路径

4）如果符合会把这个请求交给Struts2处理

5）StrutsPrepareAndExecuteFilter会分析请求路径，根据Struts.xml中的配置，请求路径匹配package标签的namespace属性加上action标签的name属性，跳转到对应的action类

6）默认执行execute方法，如果使用动态方法调用，会执行action类中的对应方法，方法执行完成后会返回一个字符串

7）这个字符串对应Struts.xml中action标签下的result标签name属性

8）根据result标签的配置跳转到对应的jsp页面，在jsp页面中呈现数据，返回给客户端

## 8、为什么要使用struts2？

1）开源

2）mvc框架

3）纯pojo的action

4）更好的标签特性

5）易测性

6）易扩展性

## 9、openSession和getCurrentSession

 1）openSession 从字面上可以看得出来，是打开一个新的session对象，而且每次使用都是打开一个新的session，假如连续使用多次，则获得的session不是同一个对象，并且使用完需要调用close方法关闭session。

  2）getCurrentSession ，从字面上可以看得出来，是获取当前上下文一个session对象，当第一次使用此方法时，会自动产生一个session对象，并且连续使用多次时，得到的session都是同一个对象，这就是与openSession的区别之一，简单而言，getCurrentSession 就是：如果有已经使用的，用旧的，如果没有，建新的。

注意 ：在实际开发中，往往使用getCurrentSession多，因为一般是处理同一个事务（即是使用一个数据库的情况），所以在一般情况下比较少使用openSession或者说openSession是比较老旧的一套接口了；

对于getCurrentSession 来说，有以下一些特点：

1.用途，界定事务边界

2.事务提交会自动close，不需要像openSession一样自己调用close方法关闭session

3.上下文配置（即在hibernate.cfg.xml）中，需要配置：

    <property name="current\_session\_context\_class">thread</property>

## 10、拦截器的作用？拦截器和过滤器的区别？

拦截器是对调用的action起作用，它提供类一种机制可以使开发者可以定义在一个action执行的前后执行的代码。拦截器只能拦截action，说白了拦截器其实就是一个action的功能块。拦截器可以抽象出一部分代码可以用来完善原来的action。同时可以减轻代码冗余提高重用率。

过滤器是拦截用户请求，范围被拦截器大。

## 11、struts.xml中result的type有哪些类型？

Dispatcher：struts2默认的结果类型，把控制权转发给应用程序里的某个资源，不能把控制权转发给一个外部资源，若需要啊控制权重定向到一个外部资源，应该使用redirect结果类型。

Redirect 把响应重定向到另一个资源

RedirectAction 把响应重定向到另一个Action

Freemarcker、velocity、chain、httpherder、xslt、plainText、stream、json.

## 12、一般情况下，关系数据模型与对象模型之间有哪些匹配关系?

表对应类

记录对应对象

表的字段对应类的属性

## 13、hibernate 数据的三个状态

1）瞬时状态（临时状态）：当new 对象时候，处于瞬时状态（如果程序运行完了，该对象会被垃圾回收）。

2）持久状态 ：跟session有关，就是持久状态，持久状态的对象，任何的修改，都会影响到数据库中与之对应的数据。

3）托管状态（游离状态）：当session不在管理对象的时候，脱离了 session 的管理，处于托管状态的对象，修改属性，对数据库数据没有任何影响。

企业开发中,使用saveOrUpdate(obj): 来替代save(obj)或update(obj)方法

避免因为状态的改变,导致方法出错, saveOrUpdate(obj)

可以根据obj的状态,来选择是save()还是update()

## 14、Hibernate中load和get的区别？

如果数据库中，没有userId的对象，如果通过get方法加载，则返回的是一个Null；如果通过Load则返回一个代理对象，如果后面代码调用user对象的某个属性,会抛出objectNotFoundException

