

# 面霸

杭州教学支持中心笔面试题汇总

文档编号  
Version 3.0

2013-01  
达内 IT 培训集团

### **杭州达内计算机有限公司专有信息声明**

本文档中所有信息均为达内公司机密，未经版权限定及达内公司明确的书面许可，任何公司，组织和个人不得为任何目的、以任何手段或形式对本文档的任何部分或全部进行复制，存储，引入检索系统或传播。虽然达内公司已经尽最大努力使本文档内容正确有效，但仍然可能有某些技术方面不够准确或存在印刷错误。这些信息将会被不定期的更新，更新的内容将包含在本文档的未来版本中。

“达内”、“Tarena”、“达内软件”、“达内计算机”、“达内信息” 等均是杭州达内计算机有限公司的注册商标。本文档中提及的其他公司、产品或服务名称可能是其他公司的商标或服务标志。

“杭州达内计算机有限公司”保留最终解释权，任何侵权行为，将以法律形式解决。

## 目录

第 1 章 文档概述.....	17
1.1 编写目的.....	17
1.2 主要读者.....	17
1.3 版本修订.....	17
第 2 章 产品概述.....	18
2.1 关于杭州教学支持中心.....	18
2.2 关于产品.....	19
第 3 章 面霸-PHP.....	21
PHP 经典面试题.....	190
第 4 章 面霸 - Android.....	230
第 5 章 面霸 - JAVA.....	254
1. Java 基础部分.....	271
1、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?.....	271
2、Java 有没有 goto?.....	271
3、说说&和&&的区别。.....	271
4、在 JAVA 中如何跳出当前的多重嵌套循环?.....	271
5、JDK1.6 的环境, switch 语句能否作用在 byte 上, 能否作用在 long 上, 能否作用在 String 上?.....	272
6、short s1 = 1; s1 = s1 + 1;有什么错? short s1 = 1; s1 += 1;有什么错?.....	272
7、char 型变量中能不能贮存一个中文汉字?为什么?.....	272
8、用最有效率的方法算出 2 乘以 8 等於几?.....	272
9、请设计一个一百亿的计算器.....	272
10、使用 final 关键字修饰一个变量时, 是引用不能变, 还是引用的对象不能变?.....	273
11、"=="和 equals 方法究竟有什么区别?.....	274
12、静态变量和实例变量的区别?.....	274

---

13、是否可以从一个 static 方法内部发出对非 static 方法的调用？.....	275
14、Integer 与 int 的区别.....	275
15、Math.round(11.5) 等於多少? Math.round(-11.5) 等於多少?.....	275
16、下面的代码有什么不妥之处?.....	275
17、请说出作用域 public , private , protected , 以及不写时的区别.....	275
18、Overload 和 Override 的区别。Overloaded 的方法是否可以改变返回值的类型?.....	276
19、构造器 Constructor 是否可被 override?.....	276
20、接口是否可继承接口？抽象类是否可实现 (implements) 接口？抽象类是否可继承具体类 (concrete class)？抽象类中是否可以有静态的 main 方法？.....	276
21、写 clone() 方法时，通常都有一行代码，是什么？.....	277
22、面向对象的特征有哪些方面.....	277
23、java 中实现多态的机制是什么？.....	278
24、abstract class 和 interface 有什么区别?.....	278
25、abstract 的 method 是否可同时是 static, 是否可同时是 native , 是否可同时是 synchronized?.....	279
26、什么是内部类？Static Nested Class 和 Inner Class 的不同。.....	280
27、内部类可以引用它的包含类的成员吗？有没有什么限制？.....	281
28、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) ?.....	282
29、super.getClass() 方法调用.....	282
30、String 是最基本的数据类型吗?.....	282
31、String s = "Hello";s = s + " world!";这两行代码执行后，原始的 String 对象中的内容到底变了没有？.....	282
32、是否可以继承 String 类?.....	283
33、String s = new String("xyz");创建了几个 String Object? 二者之间有什么区别？.....	284
34、String 和 StringBuffer 的区别.....	284
35、如何把一段逗号分割的字符串转换成一个数组?.....	284
36、数组有没有 length() 这个方法? String 有没有 length() 这个方法 ? .....	285

---

37、下面这条语句一共创建了多少个对象：String s="a"+"b"+"c"+"d"; .....	285
38、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？.....	285
39、下面的程序代码输出的结果是多少？ .....	286
40、final, finally, finalize 的区别。 .....	288
41、运行时异常与一般异常有何异同？ .....	288
42、error 和 exception 有什么区别?.....	288
43、Java 中的异常处理机制的简单原理和应用。 .....	288
44、请写出你最常见到的 5 个 runtime exception。 .....	289
45、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？ .....	289
46、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？ .....	289
47、sleep() 和 wait() 有什么区别?.....	290
48、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。 .....	293
49、下面两个方法同步吗？（自己发明） .....	293
50、多线程有几种实现方法？同步有几种实现方法？ .....	293
51、启动一个线程是用 run() 还是 start()？ .....	293
52、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？ .....	294
53、线程的基本概念、线程的基本状态以及状态之间的关系.....	294
54、简述 synchronized 和 java.util.concurrent.locks.Lock 的异同 ？ .....	294
55、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。 .....	296
56、子线程循环 10 次，接着主线程循环 100，接着又回到子线程循环 10 次，接着再回到主线程又循环 100，如此循环 50 次，请写出程序。 .....	298
57、介绍 Collection 框架的结构.....	303
58、Collection 框架中实现比较要实现什么接口.....	303
59、ArrayList 和 Vector 的区别.....	303

60、HashMap 和 Hashtable 的区别.....	304
61、List 和 Map 区别?.....	305
62、List, Set, Map 是否继承自 Collection 接口?.....	305
63、List、Map、Set 三个接口，存取元素时，各有什么特点？.....	305
64、说出 ArrayList,Vector, LinkedList 的存储性能和特性.....	306
65、去掉一个 Vector 集合中重复的元素.....	306
66、Collection 和 Collections 的区别。.....	307
67、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?... ..	307
68、你所知道的集合类都有哪些？主要方法？.....	307
69、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?.....	308
70、TreeSet 里面放对象，如果同时放入了父类和子类的实例对象，那比较时使用的是父类的 compareTo 方法，还是使用的子类的 compareTo 方法，还是抛异常！.....	308
71、说出一些常用的类，包，接口，请各举 5 个.....	309
72、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？.....	309
73、字节流与字符流的区别.....	309
74、什么是 java 序列化，如何实现 java 序列化？或者请解释 Serializable 接口的作用。 .....	311
75、描述一下 JVM 加载 class 文件的原理机制?.....	311
76、heap 和 stack 有什么区别。 .....	312
77、GC 是什么？为什么要有 GC?.....	312
78、垃圾回收的优点和原理。并考虑 2 种回收机制。 .....	312
79、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？ .....	312
80、什么时候用 assert。 .....	313
81、java 中会存在内存泄漏吗，请简单描述。 .....	313
82、能不能自己写个类，也叫 java.lang.String ? .....	317
83. Java 代码查错.....	317
<b>二. 算法与编程.....</b>	<b>321</b>
1、编写一个程序，将 a.txt 文件中的单词与 b.txt 文件中的单词交替合并到 c.txt 文件中，a.txt 文件中的单词用	

回车符分隔，b.txt 文件中用回车或空格进行分隔。	321
2、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。	323
3、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，但要保证汉字不被截取半个，如“我 ABC”，4，应该截取“我 AB”，输入“我 ABC 汉 DEF”，6，应该输出“我 ABC”，而不是“我 ABC+汉的半个”。	325
4、有一个字符串，其中包含中文字符、英文字符和数字字符，请统计和打印出各个字符的个数。	326
5、说明生活中遇到的二叉树，用 java 实现二叉树。	327
6、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：	332
7、写一个 Singleton 出来。	335
8、递归算法题 1	337
9、递归算法题 2	338
10、排序都有哪几种方法？请列举。用 JAVA 实现一个快速排序。	339
11、有数组 a[n]，用 java 代码将数组元素顺序颠倒。	340
12、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。	341
<b>三. html&amp;JavaScript&amp;ajax 部分</b>	342
1. 判断第二个日期比第一个日期大。	342
2. 用 table 显示 n 条记录，每 3 行换一次颜色，即 1, 2, 3 用红色字体，4, 5, 6 用绿色字体，7, 8, 9 用红颜色字体。	343
3. HTML 的 form 提交之前如何验证数值文本框的内容全部为数字？否则的话提示用户并终止提交？	344
4. 请写出用于校验 HTML 文本框中输入的内容全部为数字的 javascript 代码。	344
5. 说说你用过那些 ajax 技术和框架，说说它们的区别	345
<b>四. Java web 部分</b>	345
1、Tomcat 的优化经验	345
2、HTTP 请求的 GET 与 POST 方式的区别	345
3、解释一下什么是 servlet；	345
4、说一说 Servlet 的生命周期？	345

5、Servlet 的基本架构.....	346
6、SERVLET API 中 forward() 与 redirect() 的区别？.....	346
7、什么情况下调用 doGet() 和 doPost() ? .....	346
8、Request 对象的主要方法：.....	346
9、forward 和 redirect 的区别.....	347
10、request.getAttribute() 和 request.getParameter() 有何区别?.....	348
11. jsp 有哪些内置对象?作用分别是什么? 分别有什么方法?.....	348
12. jsp 有哪些动作?作用分别是什么?.....	349
13、JSP 的常用指令.....	349
14. JSP 中动态 INCLUDE 与静态 INCLUDE 的区别 ? .....	349
15、两种跳转方式分别是什么?有什么区别?.....	349
16、页面间对象传递的方法.....	350
17、JSP 和 Servlet 有哪些相同点和不同点，他们之间的联系是什么 ? .....	350
18、MVC 的各个部分都有那些技术来实现?如何实现?.....	350
19、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 ? .....	350
20 . 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.....	351
<b>五. 数据库部分.....</b>	<b>351</b>
1、用两种方式根据部门号从高到低 , 工资从低到高列出每个员工的信息。 .....	351
2、列出各个部门中工资高于本部门的平均工资的员工数和部门号 , 并按部门号排序.....	351
3、存储过程与触发器必须讲 , 经常被面试到? .....	352
4、数据库三范式是什么?.....	354
5、说出一些数据库优化方面的经验?.....	355
6、union 和 union all 有什么不同?.....	356
7.分页语句.....	358
8.用一条 SQL 语句 查询出每门课都大于 80 分的学生姓名 .....	360
9.所有部门之间的比赛组合.....	361

10. 每个月份的发生额都比 101 科目多的科目.....	362
11. 统计每年每月的信息.....	363
12. 显示文章标题，发帖人、最后回复时间.....	364
13. 删除除了 id 号不同，其他都相同的学生冗余信息.....	365
14. 航空网的几个航班查询题：.....	366
15. 查出比经理薪水还高的员工信息：.....	368
16. 求出小于 45 岁的各个老师所带的大于 12 岁的学生人数.....	368
17. 求出发帖最多的人：.....	369
18. 一个用户表中有一个积分字段，假如数据库中有 100 多万个用户，若要在每年第一天凌晨将积分清零，你将考虑什么，你将想什么办法解决？.....	369
19. 一个用户具有多个角色，请查询出该表中具有该用户的所有角色的其他用户。.....	370
20. xxx 公司的 sql 面试.....	370
21. 注册 Jdbc 驱动程序的三种方式.....	371
22. 用 JDBC 如何调用存储过程.....	371
23. JDBC 中的 PreparedStatement 相比 Statement 的好处.....	372
24. 写一个用 jdbc 连接并访问 oracle 数据的程序代码.....	373
25. Class.forName 的作用？为什么要用？.....	373
26. 大数据量下的分页解决方法。.....	373
27. 用 JDBC 查询学生成绩单，把主要代码写出来（考试概率极大）。.....	374
28. 这段代码有什么不足之处？.....	374
29. 说出数据连接池的工作机制是什么？.....	375
30. 为什么要用 ORM？ 和 JDBC 有何不一样？.....	375
六. XML 部分.....	375
1. xml 有哪些解析技术？区别是什么？.....	375
2. 你在项目中用到了 xml 技术的哪些方面？如何实现的？.....	376
3. 用 jdom 解析 xml 文件时如何解决中文问题？如何解析？.....	376
4. 编程用 JAVA 解析 XML 的方式.....	377

5、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？	379
<b>七. 流行的框架与新技术</b>	379
1、谈谈你对 Struts 的理解。	380
2、谈谈你对 Hibernate 的理解。	380
3、AOP 的作用。	381
4、你对 Spring 的理解。	381
5、谈谈 Struts 中的 Action servlet。	382
6、Struts 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。	382
7、STRUTS 的应用(如 STRUTS 架构)	383
8、说说 struts1 与 struts2 的区别。	384
9、hibernate 中的 update() 和 saveOrUpdate() 的区别，session 的 load() 和 get() 的区别。	385
10、简述 Hibernate 和 JDBC 的优缺点？如何书写一个 one to many 配置文件。	385
11、iBatis 与 Hibernate 有什么不同？	385
12、写 Hibernate 的一对多和多对一双向关联的 orm 配置？	385
9、hibernate 的 inverse 属性的作用？	385
13、在 DAO 中如何体现 DAO 设计模式？	386
14、spring+Hibernate 中委托方案怎么配置？	386
15、spring+Hibernate 中委托方案怎么配置？	386
16、hibernate 进行多表查询每个表中各取几个字段，也就是说查询出来的结果集没有一个实体类与之对应如何解决？	386
17、介绍一下 Hibernate 的二级缓存	386
18、Spring 的依赖注入是什么意思？给一个 Bean 的 message 属性，字符串类型，注入值为 "Hello" 的 XML 配置文件该怎么写？	388
19、Jdo 是什么？	388
20、什么是 spring 的 IOC AOP	389
21、STRUTS 的工作流程！	389
22、spring 与 EJB 的区别 !!	389
<b>八. 软件工程与设计模式</b>	389

1、 UML 方面.....	389
2、 j2ee 常用的设计模式？说明工厂模式。 .....	389
3、 开发中都用到了那些设计模式？用在什么场合？.....	390
九. j2ee 部分.....	390
1、 BS 与 CS 的联系与区别。 .....	390
2、 应用服务器与 WEB SERVER 的区别 ? .....	391
3、 应用服务器有那些 ? .....	391
4、 J2EE 是什么 ? .....	392
5、 J2EE 是技术还是平台还是框架？ 什么是 J2EE.....	392
6、 请对以下在 J2EE 中常用的名词进行解释 (或简单描述) .....	392
7、 如何给 weblogic 指定大小的内存?.....	393
8、 如何设定的 weblogic 的热启动模式 (开发模式) 与产品发布模式? .....	393
9、 如何启动时不需输入用户名与密码?.....	393
10、 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? .....	393
11、 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 //helloworld.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? .....	393
12、 在 weblogic 中发布 ejb 需涉及到哪些配置文件.....	393
13、 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? .....	393
14、 如何查看在 weblogic 中已经发布的 EJB? .....	394
十. EJB 部分.....	394
1、 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。 .....	394
2、 简要讲一下 EJB 的 7 个 Transaction Level? .....	394
3、 EJB 与 JAVA BEAN 的区别 ? .....	394
4、 EJB 包括 ( SessionBean,EntityBean ) 说出他们的生命周期 , 及如何管理事务的 ? .....	395

5、EJB 容器提供的服务.....	395
6、EJB 的激活机制.....	395
7、EJB 的几种类型.....	395
8、客服端调用 EJB 对象的几个基本步骤.....	396
<b>十一. webservice 部分.....</b>	<b>396</b>
1、WEB SERVICE 名词解释。JSWDL 开发包的介绍。JAXP、JAXM 的解释。SOAP、UDDI、WSDL 解释。.....	396
2、CORBA 是什么?用途是什么?.....	396
3. Linux.....	396
4、LINUX 下线程 , GDI 类的解释。.....	396
5. 问得稀里糊涂的题.....	397
6、四种会话跟踪技术.....	397
7、简述逻辑操作(&,  , ^)与条件操作(&&,   )的区别。.....	397
<b>十二. 电话面试等.....</b>	<b>397</b>
1、请用英文简单介绍一下自己.....	397
2、请把 <a href="http://tomcat.apache.org/">http://tomcat.apache.org/</a> 首页的这一段话用中文翻译一下?.....	397
3、美资软件公司 JAVA 工程师电话面试题目.....	398
<b>第 6 章 面霸 - C , C++ , 嵌入式.....</b>	<b>399</b>
<b>第 7 章 编程.....</b>	<b>484</b>
字符串实现.....	484
strcat.....	484
strncat.....	485
strcmp.....	485
strncmp.....	485
strcpy.....	485
strncpy.....	486
strlen.....	486
struprbrk.....	486
strstr.....	487
strcspn.....	487
strspn.....	488
strrchr.....	488
strrev.....	488

strnset.....	489
strset.....	489
strtok.....	489
strupr.....	490
strlwr.....	490
memcpy.....	491
memccpy.....	491
memchr.....	491
memcmp.....	492
memmove.....	492
memset.....	492
strdup.....	493
strchr_.....	493
strchr.....	493
atoi.....	493
itoa.....	494
<b>String 实现.....</b>	<b>495</b>
<b>编写一个二分查找的功能函数.....</b>	<b>497</b>
<b>字符串逆序.....</b>	<b>497</b>
<b>排序.....</b>	<b>498</b>
<b>冒泡排序.....</b>	<b>498</b>
<b>选择排序.....</b>	<b>499</b>
<b>插入排序.....</b>	<b>500</b>
<b>快速排序.....</b>	<b>500</b>
<b>链表.....</b>	<b>501</b>
<b>单链表.....</b>	<b>501</b>
<b>双链表.....</b>	<b>501</b>
<b>循环链表.....</b>	<b>501</b>
<b>单链表逆置.....</b>	<b>501</b>
<b>链表合并.....</b>	<b>501</b>
<b>写一个 Singleton 模式.....</b>	<b>503</b>
<b>如何对 String 类型数据的某个字符进行访问？.....</b>	<b>504</b>
<b>文件加密、解密.....</b>	<b>504</b>

---

1. 加密 ( encryption ):	.....	504
2. 解密 ( decryption ):	.....	505

斐波那契数列 ( Fibonacci sequence ) 507

## 第1章 文档概述

### 1.1 编写目的

本文档用来辅助杭州教学支持中心的培训产品，对每个学员毕业期的能力进行强化和巩固，在最后就业阶段进行冲刺。本文档为学员和项目经理的教学工作提供参考和指导。

### 1.2 主要读者

本文档的潜在读者包括杭州达内公司员工以及杭州达内在读学员。

### 1.3 版本修订

版本号	版本内容	修改时间	修改人
V1.0	Java 基础面试题	2011/11/1	陈天翔
V2.0	Java 综合面试题, 电话面试题	2012/04/5	陈天翔
V3.0	Java, Php, C, Android 笔面试题	2013/01/28	杭州教学支持中心

## 第 2 章 产品概述

### 2.1 关于杭州教学支持中心

达内 IT 培训集团是中国高端 IT 培训的第一品牌，致力于培养面向电信和金融领域的 Java、C++、嵌入式、C#/ .Net、3G/Android、3G/IOS、PHP、软件测试等 8 大课程方向中高端软件人才。先后获得美国国际数据集团 IDG、集富亚洲 JAFCO ASIA、美国高盛银行的三轮投资，是国内首家获得国际风险投资的 IT 培训机构。

杭州教学支持中心的支持范围包括：杭州西湖中心，杭州西溪中心，杭州黄龙中心，杭州武林门中心，杭州和平广场中心，华东大学生实训基地，福州中心等。它是杭州达内培训的中坚力量，为杭州其他各个中心提供带班教学，技术支持，实训招生，项目开发等各项任务的一个独立中心。

杭州教学支持中心拥有全达内集团最大的项目经理团队，也是技术实习最强的团队之一。

## 2.2 关于产品

经过 3 年的潜心整理和收集，该套资料已经集 JAVA , PHP , Android , C , C++ , 嵌入式等各语种的笔面试题。对刚入行和应届的毕业生具有极强的指导作用，在面试过程中可以解决大部分的问题。

## 第3章 面霸-PHP

企业一：

### 1、用 PHP 打印出前一天的时间,格式是 2006-5-10 22:21:21

```
<?php
//echo date('Y-m-d H:i:s',time()-60*60*24
echo date("Y:m:d H:i:s",strtotime("-1 day"));
?>
```

### 2、echo(),print(),print\_r()的区别

echo 是语言结构，无返回值；

print 功能和 echo 基本相同，不同的是 print 是函数，有返回值；

print\_r 是递归打印，用于输出数组对象

### 3、能够使 HTML 和 PHP 分离开使用的模板

so much,其实 PHP 本身就是一种模版引擎，我用过的是 smarty，常见的还有 PHPLib, FastTemplate, Savant 这里有个模版引擎列表：

### 4.如何实现 PHP、JSP 交互？

题目有点含糊不清,SOAP,XML\_RPC,Socket function,CURL 都可以实现这些,如果是考 PHP 和 Java 的整合,PHP 内置了这种机制(如果考 PHP 和.NET 的整合,也可以这么回答),例如\$foo = new Java('java.lang.System');

### 5.使用哪些工具进行版本控制？

CVS 和 SVN,SVN 号称下一代 CVS,功能强大,不过 CVS 是老牌,市占率很高.我一直用 SVN,题目是问用什么工具,呃,这个可能需要这么回答:CVS Server on Apache 作服务端,WinCVS 作客户端;Subversion on Apache/DAV 做服务端,TortoiseSVN 做客户端,或者 Subclipse 做客户端

### 6.如何实现字符串翻转？

其实 PHP 本身就有字符串翻转的函数: strrev(), 不妨试试 echo strrev(\$str); 不过所有的这三种方法都不能解决中文字符串翻转的问题, 会出错的。

代码

```
<?php
function reverse ($var)
{
    $res="";
    for ($i=0,$j=strlen($var) ;$i<$j;$i++)
    {
        $res=$var[$i].$res;
    }
    return $res;
}
$tmpvar="wofang";
$res=reverse($tmpvar);
echo $res;
?>
```

## 7、优化 MySQL 数据库的方法。

(1). 数据库设计方面, 这是 DBA 和 Architect 的责任, 设计结构良好的数据库, 必要的时候, 去正规化(英文是这个: denormalize, 中文翻译成啥我不知道), 允许部分数据冗余, 避免 JOIN 操作, 以提高查询效率

(2). 系统架构设计方面, 表散列, 把海量数据散列到几个不同的表里面. 快慢表, 快表只留最新数据, 慢表是历史存档. 集群, 主服务器 Read & write, 从服务器 read only, 或者 N 台服务器, 各机器互为 Master

(3).(1)和(2)超越 PHP Programmer 的要求了, 会更好, 不会没关系. 检查有没有少加索引

(4). 写高效的 SQL 语句, 看看有没有写低效的 SQL 语句, 比如生成笛卡尔积的全连接啊, 大量的 Group By 和 order by, 没有 limit 等等. 必要的时候, 把数据库逻辑封装到 DBMS 端的存储过程里面. 缓存查询结果, explain 每一个 sql 语句

(5). 所得皆必须, 只从数据库取必需的数据, 比如查询某篇文章的评论数, select count(\*) ... where article\_id = ? 就可以了, 不要先 select \* ... where article\_id = ? 然后 msql\_num\_rows.

只传送必须的 SQL 语句, 比如修改文章的时候, 如果用户只修改了标题, 那就 update ... set title = ? where article\_id = ? 不

要 set content = ?(大文本)

(6).必要的时候用不同的存储引擎.比如 InnoDB 可以减少死锁.HEAP 可以提高一个数量级的查询速度

## 8、谈谈事务处理

A 给 B 的账户转账 50 美元的例子

## 9、apache+mysql+php 实现最大负载的方法

同第 7 题相同

## 10.实现中文字串截取无乱码的方法。

mb\_substr()

### 11.

```
<?php
$empty = '';
>null = NULL;
$bool = FALSE;
$notSet;
$array = array();
//以下是问题
$a = "hello";
$b = &$a;
unset($b);
$b = "world";
//答案为:hello
echo $a;
?>
```

### 12.

```
<?php
$empty = '';
>null = NULL;
$bool = FALSE;
$notSet;
$array = array();
//以下是问题
$a = 1;
$x = &$a;
```

```
$b = $a++;
//以下为答案:1
echo $b;
?>
```

**13**<?php  
\$empty = '';  
\$null = NULL;  
\$bool = FALSE;  
\$notSet;  
\$array = array();  
//以下是问题  
\$x = empty(\$array);

```
//以下为答案:true
echo $x?"true":"false";
?>
```

#### 14、用 PHP 写出显示客户端 IP 与服务器 IP 的代码：

打印客户端 IP:echo \$\_SERVER['REMOTE\_ADDR']; 或者: getenv('REMOTE\_ADDR');

打印服务器 IP:echo gethostbyname("http://www.baidu.com/")

#### 企业二：

1.以下哪一句不会把 John 新增到 users 阵列？

\$users[] = 'john';  
成功把 John 新增到阵列 users。

array\_add(\$users,'john');

函式 array\_add() 无定义。

array\_push(\$users,'john');

成功把 John 新增到阵列 users。

\$users ||= 'john';

语法错误。

2.sort()、assort()、和 ksort() 有什么分别？它们分别在什么情况下使用？

sort()

根据阵列中元素的值，以英文字母顺序排序，索引键会由 0 到 n-1 重新编号。主要是当阵列索引键的值无关疼痛时用来把阵列排序。

assort()

PHP 没有 assort() 函数，所以可能是 asort() 的笔误。

asort()

与 sort() 一样把阵列的元素按英文字母顺序来排列，不同的是所有索引键都获得保留，特别适合替联想阵列排序。

ksort()

根据阵列中索引键的值，以英文字母顺序排序，特别适合用于希望把索引键排序的联想阵列。

3.以下的代码会产生什么？为什么？

```
$num =10;  
function multiply()  
{$num =$num *10;  
}  
multiply();  
echo $num;
```

由于函式 multiply() 没有指定 \$num 为全域变量（例如 global \$num 或者 \$\_GLOBALS['num']），所以 \$num 的值是 10。

4. reference 跟一个正规的变量有什么分别？如何 pass by reference？在什么情况下我们需要这样做？

Reference 传送的是变量的地址而非它的值，所以在函式中改变一个变量的值时，整个应用都见到这个变量的新值。

一个正规变量传送给函式的是它的值，当函式改变这个变量的值时，只有这个函式才见到新值，应用的其他部分仍然见到旧值。

```
$myVariable = "its' value";  
Myfunction(&$myVariable); // 以 reference 传送参数
```

以 reference 传送参数给函式，可以使函式改变了的变量，即使在函式结束后仍然保留新值。

5.些函式可以用来在现正执行的脚本中插入函式库？

对这道题目不同的理解会有不同的答案，我的第一个想法是插入 PHP 函式库不外乎 include()、include\_once()、require()、require\_once()，但细心再想，“函式库”也应该包括 com 物件和 .net 函式库，所以我们的答案也要分别包括 com\_load 和 dotnet\_load，下次有人提起“函式库”的时候，别忘记这两个函式。

6.foo() 与 @foo() 有什么分别？

foo() 会执行这个函式，任何解译错误、语法错误、执行错误都会在页面上显示出来。

@foo() 在执行这个函式时，会隐藏所有上述的错误讯息。

很多应用程序都使用 @mysql\_connect() 和 @mysql\_query 来隐藏 mysql 的错误讯息，我认为这是很严重的失误，因为错误不该被隐藏，你必须妥善处理它们，可能的话解决它们。

7.你如何替 PHP 的应用程序侦错？

我并不常这样做，我曾经试过很多不同的侦错工具，在 Linux 系统中设定这些工具一点也不容易。不过以下我会介绍一个近来颇受注目的侦错工具。

PHP - Advanced PHP Debugger 或称 PHP - APD，第一步是执行以下的指令安装：

pear install apd 安装后在你的脚本的开头位置加入以下的语句开始进行侦错：

apd\_set\_pprof\_trace(); 执行完毕，打开以下档案来查阅执行日志：

apd.dumpdir

你也可以使用 pprof 来格式化日志。

8.“==”是什么？试举一个“==”是真但“==”是假的例子。

“==”是给既可以送回布尔值“假”，也可以送回一个不是布尔值但却可以赋与“假”值的函式，strpos() 和 strrpos() 便是其中两个例子。

问题的第二部份有点困难，想一个“==”是假，但是“==”是真的例子却很容易，相反的例子却很少。但我终于找到以下的例子：

```
if(strpos("abc", "a") == true){ // 这部分永不会被执行，因为 "a" 的位置是 0，换算成布尔值“假”}if(strpos("abc", "a") === true){ // 这部份会被执行，  
因为“==”保证函式 strpos() 的送回值不会换算成布尔值。}
```

9.你会如何定义一个没有成员函式或特性的类别 myclass？

```
class myclass{}
```

10.你如何产生一个 myclass 的物件？

```
$obj = new myclass();
```

11.在一个类别内如何存取这个类别的特性及变改它的值？

使用语句：\$this->propertyName，例如：

```
class myclass{ private $propertyName; public function __construct() { $this->propertyName = "value"; }}
```

12.include 和 include\_once 有什么分别？require 又如何？

三者都是用来在脚本中插入其他档案，视乎 url\_allow\_fopen 是否核准，这个档案可以从系统内部或外部取得。但他们之间也有微细的分别：

include()：这个函式容许你在脚本中把同一个档案插入多次，若果档案不存在，它会发出系统警告并继续执行脚本。

include\_once()：它跟 include() 的功能相似，正如它的名字所示，在脚本的执行期间，有关档案只会被插入一次。

require()：跟 include() 差不多，它也是用来在脚本中插入其他档案，但若果档案不存在，它会发出系统警告，这个警告会引致致命错误令脚本中止执行

13.以下哪一个函式可以把浏览器转向到另一个页面？

redir()

这不是一个 PHP 函式，会引致执行错误。

header()

这个是正确答案，header() 用来插入卷头资料，可以用来使浏览器转向到另一个页面，例如：

location()

这不是一个 PHP 函数，会引致执行错误。

redirect()

这不是一个 PHP 函数，会引致执行错误。

14.以下哪一个函数可以用来开启档案以便读 / 写？

fget()

这不是一个 PHP 函数，会引致执行错误。

file\_open()

这不是一个 PHP 函数，会引致执行错误。

fopen()

这是正确答案，fopen() 可以用来开启档案以便读 / 写，事实上这个函数还有很多选项，详细资料请参阅 php.net。

open\_file()

这不是一个 PHP 函数，会引致执行错误。

15.mysql\_fetch\_row() 和 mysql\_fetch\_array() 有什么分别？

mysql\_fetch\_row() 把数据库的一列储存在一个以零为基数的阵列中，第一栏在阵列的索引 0，第二栏在索引 1，如此类推。mysql\_fetch\_assoc() 把数据库的一列储存在一个关联阵列中，阵列的索引就是栏位名称，例如我的数据库查询送回“first\_name”、“last\_name”、“email”三个栏位，阵列的索引便是“first\_name”、“last\_name”和“email”。mysql\_fetch\_array() 可以同时送回 mysql\_fetch\_row() 和 mysql\_fetch\_assoc() 的值。

16.下面的代码用做什么？请解释。

```
$date='08/26/2003';print ereg_replace("[0-9]+/[0-9]+/[0-9]+","\\2\\\\1\\\\3",$date);
```

这是把一个日期从 MM/DD/YYYY 的格式转为 DD/MM/YYYY 格式。我的一个好朋友告诉我可以把这个正规表达式拆解为以下的语句，对于如此简单的表示来说其实无须拆解，纯粹为了解说的方便：

```
// 对应一个或更多 0-9，后面紧随一个斜号$regExpression = "[0-9]+"/;// 应一个或更多 0-9，后面紧随另一个斜号$regExpression .= "[0-9]+"/;// 再次对应一个或更多 0-9$regExpression .= "[0-9]+";至于 \\2\\\\1\\\\3 则是用来对应括号，第一个括号对的是月份，第二个括号对应的是日期，第三个括号对应的是年份。
```

17.给你一行文字 \$string，你会如何编写一个正规表达式，把 \$string 内的 HTML 标签除去？

首先，PHP 有内建函数 strip\_tags() 除去 HTML 标签，为何要自行编写正规表达式？好了，便当作是面试的一道考题吧，我会这样回答：

```
$stringOfText = "<p>This is a test</p>";$expression = "/<(.*)>(.*)<V(.*)>/" ;echo preg_replace($expression, "\\2", $stringOfText); // 有人说  
也可以使用 /(<[^>]*>)/ $expression = "/(<[^>]*>)/";echo preg_replace($expression, "", $stringOfText);
```

18. PHP 和 Perl 分辨阵列和散列表的方法有什么差异？

这正是为何我老是告诉别人选择适当的编程语言，若果你只用一种语言的话你怎么能回答这道问题？这道问题很简单，Perl 所有阵列变量都是以 @ 开头，例如 @myArray，PHP 则沿用 \$ 作为所有变量的开头，例如 \$myArray。

至于 Perl 表示散列表则用 %，例如 %myHash，PHP 则没有分别，仍是使用 \$，例如 \$myHash。

19. 你如何利用 PHP 解决 HTTP 的无状态本质？

最主要的俩各选择是 session 和 cookie。使用 session 的方法是在每一页的开始加上 session\_start()，然后利用 \$\_SESSION 散列表来储存 session 变量。至于 cookie 你只需记着一个原则：在输出任何文字之前调用 set\_cookie() 函式，此外只需使用 \$\_COOKIE 散列表便可以存取所有 cookie 变量。

还有一个不那么可靠的方法，就是利用访客的 IP 地址，这个方法有特定的危险性。

20. GD 函式库用来做什么？

这个可能是我最喜欢的函式库，自从 PHP 4.3.0 版本后 GD 便内建在 PHP 系统中。这个函式库让你处理和显示各式格式的图档，它的另一个常见用途是制作所图档。

GD 以外的另一个选择是 ImageMagick，但这个函式库并不内建于 PHP 之中，必须由系统管理员安装在伺服器上。

21. 试写出几个输出一段 HTML 代码的方法。

嗯，你可以使用 PHP 中任何一种输出语句，包括 echo、print、printf，大部分人都使用如下例的 echo：

echo "My string \$variable"; 你也可以使用这种方法：

echo <<<ENDThis text is written to the screen as output and this \$variable is parsed too. If you wanted you can have <span> HTML tags in here as well.</span> The END; remarks must be on a line of its own, and can't contain any extra white space.END;

22. PHP 比 Perl 好吗？请讨论。

我们不要为一个简单的问题引发一场舌战，正如我经常说的：“为工作选择适合的语言，不要把工作迁就语言。”我个人认为 Perl 十分适合用作命令行工具，虽然它在网页应用上也有不错的表现，但是它的真正实力在命令行上才能充分发挥。同样地，PHP 虽然可以在控制台的环境中使用，但是个人认为它在网页应用上有更好的表现，PHP 有大量专门为网页应用而设计的函式，Perl 则似乎以命令行为设计之本。

个人来说两种语言我都喜欢，在大学期间我经常使用 Perl、PHP 和 Java，可惜工作上我使用 C#，但在家里我花不少时间操练 PHP、Perl、Ruby（现正学习）和 Java，保持我的技能知识在最新状态。很多人问我 C 和 C++ 怎么样，它们是否仍有机会在我的应用中占一席位，我的答案基本上是“否”，我近来的工作主要集中在网页开发，虽然 C 和 C++ 也可以用来写网页，但它们到底不是为这种工作而设计的，“为工作选择适合的语言”，若果我需要编写一个控制台应用，用来展示 bubble sort、quick sort 和 merge sort 的效能比较，我一定会使用 C / C++。若果我需要编写一个相片簿系统，我会使用 PHP 或者 C#（我认为制作用户介面方面 .NET 语言比网页更加）。

### 企业三：

**1 请说明 PHP 中传值与传引用的区别。什么时候传值什么时候传引用？**

答：传值只是把某一个变量的值传给了另一个变量，而引用则说明两者指向了同一个地方。

**2 在 PHP 中 error\_reporting 这个函数有什么作用？**

答：The `error_reporting()` function sets the `error_reporting` directive at runtime. PHP has many levels of errors, using this function sets that level for the duration (runtime) of your script.

**3 请用正则表达式 ( Regular Expression ) 写一个函数验证电子邮件的格式是否正确。**

答：

```
<?php
if(isset($_POST['action']) && $_POST['action']=='submitted')
{
    $email=$_POST['email'];
    if(!preg_match("/^(:w+.)*w+@(:w+.)*w+$/", $email))
    {
        echo "电子邮件检测失败";
    }
    else
    {
        echo "电子邮件检测成功";
    }
}
else
{
?>
<html>
<head><title>EMAIL 检测</title>
<script type="text/javascript">
    function checkEmail(sText)
    {
        var reg=/(:w+.)*w+@(:w+.)*w+$/;
        var email=document.getElementById(sText).value;
        if(!reg.test(email))
        {
            alert("电子邮件检测失败");
        }
        else
        {
            alert("电子邮件格式正确");
        }
    }
</script>
</head>
```

```
<body>
<form action=<?php echo $_SERVER['PHP_SELF'] ?>" method="POST">
电子邮件 : <input type="text" id="email" name="email" /><br />
<input type="hidden" name="action" value="submitted" />
<input type="button" name="button" value="客户端检测" onclick="checkEmail('email')" />
<input type="submit" name="submit" value="服务器端检测" />
</form>
</body>
</html>
<?php
}
?>
```

**4 简述如何得到当前执行脚本路径，包括所得到参数。**

```
<?php
echo "http://".$_SERVER['SERVER_NAME'].$_SERVER['PHP_SELF']."?". $_SERVER['QUERY_STRING'];
//echo "http://".$_SERVER['HTTP_HOST'].$_SERVER['REQUEST_URI'];
?>
```

**5 有一个一维数组，里面存储整形数据，请写一个函数，将他们按从大到小的顺序排列。要求执行效率高。并说明如何改善执行效率。（该函数必须自己实现，不能使用php 函数）**

```
<?php
function BubbleSort(&$arr)
{
    $cnt=count($arr);
    $flag=1;
    for($i=0;$i<$cnt;$i++)
    {
        if($flag==0)
        {
            return;
        }
        $flag=0;
        for($j=0;$j<$cnt-$i-1;$j++)
        {
```

```

if($arr[$j]>$arr[$j+1])
{
    $tmp=$arr[$j];
    $arr[$j]=$arr[$j+1];
    $arr[$j+1]=$tmp;
    $flag=1;
}
}
}

$test=array(1,3,6,8,2,7);
BubbleSort($test);
var_dump($test);
?>

```

## 6 请举例说明在你的开发过程中用什么方法来加快页面的加载速度

答：要用到服务器资源时才打开，及时关闭服务器资源，数据库添加索引，页面可生成静态，图片等大文件单独服务器。使用代码优化工具啦

Mysql 部分

### 1 创建 poll 表，用于记录单选投票用户的数据

字段包括 id[ Autoincreace ] , ip , time , iid(用户选则的选项 , int 型)

写出 create 上述 table 的完整 sql 语句

```

drop table if exists poll;

/*=====
/* Table: poll
/*=====*/
create table poll
(
    id          int unsigned      not null auto_increment,
    ip          varchar(15)        not null,
    time        datetime           not null,
    iid         int                not null,
    primary key (id)
)

```

### 2 写出将一个选择 2 号选项的 ip 为 127.0.0.1 的用户在当前时间的投票记录到数据库的 SQL

```
insert into poll (ip,time,iid) values('127.0.0.1',now(),2);
```

### 3 写出满足下边条件的 SQL 语句

item 表的结构为 id(就是 poll 表中的 iid) , desc(用户选择的选项的文字)

请查询并返回 10 条记录,包括 ip 和用户选择的选项的文字

### 4 现在因为投票人数太多，网站时常出现 too many connection 的错误，请提供解决方案

方法一：加大 MySql 的最大连接数

mysql 的最大连接数默认是 100, 这个数值对于并发连接很多的数据库应用是远远不够的, 当连接请求大于默认连接数后, 就会出现无法连接数据库的错误, 因此我们需要把它适当调大一些, 编辑 my.ini

修改 max\_connections=1000

方法二，不用 mysql 数据库，改为直接写文件，详细方法参照问题 5

若非要用 mysql, 还可

方法三：由于用 mysql 语句调用数据库时，在每次之执行语句前，会做一个临时的变量用来打开数据库，所以你在使用 mysql 语句的时候，记得在每次调用完 mysql 之后就关闭 mysql 临时变量

### 5 在成功解决连接数的问题后，发现程序运行缓慢，经查发现是 mysql 并发太多，表被锁定的现象严重，请提供解决方案

对于访问量大的，本来就不推荐使用数据库，可以考虑直接写到文本中，根据预测的访问量，先定义假若是 100 个文件文件名依次为 1.txt,2.txt...100.txt，每有用户投票的时候，随机往其中的一个文件中写入投票信息。统计的时候，再对所有文本文件中的数据进行分析。必要的时候，再导入数据库

```
drop table if exists item;
create table item
(
    id          int          not null,
    descp       varchar(200)  not null,
    primary key (id)
);
select A.ip,b.descp
from poll A,item B
where A.id=B.id
limit 10
```

### 6 因为用户实在太多，所以又分配给你两台服务器，你会如何来安排这 3 台服务器？

对于服务器分配，其实有好几种方案（建议采用 LINUX 主机），先列出一个解决方案。

1 . 考虑到电信，网通（南北差异）互访问速度慢的问题，可以让电信的用户走电信的线，网通的走网通的线。大致可以这样分配，国内南方用户（电信用户）拥有一台 服务器 A；北方用户（网通用户）拥有一台服务器 B。国外的用户也可以考虑给一台服务器 C。用户访问的时候，首先访问的是针对国外的服务器，那台机器是电信 网

通的用户访问速度都差不多的（可以考虑就租用香港或是什么地方的），经过服务器 C 判断后直接跳到相应的服务器。统计时三台机器的数据合起来。

**7 现在开始要求同一 ip 不能重复投票，请指出如何对数据表进行相应的修改**

```
ALTER TABLE `phpinterview`.`poll` ADD unique INDEX `IX_poll_ip`(`ip`);
```

**8. 原有数据已经有很多重复 ip 的数据了，所以我们把它导出为一个 txt，格式和上边的 poll 一致，用 TAB 键间隔，请写一段程序，删除 ip 有重复的记录，并统计每个投票选项的投票数**

```
<?php  
//读取文本并放入数组  
$apoll = file("c:\1.txt");  
//对每一行数据进行分割,从而获取了一个二维数组  
  
for ($i=0;$i<count($apoll);$i++)  
{  
    $poll[$i] = split(" ",$apoll[$i]);  
}  
  
//获取 IP、出现的次数数据  
  
$arrIP=array();  
for($i=0;$i<count($poll);$i++)  
{  
    $arrIP[$poll[$i][1]]=isset($arrIP[$poll[$i][1]])?$arrIP[$poll[$i][1]]+1:1;  
}  
  
//获取选项、投票个数  
  
$arrRes=array();  
for($i=0;$i<count($poll);$i++)  
{  
    if($arrIP[$poll[$i][1]]==1)  
    {  
        $arrRes[$poll[$i][3]]=isset($arrRes[$poll[$i][3]])?$arrRes[$poll[$i][3]]+1:1;  
    }  
}  
var_dump($arrRes);  
?>
```

mysql5.0 测试版：

```
/*=====得到测试数据 c: .txt=====*/  
SELECT *
```

```

into outfile 'c: .txt'
FROM `testok`;
/*=====载入临时表=====
create TABLE phpinterview.testok(id int,ip varchar(15),time datetime,iid int);
LOAD DATA INFILE 'c: .txt'
into table testok;
/*=====删除 ip 有重复的记录=====
delete A from testok A,(select ip from testok B group by ip having count(*) >1) B
where A.ip=B.ip
/*=====统计每个投票选项的投票数=====
select iid,count(*) from testok B group by B.iid

```

### 1、使用 php 写一段简单查询，查出所有姓名为“张三”的内容并打印出来

表名 User

Name	Tel	Content	Date
张三	13333663366	大专毕业	2006-10-11
张三	13612312331	本科毕业	2006-10-15
张四	021-55665566	中专毕业	2006-10-15

请根据上面的题目完成代码：

```

$mysql_db=mysql_connect("local","root","pass");
@mysql_select_db("DB",$mysql_db);

```

[复制代码](#)

代码

```

drop table if exists user;

/*=====
/* Table: user
/*=====
create table user
(
    `Name`          varchar(20),
    Tel            varchar(16),
    Content         varchar(255),
    `Date`          date

```

```
)  
  
insert into user(name,tel,content,`date`) values('张三','13333663366','大专毕业','2006-10-11');  
  
insert into user(name,tel,content,`date`) values('张三','13612312331','本科毕业','2006-10-15');  
  
insert into user(name,tel,content,`date`) values('张四','021-55665566','中专毕业','2006-10-1');  
  
select * from user where name='张三';
```

[复制代码](#)

[代码](#)

```
<?php  
header("content-type:text/html; charset=gbk");  
  
$mysql_db=mysql_connect("localhost","root","");
@mysql_select_db("phpinterview",$mysql_db);
echo "<table>";
mysql_query("set names gbk");
$result=mysql_query("select Name,Tel,Content,Date from user where Name='张三'" or die("错误:".mysql_error()));
while($row=mysql_fetch_array($result,MYSQL_BOTH))
{
    echo "<tr><td>".$row["Name"]."</td><td>".$row["Tel"]."</td><td>".$row["Content"]."</td><td>".$row["Date"]."</td></tr>";
}
mysql_free_result($result);
echo "</table>"  
?>
```

[复制代码](#)

### 3、如何使用下面的类，并解释下面什么意思？

```
class test{  
    function Get_test($num){  
        $num=md5(md5($num)."En");  
        return $num;  
    }  
}
```

```
<?php
/**
* 使用 md5 加密数据...
*
*/
class test{
    function Get_test($num) {
        $num=md5(md5($num)."En");
        return $num;
    }
}
$a=new test();
echo $a->Get_test("123");
?>
```

[复制代码](#)

#### 4、用 javascript 打印“上海爱吉”

```
<html>
<head><title>JS 打印</title></head>
<body>
<script type="text/javascript">
    document.write("上海吉它");
</script>
</body>
</html>
```

[复制代码](#)

#### 5、写出 SQL 语句的格式：插入，更新，删除

显示代码

```
select expression
from tablename
where condition
group by columns asc
with rollup
order by column asc
limit offset,rowCount;

insert into tablename(columnname) values(exp);
```

```
update tablename set columnname=exp where condition order by column limit rowcount;
```

```
delete from tablename where condition order by column limit rowcount;
```

[复制代码](#)

**1) 某内容管理系统中，表 message 有如下字段**

id 文章 id

title 文章标题

content 文章内容

category\_id 文章分类 id

hits 点击量

创建上表，写出 MySQL 语句

**2) 同样上述内容管理系统：表 comment 记录用户回复内容，字段如下**

comment\_id 回复 id

id 文章 id，关联 message 表中的 id

comment\_content 回复内容

现通过查询数据库需要得到以下格式的文章标题列表，并按照回复数量排序，回复最高的排在最前面

文章 id 文章标题 点击量 回复数量

用一个 SQL 语句完成上述查询，如果文章没有回复则回复数量显示为 0

**3) 上述内容管理系统，表 category 保存分类信息，字段如下**

category\_id int(4) not null auto\_increment;

categroy\_name varchar(40) not null;

用户输入文章时，通过选择下拉菜单选定文章分类

写出如何实现这个下拉菜单

```
drop table if exists Comment;

drop table if exists category;

drop table if exists message;

/*=====*/
```

```
/* Table: Comment */  
/*=====*/  
create table Comment  
(  
    comment_id      int unsigned      not null,  
    id              int unsigned      not null,  
    comment_content text,  
    primary key (comment_id)  
)  
type = InnoDB;  
  
/*=====*/  
/* Table: category */  
/*=====*/  
create table category  
(  
    category_id    int              not null AUTO_INCREMENT,  
    category_name varchar(40)       not null,  
    primary key (category_id),  
    key AK_pk_category_id (category_id)  
)  
type = InnoDB;  
  
/*=====*/  
/* Table: message */  
/*=====*/  
create table message  
(  
    id            int              not null,  
    title         varchar(120)       not null,  
    content       text             not null,  
    category_id   int unsigned,  
    hit           int unsigned,  
    primary key (id)  
)  
type = InnoDB;  
  
select A.id,A.title,A.hits,IFNULL(B.num,0)  
from message A left join (select id,count(*) as num from comment B group by id) B  
on A.id=B.id  
order by B.num desc;
```

```
<html>
<head><title>JS 打印</title></head>
<body>
<form>
<select id="category" name="category">
<?php
mysql_connect("localhost","root","");
die("db conn error:".mysql_error());
mysql_select_db("phpinterview");
die("db error:".mysql_error());
$result=mysql_query("select category_id,category_name from category");
while($row=mysql_fetch_array($result))
{
    echo "<option value='".$row["cateogry_id"]."'>".$row["category_name"]."</option>";
}
?>
</select>
</form>
```

**1) 内容管理系统：用户提交内容后，系统生成静态 HTML 页面；写出实现的基本思路**

直接通过 php 写入文件，或使用模板来替换标签

**2) 简单描述用户修改以发布内容的实现流程和基本思路**

更新内容，替换静态文件

**3) 写出以下程序的输出结果**

```
<?
$b=201;
$c=40;
$a=$b>$c?4:5;
echo $a;
?>
4
```

**4) 写出以下程序的输出结果**

```
<?
$str="cd";
$$str="hotdog";
$$str.="ok";
echo $cd;
?>
```

hotdogok

**5)有一表 menu(mainmenu,submenu,url),请用递归法写出一树形菜单，将所有的 menu 列出来**

```

<html>
<head><title>JS 打印</title></head>
<body>
<form>
<?php
function GenerateMenu($id=0,$str="")
{
    $result=mysql_query("select mainmenu,url,submenu from menu where mainmenu=$id");
    while($row=mysql_fetch_array($result))
    {
        echo $str.$row["url"]."<br />";
        GenerateMenu($row["submenu"],$str."--");
    }
    mysql_free_result($result);
}

$link=mysql_connect("localhost","root","");
mysql_select_db("phpinterview");
GenerateMenu();
mysql_close($link)
?>
</form>
</body>
</html>

```

**6)- 给你三个数，写程序求出其最大值。**

```

$var1=1;
$var2=7;
$var3=8;
$max=$var1>$var2?$var1:$var2;
$max=$max>$var3?$max:$var3;
echo $max;

```

**9)- 写出发贴数最多的十个人名字的 SQL，利用下表：**

```

members(id,username,posts,pass,email)
SELECT username,count(*) as num FROM `members` group by username desc order by count(*) desc limit 10

```

**10)如何通过 javascript 判断一个窗口是已经被屏蔽。**

```
<script>
```

```
var result = window.open("/somepage.aspx");
if(result==null)
{
    alert("浏览器不允许弹出窗口");
}
</script>
```

### 11)-写出 session 的运行机制

用户 A 访问站点 Y , 如果站点 Y 执行了 session\_start(); ( 以下假定 session\_start() 总是存在 ) 那么会产生一个 session\_id, 这个 session id 一般会以 COOKIE 的形式保存到用户 A ( 我们可以通过在 php.ini 里设置 session.use\_only\_cookies 为 1 , 强制 SESSION ID 必须以 COOKIE 传递。 ) 。这时候 SESSION ID 表现为 \$\_COOKIE['PHPSESSID']; ( PHPSESSID 可用 session\_name() 函数来修改 )

用户 A 接着访问 , 这个 session id(\$\_COOKIE['PHPSESSID']) 就会在 A 每次访问 Y 的时候传送到站点 Y 。

在站点 Y 上 , 会有这么一个目录 , 是用来保存 SESSION 的实际数据的。站点 Y 接收到 session id , 然后通过 session id 来获得与 SESSION 数据的关联 , 并返回 SESSION 数据。

### 13)-防止 SQL 注射漏洞一般用\_\_\_\_addslashes\_\_\_\_函数。

### 14)-查询在线人数 , 并能处理异常掉线的 SQL

基础题:

#### 1.表单中 get 与 post 提交方法的区别?

答: get 是发送请求 HTTP 协议通过 url 参数传递进行接收, 而 post 是实体数据, 可以通过表单提交大量信息.