Load支持延迟加载，get不支持。

## 15、Hibernate的工作原理?

1）configuration类读取并解析hibernate.cfg.xml配置文件

2）读取解析映射信息，创建sessionFactory

3）打开session

4）创建事务

5）持久化操作

6）关闭session

7）整个应用停止，关闭sessionFactory

## 16、hibernate优缺点？

优点：

1）对jdbc访问数据库的代码做了封装，大大简化了数据访问层繁琐的重复性代码。

2）Hibernate是一个基于JDBC的主流持久性框架，是一个优秀的ORM实现，他很大程度的简化DAO的编码工作，程序更加面向对象，提高开发效率。

3）程序更加面向对象，提高开发效率

4）提高了生产率，不用写SQL语句

5）hibernate使用的是hql，支持方言配置，方便数据库移植

6）hibernate是个轻量级框架，对代码无侵入性

缺点：

1）效率比JDBC略差

2）不适合批量操作

3）对表的操作不够灵活

## 17、Hibernate是如何延迟加载的？

hibernate中存在一些查询方法，在查询的时候并没有立刻访问数据库查询数据，而是返回了一个空对象，这个对象并不是null而是经过new的对象，但对象中除了ID这种属性外其他属性都是null，当程序使用对象时hibernate才会真正的发送语句去查询数据库，将返回的数据填充到对象的属性值。这种将推迟查询队形机制称为延迟加载。

为什么要用延迟加载：

1）推迟的时间内由于没有数据加载可以节约内存空间，提高内存的使用率。

2）如果对象查询出来并没有使用，那么延迟加载的对象根本没有访问数据库，可以减少数据可得访问次数。

如何使用延迟加载

1）在hibernate里面有一些方法自动支持延迟加载，只要调用就可以使用。

2）具有延迟加载的方法如下:

session.load();

query.iterate();

关联映射属性加载,属性名是lazy，如果查询不存在延迟加载就会抛异常

## 18、如果优化Hibernate？

使用双向一对多关联，不使用单向一对多

灵活使用单向一对多

不使用一对一，用多对一取代

配置对象缓存，不适用集合缓存

一对多集合使用bag，多对多使用set

继承类使用显式多态

表字段要少，表关联不要怕多，有二级缓存。

## 19、什么是ORM？

ORM是对象关系映射,类—>表，属性—>表中的列,对象—>表中的每一条数据,

是为了解决面向对象与关系数据库之间互不匹配现象的技术。

优点：使我们编程的思想更面向对象了，不用去考虑关系型数据库

## 20、Hibernate的主键生成策略？

1）sequence,通知Hibernate框架，主键的值采用指定序列生成，然后插入数据库，主要用于

Oracle,DB2,不用程序员参与

<generator class="sequence">

<param name="sequence">foo\_seq</param>// 必须加上

</generator>

2）identity,通知hibernate框架，主键值采用数据库自动增长机制，每次进行save()操作，hibernate都会根据（数据库）自增的方式，生成一个id值，不用程序员参与，主要用于

mySQL , SQLServer

<generator class="identity"></generator>

3）uuid(西方常用),hibernate 每次进行save()操作，都会随机生成一个32的不重复的字符串，不用程序员去参与维护，PO类的Id属性必须为String

4）native 根据dialect(方言)不同,来自动的选择identity或sequence智能选择。是企业中常用的

5）assigned 不推荐使用，程序言要自己维护主键的Id值，当数据量很大时候很难维护

## 21、Hibernate的级联操作

1) cascade操作

all:所有情况下都进行级练操作 ，save-update和delete

save-update: 在进行save()/update()/saveOrUpdate时候进行级练操作

delete:在进行delete时候进行级练操作

all-delete-orphan :适合集合中删除，在返回的集合中执行remove()操作

none:在任何情况下都不进行级练操作

2) inverse属性的作用

是否放弃维护关联关系 true放弃 false不放弃

## 22、Hibernate有哪5个核心接口？

Configuration接口：配置Hibernate，根据其启动hibernate，创建SessionFactory对象；

SessionFactory接口：初始化Hibernate，充当数据存储源的代理，创建session对象，sessionFactory是线程安全的，意味着它的同一个实例可以被应用的多个线程共享，是重量级、二级缓存；