#### 2.session 与 cookie 的区别?

答: session: 储存用户访问的全局唯一变量, 存储在服务器上的 php 指定的目录中的 ( session\_dir ) 的位置进行的存放

cookie: 用来存储连续访问一个页面时所使用 , 是存储在客户端 , 对于 Cookie 来说是存储在用户 WIN 的 Temp 目录中的。

两者都可通过时间来设置时间长短

#### 3.数据库中的事务是什么?

答: 事务 ( transaction ) 是作为一个单元的一组有序的数据库操作。如果组中的所有操作都成功 , 则认为事务成功 , 即使只有一个操作失败 , 事务也不成功。如果所有操作完成 ,

事务则提交 , 其修改将作用于所有其他数据库进程。如果一个操作失败 , 则事务将回滚 , 该事务所有操作的影响都将取消。

简述题:

**1、用 PHP 打印出前一天的时间格式是 2006-5-10 22:21:21(2 分)**

答:echo date('Y-m-d H:i:s', strtotime('-1 days'));

**2、echo(),print(),print\_r()的区别(3 分)**

答:echo 是 PHP 语句, print 和 print\_r 是函数,语句没有返回值,函数可以有返回值(即便没有用)

print ( ) 只能打印出简单类型变量的值(如 int,string)

print\_r ( ) 可以打印出复杂类型变量的值(如数组,对象)

echo 输出一个或者多个字符串

**3、能够使 HTML 和 PHP 分离开使用的模板(1 分)**

答:Smarty,Dwoo,TinyButStrong,Template Lite,Savant,phemplate,XTemplate

**5、使用哪些工具进行版本控制?(1 分)**

答:cvs,svn,vss;

**6、如何实现字符串翻转?(3 分)**

答:echo strrev(\$a);

**7、优化 MYSQL 数据库的方法。 (4 分 , 多写多得)**

答:

1、选取最适用的字段属性,尽可能减少定义字段长度,尽量把字段设置 NOT NULL,例如'省份,性别',最好设置为 ENUM

2、使用连接 ( JOIN ) 来代替子查询:

a.删除没有任何订单客户:DELETE FROM customerinfo WHERE customerid NOT in(SELECT customerid FROM orderinfo)

b.提取所有没有订单客户:SELECT FROM customerinfo WHERE customerid NOT in(SELECT customerid FROM orderinfo)

c.提高 b 的速度优化:SELECT FROM customerinfo LEFT JOIN orderid customerinfo.customerid=orderinfo.customerid

WHERE orderinfo.customerid IS NULL

3、使用联合(UNION)来代替手动创建的临时表

a.创建临时表:SELECT name FROM `nametest` UNION SELECT username FROM `nametest2`

4、事务处理:

a.保证数据完整性,例如添加和修改同时,两者成立则都执行,一者失败都失败

mysql\_query("BEGIN");

mysql\_query("INSERT INTO customerinfo (name) VALUES ('\$name1')");

mysql\_query("SELECT \* FROM `orderinfo` where customerid=\"\$id\"");

mysql\_query("COMMIT");

5、锁定表,优化事务处理:

a. 我们用一个 SELECT 语句取出初始数据，通过一些计算，用 UPDATE 语句将新值更新到表中。

包含有 WRITE 关键字的 LOCK TABLE 语句可以保证在 UNLOCK TABLES 命令被执行之前，不会有其它的访问来对 inventory 进行插入、更新或者删除的操作

```
mysql_query("LOCK TABLE customerinfo READ, orderinfo WRITE");
mysql_query("SELECT customerid FROM `customerinfo` where id=".$id);
mysql_query("UPDATE `orderinfo` SET ordertitle='".$title' where customerid=".$id);
mysql_query("UNLOCK TABLES");
```

## 6、使用外键,优化锁定表

a. 把 customerinfo 里的 customerid 映射到 orderinfo 里的 customerid，任何一条没有合法的 customerid 的记录不会写到 orderinfo 里

```
CREATE TABLE customerinfo
```

```
(  
    customerid INT NOT NULL,  
    PRIMARY KEY(customerid)  
)TYPE = INNODB;
```

```
CREATE TABLE orderinfo
```

```
(  
    orderid INT NOT NULL,  
    customerid INT NOT NULL,  
    PRIMARY KEY(customerid,orderid),  
    FOREIGN KEY (customerid) REFERENCES customerinfo  
    (customerid) ON DELETE CASCADE
```

```
)TYPE = INNODB;
```

注意:'ON DELETE CASCADE',该参数保证当 customerinfo 表中的一条记录删除的话同时也会删除 order

表中的该用户的所有记录,注意使用外键要定义事务安全类型为 INNODB;

## 7、建立索引:

a. 格式:

(普通索引)->

创建:CREATE INDEX <索引名> ON tablename (索引字段)

修改:ALTER TABLE tablename ADD INDEX [索引名] (索引字段)

创表指定索引:CREATE TABLE tablename(...,INDEX[索引名](索引字段))

(唯一索引)->

创建:CREATE UNIQUE <索引名> ON tablename (索引字段)

修改:ALTER TABLE tablename ADD UNIQUE [索引名] (索引字段)

创表指定索引:CREATE TABLE tablename([...],UNIQUE[索引名](索引字段))

(主键)->

它是唯一索引,一般在创建表时建立,格式为:

CREATE TABLE tablename ([...],PRIMARY KEY[索引字段])

## 8、优化查询语句

a.最好在相同字段进行比较操作,在建立好的索引字段上尽量减少函数操作

例子 1:

SELECT \* FROM order WHERE YEAR(orderDate)<2008;(慢)

SELECT \* FROM order WHERE orderDate<"2008-01-01";(快)

例子 2:

SELECT \* FROM order WHERE addtime/7<24;(慢)

SELECT \* FROM order WHERE addtime<24\*7;(快)

例子 3:

SELECT \* FROM order WHERE title like "%good%";

SELECT \* FROM order WHERE title>="good" and name<"good";

## 8、PHP 的意思(送 1 分)

答:PHP 是一个基于服务端来创建动态网站的脚本语言,您可以用 PHP 和 HTML 生成网站主页

## 9、MySQL 取得当前时间的函数是?, 格式化日期的函数是(2 分)

答:now(),date()

## 10、实现中文字串截取无乱码的方法。(3 分)

答:function GBsubstr(\$string, \$start, \$length) {

```
if(strlen($string)>$length){  
    $str=null;  
    $len=$start+$length;  
    for($i=$start;$i<$len;$i++){  
        if(ord(substr($string,$i,1))>0xa0){  
            $str.=substr($string,$i,2);  
            $i++;  
        }else{  
            $str.=substr($string,$i,1);  
        }  
    }  
}
```

```
return $str.'...';  
}  
else{  
return $string;  
}  
}  
}
```

**11、您是否用过版本控制软件？如果有您用的版本控制软件的名字是？(1 分)**

**12、您是否用过模板引擎？如果有您用的模板引擎的名字是？(1 分)**

答:用过,smarty

**13、请简单阐述您最得意的开发之作(4 分)**

答:信息分类

**14、对于大流量的网站,您采用什么样的方法来解决访问量问题?(4 分)**

答:确认服务器硬件是否足够支持当前的流量,数据库读写分离,优化数据表,  
程序功能规则,禁止外部的盗链,控制大文件的下载,使用不同主机分流主要流量

**15、用 PHP 写出显示客户端 IP 与服务器 IP 的代码 1 分)**

答:打印客户端 IP:echo \$\_SERVER['REMOTE\_ADDR']; 或者: getenv('REMOTE\_ADDR');  
打印服务器 IP:echo gethostbyname("www.bolaiwu.com")

**16、语句 include 和 require 的区别是什么?为避免多次包含同一文件 , 可用(?)语句代替它们? (2 分)**

答:require->require 是无条件包含也就是如果一个流程里加入 require,无论条件成立与否都会先执行 require  
include->include 有返回值 , 而 require 没有(可能因为如此 require 的速度比 include 快)  
注意:包含文件不存在或者语法错误的时候 require 是致命的,include 不是

**17、如何修改 SESSION 的生存时间(1 分).**

答:方法 1:将 php.ini 中的 session.gc\_maxlifetime 设置为 9999 重启 apache

```
方法 2:$savePath = "./session_save_dir/";  
$lifeTime = 小时 * 秒;  
session_save_path($savePath);  
session_set_cookie_params($lifeTime);  
session_start();  
方法 3:setcookie() and session_set_cookie_params($lifeTime);
```

**18、有一个网页地址, 比如 PHP 开发资源网主页: <http://www.phpres.com/index.html>,如何得到它的内容?(\$1 分)**

答:方法 1(对于 PHP5 及更高版本):

```
$readcontents = fopen("http://www.phpres.com/index.html", "rb");
$contents = stream_get_contents($readcontents);
fclose($readcontents);
echo $contents;
方法 2:
echo file_get_contents("http://www.phpres.com/index.html");
```

**19、在 HTTP 1.0 中，状态码 401 的含义是(?)；如果返回“找不到文件”的提示，则可用 header 函数，其语句为(?)；(2 分)**

答:状态 401 代表未被授权,header("Location:www.xxx.php");

**20、在 PHP 中，heredoc 是一种特殊的字符串，它的结束标志必须？(1 分)**

答:heredoc 的语法是用"<<<"加上自己定义成对的标签，在标签范围内的文字视为一个字符串

例子:

```
$str = <<<SHOW
my name is Jiang Qihui!
SHOW;
```

**21、谈谈 asp,php,jsp 的优缺点(1 分)**

答:ASP 全名 Active Server Pages , 是一个 WEB 服务器端的开发环境 , 利用它可以产生和运行动态的、交互的、高性能的 WEB 服务应用程序。 ASP 采用脚本语言 VB Script ( Java script ) 作为自己的开发语言。

PHP 是一种跨平台的服务器端的嵌入式脚本语言. 它大量地借用 C,Java 和 Perl 语言的语法 , 并耦合 PHP 自己的特性,使 WEB 开发者能够快速地写出动态生成页面.它支持目前绝大多数数据库。还有一点 , PHP 是完全免费的 , 不用花钱 , 你可以从 PHP 官方站点(<http://www.php.net>)自由下载。而且你可以不受限制地获得源码 , 甚至可以从中加进你自己需要的特色。

JSP 是 Sun 公司推出的新一代站点开发语言 , 他完全解决了目前 ASP,PHP 的一个通病 - - 脚本级执行 ( 据说 PHP4 也已经在 Zend 的支持下 , 实现编译运行 ) .Sun 公司借助自己在 Java 上的不凡造诣 , 将 Java 从 Java 应用程序 和 Java Applet 之外 , 又有新的硕果 , 就是 Jsp -- Java Server Page. Jsp 可以在 Serverlet 和 JavaBean 的支持下 , 完成功能强大的站点程序。

三者都提供在 HTML 代码中混合某种程序代码、由语言引擎解释执行程序代码的能力。但 JSP 代码被编译成 Servlet 并由 Java 虚拟机解释执行 , 这种编译操作仅在对 JSP 页面的第一次请求时发生。在 ASP 、 PHP 、 JSP 环境下 , HTML 代码主要负责描述信息的显示样式 , 而程序代码则用来描述处理逻辑。普通的 HTML 页面只依赖于 Web 服务器 , 而 ASP 、 PH

P、JSP 页面需要附加的语言引擎分析和执行程序代码。程序代码的执行结果被重新嵌入到 HTML 代码中，然后一起发送给浏览器。ASP、PHP、JSP 三者都是面向 Web 服务器的技术，客户端浏览器不需要任何附加的软件支持。

**22. 谈谈对 mvc 的认识(1 分)**

答:由模型(model),视图(view),控制器(controller)完成的应用程序

由模型发出要实现的功能到控制器,控制器接收组织功能传递给视图;

**23. 写出发贴数最多的十个人名字的 SQL , 利用下表 : members(id,username,posts,pass,email)(2 分)**

答:SELECT \* FROM `members` ORDER BY posts DESC limit 0,10;

**24. 请说明 php 中传值与传引用的区别。什么时候传值什么时候传引用?(2 分)**

答:按值传递：函数范围内对值的任何改变在函数外部都会被忽略

按引用传递：函数范围内对值的任何改变在函数外部也能反映出这些修改

优缺点：按值传递时，php 必须复制值。特别是对于大型的字符串和对象来说，这将会是一个代价很大的操作。

按引用传递则不需要复制值，对于性能提高很有好处。

**25. 在 PHP 中 error\_reporting 这个函数有什么作用? (1 分)**

答:设置错误级别与错误信息回报

**26. 请写一个函数验证电子邮件的格式是否正确 (2 分)**

答:function checkEmail(\$email)

```
{  
    $pregEmail = "/([a-zA-Z]*[-_.]?[a-zA-Z]+)*@[a-zA-Z]*[-_.]?[a-zA-Z]+.[a-zA-Z]{2,3}([.][a-zA-Z]{2})?/i";  
    return preg_match($pregEmail,$email);  
}
```

**27. 简述如何得到当前执行脚本路径 , 包括所得到参数。 (2 分)**

答:\$script\_name = basename(\_\_FILE\_\_); print\_r(\$script\_name);

**28、JS 表单弹出对话框函数是?获得输入焦点函数是? (2 分)**

答:弹出对话框: alert(),prompt(),confirm()

获得输入焦点 focus()

**29、JS 的转向函数是?怎么引入一个外部 JS 文件?(2 分)**

答>window.location.href,<script type="text/javascript" src="js/js\_function.js"></script>

**30、foo()和@foo()之间有什么区别?(1分)**

答:@foo()控制错误输出

**24、如何声明一个名为"myclass"的没有方法和属性的类? (1分)**

答:class myclass{ }

**25、如何实例化一个名为"myclass"的对象?(1分)**

答:new myclass()

**26、你如何访问和设置一个类的属性? (2分)**

答:\$object = new myclass();

\$newstr = \$object->test;

\$object->test = "info";

**27、mysql\_fetch\_row() 和 mysql\_fetch\_array 之间有什么区别? (1分)**

答:mysql\_fetch\_row 是从结果集取出 1 行数组,作为枚举

mysql\_fetch\_array 是从结果集取出一行数组作为关联数组,或数字数组,两者兼得

**28、GD 库是做什么用的? (1分)**

答:gd 库提供了一系列用来处理图片的 API , 使用 GD 库可以处理图片 , 或者生成图片。

在网站上 GD 库通常用来生成缩略图或者用来对图片加水印或者对网站数据生成报表。

**29、指出一些在 PHP 输入一段 HTML 代码的办法。 (1分)**

答:echo "<a href='index.php'>aaa</a>";

**30、下面哪个函数可以打开一个文件 , 以对文件进行读和写操作?(1分)**

- (a) fget() (b) file\_open() (c) fopen() (d) open\_file() [ c ]

**31、下面哪个选项没有将 john 添加到 users 数组中? (1分)**

- (a) \$users[] = 'john';  
(b) array\_add(\$users,'john');  
(c) array\_push(\$users,'john');  
(d) \$users ||= 'john'; [ a , c ]

**32、下面的程序会输入是否?(1分)**

```
$num = 10;  
function multiply(){
```

```
$num = $num * 10;
```

```
}
```

```
multiply();
```

```
echo $num;
```

```
?>
```

```
输出:10
```

### 33、使用 php 写一段简单查询，查出所有姓名为“张三”的内容并打印出来 (2 分)

表名 User

Name Tel Content Date

张三 13333663366 大专毕业 2006-10-11

张三 13612312331 本科毕业 2006-10-15

张四 021-55665566 中专毕业 2006-10-15

请根据上面的题目完成代码：

```
$mysql_db=mysql_connect("local","root","pass");
@mysql_select_db("DB",$mysql_db);
$result = mysql_query("SELECT * FROM `user` WHERE name='张三'");
while($rs = mysql_fetch_array($result)){
    echo $rs["tel"].$rs["content"].$rs["date"];
}
```

### 34、如何使用下面的类，并解释下面什么意思？(3)

```
class test{
```

```
function Get_test($num){
```

```
    $num=md5(md5($num)."En");
```

```
    return $num;
```

```
}
```

```
}
```

答:\$testnum = "123";

```
$object = new test();
```

```
$encrypt = $object->Get_test($testnum);
```

```
echo $encrypt;
```

类 test 里面包含 Get\_test 方法,实例化类调用方法多字符串加密

**35、写出 SQL 语句的格式：插入，更新，删除 (4 分)**

表名 User

Name Tel Content Date

张三 13333663366 大专毕业 2006-10-11

张三 13612312331 本科毕业 2006-10-15

张四 021-55665566 中专毕业 2006-10-15

(a) 有一新记录(小王 13254748547 高中毕业 2007-05-06)请用 SQL 语句新增至表中

```
mysql_query("INSERT INTO `user` (name,tel,content,date) VALUES  
('小王','13254748547','高中毕业','2007-05-06')")
```

(b) 请用 sql 语句把张三的时间更新成为当前系统时间

```
$nowDate = date("Y-m-d");  
mysql_query("UPDATE `user` SET date='".$nowDate."' WHERE name='张三'");
```

(c) 请写出删除名为张四的全部记录

```
mysql_query("DELETE FROM `user` WHERE name='张四'");
```

**36、请写出数据类型(int char varchar datetime text)的意思；请问 varchar 和 char 有什么区别(2 分)**

答:int 是数字类型,char 固定长度字符串,varchar 实际长度字符串,datetime 日期时间型,text 文本字符串

char 的场地固定为创建表设置的长度,varchar 为可变长度的字符

**38、写出以下程序的输出结果 (1 分)**

```
$b=201;  
$c=40;  
$a=$b>$c?4:5;  
echo $a;  
?>
```

答:4

**39、检测一个变量是否有设置的函数是否？是否为空的函数是？(2 分)**

答:isSet(\$str),empty(\$str);

**40、取得查询结果集总数的函数是？(1 分)**

答:mysql\_num\_rows(\$result);

**41、\$arr = array('james', 'tom', 'symfony'); 请打印出第一个元素的值 (1 分)**

答:echo \$array[0];

**42、请将 41 题的数组的值用';'号分隔并合并成字符串输出(1 分)**

答:for(\$i=0;\$i<count(\$array);\$i++){ echo \$array[\$i]. ";" ;}

**43、\$a = 'abcdef'; 请取出\$a 的值并打印出第一个字母(1 分)**

答:echo \$a{0} 或 echo substr(\$a,0,1)

**44、PHP 可以和 sql server/oracle 等数据库连接吗?(1 分)**

答:当然可以

**45、请写出 PHP5 权限控制修饰符(3 分)**

答:public(公共),private(私用),protected(继承)

**46、请写出 php5 的构造函数和析构函数(2 分)**

答:\_\_construct , \_\_destruct

**47、完成以下:**

(一)创建新闻发布系统，表名为 message 有如下字段 (3 分)

id 文章 id  
title 文章标题  
content 文章内容  
category\_id 文章分类 id  
hits 点击量

答:CREATE TABLE 'message' (

```
'id' int(10) NOT NULL auto_increment,  
'title' varchar(200) default NULL,  
'content' text,  
'category_id' int(10) NOT NULL,  
'hits' int(20),  
PRIMARY KEY('id');  
)ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

(二)同样上述新闻发布系统：表 comment 记录用户回复内容，字段如下 (4 分)

comment\_id 回复 id

id 文章 id , 关联 message 表中的 id

comment\_content 回复内容

现通过查询数据库需要得到以下格式的文章标题列表,并按照回复数量排序,回复最高的排在最前面

文章 id 文章标题 点击量 回复数量

用一个 SQL 语句完成上述查询,如果文章没有回复则回复数量显示为 0

答:SELECT message.id id,message.title title,IF(message.`hits` IS NULL,0,message.`hits`) hits,

IF(comment.`id` is NULL,0,count(\*)) number FROM message LEFT JOIN

comment ON message.id=comment.id GROUP BY message.`id`;

(三)上述内容管理系统,表 category 保存分类信息,字段如下 (3 分)

category\_id int(4) not null auto\_increment;

categroy\_name varchar(40) not null;

用户输入文章时,通过选择下拉菜单选定文章分类

写出如何实现这个下拉菜单

答:function categoryList()

```
{  
    $result=mysql_query("select category_id,categroy_name from category")  
    or die("Invalid query: " . mysql_error());  
    print("<select name='category' value=>/n");  
    while($rowArray=mysql_fetch_array($result))  
    {  
        print("<option value='".$rowArray['category_id']."'>".$rowArray['categroy_name']."</option>/n");  
    }  
    print("</select>");  
}
```

编程题:

1. 写一个函数,尽可能高效的,从一个标准 url 里取出文件的扩展名

例如: <http://www.sina.com.cn/abc/de/fg.php?id=1> 需要取出 php 或 .php

答案 1:

```
function getExt($url){  
    $arr = parse_url($url);  
  
    $file = basename($arr['path']);  
    $ext = explode(".", $file);
```

```
return $ext[1];  
}  
答案 2:  
  
function getExt($url) {  
    $url = basename($url);  
    $pos1 = strpos($url, ".");  
    $pos2 = strpos($url, "?");  
    if(strstr($url, "?")){  
        return substr($url,$pos1 + 1,$pos2 - $pos1 - 1);  
    } else {  
        return substr($url,$pos1);  
    }  
}
```

## 2. 在 HTML 语言中，页面头部的 meta 标记可以用来输出文件的编码格式，以下是一个标准的 meta 语句

请使用 PHP 语言写一个函数，把一个标准 HTML 页面中的类似 meta 标记中的 charset 部分值改为 big5

请注意：

1. 需要处理完整的 html 页面，即不光此 meta 语句
2. 忽略大小写
3. ' 和 " 在此处是可以互换的
4. 'Content-Type' 两侧的引号是可以忽略的，但 'text/html; charset=gbk' 两侧的不行
5. 注意处理多余空格

## 3. 写一个函数，算出两个文件的相对路径

如 \$a = '/a/b/c/d/e.php';

\$b = '/a/b/12/34/c.php';

计算出 \$b 相对于 \$a 的相对路径应该是 ../../c/d 将()添上

答:function getRelativePath(\$a, \$b) {

```
$returnPath = array(dirname($b));  
$arrA = explode('/', $a);  
$arrB = explode('/', $returnPath[0]);  
for ($n = 1, $len = count($arrB); $n < $len; $n++) {  
    if ($arrA[$n] != $arrB[$n]) {  
        break;  
    }  
}
```

```
    }
}

if ($len - $n > 0) {
    $returnPath = array_merge($returnPath, array_fill(1, $len - $n, '..'));
}

$returnPath = array_merge($returnPath, array_slice($arrA, $n));
return implode('/', $returnPath);
}

echo getRelativePath($a, $b);
```

**填空题:**

1.在 PHP 中 ,当前脚本的名称(不包括路径和查询字符串)记录在预定义变量 `$_SERVER['PHP_SELF']` 中;而链接到当前页面的 URL 记录在预定  
义变量 `$_SERVER['HTTP_REFERER']` 中

中

2.执行程序段`<?php echo 8%(-2) ?>`将输出 0。

3.在 HTTP 1.0 中 , 状态码 401 的含义是\_\_\_\_;如果返回“找不到文件”的提示 , 则可用 `header` 函数 , 其语句为\_\_\_\_。

4.数组函数 `arsort` 的作用是\_\_\_\_对数组进行逆向排序并保持索引关系\_\_\_\_;语句 `error_reporting(2047)`的作用是\_\_\_\_报告所有错误和警告\_\_\_\_。

5.PEAR 中的数据库连接字符串格式是\_\_\_\_。

6.写出一个正则表达式 ,过滤网页上的所有 JS/VBS 脚本(即把 `script` 标记及其内容都去掉):`preg_replace("/<script[^>].*?>.*?</script>/si", "newinfo", $script);`

7.以 Apache 模块的方式安装 PHP , 在文件 `http.conf` 中首先要用语句\_\_\_\_动态装载 PHP 模块 , 然后再用语句\_\_\_\_使得 Apache 把所有扩展名为 `php` 的文件都作为 PHP 脚本处理。

`LoadModule php5_module "c:/php/php5apache2.dll" , AddType application/x-httpd-php .php,`

8.语句 `include` 和 `require` 都能把另外一个文件包含到当前文件中 , 它们的区别是\_\_\_\_;为了避免多次包含同一文件 , 可以用语句  
`_require_once||include_once__` 来代替它们。

9.类的属性可以序列化后保存到 `session` 中 , 从而以后可以恢复整个类 , 这要用到的函数是\_\_\_\_。

10.一个函数的参数不能是对变量的引用，除非在 php.ini 中把 `_allow_call_time_pass_reference boolean_` 设为 on.

11.SQL 中 LEFT JOIN 的含义是 自然左外链接。如果 `tbl_user` 记录了学生的姓名(name)和学号(ID) , `tbl_score` 记录了学生(有的学生考试以后被开除了，没有其记录)的学号(ID)

和考试成绩(score)以及考试科目(subject) , 要想打印出各个学生姓名及对应的各科总成绩，则可以用 SQL 语句\_\_\_\_\_。

12.在 PHP 中，heredoc 是一种特殊的字符串，它的结束标志必须\_\_\_\_\_。

编程题:

13.写一个函数，能够遍历一个文件夹下的所有文件和子文件夹。

答:

```
function my_scandir($dir)
{
    $files = array();
    if ( $handle = opendir($dir) ) {
        while ( ($file = readdir($handle)) !== false ) {
            if ( $file != ".." && $file != "." ) {
                if ( is_dir($dir . "/" . $file) ) {
                    $files[$file] = scandir($dir . "/" . $file);
                }else {
                    $files[] = $file;
                }
            }
        }
        closedir($handle);
        return $files;
    }
}
```

14.简述论坛中无限分类的实现原理。

答:

```
<?php
```

```
/*
```

数据表结构如下:

```
CREATE TABLE `category` (
```

```
'categoryID` smallint(5) unsigned NOT NULL auto_increment,  
'categoryParentID` smallint(5) unsigned NOT NULL default '0',  
'categoryName` varchar(50) NOT NULL default "",  
PRIMARY KEY (`categoryID`)  
) ENGINE=MyISAM DEFAULT CHARSET=gbk;  
  
INSERT INTO `category` ( `categoryParentID`, `categoryName` ) VALUES  
(0, '一级类别'),  
(1, '二级类别'),  
(1, '二级类别'),  
(1, '二级类别'),  
(2, '三级类别'),  
(2, '333332'),  
(2, '234234'),  
(3, 'aqqqqqd'),  
(4, '哈哈'),  
(5, '66333666');
```

\*/

```
//指定分类 id 变量$category_id,然后返回该分类的所有子类  
//$/default_category 为默认的选中的分类  
  
function Get_Category($category_id = 0,$level = 0, $default_category = 0)  
{  
    global $DB;  
    $sql = "SELECT * FROM category ORDER BY categoryID DESC";  
    $result = $DB->query( $sql );  
    while ($rows = $DB->fetch_array($result))  
    {  
        $category_array[$rows[categoryParentID]][$rows[categoryID]] = array('id' => $rows[categoryID], 'parent' => $rows[categoryParentID], 'name' =>  
        $rows  
  
        [categoryName]);  
    }  
    if (!isset($category_array[$category_id]))
```

```

{
return "";
}

foreach($category_array[$category_id] AS $key => $category)
{
if ($category['id'] == $default_category)
{
echo "<option selected value=". $category['id']. ">";
}else
{
echo "<option value=". $category['id']. ">";
}

if ($level > 0)
{
echo ">" . str_repeat( " ", $level ) . " " . $category['name'] . "</option>/n";
}
else
{
echo ">" . $category['name'] . "</option>/n";
}

Get_Category($key, $level + 1, $default_category);
}

unset($category_array[$category_id]);
}

```

/\*

函数返回的数组格式如下所示:

```

Array
(
[1] => Array ( [id] => 1 [name] => 一级类别 [level] => 0 [ParentID] => 0 )
[4] => Array ( [id] => 4 [name] => 二级类别 [level] => 1 [ParentID] => 1 )
[9] => Array ( [id] => 9 [name] => 哈哈 [level] => 2 [ParentID] => 4 )
[3] => Array ( [id] => 3 [name] => 二级类别 [level] => 1 [ParentID] => 1 )
[8] => Array ( [id] => 8 [name] => aqqqqqqd [level] => 2 [ParentID] => 3 )

```

```
[2] => Array ( [id] => 2 [name] => 二级类别 [level] => 1 [ParentID] => 1 )
[7] => Array ( [id] => 7 [name] => 234234 [level] => 2 [ParentID] => 2 )
[6] => Array ( [id] => 6 [name] => 333332 [level] => 2 [ParentID] => 2 )
[5] => Array ( [id] => 5 [name] => 三级类别 [level] => 2 [ParentID] => 2 )
[10] => Array ( [id] => 10 [name] => 66333666 [level] => 3 [ParentID] => 5 )
)
*/
//指定分类 id,然后返回数组
function Category_array($category_id = 0,$level=0)
{
global $DB;
$sql = "SELECT * FROM category ORDER BY categoryID DESC";
$result = $DB->query($sql);
while ($rows = $DB->fetch_array($result))
{
$category_array[$rows['categoryParentID']][$rows['categoryID']] = $rows;
}

foreach ($category_array AS $key=>$val)
{
if ($key == $category_id)
{
foreach ($val AS $k=> $v)
{
$options[$k] =
array(
'id' => $v['categoryID'], 'name' => $v['categoryName'], 'level' => $level, 'ParentID'=>$v['categoryParentID']
);

$children = Category_array($k, $level+1);

if (count($children) > 0)
{
$options = $options + $children;
}
}
}
}
```

```
}

}

}

unset($category_array[$category_id]);

return $options;

}

?>

<?php

class cate

{

    function Get_Category($category_id = 0,$level = 0, $default_category = 0)
    {
        echo $category_id;
        $arr = array(
            '0' => array(
                '1' => array('id' => 1, 'parent' => 0, 'name' => '1111'),
                '2' => array('id' => 2, 'parent' => 0, 'name' => '2222'),
                '4' => array('id' => 4, 'parent' => 0, 'name' => '4444')
            ),
            '1' => array(
                '3' => array('id' => 3, 'parent' => 1, 'name' => '333333'),
                '5' => array('id' => 5, 'parent' => 1, 'name' => '555555')
            ),
            '3' => array(
                '6' => array('id' => 6, 'parent' => 3, 'name' => '66666'),
                '7' => array('id' => 7, 'parent' => 3, 'name' => '77777')
            ),
            '4' => array(
                '8' => array('id' => 8, 'parent' => 4, 'name' => '8888'),
            )
        );
    }
}
```

```
'9' => array('id' => 9, 'parent' => 4, 'name' => '9999')
)
);

if (!isset($arr[$category_id]))
{
    return "";
}

foreach($arr[$category_id] AS $key => $cate)
{
    if ($cate['id'] == $default_category)
    {
        $txt = "<option selected value=". $cate['id']. ">";
    }else{
        $txt = "<option value=". $cate['id']. ">";
    }

    if ($level > 0)
    {
        $txt1 = ">" . str_repeat( "-", $level ) . " " . $cate['name'] . "</option>/n";
    }else{
        $txt1 = ">" . $cate['name'] . "</option>/n";
    }

    $val = $txt.$txt1;
    echo $val;
    self::Get_Category($key, $level + 1, $default_category);
}

}

function getFlush($category_id = 0,$level = 0, $default_category = 0)
{
```

```
ob_start();

self::Get_Category($category_id , $level, $default_category);

$out = ob_get_contents();

ob_end_clean();

return $out;

}

}

$id = $_GET['id'];

echo "<select>";

$c = new cate();

// $c->Get_Category();

$ttt = $c->getFlush($id, '0', '3');

echo $ttt;

echo "</select>";

?>
```

## 问题

1. 考虑如下脚本。标记处应该添加什么代码才能让脚本输出字符串 php ?

```
$alpha = 'abcdefghijklmnopqrstuvwxyz';

$letters = array(15, 7, 15);

foreach($letters as $val) {/* 这里应该加入什么 */}

?>

A . echo chr($val);

B . echo asc($val);
```

C . echo substr(\$alpha, \$val, 2);

D . echo \$alpha{\$val};

E . echo \$alpha{\$val+1}

2 . 以下哪一项不能把字符串\$s1 和\$s2 组成一个字符串 ?

A . \$s1 + \$s2

B . "{\$s1}{\$s2}"

C . \$s1.\$s2

D . implode(, array(\$s1,\$s2))

E . 以上都可以

3 . 变量 \$email 的值是字符串 user@example.com , 以下哪项能把字符串转化成 example.com ?

A . substr(\$email, strpos(\$email, "@"));

B . strstr(\$email, "@");

C . strchr(\$email, "@");

D . substr(\$email, strpos(\$email, "@")+1);

E . strrpos(\$email, "@");

4 . 给定一个用逗号分隔一组值的字符串 , 以下哪个函数能在仅调用一次的情况下就把每个独立的值放入一个新创建的数组 ?

A . strstr()

B . 不可能只调用一次就完成

C . extract()

D . explode()

E . strtok()

5 . 要比较两个字符串，以下那种方法最万能？

A . 用 strpos 函数

B . 用==操作符

C . 用 strcasecmp()

D . 用 strcmp()

6 . 以下哪个 PCRE 正则表达式能匹配字符串 php|architect ?

A . .\*

B . ....|.....

C . \d{3}\|\d{8}

D . [az]{3}\|[az]{9}

E . [a-z][a-z][a-z]\|\w{9}

7 . 以下哪些函数能用来验证字符串的完整性？( 三选 )

A . md5()

B . sha1()

C . str\_rot13()

D . crypt()

E . crc32()

8 . 哪个 PHP 函数与以下脚本在 UNIX 系统下执行的效果近似 ?

```
function my_funct ($filename)
```

{

\$f = file\_get\_contents (\$filename);

return explode ("\n", \$f);

}

?>

A . fopen()

B . fread()

C . flock()

D . split\_string()

E . file()

9 . 基于指定的式样 ( pattern ) 把一个字符串分隔开并放入数组 , 以下哪些函数能做到 ? ( 双

选 )

A . preg\_split()

B . ereg()

C . str\_split()

D . explode()

E . chop()

10 . 以下脚本输出什么 ?

echo 'Testing ' . 1 + 2 . '45';

?>

A . Testing 1245

B . Testing 345

C . Testing 1+245

D . 245

E . 什么都没有

11 . 以下脚本输出什么 ?

```
$s = '12345';
```

```
$s[$s[1]] = '2';
```

```
echo $s;
```

```
?>
```

A . 12345

B . 12245

C . 22345

D . 11345

E . Array

12 . 方框中的正则表达式能与以下哪些选项匹配 ? ( 双选 )

```
/*\*123\d/
```

A . \*\*\*\*\*123

B . \*\*\*\*\*\_1234

C . \*\*\*\*\*1234

D . \_\*1234

E . \_\*123

13. 以下哪个比较将返回 true ? ( 双选 )

A . '1top' == '1'

B . 'top' == 0

C . 'top' === 0

D . 'a' == a

E . 123 == '123'

14. 如果用+操作符把一个字符串和一个整型数字相加 , 结果将怎样 ?

A . 解释器输出一个类型错误

B . 字符串将被转换成数字 , 再与整型数字相加

C . 字符串将被丢弃 , 只保留整型数字

D . 字符串和整型数字将连接成一个新字符串

E . 整形数字将被丢弃 , 而保留字符串

15. 考虑如下脚本。假设 <http://www.php.net> 能被访问 , 脚本将输出什么 ?

```
$s = file_get_contents ("http://www.php.net");
strip_tags ($s, array ('p'));
echo count ($s);
?>
```

A . www.php.net 的主页的字符数

B . 剔除

标签后的 www.php.net 主页的字符数

C . 1

D . 0

E . 剔除

以外的标签后的 www.php.net 主页的字符数

16 . 哪个函数能不区分大小写得对两个字符串进行二进制比对 ?

A . strcmp()

B . stricmp()

C . strcasecmp()

D . strstr()

E . 以上都不能

17 . 以下哪些函数能把字符串里存储的二进制数据转化成十六进制 ? ( 双选 )

A . encode\_hex()

B . pack()

C . hex2bin()

D . bin2hex()

E . printf()

18 . 哪个函数能用来确保一个字符串的字符数总是大于一个指定值 ?

答案 : \_\_\_\_\_

19 . 以下脚本输出什么 ?

\$a = 'able osts indy';

---

```
echo wordwrap ($a, 1, "c", false);
```

```
?>
```

答案：\_\_\_\_\_

20. 以下脚本输出什么？

```
$x = 'apple';
```

```
echo substr_replace ($x, 'x', 1, 2);
```

```
?>
```

A. x

B. axle

C. axxle

D. applex

E. xapple

答案

1. substr 函数能够胜任，但考虑到输出三个字母就需要三次调用该函数，所以排除此方法。

那么 \$alpha{\$val} 和 \$alpha{\$val+1} 是仅有的两个可能输出题目要求的字符串的选项。因

为 0 是数组的第一个索引，所以答案是 D。

2. 除了 A 以外的选项都能输出题目要求的字符串。PHP 中，加号 (+) 不能把两个字符  
串合并成一个。

3. substr 函数返回字符串的一部分，而 strpos 函数擅长从一个字符串中找出某个指定的子  
串。同时使用这两个函数将满足题目要求。注意，前一个函数从 0 开始索引，而后者

不是，因此需要+1。答案是 D。

4 . 答案是 D。explode 函数使用一个字符串分隔另一个字符串，并把结果放入一个新建的数组。strtok 函数也可以做同样的事，但需要多次调用。

5 . 答案是 D。strcmp()提供了安全的字符串比较机制。注意，选项 C 是错的，strcasecmp()不是一个“万能”函数，因为它不区分大小写。

6 . 选项中没有一个正则表达式能真正代表题目所给字符串的匹配方式，但是选项 A 和 E 仍然能勉强匹配。选项 A 太普通了，它能够匹配任何字符串，因此答案是 E。

7 . 正确答案是 A , B 和 E。用 crypt()和 str\_rot13()来验证一个字符串是否被改变，效率很低。crc32()比前面两个函数好些，如果能容忍一些小错误的话，它是个不错的选择。

8 . file 函数将文件的文本内容读入一个数组，每个元素是一行。因此答案 E 正确。也许你想知道为什么要把这样一个题目放在讲字符串的章节中，那是为了提醒你每一章的题目所包含的知识点并不是绝对严格区分开的，正如写 PHP 脚本时，file 函数不能脱离字符串函数单独存在一样。

9 . 尽管条件不同，但 preg\_split 和 explode 函数都能满足题目要求。ereg()拿一个正则表达式匹配一个字符串；str\_split()按固定长度分隔字符串；而 chop()则是 rtrim()别名，用来移除字符串末尾处的空格。

10 . 本题考察你对字符串操作及操作符优先级的认识。连接运算符 ( . ) 的优先级比加号 ( + ) 高。因此 PHP 解释器实际执行的运算可以表示为('Testing ' . 1) + (2 . '45')。由于字符串 test 1 不是数字，所以加号前面的运算等于 0。加号后面的运算等于 245，PHP 输出的结果是 0+245，等于 245，所以答案是 D。

11 . 可以用访问数组元素的方式访问字符串中的字符，因此脚本只是把字符串中的第二个

字符 (`$s[1]`) 替换成了字符 2，最终将输出 12245。答案是 B。

12. 本题的要点是理解这个正则表达式的含义——从左往右，首先是零个或多个任意字符

(`.*`)，跟着是一个星号 (`\*`)，然后是 123，最后是一个数字。因此答案是 C 和 D。

13. B 和 E 正确。选项 B 中，在比较时，字符串 `top` 等同于数字 0。`==` 操作符不比对数据类型，所以将返回 `true`。答案 E 中，字符串 `123` 等同于数字 `123`，比较将返回 `true`。

14. 字符串将被转换成数字（如果无法发生转换就是 0），然后与整型数字相加。答案是 B。

15. 代码的本意是剔除 `www.php.net` 主页上除了 `p` 以外的所有 HTML 标签。可实际上，在代码的最后一行使用了 `count` 函数，它统计变量中的元素数量，而不是字符串中的字符数。由于字符串是标量，对字符串使用 `count` 函数将永远返回 1。答案是 C。

16. 题目其实就是在描述 `strcasecmp` 函数的作用，因此答案是 C。

17. 正确答案是 B 和 D。`pack` 函数能对二进制数据进行复杂的格式化，包括将字符串中的字符转化成十六进制表示。`bin2hex` 函数也有同样的转化功能。注意，`printf()` 能将整数转化成十六进制数，但无法转化字符串。

18. 这是在说 `str_pad` 函数，它可以把字符串填充到指定长度。

19. 脚本将输出 `ablecostscindy`。`wordwrap` 函数通常用来把字符串切割成指定长度。然而在本题中，长度被设置为 1，因此函数将从空格处切割（第四个参数被设置为 `false`，因此函数不会从单词的中间进行切割）。填充字符串是 `c`，等于把每个空格都换成了 `c`。

20. `substr_replace` 函数是用一个指定字符串替换原字符串中的某个部分，因此脚本输出 `axle`，答案是 B。

你可能觉得 PHP 的文件操作功能并不怎样，但实际上它对开发者来说非常有用。即使你是做网站开发的，学会相关技能也能让你如虎添翼。多亏了流包装器（stream wrappers，将在第十章详细介绍），PHP 才能够打开并读取远程文件，让在本地使用第三方网站的内容变得可能。

站在更底层的角度，文件输入/输出能完成多种任务。可以用他读取预制文件的内容，比如第三方提供的内容；或者通过 PHP 脚本让浏览器打开一个二进制文件，使得你能更切实的控制它。无论如何，本章不仅考验你打开、关闭和读取文件的能力，还考查多进程下进行文件操作的基础知识——例如文件锁。

## php 面试题及答案,经典 php 笔试题与答案

1. 函数\_\_\_\_\_能读取文本文件中的一行。读取二进制文件或者其他文件时，应当使用\_\_\_\_\_函数。

- A . fgets(), fseek()
- B . fread(), fgets()
- C . fputs(), fgets()
- D . fgets(), fread()
- E . fread(), fseek()

2. 文件指针能在 PHP 脚本结束时自动关闭，但你也可以用\_\_\_\_\_函数来关闭。

答案：\_\_\_\_\_

3. 考虑如下 PHP 脚本，它一行一行的读取并显示某文本文件的内容。在问号处填入什么才

能使脚本正常运作？

\$file = fopen("test", "r");

while(!feof(\$file)) {

echo ?????????????;

}

fclose(\$file);

?>

A . file\_get\_contents(\$file)

B . file(\$file)

C . read\_file(\$file)

D . fgets(\$file)

E . fread(\$file)

4 . 以下哪种方法能保证锁在任何竞争情况下都安全？

A . 用 flock() 锁住指定文件

B . 用 fopen() 在系统的临时文件夹里打开文件

C . 用 tempnam() 创建一个临时文件

D . 用 mkdir() 创建一个文件夹来当

E . 用 tmpfile() 创建一个临时文件

5 . 以下哪个函数能够获得文件的全部内容，并能够用在表达式中？( 双选 )

A . file\_get\_contents()

B . fgets()

C . fopen()

D . file()

E . readfile()

6 . 在不把文件内容预加载到变量中的前提下，如何解析一个以特殊格式格式化过的多行文件？

A . 用 file() 函数把它分割放入数组

B . 用 sscanf()

C . 用 fscanf()

D . 用 fgets()

E . 用 fnmatch()

7 . 考虑如下脚本，最后文件 myfile.txt 的内容是什么？

```
$array = '0123456789ABCDEFGHIJKLMNPQRSTUVWXYZ';
$f = fopen ("myfile.txt", "r");
for ($i = 0; $i < 50; $i++) {
    fwrite ($f, $array[rand(0, strlen ($array) - 1)]);
}
?>
```

A . 什么都没有，因为\$array 实际上是一个字符串，而不是数组

B . 49 个随机字符

C . 50 个随机字符

D . 41 个随机字符

E . 什么都没有，或者文件不存在，脚本输出一个错误

8 . 函数 delete 是做什么的？

- A . 删除文件
- B . 删除文件夹
- C . 释放变量
- D . 移除数据库记录
- E . 没有这个函数！

9 . 考虑如下脚本，哪个 PHP 函数和它的功能最接近？

```
function my_funct ($file_name, $data)  
{  
    $f = fopen ($file_name, 'w');  
    fwrite ($f, $data);  
    fclose ($f);  
}  
?>
```

- A . file\_get\_contents()
- B . file\_put\_contents()
- C . 没有这样的函数
- D . file()
- E . fputs()

10 . 如果你的脚本无法正确识别一个存储于另一个平台上的文件的行结尾，你该怎么办？

A . 改变 auto\_detect\_line\_ending 的设置

B . 用正则表达式侦测行的最后一个字母

C . 用 fpos()

D . 用 ftok()

E . 每次读取一个字符

11 . 如果想要可读可写得打开一个文件 , 该给 fopen() 传什么参数 ? ( 双选 )

A . w

B . r

C . a

D . +

12 . 能够读写常规文件中的二进制数据的函数是\_\_\_\_\_ , 该函数返回的资源能被 fgets() 使用。

答案 : \_\_\_\_\_

13 . 以下哪些函数能读取文件的全部内容 ? ( 三选 )

A . fgets()

B . file\_get\_contents()

C . fread()

D . readfile()

E . file()

14 . 哪个函数能够往文本文件中写入一个字符串 ?

答案 : \_\_\_\_\_

15 . 考虑如下脚本。运行时 , 尽管文件 test.txt 已经被用 unlink() 函数删除 , 脚本仍然输出 1,1 。

在脚本的最后添加什么函数才能解决这个问题？

```
$f = fopen ("test.txt", "w");

fwrite ($f, "test");

fclose ($f);

echo (int) file_exists("test.txt") . ' ';

unlink ("c:\\test.txt");

echo (int) file_exists ("test.txt");

?>
```

A . clearstatcache()

B . fflush()

C . ob\_flush()

D . touch()

E . 以上都不对

16 . 函数\_\_\_\_\_能判断一个文件是否可写。

答案：\_\_\_\_\_

17 . 以下哪个选项能将文件指针移到开头？

A . reset()

B . fseek(-1)

C . fseek(0, SEEK\_END)

D . fseek(0, SEEK\_SET)

E . fseek(0, SEEK\_CUR)

18 . stat()和 fstat()有什么区别 ?

- A . stat()基于文件指针工作 , fstat()基于路径工作
- B . fstat()基于文件指针工作 , stat()基于路径工作
- C . fstat()不能处理文件
- D . stat()不能处理文件
- E . fstat()是 stat()的别名

19 . 以下哪个选项准确的描述出了方框中的脚本的作用 ?

```
echo number_format (disk_free_space ('c:\') /  
disk_total_space('c:\') * 100, 2) . '%';  
?>
```

- A . 计算 Windows 系统 C 盘的剩余磁盘空间大小
- B . 输出一个表示 C 盘剩余空间所占比例的两位小数
- C . 输出 C 盘剩余容量的 byte 数
- D . 计算 C 盘总容量与剩余空间的比率
- E . 以上都不对

20 . 假设 image.jpg 存在并能够被 PHP 读取 , 调用以下脚本时 , 浏览器上显示什么 ?