Session接口：负责保存、更新、删除、加载和查询对象，是线程不安全的，避免多个线程共享同一个session，是轻量级、一级缓存；

Transaction接口：管理事务；

Query和Criteria接口：执行数据库的查询。

## 23、什么是重量级？什么是轻量级？

轻量级是指它的创建和销毁不需要消耗太多的资源，意味着可以在程序中经常创建和销毁session的对象；重量级意味不能随意的创建和销毁它的实例，会占用很多的资源。

## 24、谈谈Spring的IOC和DI

首先想说说IoC（Inversion of Control，控制倒转）。这是spring的核心，贯穿始终。所谓IoC，对于spring框架来说，就是由spring来负责控制对象的生命周期和对象间的关系。这是什么意思呢，举个简单的例子，我们是如何找女朋友的？常见的情况是，我们到处去看哪里有长得漂亮身材又好的mm，然后打听她们的兴趣爱好、qq号、电话号、ip号、iq号………，想办法认识她们，投其所好送其所要，然后嘿嘿……这个过程是复杂深奥的，我们必须自己设计和面对每个环节。传统的程序开发也是如此，在一个对象中，如果要使用另外的对象，就必须得到它（自己new一个，或者从JNDI中查询一个），使用完之后还要将对象销毁（比如Connection等），对象始终会和其他的接口或类藕合起来。

那么IoC是如何做的呢？有点像通过婚介找女朋友，在我和女朋友之间引入了一个第三者：婚姻介绍所。婚介管理了很多男男女女的资料，我可以向婚介提出一个列表，告诉它我想找个什么样的女朋友，比如长得像李嘉欣，身材像林熙雷，唱歌像周杰伦，速度像卡洛斯，技术像齐达内之类的，然后婚介就会按照我们的要求，提供一个mm，我们只需要去和她谈恋爱、结婚就行了。简单明了，如果婚介给我们的人选不符合要求，我们就会抛出异常。整个过程不再由我自己控制，而是有婚介这样一个类似容器的机构来控制。Spring所倡导的开发方式就是如此，所有的类都会在spring容器中登记，告诉spring你是个什么东西，你需要什么东西，然后spring会在系统运行到适当的时候，把你要的东西主动给你，同时也把你交给其他需要你的东西。所有的类的创建、销毁都由 spring来控制，也就是说控制对象生存周期的不再是引用它的对象，而是spring。对于某个具体的对象而言，以前是它控制其他对象，现在是所有对象都被spring控制，所以这叫控制反转。如果你还不明白的话，我决定放弃。

IoC的一个重点是在系统运行中，动态的向某个对象提供它所需要的其他对象。这一点是通过DI（Dependency Injection，依赖注入）来实现的。比如对象A需要操作数据库，以前我们总是要在A中自己编写代码来获得一个Connection对象，有了 spring我们就只需要告诉spring，A中需要一个Connection，至于这个Connection怎么构造，何时构造，A不需要知道。在系统运行时，spring会在适当的时候制造一个Connection，然后像打针一样，注射到A当中，这样就完成了对各个对象之间关系的控制。A需要依赖 Connection才能正常运行，而这个Connection是由spring注入到A中的，依赖注入的名字就这么来的。那么DI是如何实现的呢？ Java 1.3之后一个重要特征是反射（reflection），它允许程序在运行的时候动态的生成对象、执行对象的方法、改变对象的属性，spring就是通过反射来实现注入的。

总结：IOC是用来管理控制对象的生命周期和对象之间的关系，通过配置文件进行注入，很好的实现了对象与对象之间解耦。

IOC工作原理：

IOC实现了工厂模式，通过读取application.xml配置文件中的<bean>标签的类，

注入到IOC容器中，通过构造或set方法注入，产生BeanFactory，

BeanFactory通过getBean方法获取对象。

## 25、什么是AOP？

Aspect Oriented Programming (面向方面编程)

OOP是面向对象编程,AOP是在OOP基础之上一种更高级的设计思想.

OOP和AOP之间也存在一些区别,OOP侧重于对象的提取和封装.