```
header ("Content-type: image/jpeg");
```

```
?>
```

```
readfile ("image.jpg");
```

```
?>
```

- A . 一张 JPEG 图片

- 
- B . 一个二进制文件
  - C . 下载一个二进制文件
  - D . 下载一张 JPEG 图片
  - E . 一张残破的图片
- 

### 答案

1 . fgets 函数主要用来从文本文件中读取一行，当然你也可以指定每次读取的最大长度。

fread 函数主要用来读取二进制数据。答案是 D。

2 . 函数 fclose 能关闭文件指针。

3 . fgets 函数能从文件中读取单独一行。因此答案是 D。

4 . 正确答案是 D。这题很难，而且在实践中不大可能会碰到这样的问题——但这不正是你

读这本书的原因吗？！你必须记住，flock() 使用一种“协议”锁定机制，即所有其他访

问此文件的进程也必须要使用 flock()。如果某个进程没有这么做，竞争就会产生，锁就

不安全。用 mkdir 创建一个文件夹能保证任何时刻只有一个进程处理某文件，即

保证操作的原子性。因此，你可以创建一个临时文件夹并“护”住它，直到 I/O 操作完成。

5 . 只有 file\_get\_contents 和 file 函数返回文件的全部内容，因此答案是 A 和 D。readfile 函数也能读取文件的全部内容，但它直接把内容送入了输出缓存，因此不能用在表达式中。

6 . fscanf 函数能根据指定格式解析文件内容，因此答案是 C。sscanf 函数只能用来操作字符串。

7 . 答案是 E。注意，文件被以 r 模式打开，即只读模式。因此，如果文件不存在，PHP 将输出一个错误来指出没有找到文件。如果文件存在，fopen() 将被成功调用，但由于是以只

读方式打开，fwrite()会失败。如果我们用 w 代替 r，脚本就能正常运行，并且 myfile.txt

内将有 50 个随机字符（记住，可以像访问数组那样使用索引来访问字符串）。

8 . 答案是 E。PHP 里没有叫 delete()的函数。删除文件用 unlink()，删除文件夹用 rmdir()，数据库记录用 SQL 语句删除，释放变量用 unset()。

9 . 脚本实现的功能与 file\_put\_contents()最接近，但这个函数直到 PHP5 才被引入，因此答案是 C。

10 . PHP4.3.0 开始，php.ini 引入了 auto\_detect\_line\_endings 设置，系统在保存文本文件时能够自动侦测行结束符号的类型，因此答案是 A。

11 . 要可读可写的打开文件，你必须使用 r+模式，因此答案是 B 和 D。

12 . 这是在说 fopen()函数。

13 . 正确答案是 B , D 和 E。file , readfile 和 file\_get\_contents 都能读取文件的全部内容。

14 . fwrite()和 fputs()两个函数在这里都可以，而后者其实是前者的别名。在 PHP 中，写入二进制数据和写入字符串没有区别。

15 . PHP 会缓存某些文件系统函数的返回值——包括 file\_exists()，这样能提高脚本处理重复操作时的效率。当脚本里有大量删除文件的操作时，缓存很容易就会过时，因此需要清理缓存。答案是 A。

16 . 这是在说 is\_writeable 函数，它返回一个表示文件是否可写的布尔值。

17 . 正确答案是 D。fseek()用来移动文件指针。SEEK\_SET 指出偏移量从文件开头开始计算。如果没有特别指出，SEEK\_SET 就是 fseek()的默认模式。注意，rewind 函数等效于 fseek(0,SEEK\_SET)。

18 . 答案是 B。fstat 函数通过已打开的文件指针取得文件信息，stat()获取指定路径的文件信

息。

19 . 正确答案是 B。disk\_free\_space 函数能确定指定设备上（本题中即 Windows 下的 C 盘）的剩余磁盘空间（单位是 byte），而 disk\_total\_space() 能确定设备的总容量。两者相除，再乘以百分率，最后用 number\_format() 保留两位小数，脚本输出的就是剩余磁盘空间所占的比例。最后在加上百分号以防混淆。

20 . 答案是 E。你注意到两个代码块之间的空行了吗？它将被输出到浏览器上，使得整个图片的二进制数据出错。因此浏览器将显示一个破碎的图片（或者是一条信息，指出图片出错）。译者注：原文中两个代码块之间并没有空行，而在我添加了空行之后，也没有调试出答案中描述的情况。）

## 管理

### 日期与时间

从某一点上来看，几乎所有的网站都需要处理日期与时间。假如你需要收集用户的生日，或者记录某个特定事件的发生时间，PHP 的日期函数将很好的帮助你完成任务。

但是 PHP 的日期/时间管理功能并不完美。它基于 UNIX 时间戳运行，容易受到攻击，作为开发者，你必须谨慎处理可能遇到的恶意数据。

同时，在 Web 上进行日期管理是一件国际性的事务。你必须能依据时区、地区的不同来显示对应的日期信息。

1. 在 PHP 中，当前脚本的名称（不包括路径和查询字符串）记录在预定义变量（1）中；而链接到当前页面的 URL 记录在预定义变量（2）中。

2. 执行程序段<?php echo 8%(-2) ?> 将输出（3）。

- 
3. 在 HTTP 1.0 中，状态码 401 的含义是 (4)；如果返回“找不到文件”的提示，则可用 header 函数，其语句为 (5)。
4. 数组函数 arsort 的作用是 (6)；语句 error\_reporting(2047) 的作用是 (7)。
5. PEAR 中的数据库连接字符串格式是 (8)。
6. 写出一个正则表达式，过滤网页上的所有 JS/VBS 脚本（即把 script 标记及其内容都去掉）：(9)。
7. 以 Apache 模块的方式安装 PHP，在文件 http.conf 中首先要用语句 (10) 动态装载 PHP 模块，然后再用语句 (11) 使得 Apache 把所有扩展名为 php 的文件都作为 PHP 脚本处理。
8. 语句 include 和 require 都能把另外一个文件包含到当前文件中，它们的区别是 (12)；为了避免多次包含同一文件，可以用语句 (13) 来代替它们。
9. 类的属性可以序列化后保存到 session 中，从而以后可以恢复整个类，这要用到的函数是 (14)。
10. 一个函数的参数不能是对变量的引用，除非在 php.ini 中把 (15) 设为 on。
11. SQL 中 LEFT JOIN 的含义是 (16)。[www.phpierz.com](http://www.phpierz.com)  
如果 tbl\_user 记录了学生的姓名 (name) 和学号 (ID)，  
tbl\_score 记录了学生（有的学生考试以后被开除了，没有其记录）的学号 (ID) 和考试成绩 (score) 以及考试科目 (subject)，要想打印出各个学生姓名及对应的各科总成绩，则可以用 SQL 语句 (17)。
12. 在 PHP 中，heredoc 是一种特殊的字符串，它的结束标志必须 (18)。
13. 写一个函数，能够遍历一个文件夹下的所有文件和子文件夹。
14. 简述论坛中无限分类的实现原理。
15. 设计一个网页，使得打开它时弹出一个全屏的窗口，该窗口中有一个文本框和一个按钮。用户在文本框中输入信息后点击按钮就可以把窗口关闭，而输入的信息却在主网页中显示。

[www.phpierz.com](http://www.phpierz.com)

//答案（填空）：

1. echo \$\_SERVER['PHP\_SELF']; echo \$\_SERVER["HTTP\_REFERER"];
2. 0
3. (4)未授权 (5) header("HTTP/1.0 404 Not Found");
4. (6)对数组进行逆向排序并保持索引关系 (7)All errors and warnings
5. 没弄明白

```
6. /<script[^>].*?>.*?</script>/si
7.(10) LoadModule php5_module "D:/xampp/apache/bin/php5apache2.dll"
(11) AddType application/x-httplibd-php-source .phps
AddType application/x-httplibd-php .php .php5 .php4 .php3 .phtml
8.(12) 发生异常时 include 产生警告 require 产生致命错误 (13) require_once() / include_once()
9. serialize() /unserialize()
10. allow_call_time_pass_reference
11. (16) 自然左外连接
(17) select name , count(score) as sum_score from tbl_user left join tbl_score on
tbl_user.ID=tbl_score.ID group by tbl_user.ID
12. 结束标识符所在的行不能包含任何其它字符除"; "
13.
/***
* 遍历目录，结果存入数组。支持 php4 及以上。php5 以后可用 scandir() 函数代替 while 循环。 php 程序员站
* @param string $dir
* @return array
*/
function my_scandir($dir)
{
$files = array();
if ( $handle = opendir($dir) ) {
while ( ($file = readdir($handle)) !== false ) {
if ( $file != ".." && $file != "." ) {
if ( is_dir($dir . "/" . $file) ) {
$files[$file] = rec_scandir($dir . "/" . $file);
} else {
$files[] = $file;
}
}
}
closedir($handle);
return $files;
}
}
```

做为程序员，到 IT 企业面试的时候肯定会有笔试这关，那就要考考你的 PHP 知识了，所以本站收集一些实用的 php 面试题及答案给大家。

基础题：

1. 表单中 get 与 post 提交方法的区别？

答：get 是发送请求 HTTP 协议通过 url 参数传递进行接收，而 post 是实体数据，可以通过表单提交大量信息。

2. session 与 cookie 的区别？

答：session：储存用户访问的全局唯一变量，存储在服务器上的 php 指定的目录中的（ session\_dir ）的位置进行的存放

cookie：用来存储连续访问一个页面时所使用，是存储在客户端，对于 cookie 来说是存储在用户 WIN 的 Temp 目录中的。

两者都可通过时间来设置时间长短

3. 数据库中的事务是什么？

答：事务（ transaction ）是作为一个单元的一组有序的数据库操作。如果组中的所有操作都成功，则认为事务成功，即使只有一个操作失败，事务也不成功。如果所有操作完成，

事务则提交，其修改将作用于所有其他数据库进程。如果一个操作失败，则事务将回滚，该事务所有操作的影响都将取消。

简述题：

1、用 PHP 打印出前一天的时间格式是 2006-5-10 22:21:21 (2 分)

答：echo date('Y-m-d H:i:s', strtotime('-1 days'));

2、echo(), print(), print\_r() 的区别 (3 分)

答：echo 是 PHP 语句， print 和 print\_r 是函数，语句没有返回值，函数可以有返回值（即便没有用）

print() 只能打印出简单类型变量的值（如 int, string）

print\_r() 可以打印出复杂类型变量的值（如数组，对象）

echo 输出一个或者多个字符串

3、能够使 HTML 和 PHP 分离开使用的模板 (1 分)

答：Smarty, Dwoo, TinyButStrong, Template Lite, Savant, phemplate, XTemplate

5、使用哪些工具进行版本控制? (1 分)

答：cvs, svn, vss;

6、如何实现字符串翻转? (3 分)

答：echo strrev(\$a);

7、优化 MySQL 数据库的方法。 (4 分，多写多得)

答：

1、选取最适用的字段属性，尽可能减少定义字段长度，尽量把字段设置 NOT NULL，例如‘省份，性别’，最好设置为 ENUM

2、使用连接 ( JOIN ) 来代替子查询：

a. 删除没有任何订单客户：DELETE FROM customerinfo WHERE customerid NOT in(SELECT customerid FROM orderinfo)

b. 提取所有没有订单客户：SELECT FROM customerinfo WHERE customerid NOT in(SELECT customerid FROM orderinfo)

c. 提高 b 的速度优化：SELECT FROM customerinfo LEFT JOIN orderid  
customerinfo.customerid=orderinfo.customerid  
WHERE orderinfo.customerid IS NULL

3、使用联合(UNION)来代替手动创建的临时表

a. 创建临时表：SELECT name FROM `nametest` UNION SELECT username FROM `nametest2`

4、事务处理：

a. 保证数据完整性，例如添加和修改同时，两者成立则都执行，一者失败都失败

```
mysql_query("BEGIN");
mysql_query("INSERT INTO customerinfo (name) VALUES ('$name1')");
mysql_query("SELECT * FROM `orderinfo` where customerid='".$id") ;
mysql_query("COMMIT");
```

5、锁定表，优化事务处理：

a. 我们用一个 SELECT 语句取出初始数据，通过一些计算，用 UPDATE 语句将新值更新到表中。

包含有 WRITE 关键字的 LOCK TABLE 语句可以保证在 UNLOCK TABLES 命令被执行之前，

不会有其它的访问来对 inventory 进行插入、更新或者删除的操作

```
mysql_query("LOCK TABLE customerinfo READ, orderinfo WRITE");
mysql_query("SELECT customerid FROM `customerinfo` where id='".$id") ;
mysql_query("UPDATE `orderinfo` SET ordertitle='$title' where customerid='".$id") ;
mysql_query("UNLOCK TABLES");
```

6、使用外键，优化锁定表

a. 把 customerinfo 里的 customerid 映射到 orderinfo 里的 customerid，

任何一条没有合法的 customerid 的记录不会写到 orderinfo 里

```
CREATE TABLE customerinfo
(
    customerid INT NOT NULL,
    PRIMARY KEY(customerid)
) TYPE = INNODB;
```

---

```
CREATE TABLE orderinfo
(
    orderid INT NOT NULL,
    customerid INT NOT NULL,
    PRIMARY KEY(customerid,orderid),
    FOREIGN KEY (customerid) REFERENCES customerinfo
    (customerid) ON DELETE CASCADE
) TYPE = INNODB;
```

**注意:**'ON DELETE CASCADE',该参数保证当 customerinfo 表中的一条记录删除的话同时也会删除 order 表中的该用户的所有记录,注意使用外键要定义事务安全类型为 INNODB;

## 7、建立索引:

a. 格式:

(普通索引) ->

**创建:**CREATE INDEX <索引名> ON tablename (索引字段)

**修改:**ALTER TABLE tablename ADD INDEX [索引名] (索引字段)

**创表指定索引:**CREATE TABLE tablename ([...], INDEX[索引名] (索引字段))

(唯一索引) ->

**创建:**CREATE UNIQUE <索引名> ON tablename (索引字段)

**修改:**ALTER TABLE tablename ADD UNIQUE [索引名] (索引字段)

**创表指定索引:**CREATE TABLE tablename ([...], UNIQUE[索引名] (索引字段))

(主键) ->

它是唯一索引,一般在创建表时建立,格式为:

CREATE TABLE tablename ([...], PRIMARY KEY[索引字段])

## 8、优化查询语句

a. 最好在相同字段进行比较操作,在建立好的索引字段上尽量减少函数操作

例子 1:

SELECT \* FROM order WHERE YEAR(orderDate)<2008; (慢)

SELECT \* FROM order WHERE orderDate<"2008-01-01"; (快)

例子 2:

SELECT \* FROM order WHERE addtime/7<24; (慢)

SELECT \* FROM order WHERE addtime<24\*7; (快)

例子 3：

```
SELECT * FROM order WHERE title like "%good%";  
SELECT * FROM order WHERE title>="good" and name<"good";
```

8、PHP 的意思(送 1 分)

答：PHP 是一个基于服务端来创建动态网站的脚本语言，您可以用 PHP 和 HTML 生成网站主页

9、MySQL 取得当前时间的函数是？，格式化日期的函数是(2 分)

答：now(), date()

10、实现中文字串截取无乱码的方法。(3 分)

```
function GBsubstr($string, $start, $length) {  
    if(strlen($string)>$length){  
        $str=null;  
        $len=$start+$length;  
        for($i=$start;$i<$len;$i++){  
            if(ord(substr($string,$i,1))>0xa0){  
                $str.=substr($string,$i,2);  
                $i++;  
            }else{  
                $str.=substr($string,$i,1);  
            }  
        }  
        return $str.'...';  
    }else{  
        return $string;  
    }  
}
```

11、您是否用过版本控制软件？如果有您用的版本控制软件的名字是？(1 分)

12、您是否用过模板引擎？如果有您用的模板引擎的名字是？(1 分)

答：用过，smarty

13、请简单阐述您最得意的开发之作(4 分)

答：信息分类

14、对于大流量的网站，您采用什么样的方法来解决访问量问题？(4 分)

答：确认服务器硬件是否足够支持当前的流量，数据库读写分离，优化数据表，

程序功能规则,禁止外部的盗链,控制大文件的下载,使用不同主机分流主要流量

15、用 PHP 写出显示客户端 IP 与服务器 IP 的代码 1 分)

答:打印客户端 IP:echo \$\_SERVER[ 'REMOTE\_ADDR' ] ; 或者: getenv('REMOTE\_ADDR');

打印服务器 IP:echo gethostbyname("www.bolaiwu.com")

16、语句 include 和 require 的区别是什么?为避免多次包含同一文件,可用(?)语句代替它们? (2 分)

答:require->require 是无条件包含也就是如果一个流程里加入 require,无论条件成立与否都会先执行 require  
include->include 有返回值,而 require 没有(可能因为如此 require 的速度比 include 快)

注意:包含文件不存在或者语法错误的时候 require 是致命的,include 不是

17、如何修改 SESSION 的生存时间(1 分) .

答:方法 1:将 php.ini 中的 session.gc\_maxlifetime 设置为 9999 重启 apache

方法 2:\$savePath = "./session\_save\_dir/";  
\$lifeTime = 小时 \* 秒;  
session\_save\_path(\$savePath);  
session\_set\_cookie\_params(\$lifeTime);  
session\_start();

方法 3:setcookie() and session\_set\_cookie\_params(\$lifeTime);

18、有一个网页地址,比如 PHP 开发资源网主页: http://www.phpres.com/index.html,如何得到它的内容? (\$1 分)

答:方法 1(对于 PHP5 及更高版本):

```
$readcontents = fopen("http://www.phpres.com/index.html", "rb");  
$contents = stream_get_contents($readcontents);  
fclose($readcontents);  
echo $contents;
```

方法 2:

```
echo file_get_contents("http://www.phpres.com/index.html");
```

19、在 HTTP 1.0 中,状态码 401 的含义是(?) ;如果返回“找不到文件”的提示,则可用 header 函数,其语句为(?) ; (2 分)

答:状态 401 代表未被授权,header("Location:www.xxx.php");

12、在 PHP 中,heredoc 是一种特殊的字符串,它的结束标志必须? (1 分)

答:heredoc 的语法是用"<<<"加上自己定义成对的标签,在标签范围内的文字视为一个字符串

例子：

```
$str = <<<SHOW
my name is Jiang Qihui!
SHOW;
```

13、谈谈 asp,php,jsp 的优缺点(1 分)

答:ASP 全名 Active Server Pages , 是一个 WEB 服务器端的开发环境 , 利用它可以产生和运行动态的、交互的、高性能的 WEB 服务应用程序。 ASP 采用脚本语言 VB Script ( Java script ) 作为自己的开发语言。

PHP 是一种跨平台的服务器端的嵌入式脚本语言 . 它大量地借用 C, Java 和 Perl 语言的语法 , 并耦合 PHP 自己的特性 , 使 WEB 开发者能够快速地写出动态生成页面 . 它支持目前绝大多数数据库。还有一点 , PHP 是完全免费的 , 不用花钱 , 你可以从 PHP 官方站点 (<http://www.php.net>) 自由下载。而且你可以不受限制地获得源码 , 甚至可以从中加进你自己需要的特色。

JSP 是 Sun 公司推出的新一代站点开发语言 , 他完全解决了目前 ASP, PHP 的一个通病 - - 脚本级执行 ( 据说 PHP4 也已经在 Zend 的支持下 , 实现编译运行 ) . Sun 公司借助自己在 Java 上的不凡造诣 , 将 Java 从 Java 应用程序 和 Java Applet 之外 , 又有新的硕果 , 就是 Jsp -- Java Server Page. Jsp 可以在 Serverlet 和 JavaBean 的支持下 , 完成功能强大的站点程序。

三者都提供在 HTML 代码中混合某种程序代码、由语言引擎解释执行程序代码的能力。但 JSP 代码被编译成 Servlet 并由 Java 虚拟机解释执行 , 这种编译操作仅在对 JSP 页面的第一次请求时发生。在 ASP 、 PHP 、 JSP 环境下 , HTML 代码主要负责描述信息的显示样式 , 而程序代码则用来描述处理逻辑。普通的 HTML 页面只依赖于 Web 服务器 , 而 ASP 、 PHP 、 JSP 页面需要附加的语言引擎分析和执行程序代码。程序代码的执行结果被重新嵌入到 HTML 代码中 , 然后一起发送给浏览器。 ASP 、 PHP 、 JSP 三者都是面向 Web 服务器的技术 , 客户端浏览器不需要任何附加的软件支持。

14、谈谈对 mvc 的认识(1 分)

答:由模型 (model) , 视图 (view) , 控制器 (controller) 完成的应用程序

由模型发出要实现的功能到控制器 , 控制器接收组织功能传递给视图 ;

15、写出发贴数最多的十个人名字的 SQL , 利用下表 : members (id, username, posts, pass, email) (2 分)

答:SELECT \* FROM `members` ORDER BY posts DESC limit 0,10;

16. 请说明 php 中传值与传引用的区别。什么时候传值什么时候传引用? (2 分)

答:按值传递：函数范围内对值的任何改变在函数外部都会被忽略

按引用传递：函数范围内对值的任何改变在函数外部也能反映出这些修改

优缺点：按值传递时，php 必须复制值。特别是对于大型的字符串和对象来说，这将会是一个代价很大的操作。

按引用传递则不需要复制值，对于性能提高很有好处。

17. 在 PHP 中 error\_reporting 这个函数有什么作用? (1 分)

答:设置错误级别与错误信息回报

18. 请写一个函数验证电子邮件的格式是否正确 (2 分)

答:function checkEmail(\$email)

```
{  
    $pregEmail  
    "/([a-z0-9]*[-_.]?[a-z0-9]+)*@[a-z0-9]*[-_.]?[a-z0-9]+)+[\.\.][a-z]{2,3}([\.\.][a-z]{2})?/i";  
    return preg_match($pregEmail,$email);  
}
```

19. 简述如何得到当前执行脚本路径，包括所得到参数。 (2 分)

答:\$script\_name = basename(\_\_FILE\_\_); print\_r(\$script\_name);

21、JS 表单弹出对话框函数是?获得输入焦点函数是? (2 分)

答:弹出对话框: alert(), prompt(), confirm()

获得输入焦点 focus()

22、JS 的转向函数是?怎么引入一个外部 JS 文件? (2 分)

答>window.location.href,<script type="text/javascript" src="js/js\_function.js"></script>

23、foo() 和 @foo() 之间有什么区别? (1 分)

答:@foo() 控制错误输出

24、如何声明一个名为“ myclass” 的没有方法和属性的类? (1 分)

答:class myclass{ }

25、如何实例化一个名为“ myclass” 的对象? (1 分)

答:new myclass()

26、你如何访问和设置一个类的属性? (2 分)

答:\$object = new myclass();  
\$newstr = \$object->test;  
\$object->test = "info";

27、mysql\_fetch\_row() 和 mysql\_fetch\_array 之间有什么区别? (1 分)

答:mysql\_fetch\_row 是从结果集取出 1 行数组,作为枚举

mysql\_fetch\_array 是从结果集取出一行数组作为关联数组,或数字数组,两者兼得

28、GD 库是做什么用的? (1 分)

答:gd 库提供了一系列用来处理图片的 API , 使用 GD 库可以处理图片,或者生成图片。

在网站上 GD 库通常用来生成缩略图或者用来对图片加水印或者对网站数据生成报表。

29、指出一些在 PHP 输入一段 HTML 代码的办法。 (1 分)

答:echo "<a href='index.php'>aaa</a>";

30、下面哪个函数可以打开一个文件,以对文件进行读和写操作? (1 分)

- (a) fget() (b) file\_open() (c) fopen() (d) open\_file() [ c ]

31、下面哪个选项没有将 john 添加到 users 数组中? (1 分)

- (a) \$users[] = 'john' ;  
(b) array\_add(\$users,' john' );  
(c) array\_push(\$users, 'john' );  
(d) \$users ||= 'john' ; [ a , c ]

32、下面的程序会输入是否? (1 分)

```
$num = 10;
function multiply(){
$num = $num * 10;
}
multiply();
echo $num;
?>
```

输出:10

33、使用 php 写一段简单查询,查出所有姓名为 “张三” 的内容并打印出来 (2 分)

表名 User

Name Tel Content Date

张三 13333663366 大专毕业 2006-10-11

张三 13612312331 本科毕业 2006-10-15

张四 021-55665566 中专毕业 2006-10-15

请根据上面的题目完成代码：

```
$mysql_db=mysql_connect("local","root","pass");
@mysql_select_db("DB",$mysql_db);

$result = mysql_query("SELECT * FROM `user` WHERE name='张三'");
while($rs = mysql_fetch_array($result)){
    echo $rs["tel"].$rs["content"].$rs["date"];
}
```

34、如何使用下面的类，并解释下面什么意思？(3)

```
class test{
    function Get_test($num){
        $num=md5(md5($num)."En");
        return $num;
    }
}
```

答:\$testnum = "123";

```
$object = new test();
$encrypt = $object->Get_test($testnum);
echo $encrypt;
```

类 test 里面包含 Get\_test 方法，实例化类调用方法多字符串加密

35、写出 SQL 语句的格式：插入，更新，删除（4 分）

表名 User

Name Tel Content Date

张三 13333663366 大专毕业 2006-10-11

张三 13612312331 本科毕业 2006-10-15

张四 021-55665566 中专毕业 2006-10-15

(a) 有一新记录(小王 13254748547 高中毕业 2007-05-06) 请用 SQL 语句新增至表中

```
mysql_query("INSERT INTO `user` (name,tel,content,date) VALUES
('小王','13254748547','高中毕业','2007-05-06')")
```

(b) 请用 sql 语句把张三的时间更新成为当前系统时间

```
$nowDate = date("Ymd");
mysql_query("UPDATE `user` SET date='".$nowDate."' WHERE name='张山');
```

(c) 请写出删除名为张四的全部记录

```
mysql_query("DELETE FROM `user` WHERE name='张四'");
```

36、请写出数据类型(int char varchar datetime text)的意思；请问 varchar 和 char 有什么区别(2分)

答:int 是数字类型, char 固定长度字符串, varchar 实际长度字符串, datetime 日期时间型, text 文本字符串

char 的场地固定为创建表设置的长度, varchar 为可变长度的字符

38、写出以下程序的输出结果 (1 分)

```
$b=201;  
$c=40;  
$a=$b>$c?4:5;  
echo $a;  
?>
```

答:4

39、检测一个变量是否有设置的函数是否?是否为空的函数是?(2 分)

答:isset(\$str),empty(\$str);

40、取得查询结果集总数的函数是?(1 分)

答:mysql\_num\_rows(\$result);

41、\$arr = array('james', 'tom', 'symfony'); 请打印出第一个元素的值 (1 分)

答:echo \$array[0];

42、请将 41 题的数组的值用','号分隔并合并成字符串输出(1 分)

答:for(\$i=0;\$i<count(\$array);\$i++){ echo \$array[\$i].","; }

43、\$a = 'abcdef'; 请取出\$a 的值并打印出第一个字母(1 分)

答:echo \$a{0} 或 echo substr(\$a,0,1)

44、PHP 可以和 sql server/oracle 等数据库连接吗?(1 分)

答:当然可以

45、请写出 PHP5 权限控制修饰符(3 分)

答:public(公共),private(私用),protected(继承)

46、请写出 php5 的构造函数和析构函数(2 分)

答:\_\_construct , \_\_destruct

47、完成以下:

(一) 创建新闻发布系统，表名为 message 有如下字段 (3 分)

id 文章 id

title 文章标题

content 文章内容

category\_id 文章分类 id

hits 点击量

答:CREATE TABLE 'message' (

```
'id' int(10) NOT NULL auto_increment,  
'title' varchar(200) default NULL,  
'content' text,  
'category_id' int(10) NOT NULL,  
'hits' int(20),  
PRIMARY KEY('id');  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

(二) 同样上述新闻发布系统：表 comment 记录用户回复内容，字段如下 (4 分)

comment\_id 回复 id

id 文章 id，关联 message 表中的 id

comment\_content 回复内容

现通过查询数据库需要得到以下格式的文章标题列表，并按照回复数量排序，回复最高的排在最前面

文章 id 文章标题 点击量 回复数量

用一个 SQL 语句完成上述查询，如果文章没有回复则回复数量显示为 0

答:SELECT message.id id,message.title title,IF(message.`hits` IS NULL,0,message.`hits`) hits,  
IF(comment.`id` is NULL,0,count(\*)) number FROM message LEFT JOIN  
comment ON message.id=comment.id GROUP BY message.`id`;

(三) 上述内容管理系统，表 category 保存分类信息，字段如下 (3 分)

```
category_id int(4) not null auto_increment;  
category_name varchar(40) not null;
```

用户输入文章时，通过选择下拉菜单选定文章分类

写出如何实现这个下拉菜单

答:function categoryList()

```
{  
    $result=mysql_query("select category_id,category_name from category")
```

```
        or die("Invalid query: " . mysql_error());
print("<select name='category' value=''>\n");
while($rowArray=mysql_fetch_array($result))
{
    print("<option
value='".$rowArray['category_id']."'".$rowArray['categroy_name']."'</option>\n");
}
print("</select>");
}
```

编程题：

1. 写一个函数，尽可能高效的，从一个标准 url 里取出文件的扩展名

例如：http://www.sina.com.cn/abc/de/fg.php?id=1 需要取出 php 或 .php

答案 1：

```
function getExt($url){
$arr = parse_url($url);

$file = basename($arr['path']);
$ext = explode(".", $file);
return $ext[1];
}
```

答案 2：

```
function getExt($url) {
$url = basename($url);
$pos1 = strpos($url, ".");
$pos2 = strpos($url, "?");
if(strstr($url, "?")){
    return substr($url, $pos1 + 1, $pos2 - $pos1 - 1);
} else {
    return substr($url, $pos1);
}
}
```

2. 在 HTML 语言中，页面头部的 meta 标记可以用来输出文件的编码格式，以下是一个标准的 meta 语句

请使用 PHP 语言写一个函数，把一个标准 HTML 页面中的类似 meta 标记中的 charset 部分值改为 big5

请注意：

1. 需要处理完整的 html 页面，即不光此 meta 语句

2. 忽略大小写
  3. ' 和 " 在此处是可以互换的
  4. 'Content-Type' 两侧的引号是可以忽略的，但 'text/html; charset=gbk' 两侧的不行
  5. 注意处理多余空格
3. 写一个函数，算出两个文件的相对路径

如 \$a = '/a/b/c/d/e.php';  
\$b = '/a/b/12/34/c.php';

计算出 \$b 相对于 \$a 的相对路径应该是 ../../c/d 将()添上

答:

```
function getRelativePath($a, $b) {  
    $returnPath = array(dirname($b));  
    $arrA = explode('/', $a);  
    $arrB = explode('/', $returnPath[0]);  
    for ($n = 1, $len = count($arrB); $n < $len; $n++) {  
        if ($arrA[$n] != $arrB[$n]) {  
            break;  
        }  
    }  
    if ($len - $n > 0) {  
        $returnPath = array_merge($returnPath, array_fill(1, $len - $n, '..'));  
    }  
  
    $returnPath = array_merge($returnPath, array_slice($arrA, $n));  
    return implode('/', $returnPath);  
}  
  
echo getRelativePath($a, $b);
```

填空题：

1. 在 PHP 中，当前脚本的名称（不包括路径和查询字符串）记录在预定义变量 `$_SERVER['PHP_SELF']` 中；而链接到当前页面的 URL 记录在预定义变量 `$_SERVER['HTTP_REFERER']` 中。
2. 执行程序段 `<?php echo 8%(-2) ?>` 将输出 `0`。
3. 在 HTTP 1.0 中，状态码 401 的含义是 \_\_\_\_；如果返回“找不到文件”的提示，则可用 `header` 函数，其语句为 \_\_\_\_。
4. 数组函数 `arsort` 的作用是 \_\_\_\_ 对数组进行逆向排序并保持索引关系 \_\_\_\_；语句 `error_reporting(2047)` 的作用是 \_\_\_\_ 报告所有错误和警告 \_\_\_\_。

- 
5. PEAR 中的数据库连接字符串格式是\_\_\_\_\_。
6. 写出一个正则表达式，过滤网页上的所有 JS/VBS 脚本（即把 script 标记及其内容都去掉）:preg\_replace("/<script[^>].\*?>.\*?</script>/si", "newinfo", \$script);
7. 以 Apache 模块的方式安装 PHP，在文件 http.conf 中首先要用语句\_\_\_\_\_动态装载 PHP 模块，然后再用语句\_\_\_\_\_使得 Apache 把所有扩展名为 php 的文件都作为 PHP 脚本处理。
- LoadModule php5\_module "c:/php/php5apache2.dll" , AddType application/x-httdp-php .php,
8. 语句 include 和 require 都能把另外一个文件包含到当前文件中，它们的区别是\_\_\_\_\_；为了避免多次包含同一文件，可以用语句\_\_\_\_require\_once||include\_once\_\_\_\_\_来代替它们。
9. 类的属性可以序列化后保存到 session 中，从而以后可以恢复整个类，这要用到的函数是\_\_\_\_\_。
10. 一个函数的参数不能是对变量的引用，除非在 php.ini 中把 allow\_call\_time\_pass\_reference boolean 设为 on。
11. SQL 中 LEFT JOIN 的含义是\_\_\_\_\_自然左外链接\_\_\_\_。如果 tbl\_user 记录了学生的姓名 (name) 和学号 (ID) , tbl\_score 记录了学生 (有的学生考试以后被开除了，没有其记录) 的学号 (ID) 和考试成绩 (score) 以及考试科目 (subject) , 要想打印出各个学生姓名及对应的各科总成绩，则可以用 SQL 语句\_\_\_\_\_。
12. 在 PHP 中，heredoc 是一种特殊的字符串，它的结束标志必须\_\_\_\_\_。

编程题：

13. 写一个函数，能够遍历一个文件夹下的所有文件和子文件夹。

答：

```
function my_scandir($dir)
{
    $files = array();
    if ( $handle = opendir($dir) ) {
        while ( ($file = readdir($handle)) != false ) {
            if ( $file != ".." && $file != "." ) {
                if ( is_dir($dir . "/" . $file) ) {
                    $files[$file] = scandir($dir . "/" . $file);
                }else {
                    $files[] = $file;
                }
            }
        }
        closedir($handle);
    }
    return $files;
}
```

```
}
```

14. 简述论坛中无限分类的实现原理。

答：

```
<?php
/*
```

数据表结构如下：

```
CREATE TABLE `category` (
  `categoryID` smallint(5) unsigned NOT NULL auto_increment,
  `categoryParentID` smallint(5) unsigned NOT NULL default '0',
  `categoryName` varchar(50) NOT NULL default '',
  PRIMARY KEY (`categoryID`)
) ENGINE=MyISAM DEFAULT CHARSET=gbk;
INSERT INTO `category` ( `categoryParentID` , `categoryName` ) VALUES
(0, '一级类别'),
(1, '二级类别'),
(1, '二级类别'),
(1, '二级类别'),
(2, '三级类别'),
(2, '333332'),
(2, '234234'),
(3, 'aqqqqqd'),
(4, '哈哈'),
(5, '66333666');
*/
```

//指定分类 id 变量\$category\_id,然后返回该分类的所有子类

//\$default\_category 为默认的选中的分类

```
function Get_Category($category_id = 0,$level = 0, $default_category = 0)
{
    global $DB;
    $sql = "SELECT * FROM category ORDER BY categoryID DESC";
    $result = $DB->query( $sql );
    while ($rows = $DB->fetch_array($result))
    {
        $category_array[$rows[categoryParentID]] [$rows[categoryID]] = array('id' => $rows[categoryID],
        'parent' => $rows[categoryParentID], 'name' => $rows
```

```

[categoryName]);
}
if (!isset($category_array[$category_id]))
{
return "";
}
foreach($category_array[$category_id] AS $key => $category)
{
if ($category['id'] == $default_category)
{
echo "<option selected value=\"".$category['id']."'\"";
}else
{
echo "<option value=\"".$category['id']."'\"";
}
if ($level > 0)
{
echo ">" . str_repeat( " ", $level ) . " " . $category['name'] . "</option>\n";
}
else
{
echo ">" . $category['name'] . "</option>\n";
}
Get_Category($key, $level + 1, $default_category);
}
unset($category_array[$category_id]);
}
/*

```

函数返回的数组格式如下所示：

```

Array
(
    [1] => Array ( [id] => 1 [name] => 一级类别 [level] => 0 [ParentID] => 0 )
    [4] => Array ( [id] => 4 [name] => 二级类别 [level] => 1 [ParentID] => 1 )
    [9] => Array ( [id] => 9 [name] => 哈哈 [level] => 2 [ParentID] => 4 )
    [3] => Array ( [id] => 3 [name] => 二级类别 [level] => 1 [ParentID] => 1 )
    [8] => Array ( [id] => 8 [name] => aqqqqqd [level] => 2 [ParentID] => 3 )
    [2] => Array ( [id] => 2 [name] => 二级类别 [level] => 1 [ParentID] => 1 )
    [7] => Array ( [id] => 7 [name] => 234234 [level] => 2 [ParentID] => 2 )
)

```

```
[6] => Array ( [id] => 6 [name] => 333332 [level] => 2 [ParentID] => 2 )
[5] => Array ( [id] => 5 [name] => 三级类别 [level] => 2 [ParentID] => 2 )
[10] => Array ( [id] => 10 [name] => 66333666 [level] => 3 [ParentID] => 5 )
)
*/
//指定分类 id,然后返回数组
function Category_array($category_id = 0,$level=0)
{
    global $DB;
    $sql = "SELECT * FROM category ORDER BY categoryID DESC";
    $result = $DB->query($sql);
    while ($rows = $DB->fetch_array($result))
    {
        $category_array[$rows['categoryParentID']] [$rows['categoryID']] = $rows;
    }
    foreach ($category_array AS $key=>$val)
    {
        if ($key == $category_id)
        {
            foreach ($val AS $k=> $v)
            {
                $options[$k] =
                    array(
                        'id'      => $v['categoryID'],      'name'      => $v['categoryName'],      'level'      => $level,
                        'ParentID'=>$v['categoryParentID']
                    );
                $children = Category_array($k, $level+1);
                if (count($children) > 0)
                {
                    $options = $options + $children;
                }
            }
        }
    }
    unset($category_array[$category_id]);
    return $options;
}
?>
<?php
class cate
```

```

{
    function Get_Category($category_id = 0, $level = 0, $default_category = 0)
    {
        echo $category_id;
        $arr = array(
            '0' => array(
                '1' => array('id' => 1, 'parent' => 0, 'name' => '1111'),
                '2' => array('id' => 2, 'parent' => 0, 'name' => '2222'),
                '4' => array('id' => 4, 'parent' => 0, 'name' => '4444')
            ),
            '1' => array(
                '3' => array('id' => 3, 'parent' => 1, 'name' => '333333'),
                '5' => array('id' => 5, 'parent' => 1, 'name' => '555555')
            ),
            '3' => array(
                '6' => array('id' => 6, 'parent' => 3, 'name' => '66666'),
                '7' => array('id' => 7, 'parent' => 3, 'name' => '77777')
            ),
            '4' => array(
                '8' => array('id' => 8, 'parent' => 4, 'name' => '8888'),
                '9' => array('id' => 9, 'parent' => 4, 'name' => '9999')
            )
        );
        if (!isset($arr[$category_id]))
        {
            return "";
        }

        foreach ($arr[$category_id] AS $key => $cate)
        {
            if ($cate['id'] == $default_category)
            {
                $txt = "<option selected value=\"$cate[id]\">";
            }else{
                $txt = "<option value=\"$cate[id]\">";
            }

            if ($level > 0)
            {
                $txt1 = ">" . str_repeat( "-", $level ) . " " . $cate['name'] . "</option>\n";
            }
        }
    }
}

```

```
        }else{
            $txt1 = ">" . $cate['name'] . "</option>\n";
        }
        $val = $txt.$txt1;
        echo $val;
        self::Get_Category($key, $level + 1, $default_category);
    }

}

function getFlush($category_id = 0,$level = 0, $default_category = 0)
{
    ob_start();
    self::Get_Category($category_id , $level, $default_category);
    $out = ob_get_contents();
    ob_end_clean();
    return $out;
}
$id = $_GET['id'];
echo "<select>";
$c = new cate();
// $c->Get_Category();
$ttt= $c->getFlush($id, '0', '3');
echo $ttt;
echo "</select>";
?>
```

## 1、PHP 的意思？

答:PHP 是一个基于服务端来创建动态网站的脚本语言，您可以用 PHP 和 HTML 生成网站主页

## 2、谈谈 asp,php,jsp 的优缺点？

答:ASP 全名 Active Server Pages ,是一个 WEB 服务器端的开发环境， 利用它可以产生和运行动态的、交互的、高性能的 WEB 服务应用程序。ASP 采用脚本语言 VB Script ( Java script ) 作为自己的开发语言。

---

PHP 是一种跨平台的服务器端的嵌入式脚本语言。它大量地借用 C,Java 和 Perl 语言的语法，并耦合 PHP 自己的特性，使 WEB 开发者能够快速地写出动态生成页面。它支持目前绝大多数数据库。还有一点，PHP 是完全免费的，不用花钱，你可以从 PHP 官方站点(<http://www.php.net>)自由下载。而且你可以不受限制地获得源码，甚至可以从中加进你自己需要的特色。

JSP 是 Sun 公司推出的新一代站点开发语言，他完全解决了目前 ASP,PHP 的一个通病——脚本级执行（据说 PHP4 也在 Zend 的支持下，实现编译运行）。Sun 公司借助自己在 Java 上的不凡造诣，将 Java 从 Java 应用程序 和 Java Applet 之外，又有新的硕果，就是 Jsp —— Java Server Page, Jsp 可以在 Serverlet 和 JavaBean 的支持下，完成功能强大的站点程序。

三者都提供在 HTML 代码中混合某种程序代码、由语言引擎解释执行程序代码的能力。但 JSP 代码被编译成 Servlet 并由 Java 虚拟机解释执行，这种编译操作仅在对 JSP 页面的第一次请求时发生。在 ASP 、 PHP、 JSP 环境下，HTML 代码主要负责描述信息的显示样式，而程序代码则用来描述处理逻辑。普通的 HTML 页面只依赖于 Web 服务器，而 ASP 、 PHP、 JSP 页面需要附加的语言引擎分析和执行程序代码。程序代码的执行结果被重新嵌入到 HTML 代码中，然后一起发送给浏览器。 ASP 、 PHP、 JSP 三者都是面向 Web 服务器的技术，客户端浏览器不需要任何附加的软件支持。

### 3、谈谈对 mvc 的认识？

答：由模型(Model), 视图(View), 控制器(Controller)完成的应用程序

由模型发出要实现的功能到控制器, 控制器接收组织功能传递给视图;

### 4、写出发贴数最多的十个人名字的 SQL，利用下表： members(id,username,posts,pass,email)

答：SELECT \* FROM `members` ORDER BY posts DESC limit 0,10;

### 5、GD 库是做什么用的？

答：gd 库提供了一系列用来处理图片的功能，使用 GD 库可以处理图片，或者生成图片。

在网站上 GD 库通常用来生成缩略图或者用来对图片加水印或者对网站数据生成报表。

## 6、请写出数据类型(int char varchar datetime text)的意思；请问 varchar 和 char 有什么别？

答：int 是数字类型，char 固定长度字符串，varchar 实际长度字符串，datetime 日期时间型，text 文本字符串

char 的场地固定为创建表设置的长度，varchar 为可变长度的字符

1. 在 PHP 中，当前脚本的名称（不包括路径和查询字符串）记录在预定义变量（1）中；而链接到当前页面的 URL 记录在预定义变量（2）中。

2. 执行程序段<?php echo 8%(-2) ?>将输出（3）。

3. 在 HTTP 1.0 中，状态码 401 的含义是（4）；如果返回“找不到文件”的提示，则可用 header 函数，其语句为（5）。

4. 数组函数 arsort 的作用是（6）；语句 error\_reporting(2047) 的作用是（7）。

5. PEAR 中的数据库连接字符串格式是（8）。

6. 写出一个正则表达式，过滤网页上的所有 JS/VBS 脚本（即把标记及其内容都去掉）：（9）。

7. 以 Apache 模块的方式安装 PHP，在文件 http.conf 中首先要用语句（10）动态装载 PHP 模块，  
然后再用语句（11）使得 Apache 把所有扩展名为 php 的文件都作为 PHP 脚本处理。

8. 语句 include 和 require 都能把另外一个文件包含到当前文件中，它们的区别是（12）；为了避免多次包含同一文件，可以用语句（13）来代替它们。

9. 类的属性可以序列化后保存到 session 中，从而以后可以恢复整个类，这要用到的函数是（14）。

10. 一个函数的参数不能是对变量的引用，除非在 php.ini 中把（15）设为 on。

11. SQL 中 LEFT JOIN 的含义是（16）。

如果 tbl\_user 记录了学生的姓名(name)和学号(ID)，  
tbl\_score 记录了学生（有的学生考试以后被开除了，没有其记录）的学号(ID)和考试成绩(score)以及考试科目(subject），  
要想打印出各个学生姓名及对应的各科总成绩，则可以用 SQL 语句（17）。

12. 在 PHP 中，heredoc 是一种特殊的字符串，它的结束标志必须（18）。

13. 写一个函数，能够遍历一个文件夹下的所有文件和子文件夹。

14. 简述论坛中无限分类的实现原理。

15. 设计一个网页，使得打开它时弹出一个全屏的窗口，该窗口中有一个文本框和一个按钮。用户在文本框中输入信息后点击按钮就可以把窗口关闭，而输入的信息却在主网页中显示。

答案（填空）：

1. echo \$\_SERVER['PHP\_SELF']; echo \$\_SERVER['HTTP\_REFERER'];

2. 0

3. (4)未授权 (5) header("HTTP/1.0 404 Not Found");

4. (6)对数组进行逆向排序并保持索引关系 (7)All errors and warnings

5. 没弄明白

6. /<[^>].\*?>.\*?</>/si

7.(10) LoadModule php5\_module "D:/xampp/apache/bin/php5apache2.dll"

(11) AddType application/x-httdp-php-source .phps

AddType application/x-httdp-php .php .php5 .php4 .php3 .phtml

8.(12) 发生异常时 include 产生警告 require 产生致命错误 (13) require\_once()/include\_once()

9. serialize() /unserialize()

10. allow\_call\_time\_pass\_reference

11. (16) 自然左外连接

(17) select name , count(score) as sum\_score from tbl\_user left join tbl\_score on tbl\_user.ID=tbl\_score.ID group by tbl\_user.ID

12. 结束标识符所在的行不能包含任何其它字符除";"

13.

/\*\*

\* 遍历目录，结果存入数组。支持 php4 及以上。php5 以后可用 scandir() 函数代替 while 循环。

\* @param string \$dir

\* @return array

\*/

my\_scandir(\$dir)

{

\$files = array();

if ( \$handle = opendir(\$dir) )

{

while ( (\$file = readdir(\$handle)) !== false )

{

if ( \$file != ".." && \$file != "." )

{

```
if ( is_dir($dir . "/" . $file) )
{
    $files[$file] = rec_scandir($dir . "/" . $file);
}
else
{
    $files[] = $file;
}
}

closedir($handle);
return $files;
}
}
```

=====

PHP 面试题系列二【附答案】

- 1、用 PHP 打印出前一天的时间格式是 2006-5-10 22:21:21
- 2、echo(),print(),print\_r()的区别
- 3、能够使 HTML 和 PHP 分离开使用的模板
- 4、如何实现 PHP、JSP 交互？
- 5、使用哪些工具进行版本控制？
- 6、如何实现字符串翻转？
- 7、优化 MYSQL 数据库的方法。
- 8、谈谈事务处理
- 9、apache+mysql+php 实现最大负载的方法
- 10、实现中文字串截取无乱码的方法。

答案：

1.echo date('Y-m-d H:i:s', strtotime('-1 day'));

2.echo 是语言结构，无返回值;print 功能和 echo 基本相同，不同的是 print 是函数，有返回值;print\_r 是递归打印，用于输出数组对象

3.so much, 其实 PHP 本身就是一种模版引擎，我用过的是 smarty，常见的还有 PHPLib, FastTemplate, Savant 这里有个模版引擎列表：  
<http://www.sitepoint.com/forums/showthread.php?t=123769>

4. 题目有点含糊不清, SOAP, XML\_RPC, Socket, CURL 都可以实现这些, 如果是考 PHP 和 Java 的整合, PHP 内置了这种机制(如果考 PHP 和.NET 的整合, 也可以这么回答), 例如\$foo = new Java('java.lang.System');

5. CVS 和 SVN, SVN 号称下一代 CVS, 功能强大, 不过 CVS 是老牌, 市占率很高. 我一直用 SVN, 题目是问用什么工具, 呃, 这个可能需要这么回答: CVS Server on Apache 作服务端, WinCVS 作客户端; Subversion on Apache/DAV 做服务端, TortoiseSVN 做客户端, 或者 Subclipse 做客户端.

6.用 strrev 函数呗,不准用 PHP 内置的就自己写:

```
strrev($str)
{
    $len=strlen($str);
    $newstr = "";
    for($i=$len;$i>=0;$i--)
    {
        $newstr .= $str{$i};
    }
    return $newstr;
}
```

7.高考政治题,把你知道的知识点都写上吧,我的答案:

(1).数据库设计方面,这是 DBA 和 Architect 的责任,设计结构良好的数据库,必要的时候,去正规化(英文是这个:denormalize,中文翻译成啥我不知道),允许部分数据冗余,避免 JOIN 操作,以提高查询效率

(2).系统架构设计方面,表散列,把海量数据散列到几个不同的表里面.快慢表,快表只留最新数据,慢表是历史存档.集群,主服务器 Read & write,从服务器 read only,或者 N 台服务器,各机器互为 Master

(3).(1)和(2)超越 PHP Programmer 的要求了,会更好,不会没关系.检查有没有少加索引

(4).写高效的 SQL 语句,看看有没有写低效的 SQL 语句,比如生成笛卡尔积的全连接啊,大量的 Group By 和 order by,没有 limit 等等.必要的时候,把数据库逻辑封装到 DBMS 端的存储过程里面.缓存查询结果,explain 每一个 sql 语句

(5). 所得皆必须,只从数据库取必需的数据,比如查询某篇文章的评论数,select count(\*) ... where article\_id = ? 就可以了,不要先 select \* ... where article\_id = ?然后 mysql\_num\_rows.

只传送必须的 SQL 语句,比如修改文章的时候,如果用户只修改了标题,那就 update ... set title = ? where article\_id = ?不要 set content = ?(大文本)

(6).必要的时候用不同的存储引擎.比如 InnoDB 可以减少死锁. HEAP 可以提高一个数量级的查询速度.

8.如同是个编程语言都会有答应 Hello World 的例子一样,是本数据库的教材都会讲 A 给 B 的账户转账 50 美元的例子,回答这个就好了.不过据我所知,用 MySQL 的企业,很少用 MySQL 来实现事务处理.何况现在 Oracle 收购了 InnoDB 的公司.

9.参见第七题的答案,那个地方搞好了这个问题就迎刃而解了.

10.哈哈哈,我猜出题的人是不是被 substr 的中文处理问题烦恼很久了,是不是还用了网上流传的用正则匹配中文字符然后截取的函数,其实,有非常简单的解决方法:mb\_substr()

### =====

### PHP 面试题系列三【附答案】

1、使用 php 写一段简单查询,查出所有姓名为“张三”的内容并打印出来

表名 User

Name	Tel	Content	Date
张三	13333663366	大专毕业	2006-10-11
张三	13612312331	本科毕业	2006-10-15

---

张四 021-55665566 中专毕业 2006-10-15

2、请根据上面的题目完成代码：

```
$mysql_db=mysql_connect("local","root","pass");
@mysql_select_db("DB",$mysql_db);
```

3、如何使用下面的类，并解释下面什么意思？

```
class test
{
    Get_test($num)
    {
        $num=md5(md5($num)."En");
        return $num;
    }
}
```

4、用 javascript 打印“上海爱吉”

5、写出 SQL 语句的格式：插入，更新，删除

6、谈谈对你 PHP 认识或你擅长的技术？

答案：

1. SELECT Name,Tel,Content,Date FROM User WHERE Name='张三'

2.

```
$mysql_db = mysql_connect("local","root","pass");
@mysql_select_db("DB",$mysql_db);

$sql = "SELECT Name,Tel,Content,Date FROM User WHERE Name='张三'";
$result = mysql_query($sql);
while ($row = mysql_fetch_array($result))
{
    echo $row['Name'] . ' ' . $row['Tel'] . ' ' . $row['Content'] . ' ' . $row['Date'] . "<br>\r\n";
}
```

3.

用法：

```
$get_test = new test();
$result = $get_test->Get_test(2);
```

将\$num 变量进行两次 md5 后返回，第 2 次的 md5 中的参数，在第一次 md5(\$num)后多加了 En

4.

<>

```
write('上海爱吉');
</>
```

5.

插入

```
INSERT INTO table (a1,a2,a3) S ($val1, '$val2', $val3);
```

修改

```
UPDATE table SET a1=$a1, a2='$a2' WHERE id=3;
```

删除

```
DELETE FROM table WHERE id=3;
```

6.

自己发挥

---

=====

PHP 面试题系列四【附答案】

1、假定要使用 Apache+Php 的配置，并将 php3 编译成 Apache 的一个模块。那么以下 httpd.conf 文件的语句是必须的：【】

- A、AddModule mod\_php3.c
- B、LoadModule php3\_module libexec/libphp3.so
- C、AddType application/x-httpd-php3.php3
- D、setup
- E、make install

2、PHP 支持的数据类型有七种,以下被支持的有: 【】

- A、array
- B、floating-point numbers(double)
- C、integer
- D、date
- E、string

3、以下程序:

```
<HTML>
<HEAD>
<TITLE></TITLE>
<HEAD>
<BODY>
<?php
$num1 = 15;
$num2 = $num1;
echo "<p>$num2</p>";
$num2 = &$num1;
$num2 = 20;
echo "<p>$num1</p>";
?>
</BODY>
</HTML>
```

程序输出为：[ ]

- A、15
- B、35
- C、20
- D、5

4、以下程序

```
<HTML>
<HEAD>
<TITLE></TITLE>
<HEAD>
<BODY>
<?php
    $str1 = "01";
    $str1++;
    $str1 += 1;
    echo "<p>/$str1 => $str1</p>";
?>
</BODY>
</HTML>
```

程序输出为：[ ]

- A、\$str1 => 01
- B、\$str1 => 2
- C、\$str1 => 03
- D、\$str1 => 3
- E、\$str1 => 1

5、全局变量与局部变量

```
$a=1;
sum()
{
    echo $a;
}
sum();
```

程序输出为：[ ]

- A、1
- B、10
- C、100
- D、1000
- E、空值

6、PHP 的控制语句

```
<?php
```

```
$a = 3;  
$b = $a++;  
if ($a > $b)  
{  
    echo "a 比 b 大";  
}  
elseif ($a == $b)  
{  
    echo "a 等于 b";  
}  
else  
{  
    echo "a 比 b 小";  
}  
?  
?>
```

输出结果为 : [ ]

- A、a 比 b 大
- B、a 等于 b
- C、a 比 b 小
- D、"a 比 b 小"
- E、无输出

7、include 的功能和 require 一样，不同的是，require 不能用在[ ]

- A、判断语句或循环里，
- B、连接语句里
- C、声明语句里
- D、文件的开头
- E、文件的中间

8、PHP 对字符串的处理程序

```
$name="Jollen";  
echo 'Name:$name';  
echo "Name:$name";
```

输出结果为 : [ ]

- A、Name:Jollen  
    Name:Jollen
- B、Name:Jollen  
    Name:\$name
- C、Name:\$name  
    Name:Jollen
- D、Name:\$name  
    Name:\$name
- E、Name:"Jollen"

Name:Jollen

9、数据处理程序

```
$string="This is a test.";
echo ereg_replace(" is"," was",$string)."<br>";
echo ereg_replace("( ) is","\1was",$string)."<br>";
echo ereg_replace("(() is)","\\2was",$string)."<br>";
```

输出为 : [ ]

A、This was a test.

This is a test.

This was a test.

B、This is a test.

This is a test.

This was a test.

C、This is a test.

This is a test.

This is a test.

D、This was a test.

This is a test.

This is a test.

E、This was a test.

This was a test.

This was a test.