AOP侧重于方面组件,方面组件可以理解成封装了通用功能的组件,

方面组件可以通过配置方式灵活的切入到某一批目标对象方法上.

aop是面向切面编程，可以动态的将主线业务逻辑代码与实现功能代码分离，没有侵入性。为了更清晰的逻辑，可以让你的业务逻辑去关注自己本身的业务，而不去想一些其他的事情，将日志记录，性能统计，安全控制，事务处理，异常处理等代码从业务逻辑代码中划分出来，通过对这些行为的分离，我们希望可以将它们独立到非指导业务逻辑的方法中，进而改变这些行为的时候不影响业务逻辑的代码。

## 26、Spring的通知类型有哪些？

通知决定了方面组件功能在目标对象方法上执行的时机.

Spring框架提供了以下5中类型通知.

1）.前置通知<aop:before>

方面功能在目标方法之前调用.

2）后置通知<aop:afterReturning>

方面功能在目标方法之后调用.(如果目标方法抛出异常则不会执行方面功能)

3）最终通知<aop:after>

方面功能在目标方法之后调用.(目标方法有无异常都会执行方面功能)

4）环绕通知<aop:around>

方面功能在目标方法之前和之后调用.

5）异常通知<aop:afterThrowing>

方面功能在目标方法抛出异常之后调用.

## 27、什么是MVC？

MVC是一种开发模式，把业务逻辑代码与视图代码分离，通过控制器连接业务逻辑与视图。

MVC将一个应用程序分为三个部分：

Model：业务逻辑的处理。

View：代表和用户交互的界面。

Controller：控制器，连接模型层与视图层

优点:

1)代码结构层次清晰

2)就是低耦合

3)重用性高

缺点:

一个应用程序分成了三个部分开发，增加开发工作量。

## 28、hibernate查询方式有多少种？

主键查找：session.get()或load()

hql查询：session.createQuery(“hql”)

sql查询：session.createSQLQuery(“sql”)

criteria查询（QBC）: session.createCriteria()

## 29、spring中Bean的scope

目前，scope的取值有5种。

在Spring 2.0之前，有singleton和prototype两种

在Spring 2.0之后，为支持web应用的ApplicationContext，推出另外三种：request，session和global session类型

singleton：在IOC容器中只存在一个实例

prototype：在IOC容器中只存在多个实例

request：使用在web应用中，相当于Servlet中的Request

session：使用在web应用中，相当于Servlet中的Session

global session：使用在web应用中，相当于Servlet中的application

## 30、SSH对应MVC的哪些层

Struts2：用于处理请求，调用业务逻辑

Hibernate：用于操作数据库，做持久化操作

Spring：用于管理对象，处理对象与对象之间的关系

MVC是一种开发模式，模型、视图、控制，与SSH框架的作用是两个东西，不能相互对应。

## 31、spring注入方式有几种

Spring四种依赖注入方式，常用1、2种，

1）Set方法注入

2）构造器注入

3）静态工厂方法注入

4）实例工厂方法注入

# 八、XML部分

## 1、xml有哪些解析技术?区别是什么?

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?

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、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>

## 4、XML文档定义有几种形式？它们之间有何本质区别？

1）两种形式 dtd schema。

2）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)，

## 5、XML和HTML的区别？

1）设计上的区别：XML用来存储数据，重点在于数据本身，HTML用来定义数据，重在数据的显示模式。

2）XML可扩展性强，因为他本身就是可拓展性标记语言，课创建个性化的标记语言，提供更多数据操作。

3）XML语法比HTML严格。

4）起始标签和结束标签要匹配

5）嵌套标签不能相互嵌套

6）区分大小写

7）XML属性必须放在引号中，HTML可有可无。

8）XML必须有相应值，但HTML可以有不带属性的属性名。

## 6、XML文件和普通文件相比有哪些优点？

1）XML是一个标准的技术，在处理上可以更加的统一

2）从对配置文件的后期处理上，对XML的操作的API更多，更方便，而文本文件不是特别方便。

3）XML可以有定义语法的DTD文件，这样兑取配置信息的时候，可以先根据DTD检查当前的XML配置文件是否语法正确，而文本文件没有办法检查语法。

4）XML可以非常方便的转换成其他格式的文件，而文本不可以。

5）XML利用xslt可以非常方便的转换成其他格式的文件，文本文件很难做到。

6）XML可以非常方便的搜索其中的一些配置信息，试想如果配置文件很大，优点还是很明显的，而文本文件则不太方便。

7）XML可以携带很多、更丰富的配置信息，文本文件不容易做到

# 九、设计模式部分

**请写出你所知道的设计模式？**

设计模式主要分三个类型:创建型、结构型和行为型。

**其中创建型有：**

一、Singleton，单例模式：保证一个类只有一个实例，并提供一个访问它的全局访问点

二、Abstract Factory，抽象工厂：提供一个创建一系列相关或相互依赖对象的接口，而无须指定它们的具体类。

三、Factory Method，工厂方法：定义一个用于创建对象的接口，让子类决定实例化哪一个类，Factory Method使一个类的实例化延迟到了子类。

四、Builder，建造模式：将一个复杂对象的构建与他的表示相分离，使得同样的构建过程可以创建不同的表示。

五、Prototype，原型模式：用原型实例指定创建对象的种类，并且通过拷贝这些原型来创建新的对象。

**行为型有：**

六、Iterator，迭代器模式：提供一个方法顺序访问一个聚合对象的各个元素，而又不需要暴露该对象的内部表示。

七、Observer，观察者模式：定义对象间一对多的依赖关系，当一个对象的状态发生改变时，所有依赖于它的对象都得到通知自动更新。

八、Template Method，模板方法：定义一个操作中的算法的骨架，而将一些步骤延迟到子类中，TemplateMethod使得子类可以不改变一个算法的结构即可以重定义该算法得某些特定步骤。