10、下面建立与 MySQL Server 的连接语法正确的是 : [ ]

- A、\$link=connect("host\_name","user\_name","password");
- B、\$link=mysql\_connect("host\_name","user\_name","password");
- C、\$link=mysqlconnect("host\_name","user\_name","password");
- D、\$link=mysql\_pconnect("host\_name","user\_name","password");
- E、\$link=pconnect("host\_name","user\_name","password");

11、下面程序

```
<?
$message="abcdefghijklmnopqrstuvwxyz";
mail("php@wilson.gs", "No topic", $message, "From:
someone@wahaha.org.tw\nReply-To: reply@wahaha.org.tw\nX-Mailer: PHP/
.phpversion());
?>[ ]
```

A、从 [php@wilson.gs](mailto:php@wilson.gs) 接收邮件

B、发送邮件到 [reply@wahaha.org.tw](mailto:reply@wahaha.org.tw)

C、发送邮件到 [php@wilson.gs](mailto:php@wilson.gs)

D、从 [reply@wahaha.org.tw](mailto:reply@wahaha.org.tw) 接收邮件

E、不能发送任何邮件

12、rawurlencode()的作用是[ ]

- A、对 PHP3 将要输出的 URL 部分进行编码
- B、对 PHP3 将要输入的 URL 部分进行编码
- C、对 PHP3 已经输出的 URL 部分进行编码
- D、对 PHP3 已经输出的 URL 部分进行编码
- E、对 PHP3 将要输出的 URL 部分进行解码

13、假如我们要删除一个，再建立一个同样的，应写成：[ ]

- A、set("fullname");  
set("fullname","Jacky");
- B、set("fullname","Jacky");  
set("fullname","Jacky");
- C、set("fullname","Jacky");  
set("fullname");
- D、set("fullname");  
sets("fullname","Jacky");
- E、sets("fullname","Jacky");  
sets("fullname");

14、如果要在大量的数据里读取一个字段的数据，则最好使用 mysql\_fetch\_row()、mysql\_fetch\_array()  
mysql\_fetch\_object()函数。因为这几个函数的速度都比[ ]

- A、mysql\_num\_rows
- B、mysql\_num\_fields
- C、mysql\_result
- D、mysql\_list\_fields
- E、mysql\_insert\_id

来得快。

15、在 PHP 中，如果派生类与父类有相同名字的函数，则派生类的函数会替换父类的函数，程序

```
class A
{
    disName()
    {
        echo "Picachu";
    }
}

class B extends A
{
    var tmp;
    disName()
    {
        echo "Doraemon";
    }
}

$cartoon = new B;
$cartoon->disName();
```

结果为 : [ ]

- A、tmp
- B、Picachu
- C、disName
- D、Doraemon
- E、无输出

答案 :

1.[C]

2.[A B C E ]

PHP 的变量属于松散数据类型，在计算时动态(dynamic)决定。如果要强制设置变量的数据类型的话，可以利用 `settype()` 函数。或利用 c 语言的强制转型方式(type casting)。

3.[ A C ]

4.[ D ]

5.[ E ]

6.[ A ]

7.[ E ]

但 `include` 可以。

8.[ C ]

9.[ E ]

10.[ B D ]

11.[ C ]

12.[ C D ]

13.[ B ]

14.[ C ]

15.[ D ]

=====

PHP 面试题系列五【附答案】

一、基础题

1. 写出如下程序的输出结果

```
<?
$str1 = null;
$str2 = false;
echo $str1==$str2 ? '相等' : '不相等';

$str3 = "";
$str4 = 0;
echo $str3==$str4 ? '相等' : '不相等';

$str5 = 0;
$str6 = '0';
echo $str5===$str6 ? '相等' : '不相等';

?>
```

2. 写出如下程序的输出结果

```
<?
$a1 = null;
$a2 = false;
$a3 = 0;
$a4 = "";
$a5 = '0';
$a6 = 'null';
$a7 = array();
$a8 = array(array());

echo empty($a1) ? 'true' : 'false';
echo empty($a2) ? 'true' : 'false';
echo empty($a3) ? 'true' : 'false';
echo empty($a4) ? 'true' : 'false';
echo empty($a5) ? 'true' : 'false';
echo empty($a6) ? 'true' : 'false';
echo empty($a7) ? 'true' : 'false';
echo empty($a8) ? 'true' : 'false';

?>
```

3. 写出如下程序的输出结果

```
<?
$test = 'aaaaaa';
$abc = & $test;
unset($test);

echo $abc;

?>
```

4. 写出如下程序的输出结果

```
<?
$count = 5;
get_count()
{
```

```

static $count = 0;
return $count++;
}

echo $count;
++$count;
echo get_count();
echo get_count();
?>

```

5. 写出如下程序的输出结果

```

<?
$GLOBALS['var1'] = 5;
$var2 = 1;
get_()
{
    global $var2;
    $var1 = 0;
    return $var2++;
}
get_();

echo $var1;
echo $var2;
?>

```

6. 写出如下程序的输出结果

```

<?
get_arr($arr)
{
    unset($arr[0]);
}
$arr1 = array(1, 2);
$arr2 = array(1, 2);

get_arr(&$arr1);
get_arr($arr2);

echo count($arr1);
echo count($arr2);
?>

```

7. 使用五种以上方式获取一个文件的扩展名

要求 : dir/upload.image.jpg , 找出 .jpg 或者 jpg ,

必须使用 PHP 自带的处理函数进行处理 , 方法不能明显重复 , 可以封装成函数 , 比如 get\_ext1(\$file\_name), get\_ext2(\$file\_name)

## 二、算法题

1. 使用 PHP 描述冒泡排序和快速排序算法 , 对象可以是一个数组

2. 使用 PHP 描述顺序查找和二分查找（也叫做折半查找）算法，顺序查找必须考虑效率，对象可以是一个有序数组
3. 写一个二维数组排序算法函数，能够具有通用性，可以调用 php 内置函数

答案（以下答案不一定是最好的，只是一个简单的参考）

### 一、基础题

1. 相等 相等 不相等
2. true true true true true false true false
3. aaaaaa
4. 5 0 1
5. 5 2
6. 1 2
7. 使用五种以上方式获取一个文件的扩展名

```
1)
get_ext1($file_name)
{
    return strrchr($file_name, '.');
}

2)
get_ext2($file_name)
{
    return substr($file_name, strpos($file_name, '.'));
}

3)
get_ext3($file_name)
{
    return array_pop(explode('.', $file_name));
}

4)
get_ext4($file_name)
{
    $p = pathinfo($file_name);
    return $p['extension'];
}

5)
get_ext5($file_name)
{
```

---

```

    return strrev(substr(strrev($file_name), 0, strpos(strrev($file_name), '.')));
}

```

## 二、算法题

- 使用 PHP 描述冒泡排序和快速排序算法，对象可以是一个数组

```

//冒泡排序（数组排序）
bubble_sort($array)
{
    $count = count($array);
    if ($count <= 0)
        return false;

    for($i=0; $i<$count; $i++)
    {
        for($j=$count-1; $j>$i; $j--)
        {
            if ($array[$j] < $array[$j-1])
            {
                $tmp = $array[$j];
                $array[$j] = $array[$j-1];
                $array[$j-1] = $tmp;
            }
        }
    }
    return $array;
}

//快速排序（数组排序）
quick_sort($array)
{
    if (count($array) <= 1)
        return $array;

    $key = $array[0];
    $left_arr = array();
    $right_arr = array();

    for ($i=1; $i<count($array); $i++)
    {
        if ($array[$i] <= $key)
            $left_arr[] = $array[$i];
        else
            $right_arr[] = $array[$i];
    }

    $left_arr = quick_sort($left_arr);
    $right_arr = quick_sort($right_arr);

    return array_merge($left_arr, array($key), $right_arr);
}

```

2. 使用 PHP 描述顺序查找和二分查找（也叫做折半查找）算法，顺序查找必须考虑效率，对象可以是一个有序数组

```
//二分查找（数组里查找某个元素）
bin_sch($array, $low, $high, $k)
{
    if ($low <= $high)
    {
        $mid = intval(($low+$high)/2);
        if ($array[$mid] == $k)
        {
            return $mid;
        }
        elseif ($k < $array[$mid])
        {
            return bin_sch($array, $low, $mid-1, $k);
        }
        else
        {
            return bin_sch($array, $mid+1, $high, $k);
        }
    }
    return -1;
}
```

```
//顺序查找（数组里查找某个元素）
seq_sch($array, $n, $k)
{
    $array[$n] = $k;
    for($i=0; $i<$n; $i++)
    {
        if($array[$i]==$k)
        {
            break;
        }
    }
    if ($i<$n)
    {
        return $i;
    }
    else
    {
        return -1;
    }
}
```

3. 写一个二维数组排序算法函数，能够具有通用性，可以调用 php 内置函数

```
//二维数组排序， $arr 是数据，$keys 是排序的键值，$order 是排序规则，1 是升序，0 是降序
array_sort($arr, $keys, $order=0)
{
```

```
if (!is_array($arr))
{
    return false;
}
$keys = array();
foreach($arr as $key => $val)
{
    $keys[$key] = $val[$keys];
}
if($order == 0)
{
    asort($keys);
}
else
{
    arsort($keys);
}
reset($keys);
foreach($keys as $key => $vals)
{
    $keysort[$key] = $key;
}
$new_array = array();
foreach($keysort as $key => $val)
{
    $new_array[$key] = $arr[$val];
}
return $new_array;
}
```

基础题：

1. 表单中 get 与 post 提交方法的区别？

答：get 是发送请求 HTTP 协议通过 url 参数传递进行接收，而 post 是实体数据，可以通过表单提交大量信息。

2. session 与 cookie 的区别？

答：session：储存用户访问的全局唯一变量，存储在服务器上的 php 指定的目录中的（ session\_dir ）的位置进行的存放

cookie：用来存储连续访问一个页面时所使用，是存储在客户端，对于 Cookie 来说是存储在用户 WIN 的 Temp 目录中的。

两者都可通过时间来设置时间长短

3. 数据库中的事务是什么？

答：事务（ transaction ）是作为一个单元的一组有序的数据库操作。如果组中的所有操作都成功，则认为事务成功，即使只有一个操作失败，事务也不成功。如果所有操作完成，

---

事务则提交，其修改将作用于所有其他数据库进程。如果一个操作失败，则事务将回滚，该事务所有操作的影响都将取消。

简述题：

1、用 PHP 打印出前一天的时间格式是 2006-5-10 22:21:21(2 分)

答:echo date('Y-m-d H:i:s', strtotime('-1 days'));

2、echo(), print(), print\_r() 的区别(3 分)

答:echo 是 PHP 语句, print 和 print\_r 是函数, 语句没有返回值, 函数可以有返回值(即便没有用)

print( ) 只能打印出简单类型变量的值(如 int, string)

print\_r( ) 可以打印出复杂类型变量的值(如数组, 对象)

echo 输出一个或者多个字符串

3、能够使 HTML 和 PHP 分离开使用的模板(1 分)

答:Smarty, Dwoo, TinyButStrong, Template Lite, Savant, phemplate, XTemplate

5、使用哪些工具进行版本控制?(1 分)

答:cvs, svn, vss;

6、如何实现字符串翻转?(3 分)

答:echo strrev(\$a);

7、优化 MYSQL 数据库的方法。(4 分, 多写多得)

答:

1、选取最适用的字段属性, 尽可能减少定义字段长度, 尽量把字段设置 NOT NULL, 例如'省份, 性别', 最好设置为 ENUM

2、使用连接 ( JOIN ) 来代替子查询:

a. 删除没有任何订单客户:DELETE FROM customerinfo WHERE customerid NOT in(SELECT customerid FROM orderinfo)

b. 提取所有没有订单客户:SELECT FROM customerinfo WHERE customerid NOT in(SELECT customerid FROM orderinfo)

c. 提高 b 的速度优化:SELECT FROM customerinfo LEFT JOIN orderid customerinfo.customerid=orderinfo.customerid WHERE orderinfo.customerid IS NULL

3、使用联合(UNION)来代替手动创建的临时表

a. 创建临时表:SELECT name FROM `nametest` UNION SELECT username FROM `nametest2`

#### 4、事务处理:

a. 保证数据完整性, 例如添加和修改同时, 两者成立则都执行, 一者失败都失败

```
mysql_query("BEGIN");
mysql_query("INSERT INTO customerinfo (name) VALUES ('$name1')");
mysql_query("SELECT * FROM `orderinfo` WHERE customerid=\"$id\"");
mysql_query("COMMIT");
```

#### 5、锁定表, 优化事务处理:

a. 我们用一个 SELECT 语句取出初始数据, 通过一些计算, 用 UPDATE 语句将新值更新到表中。

包含有 WRITE 关键字的 LOCK TABLE 语句可以保证在 UNLOCK TABLES 命令被执行之前,  
不会有其它的访问来对 inventory 进行插入、更新或者删除的操作

```
mysql_query("LOCK TABLE customerinfo READ, orderinfo WRITE");
mysql_query("SELECT customerid FROM `customerinfo` WHERE id=\"$id\"");
mysql_query("UPDATE `orderinfo` SET ordertitle='$title' WHERE customerid=\"$id\"");
mysql_query("UNLOCK TABLES");
```

#### 6、使用外键, 优化锁定表

a. 把 customerinfo 里的 customerid 映射到 orderinfo 里的 customerid,  
任何一条没有合法的 customerid 的记录不会写到 orderinfo 里

```
CREATE TABLE customerinfo
(
    customerid INT NOT NULL,
    PRIMARY KEY(customerid)
) TYPE = INNODB;
CREATE TABLE orderinfo
(
    orderid INT NOT NULL,
    customerid INT NOT NULL,
    PRIMARY KEY(customerid, orderid),
    FOREIGN KEY (customerid) REFERENCES customerinfo
        (customerid) ON DELETE CASCADE
) TYPE = INNODB;
```

注意: 'ON DELETE CASCADE', 该参数保证当 customerinfo 表中的一条记录删除的话同时也会删除 order

表中的该用户的所有记录, 注意使用外键要定义事务安全类型为 INNODB;

#### 7、建立索引:

##### a. 格式:

(普通索引) ->

创建:CREATE INDEX <索引名> ON tablename (索引字段)

修改:ALTER TABLE tablename ADD INDEX [索引名] (索引字段)

创表指定索引:CREATE TABLE tablename ([...], INDEX[索引名](索引字段))

(唯一索引) ->

创建:CREATE UNIQUE <索引名> ON tablename (索引字段)

修改:ALTER TABLE tablename ADD UNIQUE [索引名] (索引字段)

创表指定索引:CREATE TABLE tablename ([...], UNIQUE[索引名](索引字段))

(主键) ->

它是唯一索引, 一般在创建表时建立, 格式为:

CREATE TABLE tablename ([...], PRIMARY KEY[索引字段])

#### 8、优化查询语句

##### a. 最好在相同字段进行比较操作, 在建立好的索引字段上尽量减少函数操作

例子 1:

SELECT \* FROM order WHERE YEAR(orderDate)<2008; (慢)

SELECT \* FROM order WHERE orderDate<"2008-01-01"; (快)

例子 2:

SELECT \* FROM order WHERE addtime/7<24; (慢)

SELECT \* FROM order WHERE addtime<24\*7; (快)

例子 3:

SELECT \* FROM order WHERE title like "%good%";

SELECT \* FROM order WHERE title>="good" and name<"good";

#### 8、PHP 的意思(送 1 分)

答: PHP 是一个基于服务端来创建动态网站的脚本语言, 您可以用 PHP 和 HTML 生成网站主页

9、MySQL 取得当前时间的函数是? , 格式化日期的函数是(2 分)

答:now(), date()

10、实现中文字串截取无乱码的方法。(3 分)

```
答:function GBsubstr($string, $start, $length) {  
    if(strlen($string)>$length){  
        $str=null;  
        $len=$start+$length;  
        for($i=$start;$i<$len;$i++){  
            if(ord(substr($string, $i, 1))>0xa0){  
                $str.=substr($string, $i, 2);  
                $i++;  
            }else{  
                $str.=substr($string, $i, 1);  
            }  
        }  
        return $str.'...';  
    }else{  
        return $string;  
    }  
}
```

11、您是否用过版本控制软件? 如果有您用的版本控制软件的名字是?(1 分)

12、您是否用过模板引擎? 如果有您用的模板引擎的名字是?(1 分)

答:用过, smarty

13、请简单阐述您最得意的开发之作(4 分)

答:信息分类

14、对于大流量的网站, 您采用什么样的方法来解决访问量问题?(4 分)

答:确认服务器硬件是否足够支持当前的流量, 数据库读写分离, 优化数据表,

程序功能规则, 禁止外部的盗链, 控制大文件的下载, 使用不同主机分流主要流量

15、用 PHP 写出显示客户端 IP 与服务器 IP 的代码 1 分)

答: 打印客户端 IP: echo \$\_SERVER['REMOTE\_ADDR']; 或者: getenv('REMOTE\_ADDR');

打印服务器 IP: echo gethostbyname("www.bolaiwu.com")

16、语句 include 和 require 的区别是什么? 为避免多次包含同一文件, 可用(?)语句代替它们? (2 分)

答: require->require 是无条件包含也就是如果一个流程里加入 require, 无论条件成立与否都会先执行 require

include->include 有返回值, 而 require 没有(可能因为如此 require 的速度比 include 快)

注意: 包含文件不存在或者语法错误的时候 require 是致命的, include 不是

17、如何修改 SESSION 的生存时间(1 分).

答: 方法 1: 将 php.ini 中的 session.gc\_maxlifetime 设置为 9999 重启 apache

方法 2: \$savePath = "./session\_save\_dir/";

```
$lifeTime = 小时 * 秒;  
session_save_path($savePath);  
session_set_cookie_params($lifeTime);  
session_start();
```

方法 3: setcookie() and session\_set\_cookie\_params(\$lifeTime);

18、有一个网页地址, 比如 PHP 开发资源网主页: <http://www.phpres.com/index.html>, 如何得到它的内容? (\$1 分)

答: 方法 1(对于 PHP5 及更高版本):

```
$readcontents = fopen("http://www.phpres.com/index.html", "rb");  
$contents = stream_get_contents($readcontents);  
fclose($readcontents);  
echo $contents;
```

方法 2:

```
echo file_get_contents("http://www.phpres.com/index.html");
```

19、在 HTTP 1.0 中, 状态码 401 的含义是(?) ; 如果返回“找不到文件”的提示, 则可用 header 函数, 其语句为(?) ; (2 分)

答: 状态 401 代表未被授权, header("Location: www.xxx.php");

12、在 PHP 中，heredoc 是一种特殊的字符串，它的结束标志必须?(1 分)

答:heredoc 的语法是用“<<<”加上自己定义成对的标签，在标签范围内的文字视为一个字符串

例子:

```
$str = <<<SHOW
my name is Jiang Qihui!
SHOW;
```

13、谈谈 asp, php, jsp 的优缺点(1 分)

答:ASP 全名 Active Server Pages , 是一个 WEB 服务器端的开发环境 , 利用它可以产生和运行动态的、交互的、高性能的 WEB 服务应用程序。 ASP 采用脚本语言 VB Script ( Java script ) 作为自己的开发语言。

PHP 是一种跨平台的服务器端的嵌入式脚本语言. 它大量地借用 C, Java 和 Perl 语言的语法 , 并耦合 PHP 自己的特性, 使 WEB 开发者能够快速地写出动态生成页面. 它支持目前绝大多数数据库。还有一点 , PHP 是完全免费的 , 不用花钱 , 你可以从 PHP 官方站点 (<http://www.php.net>) 自由下载。而且你可以不受限制地获得源码 , 甚至可以从中加进你自己需要的特色。

JSP 是 Sun 公司推出的新一代站点开发语言 , 他完全解决了目前 ASP, PHP 的一个通病 - - 脚本级执行 ( 据说 PHP4 也在 Zend 的支持下 , 实现编译运行 ) . Sun 公司借助自己在 Java 上的不凡造诣 , 将 Java 从 Java 应用程序 和 Java Applet 之外 , 又有新的硕果 , 就是 Jsp -- Java Server Page. Jsp 可以在 Serverlet 和 JavaBean 的支持下 , 完成功能强大的站点程序。

三者都提供在 HTML 代码中混合某种程序代码、由语言引擎解释执行程序代码的能力。但 JSP 代码被编译成 Servlet 并由 Java 虚拟机解释执行 , 这种编译操作仅在对 JSP 页面的第一次请求时发生。在 ASP 、 PHP 、 JSP 环境下 , HTML 代码主要负责描述信息的显示样式 , 而程序代码则用来描述处理逻辑。普通的 HTML 页面只依赖于 Web 服务器 , 而 ASP 、 PHP 、 JSP 页面需要附加的语言引擎分析和执行程序代码。程序代码的执行结果被重新嵌入到 HTML 代码中 , 然后一起发送给浏览器。 ASP 、 PHP 、 JSP 三者都是面向 Web 服务器的技术 , 客户端浏览器不需要任何附加的软件支持。

14、谈谈对 mvc 的认识(1 分)

答:由模型(model), 视图(view), 控制器(controller)完成的应用程序

由模型发出要实现的功能到控制器, 控制器接收组织功能传递给视图;

15、写出发帖数最多的十个人名字的 SQL , 利用下表 : members(id, username, posts, pass, email) (2 分)

答:SELECT \* FROM `members` ORDER BY posts DESC limit 0, 10;

16. 请说明 php 中传值与传引用的区别。什么时候传值什么时候传引用?(2 分)

答:按值传递 : 函数范围内对值的任何改变在函数外部都会被忽略

按引用传递 : 函数范围内对值的任何改变在函数外部也能反映出这些修改

优缺点 : 按值传递时 , php 必须复制值。特别是对于大型的字符串和对象来说 , 这将会是一个代价很大的操作。

按引用传递则不需要复制值 , 对于性能提高很有好处。

17. 在 PHP 中 error\_reporting 这个函数有什么作用? (1 分)

答:设置错误级别与错误信息回报

18. 请写一个函数验证电子邮件的格式是否正确 (2 分)

答:function checkEmail(\$email)

```
{  
    $pregEmail = "/([a-z0-9]*[-_.]?[a-z0-9]+)*@[a-z0-9]*[-_.]?[a-z0-9]+)[/.][a-z]{2,3}([/.][a-z]{2})?/i";  
    return preg_match($pregEmail, $email);  
}
```

19. 简述如何得到当前执行脚本路径 , 包括所得到参数。 (2 分)

答:\$script\_name = basename(\_\_file\_\_); print\_r(\$script\_name);

21、JS 表单弹出对话框函数是?获得输入焦点函数是? (2 分)

答:弹出对话框: alert(), prompt(), confirm()

获得输入焦点 focus()

22、JS 的转向函数是?怎么引入一个外部 JS 文件?(2 分)

答:window.location.href,<script type="text/javascript" src="js/js\_function.js"></script>

23、foo() 和 @foo() 之间有什么区别?(1 分)

答:@foo() 控制错误输出

24、如何声明一个名为”myclass”的没有方法和属性的类?(1 分)

答:class myclass{ }

25、如何实例化一个名为”myclass”的对象?(1 分)

答:new myclass()

26、你如何访问和设置一个类的属性?(2 分)

答:\$object = new myclass();  
\$newstr = \$object->test;  
\$object->test = "info";

27、mysql\_fetch\_row() 和 mysql\_fetch\_array 之间有什么区别?(1 分)

答:mysql\_fetch\_row 是从结果集取出 1 行数组, 作为枚举

mysql\_fetch\_array 是从结果集取出一行数组作为关联数组, 或数字数组, 两者兼得

28、GD 库是做什么用的?(1 分)

答:gd 库提供了一系列用来处理图片的 API, 使用 GD 库可以处理图片, 或者生成图片。

在网站上 GD 库通常用来生成缩略图或者用来对图片加水印或者对网站数据生成报表。

29、指出一些在 PHP 输入一段 HTML 代码的办法。(1 分)

答:echo "<a href='index.php'>aaa</a>";

30、下面哪个函数可以打开一个文件, 以对文件进行读和写操作?(1 分)

- (a) fget() (b) file\_open() (c) fopen() (d) open\_file() [ c ]

31、下面哪个选项没有将 john 添加到 users 数组中? (1 分)

- (a) \$users[] = 'john' ;
- (b) array\_add(\$users, 'john') ;
- (c) array\_push(\$users, 'john') ;
- (d) \$users |= 'john' ; [ a , c ]

32、下面的程序会输入是否?(1 分)

```
$num = 10;  
function multiply() {  
    $num = $num * 10;  
}  
multiply();  
echo $num;  
?>
```

输出:10

33、使用 php 写一段简单查询，查出所有姓名为“张三”的内容并打印出来 (2 分)

表名 User

Name	Tel	Content	Date
张三	13333663366	大专毕业	2006-10-11
张三	13612312331	本科毕业	2006-10-15
张四	021-55665566	中专毕业	2006-10-15

请根据上面的题目完成代码：

```
$mysql_db=mysql_connect("local","root","pass");  
@mysql_select_db("DB",$mysql_db);  
  
$result = mysql_query("SELECT * FROM `user` WHERE name='张三'");  
while($rs = mysql_fetch_array($result)){  
    echo $rs["tel"].$rs["content"].$rs["date"];  
}
```

34、如何使用下面的类，并解释下面什么意思?(3)

```
class test{  
    function Get_test($num){
```

```

    $num=md5(md5($num). "En");
    return $num;
}
}

答:$testnum = "123";
$object = new test();
$encrypt = $object->Get_test($testnum);
echo $encrypt;
类 test 里面包含 Get_test 方法, 实例化类调用方法多字符串加密

```

35、写出 SQL 语句的格式：插入，更新，删除（4 分）

表名 User

Name Tel Content Date

张三 13333663366 大专毕业 2006-10-11

张三 13612312331 本科毕业 2006-10-15

张四 021-55665566 中专毕业 2006-10-15

(a) 有一新记录(小王 13254748547 高中毕业 2007-05-06)请用 SQL 语句新增至表中

```
mysql_query("INSERT INTO `user` (name, tel, content, date) VALUES
('小王', '13254748547', '高中毕业', '2007-05-06')")
```

(b) 请用 sql 语句把张三的时间更新成为当前系统时间

```
$nowDate = date("Y-m-d");
mysql_query("UPDATE `user` SET date='". $nowDate ."' WHERE name='张山'");
```

(c) 请写出删除名为张四的全部记录

```
mysql_query("DELETE FROM `user` WHERE name='张四'");
```

36、请写出数据类型(int char varchar datetime text)的意思；请问 varchar 和 char 有什么区别(2 分)

答:int 是数字类型, char 固定长度字符串, varchar 实际长度字符串, datetime 日期时间型, text 文本字符串

char 的场地固定为创建表设置的长度, varchar 为可变长度的字符

38、写出以下程序的输出结果 (1 分)

```
$b=201;  
$c=40;  
$a=$b>$c?4:5;  
echo $a;  
?>
```

答:4

39、检测一个变量是否有设置的函数是否?是否为空的函数是?(2 分)

答:isset(\$str), empty(\$str);

40、取得查询结果集总数的函数是?(1 分)

答:mysql\_num\_rows(\$result);

41、\$arr = array('james', 'tom', 'symfony'); 请打印出第一个元素的值 (1 分)

答:echo \$array[0];

42、请将 41 题的数组的值用'，'号分隔并合并成字符串输出(1 分)

答:for(\$i=0;\$i<count(\$array);\$i++) { echo \$array[\$i].",";}

43、\$a = 'abcdef'; 请取出\$a 的值并打印出第一个字母(1 分)

答:echo \$a{0} 或 echo substr(\$a, 0, 1)

44、PHP 可以和 sql server/oracle 等数据库连接吗?(1 分)

答:当然可以

45、请写出 PHP5 权限控制修饰符(3 分)

答:public(公共), private(私用), protected(继承)

46、请写出 php5 的构造函数和析构函数(2 分)

答:\_\_construct , \_\_destruct

47、完成以下：

(一) 创建新闻发布系统，表名为 message 有如下字段 (3 分)

id 文章 id

title 文章标题

content 文章内容

category\_id 文章分类 id

hits 点击量

答:CREATE TABLE 'message' (

```
'id' int(10) NOT NULL auto_increment,  
'title' varchar(200) default NULL,  
'content' text,  
'category_id' int(10) NOT NULL,  
'hits' int(20),  
PRIMARY KEY('id')  
)ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

(二) 同样上述新闻发布系统：表 comment 记录用户回复内容，字段如下 (4 分)

comment\_id 回复 id

id 文章 id，关联 message 表中的 id

comment\_content 回复内容

现通过查询数据库需要得到以下格式的文章标题列表，并按照回复数量排序，回复最高的排在最前面

文章 id 文章标题 点击量 回复数量

用一个 SQL 语句完成上述查询，如果文章没有回复则回复数量显示为 0

答:SELECT message.id id, message.title title, IF(message.`hits` IS NULL, 0, message.`hits`) hits,  
IF(comment.`id` IS NULL, 0, count(\*)) number FROM message LEFT JOIN  
comment ON message.id=comment.id GROUP BY message.`id`;

(三) 上述内容管理系统，表 category 保存分类信息，字段如下 (3 分)

category\_id int(4) not null auto\_increment;

categroy\_name varchar(40) not null;

用户输入文章时，通过选择下拉菜单选定文章分类

写出如何实现这个下拉菜单

答: function categoryList()

```
{  
    $result=mysql_query("select category_id,categroy_name from category"  
        or die("Invalid query: " . mysql_error());  
    print("<select name='category' value='>/n");  
    while($rowArray=mysql_fetch_array($result))  
    {  
        print("<option value='". $rowArray['category_id']."' >". $rowArray['categroy_name'] . "</option>/n");  
    }  
    print("</select>");  
}
```

编程题:

1. 写一个函数，尽可能高效的，从一个标准 url 里取出文件的扩展名

例如: <http://www.sina.com.cn/abc/de/fg.php?id=1> 需要取出 php 或 .php

答案 1:

```
function getExt($url) {  
    $arr = parse_url($url);  
  
    $file = basename($arr['path']);  
    $ext = explode(".", $file);  
    return $ext[1];  
}
```

答案 2:

```
function getExt($url) {  
    $url = basename($url);  
    $pos1 = strpos($url, ".");  
    $pos2 = strpos($url, "?");  
    if(strstr($url, "?")) {  
        return substr($url, $pos1 + 1, $pos2 - $pos1 - 1);  
    } else {  
        return substr($url, $pos1);  
    }  
}
```

```
    }  
}
```

2. 在 HTML 语言中，页面头部的 meta 标记可以用来输出文件的编码格式，以下是一个标准的 meta 语句

请使用 PHP 语言写一个函数，把一个标准 HTML 页面中的类似 meta 标记中的 charset 部分值改为 big5

请注意：

1. 需要处理完整的 html 页面，即不光此 meta 语句
  2. 忽略大小写
  3. ' 和 " 在此处是可以互换的
  4. 'Content-Type' 两侧的引号是可以忽略的，但 'text/html; charset=gbk' 两侧的不行
  5. 注意处理多余空格
3. 写一个函数，算出两个文件的相对路径

如 \$a = '/a/b/c/d/e.php' ;

\$b = '/a/b/12/34/c.php' ;

计算出 \$b 相对于 \$a 的相对路径应该是 ../../c 将()添上

答:function getRelativePath(\$a, \$b) {

```
    $returnPath = array(dirname($b));  
    $arrA = explode(' /', $a);  
    $arrB = explode(' /', $returnPath[0]);  
    for ($n = 1, $len = count($arrB); $n < $len; $n++) {  
        if ($arrA[$n] != $arrB[$n]) {  
            break;  
        }  
    }  
    if ($len - $n > 0) {  
        $returnPath = array_merge($returnPath, array_fill(1, $len - $n, '..'));  
    }  
  
    $returnPath = array_merge($returnPath, array_slice($arrA, $n));  
    return implode(' /', $returnPath);
```

```
}
```

```
echo getRelativePath($a, $b);
```

填空题：

1. 在 PHP 中，当前脚本的名称(不包括路径和查询字符串)记录在预定义变量 `$_SERVER['PHP_SELF']` 中；而链接到当前页面的 URL 记录在预定义变量 `$_SERVER['HTTP_REFERER']` 中

中

2. 执行程序段 `<?php echo 8%(-2) ?>` 将输出 `0`。

3. 在 HTTP 1.0 中，状态码 401 的含义是\_\_\_\_；如果返回“找不到文件”的提示，则可用 `header` 函数，其语句为\_\_\_\_。

4. 数组函数 `arsort` 的作用是\_\_\_\_对数组进行逆向排序并保持索引关系\_\_\_\_；语句 `error_reporting(2047)` 的作用是\_\_\_\_报告所有错误和警告\_\_\_\_。

5. PEAR 中的数据库连接字符串格式是\_\_\_\_。

6. 写出一个正则表达式，过滤网页上的所有 JS/VBS 脚本(即把 `script` 标记及其内容都去掉)：`preg_replace('/<script[^>].*?>.*/</script>/si', "newinfo", $script);`

7. 以 Apache 模块的方式安装 PHP，在文件 `http.conf` 中首先要用语句\_\_\_\_动态装载 PHP 模块，然后再用语句\_\_\_\_使得 Apache 把所有扩展名为 `php` 的文件都作为 PHP 脚本处理。

```
LoadModule php5_module "c:/php/php5apache2.dll" , AddType application/x-httpd-php .php,
```

8. 语句 `include` 和 `require` 都能把另外一个文件包含到当前文件中，它们的区别是\_\_\_\_；为了避免多次包含同一文件，可以用语句\_\_\_\_`require_once` | `include_once`\_\_\_\_来代替它们。

9. 类的属性可以序列化后保存到 `session` 中，从而以后可以恢复整个类，这要用到的函数是\_\_\_\_。

10. 一个函数的参数不能是对变量的引用，除非在 `php.ini` 中把 `allow_call_time_pass_reference boolean` 设为 `on`。

11. SQL 中 LEFT JOIN 的含义是自然左外链接。如果 `tbl_user` 记录了学生的姓名 (name) 和学号 (ID) , `tbl_score` 记录了学生 (有的学生考试以后被开除了，没有其记录) 的学号 (ID)

和考试成绩 (score) 以及考试科目 (subject) , 要想打印出各个学生姓名及对应的的各科总成绩，则可以用 SQL 语句\_\_\_\_\_。

12. 在 PHP 中 , heredoc 是一种特殊的字符串 , 它的结束标志必须\_\_\_\_\_。

编程题:

13. 写一个函数 , 能够遍历一个文件夹下的所有文件和子文件夹。

答:

```
function my_scandir($dir)
{
    $files = array();
    if ( $handle = opendir($dir) ) {
        while ( ($file = readdir($handle)) !== false ) {
            if ( $file != ".." && $file != "." ) {
                if ( is_dir($dir . "/" . $file) ) {
                    $files[$file] = scandir($dir . "/" . $file);
                } else {
                    $files[] = $file;
                }
            }
        }
        closedir($handle);
        return $files;
    }
}
```

14. 简述论坛中无限分类的实现原理。

答:

```
<?php
/*
```

数据表结构如下:

```
CREATE TABLE `category` (
```

```

`categoryID` smallint(5) unsigned NOT NULL auto_increment,
`categoryParentID` smallint(5) unsigned NOT NULL default '0',
`categoryName` varchar(50) NOT NULL default '',
PRIMARY KEY (`categoryID`)
) ENGINE=MyISAM DEFAULT CHARSET=gbk;

INSERT INTO `category` ( `categoryParentID` , `categoryName` ) VALUES
(0, '一级类别'),
(1, '二级类别'),
(1, '二级类别'),
(1, '二级类别'),
(2, '三级类别'),
(2, '333332'),
(2, '234234'),
(3, 'aqqqqqd'),
(4, '哈哈'),
(5, '66333666');

*/
//指定分类 id 变量$category_id, 然后返回该分类的所有子类
//{$default_category 为默认的选中的分类
function Get_Category($category_id = 0, $level = 0, $default_category = 0)
{
    global $DB;
    $sql = "SELECT * FROM category ORDER BY categoryID DESC";
    $result = $DB->query( $sql );
    while ($rows = $DB->fetch_array($result))
    {
        $category_array[$rows[categoryParentID]][$rows[categoryID]] = array('id' => $rows[categoryID], 'parent' =>
$rows[categoryParentID], 'name' => $rows
[categoryName]);
    }
    if (!isset($category_array[$category_id]))

```

```

{
    return "";
}

foreach($category_array[$category_id] AS $key => $category)
{
    if ($category['id'] == $default_category)
    {
        echo "<option selected value=". $category['id']. ">";
    } else
    {
        echo "<option value=". $category['id']. ">";
    }

    if ($level > 0)
    {
        echo ">" . str_repeat( " ", $level ) . " " . $category['name'] . "</option>/n";
    }
    else
    {
        echo ">" . $category['name'] . "</option>/n";
    }
    Get_Category($key, $level + 1, $default_category);
}
unset($category_array[$category_id]);
}

/*

```

函数返回的数组格式如下所示：

```

Array
(
    [1] => Array ( [id] => 1 [name] => 一级类别 [level] => 0 [ParentID] => 0 )
    [4] => Array ( [id] => 4 [name] => 二级类别 [level] => 1 [ParentID] => 1 )
    [9] => Array ( [id] => 9 [name] => 哈哈 [level] => 2 [ParentID] => 4 )
    [3] => Array ( [id] => 3 [name] => 二级类别 [level] => 1 [ParentID] => 1 )
    [8] => Array ( [id] => 8 [name] => aqqqqqd [level] => 2 [ParentID] => 3 )

```

```

[2] => Array ( [id] => 2 [name] => 二级类别 [level] => 1 [ParentID] => 1 )
[7] => Array ( [id] => 7 [name] => 234234 [level] => 2 [ParentID] => 2 )
[6] => Array ( [id] => 6 [name] => 333332 [level] => 2 [ParentID] => 2 )
[5] => Array ( [id] => 5 [name] => 三级类别 [level] => 2 [ParentID] => 2 )
[10] => Array ( [id] => 10 [name] => 66333666 [level] => 3 [ParentID] => 5 )
)

/*
//指定分类 id, 然后返回数组
function Category_array($category_id = 0,$level=0)
{
    global $DB;
    $sql = "SELECT * FROM category ORDER BY categoryID DESC";
    $result = $DB->query($sql);
    while ($rows = $DB->fetch_array($result))
    {
        $category_array[$rows['categoryParentID']][$rows['categoryID']] = $rows;
    }

    foreach ($category_array AS $key=>$val)
    {
        if ($key == $category_id)
        {
            foreach ($val AS $k=> $v)
            {
                $options[$k] =
                    array(
                        'id' => $v['categoryID'], 'name' => $v['categoryName'], 'level' => $level, 'ParentID'=>$v['categoryParentID']
                    );
            }

            $children = Category_array($k, $level+1);

            if (count($children) > 0)
            {
                $options = $options + $children;
            }
        }
    }
}

```

```
}

}

unset($category_array[$category_id]);
return $options;
}

?>

<?php

class cate
{

    function Get_Category($category_id = 0, $level = 0, $default_category = 0)
    {
        echo $category_id;
        $arr = array(
            '0' => array(
                '1' => array('id' => 1, 'parent' => 0, 'name' => '1111'),
                '2' => array('id' => 2, 'parent' => 0, 'name' => '2222'),
                '4' => array('id' => 4, 'parent' => 0, 'name' =>
                '4444')
            ),
            '1' => array(
                '3' => array('id' => 3, 'parent' => 1, 'name' =>
                '333333'),
                '5' => array('id' => 5, 'parent' => 1, 'name' =>
                '555555')
            ),
            '3' => array(
                '6' => array('id' => 6, 'parent' => 3, 'name' => '66666'),
                '7' => array('id' => 7, 'parent' => 3, 'name' => '77777')
            ),
            '4' => array(
```

```

        '8' => array('id' => 8, 'parent' => 4, 'name' => '8888'),
        '9' => array('id' => 9, 'parent' => 4, 'name' => '9999')
    )
);

if (!isset($arr[$category_id]))
{
    return "";
}

foreach($arr[$category_id] AS $key => $cate)
{
    if ($cate['id'] == $default_category)
    {
        $txt = "<option selected value=". $cate['id']. ">";
    } else{
        $txt = "<option value=". $cate['id']. ">";
    }

    if ($level > 0)
    {
        $txt1 = ">" . str_repeat( "-", $level ) . " " . $cate['name'] . "</option>/n";
    } else{
        $txt1 = ">" . $cate['name'] . "</option>/n";
    }
    $val = $txt.$txt1;
    echo $val;
    self::Get_Category($key, $level + 1, $default_category);
}

function getFlush($category_id = 0, $level = 0, $default_category = 0)
{

```

```
ob_start();

self::Get_Category($category_id , $level, $default_category);

$out = ob_get_contents();

ob_end_clean();
return $out;
}

}

$id = $_GET['id'];
echo "<select>";
$c = new cate();
// $c->Get_Category();
$ttt= $c->getFlush($id, '0', '3');
echo $ttt;
echo "</select>";
?>
```

**1、抓取远程图片到本地，你会用什么函数？**

fsockopen

**2、用最少的代码写一个求 3 值最大值的函数。**

```
function($a, $b, $c) {
    return $a > $b ? ($a > $c ? $a : $c) : ($b > $c ? $b : $c);
}
```

**3、用 PHP 打印出前一天的时间，打印格式是 2007 年 5 月 10 日 22:21:21**

```
Echo date( 'Y-m-d H:i:s' , strtotime( '-1 day' ));
```

**4、javascript 能否定义二维数组，如果不能你如何解决？**

javascript 不支持二维数组定义，可以用 arr[0] = new array() 来解决

**5、假设 a.html 和 b.html 在同一个文件夹下面，用 javascript 实现当打开 a.html 五秒钟后，自动跳转到 b.html。**

```
<script>
function go2b() {
```

```
window.location = "b.html";
window.close();
}

setTimeout("go2b()", 5000); //5秒钟后自动执行 go2b()
</script>
```

**6、正在浏览当前页面用户的 IP 地址:127.0.0.1**

```
echo $_SERVER["REMOTE_ADDR"]."<br />";
//查询(query)的字符串(URL中第一个问号?之后的内容):id=1&bi=2
echo $_SERVER["QUERY_STRING"]."<br />";
//当前运行脚本所在的文档根目录:d:\inetpub\wwwroot
echo $_SERVER["DOCUMENT_ROOT"]."<br />";
```

**7、在 HTTP 1.0 中，状态码 401 的含义是\_\_；如果返回“找不到文件”的提示，则可用 header 函数，其语句为**

```
header("HTTP/1.0 404 Not Found");
```

答：401 表示未授权;header("HTTP/1.0 404 Not Found");

**8、写一个函数，能够遍历一个文件夹下的所有文件和子文件夹。**

```
<?php

function my_scandir($dir)
{
    $files=array();
    if(is_dir($dir))
    {
        if($handle=opendir($dir))
        {
            while(($file=readdir($handle))!==false)
            {
                if($file!="." && $file!="..")
                {
                    if(is_dir($dir."/". $file))
                    {
```

```

        $files[$file]=my_scandir($dir."/".$file);

    }

else

{

    $files[]=$dir."/".$file;

}

}

closedir($handle);

return $files;

}

}

print_r(my_scandir("D:\Program Files\Internet Explorer\MUI"));

?>

```

### 9、把 John 新增到 users 阵列？

```
$users[] = 'john'; array_push($users, 'john');
```

### 10、在 PHP 中 error\_reporting 这个函数有什么作用？

答：error\_reporting() 设置 PHP 的报错级别并返回当前级别。

### 11、请用正则表达式 ( Regular Expression ) 写一个函数验证电子邮件的格式是否正确。

答：

```
<?php

if(isset($_POST['action']) &&
$_POST['action']=='submitted')
{
    $email=$_POST['email'];
    if(!preg_match("/^(\?:\w+.\?)*\w+@\(\?:\w+.\?)*\w+\$/", $email))
        ("/^\\w+@{1}\\w+\\.{1}\\w+$/", $a)
    {
        echo
    "电子邮件检测失败";
    }
}
```

```
        else
        {
            echo
        "电子邮件检测成功";
    } }

else
{
?>
<html>
<head><title>EMAIL 检测</title>
<script type="text/javascript">
function checkEmail(sText)
{
    var reg=/^ (:w+.) *w+@(:w+.) *w+$/;
    var email=document.getElementById(sText).value;
    if(!reg.test(email))
    {
        alert("电子邮件检测失败");
    }
    else
    {
        alert("电子邮件格式正确");
    }
}
</script>
</head>
<body>
<form action=<?php echo $_SERVER['PHP_SELF'] ?>" method="POST">
电子邮件:<input type="text" id="email" name="email"
/><br />
<input type="hidden" name="action" value="submitted"
/>
<input type="button" name="button" value="客户端检测" onclick="checkEmail('email')"
/>
<input type="submit" name="submit" value="服务器端检测"
/>
</form>
</body>
</html>
```

```
<?php  
}  
?>
```

**12、用 PHP 写出显示客户端 IP 与服务器 IP 的代码**

答: 打印客户端 IP: echo \$\_SERVER['REMOTE\_ADDR']; 或者: getenv('REMOTE\_ADDR');  
打印服务器 IP: echo gethostbyname("www.bolaiwu.com")

**13、如何修改 SESSION 的生存时间(1 分).**

答: 方法 1: 将 php.ini 中的 session.gc\_maxlifetime 设置为 9999 重启 apache

方法 2: \$savePath = "./session\_save\_dir/";  
\$lifeTime = 小时 \* 秒;  
session\_save\_path(\$savePath);  
session\_set\_cookie\_params(\$lifeTime);  
session\_start();

方法 3: setcookie() and session\_set\_cookie\_params(\$lifeTime);

**14、有一个网页地址，比如 PHP 开发资源网主页: http://www.phpres.com/index.html, 如何得到它的内容? (\$1 分)**

答: 方法 1 (对于 PHP5 及更高版本):

```
$readcontents = fopen("http://www.phpres.com/index.html", "rb");  
$contents = stream_get_contents($readcontents);  
fclose($readcontents);  
echo $contents;
```

方法 2:

```
echo file_get_contents("http://www.phpres.com/index.html");
```

**15、请说明 php 中传值与传引用的区别。什么时候传值什么时候传引用? (2 分)**

答: 按值传递: 函数范围内对值的任何改变在函数外部都会被忽略

按引用传递: 函数范围内对值的任何改变在函数外部也能反映出这些修改

优缺点: 按值传递时, php 必须复制值。特别是对于大型的字符串和对象来说, 这将会是一个代价很大的操作。

按引用传递则不需要复制值, 对于性能提高很有好处。

**16、写一个函数, 尽可能高效的, 从一个标准 url 里取出文件的扩展名**

例如: http://www.sina.com.cn/abc/de/fg.php?id=1 需要取出 .php 或 .php

答案 1:

---

```

function getExt($url) {
    $arr = parse_url($url);

    $file = basename($arr['path']);
    $ext = explode(".", $file);
    return $ext[1];
}

```

答案 2:

```

function getExt($url) {
    $url = basename($url);
    $pos1 = strpos($url, ".");
    $pos2 = strpos($url, "?");
    if(strstr($url, "?")) {
        return substr($url, $pos1 + 1, $pos2 - $pos1 - 1);
    } else {
        return substr($url, $pos1);
    }
}

```

## 17. 使用五种以上方式获取一个文件的扩展名

要求：dir/upload.image.jpg，找出 .jpg 或者 jpg ，

必须使用 PHP 自带的处理函数进行处理，方法不能明显重复，可以封装成函数，比如 get\_ext1(\$file\_name), get\_ext2(\$file\_name)

```

function get_ext1($file_name) {
    return strrchr($file_name, '.');
}

function get_ext2($file_name) {
    return substr($file_name, strrpos($file_name, '.'));
}

function get_ext3($file_name) {
    return array_pop(explode('.', $file_name));
}

function get_ext4($file_name) {
    $p = pathinfo($file_name);
    return $p['extension'];
}

function get_ext5($file_name) {
    return strrev(substr(strrev($file_name), 0, strpos(strrev($file_name), '.')));
}

```

18、<?php  
\$str1 = null;  
\$str2 = false;  
echo \$str1==\$str2 ? '相等' : '不相等';  
\$str3 = '';  
\$str4 = 0;  
echo \$str3==\$str4 ? '相等' : '不相等';  
\$str5 = 0;  
\$str6 = '0';  
echo \$str5==\$str6 ? '相等' : '不相等';  
?>  
相等 相等 不相等

19、MySQL 数据库中的字段类型 varchar 和 char 的主要区别是什么？那种字段的查找效率要高，为什么？

Varchar 是变长，节省存储空间，char 是固定长度。查找效率要 char 型快，因为 varchar 是非定长，必须先查找长度，然后进行数据的提取，比 char 定长类型多了一个步骤，所以效率低一些

20、请使用 JavaScript 写出三种产生一个 Image 标签的方法（提示：从方法、对象、HTML 角度考虑）

(1) var img = new Image();  
(2) var img = document.createElement("image")  
(3) img.innerHTML = "<img src='xxx.jpg' />"

21、16. 请描述出两点以上 XHTML 和 HTML 最显著的区别

(1) XHTML 必须强制指定文档类型 DocType，HTML 不需要  
(2) XHTML 所有标签必须闭合，HTML 比较随意

22、写一个排序算法，可以是冒泡排序或者是快速排序，假设待排序对象是一个维数组。

```
//冒泡排序(数组排序)  
function bubble_sort($array)  
{  
    $count = count($array);  
    if ($count <= 0) return false;  
    for($i=0; $i<$count; $i++) {  
        for($j=$count-1; $j>$i; $j--) {
```

```

if  ($array[$j]  < $array[$j-1]) {
$tmp = $array[$j];
$array[$j] = $array[$j-1];
$array[$j-1] = $tmp;
}
}
}

return $array;
}

//快速排序(数组排序)

function quicksort($array) {
if (count($array) <= 1) return $array;
$key = $array[0];
$left_arr = array();
$right_arr = array();
for ($i=1; $i<count($array); $i++) {
if ($array[$i] <= $key)
$left_arr[] = $array[$i];
else
$right_arr[] = $array[$i];
}
$left_arr = quicksort($left_arr);
$right_arr = quicksort($right_arr);
return array_merge($left_arr, array($key), $right_arr);
}

```

### 23、写出三种以上 MySQL 数据库存储引擎的名称 ( 提示 : 不区分大小写 )

MyISAM、InnoDB、BDB ( Berkeley DB ) 、Merge、Memory ( Heap ) 、Example、Federated、Archive、CSV、Blackhole、MaxDB 等等十几个引擎

### 24、求两个日期的差数，例如 2007-2-5 ~ 2007-3-6 的日期差数

方法一：

```

<?php
class Dtime
{
function get_days($date1, $date2)
{
$time1 = strtotime($date1);
$time2 = strtotime($date2);
return ($time2-$time1)/86400;
}

```

```

}
}

$Dtime = new Dtime;
echo $Dtime->get_days('2007-2-5', '2007-3-6');
?>

```

**方法二：**

```

<?php
$temp = explode('-', '2007-2-5');
$time1 = mktime(0, 0, 0, $temp[1], $temp[2], $temp[0]);
$temp = explode('-', '2007-3-6');
$time2 = mktime(0, 0, 0, $temp[1], $temp[2], $temp[0]);
echo ($time2-$time1)/86400;

```

**方法三：**echo abs(strtotime("2007-2-1")-strtotime("2007-3-1"))/60/60/24 计算时间差

**25、请写一个函数，实现以下功能：**

字符串 “open\_door” 转换成 “OpenDoor” 、“ make\_by\_id” 转换成 “ MakeById” 。

**方法一：**

```

function str_explode($str) {
$str_arr=explode("_",$str);$str_implode=implode(" ",$str_arr); $str_implode=implode
("",explode(" ",ucwords($str_implode)));
return $str_implode;
}
$strexplode=str_explode("make_by_id");print_r($strexplode);

```

**方法二：**\$str="make\_by\_id!";

```

$expStr=explode("_",$str);
for($i=0;$i<count($expStr);$i++)
{
echo ucwords($expStr[$i]);
}

```

**方法三：**

```

echo str_replace(' ',' ',ucwords(str_replace('_', ' ', 'open_door')));

```

**26、一个表中的 Id 有多个记录，把所有这个 id 的记录查出来，并显示共有多少条记录数，用 SQL 语句及视图、存储过程分别实现。**

```

DELIMITER //
create procedure proc_countNum(in columnId int,out rowsNo int)
begin
select count(*) into rowsNo from member where member_id=columnId;
end
call proc_countNum(1,@no);
select @no;

```

方法：视图：

```
create view v_countNum as select member_id, count(*) as countNum from member group by member_id
select countNum from v_countNum where member_id=1
```

27、js 中网页前进和后退的代码 （ 前进： history.forward();=history.go(1); 后退： history.back()  
() ;=history.go(-1); ）

28、echo count("abc"); 输出什么？

答案:1

29、有一个一维数组，里面存储整形数据，请写一个函数，将他们按从大到小的顺序排列。要求执行效率高。并说明如何改善执行效率。（该函数必须自己实现，不能使用 php 函数）

```
<?php
function BubbleSort(&$arr)
{
    $cnt=count($arr);
    $flag=1;
    for($i=0;$i<$cnt;$i++)
    {
        if($flag==0)
        {
            return;
        }
        $flag=0;
        for($j=0;$j<$cnt-$i-1;$j++)
        {
            if($arr[$j]>$arr[$j+1])
            {
                $tmp=$arr[$j];
                $arr[$j]=$arr[$j+1];
                $arr[$j+1]=$tmp;
                $flag=1;
            }
        }
    }
}
```

```
}
```

```
$test=array(1, 3, 6, 8, 2, 7);
```

```
BubbleSort($test);
```

```
var_dump($test);
```

```
?>
```

### 30、请举例说明在你的开发过程中用什么方法来加快页面的加载速度

答：要用到服务器资源时才打开，及时关闭服务器资源，数据库添加索引，页面可生成静态，图片等大文件单独服务器。使用代码优化工具

### 31、以下的代码会产生什么？为什么？

```
$num =10;
```

```
function multiply() {
```

```
$num = $num *10;
```

```
}
```

```
multiply();
```

```
echo $num;
```

由于函式 multiply() 没有指定 \$num 为全域变量（例如 global \$num 或者 \$\_GLOBALS['num']），所以 \$num 的值是 10。

### . 使用 PHP 描述冒泡排序和快速排序算法，对象可以是一个数组

```
//冒泡排序（数组排序）
```

```
function bubble_sort($array)
```

```
{
```

```
    $count = count($array);
```

```
    if ($count <= 0) return false;
```

```
    for($i=0; $i<$count; $i++) {
```

```
        for($j=$count-1; $j>$i; $j--) {
```

```
            if ($array[$j] < $array[$j-1]) {
```

```
                $tmp = $array[$j];
```

```
                $array[$j] = $array[$j-1];
```

```
                $array[$j-1] = $tmp;
```

```
            }
```

```
        }
    }

    return $array;
}

//快速排序(数组排序)
function quick_sort($array) {
    if (count($array) <= 1) return $array;

    $key = $array[0];
    $left_arr = array();
    $right_arr = array();

    for ($i=1; $i<count($array); $i++) {
        if ($array[$i] <= $key)
            $left_arr[] = $array[$i];
        else
            $right_arr[] = $array[$i];
    }

    $left_arr = quick_sort($left_arr);
    $right_arr = quick_sort($right_arr);

    return array_merge($left_arr, array($key), $right_arr);
}
```

万网面试 PHP 笔试题(部分),转自网络:

基础题:

1. 表单中 get 与 post 提交方法的区别?

答: get 是发送请求 HTTP 协议通过 url 参数传递进行接收, 而 post 是实体数据, 可以通过表单提交大量信息.

2. session 与 cookie 的区别?

答: session: 储存用户访问的全局唯一变量, 存储在服务器上的 php 指定的目录中的 ( session\_dir ) 的位置进行的存放

cookie:用来存储连续访问一个页面时所使用，是存储在客户端，对于 Cookie 来说是存储在用户 WIN 的 Temp 目录中的。

两者都可通过时间来设置时间长短

3. 数据库中的事务是什么？

答:事务 (transaction) 是作为一个单元的一组有序的数据库操作。如果组中的所有操作都成功，则认为事务成功，即使只有一个操作失败，事务也不成功。如果所有操作完成，

事务则提交，其修改将作用于所有其他数据库进程。如果一个操作失败，则事务将回滚，该事务所有操作的影响都将取消。

简述题：

1、用 PHP 打印出前一天的时间格式是 2006-5-10 22:21:21 (2 分)

答:`echo date('Y-m-d H:i:s', strtotime('-1 days'));`

2、`echo()`,`print()`,`print_r()`的区别 (3 分)

答:`echo` 是 PHP 语句, `print` 和 `print_r` 是函数, 语句没有返回值, 函数可以有返回值(即便没有用)

`print()` 只能打印出简单类型变量的值(如 `int`, `string`)

`print_r()` 可以打印出复杂类型变量的值(如数组, 对象)

`echo` 输出一个或者多个字符串

3、能够使 HTML 和 PHP 分离开使用的模板 (1 分)

答:`Smarty`, `Dwoo`, `TinyButStrong`, `Template Lite`, `Savant`, `phemplate`, `XTemplate`

5、使用哪些工具进行版本控制? (1 分)

答:`cvs`, `svn`, `vss`;

6、如何实现字符串翻转? (3 分)

答:`echo strrev($a);`

7、优化 MySQL 数据库的方法。 (4 分, 多写多得)

答:

1、选取最适用的字段属性,尽可能减少定义字段长度,尽量把字段设置 NOT NULL,例如'省份,性别',最好设置为 ENUM

2、使用连接 ( JOIN ) 来代替子查询:

a. 删除没有任何订单客户: DELETE FROM customerinfo WHERE customerid NOT in(SELECT customerid FROM orderinfo)

b. 提取所有没有订单客户: SELECT FROM customerinfo WHERE customerid NOT in(SELECT customerid FROM orderinfo)

c. 提高 b 的速度优化: SELECT FROM customerinfo LEFT JOIN orderid  
customerinfo.customerid=orderinfo.customerid  
WHERE orderinfo.customerid IS NULL

3、使用联合(UNION)来代替手动创建的临时表

a. 创建临时表: SELECT name FROM `nametest` UNION SELECT username FROM `nametest2`

4、事务处理:

a. 保证数据完整性,例如添加和修改同时,两者成立则都执行,一者失败都失败