九、Command，命令模式：将一个请求封装为一个对象，从而使你可以用不同的请求对客户进行参数化，对请求排队和记录请求日志，以及支持可撤销的操作。

十、State，状态模式：允许对象在其内部状态改变时改变他的行为。对象看起来似乎改变了他的类。

十一、Strategy，策略模式：定义一系列的算法，把他们一个个封装起来，并使他们可以互相替换，本模式使得算法可以独立于使用它们的客户。

十二、China of Responsibility，职责链模式：使多个对象都有机会处理请求，从而避免请求的送发者和接收者之间的耦合关系

十三、Mediator，中介者模式：用一个中介对象封装一些列的对象交互。

十四、Visitor，访问者模式：表示一个作用于某对象结构中的各元素的操作，它使你可以在不改变各元素类的前提下定义作用于这个元素的新操作。

十五、Interpreter，解释器模式：给定一个语言，定义他的文法的一个表示，并定义一个解释器，这个解释器使用该表示来解释语言中的句子。

十六、Memento，备忘录模式：在不破坏对象的前提下，捕获一个对象的内部状态，并在该对象之外保存这个状态。

**结构型有：**

十七、Composite，组合模式：将对象组合成树形结构以表示部分整体的关系，Composite使得用户对单个对象和组合对象的使用具有一致性。

十八、Facade，外观模式：为子系统中的一组接口提供一致的界面，fa?ade提供了一高层接口，这个接口使得子系统更容易使用。

十九、Proxy，代理模式：为其他对象提供一种代理以控制对这个对象的访问

二十、Adapter,适配器模式：将一类的接口转换成客户希望的另外一个接口，Adapter模式使得原本由于接口不兼容而不能一起工作那些类可以一起工作。

二十一、Decrator，装饰模式：动态地给一个对象增加一些额外的职责，就增加的功能来说，Decorator模式相比生成子类更加灵活。

二十二、Bridge，桥模式：将抽象部分与它的实现部分相分离，使他们可以独立的变化。

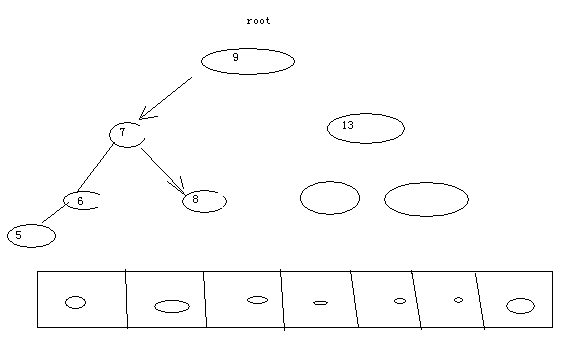
二十三、Flyweight，享元模式

# 十、算法部分

**1、说明生活中遇到的二叉树，用java实现二叉树**

这是组合设计模式。

我有很多个(假设10万个)数据要保存起来，以后还需要从保存的这些数据中检索是否存在某个数据，（我想说出二叉树的好处，该怎么说呢？那就是说别人的缺点），假如存在数组中，那么，碰巧要找的数字位于99999那个地方，那查找的速度将很慢，因为要从第1个依次往后取，取出来后进行比较。平衡二叉树（构建平衡二叉树需要先排序，我们这里就不作考虑了）可以很好地解决这个问题，但二叉树的遍历（前序，中序，后序）效率要比数组低很多，原理如下图：



代码如下：

package com.huawei.interview;

public class Node {

public int value;

public Node left;

public Node right;

public void store(int value)

{

if(value<this.value)

{

if(left == null)

{

left = new Node();

left.value=value;

}

else

{

left.store(value);

}

}

else if(value>this.value)

{

if(right == null)

{

right = new Node();

right.value=value;

}

else

{

right.store(value);

}

}

}

public boolean find(int value)

{

System.out.println("happen " + this.value);

if(value == this.value)

{

return true;

}

else if(value>this.value)

{

if(right == null) return false;

return right.find(value);

}else

{

if(left == null) return false;

return left.find(value);

}

}

public void preList()

{

System.out.print(this.value + ",");

if(left!=null) left.preList();

if(right!=null) right.preList();

}

public void middleList()

{

if(left!=null) left.preList();

System.out.print(this.value + ",");

if(right!=null) right.preList();

}

public void afterList()

{

if(left!=null) left.preList();

if(right!=null) right.preList();

System.out.print(this.value + ",");

}

public static void main(String [] args)

{

int [] data = new int[20];

for(int i=0;i<data.length;i++)

{

data[i] = (int)(Math.random()\*100) + 1;

System.out.print(data[i] + ",");

}

System.out.println();

Node root = new Node();

root.value = data[0];

for(int i=1;i<data.length;i++)

{

root.store(data[i]);

}

root.find(data[19]);

root.preList();

System.out.println();

root.middleList();

System.out.println();

root.afterList();

}

}

-----------------又一次临场写的代码---------------------------

import java.util.Arrays;

import java.util.Iterator;

public class Node {

private Node left;

private Node right;

private int value;

//private int num;

public Node(int value){

this.value = value;

}

public void add(int value){

if(value > this.value)

{

if(right != null)

right.add(value);

else

{

Node node = new Node(value);

right = node;

}

}

else{

if(left != null)

left.add(value);

else

{

Node node = new Node(value);

left = node;

}

}

}

public boolean find(int value){

if(value == this.value) return true;

else if(value > this.value){

if(right == null) return false;

else return right.find(value);

}else{

if(left == null) return false;

else return left.find(value);

}

}

public void display(){

System.out.println(value);

if(left != null) left.display();

if(right != null) right.display();

}

/\*public Iterator iterator(){

}\*/

public static void main(String[] args){

int[] values = new int[8];

for(int i=0;i<8;i++){

int num = (int)(Math.random() \* 15);

//System.out.println(num);

//if(Arrays.binarySearch(values, num)<0)

if(!contains(values,num))

values[i] = num;

else

i--;

}

System.out.println(Arrays.toString(values));

Node root = new Node(values[0]);

for(int i=1;i<values.length;i++){

root.add(values[i]);

}

System.out.println(root.find(13));

root.display();

}

public static boolean contains(int [] arr, int value){

int i = 0;

for(;i<arr.length;i++){

if(arr[i] == value) return true;

}

return false;

}

## 2、第1个人10，第2个比第1个人大2岁，依次递推，请用递归方式计算出第8个人多大？

package cn.itcast;

import java.util.Date;

public class A1 {

public static void main(String [] args)

{

System.out.println(computeAge(8));

}

public static int computeAge(int n)

{

if(n==1) return 10;

return computeAge(n-1) + 2;

}

}

public static void toBinary(int n,StringBuffer result)

{

if(n/2 != 0)

toBinary(n/2,result);

result.append(n%2);

}

## 3、排序都有哪几种方法？请列举。用JAVA实现一个快速排序。

public class QuickSort {

/\*\*

\* 快速排序

\* @param strDate

\* @param left

\* @param right

\*/

public void quickSort(String[] strDate,int left,int right){

String middle,tempDate;

int i,j;

i=left;

j=right;

middle=strDate[(i+j)/2];

do{

while(strDate[i].compareTo(middle)<0&& i<right)

i++; //找出左边比中间值大的数

while(strDate[j].compareTo(middle)>0&& j>left)

j--; //找出右边比中间值小的数

if(i<=j){ //将左边大的数和右边小的数进行替换

tempDate=strDate[i];

strDate[i]=strDate[j];

strDate[j]=tempDate;

i++;

j--;

}

}while(i<=j); //当两者交错时停止

if(i<right){

quickSort(strDate,i,right);//从

}

if(j>left){

quickSort(strDate,left,j);

}

}

/\*\*

  \* @param args

  \*/

public static void main(String[] args){

String[] strVoid=new String[]{"11","66","22","0","55","22","0","32"};

QuickSort sort=new QuickSort();

sort.quickSort(strVoid,0,strVoid.length-1);

for(int i=0;i<strVoid.length;i++){

System.out.println(strVoid[i]+" ");

}

}

}

## 4、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。

去零的代码：

return sb.reverse().toString().replaceAll("零[拾佰仟]","零").replaceAll("零+万","万").replaceAll("零+元","元").replaceAll("零+","零");

public class RenMingBi {

/\*\*

\* @param args add by zxx ,Nov 29, 2008

\*/

private static final char[] data = new char[]{

'零','壹','贰','叁','肆','伍','陆','柒','捌','玖'

};

private static final char[] units = new char[]{

'元','拾','佰','仟','万','拾','佰','仟','亿'

};

public static void main(String[] args) {

// TODO Auto-generated method stub

System.out.println(

convert(135689123));

}

public static String convert(int money)

{

StringBuffer sbf = new StringBuffer();

int unit = 0;

while(money!=0)

{

sbf.insert(0,units[unit++]);

int number = money%10;

sbf.insert(0, data[number]);

money /= 10;

}

return sbf.toString();

}

}

## 5、从类似如下的文本文件中读取出所有的姓名，并打印出重复的姓名和重复的次数，并按重复次数排序：

1,张三,28

2,李四,35

3,张三,28

4,王五,35

5,张三,28

6,李四,35

7,赵六,28

8,田七,35

程序代码如下（答题要博得用人单位的喜欢，包名用该公司，面试前就提前查好该公司的网址，如果查不到，现场问也是可以的。还要加上实现思路的注释）：

package com.huawei.interview;

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStream;

import java.io.InputStreamReader;

import java.util.Comparator;

import java.util.HashMap;

import java.util.Iterator;

import java.util.Map;

import java.util.TreeSet;

public class GetNameTest {

/\*\*

\* @param args

\*/

public static void main(String[] args) {

// TODO Auto-generated method stub

//InputStream ips = GetNameTest.class.getResourceAsStream("/com/huawei/interview/info.txt");

//用上一行注释的代码和下一行的代码都可以，因为info.txt与GetNameTest类在同一包下面，所以，可以用下面的相对路径形式

Map results = new HashMap();

InputStream ips = GetNameTest.class.getResourceAsStream("info.txt");

BufferedReader in = new BufferedReader(new InputStreamReader(ips));

String line = null;

try {

while((line=in.readLine())!=null)

{

dealLine(line,results);

}

sortResults(results);

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

static class User

{

public String name;

public Integer value;

public User(String name,Integer value)

{

this.name = name;

this.value = value;

}

@Override

public boolean equals(Object obj) {

// TODO Auto-generated method stub

//下面的代码没有执行，说明往treeset中增加数据时，不会使用到equals方法。

boolean result = super.equals(obj);