```
mysql_query("BEGIN");
mysql_query("INSERT INTO customerinfo (name) VALUES ('$name1')");
mysql_query("SELECT * FROM `orderinfo` where customerid=\"$id\"");
mysql_query("COMMIT");
```

5、锁定表,优化事务处理:

a. 我们用一个 SELECT 语句取出初始数据,通过一些计算,用 UPDATE 语句将新值更新到表中。

包含有 WRITE 关键字的 LOCK TABLE 语句可以保证在 UNLOCK TABLES 命令被执行之前,

不会有其它的访问来对 inventory 进行插入、更新或者删除的操作

```
mysql_query("LOCK TABLE customerinfo READ, orderinfo WRITE");
mysql_query("SELECT customerid FROM `customerinfo` wheremso-spacerun: yes"> mysql_query("UPDATE
`orderinfo` SET ordertitle='$title' where customerid=\"$id\"");
mysql_query("UNLOCK TABLES");
```

6、使用外键,优化锁定表

a. 把 customerinfo 里的 customerid 映射到 orderinfo 里的 customerid,

任何一条没有合法的 customerid 的记录不会写到 orderinfo 里

```
CREATE TABLE customerinfo
(
customerid INT NOT NULL,
PRIMARY KEY(customerid)
) TYPE = INNODB;
CREATE TABLE orderinfo
(
orderid INT NOT NULL,
customerid INT NOT NULL,
PRIMARY KEY(customerid,orderid),
FOREIGN KEY (customerid) REFERENCES customerinfo
(customerid) ON DELETE CASCADE
) TYPE = INNODB;
```

注意: 'ON DELETE CASCADE', 该参数保证当 customerinfo 表中的一条记录删除的话同时也会删除 order 表中的该用户的所有记录, 注意使用外键要定义事务安全类型为 INNODB;

## 7、建立索引:

### a. 格式:

(普通索引) ->

创建: CREATE INDEX <索引名> ON tablename (索引字段)

修改: ALTER TABLE tablename ADD INDEX [索引名] (索引字段)

创表指定索引: CREATE TABLE tablename ([...], INDEX[索引名] (索引字段))

(唯一索引) ->

创建: CREATE UNIQUE <索引名> ON tablename (索引字段)

修改: ALTER TABLE tablename ADD UNIQUE [索引名] (索引字段)

创表指定索引: CREATE TABLE tablename ([...], UNIQUE[索引名] (索引字段))

(主键) ->

它是唯一索引, 一般在创建表时建立, 格式为:

CREATE TABLE tablename ([...], PRIMARY KEY[索引字段])

## 8、优化查询语句

a. 最好在相同字段进行比较操作，在建立好的索引字段上尽量减少函数操作

例子 1：

```
SELECT * FROM order WHERE YEAR(orderDate)<2008; (慢)
```

```
SELECT * FROM order WHERE orderDate<"2008-01-01"; (快)
```

例子 2：

```
SELECT * FROM order WHERE addtime/7<24; (慢)
```

```
SELECT * FROM order WHERE addtime<24*7; (快)
```

例子 3：

```
SELECT * FROM order WHERE title like "%good%";
```

```
SELECT * FROM order WHERE title>="good" and name<"good";
```

8、 PHP 的意思(送 1 分)

答：PHP 是一个基于服务端来创建动态网站的脚本语言，您可以用 PHP 和 HTML 生成网站主页

9、 MySQL 取得当前时间的函数是？，格式化日期的函数是(2 分)

答：now(), date()

10、 实现中文字串截取无乱码的方法。(3 分)

答：

```
function GBsubstr($string, $start, $length) {  
if(strlen($string)>$length){  
$str=null;  
$len=$start+$length;  
for($i=$start;$i<$len;$i++){  
if(ord(substr($string,$i,1))>0xa0){  
  
$str.=substr($string,$i,2);  
$i++;  
}  
}  
}  
return $str.'...';  
}  
else{  
$str.=substr($string,$i,1);  
}  
}
```

```
return $string;  
}  
}
```

11、您是否用过版本控制软件？如果有您用的版本控制软件的名字是？(1分)

12、您是否用过模板引擎？如果有您用的模板引擎的名字是？(1分)

答：用过，smarty

13、请简单阐述您最得意的开发之作(4分)

答：信息分类

14、对于大流量的网站，您采用什么样的方法来解决访问量问题？(4分)

答：确认服务器硬件是否足够支持当前的流量，数据库读写分离，优化数据表，  
程序功能规则，禁止外部的盗链，控制大文件的下载，使用不同主机分流主要流量

15、用 PHP 写出显示客户端 IP 与服务器 IP 的代码 1 分)

答：打印客户端 IP：echo \$\_SERVER['REMOTE\_ADDR']；或者：getenv('REMOTE\_ADDR');

打印服务器 IP：echo gethostbyname("www.bolaiwu.com")

16、语句 include 和 require 的区别是什么？为避免多次包含同一文件，可用（?）语句代替它们？(2分)

答：require->require 是无条件包含也就是如果一个流程里加入 require，无论条件成立与否都会先执行 require  
include->include 有返回值，而 require 没有（可能因为如此 require 的速度比 include 快）

注意：包含文件不存在或者语法错误的时候 require 是致命的，include 不是

17、如何修改 SESSION 的生存时间(1分)。

答：方法 1：将 php.ini 中的 session.gc\_maxlifetime 设置为 9999 重启 apache

方法 2：\$savePath = "./session\_save\_dir/";

\$lifeTime = 小时 \* 秒；

session\_save\_path(\$savePath);

session\_set\_cookie\_params(\$lifeTime);

session\_start();

方法 3：setcookie() and session\_set\_cookie\_params(\$lifeTime);

18、有一个网页地址，比如 PHP 开发资源网主页：<http://www.phpres.com/index.html>，如何得到它的内容？(\$1 分)

答：方法 1(对于 PHP5 及更高版本)：

```
$readcontents = fopen("http://www.phpres.com/index.html", "rb");
$contents = stream_get_contents($readcontents);
fclose($readcontents);
echo $contents;
```

方法 2：

```
echo file_get_contents("http://www.phpres.com/index.html");
```

19、在 HTTP 1.0 中，状态码 401 的含义是(?)；如果返回“找不到文件”的提示，则可用 header 函数，其语句为(?)；(2 分)

答：状态 401 代表未被授权，`header("Location:www.xxx.php")`；

12、在 PHP 中，heredoc 是一种特殊的字符串，它的结束标志必须？(1 分)

答：heredoc 的语法是用“<<<”加上自己定义成对的标签，在标签范围内的文字视为一个字符串

例子：

```
$str = <<<SHOW
my name is Jiang Qihui!
SHOW;
```

13、谈谈 asp,php,jsp 的优缺点(1 分)

答：ASP 全名 Active Server Pages，是一个 WEB 服务器端的开发环境，利用它可以产生和运行动态的、交互的、高性能的 WEB 服务应用程序。ASP 采用脚本语言 VB Script ( Java script ) 作为自己的开发语言。

PHP 是一种跨平台的服务器端的嵌入式脚本语言。它大量地借用 C, Java 和 Perl 语言的语法，并耦合 PHP 自己的特性，使 WEB 开发者能够快速地写出动态生成页面。它支持目前绝大多数数据库。还有一点，PHP 是完全免费的，不用花钱，你可以从 PHP 官方站点 (<http://www.php.net>) 自由下载。而且你可以不受限制地获得源码，甚至可以从中加进你自己需要的特色。

JSP 是 Sun 公司推出的新一代站点开发语言，他完全解决了目前 ASP, PHP 的一个通病——脚本级执行（据说 PHP4 也已经在 Zend 的支持下，实现编译运行）。Sun 公司借助自己在 Java 上的不凡造诣，将 Java 从 Java 应用程序 和 Java Applet 之外，又有新的硕果，就是 Js

---

p - - Java Server Page。Jsp 可以在 Serverlet 和 JavaBean 的支持下，完成功能强大的站点程序。

三者都提供在 HTML 代码中混合某种程序代码、由语言引擎解释执行程序代码的能力。但 JSP 代码被编译成 Servlet 并由 Java 虚拟机解释执行，这种编译操作仅在对 JSP 页面的第一次请求时发生。在 ASP 、 PHP、 JSP 环境下， HTML 代码主要负责描述信息的显示样式，而程序代码则用来描述处理逻辑。普通的 HTML 页面只依赖于 Web 服务器，而 ASP 、 PHP 、 JSP 页面需要附加的语言引擎分析和执行程序代码。程序代码的执行结果被重新嵌入到 HTML 代码中，然后一起发送给浏览器。 ASP 、 PHP 、 JSP 三者都是面向 Web 服务器的技术，客户端浏览器不需要任何附加的软件支持。

14、谈谈对 mvc 的认识(1 分)

答:由模型(model),视图(view),控制器(controller)完成的应用程序

由模型发出要实现的功能到控制器,控制器接收组织功能传递给视图;

15、写出发贴数最多的十个人名字的 SQL , 利用下表 : members(id,username,posts,pass,email) (2 分)

答:SELECT \* FROM `members` ORDER BY posts DESC limit 0,10;

16. 请说明 php 中传值与传引用的区别。什么时候传值什么时候传引用? (2 分)

答:按值传递：函数范围内对值的任何改变在函数外部都会被忽略

按引用传递：函数范围内对值的任何改变在函数外部也能反映出这些修改

优缺点：按值传递时，php 必须复制值。特别是对于大型的字符串和对象来说，这将会是一个代价很大的操作。

按引用传递则不需要复制值，对于性能提高很有好处。

17. 在 PHP 中 error\_reporting 这个函数有什么作用? (1 分)

答:设置错误级别与错误信息回报

18. 请写一个函数验证电子邮件的格式是否正确 (2 分)

答:function checkEmail (\$email)

```
{  
$pregEmail
```

=

```
"/([a-z0-9]*[-_\.]?[a-z0-9]+)*@[a-z0-9]*[-_]?[a-z0-9]+)[\.\.][a-z]{2,3}([\.\.][a-z]{2})?/i";
return preg_match($pregEmail,$email);
}
```

19. 简述如何得到当前执行脚本路径，包括所得到参数。 (2 分)

答:\$script\_name = basename(\_\_file\_\_); print\_r(\$script\_name);

21、JS 表单弹出对话框函数是?获得输入焦点函数是? (2 分)

答:弹出对话框: alert(),prompt(),confirm()

获得输入焦点 focus()

22、JS 的转向函数是?怎么引入一个外部 JS 文件? (2 分)

答>window.location.href,<script src="js/js\_function.js"></script>

23、foo() 和 @foo() 之间有什么区别? (1 分)

答:@foo() 控制错误输出

24、如何声明一个名为“myclass”的没有方法和属性的类? (1 分)

答:class myclass{ }

25、如何实例化一个名为“myclass”的对象? (1 分)

答:new myclass()

26、你如何访问和设置一个类的属性? (2 分)

答:\$object = new myclass();
\$newstr = \$object->test;
\$object->test = "info";

27、mysql\_fetch\_row() 和 mysql\_fetch\_array 之间有什么区别? (1 分)

答:mysql\_fetch\_row 是从结果集取出 1 行数组,作为枚举  
mysql\_fetch\_array 是从结果集取出一行数组作为关联数组,或数字数组,两者兼得

28、GD 库是做什么用的？(1 分)

答：gd 库提供了一系列用来处理图片的 API，使用 GD 库可以处理图片，或者生成图片。

在网站上 GD 库通常用来生成缩略图或者用来对图片加水印或者对网站数据生成报表。

29、指出一些在 PHP 输入一段 HTML 代码的办法。(1 分)

答：echo "<a href='index.php'>aaa</a>";

30、下面哪个函数可以打开一个文件，以对文件进行读和写操作？(1 分)

- (a) fget() (b) file\_open() (c) fopen() (d) open\_file() [ c ]

31、下面哪个选项没有将 john 添加到 users 数组中？(1 分)

- (a) \$users[] = 'john';  
(b) array\_add(\$users, 'john');  
(c) array\_push(\$users, 'john');  
(d) \$users &#124;&#124;= 'john'; [ a , c ]

32、下面的程序会输入是否？(1 分)

```
$num = 10;  
function multiply(){  
    $num = $num * 10;  
}  
multiply();  
echo $num;  
?>
```

输出：10

33、使用 php 写一段简单查询，查出所有姓名为“张三”的内容并打印出来 (2 分)

表名 User

Name Tel Content Date

张三 13333663366 大专毕业 2006-10-11

张三 13612312331 本科毕业 2006-10-15

张四 021-55665566 中专毕业 2006-10-15

请根据上面的题目完成代码：

```
$mysql_db=mysql_connect("local","root","pass");  
@mysql_select_db("DB", $mysql_db);
```

```
$result = mysql_query("SELECT * FROM `user` WHERE name='张三'");
while($rs = mysql_fetch_array($result)){
echo $rs["tel"].$rs["content"].$rs["date"];
}
```

34、如何使用下面的类，并解释下面什么意思？(3)

```
class test{
function Get_test($num){
    $num=md5($num)."En";
    return $num;
}
}
```

答:\$testnum = "123";

```
$object = new test();
$encrypt = $object->Get_test($testnum);
echo $encrypt;
```

类 test 里面包含 Get\_test 方法，实例化类调用方法多字符串加密

35、写出 SQL 语句的格式：插入，更新，删除（4 分）

表名 User

Name Tel Content Date

张三 13333663366 大专毕业 2006-10-11

张三 13612312331 本科毕业 2006-10-15

张四 021-55665566 中专毕业 2006-10-15

(a) 有一新记录(小王 13254748547 高中毕业 2007-05-06)请用 SQL 语句新增至表中

```
mysql_query("INSERT INTO `user` (name,tel,content,date) VALUES
('小王','13254748547','高中毕业','2007-05-06')")
```

(b) 请用 sql 语句把张三的时间更新成为当前系统时间

```
$nowDate = date("Y-m-d");
mysql_query("UPDATE `user` SET date='".$nowDate."' WHERE name='张山'");
```

(c) 请写出删除名为张四的全部记录

```
mysql_query("DELETE FROM `user` WHERE name='张四'");
```

36、请写出数据类型(int char varchar datetime text)的意思；请问 varchar 和 char 有什么区别(2分)

答:int 是数字类型, char 固定长度字符串, varchar 实际长度字符串, datetime 日期时间型, text 文本字符串  
char 的场地固定为创建表设置的长度, varchar 为可变长度的字符

38、写出以下程序的输出结果 (1 分)

```
$b=201;  
$c=40;  
$a=$b>$c?5:  
echo $a;  
?>
```

答:4

39、检测一个变量是否有设置的函数是否?是否为空的函数是?(2 分)

答:isset(\$str),empty(\$str);

40、取得查询结果集总数的函数是?(1 分)

答:mysql\_num\_rows(\$result);

41、\$arr = array('james', 'tom', 'symfony'); 请打印出第一个元素的值 (1 分)

答:echo \$array[0];

42、请将 41 题的数组的值用'，'号分隔并合并成字符串输出(1 分)

答:for(\$i=0;\$i<count(\$array);\$i++){ echo \$array[\$i].","; }

43、\$a = 'abcdef'；请取出\$a 的值并打印出第一个字母(1 分)

答:echo \$a{0} 或 echo substr(\$a,0,1)

44、PHP 可以和 sql server/oracle 等数据库连接吗?(1 分)

答:当然可以

45、请写出 PHP5 权限控制修饰符(3 分)

答:public(公共), private(私用), protected(继承)

46、请写出 php5 的构造函数和析构函数 (2 分)

答:\_\_construct , \_\_destruct

47、完成以下:

(一) 创建新闻发布系统, 表名为 message 有如下字段 (3 分)

id 文章 id  
title 文章标题  
content 文章内容  
category\_id 文章分类 id  
hits 点击量

答:CREATE TABLE 'message' (  
'id' int(10) NOT NULL auto\_increment,  
'title' varchar(200) default NULL,  
'content' text,  
'category\_id' int(10) NOT NULL,  
'hits' int(20),  
PRIMARY KEY('id'),  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

(二) 同样上述新闻发布系统: 表 comment 记录用户回复内容, 字段如下 (4 分)

comment\_id 回复 id  
id 文章 id, 关联 message 表中的 id  
comment\_content 回复内容

现通过查询数据库需要得到以下格式的文章标题列表, 并按照回复数量排序, 回复最高的排在最前面

文章 id 文章标题 点击量 回复数量

用一个 SQL 语句完成上述查询, 如果文章没有回复则回复数量显示为 0

答:SELECT message.id id,message.title title,IF(message.`hits` IS NULL,0,message.`hits`) hits,  
IF(comment.`id` is NULL,0,count(\*)) number FROM message LEFT JOIN  
comment ON message.id=comment.id GROUP BY message.`id`;

(三) 上述内容管理系统，表 category 保存分类信息，字段如下 (3 分)

```
category_id int(4) not null auto_increment;
category_name varchar(40) not null;
```

用户输入文章时，通过选择下拉菜单选定文章分类

写出如何实现这个下拉菜单

答:

```
function categoryList()
{
$result=mysql_query("select category_id,category_name from category")
or die("Invalid query: " . mysql_error());
print("<select name='category' value=>\n");
while($rowArray=mysql_fetch_array($result))
{
print("<option
value='".$rowArray['category_id']."' >" . $rowArray['category_name'] . "</option>\n");
}
print("</select>");
}
```

编程题：

1. 写一个函数，尽可能高效的，从一个标准 url 里取出文件的扩展名

例如：http://www.sina.com.cn/abc/de/fg.php?id=1 需要取出 php 或 .php

答案 1：

```
function getExt($url){
$arr = parse_url($url);

$file = basename($arr['path']);
$ext = explode(".", $file);
return $ext[1];
}
```

答案 2：

```
function getExt($url) {
$url = basename($url);
$pos1 = strpos($url, ".");
$pos2 = strpos($url, "?");
if(strstr($url, "?")){
return substr($url, $pos1 + 1, $pos2 - $pos1 - 1);
} else {
```

```
return substr($url,$pos1);
}
}
```

2. 在 HTML 语言中，页面头部的 meta 标记可以用来输出文件的编码格式，以下是一个标准的 meta 语句

请使用 PHP 语言写一个函数，把一个标准 HTML 页面中的类似 meta 标记中的 charset 部分值改为 big5

请注意：

1. 需要处理完整的 html 页面，即不光此 meta 语句
2. 忽略大小写
3. ‘ 和 ” 在此处是可以互换的
4. ‘Content-Type’ 两侧的引号是可以忽略的，但 ‘text/html; charset=gbk’ 两侧的不行
5. 注意处理多余空格

3. 写一个函数，算出两个文件的相对路径

如 \$a = '/a/b/c/d/e.php';  
\$b = '/a/b/12/34/c.php';

计算出 \$b 相对于 \$a 的相对路径应该是 ../../c/d 将()添上

答:

```
function getRelativePath($a, $b) {
$returnPath = array(dirname($b));
$arrA = explode('/', $a);
$arrB = explode('/', $returnPath[0]);
for ($n = 1, $len = count($arrB); $n < $len; $n++) {
if ($arrA[$n] != $arrB[$n]) {
break;
}
}
if ($len - $n > 0) {
$returnPath = array_merge($returnPath, array_fill(1, $len - $n, '..'));
}

$returnPath = array_merge($returnPath, array_slice($arrA, $n));
return implode('/', $returnPath);
}
echo getRelativePath($a, $b);
```

## PHP100 面试题题

### 1、PHP 的意思？

**答：**PHP 是一个基于服务端来创建动态网站的脚本语言，您可以用 PHP 和 HTML 生成网站主页

### 2、谈谈 asp,php,jsp 的优缺点？

**答：**ASP 全名 Active Server Pages，是一个 WEB 服务器端的开发环境，利用它可以产生和运行动态的、交互的、高性能的 WEB 服务应用程序。ASP 采用脚本语言 VB Script ( Java script ) 作为自己的开发语言。

PHP 是一种跨平台的服务器端的嵌入式脚本语言。它大量地借用 C, Java 和 Perl 语言的语法，并耦合 PHP 自己的特性，使 WEB 开发者能够快速地写出动态生成页面。它支持目前绝大多数数据库。还有一点，PHP 是完全免费的，不用花钱，你可以从 PHP 官方站点 (<http://www.php.net>) 自由下载。而且你可以不受限制地获得源码，甚至可以从中加进你自己需要的特色。

JSP 是 Sun 公司推出的新一代站点开发语言，他完全解决了目前 ASP, PHP 的一个通病 - - 脚本级执行（据说 PHP4 也已经在 Zend 的支持下，实现编译运行）。Sun 公司借助自己在 Java 上的不凡造诣，将 Java 从 Java 应用程序 和 Java Applet 之外，又有新的硕果，就是 Jsp - - Java Server Page。Jsp 可以在 Serverlet 和 JavaBean 的支持下，完成功能强大的站点程序。

三者都提供在 HTML 代码中混合某种程序代码、由语言引擎解释执行程序代码的能力。但 JSP 代码被编译成 Servlet 并由 Java 虚拟机解释执行，这种编译操作仅在对 JSP 页面的第一次请求时发生。在 ASP 、 PHP、 JSP 环境下，HTML 代码主要负责描述信息的显示样式，而程序代码则用来描述处理逻辑。普通的 HTML 页面只依赖于 Web 服务器，而 ASP 、 PHP、 JSP 页面需要附加的语言引擎分析和执行程序代码。程序代码的执行结果被重新嵌入到 HTML 代码中，然后一起发送给浏览器。 ASP 、 PHP、 JSP 三者都是面向 Web 服务器的技术，客户端浏览器不需要任何附加的软件支持。

### 3、谈谈对 mvc 的认识？

**答：**由模型 (Model)，视图 (View)，控制器 (Controller) 完成的应用程序

由模型发出要实现的功能到控制器，控制器接收组织功能传递给视图；

### 4、写出发帖数最多的十个人名字的 SQL，利用下表：members (id,username,posts,pass,email)

答:SELECT \* FROM `members` ORDER BY **posts** DESC limit 0,10;

**5、GD 库是做什么用的？**

答:gd 库提供了一系列用来处理图片的功能，使用 GD 库可以处理图片，或者生成图片。

在网站上 GD 库通常用来生成缩略图或者用来对图片加水印或者对网站数据生成报表。

**6、请写出数据类型(**int char varchar datetime text**)的意思；请问 varchar 和 char 有什么区别？**

答:**int** 是数字类型, **char** 固定长度字符串, **varchar** 实际长度字符串, **datetime** 日期时间型, **text** 文本字符串  
**char** 的场地固定为创建表设置的长度, **varchar** 为可变长度的字符

**7、写出以下程序的输出结果？**

```
<? Php  
$b=201;  
$c=40;  
$a=$b>$c?4:5;  
echo $a;  
?>
```

答:4

**8、检测一个变量是否有设置的函数是？是否为空的函数是？**

答:**isset(\$str)**, **empty(\$str)**;

**9、取得查询结果集总数的函数是？**

答:**mysql\_num\_rows(\$result)**;

**10、**\$arr = array('james', 'tom', 'symfony');** 请打印出第一个元素的值？**

答:**echo \$arr[0];**

**11、PHP 可以和 sql server/oracle 等数据库连接吗？**

答:可以

**12、请写出 PHP5 权限控制修饰符？**

答:public(公共),private(私用),protected(继承)

**13、请写出 php5 的构造函数和析构函数 ?**

答: 构造函数 : 官方称自定义函数

析构函数 : 垃圾回收函数 ( \_\_destruct )

**14、表单中 get 与 post 提交方法的区别?**

答: get 是发送请求 HTTP 协议通过 url 参数传递进行接收, 而 post 是实体数据, 可以通过表单提交大量信息.

**15、session 与 cookie 的区别?**

答: session: 储存用户访问的全局唯一变量, 存储在服务器上的 php 指定的目录中的 ( session\_dir ) 的位置进行的存放

cookie: 用来存储连续访问一个页面时所使用, 是存储在客户端, 对于 Cookie 来说是存储在用户 WIN 的 Temp 目录中的。

两者都可通过时间来设置时间长短

**16、用 PHP 打印出前一天的时间格式是 2010-7-3 12:28:21 ?**

答: echo date('Y-m-d H:i:s', strtotime('-1 days'));

**17、echo() ,print() ,print\_r() 的区别 ?**

答: echo 是 PHP 语句, print 和 print\_r 是函数, 语句没有返回值, 函数可以有返回值 (即便没有用)

print( ) 只能打印出简单类型变量的值 (如 int, string)

print\_r( ) 可以打印出复杂类型变量的值 (如数组, 对象)

echo 输出一个或者多个字符串

**18、能够使 HTML 和 PHP 分离开使用的模板 ?**

答: Smarty, TinyButStrong, XTemplate, Savant, Template Lite, Dwoo, phemplate

**19、使用哪些工具进行版本控制?**

答:cvs, svn, vss;

**20、如何实现字符串翻转?**

答:echo strrev(\$a); //strrev --- 颠倒字符串

## 21、优化查询语句？

a. 最好在相同字段进行比较操作，在建立好的索引字段上尽量减少函数操作

例子 1：

SELECT \* FROM order WHERE YEAR(orderDate)<2008; (慢)

SELECT \* FROM order WHERE orderDate<"2008-01-01"; (快)

例子 2：

SELECT \* FROM order WHERE addtime/7<24; (慢)

SELECT \* FROM order WHERE addtime<24\*7; (快)

例子 3：

SELECT \* FROM order WHERE title like "%good%";

SELECT \* FROM order WHERE title>="good" and name<"good";

## 22、对于大流量的网站，您采用什么样的方法来解决访问量问题？

答：确认服务器硬件是否足够支持当前的流量，数据库读写分离，优化数据表，

程序功能规则，禁止外部的盗链，控制大文件的下载，使用不同主机分流主要流量

## 23、用 PHP 写出显示客户端 IP 与服务器 IP 的代码？

答：打印客户端 IP:echo \$\_SERVER[ 'REMOTE\_ADDR' ]；或者： getenv('REMOTE\_ADDR');//getenv 取得开发环境变量

打印服务器 IP:echo gethostbyname("www.bolaiwu.com") // gethostbyname 取得 IP 地址函数

## 24、语句 include 和 require 的区别是什么？为避免多次包含同一文件，可用(?)语句代替它们？

答：require->require 是无条件包含也就是如果一个流程里加入 require，无论条件成立与否都会先执行 require  
include->include 有返回值，而 require 没有（可能因为如此 require 的速度比 include 快）

注意：包含文件不存在或者语法错误的时候 require 是致命的，include 不是

## 25、如何修改 SESSION 的生存时间？.

答：方法 1：将 php.ini 中的 session.gc\_maxlifetime 设置为 9999 重启 apache

```
方法 2:$savePath = "./session_save_dir/";  
$lifeTime = 小时 * 秒;  
session_save_path($savePath);  
session_set_cookie_params($lifeTime);  
session_start();  
  
方法 3:setcookie() and session_set_cookie_params($lifeTime);
```

**26、有一个网页地址，比如 PHP 开发资源网主页：http://www.php100.com/index.html，如何得到它的内容？**

**答：**方法 1(对于 PHP5 及更高版本)：

```
$readcontents = fopen("http://www.php100.com/index.html", "rb");  
$contents = stream_get_contents($readcontents); // stream_get_contents 取得字符串赋值给$contents  
fclose($readcontents);  
echo $contents;
```

方法 2：

```
echo file_get_contents("http://www.php100.com/index.html");
```

// file\_get\_contents() 函数把整个文件读入一个字符串中。

**27、在 HTTP 1.0 中，状态码 401 的含义是(?)；如果返回“找不到文件”的提示，则可用 header 函数，其语句为？**

**答：**状态 401 代表未被授权，header("Location:www.xxx.php");

**28、在 PHP 中，heredoc 是一种特殊的字符串，它的结束标志必须？**

**答：**heredoc 的语法是用"<<<"加上自己定义成对的标签，在标签范围内的文字视为一个字符串

例子：

```
$str = <<<SHOW  
my name is Jiang Qihui!  
SHOW;
```

**29、foo() 和 @foo() 之间有什么区别？**

**答：**@foo() 控制错误输出

**30、如何声明一个名为“myclass”的没有方法和属性的类？**

答: class myclass{ }

**31、如何实例化一个名为“myclass”的对象？**

答: new myclass()

**32、你如何访问和设置一个类的属性？**

答:\$object = new myclass();  
\$newstr = \$object->test;  
\$object->test = "info";

**33、mysql\_fetch\_row() 和 mysql\_fetch\_array 之间有什么区别？**

答: mysql\_fetch\_row 是从结果集取出 1 行数组，作为枚举  
mysql\_fetch\_array 是从结果集取出一行数组作为关联数组，或数字数组，两者兼得

**34、下面哪个函数可以打开一个文件，以对文件进行读和写操作？**

- (a) fget() (b) file\_open() (c) fopen() (d) open\_file()

答：c

**35、下面的程序会输入是否？**

```
<?php  
$num = 10;  
function multiply(){  
$num = $num * 10;  
}  
multiply();  
echo $num;  
?>
```

否，局部变量

**36、JS 表单弹出对话框函数是？获得输入焦点函数是？**

答: 弹出对话框: alert(), prompt(), confirm()

获得输入焦点 focus()

**37、JS 的转向函数是?怎么引入一个外部 JS 文件?**

答:window.location.href;

```
<script type="text/javascript" src="js/js_function.js"></script>
```

**38、\$a = 'abcdef'；请取出\$a 的值并打印出第一个字母？**

答:echo \$a{0} 或 echo substr(\$a,0,1)

**39、优化 MySQL 数据库的方法。**

- (1) .选取最适用的字段属性,应该尽量把字段设置为 NOT NULL ,这样在将来执行查询的时候,数据库不用去比较 NULL 值。
- (2) .使用连接 ( JOIN ) 来代替子查询 (Sub-Queries)
- (3) .尽量少使用 LIKE 关键字和通配符

**40、如何使用下面的类,并解释下面什么意思?**

```
class test{  
function Get_test($num){  
$num=md5(md5($num)."En");  
return $num;  
} }
```

双重 md5 加密

```
$testObject = new test();  
$encryption = $testObject->Get_test("xiaotian_ls");
```

**41 、请举例说明在你的开发过程中用什么方法来加快页面的加载速**

答 :要用到服务器资源时才打开,及时关闭服务器资源,数据库添加索引,页面可生成静态,图片等大文件单独服务器。使用代码优化工具啦

**42.写出一个正则表达式,过滤网页上的所有 JS/VBS 脚本(即把 script 标记及其内容都去掉):**

```
<?php  
$script="以下内容不显示：<script language='javascript'>alert('cc');</script>";  
echo preg_replace("/<script[^>].*?>.*?</script>/si", "替换内容", $script);  
?>
```

**43.以 Apache 模块的方式安装 PHP，在文件 http.conf 中首先要用语句\_\_\_\_动态装载 PHP 模块，然后再用语句\_\_\_\_使得 Apache 把所有扩展名为 php 的文件都作为 PHP 脚本处理。**

答：LoadModule

```
php5_module  
"c:/php/php5apache2.dll";AddType application/x-httdp-php .php
```

**44.sort()、asort()、和 ksort() 有什么分别？它们分别在什么情况下使用？**

sort()

根据阵列中元素的值，以英文字母顺序排序，索引键会由 0 到 n-1 重新编号。主要是当阵列索引键的值无关疼痛时用来把阵列排序。

asort()

与 sort() 一样把阵列的元素按英文字母顺序来排列，不同的是所有索引键都获得保留，特别适合替联想阵列排序。

ksort()

根据阵列中索引键的值，以英文字母顺序排序，特别适合用于希望把索引键排序的联想阵列。

**45.“==”是什么？试举一个“==”是真但“==”是假的例子。**

“==”既可以返回布尔值“假”，也可以返回一个不是布尔值但却可以赋与“假”值的函式，strpos() 和 strrpos() 便是其中两个例子。

```
if(strpos("abc", "a") == true){ // 这部分永不会被执行，因为 "a" 的位置是 0，换算成布尔值"假"}if(strpos("abc", "a") ===  
true){ // 这部份会被执行，因为“==”保证函式 strpos() 的送回值不会换算成布尔值.}
```

**46、写出以下程序的输出结果**

```
<?php  
$str= "cd" ;  
$$str= "hotdog" ;  
$$str.= "ok" ;  
echo $cd;  
?>
```

答案：hotdogok

**47、给你三个数，写程序求出其最大值。**

```
$var1=1;  
$var2=7;  
$var3=8;  
$max=$var1>$var2?$var1:$var2;  
$max=$max>$var3?$max:$var3;  
echo $max;
```

**48 写出将一个选择 2 号选项的 ip 为 127.0.0.1 的用户在当前时间的投票记录到数据库的 SQL**

答：insert into poll (ip,time,iid) values('127.0.0.1',now(),2);

**49. 请写一个函数验证电子邮件的格式是否正确？**

```
答:function checkEmail($email)  
{  
    $pregEmail =  
"/([a-z0-9]*[-_\.]?[a-z0-9]+)*@[a-z0-9]*[-_]?[a-z0-9]+)[\.\.][a-z]{2,3}([\.\.][a-z]{2})?/i";  
    return preg_match($pregEmail,$email);  
}
```

**50、有一表 menu (mainmenu, submenu, url), 请用递归法写出一树形菜单，将所有的 menu 列出来**

```
<html>  
<head><title>这里是标题</title></head>  
<body>  
<form>  
<?php  
function GenerateMenu($id=0,$str="")  
{  
    $result=mysql_query("select mainmenu,url,submenu from menu where mainmenu=$id");  
    while($row=mysql_fetch_array($result))  
    {  
        echo $str.$row["url"]."<br />";  
        GenerateMenu($row["submenu"],$str.--);  
    }  
    mysql_free_result($result);  
}  
$link=mysql_connect("localhost","root","");
mysql_select_db("phpinterview");
```

---

```
GenerateMenu();
mysql_close($link)
?>
```

## 51、写出 SQL 语句的格式：插入，更新，删除

表名 User

	Name	Tel	Content	Date
张三	13333663366	大专毕业	2006-10-11	
张三	13612312331	本科毕业	2006-10-15	
张四	021-55665566	中专毕业	2006-10-15	

(a) 有一新记录 (小王 13254748547 高中毕业 2007-05-06) 请用 SQL 语句新增至表中

```
mysql_query("INSERT INTO `user` (name,tel,content,date) VALUES
('小王','13254748547','高中毕业','2007-05-06')")
```

(b) 请用 sql 语句把张三的时间更新成为当前系统时间

```
$nowDate = date("Ymd");
mysql_query("UPDATE `user` SET date='".$nowDate."' WHERE name='张山'");
```

(c) 请写出删除名为张四的全部记录

```
mysql_query("DELETE FROM `user` WHERE name='张四'");
```

1. 以下哪一句不会把 John 新增到 users 阵列？

```
$users[] = 'john';
```

**成功把 John 新增到阵列 users。**

```
array_add($users,'john');
```

**函式 array\_add() 无定义。**

```
array_push($users,'john');
```

**成功把 John 新增到阵列 users。**

```
$users ||= 'john';
```

**语法错误。**

2. sort()、assort()、和 ksort() 有什么分别？它们分别在什么情况下使用？

```
sort()
```

根据阵列中元素的值，以英文字母顺序排序，索引键会由 0 到  $n-1$  重新编号。主要是当阵列索引键的值无关疼痛时用来把阵列排序。

assort()

PHP 没有 assort() 函数，所以可能是 asort() 的笔误。

asort()

与 sort() 一样把阵列的元素按英文字母顺序来排列，不同的是所有索引键都获得保留，特别适合替联想阵列排序。

ksort()

根据阵列中索引键的值，以英文字母顺序排序，特别适合用于希望把索引键排序的联想阵列。

3. 以下的代码会产生什么？为什么？

```
$num = 10;  
function multiply() {  
    $num = $num * 10;  
}  
multiply();  
echo $num;
```

由于函数 multiply() 没有指定 \$num 为全域变量（例如 global \$num 或者 \$\_GLOBALS['num']），所以 \$num 的值是 10。

4. reference 跟一个正规的变量有什么分别？如何 pass by reference？在什么情况下我们需要这样做？

Reference 传送的是变量的地址而非它的值，所以在函数中改变一个变量的值时，整个应用都见到这个变量的新值。

一个正规变量传送给函数的是它的值，当函数改变这个变量的值时，只有这个函数才见到新值，应用的其他部分仍然见到旧值。

```
$myVariable = "its' value";  
Myfunction(&$myVariable); // 以 reference 传送参数以 reference 传送参数给函数，可以使函数改变了的变量，即使在函数结束后仍然保留新值。
```

5. 些函数可以用来在现正执行的脚本中插入函数库？

对这道题目不同的理解会有不同的答案，我的第一个想法是插入 PHP 函数库不外乎 include()、include\_once()、require()、require\_once()，但细心再想，“函数库”也应该包括 com 物件和 .net 函数库，所以我们的答案也要分别包括 com\_load 和 dotnet\_load，下次有人提起“函数库”的时候，别忘记这两个函数。

#### 6. `foo()` 与 `@foo()` 有什么分别？

`foo()` 会执行这个函式，任何解译错误、语法错误、执行错误都会在页面上显示出来。

`@foo()` 在执行这个函式时，会隐藏所有上述的错误讯息。

很多应用程序都使用 `@mysql_connect()` 和 `@mysql_query` 来隐藏 mysql 的错误讯息，我认为这是很严重的失误，因为错误不该被隐藏，你必须妥善处理它们，可能的话解决它们。

#### 7. 你如何替 PHP 的应用程序侦错？

我并不常这样做，我曾经试过很多不同的侦错工具，在 Linux 系统中设定这些工具一点也不容易。不过以下我会介绍一个近来颇受注目的侦错工具。

PHP - Advanced PHP Debugger 或称 PHP - APD，第一步是执行以下的指令安装：

`pear install apd` 安装后在你的脚本的开头位置加入以下的语句开始进行侦错：

`apd_set_pprof_trace();` 执行完毕，打开以下档案来查阅执行日志：

`apd.dumpdir`

你也可以使用 `pprof` 来格式化日志。

详细的资料可以参阅 <http://us.php.net/manual/en/ref.apd.php>。

#### 8. “==”是什么？试举一个“==”是真但“==”是假的例子。

“==”是给既可以送回布尔值“假”，也可以送回一个不是布尔值但却可以赋与“假”值的函式，`strpos()` 和 `strrpos()` 便是其中两个例子。

问题的第二部份有点困难，想一个“==”是假，但是“==”是真的例子却很容易，相反的例子却很少。但我终于找到以下的例子：

```
if (strpos("abc", "a") == true) { // 这部分永不会被执行，因为 "a" 的位置是 0，换算成布尔值"假"} if  
(strpos("abc", "a") === true) { // 这部份会被执行，因为"=="保证函式 strpos() 的送回值不会换算成布尔  
值.}
```

#### 9. 你会如何定义一个没有成员函式或特性的类别 `myclass`？

```
class myclass{}
```

10. 你如何产生一个 myclass 的物件？

```
$obj = new myclass();
```

11. 在一个类别内如何存取这个类别的特性及变更它的值？

使用语句：`$this->propertyName`，例如：

```
class myclass{ private $propertyName; public function __construct() { $this->propertyName = "value"; }}
```

12. `include` 和 `include_once` 有什么分别？`require` 又如何？

三者都是用来在脚本中插入其他档案，视乎 `url_allow_fopen` 是否核准，这个档案可以从系统内部或外部取得。但他们之间也有微细的分别：

`include()`：这个函式容许你在脚本中把同一个档案插入多次，若果档案不存在，它会发出系统警告并继续执行脚本。

`include_once()`：它跟 `include()` 的功能相似，正如它的名字所示，在脚本的执行期间，有关档案只会被插入一次。

`require()`：跟 `include()` 差不多，它也是用来在脚本中插入其他档案，但若果档案不存在，它会发出系统警告，这个警告会引致致命错误令脚本中止执行

13. 以下哪一个函式可以把浏览器转向到另一个页面？

`redir()`

这不是一个 PHP 函式，会引致执行错误。

`header()`

这个是正确答案，`header()` 用来插入卷头资料，可以用来使浏览器转向到另一个页面，例如：

```
header("Location: http://www.search-this.com/");  
location()
```

这不是一个 PHP 函式，会引致执行错误。

`redirect()`

这不是一个 PHP 函式，会引致执行错误。

14. 以下哪一个函式可以用来开启档案以便读 / 写？

fget()

这不是一个 PHP 函数，会引致执行错误。

file\_open()

这不是一个 PHP 函数，会引致执行错误。

fopen()

这是正确答案，fopen() 可以用来开启档案以便读 / 写，事实上这个函数还有很多选项，详细资料请参阅 [php.net](http://php.net)。

open\_file()

这不是一个 PHP 函数，会引致执行错误。

15.mysql\_fetch\_row() 和 mysql\_fetch\_array() 有什么分别？

mysql\_fetch\_row() 把数据库的一列储存在一个以零为基数的阵列中，第一栏在阵列的索引 0，第二栏在索引 1，如此类推。mysql\_fetch\_assoc() 把数据库的一列储存在一个关联阵列中，阵列的索引就是栏位名称，例如我的数据库查询送回“first\_name”、“last\_name”、“email”三个栏位，阵列的索引便是“first\_name”、“last\_name”和“email”。

mysql\_fetch\_array() 可以同时送回 mysql\_fetch\_row() 和 mysql\_fetch\_assoc() 的值。

16.下面的代码用做什么？请解释。

```
$date='08/26/2003';print  
ereg_replace("([0-9]+)/([0-9]+)/([0-9]+)", "[url=file://2///1///3%22,$date]\\"2/"\\1/"\\3", $date  
/[url]);
```

这是把一个日期从 MM/DD/YYYY 的格式转为 DD/MM/YYYY 格式。我的一个好朋友告诉我可以把这个正规表达式拆解为以下的语句，对于如此简单的表示来说其实无须拆解，纯粹为了解说的方便：

```
// 对应一个或更多 0-9，后面紧随一个斜号$regExpression = "([0-9]+)/"; // 应一个或更多 0-9，后面紧随另一个  
斜号$regExpression .= "([0-9]+)/"; // 再次对应一个或更多 0-9$regExpression .= "([0-9]+)"; 至于  
[url=file://2///1///3]\\"2/"\\1/"\\3[/url] 则是用来对应括号，第一个括号对的是月份，第二个括号对应的是日期，  
第三个括号对应的是年份。
```

17.给你一行文字 \$string，你会如何编写一个正规表达式，把 \$string 内的 HTML 标签除去？

首先，PHP 有内建函式 `strip_tags()` 除去 HTML 标签，为何要自行编写正规表达式？好了，便当作是面试的一道考题吧，我会这样回答：

```
$stringOfText = "<p>This is a test</p>";$expression = "/<(.*)>(.*)<\/(.*)>/" ;echo preg_replace($expression, "[url=file://2/]\\2[/url]", $stringOfText); // 有人说也可以使用 /(<[^>]*>)/ $expression = "/(<[^>]*>)/";echo preg_replace($expression, "", $stringOfText);
```

#### 18. PHP 和 Perl 分辨阵列和散列表的方法有什么差异？

这正是为何我老是告诉别人选择适当的编程语言，若果你只用一种语言的话你怎么能回答这道问题？这道问题很简单，Perl 所有阵列变量都是以 @ 开头，例如 @myArray，PHP 则沿用 \$ 作为所有变量的开头，例如 \$myArray。

至于 Perl 表示散列表则用 %，例如 %myHash，PHP 则没有分别，仍是使用 \$，例如 \$myHash。

#### 19. 你如何利用 PHP 解决 HTTP 的无状态本质？

最主要的俩各选择是 session 和 cookie。使用 session 的方法是在每一页的开始加上 `session_start()`，然后利用 `$_SESSION` 散列表来储存 session 变量。至于 cookie 你只需记着一个原则：在输出任何文字之前调用 `set_cookie()` 函数，此外只需使用 `$_COOKIE` 散列表便可以存取所有 cookie 变量。

还有一个不那么可靠的方法，就是利用访客的 IP 地址，这个方法有特定的危险性。

#### 20. GD 函式库用来做什么？

这个可能是我最喜欢的函式库，自从 PHP 4.3.0 版本后 GD 便内建在 PHP 系统中。这个函式库让你处理和显示各式格式的图档，它的另一个常见用途是制作所图档。GD 以外的另一个选择是 ImageMagick，但这个函式库并不内建于 PHP 之中，必须由系统管理员安装在伺服器上。

#### 21. 试写出几个输出一段 HTML 代码的方法。

嗯，你可以使用 PHP 中任何一种输出语句，包括 `echo`、`print`、`printf`，大部分人都使用如下例的 `echo`：

`echo "My string $variable";` 你也可以使用这种方法：

```
echo <<<ENDThis text is written to the screen as output and this $variable is parsed too. If you wanted you can have <span> HTML tags in here as well.</span> The END; remarks must be on a line
```

---

of its own, and can't contain any extra white space.END;

## 22. PHP 比 Perl 好吗？请讨论。

我们不要为一个简单的问题引发一场舌战，正如我经常说的：“为工作选择适合的语言，不要把工作迁就语言。”我个人认为 Perl 十分适合用作命令行工具，虽然它在网页应用上也有不错的表现，但是它的真正实力在命令行上才能充分发挥。同样地，PHP 虽然可以在控制台的环境中使用，但是个人认为它在网页应用上有更好的表现，PHP 有大量专门为网页应用而设计的函式，Perl 则似乎以命令行为设计之本。

个人来说两种语言我都喜欢，在大学期间我经常使用 Perl、PHP 和 Java，可惜工作上我使用 C#，但在家里我花不少时间操练 PHP、Perl、Ruby（现正学习）和 Java，保持我的技能知识在最新状态。很多人问我 C 和 C++ 怎么样，它们是否仍有机会在我的应用中占一席位，我的答案基本上是“否”，我近来的工作主要集中在网页开发，虽然 C 和 C++ 也可以用来写网页，但它们到底不是为这种工作而设计的，“为工作选择适合的语言”，若果我需要编写一个控制台应用，用来展示 bubble sort、quick sort 和 merge sort 的效能比较，我一定会使用 C / C++。若果我需要编写一个相片簿系统，我会使用 PHP 或者 C#（我认为制作用户界面方面 .NET 语言比网页更加）。

### 1. 优化 SQL: `select * from t where id in (1,2,3) order by s desc;`

答：

不要用 `select *` 就算有几十个字段要读也一个一个写出来。

用 `exists` 代替 `in, exists` 的效率要高很多，测试过有时速度可以相差5倍。

`s` 字段建索引。

---

### 2. 写一段 Mysql 连接：

答：

```
mysql_connect("mysql_host", "mysql_user", "mysql_password");
mysql_select_db("my_database");
$query = "SELECT * FROM my_table";
$result = mysql_query($query);
while ($line = mysql_fetch_array($result, MYSQL_ASSOC)) {
echo $row[0];
}
```

---

### 3. 写出 PHP 的变量类型：

答：

标量类型：

boolean ( 布尔型 )

integer ( 整型 )

float ( 浮点型 , 也作 "double" )

string ( 字符串 )

复合类型 :

array ( 数组 )

object ( 对象 )

特殊类型 :

resource ( 资源 )

NULL

---

#### 4. 文件上传进度条怎么做:

答:

1. APC 扩展模块

2. PHP+Flash

3. 第三方类库

---

#### 5. echo() , print() , print\_r() , var\_dump() 的区别:

答:

print 和 echo 功能基本相同, 不同的是 print 有返回值

print\_r 和 var\_dump 是递归打印, 用于输出数组对象, 不同的是 var\_dump 可以给出数组中值的类型

---

#### 6. 实现中文字串截取无乱码的方法:

答:

装扩展库, 用 mb\_string()

---

#### 7. 用 PHP 打印出前一天的时间格式是 2006-5-10 22:21:21

答:

echo date('Y-m-d H:i:s', strtotime('-1 day'));

---

#### 8. 实现字符串翻转:

答:

php 自带 strrev 函数

如不准用 PHP 内置的就如下写

```
function my_strrev($str)
{
    $len=strlen($str);
    $newstr = "";
    for($i=$len;$i>=0;$i--)
    {
        $newstr .= $str{$i};
    }
    return $newstr;
}
```

## 9. 三元运算:

The expression (expr1) ? (expr2) : (expr3) evaluates to expr2 if expr1 evaluates to TRUE, and expr3 if expr1 evaluates to FALSE.

考一些运算符:

```
1 echo "<h3>Postincrement</h3>";
2 $a = 5;
3 echo "Should be 5: " . $a++ . "\n";
4 echo "Should be 6: " . $a . "\n";
5
6 echo "<h3>Preincrement</h3>";
7 $a = 5;
8 echo "Should be 6: " . ++$a . "\n";
9 echo "Should be 6: " . $a . "\n";
10
11 echo "<h3>Postdecrement</h3>";
12 $a = 5;
13 echo "Should be 5: " . $a-- . "\n";
14 echo "Should be 4: " . $a . "\n";
15
16 echo "<h3>Predecrement</h3>";
17 $a = 5;
18 echo "Should be 4: " . --$a . "\n";
19 echo "Should be 4: " . $a . "\n";
20
21 // foo() will never get called as those operators are short-circuit
22 $a = ($false && foo());
23 $b = ($true || foo());
24 $c = ($false and foo());
25 $d = ($true or foo());
26
27 // "||" has a greater precedence than "or"
28 $e = $false || $true; // $e will be assigned to (false || true) which is true
29 $f = $false or $true; // $f will be assigned to false
30 var_dump($e, $f);
31
32 // "&&" has a greater precedence than "and"
33 $g = $true && $false; // $g will be assigned to (true && false) which is false
34 $h = $true and $false; // $h will be assigned to true
35 var_dump($g, $h);
36 /*
37 The above example will output something similar to:
38 bool(true)
```

```
39 bool(false)
40 bool(false)
41 bool(true)
42 */
```

- 在 PHP 中，当前脚本的名称（不包括路径和查询字符串）记录在预定义变量（1）中；而链接到当前页面的前一页 URL 记录在预定义变量（2）中 <?php

```
//本页地址，SCRIPT_NAME 也可以:php/test.php
echo $_SERVER['PHP_SELF']."<br />";
//链接到当前页面的前一页的 URL 地址:
echo $_SERVER['HTTP_REFERER']."<br />";
//其它的见参考手册：语言参考》变量》预定义变量
//前执行脚本的绝对路径名:D:\inetpub\wwwroot\php\est.php
echo $_SERVER["SCRIPT_FILENAME"]."<br />";
//正在浏览当前页面用户的 IP 地址:127.0.0.1
echo $_SERVER["REMOTE_ADDR"]."<br />";
//查询(query)的字符串(URL 中第一个问号 ? 之后的内容):id=1&bi=2
echo $_SERVER["QUERY_STRING"]."<br />";
//当前运行脚本所在的文档根目录:d:\inetpub\wwwroot
echo $_SERVER["DOCUMENT_ROOT"]."<br />";
?>
```

- 执行程序段<?php echo 8%(-2) ?>将输出\_\_。

```
<?php

//参考手册》语言参考》运算符》算术运算符》%为取模运算,输出0

echo 8%(-2)."<br />";

//取模 $a % $b 在 $a 为负值时的结果也是负值。输出-2

echo ((-8)%3)."<br />";

//输出2

echo (8%(-3))."<br />";

?>
```

3. 在 HTTP 1.0 中，状态码 401 的含义是\_\_\_\_；如果返回“找不到文件”的提示，则可用 header 函数，其语句为\_\_\_\_。

答：401 表示未授权;header("HTTP/1.0 404 Not Found");[见参考手册》函数参考》HTTP 函数》header]

```
header("Location:www.xxxx.php");
```

4. 数组函数 arsort 的作用是\_\_\_\_；语句 error\_reporting(2047)的作用是\_\_\_\_。

答：arsort:对数组进行逆向排序并保持索引关系 error\_reporting(2047)的作用是:report All errors and warnings

5. 写出一个正则表达式，过滤网页上的所有 JS/VBS 脚本（即把 script 标记及其内容都去掉）：

```
<?php

$script="以下内容不显示:<script language='javascript'>alert('cc');</script>";

echo preg_replace("/<script[^>].*?>.*?</script>/si", "替换内容", $script);

?>
```

6. 以 Apache 模块的方式安装 PHP，在文件 http.conf 中首先要用语句\_\_\_\_动态装载 PHP 模块，

然后再用语句\_\_\_\_使得 Apache 把所有扩展名为 php 的文件都作为 PHP 脚本处理。

答：LoadModule php5\_module "c:/php/php5apache2.dll";AddType application/x-httdp-php .php

见参考手册》目录》II. 安装与配置》6. Windows 系统下的安装》Microsoft Windows 下的 Apache 2.0.x

7. 语句 include 和 require 都能把另外一个文件包含到当前文件中，它们的区别是\_\_\_\_；为了避免多次包含同一文件，可以用语句\_\_\_\_来代替它们。

答：在如何处理失败时，include() 产生一个警告而 require() 则导致一个致命错误;require\_once()/include\_once()

8. 一个函数的参数不能是对变量的引用，除非在 php.ini 中把\_\_\_\_设为 on.

答：allow\_call\_time\_pass\_reference boolean :是否启用在函数调用时强制参数被按照引用传递，见参考手册》附录 G

9. SQL 中 LEFT JOIN 的含义是\_\_\_\_，如果 tbl\_user 记录了学生的姓名(name)和学号(ID)，

tbl\_score 记录了学生（有的学生考试以后被开除了，没有其记录）的学号(ID)和考试成绩(score)以及考试科目(subject），要想打印出各个学生姓名及对应的各科总成绩，则可以用 SQL 语句\_\_\_\_.

答：自然左外连接

```
create database phpinterview;  
use phpinterview  
create table tbl_user  
(  
    ID int      not null,  
    name  varchar(50)      not null,  
    primary key (ID)  
create table tbl_score  
(
```

```
ID int not null,  
score dec(6, 2) not null,  
subject varchar(20) not null  
);  
  
insert into tbl_user (ID, name) values (1, 'beimu');  
insert into tbl_user (ID, name) values (2, 'aihui');  
  
insert into tbl_score (ID, score, subject) values (1, 90, '语文');  
  
insert into tbl_score (ID, score, subject) values (1, 80, '数学');  
  
insert into tbl_score (ID, score, subject) values (2, 86, '数学');  
  
insert into tbl_score (ID, score, subject) values (2, 96, '语文');  
  
select A.id, sum(B.score) as sumscore  
from tbl_user A left join tbl_score B  
on A.ID=B.ID  
group by A.id
```

10. 在 PHP 中，heredoc 是一种特殊的字符串，它的结束标志必须\_\_\_\_\_

答：结束标识符所在的行不能包含任何其它字符除”;"

11. 写一个函数，能够遍历一个文件夹下的所有文件和子文件夹。

```
<?php  
  
function my_scandir($dir)  
{  
    $files=array();  
    if(is_dir($dir))
```

```
{  
  
    if($handle=opendir($dir))  
  
    {  
  
        while(($file=readdir($handle))!==false)  
  
        {  
  
            if($file!="." && $file!="..")  
  
            {  
  
                if(is_dir($dir."/".$file))  
  
                {  
  
                    $files[$file]=my_scandir($dir."/".$file);  
  
                }  
  
                else  
  
                {  
  
                    $files[]=$dir."/".$file;  
  
                }  
  
            }  
  
        }  
  
        closedir($handle);  
  
        return $files;  
  
    }  
  
}  
  
print_r(my_scandir("D:\Program Files\Internet Explorer\MUI"));  
?>
```

## PHP 经典面试题

1 请说明 PHP 中传值与传引用的区别。什么时候传值什么时候传引用？

答：传值只是把某一个变量的值传给了另一个变量，而引用则说明两者指向了同一个地方。

2 在 PHP 中 error\_reporting 这个函数有什么作用？

答：The `error_reporting()` function sets the `error_reporting` directive at runtime. PHP has many levels of errors, using this function sets that level for the duration (runtime) of your script.

3 请用正则表达式 ( Regular Expression ) 写一个函数验证电子邮件的格式是否正确。

答：

```
<?php
if(isset($_POST['action']) &&
$_POST['action']=='submitted')
{
    $email=$_POST['email'];
    if(!preg_match("/^(:w+.?)*w+@(:w+.?)*w+$/", $email))
        {
            echo
        "电子邮件检测失败";
    }
}
else
```

```
        {
            echo
        "电子邮件检测成功";
    }

}

else
{
?>

<html>
<head><title>EMAIL 检测</title>
<script type="text/javascript">
function checkEmail(sText)
{
    var reg=/^ (:w+.)?*w+@(:w+.)?*w+\$/;
    var email=document.getElementById(sText).value;
    if(!reg.test(email))
    {
        alert("电子邮件检测失败");
    }
    else
    {
        alert("电子邮件格式正确");
    }
}
```

}

```
</script>

</head>

<body>

<form action=<?php echo $_SERVER['PHP_SELF'] ?> method="POST">

电子邮件 :<input type="text" id="email" name="email"

/><br />

<input type="hidden" name="action" value="submitted"

/>

<input type="button" name="button" value="客户端检测" onclick="checkEmail('email')"

/>

<input type="submit" name="submit" value="服务器端检测"

/>

</form>

</body>

</html>

<?php

}

?>
```

4 简述如何得到当前执行脚本路径，包括所得到参数。

```
<?php
echo
```

```
"http://". $_SERVER[' SERVER_NAME' ]. $_SERVER[' PHP_SELF' ]. "?". $_SERVER[' QUERY_STRING' ];  
  
//echo "http://". $_SERVER[' HTTP_HOST' ]. $_SERVER[' REQUEST_URI' ];  
  
?>
```

5 有一个一维数组，里面存储整形数据，请写一个函数，将他们按从大到小的顺序排列。要求执行效率高。并说明如何改善执行效率。（该函数必须自己实现，不能使用 php 函数）

```
<?php  
  
function BubbleSort (&$arr)  
{  
  
    $cnt=count ($arr) ;  
  
    $flag=1;  
  
    for ($i=0; $i<$cnt; $i++)  
    {  
  
        if ($flag==0)  
        {  
  
            return;  
        }  
  
        $flag=0;  
  
        for ($j=0; $j<$cnt-$i-1; $j++)  
        {  
  
            if ($arr[$j]>$arr[$j+1])  
            {  
  
                $tmp=$arr[$j];
```

```
$arr[$j] = $arr[$j+1];  
  
$arr[$j+1] = $tmp;  
  
$flag = 1;  
  
}  
  
}  
  
}  
  
}  
  
}  
  
$test = array(1, 3, 6, 8, 2, 7);  
  
BubbleSort($test);  
  
var_dump($test);  
  
?>
```

## 6 请举例说明在你的开发过程中用什么方法来加快页面的加载速度

答：要用到服务器资源时才打开，及时关闭服务器资源，数据库添加索引，页面可生成静态，图片等大文件单独服务器。

使用代码优化工具啦

新浪面试题：

### 1. echo count("abc"); 输出什么？

答："1"

count — 计算数组中的单元数目或对象中的属性个数

int count ( mixed \$var [, int \$mode ] ), 如果 var 不是数组类型或者实现了 Countable 接口的对象，将返回 1，

有一个例外，如果 var 是 NULL 则结果是 0。

对于对象，如果安装了 SPL，可以通过实现 Countable 接口来调用 count()。该接口只有一个方法 count()，此方法返回 count() 函数的返回值。

## 2. 用 PHP 写出显示客户端 IP 与服务器 IP 的代码

答：`$_SERVER['SERVER_ADDR']` 服务器

```
$_SERVER['REMOTE_ADDR'] 客户端
function getOnlineIP(){
if (getenv('HTTP_CLIENT_IP')) return getenv('HTTP_CLIENT_IP');
if (getenv('HTTP_X_FORWARDED_FOR')) return getenv('HTTP_X_FORWARDED_FOR');
if ($_SERVER["REMOTE_ADDR"]) return $_SERVER["REMOTE_ADDR"];
if (!empty($HTTP_SERVER_VARS['REMOTE_ADDR'])) return $HTTP_SERVER_VARS['REMOTE_ADDR'];
}
```

## 3. `error_reporting(2047)` 什么作用？

答：PHP 显示所有错误 `E_ALL`

## 4. `echo`, `print()` 和 `print_r()` 有什么区别？

答：`echo` 是一个语言结构，没有返回值。

`print` 是一个函数，返回 `int` 类型的值。[只能打印 `int string`]

`print_r()` 是一个函数，返回 `bool` 类型值，按结构输出变量的值。打印关于变量的易于理解的信息 [数组、对象等]

## 5. 打开 `php.ini` 中的 `safe_mode`，会影响哪些函数？至少说出 6 个。

答：1: 用户输入输出函数 (`fopen()` `file()` `require()`)，只能用于调用这些函数有相同脚本的拥有者)

2: 创建新文件 (限制用户只在该用户拥有目录下创建文件)

3: 用户调用 `popen()` `system()` `exec()` 等脚本，只有脚本处在 `safe_mode_exec_dir` 配置指令指定的目录中才可能

4: 加强 HTTP 认证，认证脚本拥有者的 `UID` 的划入认证领域范围内，此外启用安全模式下，不会设置 `PHP_AUTH`

5:mysql 服务器所用的用户名必须与调用 mysql\_connect() 的文件的拥有者用户名相同

6:受影响的函数变量以及配置命令达到 40 个

## 6. 写个函数来解决多线程同时读写一个文件的问题。

答：flock(\$hander,LOCK\_EX); 这个可是内置函数啊，

这个尚待解决

## 7. 请写一个函数验证电子邮件的格式是否正确（要求使用正则）

答：preg\_match('/^[\w\-\.\.]+\@[ \w\-\.]+\(\.\.\w+\)+\$/ ', \$email);

## 8. 考 SQL 语句的题，题太长了，实在不好回忆了。

答：去理解别人的回忆是件很困难的事情

## 9. MySQL 数据库，一天一万条以上的增量，怎么优化？

答：我们曾做过短信 SP 的东西，有个短信发送的日志表，每天增量也很大，处理的方法是按月进行分表，因为是日志表，主要操作是 insert 操作，所以每月初自动生成新的数据表，数据插入到对应月份的那张数据表。[比如表明前缀是 cdb\_smslog 后面加 200910 及时 cdb\_smslog\_200910]

其他优化方式暂时想不起来，对于 myISAM，考虑容量的话，也有优化的方案

但是对于那种查询操作的表的话，我的思路是根据作者的发布时间存储到不同的表里面

所以对 sina 那种海量数据的处理很感兴趣，很好奇他们的处理方法，[以前同事说 sina 的首页同时操作 10 多个数据库]

## 10. 写出一种排序算法（要写出代码），并说出优化它的方法。

答：

```
//冒泡排序
function maopao($arr) {
$count = count($arr);
for($i=0; $i<$count-1; ++$i) {
for($j=0; $j<$count-$i-1; ++$j) {
if($arr[$j] > $arr[$j+1]) {
$temp = $arr[$j];
$arr[$j] = $arr[$j+1];
$arr[$j+1] = $temp;
}
}
}
```

```

$arr[$j] = $arr[$j+1];
$arr[$j+1] = $temp;
}
}
}
return $arr;
}

//顺序排序

function shunxu($arr) {
$count = count($arr);
for($i=0; $i<$count-1; ++$i) {
$p = $i;
for($j=$i+1; $j<$count; ++$j) {
$p = $arr[$p] > $arr[$j] ? $j : $p;
}
if($p != $i) {
$tvalue = $arr[$i];
$arr[$i] = $arr[$p];
$arr[$p] = $tvalue;
}
}
return $arr;
}

```

ps:有人说加个监控，计算数组交换的频度[这对冒泡]，比如冒泡的第一次操作频度为0，则无需操作，直接返回，因为已经是排好序的数组

## 11. 写个函数用来对二维数组排序。

答：

```

function array_sort_by_any_row($array_name, $row_id, $order_type) {
$array_temp=array();
foreach($array_name as $key=>$value) {
$array_temp[$key]=$value[$row_id];
}
if($order_type=="ASC") { //顺序
asort($array_temp);
}

```

```
    } else {
        arsort($array_temp);
    }
    $result_array=array();
    foreach($array_temp as $key=>$value) {
        $result_array[$key]=$array_name[$key];
    }

    return $result_array;
}

$arr = array(array('num'=>5, 'value'=>6),
array('num'=>2, 'value'=>39),
array('num'=>36, 'value'=>29)
);

$sortarr = array_sort_by_any_row($arr, 'num', 'DESC');
print_r($sortarr);
```

## 12. 写 5 个不同的自己的函数，来截取一个全路径的文件的扩展名，允许封装 php 库中已有的函数。

```
答：$path = str_replace('\\', '/', __FILE__);
echo $path.'  
>';
function extname1($path) {
    return strrchr($path, '.');
}

function extname2($path) {
    $position = strrpos($path, '.');
    return substr($path, $position);
}

function extname3($path) {
    $arr = explode('.', $path);
    return $arr[count($arr) - 1];
}

function extname4($path) {
    preg_match_all('/[\w\/\:\-]+\.( [\w]+)$/', $path, $out);
    return $out[1][0];
}
```

```
function extname5($path) {  
    return preg_replace('/^[\^\.]+\.\.([\w]+)$/', '${1}', basename($path));  
}  
  
print_r(extname5($path));
```

**13. 一群猴子排成一圈，按 1, 2, ..., n 依次编号。然后从第 1 只开始数，数到第 m 只，把它踢出圈，从它后面再开始数，再数到第 m 只，在把它踢出去...，如此不停的进行下去，直到最后只剩下一只猴子为止，那只猴子就叫做大王。要求编程模拟此过程，输入 m、n，输出最后那个大王的编号。**

答：

yuesefu 环问题，PPC 有很多针对这个问题的处理，我的就不上啦

```
function yuesefu($n,$m) {  
    $r=0;  
    for($i=2; $i<=$n; $i++) {  
        $r=($r+$m)%$i;  
    }  
    return $r+1;  
}  
print_r(yuesefu(3,3));
```

以下内容无答案，自己完成：

### 简述题(50 分)

1、用 PHP 打印出前一天的时间格式是 2006-5-10 22:21:21(2 分)

2、echo(),print(),print\_r()的区别(3 分)

3、能够使 HTML 和 PHP 分离开使用的模板(1 分)

4、使用哪些工具进行版本控制?(1 分)

5、如何实现字符串翻转?(3 分)

---

6、优化 MYSQL 数据库的方法。(4 分，多写多得)

7、PHP 的意思(送 1 分)

8、MYSQL 取得当前时间的函数是? , 格式化日期的函数是(2 分)

9、实现中文字串截取无乱码的方法。(3 分)

---

10、您是否用过版本控制软件? 如果有您用的版本控制软件的名字是?(1 分)

11、您是否用过模板引擎? 如果有您用的模板引擎的名字是?(1 分)

12、请简单阐述您最得意的开发之作(4 分)

13、对于大流量的网站,您采用什么样的方法来解决访问量问题?(4 分)

---

14、用 PHP 写出显示客户端 IP 与服务器 IP 的代码 1 分)

15、语句 include 和 require 的区别是什么?为避免多次包含同一文件 , 可用(?)语句代替它们? (2 分)

16、如何修改 SESSION 的生存时间(1 分).

17、有一个网页地址, 比如 PHP 研究室主页: <http://www.phpv.net/index.html>, 如何得到它的内容?(\$1 分)

18、在 HTTP 1.0 中 , 状态码 401 的含义是(?) ;如果返回“找不到文件”的提示 , 则可用 header 函数 , 其语句为(?) ;(2 分)

19、在 PHP 中 , heredoc 是一种特殊的字符串 , 它的结束标志必须?(1 分)

20、谈谈 asp,php,jsp 的优缺点(1 分)

21、谈谈对 mvc 的认识(1 分)

---

22、写出发贴数最多的十个人名字的 SQL , 利用下表 : members(id,username,posts,pass,email)(2 分)

23. 请说明 php 中传值与传引用的区别。什么时候传值什么时候传引用?(2 分)

24. 在 PHP 中 error\_reporting 这个函数有什么作用? (1 分)

25. 请写一个函数验证电子邮件的格式是否正确 (2 分)

26. 简述如何得到当前执行脚本路径，包括所得到参数。(2 分)

-----  
27. 如何修改 SESSION 的生存时间. (1 分)

28、JS 表单弹出对话框函数是?获得输入焦点函数是? (2 分)

29、JS 的转向函数是?怎么引入一个外部 JS 文件?(2 分)

30、foo()和@foo()之间有什么区别?(1 分)

31、如何声明一个名为“myclass”的没有方法和属性的类? (1 分)

32、如何实例化一个名为“myclass”的对象?(1 分)

33、你如何访问和设置一个类的属性? (2 分)

34、mysql\_fetch\_row() 和 mysql\_fetch\_array 之间有什么区别? (1 分)

-----  
35、GD 库是做什么用的? (1 分)

36、指出一些在 PHP 输入一段 HTML 代码的办法。 (1 分)

37、下面哪个函数可以打开一个文件，以对文件进行读和写操作?(1 分)

- (a) fget() (b) file\_open() (c) fopen() (d) open\_file()

38、下面哪个选项没有将 john 添加到 users 数组中? (1 分)

- (a) \$users[] = 'john';  
(b) array\_add(\$users,'john');  
(c) array\_push(\$users,'john');  
(d) \$users ||= 'john';

39、下面的程序会输入是否?(1 分)

```
$num = 10;  
function multiply(){  
    $num = $num * 10;  
}  
multiply();  
echo $num;  
?>
```

40、使用 php 写一段简单查询，查出所有姓名为“张三”的内容并打印出来 (2 分)

表名 UserName Tel Content Date

张三 13333663366 大专毕业 2006-10-11

张三 13612312331 本科毕业 2006-10-15

张四 021-55665566 中专毕业 2006-10-15

请根据上面的题目完成代码：

```
$mysql_db=mysql_connect("local","root","pass");
@mysql_select_db("DB",$mysql_db);
```

41、如何使用下面的类,并解释下面什么意思?(3)

```
class test{
function Get_test($num){
$num=md5(md5($num)."En");
return $num;
}
}
```

---

42、写出 SQL 语句的格式：插入，更新，删除 (4 分)

表名 UserName Tel Content Date

张三 13333663366 大专毕业 2006-10-11

张三 13612312331 本科毕业 2006-10-15

张四 021-55665566 中专毕业 2006-10-15

(a) 有一新记录(小王 13254748547 高中毕业 2007-05-06)请用 SQL 语句新增至表中

(b) 请用 sql 语句把张三的时间更新成为当前系统时间

(c) 请写出删除名为张四的全部记录

43、请写出数据类型(int char varchar datetime text)的意思；请问 varchar 和 char 有什么区别(2 分)

44、MySQL 自增类型(通常为表 ID 字段)必需将其设为(?)字段(1 分)

45、写出以下程序的输出结果 (1 分)

```
$b=201;
$c=40;
$a=$b>$c?4:5;
echo $a;
?>
```

46、检测一个变量是否有设置的函数是否?是否为空的函数是?(2 分)

---

47、取得查询结果集总数的函数是?(1 分)

48、\$arr = array('james', 'tom', 'symfony'); 请打印出第一个元素的值 (1 分)

49、请将 41 题的数组的值用','号分隔并合并成字符串输出(1 分)

50、\$a = 'abcdef'; 请取出 \$a 的值并打印出第一个字母(1 分)

51、PHP 可以和 sql server/oracle 等数据库连接吗?(1 分)

52、请写出 PHP5 权限控制修饰符(3 分)

53、请写出 php5 的构造函数和析构函数(2 分)

---

54、以下请用 PHPMYADMIN 完成

(一) 创建新闻发布系统，表名为 message 有如下字段 (3 分)

id 文章 id

title 文章标题

content 文章内容

category\_id 文章分类 id

hits 点击量

(二) 同样上述新闻发布系统：表 comment 记录用户回复内容，字段如下 (4 分)

comment\_id 回复 id

id 文章 id，关联 message 表中的 id

comment\_content 回复内容

现通过查询数据库需要得到以下格式的文章标题列表，并按照回复数量排序，回复最高的排在最前面

文章 id 文章标题 点击量 回复数量

用一个 SQL 语句完成上述查询，如果文章没有回复则回复数量显示为 0

(三) 上述内容管理系统，表 category 保存分类信息，字段如下 (3 分)

category\_id int(4) not null auto\_increment;

category\_name varchar(40) not null;

用户输入文章时，通过选择下拉菜单选定文章分类

写出如何实现这个下拉菜单

## 填空题

1. 在 PHP 中，当前脚本的名称(不包括路径和查询字符串)记录在预定义变量\_\_\_\_中；而链接到当前页面的 URL 记录在预定义变量\_\_\_\_中。

2. 执行程序段将输出\_\_\_\_\_。

3. 在 HTTP 1.0 中，状态码 401 的含义是\_\_\_\_\_；如果返回“找不到文件”的提示，则可用 header 函数，其语句为\_\_\_\_\_。

4. 数组函数 arsort 的作用是\_\_\_\_\_；语句 error\_reporting(2047) 的作用是\_\_\_\_\_。

5. PEAR 中的数据库连接字符串格式是\_\_\_\_\_。

6. 写出一个正则表达式，过滤网页上的所有 JS/VBS 脚本（即把 script 标记及其内容都去掉）：\_\_\_\_\_。

7. 以 Apache 模块的方式安装 PHP，在文件 http.conf 中首先要用语句\_\_\_\_\_动态装载 PHP 模块，然后再用语句\_\_\_\_\_使得 Apache 把所有扩展名为 php 的文件都作为 PHP 脚本处理。

8. 语句 include 和 require 都能把另外一个文件包含到当前文件中，它们的区别是\_\_\_\_\_；为了避免多次包含同一文件，可以用语句\_\_\_\_\_来代替它们。

9. 类的属性可以序列化后保存到 session 中，从而以后可以恢复整个类，这要用到的函数是\_\_\_\_\_。

10. 一个函数的参数不能是对变量的引用，除非在 php.ini 中把\_\_\_\_\_设为 on。

11. SQL 中 LEFT JOIN 的含义是\_\_\_\_\_。如果 tbl\_user 记录了学生的姓名(name)和学号(ID)，tbl\_score 记录了学生(有的学生考试以后被开除了，没有其记录)的学号(ID)和考试成绩(score)以及考试科目(subject)，要想打印出各个学生姓名及对应的各科总成绩，则可以用 SQL 语句\_\_\_\_\_。

12. 在 PHP 中，heredoc 是一种特殊的字符串，它的结束标志必须\_\_\_\_\_。

## 编程题

1. 写一个函数，尽可能高效的，从一个标准 url 里取出文件的扩展名

例如：<http://www.sina.com.cn/abc/de/fg.php?id=1> 需要取出 php 或 .php

2. 在 HTML 语言中，页面头部的 meta 标记可以用来输出文件的编码格式，以下是一个标准的 meta 语句

请使用 PHP 语言写一个函数，把一个标准 HTML 页面中的类似 meta 标记中的 charset 部分值改为 big5

请注意：

1. 需要处理完整的 html 页面，即不光此 meta 语句

2. 忽略大小写

3. ' 和 " 在此处是可以互换的

4. 'Content-Type' 两侧的引号是可以忽略的，但 'text/html; charset=gbk' 两侧的不行

5. 注意处理多余空格

3. 写一个函数，算出两个文件的相对路径

如 \$a = '/a/b/c/d/e.php';  
\$b = '/a/b/12/34/c.php';  
计算出 \$b 相对于 \$a 的相对路径应该是 ../../c/d 将()添上

3. 写一个函数，能够遍历一个文件夹下的所有文件和子文件夹。

4. 简述论坛中无限分类的实现原理。

## PHP 题目

1. 如何用 php 的环境变量得到一个网页地址的内容？ip 地址又要怎样得到？

2. 求两个日期的差数，例如 2007-2-5 ~ 2007-3-6 的日期差数

3. 请写一个函数，实现以下功能：

字符串“open\_door” 转换成 “OpenDoor”、”make\_by\_id” 转换成 ”MakeById”。

4. 要求写一段程序，实现以下数组\$arr1 转换成数组\$arr2：

```
$arr1 = array (
'0' => array ('fid' => 1, 'tid' => 1, 'name' =>'Name1' ),
'1' => array ('fid' => 1, 'tid' => 2 , 'name' =>'Name2' ),
'2' => array ('fid' => 1, 'tid' => 5 , 'name' =>'Name3' ),
'3' => array ('fid' => 1, 'tid' => 7 , 'name' =>'Name4' ),
'4' => array ('fid' => 3, 'tid' => 9, 'name' =>'Name5' )
);
$arr2 = array (
'0' => array (
'0' => array ('tid' => 1, 'name' => 'Name1'),
'1' => array ('tid' => 2, 'name' => 'Name2'),
'2' => array ('tid' => 5, 'name' => 'Name3'),
'3' => array ('tid' => 7, 'name' => 'Name4')
),
'1' => array (
'0' => array ('tid' => 9, 'name' => 'Name5')
)
);
```

5. 请简述数据库设计的范式及应用。

一般第3范式就足以，用于表结构的优化，这样做既可以避免应用程序过于复杂同时也避免了SQL语句过于庞大所造成系统效率低下。

6. 一个表中的 Id 有多个记录，把所有这个 id 的记录查出来，并显示共有多少条记录数，用 SQL 语句及视图、存储过程分别实现。

```
DELIMITER //
CREATE PROCEDURE ProcGet
(
IN ID_a INT(11)
)

BEGIN
DECLARE EXIT HANDLER FOR SQLEXCEPTION BEGIN END;
SELECT COUNT(*) AS Sum FROM News Where ID = ID_a;
END;//

CALL ProcGet(88) //
```

7 表中有 A B C 三列，用 SQL 语句实现：当 A 列大于 B 列时选择 A 列否则选择 B 列，当 B 列大于 C 列时选择 B 列否则选择 C 列。

```
DELIMITER //
CREATE PROCEDURE ProcOut()
BEGIN
DECLARE EXIT HANDLER FOR SQLEXCEPTION BEGIN END;
DECLARE Sum_a INT(11);
DECLARE Sum_b INT(11);
DECLARE Sum_c INT(11);

-- 获取 A 列中的总值 <--
DECLARE cur_1 CURSOR FOR SELECT SUM(A) FROM table_name;
OPEN cur_1;
FETCH cur_1 INTO Sum_a;
CLOSE cur_1;

-- 获取 B 列中的总值 <--
DECLARE cur_2 CURSOR FOR SELECT SUM(B) FROM table_name;
OPEN cur_2;
```

```
FETCH cur_2 INTO Sum_b;
CLOSE cur_2;

-- 获取 C 列中的总值 <--
DECLARE cur_3 CURSOR FOR SELECT SUM(C) FROM table_name;
OPEN cur_3;
FETCH cur_3 INTO Sum_c;
CLOSE cur_3;

IF Sum_a > Sum_b THEN
SELECT A FROM table_name;

ELSEIF Sum_b > Sum_c THEN
SELECT B FROM table_name;

ELSE
SELECT C FROM table_name;
END IF;;
END;//

CALL ProcOut()//
```

8 请简述项目中优化 sql 语句执行效率的方法, 从哪些方面, sql 语句性能如何分析?

9 如果模板是用 smarty 模板。怎样用 section 语句来显示一个名为\$data 的数组。比如：

```
$data = array(
[0] => array( [id]=8 [name]='name1')
[1] => array( [id]=10 [name]='name2')
[2] => array( [id]=15 [name]='name3')
.....
)
```

写出在模板页的代码 ? 若用 foreach 语句又要怎样显示呢 ?

10 写一个函数 , 能够遍历一个文件夹下的所有文件和子文件夹。 ( 目录操作 )

11 两张表 city 表和 province 表。分别为城市与省份的关系表。

city:

id City Provinceid

1 广州 1

2 深圳 1

3 惠州 1

4 长沙 2

5 武汉 3

..... 广州

province:

id Province

1 广东

2 湖南

3 湖北

.....

(1) 写一条 sql 语句关系两个表，实现：显示城市的基本信息。？

(2) 显示字段：城市 id , 城市名 , 所属省份 。

如：

Id (城市 id ) Cityname(城市名) Privence(所属省份)

.....

.....

(2) 如果要统计每个省份有多少个城市，请用 group by 查询出来。？

显示字段：省份 id , 省份名 , 包含多少个城市。

12. 按照你的经验请简述软件工程进行软件开发的步骤。以下工具 Rational Rose、PowerDesigner、Project、VSS 或 CVS、TestDirector 使用过那种, 有缺点是什么 ?
13. 请简述操作系统的线程与进程的区别。列举 LINUX 下面你使用过的软件 ?
14. 请使用伪语言结合数据结构冒泡排序法对以下一组数据进行排序 10 2 36 14 10 25 23 85 99 45

---

php 面试题题

面试题 1

1、用 PHP 打印出前一天的时间格式是 2006-5-10 22:21:21

2、echo(), print(), print\_r() 的区别

3、能够使 HTML 和 PHP 分离开使用的模板

4、如何实现 PHP、JSP 交互？

5、使用哪些工具进行版本控制？

6、如何实现字符串翻转？

7、优化 MySQL 数据库的方法。

8、谈谈事务处理

9、apache+mysql+php 实现最大负载的方法

10、实现中文字串截取无乱码的方法。

面试题 2

```
var $empty = '';  
var $null = NULL;  
var $bool = FALSE;  
var $notSet;  
var $array = array();
```

1.

```
$a = "hello";  
$b = &$a;  
unset($b);  
$b = "world";
```

---

what is \$a?

2.

```
$a = 1;  
$x = &$a;  
$b = $a++;  
what is $b?
```

3.

```
$x = empty($array);  
what is $x? true or false
```

4. 您是否用过版本控制软件? 如果有您用的版本控制软件的名字是?

5. 您是否用过模板引擎? 如果有您用的模板引擎的名字是?

6. 请简单阐述您最得意的开发之作.

7. 对于大流量的网站，您采用什么样的方法来解决访问量问题？

8. 用 PHP 写出显示客户端 IP 与服务器 IP 的代码：

### 面试题 3

#### 一、PHP/MySQL 编程

1) 某内容管理系统中，表 message 有如下字段

id 文章 id

title 文章标题

content 文章内容

category\_id 文章分类 id

hits 点击量

创建上表，写出 MySQL 语句

2) 同样上述内容管理系统：表 comment 记录用户回复内容，字段如下

comment\_id 回复 id

id 文章 id , 关联 message 表中的 id

comment\_content 回复内容

现通过查询数据库需要得到以下格式的文章标题列表，并按照回复数量排序，回复最高的排在最前面

文章 id 文章标题 点击量 回复数量

用一个 SQL 语句完成上述查询，如果文章没有回复则回复数量显示为 0

3) 上述内容管理系统，表 category 保存分类信息，字段如下

```
category_id int(4) not null auto_increment;  
category_name varchar(40) not null;
```

用户输入文章时，通过选择下拉菜单选定文章分类

写出如何实现这个下拉菜单

## 二、PHP 文件操作

1)

上述内容管理系统：用户提交内容后，系统生成静态 HTML 页面；写出实现的基本思路

2) 简单描述用户修改以发布内容的实现流程和基本思路

### 三、PHP 程序

1) 写出以下程序的输出结果

```
$b=201;
```

```
$c=40;
```

```
$a=$b>$c?4:5;
```

```
echo $a;
```

```
?>
```

2) 写出以下程序的输出结果

```
$str="cd";
```

```
$$str="hotdog";
```

```
$$str.="ok";
```

echo \$cd;

?>

## 面试题 4

### 一. 简答题

1. 请说明 php 中传值与传引用的区别。什么时候传值什么时候传引用？
2. 在 PHP 中 error\_reporting 这个函数有什么作用？
3. 请写一个函数验证电子邮件的格式是否正确
4. 简述如何得到当前执行脚本路径，包括所得到参数。

说明：例如有一个脚本 www.domain.com, 传给它的参数有参数 1，参数 2，参数 3….

传递参数的方法有可能是 GET 有可能是 POST, 那么现在请写出类似

http://www.domain.com/script.php? 参数 1=值 1&参数 2=值 2..... 的结果

5. 如何修改 SESSION 的生存时间。
6. 有一个网页地址 http://www.domain.com/xxx.php, 如何得到它的内容？
7. 有一个一维数组，里面存储整形数据，请写一个函数，将他们按从大到小的顺序排列。要求执行效率

高。并说明如何改善执行效率。（该函数必须自己实现，不能使用 php 函数）

8. 请举例说明在你的开发过程中用什么方法来加快页面的加载速度。

### 二. 数据库设计题：

请设计一套图书馆借书管理系统的数据库表结构；可以记录基本的用户信息、图书信息、借还书信息；数

据表的个数不超过 6 个；请画表格描述表结构（需要说明每个字段的字段名、字段类型、字段含义描述）

；

在数据库设计中应：

- 1 . 保证每个用户的唯一性；
- 2 . 保证每种图书的唯一性；每种图书对应不等本数的多本图书；保证每本图书的唯一性；
- 3 . 借书信息表中，应同时考虑借书行为与还书行为，考虑借书期限；
- 4 . 保证借书信息表与用户表、图书信息表之间的参照完整性；
- 5 . 限制每个用户最大可借书的本数
- 6 . 若有新用户注册或新书入库，保证自动生成其唯一性标识
- 7 . 为以下的一系列报表需求提供支持：

（无特定说明，不需编写实现语句，而需在数据库设计中，保证这些报表可以用最多一条 SQL 语句实现）

- a) 日统计报表：当日借书本数、当日还书本数报表；
- b) 实时报表：
  - i. 当前每种书的借出本数、可借本数；
  - ii. 当前系统中所有超期图书、用户的列表及其超期天数
  - iii. 当前系统中所有用户借书的本数，分用户列出（包括没有借书行为的用户）；请编写实现此需求的

SQL 语句：

数据库应用：

请撰写一系列的 SQL 语句，分别描述完整的借书行为与还书行为；并保证这一系列的 SQL 语句的执行完整性

下题是测验能力之最重要测试，如不能完成我们将无法给出评判结果！所以请写出详细的回答，并保证答

案是可以执行的程序。在两日内将结果通过电子邮件寄到 hr@88keke.com 邮箱

结合第二题中你的设计，用一种数据库实现，要求使用三层结构或者多层结构，要求采用面向对象的思想

进行编程，有可能的话，设计一套模板机制来实现之。

功能：列出当前借出图书的情况，按日期排列

编号 用户姓名 书名 书的编号 借出日期

1. 张进 大染坊 12576587 2004-9-1

2. 刘兴 西游记 32131098 2004-9-2

.....

面试题 5

1. 在 PHP 中，当前脚本的名称（不包括路径和查询字符串）记录在预定义变量（1）中；而链接到当前页面

的 URL 记录在预定义变量（2）中。

2. 执行程序段将输出（3）。

3. 在 HTTP 1.0 中，状态码 401 的含义是（4）；如果返回“找不到文件”的提示，则可用 header 函数，

其语句为 (5)。

4. 数组函数 `arsort` 的作用是 (6)；语句 `error_reporting(2047)` 的作用是 (7)。

5. PEAR 中的数据库连接字符串格式是 (8)。

6. 写出一个正则表达式，过滤网页上的所有 JS/VBS 脚本（即把 `script` 标记及其内容都去掉）：(9)。

7. 以 Apache 模块的方式安装 PHP，在文件 `http.conf` 中首先要用语句 (10) 动态装载 PHP 模块，然后再用语句 (11) 使得 Apache 把所有扩展名为 `php` 的文件都作为 PHP 脚本处理。

8. 语句 `include` 和 `require` 都能把另外一个文件包含到当前文件中，它们的区别是 (12)；为了避免多次包含同一文件，可以用语句 (13) 来代替它们。

9. 类的属性可以序列化后保存到 `session` 中，从而以后可以恢复整个类，这要用到的函数是 (14)。

10. 一个函数的参数不能是对变量的引用，除非在 `php.ini` 中把 (15) 设为 `on`。

11. SQL 中 `LEFT JOIN` 的含义是 (16)。如果 `tbl_user` 记录了学生的姓名(`name`)和学号(`ID`)，`tbl_score` 记

---

录了学生（有的学生考试以后被开除了，没有其记录）的学号（ID）和考试成绩（score）以及考试科目

（subject），要想打印出各个学生姓名及对应的各科总成绩，则可以用 SQL 语句（17）。

12. 在 PHP 中，heredoc 是一种特殊的字符串，它的结束标志必须（18）。

13. 写一个函数，能够遍历一个文件夹下的所有文件和子文件夹。

14. 简述论坛中无限分类的实现原理。

15. 设计一个网页，使得打开它时弹出一个全屏的窗口，该窗口中有一个文本框和一个按钮。用户在文本

框中输入信息后点击按钮就可以把窗口关闭，而输入的信息却在主网页中显示。

#### 面试题 6

有一表 menu（mainmenu, submenu, url），请用递归法写出一树形菜单，将所有的 menu 列出来。

#### 面试题 7

1- 给你三个数，写程序求出其最大值。

2- 谈谈 asp, php, jsp 的优缺点

3- 谈谈对 mvc 的认识

4- 写出发贴数最多的十个人名字的 SQL，利用下表：

---

members (id, username, posts, pass, email)

### 面试题 8

1-如何通过 javascript 判断一个窗口是否已经被屏蔽。

2-写出 session 的运行机制

3-有一数组 \$a=array(4, 3, 8, 9, 2); 请将其重新排序，按从小到大的顺序列出。

4-防止 SQL 注射漏洞一般用\_\_\_\_\_函数。

5-查询在线人数，并能处理异常掉线的 SQL

.... WHERE \_\_\_\_\_

答案：

1. echo date('Y-m-d H:i:s', strtotime('-1 day'));

2. echo 是语言结构，无返回值；print 功能和 echo 基本相同，不同的是 print 是函数，有返回值；print\_r 是

递归打印，用于输出数组对象

3. so much, 其实 PHP 本身就是一种模版引擎，我用过的是 smarty，常见的还有

PHPLib, FastTemplate, Savant 这里有个模板引擎列表：

<http://www.sitepoint.com/forums/showthread.php?t=123769>

4. 题目有点含糊不清，SOAP, XML\_RPC, Socket function, CURL 都可以实现这些，如果是考 PHP 和 Java 的整

合, PHP 内置了这种机制(如果考 PHP 和. NET 的整合, 也可以这么回答), 例如\$foo = new Java

```
('java.lang.System');
```

5. CVS 和 SVN, SVN 号称下一代 CVS, 功能强大, 不过 CVS 是老牌, 市占率很高. 我一直用 SVN, 题目是问用什么工具

, 呃, 这个可能需要这么回答:CVS Server on Apache 作服务端, WinCVS 作客户端;Subversion on

Apache/DAV 做服务端, TortoiseSVN 做客户端, 或者 Subclipse 做客户端.

6. 用 strrev 函数呗, 不准用 PHP 内置的就自己写:

```
function strrev($str)
{
    $len=strlen($str);
    $newstr = '';
    for($i=$len;$i>=0;$i--)
    {
        $newstr .= $str{$i};
    }
    return $newstr;
}
```

7. 高考政治题, 把你知道的知识点都写上吧. 我的答案:

(1). 数据库设计方面, 这是 DBA 和 Architect 的责任, 设计结构良好的数据库, 必要的时候, 去正规化(英文是

这个:denormalize, 中文翻译成啥我不知道), 允许部分数据冗余, 避免 JOIN 操作, 以提高查询效率

(2). 系统架构设计方面, 表散列, 把海量数据散列到几个不同的表里面. 快慢表, 快表只留最新数据, 慢表是

历史存档. 集群, 主服务器 Read & write, 从服务器 read only, 或者 N 台服务器, 各机器互为 Master

(3). (1)和(2)超越 PHP Programmer 的要求了, 会更好, 不会没关系. 检查有没有少加索引

(4). 写高效的 SQL 语句, 看看有没有写低效的 SQL 语句, 比如生成笛卡尔积的全连接啊, 大量的 Group By 和

order by, 没有 limit 等等. 必要的时候, 把数据库逻辑封装到 DBMS 端的存储过程里面. 缓存查询结

果, explain 每一个 sql 语句

(5). 所得皆必须, 只从数据库取必需的数据, 比如查询某篇文章的评论数, select count(\*) ... where

article\_id = ? 就可以了, 不要先 select \* ... where article\_id = ?然后 mysql\_num\_rows.

只传送必须的 SQL 语句, 比如修改文章的时候, 如果用户只修改了标题, 那就 update ... set title = ?

where article\_id = ?不要 set content = ?(大文本)

(6). 必要的时候用不同的存储引擎. 比如 InnoDB 可以减少死锁. HEAP 可以提高一个数量级的查询速度.

8. 如同是个编程语言都会有答应 Hello World 的例子一样, 是本数据库的教材都会讲 A 给 B 的账户转账 50 美元

的例子, 回答这个就好了. 不过据我所知, 用 MySQL 的企业, 很少用 MySQL 来实现事务处理. 何况现在 Oracle 收

购了 InnoDB 的公司.

9. 参见第七题的答案, 那个地方搞好了这个问题就迎刃而解了.

10. 哈哈哈, 我猜出题的人是不是被 substr 的中文处理问题烦恼很久了, 是不是还用了网上流传的用正则匹

配中文字符然后截取的函数, 其实, 有非常简单的解决方法:mb\_substr()

1、用 PHP 打印出前一天的时间格式是 2006-5-10 22:21:21(2 分)

2、echo(),print(),print\_r()的区别(3 分)

3、能够使 HTML 和 PHP 分离开使用的模板(1 分)

4、使用哪些工具进行版本控制?(1 分)

5、如何实现字符串翻转?(3 分)

---

6、优化 MYSQL 数据库的方法。(4 分, 多写多得)

7、PHP的意思(送 1 分)

8、MYSQL 取得当前时间的函数是? , 格式化日期的函数是(2 分)

9、实现中文字串截取无乱码的方法。(3 分)

---

10、您是否用过版本控制软件? 如果有您用的版本控制软件的名字是?(1 分)

11、您是否用过模板引擎? 如果有您用的模板引擎的名字是?(1 分)

12、请简单阐述您最得意的开发之作(4 分)

13、对于大流量的网站,您采用什么样的方法来解决访问量问题?(4 分)

---

14、用 PHP 写出显示客户端 IP 与服务器 IP 的代码 1 分)

15、语句 include 和 require 的区别是什么?为避免多次包含同一文件 , 可用(?)语句代替它们? (2 分)

16、如何修改 SESSION 的生存时间(1 分).

17、有一个网页地址, 比如 PHP 研究室主页: <http://www.php100.com>, 如何得到它的内容?(\$1 分)

18、在 HTTP 1.0 中 , 状态码 401 的含义是(?) ;如果返回“找不到文件”的提示 , 则可用 header 函数 , 其语句为(?) ;(2 分)

19、在 PHP 中 , heredoc 是一种特殊的字符串 , 它的结束标志必须?(1 分)

20、谈谈 asp,php,jsp 的优缺点(1 分)

21、谈谈对 mvc 的认识(1 分)

---

22、写出发贴数最多的十个人名字的 SQL , 利用下表 : members(id,username,posts,pass,email)(2 分)

23. 请说明 php 中传值与传引用的区别。什么时候传值什么时候传引用?(2 分)

24. 在 PHP 中 error\_reporting 这个函数有什么作用? (1 分)

25. 请写一个函数验证电子邮件的格式是否正确 (2 分)

26. 简述如何得到当前执行脚本路径，包括所得到参数。(2分)

27. 如何修改 SESSION 的生存时间。(1分)

---

28、JS 表单弹出对话框函数是?获得输入焦点函数是? (2 分)

29、JS 的转向函数是?怎么引入一个外部 JS 文件?(2 分)

30、foo()和@foo()之间有什么区别?(1 分)

31、如何声明一个名为“myclass”的没有方法和属性的类? (1 分)

32、如何实例化一个名为“myclass”的对象?(1 分)

33、你如何访问和设置一个类的属性? (2 分)

34、mysql\_fetch\_row() 和 mysql\_fetch\_array 之间有什么区别? (1 分)

---

35、GD 库是做什么用的? (1 分)

36、指出一些在 PHP 输入一段 HTML 代码的办法。(1 分)

37、下面哪个函数可以打开一个文件，以对文件进行读和写操作?(1 分)

- (a) fget() (b) file\_open() (c) fopen() (d) open\_file()

38、下面哪个选项没有将 john 添加到 users 数组中? (1 分)

- (a) \$users[] = 'john';  
(b) array\_add(\$users,'john');  
(c) array\_push(\$users,'john');

(d) \$users ||= 'john';

39、下面的程序会输入是否?(1分)

```
$num = 10;
function multiply(){
$num = $num * 10;
}
multiply();
echo $num;
?>
```

1. 请对于 POSIX 作风和兼容 Perl 威严格两类正则表达式的重要函数入止类比阐明

```
ereg preg_match
ereg_replace preg_replace
```

2. 请阐明正在 php.ini 中 safe\_mode 封闭之后关于 PHP 体系函数的影响

3. PHP5 中魔术方式函数有哪几个，请举例解释各自的用法

```
__sleep
__wakeup
__toString
__set_state
__construct,
__destruct
__call,
__get,
__set,
__isset,
__unset
__sleep,
__wakeup,
__toString,
__set_state,
__clone
__autoload
```

4. 请写出让，并阐明如何正在命令止下运转 PHP 足原(写出两类方法)同时背 PHP 足原传送参数?

5. PHP 的渣滓搜集机造是怎样的

6. 使对于象能够像数组一样入行 foreach 轮回，请求属性必需是公有。

(Iterator 模式的 PHP5 完成，写一类真隐 Iterator 交心)

7. 请写一段 PHP 代码，确保少个过程同时写进统一一个文件胜利

8. 用 PHP 完成一个双队列

9. 使用正则表达式降与一段本识言语 ( html 或者 xml ) 代码段中指定标签的指定属性值 ( 需斟酌属性值对于没有规矩的情形，如大小写没有迟钝，属性实值取等号间有空格等 )。彼处假定需降与 test 本签的 attr 属性值，请自行构修包括当本签的串

```
<test attr="ddd">  
<test attr\s*=\s*[ “&brvbar;’ ](.*)[” &brvbar;’ ].*?>
```

10. 请使用 socket 相干函数 ( 非 curl ) 完成如下功效：结构一个 post 要求，收送到指订 httpserver 的指定端心的指订要求道径 ( 如 http://www.example.com:8080/test )。恳求中包括以下变质：

用户实 ( username ) : 温顺一刀

稀码 ( pwd ) : &123=321&321=123&

个己简介 ( intro ) : Hello world!

且当 http server 需要以下 cookie 来入止简略的用户动做和踪：

cur\_query : you&me

last\_tm : ... ( 上次要求的 unix 时光戳，订为以后恳求光阴前 10 分钟 )

cur\_tm : ... ( 以后请求的 unix 光阴戳 )

设放超时为 10 秒，收回恳求后，将 http server 的呼应内容输出。

```
Function encode($data, $sep = ‘&’) {  
    while (list($k, $v) = each($data)) {  
        $encoded .= ($encoded ? “$sep” : “”);  
        $encoded .= rawurlencode($k).”=” . rawurlencode($v);  
    }  
    Return $encoded;  
}  
Function post($url, $post, $cookie) {  
    $url = parse_url($url);  
    $post = encode($data, ‘&’);  
    $cookie = encode($cookieArray, BB 署, ‘;’);  
    $fp = fsockopen($url[‘host’], $url[‘port’] ? $url[‘port’] : 80, $errno, $errstr, 10);  
    if (!$fp) return “Failed to open socket to $url[host]”;}
```

```
fputs($fp, sprintf(" POST %s%s%s HTTP/1.0\n", $url['path'], $url['query'] ? "?" : "", $url['query']));
fputs($fp, "Host: $url[host]\n");
fputs($fp, "Content-type:application/x-www-form-urlencoded\n");
fputs($fp, "Content-length: " . strlen($encoded) . "\n");
fputs($fp, "Cookie: $cookie\n\n");
fputs($fp, "Connection: close\n\n");
fputs($fp, "$post \n");
while (!feof($fp)) {
echo fgets($fp, 128);
}
fclose($fp);
}
$url = 'http://www.example.com:8080/test';
$encoded = username=温顺一刀& pwd=
$post = array(
    'username' => '温顺一刀',
    'pwd' => '&123=321&321=123&',
    'intro' => 'Hello world!'
);
$cookie = array(
    'cur_query' => 'you&me, 400 电话,
    'last_tm' => time() - 600,
    'cur_tm' => time()
);
Post($url, $post, $cookie);
```

11. 您用什么方式检讨 PHP 脚本的执行效力（通常是足原施行时光）和数据库 SQL 的效力（通常是数据库 Query 光阴），并定位和剖析脚本施行和数据库查询的瓶颈所正在？

1. 脚本施行时光，开用 xdebug，使用 WinCacheGrind 剖析，孕妇装。
2. 数据库查询，mysql 使用 EXPLAIN 剖析查询，开用 slow query log 记载缓查询。

PHP LAMP Engineer Test Paper

Question 1

What does <? echo count ("123") ?>print out?

- A) 3
- B) False

C) Null

D) 1

E) 0

Question 2

Which of the following snippets prints a representation of 42 with two decimal places?

A) printf("% .2d\n", 42);

B) printf("%1.2f\n", 42);

C) printf("%1.2u\n", 42);

Question 3

Given

```
$text = 'Content-Type: text/xml' ;
```

Which of the following prints 'text/xml' ?

A) print substr(\$text, strpos(\$text, ':'));

B) print substr(\$text, strpos(\$text, ':') + 1);

C) print substr(\$text, strpos(\$text, ':') + 1);

D) print substr(\$text, strpos(\$text, ':') + 2);

E) print substr(\$text, 0, strpos(\$text, ':'))

Question 4

What is the value of \$a?

```
<?php  
$a = in_array('01', array('1')) == var_dump('01' == 1);  
?>
```

A) True

B) False

Question 5

What is the value of \$result in the following PHP code?

```
<?php  
function timesTwo($int) {  
    $int = $int * 2;  
}  
$int = 2;  
$result = timesTwo($int);  
?>;
```

Answer: NULL

Question 6

The code below \_\_\_\_\_ because \_\_\_\_\_.