System.out.println(result);

return result;

}

}

private static void sortResults(Map results) {

// TODO Auto-generated method stub

TreeSet sortedResults = new TreeSet(

new Comparator(){

public int compare(Object o1, Object o2) {

// TODO Auto-generated method stub

User user1 = (User)o1;

User user2 = (User)o2;

/\*如果compareTo返回结果0，则认为两个对象相等，新的对象不会增加到集合中去

\* 所以，不能直接用下面的代码，否则，那些个数相同的其他姓名就打印不出来。

\* \*/

//return user1.value-user2.value;

//return user1.value<user2.value?-1:user1.value==user2.value?0:1;

if(user1.value<user2.value)

{

return -1;

}else if(user1.value>user2.value)

{

return 1;

}else

{

return user1.name.compareTo(user2.name);

}

}

}

);

Iterator iterator = results.keySet().iterator();

while(iterator.hasNext())

{

String name = (String)iterator.next();

Integer value = (Integer)results.get(name);

if(value > 1)

{

sortedResults.add(new User(name,value));

}

}

printResults(sortedResults);

}

private static void printResults(TreeSet sortedResults)

{

Iterator iterator = sortedResults.iterator();

while(iterator.hasNext())

{

User user = (User)iterator.next();

System.out.println(user.name + ":" + user.value);

}

}

public static void dealLine(String line,Map map)

{

if(!"".equals(line.trim()))

{

String [] results = line.split(",");

if(results.length == 3)

{

String name = results[1];

Integer value = (Integer)map.get(name);

if(value == null) value = 0;

map.put(name,value + 1);

}

}

}

}

## 6、写一个Singleton出来。

第一种：饱汉模式

public class SingleTon {

private SingleTon(){

}

//实例化放在静态代码块里可提高程序的执行效率，但也可能更占用空间

private final static SingleTon instance = new SingleTon();

public static SingleTon getInstance(){

return instance;

}

}

第二种：饥汉模式

public class SingleTon {

private SingleTon(){}

private static instance = null;//new SingleTon();

public static synchronized SingleTon getInstance(){

if(instance == null)

instance = new SingleTon();

return instance;

}

}

第三种：用枚举

public enum SingleTon{

ONE;

}

第三：更实际的应用（在什么情况用单例）

public class SequenceGenerator{

//下面是该类自身的业务功能代码

private int count = 0;

public synchronized int getSequence(){

++count;

}

//下面是把该类变成单例的代码

private SequenceGenerator(){}

private final static instance = new SequenceGenerator();

public static SingleTon getInstance(){

return instance;

}

}

第四：

public class MemoryDao

{

private HashMap map = new HashMap();

public void add(Student stu1){

map.put(SequenceGenerator.getInstance().getSequence(),stu1);

}

//把MemoryDao变成单例

}

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

## 7、古典问题：有一对兔子，从出生后第3个月起每个月都生一对兔子，小兔子长到第三个月后每个月又生一对兔子，假如兔子都不死，问每个月的兔子总数为多少？

这是一个菲波拉契数列问题  
public class lianxi01 {  
public static void main(String[] args) {  
System.out.println("第1个月的兔子对数:    1");  
System.out.println("第2个月的兔子对数:    1");  
int f1 = 1, f2 = 1, f, M=24;  
     for(int i=3; i<=M; i++) {  
      f = f2;  
      f2 = f1 + f2;  
      f1 = f;  
      System.out.println("第" + i +"个月的兔子对数: "+f2);  
         }  
}  
}

## 8、简单的说个递归

N阶乘：

public int factorial(int m)  
 {  
 if (m < 0)  
 return 0;  
 else if ( m == 1)  
 reteurn 1;  
 else if (m > 1)  
 return m \* factorial(m-1);  
 }

## 9、什么是平衡二叉树

平衡二叉树是一棵空树或它的左右两个子树的高度差的绝对值不超过1，并且左右两个子树都是一棵平衡二叉树。

## 10、怎么判断二叉树是否有环

在遍历二叉树时，能循环到起点指针称为有环。