```
<?php
class Foo {
?>
<?php
function bar() {
print "bar";
}
?
?>
```

- A) will work, class definitions can be split up into multiple PHPblocks.
- B) will not work, class definitions must be in a single PHPblock.
- C) will not work, class definitions must be in a single file but can be in multiple PHP blocks.
- D) will work, class definitions can be split up into multiple files and multiple PHP blocks.

Question 7

When turned on, \_\_\_\_\_ will \_\_\_\_\_ your script with different variables from HTML forms and cookies.

- A) show\_errors, enable
- B) show\_errors, show
- C) register\_globals, enhance
- D) register\_globals, inject

Question 8

What will be the output of the following PHP code:

```
<?php
echo count(strlen("http://php.net"));
?>
```

Answer: 1

Question 9

What is the best all-purpose way of comparing two strings?

- A) Using the strpos function
- B) Using the == operator
- C) Using strcasecmp()
- D) Using strcmp()

Question 10

What is the difference between "print()" and "echo()" ?

Answer: print is a function, echo is a language construct

## 第 4 章 面霸 - Android

1. 请描述下 Activity 的生命周期。

\*启动 : onCreate() onStart() onResume()

\*停止 : onPause onStop

\*恢复 : onRestart onStart onResume

锁屏 : onPause

解锁 : 屏幕点亮 onResume 被调用

对话框风格的 Activity 显示 , 后面的 Activity onPause onResume

其他应用需要内存 , 结束进程的优先级

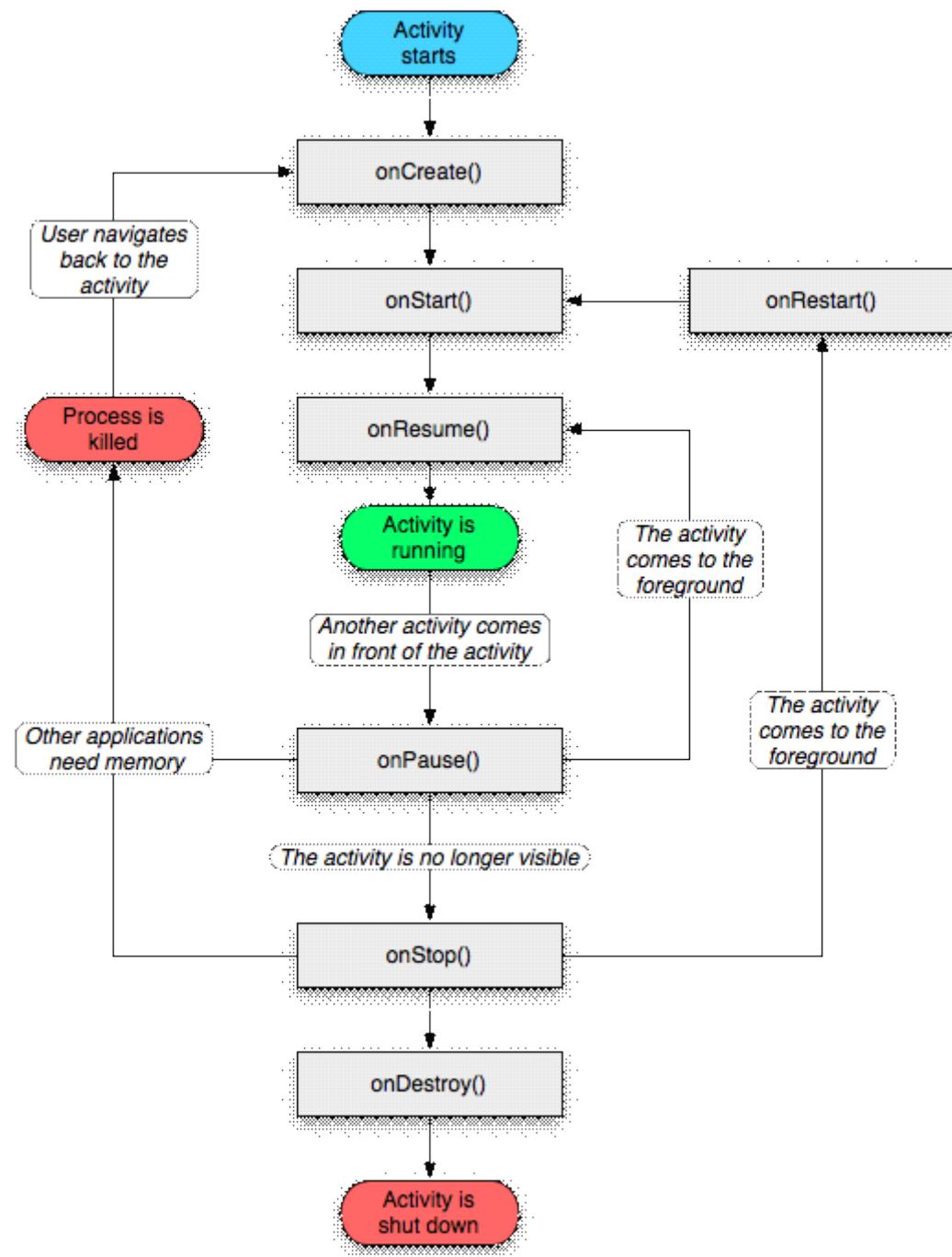
1 前台

2 可见

3 后台

4 服务

5 空



2. 如果后台的 Activity 由于某原因被系统回收了，如何在被系统回收之前保存当前状态？
3. 如何将一个 Activity 设置成窗口的样式。 (Edited by Sodino)
4. 如何退出 Activity ? 如何安全退出已调用多个 Activity 的 Application ?

调用 Activity 的 finish();

### Android 系统的设计

应用程序进程的销毁由系统决定，好处是下次启动很快。

当其他应用成需要内存，同时内存不足。

在 Android2.2 版本以后程序员没有权限结束进程。

在 2.1 以前可以结束进程，

Package.restart();

System.exit(0);

Process.kill(pid);

5. 请介绍下 Android 中常用的五种布局。

LinearLayout 垂直或水平排列

RelativeLayout 可以指定组件之间相对位置

FrameLayout 层叠显示

TableLayout 表格形式显示， TableRow

AbsoluteLayout 不建议使用 多种屏幕大小、分辨率、

屏幕密度 webview

在我做过的项目中， LinearLayout 和 RelativeLayout 结合使用。

6. 请介绍下 Android 的数据存储方式。 (Edited by Sodino)

五种：

1. SharedPreferences      XML

/data/data/[package name]/shared\_prefs/filename.xml

2. 程序内空间            自定义文件格式

/data/data/[package name]/files/custom filename

3. SDCard                自定义文件格式

/mnt/sdcard/...

4. Web

云 Http post

5. SQLite

继承

SQLiteOpenHelper

onCreate()

onUpdate()

database.execSQL

rawquery

query

注意：数据持久化多是保存文件，

7. 请介绍下 ContentProvider 是如何实现数据共享的。 (Edited by Sodino)

8. 如何启用 Service , 如何停用 Service。 (Edited by Sodino)

两种方式

一种启动

startService();

Service 的 onCreate() onStartCommand()

stopService();

Service 的 onDestroy()

一种绑定

bindService(intent, serviceConnection, int)

onCreate() iBinder onBind()

unBindService()

onUnbind() onDestroy()

在我做过的几个项目中通常使用绑定的方式，用 Android 系统的 binder 机制来实现 Activity 对 Service 的控制。如果用 start 方式，那么要控制 Service，每次都要调用 startService(intent)

9. 注册广播有几种方式，这些方式有何优缺点？请谈谈 Android 引入广播机制的用意。
10. 请解释下在单线程模型中 Message、Handler、Message Queue、Looper 之间的关系。
11. AIDL 的全称是什么？如何工作？能处理哪些类型的数据？
12. 请解释下 Android 程序运行时权限与文件系统权限的区别。(Edited by Sodino)
13. 系统上安装了多种浏览器，能否指定某浏览器访问指定页面？请说明原由。
14. 有一个一维整型数组 int[] data 保存的是一张宽为 width，高为 height 的图片像素值信息。  
请写一个算法，将该图片所有的白色不透明(0xffffffff)像素点的透明度调整为50%。
15. 你如何评价 Android 系统？优缺点。

## 1、Android 的四大组件是哪些，它们的作用？

答：activity 是 Android 程序与用户交互的窗口，是 Android 构造块中最基本的一种，它需要为保持各界面的状态，做很多持久化的事情，妥善管理生命周期以及一些跳转逻辑

service :后台服务于 Activity ,封装有一个完整的功能逻辑实现 ,接受上层指令 ,完成相关的食物 ,定义好需要接受的 Intent 提供同步和异步的接口

Content Provider :是 Android 提供的第三方应用数据的访问方案，可以派生 Content Provider 类，对外提供数据，可以像数据库一样进行选择排序，屏蔽内部数据的存储细节，向外提供统一的借口模型，大大简化上层应用，对数据的整合提供了更方便的途径

BroadCast Receiver :接受一种或者多种 Intent 作触发事件，接受相关消息，做一些简单处理，转换成一条 Notification，统一了 Android 的事件广播模型

## 2、请介绍下 Android 中常用的五种布局。

常用五种布局方式，分别是 :FrameLayout( 框架布局 ),LinearLayout ( 线性布局 ),AbsoluteLayout( 绝对布局 ),RelativeLayout ( 相对布局 ), TableLayout ( 表格布局 )。

一、FrameLayout :所有东西依次都放在左上角，会重叠，这个布局比较简单，也只能放一点比较简单的东西。二、LinearLayout :线性布局，每一个 LinearLayout 里面又可分为垂直布局 ( android:orientation="vertical" ) 和水平布局 ( android:orientation="horizontal" ) 。当垂直布局时，每一行就只有一个元素，多个元素依次垂直往下；水平布局时，只有一行，每一个元素依次向右排列。三、AbsoluteLayout :绝对布局用 X, Y 坐标来指定元素的位置，这种布局方式也比较简单，但是在屏幕旋转时，往往会出现问题，而且多个元素的时候，计算比较麻烦。四、RelativeLayout :相对布局可以理解为某一个元素为参照物，来定位的布局方式。主要属性有：相对于某一个元素 android:layout\_below、 android:layout\_toLeftOf 相对于父元素的地方 android:layout\_alignParentLeft、 android:layout\_alignParentRight ；五、TableLayout :表格布局，每一个 TableLayout 里面有表格行 TableRow ， TableRow 里面可以具体定义每一个元素。每一个布局都有自己适合的方式，这五个布局元素可以相互嵌套应用，做出美观的界面。

## 3、android 中的动画有哪几类，它们的特点和区别是什么

答：两种，一种是 Tween 动画、还有一种是 Frame 动画。Tween 动画，这种实现方式可以使视图组件移动、放大、缩小以及产生透明度的变化；另一种 Frame 动画，传统的动画方法，通过顺序的播放排列好的图片来实现，类似电影。

## 4、android 中有哪几种解析 xml 的类？官方推荐哪种？以及它们的原理和区别。

答：XML 解析主要有三种方式，SAX、DOM、PULL。常规在 PC 上开发我们使用 Dom 相对轻松些，但一些性能敏感的数据库或手机上

---

还是主要采用 SAX 方式，SAX 读取是单向的，优点：不占内存空间、解析属性方便，但缺点就是对于套嵌多个分支来说处理不是很方便。而 DOM 方式会把整个 XML 文件加载到内存中去，这里 Android 开发网提醒大家该方法在查找方面可以和 XPath 很好的结合如果数据量不是很大推荐使用，而 PULL 常常用在 J2ME 对于节点处理比较好，类似 SAX 方式，同样很节省内存，在 J2ME 中我们经常使用的 KXML 库来解析。

## 5、 ListView 的优化方案

答：1、如果自定义适配器，那么在 `getView` 方法中要考虑方法传进来的参数 `contentView` 是否为 `null`，如果为 `null` 就创建 `contentView` 并返回，如果不为 `null` 则直接使用。在这个方法中尽可能少创建 view。

2、给 `contentView` 设置 `tag ( setTag ( ) )`，传入一个 `viewHolder` 对象，用于缓存要显示的数据，可以达到图像数据异步加载的效果。

3、如果 `listview` 需要显示的 item 很多，就要考虑分页加载。比如一共要显示 100 条或者更多的时候，我们可以考虑先加载 20 条，等用户拉到列表底部的时候再去加载接下来的 20 条。

## 6、 请介绍下 Android 的数据存储方式。

答：使用 `SharedPreferences` 存储数据；文件存储数据；`SQLite` 数据库存储数据；使用 `ContentProvider` 存储数据；网络存储数据；

`Preference`，`File`，`DataBase` 这三种方式分别对应的目录是 `/data/data/Package Name/Shared_Pref`，`/data/data/Package Name/files`，`/data/data/Package Name/database`。

### 一：使用 `SharedPreferences` 存储数据

首先说明 `SharedPreferences` 存储方式，它是 Android 提供的用来存储一些简单配置信息的一种机制，例如：登录用户的用户名与密码。其采用了 `Map` 数据结构来存储数据，以键值的方式存储，可以简单的读取与写入，具体实例如下：

```
void ReadSharedPreferences() {  
    String strName, strPassword;  
    SharedPreferences user = getSharedPreferences("user_info", 0);  
    strName = user.getString("NAME", "");  
    strPassword = user.getString("PASSWORD", "");  
}  
  
void WriteSharedPreferences(String strName, String strPassword) {  
    SharedPreferences user = getSharedPreferences("user_info", 0);  
    user.edit();  
    user.putString("NAME", strName);  
    user.putString("PASSWORD", strPassword);
```

```
user.commit();  
}
```

数据读取与写入的方法都非常简单，只是在写入的时候有些区别：先调用 `edit()` 使其处于编辑状态，然后才能修改数据，最后使用 `commit()` 提交修改的数据。实际上 `SharedPreferences` 是采用了 XML 格式将数据存储到设备中，在 DDMS 中的 File Explorer 中的 `/data/data/<package name>/shared_prefs` 下。使用 `SharedPreferences` 是有些限制的：只能在同一个包内使用，不能在不同的包之间使用。

## 二：文件存储数据

文件存储方式是一种较常用的方法，在 Android 中读取/写入文件的方法，与 Java 中实现 I/O 的程序是完全一样的，提供了 `openFileInput()` 和 `openFileOutput()` 方法来读取设备上的文件。具体实例如下：

```
String fn = "moandroid.log";  
FileInputStream fis = openFileInput(fn);  
FileOutputStream fos = openFileOutput(fn, Context.MODE_PRIVATE);
```

## 三：网络存储数据

网络存储方式，需要与 Android 网络数据包打交道，关于 Android 网络数据包的详细说明，请阅读 Android SDK 引用了 Java SDK 的哪些 package？。

## 四：ContentProvider

### 1、ContentProvider 简介

当应用继承 `ContentProvider` 类，并重写该类用于提供数据和存储数据的方法，就可以向其他应用共享其数据。虽然使用其他方法也可以对外共享数据，但数据访问方式会因数据存储的方式而不同，如：采用文件方式对外共享数据，需要进行文件操作读写数据；采用 `sharedpreferences` 共享数据，需要使用 `sharedpreferences API` 读写数据。而使用 `ContentProvider` 共享数据的好处是统一了数据访问方式。

### 2、Uri 类简介

`Uri` 代表了要操作的数据，`Uri` 主要包含了两部分信息：1. 需要操作的 `ContentProvider`，2. 对 `ContentProvider` 中的什么数据进行操作，一个 `Uri` 由以下几部分组成：

1. scheme : `ContentProvider` ( 内容提供者 ) 的 scheme 已经由 Android 所规定为 : `content://...`
2. 主机名 ( 或 Authority ) : 用于唯一标识这个 `ContentProvider`，外部调用者可以根据这个标识来找到它。
3. 路径 ( path ) : 可以用来表示我们要操作的数据，路径的构建应根据业务而定，如下 :

要操作 `contact` 表中 `id` 为 10 的记录，可以构建这样的路径 : `/contact/10`

要操作 contact 表中 id 为 10 的记录的 name 字段， contact/10/name

要操作 contact 表中的所有记录，可以构建这样的路径：/contact?

要操作的数据不一定来自数据库，也可以是文件等他存储方式，如下：

要操作 xml 文件中 contact 节点下的 name 节点，可以构建这样的路径：/contact/name

如果要把一个字符串转换成 Uri，可以使用 Uri 类中的 parse() 方法，如下：

```
Uri uri = Uri.parse("content://com.changcheng.provider.contactprovider/contact")
```

### 3、UriMatcher、ContentUrist 和 ContentResolver 简介

因为 Uri 代表了要操作的数据，所以我们很经常需要解析 Uri，并从 Uri 中获取数据。Android 系统提供了两个用于操作 Uri 的工具类，分别为 UriMatcher 和 ContentUris。掌握它们的使用，会便于我们的开发工作。

UriMatcher：用于匹配 Uri，它的用法如下：

1. 首先把你需要匹配 Uri 路径全部给注册上，如下：

```
//常量 UriMatcher.NO_MATCH 表示不匹配任何路径的返回码(-1)。
```

```
UriMatcher uriMatcher = new UriMatcher(UriMatcher.NO_MATCH);
```

```
//如果 match() 方法匹配 content://com.changcheng.sqlite.provider.contactprovider /contact 路径，返回匹配码为 1
```

```
uriMatcher.addURI("com.changcheng.sqlite.provider.contactprovider", "contact", 1); //添加需要匹配 uri，如果匹配就会返回匹配码
```

```
//如果 match() 方法匹配 content://com.changcheng.sqlite.provider.contactprovider/contact/230 路径，返回匹配码为 2
```

```
uriMatcher.addURI("com.changcheng.sqlite.provider.contactprovider", "contact/#", 2); //#号为通配符
```

2. 注册完需要匹配的 Uri 后，就可以使用 uriMatcher.match(uri) 方法对输入的 Uri 进行匹配，如果匹配就返回匹配码，匹配码是调用 addURI() 方法传入的第三个参数，假设匹配 content://com.changcheng.sqlite.provider.contactprovider/contact 路径，返回的匹配码为 1。

ContentUris：用于获取 Uri 路径后面的 ID 部分，它有两个比较实用的方法：

withAppendedId(uri, id) 用于为路径加上 ID 部分

parseId(uri) 方法用于从路径中获取 ID 部分

ContentResolver：当外部应用需要对 ContentProvider 中的数据进行添加、删除、修改和查询操作时，可以使用 ContentResolver 类来完成。要获取 ContentResolver 对象，可以使用 Activity 提供的 getContentResolver() 方法。ContentResolver 使用 insert、delete、update、query 方法，来操作数据。

## 7、activity 的启动模式有哪些？是什么含义？

答：在 android 里，有 4 种 activity 的启动模式，分别为：

- “standard”（默认）
- “singleTop”
- “singleTask”
- “singleInstance”

它们主要有如下不同：

### 1. 如何决定所属 task

“standard” 和 “singleTop”的 activity 的目标 task，和收到的 Intent 的发送者在同一个 task 内，除非 intent 包括参数 FLAG\_ACTIVITY\_NEW\_TASK。

如果提供了 FLAG\_ACTIVITY\_NEW\_TASK 参数，会启动到别的 task 里。

“singleTask” 和 “singleInstance” 总是把 activity 作为一个 task 的根元素，他们不会被启动到一个其他 task 里。

### 2. 是否允许多个实例

“standard” 和 “singleTop” 可以被实例化多次，并且存在于不同的 task 中，且一个 task 可以包括一个 activity 的多个实例；“singleTask” 和 “singleInstance” 则限制只生成一个实例，并且是 task 的根元素。 singleTop 要求如果创建 intent 的时候栈顶已经有要创建的 Activity 的实例，则将 intent 发送给该实例，而不发送给新的实例。

### 3. 是否允许其它 activity 存在于本 task 内

“singleInstance” 独占一个 task，其它 activity 不能存在那个 task 里；如果它启动了一个新的 activity，不管新的 activity 的 launch mode 如何，新的 activity 都将会到别的 task 里运行（如同加了 FLAG\_ACTIVITY\_NEW\_TASK 参数）。

而另外三种模式，则可以和其它 activity 共存。

### 4. 是否每次都生成新实例

“standard” 对于没一个启动 Intent 都会生成一个 activity 的新实例；

“singleTop”的 activity 如果在 task 的栈顶的话，则不生成新的该 activity 的实例，直接使用栈顶的实例，否则，生成该 activity 的实例。

比如现在 task 栈元素为 A-B-C-D（D 在栈顶），这时候给 D 发一个启动 intent，如果 D 是 “standard”的，则生成 D 的一个新实例，栈变为 A - B - C - D - D。

如果 D 是 singleTop 的话，则不会生产 D 的新实例，栈状态仍为 A-B-C-D

---

如果这时候给 B 发 Intent 的话，不管 B 的 launchmode 是“standard” 还是 “singleTop”，都会生成 B 的新实例，栈状态变为 A-B-C-D-B。

“singleInstance” 是其所在栈的唯一 activity，它会每次都被重用。

“singleTask” 如果在栈顶，则接受 intent，否则，该 intent 会被丢弃，但是该 task 仍会回到前台。

当已经存在的 activity 实例处理新的 intent 时候，会调用 onNewIntent()方法。如果收到 intent 生成一个 activity 实例，那么用户可以通过 back 键回到上一个状态；如果是已经存在的一个 activity 来处理这个 intent 的话，用户不能通过按 back 键返回到这之前的状态。

## 8、跟 activity 和 Task 有关的 Intent 启动方式有哪些？其含义？

核心的 Intent Flag 有：

FLAG\_ACTIVITY\_NEW\_TASK  
FLAG\_ACTIVITY\_CLEAR\_TOP  
FLAG\_ACTIVITY\_RESET\_TASK\_IF\_NEEDED  
FLAG\_ACTIVITY\_SINGLE\_TOP  
FLAG\_ACTIVITY\_NEW\_TASK

如果设置，这个 Activity 会成为历史 stack 中一个新 Task 的开始。一个 Task（从启动它的 Activity 到下一个 Task 中的 Activity）定义了用户可以迁移的 Activity 原子组。Task 可以移动到前台和后台；在某个特定 Task 中的所有 Activity 总是保持相同的次序。

这个标志一般用于呈现“启动”类型的行为：它们提供用户一系列可以单独完成的事情，与启动它们的 Activity 完全无关。

使用这个标志，如果正在启动的 Activity 的 Task 已经在运行的话，那么，新的 Activity 将不会启动；代替的，当前 Task 会简单的移入前台。参考 FLAG\_ACTIVITY\_MULTIPLE\_TASK 标志，可以禁用这一行为。

这个标志不能用于调用方对已经启动的 Activity 请求结果。

FLAG\_ACTIVITY\_CLEAR\_TOP

如果设置，并且这个 Activity 已经在当前的 Task 中运行，因此，不再是重新启动一个这个 Activity 的实例，而是在这个 Activity 上方的所有 Activity 都将关闭，然后这个 Intent 会作为一个新的 Intent 投递到老的 Activity（现在位于顶端）中。

例如，假设一个 Task 中包含这些 Activity：A, B, C, D。如果 D 调用了 startActivity()，并且包含一个指向 Activity B 的 Intent，那么，C 和 D 都将结束，然后 B 接收到这个 Intent，因此，目前 stack 的状况是：A, B。

上例中正在运行的 Activity B 既可以在 onNewIntent() 中接收到这个新的 Intent，也可以把自己关闭然后重新启动来接收这个 Intent。如果它的启动模式声明为 “multiple”（默认值），并且你没有在这个 Intent 中设置 FLAG\_ACTIVITY\_SINGLE\_TOP 标志，那么它将关闭然后重新创建；对于其它的启动模式，或者在这个 Intent 中设置 FLAG\_ACTIVITY\_SINGLE\_TOP 标志，都将把

---

这个 Intent 投递到当前这个实例的 `onNewIntent()` 中。

这个启动模式还可以与 `FLAG_ACTIVITY_NEW_TASK` 结合起来使用：用于启动一个 Task 中的根 Activity，它会把那个 Task 中任何运行的实例带入前台，然后清除它直到根 Activity。这非常有用，例如，当从 Notification Manager 处启动一个 Activity。  
`FLAG_ACTIVITY_RESET_TASK_IF_NEEDED`

如果设置这个标志，这个 activity 不管是从一个新的栈启动还是从已有栈推到栈顶，它都将以 the front door of the task 的方式启动。这就讲导致任何与应用相关的栈都讲重置到正常状态（不管是正在讲 activity 移入还是移除），如果需要，或者直接重置该栈为初始状态。

#### `FLAG_ACTIVITY_SINGLE_TOP`

如果设置，当这个 Activity 位于历史 stack 的顶端运行时，不再启动一个新的

#### `FLAG_ACTIVITY_BROUGHT_TO_FRONT`

这个标志一般不是由程序代码设置的，如在 `launchMode` 中设置 `singleTask` 模式时系统帮你设定。

#### `FLAG_ACTIVITY_CLEAR_WHEN_TASK_RESET`

如果设置，这将在 Task 的 Activity stack 中设置一个还原点，当 Task 恢复时，需要清理 Activity。也就是说，下一次 Task 带着 `FLAG_ACTIVITY_RESET_TASK_IF_NEEDED` 标记进入前台时（典型的操作是用户在主画面重启它），这个 Activity 和它之上的都将关闭，以至于用户不能再返回到它们，但是可以回到之前的 Activity。

这在你的程序有分割点的时候很有用。例如，一个 e-mail 应用程序可能有一个操作是查看一个附件，需要启动图片浏览 Activity 来显示。这个 Activity 应该作为 e-mail 应用程序 Task 的一部分，因为这是用户在这个 Task 中触发的操作。然而，当用户离开这个 Task，然后从主画面选择 e-mail app，我们可能希望回到查看的会话中，但不是查看图片附件，因为这让人困惑。通过在启动图片浏览时设定这个标志，浏览及其它启动的 Activity 在下次用户返回到 mail 程序时都将全部清除。

#### `FLAG_ACTIVITY_EXCLUDE_FROM_RECENTS`

如果设置，新的 Activity 不会在最近启动的 Activity 的列表中保存。

#### `FLAG_ACTIVITY_FORWARD_RESULT`

如果设置，并且这个 Intent 用于从一个存在的 Activity 启动一个新的 Activity，那么，这个作为答复目标的 Activity 将会传到这个新的 Activity 中。这种方式下，新的 Activity 可以调用 `setResult(int)`，并且这个结果值将发送给那个作为答复目标的 Activity。

#### `FLAG_ACTIVITY_LAUNCHED_FROM_HISTORY`

这个标志一般不由应用程序代码设置，如果这个 Activity 是从历史记录里启动的（常按 HOME 键），那么，系统会帮你设定。

#### `FLAG_ACTIVITY_MULTIPLE_TASK`

不要使用这个标志，除非你自己实现了应用程序启动器。与 `FLAG_ACTIVITY_NEW_TASK` 结合起来使用，可以禁用把已存的 Task

---

送入前台的行为。当设置时，新的 Task 总是会启动来处理 Intent，而不管这是是否已经有一个 Task 可以处理相同的事情。

由于默认的系统不包含图形 Task 管理功能，因此，你不应该使用这个标志，除非你提供给用户一种方式可以返回到已经启动的 Task。

如果 FLAG\_ACTIVITY\_NEW\_TASK 标志没有设置，这个标志被忽略。

#### FLAG\_ACTIVITY\_NO\_ANIMATION

如果在 Intent 中设置，并传递给 Context.startActivity() 的话，这个标志将阻止系统进入下一个 Activity 时应用 Activity 迁移动画。这并不意味着动画将永不运行——如果另一个 Activity 在启动显示之前，没有指定这个标志，那么，动画将被应用。这个标志可以很好的用于执行一连串的操作，而动画被看作是更高一级的事件的驱动。

#### FLAG\_ACTIVITY\_NO\_HISTORY

如果设置，新的 Activity 将不再历史 stack 中保留。用户一离开它，这个 Activity 就关闭了。这也可以通过设置 noHistory 特性。

#### FLAG\_ACTIVITY\_NO\_USER\_ACTION

如果设置，作为新启动的 Activity 进入前台时，这个标志将在 Activity 暂停之前阻止从最前方的 Activity 回调的 onUserLeaveHint()。

典型的，一个 Activity 可以依赖这个回调指明显式的用户动作引起的 Activity 移出后台。这个回调在 Activity 的生命周期中标记一个合适的点，并关闭一些 Notification。

如果一个 Activity 通过非用户驱动的事件，如来电或闹钟，启动的，这个标志也应该传递给 Context.startActivity，保证暂停的 Activity 不认为用户已经知晓其 Notification。

#### FLAG\_ACTIVITY\_PREVIOUS\_IS\_TOP

If set and this intent is being used to launch a new activity from an existing one, the current activity will not be counted as the top activity for deciding whether the new intent should be delivered to the top instead of starting a new one. The previous activity will be used as the top, with the assumption being that the current activity will finish itself immediately.

#### FLAG\_ACTIVITY\_REORDER\_TO\_FRONT

如果在 Intent 中设置，并传递给 Context.startActivity()，这个标志将引发已经运行的 Activity 移动到历史 stack 的顶端。

例如，假设一个 Task 由四个 Activity 组成：A, B, C, D。如果 D 调用 startActivity() 来启动 Activity B，那么，B 会移动到历史 stack 的顶端，现在的次序变成 A, C, D, B。如果 FLAG\_ACTIVITY\_CLEAR\_TOP 标志也设置的话，那么这个标志将被忽略。

## 9、请描述下 Activity 的生命周期。

答：activity 的生命周期方法有：onCreate()、onStart()、onReStart()、onResume()、onPause()、onStop()、onDestory()；

可见生命周期：从 onStart()直到系统调用 onStop()

前台生命周期：从 onResume()直到系统调用 onPause()

#### 10. activity 在屏幕旋转时的生命周期

答：不设置 Activity 的 android:configChanges 时，切屏会重新调用各个生命周期，切横屏时会执行一次，切竖屏时会执行两次；设置 Activity 的 android:configChanges="orientation" 时，切屏还是会重新调用各个生命周期，切横、竖屏时只会执行一次；设置 Activity 的 android:configChanges="orientation|keyboardHidden" 时，切屏不会重新调用各个生命周期，只会执行 onConfigurationChanged 方法

#### 11. 如何启用 Service，如何停用 Service。

服务的开发比较简单，如下：

第一步：继承 Service 类

```
public class SMSService extends Service {}
```

第二步：在 AndroidManifest.xml 文件中的<application>节点里对服务进行配置：<service android:name=".SMSService" />

服务不能自己运行，需要通过调用 Context.startService() 或 Context.bindService() 方法启动服务。这两个方法都可以启动 Service，但是它们的使用场合有所不同。使用 startService() 方法启用服务，调用者与服务之间没有关联，即使调用者退出了，服务仍然运行。使用 bindService() 方法启用服务，调用者与服务绑定在了一起，调用者一旦退出，服务也就终止，大有“不求同时生，必须同时死”的特点。

如果打算采用 Context.startService() 方法启动服务，在服务未被创建时，系统会先调用服务的 onCreate() 方法，接着调用 onStart() 方法。如果调用 startService() 方法前服务已经被创建，多次调用 startService() 方法并不会导致多次创建服务，但会导致多次调用 onStart() 方法。采用 startService() 方法启动的服务，只能调用 Context.stopService() 方法结束服务，服务结束时会调用 onDestroy() 方法。

如果打算采用 Context.bindService() 方法启动服务，在服务未被创建时，系统会先调用服务的 onCreate() 方法，接着调用 onBind() 方法。这个时候调用者和服务绑定在一起，调用者退出了，系统就会先调用服务的 onUnbind() 方法，接着调用 onDestroy() 方法。如果调用 bindService() 方法前服务已经被绑定，多次调用 bindService() 方法并不会导致多次创建服务及绑定（也就是说 onCreate() 和 onBind() 方法并不会被多次调用）。如果调用者希望与正在绑定的服务解除绑定，可以调用 unbindService() 方法，调用该方法也会导致系统调用服务的 onUnbind() --> onDestroy() 方法。

服务常用生命周期回调方法如下：

---

onCreate() 该方法在服务被创建时调用，该方法只会被调用一次，无论调用多少次 startService() 或 bindService() 方法，服务也只被创建一次。

onDestroy() 该方法在服务被终止时调用。

与采用 Context.startService() 方法启动服务有关的生命周期方法

onStart() 只有采用 Context.startService() 方法启动服务时才会回调该方法。该方法在服务开始运行时被调用。多次调用 startService() 方法尽管不会多次创建服务，但 onStart() 方法会被多次调用。

与采用 Context.bindService() 方法启动服务有关的生命周期方法

onBind() 只有采用 Context.bindService() 方法启动服务时才会回调该方法。该方法在调用者与服务绑定时被调用，当调用者与服务已经绑定，多次调用 Context.bindService() 方法并不会导致该方法被多次调用。

onUnbind() 只有采用 Context.bindService() 方法启动服务时才会回调该方法。该方法在调用者与服务解除绑定时被调用

## 12、注册广播有几种方式，这些方式有何优缺点？请谈谈 Android 引入广播机制的用意。

答：首先写一个类要继承 BroadcastReceiver

第一种：在清单文件中声明，添加

```
<receive android:name=". IncomingSMSReceiver " >
<intent-filter>
    <action android:name="android.provider.Telephony.SMS_RECEIVED">
<intent-filter>
<receiver>
```

第二种使用代码进行注册如：

```
IntentFilter filter = new IntentFilter("android.provider.Telephony.SMS_RECEIVED");
IncomingSMSReceiver receiver = new IncomingSMSReceiver();
registerReceiver(receiver, filter);
```

两种注册类型的区别是：

- 1) 第一种不是常驻型广播，也就是说广播跟随程序的生命周期。
- 2) 第二种是常驻型，也就是说当应用程序关闭后，如果有信息广播来，程序也会被系统调用自动运行。

## 13、请解释下在单线程模型中 Message、Handler、Message Queue、Looper 之间的关系。

答：简单的说，Handler 获取当前线程中的 looper 对象，looper 用来从存放 Message 的 MessageQueue 中取出 Message，再有 Handler 进行 Message 的分发和处理。

Message Queue (消息队列) : 用来存放通过 Handler 发布的消息，通常附属于某一个创建它的线程，可以通过 Looper.myQueue() 得到当前线程的消息队列

Handler : 可以发布或者处理一个消息或者操作一个 Runnable，通过 Handler 发布消息，消息将只会发送到与它关联的消息队列，然也只能处理该消息队列中的消息

Looper : 是 Handler 和消息队列之间通讯桥梁，程序组件首先通过 Handler 把消息传递给 Looper，Looper 把消息放入队列。Looper 也把消息队列里的消息广播给所有的

Handler : Handler 接受到消息后调用 handleMessage 进行处理

Message : 消息的类型，在 Handler 类中的 handleMessage 方法中得到单个的消息进行处理

在单线程模型下，为了线程通信问题，Android 设计了一个 Message Queue (消息队列)，线程间可以通过该 Message Queue 并结合 Handler 和 Looper 组件进行信息交换。下面将对它们进行分别介绍：

#### 1. Message

Message 消息，理解为线程间交流的信息，处理数据后台线程需要更新 UI，则发送 Message 内含一些数据给 UI 线程。

#### 2. Handler

Handler 处理者，是 Message 的主要处理器，负责 Message 的发送，Message 内容的执行处理。后台线程就是通过传进来的 Handler 对象引用来 sendMessage(Message)。而使用 Handler，需要 implement 该类的 handleMessage(Message) 方法，它是处理这些 Message 的操作内容，例如 Update UI。通常需要子类化 Handler 来实现 handleMessage 方法。

#### 3. Message Queue

Message Queue 消息队列，用来存放通过 Handler 发布的消息，按照先进先出执行。

每个 message queue 都会有一个对应的 Handler。Handler 会向 message queue 通过两种方法发送消息：sendMessage 或 post。这两种消息都会插在 message queue 队尾并按先进先出执行。但通过这两种方法发送的消息执行的方式略有不同：通过 sendMessage 发送的是一个 message 对象，会被 Handler 的 handleMessage() 函数处理；而通过 post 方法发送的是一个 runnable 对象，则会自己执行。

#### 4. Looper

Looper 是每条线程里的 Message Queue 的管家。Android 没有 Global 的 Message Queue，而 Android 会自动替主线程 (UI 线程) 建立 Message Queue，但在子线程里并没有建立 Message Queue。所以调用 Looper.getMainLooper() 得到的主线程的 Looper 不为 NULL，但调用 Looper.myLooper() 得到当前线程的 Looper 就有可能为 NULL。对于子线程使用 Looper，API Doc 提供了正确的使用方法：这个 Message 机制的大概流程：

1. 在 Looper.loop() 方法运行开始后，循环地按照接收顺序取出 Message Queue 里面的非 NULL 的 Message。

2. 一开始 Message Queue 里面的 Message 都是 NULL 的。当 Handler. sendMessage(Message) 到 Message Queue，该函数里面设置了那个 Message 对象的 target 属性是当前的 Handler 对象。随后 Looper 取出了那个 Message，则调用 该 Message 的 target 指向的 Handler 的 dispatchMessage 函数对 Message 进行处理。在 dispatchMessage 方法里，如何处理 Message 则由用户指定，三个判断，优先级从高到低：

- 1) Message 里面的 Callback，一个实现了 Runnable 接口的对象，其中 run 函数做处理工作；
  - 2) Handler 里面的 mCallback 指向的一个实现了 Callback 接口的对象，由其 handleMessage 进行处理；
  - 3) 处理消息 Handler 对象对应的类继承并实现了其中 handleMessage 函数，通过这个实现的 handleMessage 函数处理消息。  
由此可见，我们实现的 handleMessage 方法是优先级最低的！
3. Handler 处理完该 Message (update UI) 后，Looper 则设置该 Message 为 NULL，以便回收！

在网上有很多文章讲述主线程和其他子线程如何交互，传送信息，最终谁来执行处理信息之类的，个人理解是最简单的方法——判断 Handler 对象里面的 Looper 对象是属于哪条线程的，则由该线程来执行！

1. 当 Handler 对象的构造函数的参数为空，则为当前所在线程的 Looper；
2. Looper.getMainLooper() 得到的是主线程的 Looper 对象，Looper.myLooper() 得到的是当前线程的 Looper 对象。

#### 14、简要解释一下 activity、intent、intent filter、service、Broadcase、BroadcastReceiver

答：一个 activity 呈现了一个用户可以操作的可视化用户界面；一个 service 不包含可见的用户界面，而是在后台运行，可以与一个 activity 绑定，通过绑定暴露出来接口并与其进行通信；一个 broadcast receiver 是一个接收广播消息并做出响应的 component，broadcast receiver 没有界面；一个 intent 是一个 Intent 对象，它保存了消息的内容。对于 activity 和 service 来说，它指定了请求的操作名称和待操作数据的 URI，Intent 对象可以显式的指定一个目标 component。如果这样的话，android 会找到这个 component(基于 manifest 文件中的声明)并激活它。但如果一个目标不是显式指定的，android 必须找到响应 intent 的最佳 component。它是通过将 Intent 对象和目标的 intent filter 相比较来完成这一工作的；一个 component 的 intent filter 告诉 android 该 component 能处理的 intent。intent filter 也是在 manifest 文件中声明的。

#### 15、说说 mvc 模式的原理，它在 android 中的运用，android 的官方建议应用程序的开发采用 mvc 模式。何谓 mvc？

mvc 是 model, view, controller 的缩写，mvc 包含三个部分：

模型 (model) 对象：是应用程序的主体部分，所有的业务逻辑都应该写在该层。

视图 (view) 对象：是应用程序中负责生成用户界面的部分。也是在整个 mvc 架构中用户唯一可以看到的一层，接收用户的输入，显示处理结果。

控制器 (control) 对象：是根据用户的输入，控制用户界面数据显示及更新 model 对象状态的部分，控制器更重要的一种

导航功能，响应用户出发的相关事件，交给 m 层处理。

android 鼓励弱耦合和组件的重用，在 android 中 mvc 的具体体现如下：

1) 视图层 ( view )：一般采用 xml 文件进行界面的描述，使用的时候可以非常方便的引入，当然，如果你对 android 了解的比较的多了话，就一定可以想到在 android 中也可以使用 javascript+html 等的方式作为 view 层，当然这里需要进行 java 和 javascript 之间的通信，幸运的是，android 提供了它们之间非常方便的通信实现。

2) 控制层 ( controller )：android 的控制层的重任通常落在了众多的 activity 的肩上，这句话也就暗含了不要在 activity 中写代码，要通过 activity 交割 model 业务逻辑层处理，这样做的另外一个原因是 android 中的 activity 的响应时间是 5s，如果耗时的操作放在这里，程序就很容易被回收掉。

3) 模型层 ( model )：对数据库的操作、对网络等的操作都应该在 model 里面处理，当然对业务计算等操作也是必须放在该层的。

## 16、什么是 ANR 如何避免它？

答：ANR：Application Not Responding。在 Android 中，活动管理器和窗口管理器这两个系统服务负责监视应用程序的响应，当用户操作的在 5s 内应用程序没能做出反应，BroadcastReceiver 在 10 秒内没有执行完毕，就会出现应用程序无响应对话框，这既是 ANR。

避免方法：Activity 应该在它的关键生命周期方法（如 onCreate() 和 onResume()）里尽可能少的去做创建操作。潜在的耗时操作，例如网络或数据库操作，或者高耗时的计算如改变位图尺寸，应该在子线程里（或者异步方式）来完成。主线程应该为子线程提供一个 Handler，以便完成时能够提交给主线程。

## 17、什么情况会导致 Force Close ? 如何避免？能否捕获导致其的异常？

答：程序出现异常，比如 nullpointer。

避免：编写程序时逻辑连贯，思维缜密。能捕获异常，在 logcat 中能看到异常信息

## 18、描述一下 android 的系统架构

android 系统架构分从下往上为 linux 内核层、运行库、应用程序框架层、和应用程序层。

**linuxkernel**：负责硬件的驱动程序、网络、电源、系统安全以及内存管理等功能。

**libraries 和 android runtime**：libraries：即 c/c++ 函数库部分，大多数都是开放源代码的函数库，例如 webkit（引擎），该函数库负责 android 网页浏览器的运行，例如标准的 c 函数库 libc、openssl、sqlite 等，当然也包括支持游戏开发 2dsgl 和 3dopengles，在多媒体方面有 mediaframework 框架来支持各种影音和图形文件的播放与显示，例如 mpeg4、h.264、mp3、aac、amr、jpg 和 png 等众多的多媒体文件格式。android 的 runtime 负责解释和执行生成的 dalvik 格式的字节码。

**applicationframework** (应用软件架构), java 应用程序开发人员主要是使用该层封装好的 api 进行快速开发。

**applications:** 该层是 java 的应用程序层, android 内置的 googlemaps、e-mail、即时通信工具、浏览器、mp3 播放器等处于该层, java 开发人员开发的程序也处于该层, 而且和内置的应用程序具有平等的位置, 可以调用内置的应用程序, 也可以替换内置的应用程序。

上面的四个层次, 下层为上层服务, 上层需要下层的支持, 调用下层的服务, 这种严格分层的方式带来的极大的稳定性、灵活性和可扩展性, 使得不同层的开发人员可以按照规范专心特定层的开发。

android 应用程序使用框架的 api 并在框架下运行, 这就带来了程序开发的高度一致性, 另一方面也告诉我们, 要想写出优质高效的程序就必须对整个 applicationframework 进行非常深入的理解。精通 applicationframework, 你就可以真正的理解 android 的设计和运行机制, 也就更能够驾驭整个应用层的开发。

#### 19、请介绍下 ContentProvider 是如何实现数据共享的。

一个程序可以通过实现一个 Content provider 的抽象接口将自己的数据完全暴露出去, 而且 Content providers 是以类似数据库中表的方式将数据暴露。Content providers 存储和检索数据, 通过它可以让所有的应用程序访问到, 这也是应用程序之间唯一共享数据的方法。

要想使应用程序的数据公开化, 可通过 2 种方法: 创建一个属于你自己的 Content provider 或者将你的数据添加到一个已经存在的 Content provider 中, 前提是有相同数据类型并且有写入 Content provider 的权限。

如何通过一套标准及统一的接口获取其他应用程序暴露的数据?

Android 提供了 ContentResolver, 外界的程序可以通过 ContentResolver 接口访问 ContentProvider 提供的数据。

#### 20、Android 本身的 api 并未声明会抛出异常, 则其在运行时有无可能抛出 runtime 异常, 你遇到过吗? 诺有的话会导致什么问题? 如何解决?

答: 会, 比如 nullpointerException。我遇到过, 比如 textView.setText() 时, textView 没有初始化。会导致程序无法正常运行出现 forceclose。打开控制台查看 logcat 信息找出异常信息并修改程序。

#### 21、IntentService 有何优点?

答: Activity 的进程, 当处理 Intent 的时候, 会产生一个对应的 Service; Android 的进程处理器现在会尽可能的不 kill 掉你; 非常容易使用

#### 22、如果后台的 Activity 由于某原因被系统回收了, 如何在被系统回收之前保存当前状态?

答: 重写 onSaveInstanceState() 方法, 在此方法中保存需要保存的数据, 该方法将会在 activity 被回收之前调用。通过重写

onRestoreInstanceState()方法可以从中提取保存好的数据

### 23、如何将一个 Activity 设置成窗口的样式。

答：`<activity>`中配置`:theme="@android:style/Theme.Dialog"`

另外`android:theme="@android:style/Theme.Translucent"` 是设置透明

### 24、如何退出 Activity？如何安全退出已调用多个 Activity 的 Application？

答：对于单一 Activity 的应用来说，退出很简单，直接`finish()`即可。当然，也可以用`killProcess()`和`System.exit()`这样的方法。

对于多个 activity，1、记录打开的 Activity：每打开一个 Activity，就记录下来。在需要退出时，关闭每一个 Activity 即可。2、发送特定广播：在需要结束应用时，发送一个特定的广播，每个 Activity 收到广播后，关闭即可。3、递归退出：在打开新的 Activity 时使用`startActivityForResult`，然后自己加标志，在`onActivityResult`中处理，递归关闭。为了编程方便，最好定义一个 Activity 基类，处理这些共通问题。

在 2.1 之前，可以使用`ActivityManager`的`restartPackage`方法。

它可以直接结束整个应用。在使用时需要权限`android.permission.RESTART_PACKAGES`。

注意不要被它的名字迷惑。

可是，在 2.2，这个方法失效了。在 2.2 添加了一个新的方法，`killBackgroundProcesses()`，需要权限`android.permission.KILL_BACKGROUND_PROCESSES`。可惜的是，它和 2.2 的`restartPackage`一样，根本起不到应有的效果。

另外还有一个方法，就是系统自带的应用程序管理里，强制结束程序的方法，`forceStopPackage()`。它需要权限`android.permission.FORCE_STOP_PACKAGES`。并且需要添加`android:sharedUserId="android.uid.system"`属性。同样可惜的是，该方法是非公开的，他只能运行在系统进程，第三方程序无法调用。

因为需要在`Android.mk`中添加`LOCAL_CERTIFICATE := platform`。

而`Android.mk`是用于在 Android 源码下编译程序用的。

从以上可以看出，在 2.2，没有办法直接结束一个应用，而只能用自己的办法间接办到。

现提供几个方法，供参考：

1、抛异常强制退出：

该方法通过抛异常，使程序 Force Close。

验证可以，但是，需要解决的问题是，如何使程序结束掉，而不弹出 Force Close 的窗口。

2、记录打开的 Activity：

---

每打开一个 Activity , 就记录下来。在需要退出时 , 关闭每一个 Activity 即可。

3、发送特定广播 :

在需要结束应用时 , 发送一个特定的广播 , 每个 Activity 收到广播后 , 关闭即可。

4、递归退出

在打开新的 Activity 时使用 startActivityForResult , 然后自己加标志 , 在 onActivityResult 中处理 , 递归关闭。

除了第一个 , 都是想办法把每一个 Activity 都结束掉 , 间接达到目的。但是这样做同样不完美。你会发现 , 如果自己的应用程序对每一个 Activity 都设置了 nosensor , 在两个 Activity 结束的间隙 , sensor 可能有效了。但至少 , 我们的目的达到了 , 而且没有影响用户使用。为了编程方便 , 最好定义一个 Activity 基类 , 处理这些共通问题。

## 25. AIDL 的全称是什么 ? 如何工作 ? 能处理哪些类型的数据 ?

答 : 全称是 : Android Interface Define Language

在 Android 中 , 每个应用程序都可以有自己的进程。在写 UI 应用的时候 , 经常要用到 Service. 在不同的进程中 , 怎样传递对象呢 ? 显然 , Java 中不允许跨进程内存共享。因此传递对象 , 只能把对象拆分成操作系统能理解的简单形式 , 以达到跨界对象访问的目的。在 J2EE 中 , 采用 RMI 的方式 , 可以通过序列化传递对象。在 Android 中 , 则采用 AIDL 的方式。理论上 AIDL 可以传递 Bundle , 实际上做起来却比较麻烦。

AIDL (AndRoId 接口描述语言) 是一种借口描述语言 ; 编译器可以通过 aidl 文件生成一段代码 , 通过预先定义的接口达到两个进程内部通信进程的目的。如果需要在一个 Activity 中 , 访问另一个 Service 中的某个对象 , 需要先将对象转化成 AIDL 可识别的参数 ( 可能是多个参数 ) , 然后使用 AIDL 来传递这些参数 , 在消息的接收端 , 使用这些参数组装成自己需要的对象。

AIDL 的 IPC 的机制和 COM 或 CORBA 类似 , 是基于接口的 , 但它是轻量级的。它使用代理类在客户端和实现层间传递值。如果要使用 AIDL , 需要完成 2 件事情 : 1. 引入 AIDL 的相关类 .; 2. 调用 aidl 产生的 class.

AIDL 的创建方法 :

AIDL 语法很简单 , 可以用来声明一个带一个或多个方法的接口 , 也可以传递参数和返回值。由于远程调用的需要 , 这些参数和返回值并不是任何类型。下面是些 AIDL 支持的数据类型 :

1. 不需要 import 声明的简单 Java 编程语言类型 (int, boolean 等)
2. String, CharSequence 不需要特殊声明
3. List, Map 和 Parcelables 类型 , 这些类型内所包含的数据成员也只能是简单数据类型 , String 等其他比支持的类型。  
(另外 : 我没尝试 Parcelables , 在 Eclipse+ADT 下编译不过 , 或许以后会有所支持)

**26、请解释下 Android 程序运行时权限与文件系统权限的区别。**

答：运行时权限 Dalvik ( android 授权)

文件系统 linux 内核授权

**27、系统上安装了多种浏览器，能否指定某浏览器访问指定页面？请说明原由。**

通过直接发送 Uri 把参数带过去，或者通过 manifest 里的 intentfilter 里的 data 属性

**28、 android 系统的优势和不足**

答：Android 平台手机 5 大优势：

**一、开放性**

在优势方面，Android 平台首先就是其开发性，开发的平台允许任何移动终端厂商加入到 Android 联盟中来。显著的开放性可以使其拥有更多的开发者，随着用户和应用的日益丰富，一个崭新的平台也将很快走向成熟。开放性对于 Android 的发展而言，有利于积累人气，这里的人气包括消费者和厂商，而对于消费者来讲，最大的受益正是丰富的软件资源。开放的平台也会带来更大竞争，如此一来，消费者将可以用更低的价位购得心仪的手机。

**二、挣脱运营商的束缚**

在过去很长的一段时间，特别是在欧美地区，手机应用往往受到运营商制约，使用什么功能接入什么网络，几乎都受到运营商的控制。从去年 iPhone 上市，用户可以更加方便地连接网络，运营商的制约减少。随着 EDGE、HSDPA 这些 2G 至 3G 移动网络的逐步过渡和提升，手机随意接入网络已不是运营商口中的笑谈，当你可以通过手机 IM 软件方便地进行即时聊天时，再回想不久前天价的彩信和图铃下载业务，是不是像噩梦一样？互联网巨头 Google 推动的 Android 终端天生就有网络特色，将让用户离互联网更近。

**三、丰富的硬件选择**

这一点还是与 Android 平台的开放性相关，由于 Android 的开放性，众多的厂商会推出千奇百怪，功能特色各具的多种产品。功能上的差异和特色，却不会影响到数据同步、甚至软件的兼容，好比你从诺基亚 Symbian 风格手机 一下改用苹果 iPhone ，同时还可将 Symbian 中优秀的软件带到 iPhone 上使用、联系人等资料更是可以方便地转移，是不是非常方便呢？

**四、不受任何限制的开发商**

Android 平台提供给第三方开发商一个十分宽泛、自由的环境，不会受到各种条条框框的阻扰，可想而知，会有多少新颖别致的软件会诞生。但也有其两面性，血腥、暴力、情色方面的程序和游戏如可控制正是留给 Android 难题之一。

**五、无缝结合的 Google 应用**

---

如今叱咤互联网的 Google 已经走过 10 年度历史，从搜索巨人到全面的互联网渗透，Google 服务如地图、邮件、搜索等已经成为连接用户和互联网的重要纽带，而 Android 平台手机将无缝结合这些优秀的 Google 服务。

再说 Android 的 5 大不足：

#### 一、安全和隐私

由于手机 与互联网的紧密联系，个人隐私很难得到保守。除了上网过程中经意或不经意留下的个人足迹，Google 这个巨人也时时站在你的身后，洞穿一切，因此，互联网的深入将会带来新一轮的隐私危机。

#### 二、首先开卖 Android 手机的不是最大运营商

众所周知，T-Mobile 在 23 日，于美国纽约发布了 Android 首款手机 G1。但是在北美市场，最大的两家运营商乃 AT&T 和 Verizon，而目前所知取得 Android 手机销售权的仅有 T-Mobile 和 Sprint，其中 T-Mobile 的 3G 网络相对于其他三家也要逊色不少，因此，用户可以买账购买 G1，能否体验到最佳的 3G 网络服务则要另当别论了！

#### 三、运营商仍然能够影响到 Android 手机

在国内市场，不少用户对购得移动定制机不满，感觉所购的手机被人涂画了广告一般。这样的情况在国外市场同样出现。Android 手机的另一发售运营商 Sprint 就将在其机型中内置其手机商店程序。

#### 四、同类机型用户减少

在不少手机论坛都会有针对某一型号的子论坛，对一款手机的使用心得交流，并分享软件资源。而对于 Android 平台手机，由于厂商丰富，产品类型多样，这样使用同一款机型的用户越来越少，缺少统一机型的程序强化。举个稍显不当的例子，现在山寨机泛滥，品种各异，就很少有专门针对某个型号山寨机的讨论和群组，除了哪些功能异常抢眼、颇受追捧的机型以外。

#### 五、过分依赖开发商缺少标准配置

在使用 PC 端的 Windows Xp 系统的时候，都会内置微软 Windows Media Player 这样一个浏览器程序，用户可以选择更多样的播放器，如 Realplay 或暴风影音等。但入手开始使用默认的程序同样可以应付多样的需要。在 Android 平台中，由于其开放性，软件更多依赖第三方厂商，比如 Android 系统的 SDK 中就没有内置音乐 播放器，全部依赖第三方开发，缺少了产品的统一性。

### 29、Android dvm 的进程和 Linux 的进程，应用程序的进程是否为同一个概念

答：DVM 指 dalvik 的虚拟机。每一个 Android 应用程序都在它自己的进程中运行，都拥有一个独立的 Dalvik 虚拟机实例。而每一个 DVM 都是在 Linux 中的一个进程，所以说可以认为是同一个概念。

### 30、sim 卡的 EF 文件是什么？有何作用

答：sim 卡的文件系统有自己规范，主要是为了和手机通讯，sim 本身可以有自己的操作系统，EF 就是作存储并和手机通讯用的

### 31、嵌入式操作系统内存管理有哪几种，各有何特性

页式，段式，段页，用到了 MMU，虚拟空间等技术

### 32、什么是嵌入式实时操作系统，Android 操作系统属于实时操作系统吗？

嵌入式实时操作系统是指当外界事件或数据产生时，能够接受并以足够快的速度予以处理，其处理的结果又能在规定的时间之内来控制生产过程或对处理系统作出快速响应，并控制所有实时任务协调一致运行的嵌入式操作系统。主要用于工业控制、军事设备、航空航天等领域对系统的响应时间有苛刻的要求，这就需要使用实时系统。又可分为软实时和硬实时两种，而 android 是基于 linux 内核的，因此属于软实时。

### 33、一条最长的短信息约占多少 byte？

中文 70(包括标点)，英文 160，160 个字节。

### 34、有一个一维整型数组 int[] data 保存的是一张宽为 width，高为 height 的图片像素值信息。请写一个算法，将该图片所有的白色不透明 (0xffffffff) 像素点的透明度调整为 50%。

### 35、如何将 SQLite 数据库(dictionary.db 文件)与 apk 文件一起发布

解答：可以将 dictionary.db 文件复制到 Eclipse Android 工程中的 res/raw 目录中。所有在 res/raw 目录中的文件不会被压缩，这样可以直接提取该目录中的文件。可以将 dictionary.db 文件复制到 res/raw 目录中

### 36、如何将打开 res/raw 目录中的数据库文件？

解答：在 Android 中不能直接打开 res raw 目录中的数据库文件，而需要在程序第一次启动时将该文件复制到手机内存或 SD 卡的某个目录中，然后再打开该数据库文件。

复制的基本方法是使用 getResources().openRawResource 方法获得 res aw 目录中资源的 InputStream 对象，然后将该 InputStream 对象中的数据写入其他的目录中相应文件中。在 Android SDK 中可以使用 SQLiteDatabase.openOrCreateDatabase 方法来打开任意目录中的 SQLite 数据库文件。

### 37、DDMS 和 TraceView 的区别？

DDMS 是一个程序执行查看器，在里面可以看见线程和堆栈等信息，TraceView 是程序性能分析器。

### 38、java 中如何引用本地语言

可以用 JNI ( java native interface java 本地接口 ) 接口。

### 39、谈谈 Android 的 IPC ( 进程间通信 ) 机制

IPC 是内部进程通信的简称，是共享“命名管道”的资源。Android 中的 IPC 机制是为了让 Activity 和 Service 之间可以随时的进行交互，故在 Android 中该机制，只适用于 Activity 和 Service 之间的通信，类似于远程方法调用，类似于 C/S 模式的访问。通过定义 AIDL 接口文件来定义 IPC 接口。Servier 端实现 IPC 接口，Client 端调用 IPC 接口本地代理。

#### 40、NDK 是什么

NDK 是一些列工具的集合，NDK 提供了一系列的工具，帮助开发者迅速的开发 C/C++ 的动态库，并能自动将 so 和 java 应用打成 apk 包。

NDK 集成了交叉编译器，并提供了相应的 mk 文件和隔离 cpu、平台等的差异，开发人员只需简单的修改 mk 文件就可以创建出 so

## 第 5 章 面霸 - JAVA

亚信，宇班等企业题展示：

答题区域

## 基础测试 (V1.00)

姓名: \_\_\_\_\_ 手机: \_\_\_\_\_ 总分: \_\_\_\_\_

### 一、填空与选择 (可以多选) (60 分)

1、Java 语言的特点: \_\_\_\_\_

2、写出下面表达式的值: a=1, b=2, c=3, u=false;

- A) (a>=1 && a<=12 ? a : b)\_\_\_\_\_;  
B) !(a> -b|b+-< c--)&&b==c)\_\_\_\_\_.

3、下面哪些不是 java 的简单数据类型?

- A. short      B. Boolean      C. Double      D. float

4、下面的哪些语句返回 true?

- A. "john" == "john"      B. "john".equals("john")  
C. "john" = "john"      D. "john".equals(new Button("john"))

5、什么情况下, 类里面被自动加上缺省构造函数?

- A. 当定义任何一个类的时候  
B. 当一个类没有其他的构造函数时  
C. 当类里面定义了至少一个构造函数时

6、给出下列类定义:

```
public class Ombersley{  
    public static void main(String argv[]){  
        boolean b1 = true;  
        if((b1 ==true) || place(true)){  
            System.out.println("Hello Crowle");  
        }  
    }  
  
    public static boolean place(boolean location){  
        if(location==true){  
            System.out.println("Borcestshire");  
        }  
        System.out.println("Powick");  
        return true;  
    }  
}
```

当把它们编译并运行的时候会发生什么现象?

字符串

- 1) 编译错误;
- 2) 输出: "Hello Crowle"
- 3) 输出:  
Hello Crowle  
Boreetshire  
Powick
- 4) 无输出

7、代码如下:

```
1. public class WhileExam {  
2.     public static void main (String [] args) {  
3.         int x= 1, y = 6;  
4.         while (y--){x--;}  
5.         System.out.println("x=" + x + "y=" + y);  
6.     }  
7. }
```

结果是什么?

- A. The output is x = 6 y = 0      B. The output is x = 7 y = 0  
C. The output is x = 6 y = -1      D. The output is x = 7 y = -1  
E. 编译失败

8、代码如下:

```
1. public class Foo {  
2.     private int val;  
3.     public Foo(int v){ val = v; }  
4.     public static void main (String [] args) {  
5.         Foo a = new Foo (10);  
6.         Foo b = new Foo (10);  
7.         Foo c = a;  
8.         int d = 10;  
9.         double e = 10.0;  
10.    }  
11. }
```

下面哪个逻辑表达式为真?

- A. (a == c)      B. (b == d)      C. (a == b)  
D. (b == c)      E. (d == 10.0)

9、代码如下:

```
1. int i= 1, j=0;  
2. switch(i){  
3.     case 2:  
4.         j+=6;  
5.     case 4:  
6.         j+= 1;
```

二、易错题

7. case 1:  
8. j = 2;  
9. case 0:  
10. j + 4;  
11. }  
最终 j 的值是什么?  
A. 0      B. 1      C. 2      D. 4      E. 6

10. 看下面的程序段，当 oneMethod() 方法正常运行时，会显示什么结果？

```
public void test() {  
    try { oneMethod();  
        System.out.println("condition 1");  
    } catch (ArrayIndexOutOfBoundsException e) {  
        System.out.println("condition 2");  
    } catch (Exception e) {  
        System.out.println("condition 3");  
    } finally {  
        System.out.println("finally");  
    }  
}
```

最终显示结果是什么?  
A. condition 1      B. condition 2      C. condition 3      D. finally

## 二、编程题（40 分）

下面是一个由\*号组成的 4 行倒三角形图案。要求：1、输入倒三角形的行数，行数的取值 3-21 之间，对于非法的行数，要求抛出提示“非法行数！”；2、在屏幕上打印这个指定了行数的倒三角形。（程序请写在本纸张的背面）

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

**亚信联创 JAVA 及数据库笔试考卷—A**  
**亚信联创 JAVA 及数据库笔试考卷**

学员基本信息（请务必填写清晰）

姓名			身份证号				民族		
性别		电话、邮箱				家乡			
毕业院校			专业			学历、学位			
培训校区		根据自评成绩填报工作志愿地			①	②	③		
毕业时间			若明年毕业，能否参加实习工作				国家英语等级		

注意事项：

1. 亚信联创版权所有，考生不可带离考场。不可泄题，未经许可不可引用；
2. 考生带身份证入场，监考老师核实身份。代考是严重的作弊行为。
3. 闭卷考试，夹带、商议、抄袭、使用工具书、电脑、手机、传呼机都视为作弊行为。将取消成绩，并进入黑名单；
4. 考试时间为 2 小时（09:30-11:30）。迟到 15 分钟后不得入场。开始 45 分钟后才可提前交卷。不鼓励提前交卷；
5. 自带黑、蓝色圆珠笔或碳墨笔。字迹要求清晰、工整。用铅笔或字迹不可辨识的不能得分；
6. 考试用纸统一发放，额外需要可向监考老师申请。考生不可将考卷或草稿用纸带出考场；
7. 请考试自行填报姓名、身份证、联系方式等个人真是信息。
8. 查看考卷是否页数完整，四类大题，共 13 张。草稿可用试卷背面。

**一、选择题（不定项选择，答案可能不唯一。每题 2 分，共 40 分）**

1、Given the following class, which statements can be inserted at position 1 without causing the code to fail compilation?      1 答案为：\_\_\_\_\_

```

public class Q6db8 {
    int a;
    int b = 0;
    static int c;
    public void m() {
        int d;
        int e = 0;
        // Position 1
    }
}
  
```

亚信联创 JAVA 及数据库笔试考卷—A

- A. a++;
- B. b++;
- C. c++;
- D. d++;
- E. e++;

2、Consider the following line of code: int x[] = new int[25]; After execution, which statement or statements are true?  
2 答案为: \_\_\_\_\_

- A. x[24] is 0.
- B. x[24] is undefined.
- C. x[25] is 0.
- D. x[0] is null.
- E. x.length is 25.

3、Consider the following application:

```
1. class Q6 {  
2.     public static void main(String args[]) {  
3.         Holder h = new Holder();  
4.         h.held = 100;  
5.         h.bump(h);  
6.         System.out.println(h.held);  
7.     }  
8. }  
9.  
10. class Holder {  
11.     public int held;  
12.     public void bump(Holder theHolder) { theHolder.held++; }  
13. }
```

What value is printed out at line 6?

3 答案为: \_\_\_\_\_

- A. 0
- B. 1
- C. 100
- D. 101

亚信联创 JAVA 及数据库笔试考卷一A

4、The demonstrate(s) belongs to ‘has a’ relationship? 4 答案为: \_\_\_\_\_

A. public interface Person { }

public class Employee extends Person{ }

B. public interface Shape { }

public interface Rectandle extends Shape { }

C. public interface Colorable { }

public class Shape implements Colorable { }

D. public class Species{ }

public class Animal{private Species species;}

E. interface Component{ }

class Container implements Component{ private Component[] children; }

5、Consider the following code:

```
1. for (int i = 0; i < 2; i++) {  
2.     for (int j = 0; j < 3; j++) {  
3.         if (i == j) {  
4.             continue;  
5.         }  
6.         System.out.println("i= " + i + " j=" + j);  
7.     }  
8. }
```

Which lines would be part of the output?

5 答案为: \_\_\_\_\_

A. i=0 j=0

B. i=0 j=1

C. i=0 j=2

D. i=1 j=0

E. i=1 j=1

F. i=1 j=2

6、Consider the following class hierarchy and code fragments:

java.lang.Exception

\

java.io.IOException

/

\

java.io.StreamCorruptedException java.net.MalformedURLException

亚信联创 JAVA 及数据库笔试考卷一A

- A. Test t = new Test();
- B. Test t = new Test(1);
- C. Test t = new Test(1, 2);
- D. Test t = new Test(1, 2, 3);
- E. Test t = (new Base()).new Test(1);

8、Which one statement is true about the code fragment below?

- 1. String s = "abcde";
- 2. StringBuffer s1 = new StringBuffer("abcde");
- 3. if (s.equals(s1))
- 4.     s1 = null;
- 5. if (s1.equals(s))
- 6.     s = null;

8 答案为: \_\_\_\_\_

- A. Compilation fails at line 1, because the String constructor must be called explicitly.
- B. Compilation fails at line 3, because s and s1 have different types.
- C. Compilation succeeds. During execution, an exception is thrown at line 3.
- D. Compilation succeeds. During execution, an exception is thrown at line 5.
- E. Compilation succeeds. No exception is thrown during execution.

9、Which would be most suitable for storing data elements that must not appear in the store more than once, if searching is not a priority?

- A. Collection
- B. List
- C. Set
- D. Map
- E. Vector

9 答案为: \_\_\_\_\_

10、Given that Thing is a class, how many objects and reference variables are created by the following code?

Thing item, stuff;

item = new Object();

Thing entity = new Object();

10 答案为: \_\_\_\_\_

- A. One object is created.
- B. Two objects are created.
- C. Three objects are created.
- D. One reference variable is created.
- E. Two reference variables are created.
- F. Three reference variables are created.

亚信联创 JAVA 及数据库笔试考卷—A

11、Examine the structure of the EMPLOYEES table:

EMPLOYEE_ID	NUMBER	Primary Key
FIRST_NAME	VARCHAR2(25)	
LAST_NAME	VARCHAR2(25)	

Which three statements insert a row into the table? (Choose three.)

- A. `INSERT INTO employees VALUES (NULL, 'John', 'Smith');`
- B. `INSERT INTO employees(first_name, last_name) VALUES('John', 'Smith');`
- C. `INSERT INTO employees VALUES ('1000', 'John', NULL);`
- D. `INSERT INTO employees(first_name, last_name, employee_id) VALUES (1000, 'John', 'Smith');`
- E. `INSERT INTO employees(employee_id) VALUES (1000);`
- F. `INSERT INTO employees(employee_id, first_name, last_name) VALUES (1000, 'John', '');`

11 答案为: \_\_\_\_\_

12、Evaluate the SQL statement:

`SELECT ROUND(45.953, -1), TRUNC(45.936, 2) FROM dual;`

Which values are displayed?

- A. 46 and 45
- B. 46 and 45.93
- C. 50 and 45.93
- D. 50 and 45.9
- E. 45 and 45.93
- F. 45.95 and 45.93

12 答案为: \_\_\_\_\_

13、Click the Exhibit button to examine the data of the EMPLOYEES table.

Which statement lists the ID, name, and salary of the employee, and the ID and name of the employee's manager, for all the employees who have a manager and earn more than 4000?

13 答案为: \_\_\_\_\_

EMPLOYEES (EMPLOYEE\_ID is the primary key. MGR\_ID is the ID of managers and refers to the EMPLOYEE\_ID)

EMPLOYEE_ID	EMP_NAME	DEPT_ID	MGR_ID	JOB_ID	SALARY
101	Smith	20	120	SA REP	4000
102	Martin	10	105	CLERK	2500
103	Chris	20	120	IT ADMIN	4200
104	John	30	108	HR CLERK	2500
105	Diana	30	108	HR MGR	5000
106	Bryan	40	110	AD ASST	3000
108	Jennifer	30	110	HR DIR	6500
110	Bob	40		EX DIR	8000
120	Ravi	20	110	SA DIR	6500

亚信联创 JAVA 及数据库笔试考卷—A

```

A.   SELECT employee_id "Emp_id", emp_name "Employee",
      salary,
      employee_id "Mgr_id", emp_name "Manager"
     FROM   employees
    WHERE   salary > 4000;
B.   SELECT e.employee_id "Emp_id", e.emp_name "Employee",
      e.salary,
      m.employee_id "Mgr_id", m.emp_name "Manager"
     FROM   employees e JOIN employees m
    WHERE   e.mgr_id = m.mgr_id
    AND    e.salary > 4000;
C.   SELECT e.employee_id "Emp_id", e.emp_name "Employee", e.salary, m.employee_id "Mgr_id",
      m.emp_name "Manager" FROM employees e JOIN employees m ON (e.mgr_id = m.employee_id) AND
      e.salary > 4000;
D.   SELECT e.employee_id "Emp_id", e.emp_name "Employee",
      e.salary,
      m.mgr_id "Mgr_id", m.emp_name "Manager"
     FROM   employees e SELF JOIN employees m
    WHERE   e.mgr_id = m.employee_id
    AND    e.salary > 4000;
E.   SELECT e.employee_id "Emp_id", e.emp_name "Employee",
      e.salary,
      m.mgr_id "Mgr_id", m.emp_name "Manager"
     FROM   employees e JOIN employees m
    USING  (e.employee_id = m.employee_id)
    AND    e.salary > 4000;

```

14、Which two statements about sequences are true? (Choose two.) 14答案为: \_\_\_\_\_

- A. You use a **NEXTVAL** pseudo column to look at the next possible value that would be generated from a sequence, without actually retrieving the value.
- B. You use a **CURRVAL** pseudo column to look at the current value just generated from a sequence, without affecting the further values to be generated from the sequence.
- C. You use a **NEXTVAL** pseudo column to obtain the next possible value from a sequence by actually retrieving the value from the sequence.
- D. You use a **CURRVAL** pseudo column to generate a value from a sequence that would be used for a specified database column.

亚信联创 JAVA 及数据库笔试考卷—A

- E. If a sequence starting from a value 100 and incremented by 1 is used by more than one application, then all of these applications could have a value of 105 assigned to their column whose value is being generated by the sequence.
- F. You use a REUSE clause when creating a sequence to restart the sequence once it generates the maximum value defined for the sequence.

15、In which two cases would you use an outer join? (Choose two.) 15答案为: \_\_\_\_\_

- A. The tables being joined have NOT NULL columns.
- B. The tables being joined have only matched data.
- C. The columns being joined have NULL values.
- D. The tables being joined have only unmatched data.
- E. The tables being joined have both matched and unmatched data.
- F. Only when the tables have a primary key-foreign key relationship.

16、Which two statements complete a transaction? (Choose two.) 16答案为: \_\_\_\_\_

- A. DELETE employees;
  - B. DESCRIBE employees;
  - C. ROLLBACK TO SAVEPOINT C;
  - D. GRANT SELECT ON employees TO SCOTT;
  - E. ALTER TABLE employees SET UNUSED COLUMN sal;
  - F. SELECT MAX(sal)
- ```

FROM employees
WHERE department_id = 20;
  
```

17、Click the Exhibit button and examine the data in the EMPLOYEES table. Which three subqueries work? (Choose three.)

17答案为: \_\_\_\_\_

| LAST NAME | DEPARTMENT ID | SALARY |
|-----------|---------------|--------|
| Getz      | 10            | 3000   |
| Davis     | 20            | 1500   |
| King      | 20            | 2200   |
| Davis     | 30            | 5000   |
| ...       |               |        |

- A. SELECT \*  
FROM employees  
where salary > (SELECT MIN(salary)  
FROM employees  
GROUP BY department\_id);

亚信联创 JAVA 及数据库笔试考卷—A

B. **SELECT \***

```
FROM employees
WHERE salary = (SELECT AVG(salary)
FROM employees
GROUP BY department_id);
```

C. **SELECT distinct department\_id FROM employees WHERE salary > ANY (SELECT AVG(salary) FROM employees GROUP BY department\_id);**

D. **SELECT department\_id FROM employees WHERE salary > ALL (SELECT AVG(salary) FROM employees GROUP BY department\_id);**

E. **SELECT last\_name FROM employees WHERE salary > ANY (SELECT MAX(salary) FROM employees GROUP BY department\_id);**

F. **SELECT department\_id**

```
FROM employees
WHERE salary > ALL (SELECT AVG(salary)
FROM employees
GROUP BY AVG(SALARY));
```

18、Your production database is running in the ARCHIVELOG mode and the ARCn process is functional. You have two online redo log groups. Which three background processes would be involved when a log switch happens? (Choose three.)      18答案为：\_\_\_\_\_

- A.archival
- B.log writer
- C.database writer
- D.system monitor
- E.process monitor
- F.change tracking writer

19、You have many users complaining about slow inserts into a large table. While investigating the reason, you find that the number of indexes on the table is high. You want to find out which indexes are not being used.

Which method would you follow to achieve this?

19答案为：\_\_\_\_\_

- A.enable index monitoring and query the DBA\_OBJECTS view
- B.enable index monitoring and query the DBA\_INDEXES view
- C.enable index monitoring and query the V\$OBJECT\_USAGE view
- D.enable index monitoring and view the DBA\_INDEXTYPE\_COMMENTS view

**亚信联创 JAVA 及数据库笔试考卷一A**

20、 You executed the STARTUP MOUNT command to start your database. For which database operation do you need to start the database in the MOUNT state?

20答案为：\_\_\_\_\_

- A) renaming the control files
- B) dropping a user in your database
- C) enabling or disabling redo log archiving
- D) dropping a tablespace in your database
- E) re-creating the control files, after you lost all the control files in your database

**二、问答题（每题 5 分，共 20 分）**

1、写出至少 5 个 JSP 的隐含对象。

2、以文字形式描述 servlet 的生命周期。

3、简要说明 Overload 和 Override 的区别。

4、简述 oracle 数据库打开的状态，及各个状态的特点。

亚信联创 JAVA 及数据库笔试考卷—A

三、Java 编程题（每题 10 分，共 20 分）

- 现在有方法 do1(); do2(); do3(), 请书写一个公有静态函数 void doSomething(), 传入参数 int iParam; 函数功能：如果 iParam==1 则调用 do1(), 如果 iParam==2 则调用 do2(), 如果 iParam==3 则调用 do3(), 否则抛出异常。请按标准格式编写。

亚信联创 JAVA 及数据库笔试考卷一A

2. 写一个分割函数，将一个 String 按某个符号分割为一个 String 数组。  
例如：ab,c,4,5 如果用逗号分割，则结果为 [ab c 4 5];要求自己实现算法，不得调用 String  
类的 split 方法。如果记不清楚 java api 函数，可以用伪代码描述。

亚信联创 JAVA 及数据库笔试考卷—A

四、数据库基础知识（每题10分，共20分）

1、有 3 张表：

定单表(t\_order)含字段：定单号(orderid), 定单名(name), 定单类型编号(typeid), 订购日期(orderdate)

定单类型表(t\_ordertype)含字段：定单类型编号 (typeid), 定单类型名称 (name)

定单明细表(t\_orderitem)含字段：定单号 (orderid), 定单明细编号 (itemid), 销售金额 (money\_detail)

说明：定单类型(t\_ordertype)与定单(t\_order)是 1:n 的关系，通过两表的字段 typeid 作外键关联，定单 (t\_order) 与定单明细(t\_orderitem)是 1:n 的关系，通过两表的字段 orderid 作外键关联

问题：查询符合如下条件的结果集

- (1) 定单类型名称为'abc'
- (2) 并且定单明细的销售金额大于 60
- (3) 并且定购日期是 2004 年 1 月 1 日

2. 表 acct\_item(acct\_item\_id,acct\_id,acct\_item\_type\_id,charge (费用, 单位分))

问题：查询 acct\_id 总费用大于 100 元的合同号，按总费用降序排序，总费用相同时，请按 acct\_id 升序排序

学员重点掌握部分！

## 1. Java 基础部分

基础部分的顺序：基本语法，类相关的语法，内部类的语法，继承相关的语法，异常的语法，线程的语法，集合的语法，io 的语法，虚拟机方面的语法。

### 1、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？

可以有多个类，但只能有一个 public 的类，并且 public 的类名必须与文件名相一致。

### 2、Java 有没有 goto？

java 中的保留字，现在没有在 java 中使用。

### 3、说说&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，例如，对于 if(str != null && !str.equals("")) 表达式，当 str 为 null 时，后面的表达式不会执行，所以不会出现 NullPointerException。如果将&&改为&，则会抛出 NullPointerException 异常。If(x==33 & ++y>0) y 会增长，If(x==33 && ++y>0) 不会增长

&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，我们通常使用 0x0f 来与一个整数进行&运算，来获取该整数的最低 4 个 bit 位，例如，0x31 & 0x0f 的结果为 0x01。

备注：这道题先说两者的共同点，再说出&&和&的特殊之处，并列举一些经典的例子来表明自己理解透彻深入、实际经验丰富。

### 4、在 JAVA 中如何跳出当前的多重嵌套循环？

在 Java 中，要想跳出多重循环，可以在外面的循环语句前定义一个标号，然后在里层循环体的代码中使用带有标号的 break 语句，即可跳出外层循环。例如，

```
ok:  
for(int i=0;i<10;i++) {  
    for(int j=0;j<10;j++) {  
        System.out.println("i=" + i + ",j=" + j);  
        if(j == 5) break ok;  
    }  
}
```

另外，我个人通常并不使用标号这种方式，而是让外层的循环条件表达式的结果可以受到里层循环体代码的控制，例如，要在二维数组中查找到某个数字。

```
int arr[][] = {{1,2,3},{4,5,6,7},{9}};  
boolean found = false;  
for(int i=0;i<arr.length && !found;i++) {  
    for(int j=0;j<arr[i].length;j++){  
        System.out.println("i=" + i + ",j=" + j);  
        if(arr[i][j] == 5) {  
            found = true;  
            break;  
        }  
    }  
}
```

```
}
```

## 5、JDK1.6 的环境，switch 语句能否作用在 byte 上，能否作用在 long 上，能否作用在 String 上？

在 `switch(expr1)` 中，`expr1` 只能是一个整数表达式或者枚举常量（更大字体），整数表达式可以是 `int` 基本类型或 `Integer` 包装类型，由于 `byte, short, char` 都可以隐含转换为 `int`，所以，这些类型以及这些类型的包装类型也是可以的。显然，`long` 和 `String` 类型都不符合 `switch` 的语法规规定，并且不能被隐式转换成 `int` 类型，所以，它们不能作用于 `switch` 语句中。

## 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 型变量中能不能贮一个中文汉字?为什么?

`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汉字，所以，`char` 型变量中当然可以存储汉字啦。不过，如果某个特殊的汉字没有被包含在 `unicode` 编码字符集中，那么，这个 `char` 型变量中就不能存储这个特殊汉字。补充说明：`unicode` 编码占用两个字节，所以，`char` 类型的变量也是占用两个字节。

备注：后面一部分回答虽然不是在正面回答题目，但是，为了展现自己的学识和表现自己对问题理解的透彻深入，可以回答一些相关的知识，做到知无不言，言无不尽。

## 8、用最有效率的方法算出 2 乘以 8 等於几?

```
2 << 3,
```

因为将一个数左移 `n` 位，就相当于乘以了 `2` 的 `n` 次方，那么，一个数乘以 `8` 只要将其左移 `3` 位即可，而位运算 `cpu` 直接支持的，效率最高，所以，`2` 乘以 `8` 等於几的最效率的方法是 `2 << 3`。

## 9、请设计一个一百亿的计算器

首先要明白这道题目的考查点是什么，一是大家首先要对计算机原理的底层细节要清楚、要知道加减法的位运算原理和知道计算机中的算术运算会发生越界的情况，二是要具备一定的面向对象的设计思想。

首先，计算机中用固定数量的几个字节来存储的数值，所以计算机中能够表示的数值是有一定的范围的，为了便于讲解和理解，我们先以 `byte` 类型的整数为例，它用 1 个字节进行存储，表示的最大数值范围为 -128 到 +127。-1 在内存中对应的二进制数据为 11111111，如果两个 -1 相加，不考虑 Java 运算时的类型提升，运算后会产生进位，二进制结果为 1,11111110，由于进位后超过了 `byte` 类型的存储空间，所以进位部分被舍弃，即最终的结果为 11111110，也就是 -2，这正好利用溢位的方式实现了负数的运算。-128 在内存中对应的二进制数据为 10000000，如果两个 -128 相加，不考虑 Java 运算时的类型提升，运算后会产生进位，二进制结果为 1,00000000，由于进位后超过了 `byte` 类型的存储空间，所以进位部分被舍弃，即最终的结果为 00000000，也就是 0，这样的结果显然不是我们期望的，这说明 **计算机中的算术运算是会发生越界情况的，两个数值的运算结果不能超过计算机中的该类型的数值范围**。由于 Java 中涉及表达式运算时的类型自动提升，我们无法用 `byte` 类型来做演示这种问题和现象的实验，大家可以用下面一个使用整数做实验的例子程序体验一下：

```
int a = Integer.MAX_VALUE;
int b = Integer.MAX_VALUE;
int sum = a + b;
System.out.println("a="+a+",b="+b+",sum="+sum);
```

先不考虑 long 类型，由于 int 的正数范围为 2 的 31 次方，表示的最大数值约等于  $2 \times 1000 \times 1000 \times 1000$ ，也就是 20 亿的大小，所以，要实现一个一百亿的计算器，我们得自己设计一个类可以用于表示很大的整数，并且提供了与另外一个整数进行加减乘除的功能，大概功能如下：

( ) 这个类内部有两个成员变量，一个表示符号，另一个用字节数组表示数值的二进制数

( ) 有一个构造方法，把一个包含有多位数值的字符串转换到内部的符号和字节数组中

( ) 提供加减乘除的功能

```
public class BigInteger{
    int sign;
    byte[] val;
    public BigInteger(String val) {
        sign = ;
        val = ;
    }
    public BigInteger add(BigInteger other) {
    }
    public BigInteger subtract(BigInteger other) {
    }
    public BigInteger multiply(BigInteger other) {
    }
    public BigInteger divide(BigInteger other) {
    }
}
```

**备注：**要想写出这个类的完整代码，是非常复杂的，如果有兴趣的话，可以参看 jdk 中自带的 `java.math.BigInteger` 类的源码。面试的人也知道谁都不可能在短时间内写出这个类的完整代码的，他要的是你是否有这方面的概念和意识，他最重要的还是考查你的能力，所以，你不要因为自己无法写出完整的最终结果就放弃答这道题，你要做的就是你比别人写得多，证明你比别人强，你有这方面的思想意识就可以了，毕竟别人可能连题目的意思都看不懂，什么都没写，你要敢于答这道题，即使只答了一部分，那也与那些什么都不懂的人区别出来，拉开了距离，算是矮子中的高个，机会当然就属于你了。另外，答案中的框架代码也很重要，体现了一些面向对象设计的功底，特别是其中的方法命名很专业，用的英文单词很精准，这也是能力、经验、专业性、英语水平等多个方面的体现，会给人留下很好的印象，在编程能力和其他方面条件差不多的情况下，英语好除了可以使你获得更多机会外，薪水可以高出一千元。

## 10、使用 final 关键字修饰一个变量时，是引用不能变，还是引用的对象不能变？

使用 final 关键字修饰一个变量时，是指引用变量不能变，引用变量所指向的对象中的内容还是可以改变的。例如，对于如下语句：

```
final StringBuffer a=new StringBuffer("immutable");
```

执行如下语句将报告编译期错误：

```
a=new StringBuffer("");
```

但是，执行如下语句则可以通过编译：

```
a.append(" broken!");
```

有人在定义方法的参数时，可能想采用如下形式来阻止方法内部修改传进来的参数对象：

```
public void method(final StringBuffer param){}
```

实际上，这是办不到的，在该方法内部仍然可以增加如下代码来修改参数对象：

```
param.append("a");
```

## 11、"=="和 equals 方法究竟有什么区别？

(单独把一个东西说清楚，然后再说清楚另一个，这样，它们的区别自然就出来了，混在一起说，则很难说清楚)

==操作符专门用来比较两个变量的值是否相等，也就是用于比较变量所对应的内存中所存储的数值是否相同，要比较两个基本类型的数据或两个引用变量是否相等，只能用==操作符。

如果一个变量指向的数据是对象类型的，那么，这时候涉及了两块内存，对象本身占用一块内存（堆内存），变量也占用一块内存，例如 `Object obj = new Object();`；变量 `obj` 是一个内存，`new Object()` 是另一个内存，此时，变量 `obj` 所对应的内存中存储的数值就是对象占用的那块内存的首地址。对于指向对象类型的变量，如果要比较两个变量是否指向同一个对象，即要看这两个变量所对应的内存中的数值是否相等，这时候就需要用==操作符进行比较。

equals 方法是用于比较两个独立对象的内容是否相同，就好比去比较两个人的长相是否相同，它比较的两个对象是独立的。例如，对于下面的代码：

```
String a=new String("foo");
String b=new String("foo");
```

两条 new 语句创建了两个对象，然后用 `a, b` 这两个变量分别指向了其中一个对象，这是两个不同的对象，它们的首地址是不同的，即 `a` 和 `b` 中存储的数值是不相同的，所以，表达式 `a==b` 将返回 `false`，而这两个对象中的内容是相同的，所以，表达式 `a.equals(b)` 将返回 `true`。

在实际开发中，我们经常要比较传递进来的字符串内容是否等，例如，`String input = ...; input.equals("quit")`，许多人稍不注意就使用==进行比较了，这是错误的，随便从网上找几个项目实战的教学视频看看，里面有大量这样的错误。记住，字符串的比较基本上都是使用 equals 方法。

如果一个类没有自己定义 equals 方法，那么它将继承 Object 类的 equals 方法，Object 类的 equals 方法的实现代码如下：

```
boolean equals(Object o) {
    return this==o;
}
```

这说明，如果一个类没有自己定义 equals 方法，它默认的 equals 方法（从 Object 类继承的）就是使用==操作符，也是在比较两个变量指向的对象是否是同一对象，这时候使用 equals 和使用==会得到同样的结果，如果比较的是两个独立的对象则总返回 false。如果你编写的类希望能够比较该类创建的两个实例对象的内容是否相同，那么你必须覆盖 equals 方法，由你自己写代码来决定在什么情况即可认为两个对象的内容是相同的。

## 12、静态变量和实例变量的区别？

在语法定义上的区别：静态变量前要加 static 关键字，而实例变量前则不加。

在程序运行时的区别：实例变量属于某个对象的属性，必须创建了实例对象，其中的实例变量才会被分配空间，才能使用这个实例变量。静态变量不属于某个实例对象，而是属于类，所以也称为类变量，只要程序加载了类的字节码，不用创建任何实例对象，静态变量就会被分配空间，静态变量就可以被使用了。总之，实例变量必须创建对象后才可以使用，静态变量则可以直接使用类名来引用。

例如，对于下面的程序，无论创建多少个实例对象，永远都只分配了一个 staticVar 变量，并且每创建一个实例对象，这个 staticVar 就会加 1；但是，每创建一个实例对象，就会分配一个 instanceVar，即可能分配多个 instanceVar，并且每个 instanceVar 的值都只自加了 1 次。

```
public class VariantTest{
    public static int staticVar = 0;
    public int instanceVar = 0;
    public VariantTest(){
        staticVar++;
        instanceVar++;
        System.out.println("staticVar=" + staticVar + ",instanceVar=" + instanceVar);
    }
}
```

备注：这个解答除了说清楚两者的区别外，最后还用一个具体的应用例子来说明两者的差异，体现了自己有很好的解决问题和设计案例的能力，思维敏捷，超过一般程序员，有写作能力！

### 13、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。因为非 **static** 方法是要与对象关联在一起的，必须创建一个对象后，才可以在该对象上进行方法调用，而 **static** 方法调用时不需要创建对象，可以直接调用。也就是说，当一个 **static** 方法被调用时，可能还没有创建任何实例对象，如果从一个 **static** 方法中发出对非 **static** 方法的调用，那个非 **static** 方法是关联到哪个对象上的呢？这个逻辑无法成立，所以，一个 **static** 方法内部发出对非 **static** 方法的调用。

### 14、**Integer** 与 **int** 的区别

**int** 是 Java 提供的 8 种原始数据类型之一。Java 为每个原始类型提供了封装类，**Integer** 是 Java 为 **int** 提供的封装类。**int** 的默认值为 0，而 **Integer** 的默认值为 **null**，即 **Integer** 可以区分出未赋值和值为 0 的区别，**int** 则无法表达出未赋值的情况，例如，要想表达出没有参加考试和考试成绩为 0 的区别，则只能使用 **Integer**。在 JSP 开发中，**Integer** 的默认为 **null**，所以用 **el** 表达式在文本框中显示时，值为空白字符串，而 **int** 默认的默认值为 0，所以用 **el** 表达式在文本框中显示时，结果为 0，所以，**int** 不适合作为 web 层的表单数据的类型。

在 Hibernate 中，如果将 **OID** 定义为 **Integer** 类型，那么 Hibernate 就可以根据其值是否为 **null** 而判断一个对象是否是临时的，如果将 **OID** 定义为了 **int** 类型，还需要在 **hbm** 映射文件中设置其 **unsaved-value** 属性为 0。

另外，**Integer** 提供了多个与整数相关的操作方法，例如，将一个字符串转换成整数，**Integer** 中还定义了表示整数的最大值和最小值的常量。

### 15、**Math.round(11.5)** 等於多少？**Math.round(-11.5)** 等於多少？

**Math** 类中提供了三个与取整有关的方法：**ceil**、**floor**、**round**，这些方法的作用与它们的英文名称的含义相对应，例如，**ceil** 的英文意义是天花板，该方法就表示向上取整，**Math.ceil(11.3)** 的结果为 12，**Math.ceil(-11.3)** 的结果是 -11；**floor** 的英文意义是地板，该方法就表示向下取整，**Math.floor(11.6)** 的结果为 11，**Math.floor(-11.6)** 的结果是 -12；最难掌握的是 **round** 方法，它表示“四舍五入”，算法为 **Math.floor(x+0.5)**，即将原来的数字加上 0.5 后再向下取整，所以，**Math.round(11.5)** 的结果为 12，**Math.round(-11.5)** 的结果为 -11。

### 16、下面的代码有什么不妥之处？

```
1. if(username.equals("zxx")) {}  
2. int x = 1;  
return x==1?true:false;
```

### 17、请说出作用域 **public**, **private**, **protected**, 以及不写时的区别

这四个作用域的可见范围如下表所示。

说明：如果在修饰的元素上面没有写任何访问修饰符，则表示 **friendly**。

| 作用域              | 当前类 | 同一 package | 子孙类 | 其他 package |
|------------------|-----|------------|-----|------------|
| <b>public</b>    | √   | √          | √   | √          |
| <b>protected</b> | √   | √          | √   | ✗          |
| <b>friendly</b>  | √   | √          | ✗   | ✗          |
| <b>private</b>   | √   | ✗          | ✗   | ✗          |

备注：只要记住了有 4 种访问权限，4 个访问范围，然后将全选和范围在水平和垂直方向上分别按排从小到大或从大到小的顺序排列，就很容易画出上面的图了。

## 18、Overload 和 Override 的区别。Overloaded 的方法是否可以改变返回值的类型？

Overload 是重载的意思，Override 是覆盖的意思，也就是重写。

重载 Overload 表示同一个类中可以有多个名称相同的方法，但这些方法的参数列表各不相同（即参数个数或类型不同）。

重写 Override 表示子类中的方法可以与父类中的某个方法的名称和参数完全相同，通过子类创建的实例对象调用这个方法时，将调用子类中的定义方法，这相当于把父类中定义的那个完全相同的方法给覆盖了，这也是面向对象编程的多态性的一种表现。子类覆盖父类的方法时，只能比父类抛出更少的异常，或者是抛出父类抛出的异常的子异常，因为子类可以解决父类的一些问题，不能比父类有更多的问题。子类方法的访问权限只能比父类的更大，不能更小。如果父类的方法是 private 类型，那么，子类则不存在覆盖的限制，相当于子类中增加了一个全新的方法。

至于 Overloaded 的方法是否可以改变返回值的类型这个问题，要看你到底想问什么呢？这个题目很模糊。如果几个 Overloaded 的方法的参数列表不一样，它们的返回者类型当然也可以不一样。但我估计你想问的问题是：如果两个方法的参数列表完全一样，是否可以让它们的返回值不同来实现重载 Overload。这是不行的，我们可以用反证法来说明这个问题，因为我们有时候调用一个方法时也可以不定义返回结果变量，即不要关心其返回结果，例如，我们调用 map.remove(key) 方法时，虽然 remove 方法有返回值，但是我们通常都不会定义接收返回结果的变量，这时候假设该类中有两个名称和参数列表完全相同的方法，仅仅是返回类型不同，java 就无法确定编程者到底是想调用哪个方法了，因为它无法通过返回结果类型来判断。

override 可以翻译为覆盖，从字面就可以知道，它是覆盖了一个方法并且对其重写，以求达到不同的作用。对我们来说最熟悉的覆盖就是对接口方法的实现，在接口中一般只是对方法进行了声明，而我们在实现时，就需要实现接口声明的所有方法。除了这个典型的用法以外，我们在继承中也可能会在子类覆盖父类中的方法。在覆盖要注意以下的几点：

- 1、覆盖的方法的标志必须要和被覆盖的方法的标志完全匹配，才能达到覆盖的效果；
- 2、覆盖的方法的返回值必须和被覆盖的方法的返回一致；
- 3、覆盖的方法所抛出的异常必须和被覆盖方法的所抛出的异常一致，或者是其子类；
- 4、被覆盖的方法不能为 private，否则在其子类中只是新定义了一个方法，并没有对其进行覆盖。

overload 对我们来说可能比较熟悉，可以翻译为重载，它是指我们可以定义一些名称相同的方法，通过定义不同的输入参数来区分这些方法，然后再调用时，VM 就会根据不同的参数样式，来选择合适的方法执行。在使用重载要注意以下的几点：

- 1、在使用重载时只能通过不同的参数样式。例如，不同的参数类型，不同的参数个数，不同的参数顺序（当然，同一方法内的几个参数类型必须不一样，例如可以是 fun(int,float)，但是不能为 fun(int,int)）；
- 2、不能通过访问权限、返回类型、抛出的异常进行重载；
- 3、方法的异常类型和数目不会对重载造成影响；
- 4、对于继承来说，如果某一方法在父类中是访问权限是 private，那么就不能在子类对其进行重载，如果定义的话，也只是定义了一个新方法，而不会达到重载的效果。

## 19、构造器 Constructor 是否可被 override？

构造器 Constructor 不能被继承，因此不能重写 Override，但可以被重载 Overload。

## 20、接口是否可继承接口？抽象类是否可实现(implements)接口？抽象类是否可继承具体类(concrete class)？抽象类

## 中是否可以有静态的 **main** 方法？

接口可以继承接口。抽象类可以实现 (`implements`) 接口，抽象类是否可继承具体类。抽象类中可以有静态的 `main` 方法。

备注：只要明白了接口和抽象类的本质和作用，这些问题都很好回答，你想想，如果你是 `java` 语言的设计者，你是否会提供这样的支持，如果不提供的话，有什么理由吗？如果你没有道理不提供，那答案就是肯定的了。

只有记住抽象类与普通类的唯一区别就是不能创建实例对象和允许有 `abstract` 方法。

## 21、写 `clone()` 方法时，通常都有一行代码，是什么？

`clone` 有缺省行为，`super.clone()`；因为首先要把父类中的成员复制到位，然后才是复制自己的成员。

## 22、面向对象的特征有哪些方面

计算机软件系统是现实生活中的业务在计算机中的映射，而现实生活中的业务其实就是一个个对象协作的过程。面向对象编程就是按现实业务一样的方式将程序代码按一个个对象进行组织和编写，让计算机系统能够识别和理解用对象方式组织和编写的程序代码，这样就可以把现实生活中的业务对象映射到计算机系统中。

面向对象的编程语言有封装、继承、抽象、多态等 4 个主要的特征。

1 封装：

封装是保证软件部件具有优良的模块性的基础，封装的目标就是要实现软件部件的“高内聚、低耦合”，防止程序相互依赖性而带来的变动影响。在面向对象的编程语言中，对象是封装的最基本单位，面向对象的封装比传统语言的封装更为清晰、更为有力。面向对象的封装就是把描述一个对象的属性和行为的代码封装在一个“模块”中，也就是一个类中，属性用变量定义，行为用方法进行定义，方法可以直接访问同一个对象中的属性。通常情况下，**只要记住让变量和访问这个变量的方法放在一起，将一个类中的成员变量全部定义成私有的，只有这个类自己的方法才可以访问到这些成员变量，这就基本上实现对象的封装，就很容易找出要分配到这个类上的方法了，就基本上算是会面向对象的编程了。把握一个原则：把对同一事物进行操作的方法和相关的方法放在同一个类中，把方法和它操作的数据放在同一个类中。**

例如，人要在黑板上画圆，这一共涉及三个对象：人、黑板、圆，画圆的方法要分配给哪个对象呢？由于画圆需要使用到圆心和半径，圆心和半径显然是圆的属性，如果将它们在类中定义成了私有的成员变量，那么，画圆的方法必须分配给圆，它才能访问到圆心和半径这两个属性，人以后只是调用圆的画圆方法、表示给圆发消息而已，画圆这个方法不应该分配在人这个对象上，**这就是面向对象的封装性，即将对象封装成一个高度自治和相对封闭的个体，对象状态（属性）由这个对象自己的行为（方法）来读取和改变。**一个更便于理解的例子就是，司机将火车刹住了，刹车的动作是分配给司机，还是分配给火车，显然，应该分配给火车，因为司机自身是不可能有那么大的力气将一个火车给停下来的，只有火车自己才能完成这一动作，火车需要调用内部的离合器和刹车片等多个器件协作才能完成刹车这个动作，司机刹车的过程只是给火车发了一个消息，通知火车要执行刹车动作而已。

抽象：

抽象就是找出一些事物的相似和共性之处，然后将这些事物归为一个类，这个类只考虑这些事物的相似和共性之处，并且会忽略与当前主题和目标无关的那些方面，将注意力集中在与当前目标有关的方面。例如，看到一只蚂蚁和大象，你能够想象出它们的相同之处，那就是抽象。抽象包括行为抽象和状态抽象两个方面。例如，定义一个 `Person` 类，如下：

```
class Person{  
    String name;  
    int age;  
}
```

人本来是很复杂的事物，有很多方面，但因为当前系统只需要了解人的姓名和年龄，所以上面定义的类中只包含姓名和年龄这两个属性，这就是一种抽象，使用抽象可以避免考虑一些与目标无关的细节。我对抽象的理解就是不要用显微镜去看一个事物的所有方面，这样涉及的内容就太多了，而是要善于划分问题的边界，当前系统需要什么，就只考虑什么。

继承：

在定义和实现一个类的时候，可以在一个已经存在的类的基础之上进行，把这个已经存在的类所定义的内容作为自己的内容，并可以加入若干新的内容，或修改原来的方法使之更适合特殊的需要，这就是继承。继承是子类自动共享父类数据和方法的机制，这是类之间的一种关系，提高了软件的可重用性和可扩展性。

多态：

多态是指程序中定义的引用变量所指向的具体类型和通过该引用变量发出的方法调用在编程时并不确定，而是在程序运行期间才确定，即一个引用变量到底会指向哪个类的实例对象，该引用变量发出的方法调用到底是哪个类中实现的方法，必须在程序运行期间才能决定。因为在程序运行时才确定具体的类，这样不用修改源程序代码，就可以让引用变量绑定到各种不同的类实现上，从而导致该引用调用的具体方法随之改变，即不修改程序代码就可以改变程序运行时所绑定的具体代码，让程序可以选择多个运行状态，这就是多态性。多态性增强了软件的灵活性和扩展性。例如，下面代码中的 UserDao 是一个接口，它定义引用变量 userDao 指向的实例对象由 daofactory.getDao() 在执行的时候返回，有时候指向的是 UserJdbcDao 这个实现，有时候指向的是 UserHibernateDao 这个实现，这样，不用修改源代码，就可以改变 userDao 指向的具体类实现，从而导致 userDao.insertUser() 方法调用的具体代码也随之改变，即有时候调用的是 UserJdbcDao 的 insertUser 方法，有时候调用的是 UserHibernateDao 的 insertUser 方法：

```
UserDao userDao = daofactory.getDao();  
userDao.insertUser(user);
```

比喻：人吃饭，你看到的是左手，还是右手？

## 23、java 中实现多态的机制是什么？

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

## 24、abstract class 和 interface 有什么区别？

含有 abstract 修饰符的 class 即为抽象类，abstract 类不能创建的实例对象。含有 abstract 方法的类必须定义为 abstract class，abstract class 类中的方法不必是抽象的。abstract class 类中定义抽象方法必须在具体(Concrete)子类中实现，所以，不能有抽象构造方法或抽象静态方法。如果的子类没有实现抽象父类中的所有抽象方法，那么子类也必须定义为 abstract 类型。

接口(interface)可以说成是抽象类的一种特例，接口中的所有方法都必须是抽象的。接口中的方法定义默认为 public abstract 类型，接口中的成员变量类型默认为 public static final。

下面比较一下两者的语法区别：

1. 抽象类可以有构造方法，接口中不能有构造方法。
2. 抽象类中可以有普通成员变量，接口中没有普通成员变量
3. 抽象类中可以包含非抽象的普通方法，接口中的所有方法必须都是抽象的，不能有非抽象的普通方法。
4. 抽象类中的抽象方法的访问类型可以是 public, protected 和 (默认类型, 虽然 eclipse 下不报错，但应该也不行)，但接口中的抽象方法只能是 public 类型的，并且默认即为 public abstract 类型。
5. 抽象类中可以包含静态方法，接口中不能包含静态方法

6. 抽象类和接口中都可以包含静态成员变量，抽象类中的静态成员变量的访问类型可以任意，但接口中定义的变量只能是 `public static final` 类型，并且默认即为 `public static final` 类型。

## 7. 一个类可以实现多个接口，但只能继承一个抽象类。

下面接着再说说两者在应用上的区别：

接口更多的是在系统架构设计方法发挥作用，主要用于定义模块之间的通信契约。而抽象类在代码实现方面发挥作用，可以实现代码的重用，例如，模板方法设计模式是抽象类的一个典型应用，假设某个项目的所有 `Servlet` 类都要用相同的方式进行权限判断、记录访问日志和处理异常，那么就可以定义一个抽象的基类，让所有的 `Servlet` 都继承这个抽象基类，在抽象基类的 `service` 方法中完成权限判断、记录访问日志和处理异常的代码，在各个子类中只是完成各自的业务逻辑代码，伪代码如下：

```
public abstract class BaseServlet extends HttpServlet{
    public final void service(HttpServletRequest request, HttpServletResponse response) throws IOException, ServletException
    {
        记录访问日志
        进行权限判断
        if(具有权限) {
            try{
                doService(request, response);
            }
            catch(Excetpion e) {
                记录异常信息
            }
        }
    }
    protected abstract void doService(HttpServletRequest request, HttpServletResponse response) throws
    IOException, ServletException;
    //注意访问权限定义成 protected，显得既专业，又严谨，因为它是专门给子类用的
}

public class MyServlet1 extends BaseServlet
{
    protected void doService(HttpServletRequest request, HttpServletResponse response) throws IOException, ServletException
    {
        本 Servlet 只处理的具体业务逻辑代码
    }
}
```

父类方法中间的某段代码不确定，留给子类干，就用模板方法设计模式。

备注：这道题的思路是先从总体解释抽象类和接口的基本概念，然后再比较两者的语法细节，最后再说两者的应用区别。比较两者语法细节区别的条理是：先从一个类中的构造方法、普通成员变量和方法（包括抽象方法），静态变量和方法，继承性等 6 个方面逐一去比较回答，接着从第三者继承的角度的回答，特别是最后用了一个典型的例子来展现自己深厚的技术功底。

## 25、`abstract` 的 `method` 是否可同时是 `static`,是否可同时是 `native`, 是否可同时是 `synchronized`?

`abstract` 的 `method` 不可以是 `static` 的，因为抽象的方法是要被子类实现的，而 `static` 与子类扯不上关系！

`native` 方法表示该方法要用另外一种依赖平台的编程语言实现的，不存在着被子类实现的问题，所以，它也不能是抽象的，不能与 `abstract` 混用。例如，

---

`FileOutputStream` 类要硬件打交道，底层的实现用的是操作系统相关的 api 实现，例如，在 windows 用 c 语言实现的，所以，查看 jdk 的源代码，可以发现 `FileOutputStream` 的 `open` 方法的定义如下：

```
private native void open(String name) throws FileNotFoundException;
```

如果我们要用 java 调用别人写的 c 语言函数，我们是无法直接调用的，我们需要按照 java 的要求写一个 c 语言的函数，又我们的这个 c 语言函数去调用别人的 c 语言函数。由于我们的 c 语言函数是按 java 的要求来写的，我们这个 c 语言函数就可以与 java 对接上，java 那边的对接方式就是定义出与我们这个 c 函数相对应的方法，java 中对应的方法不需要写具体的代码，但需要在前面声明 `native`。

关于 `synchronized` 与 `abstract` 合用的问题，我觉得也不行，因为在我几年的学习和开发中，从来没见到过这种情况，并且我觉得 `synchronized` 应该是作用在一个具体的方法上才有意义。而且，方法上的 `synchronized` 同步所使用的同步锁对象是 `this`，而抽象方法上无法确定 `this` 是什么。

## 26、什么是内部类？`Static Nested Class` 和 `Inner Class` 的不同。

内部类就在一个类的内部定义的类，内部类中不能定义静态成员（静态成员不是对象的特性，只是为了找一个容身之处，所以需要放到一个类中而已，这么一点小事，你还要把它放到类内部的一个类中，过分了啊！提供内部类，不是为让你干这种事情，无聊，不让你干。我想可能是既然静态成员类似 c 语言的全局变量，而内部类通常是用于创建内部对象用的，所以，把“全局变量”放在内部类中就是毫无意义的事情，既然是毫无意义的事情，就应该被禁止），内部类可以直接访问外部类中的成员变量，内部类可以定义在外部类的方法外面，也可以定义在外部类的方法体中，如下所示：

```
public class Outer
{
    int out_x = 0;
    public void method()
    {
        Inner1 inner1 = new Inner1();
        public class Inner2 //在方法体内部定义的内部类
        {
            public method()
            {
                out_x = 3;
            }
        }
        Inner2 inner2 = new Inner2();
    }

    public class Inner1 //在方法体外面定义的内部类
    {
    }
}
```

在方法体外面定义的内部类的访问类型可以是 `public, protected, private` 等 4 种类型，这就好像类中定义的成员变量有 4 种访问类型一样，它们决定这个内部类的定义对其他类是否可见；对于这种情况，我们也可以在外面创建内部类的实例对象，创建内部类的实例对象时，一定要先创建外部类的实例对象，然后用这个外部类的实例对象去创建内部类的实例对象，代码如下：

```
Outer outer = new Outer();
Outer.Inner1 inner1 = outer.new Innner1();
```

在方法内部定义的内部类前面不能有访问类型修饰符，就好像方法中定义的局部变量一样，但这种内部类的前面可以使用 `final` 或 `abstract` 修饰符。这种内部类对其他类是不可见的，其他类无法引用这种内部类，但是这种内部类创建的实例对象可以传递给其他类访问。这种内部类必须是先定义，后使用，即内部类的定义代码必须出现在使用该类之前，这与方法中的局部变量必须先定义后使用的道理也是一样的。这种内部类可以访问方法体中的局部变量，但是，该局部变量前必须加 `final` 修饰符。

对于这些细节，只要在 eclipse 写代码试试，根据开发工具提示的各类错误信息就可以马上了解到。

在方法体内部还可以采用如下语法来创建一种匿名内部类，即定义某一接口或类的子类的同时，还创建了该子类的实例对象，无需为该子类定义名称：

```
public class Outer
{
    public void start()
    {
        new Thread(
            new Runnable(){
                public void run(){}
            }
        ).start();
    }
}
```

最后，在方法外部定义的内部类前面可以加上 static 关键字，从而成为 Static Nested Class，它不再具有内部类的特性，所有，从狭义上讲，它不是内部类。Static Nested Class 与普通类在运行时的行为和功能上没有什么区别，只是在编程引用时的语法上有一些差别，它可以定义成 public、protected、默认的、private 等多种类型，而普通类只能定义成 public 和默认的这两种类型。在外面引用 Static Nested Class 类的名称为“外部类名.内部类名”。在外面不需要创建外部类的实例对象，就可以直接创建 Static Nested Class，例如，假设 Inner 是定义在 Outer 类中的 Static Nested Class，那么可以使用如下语句创建 Inner 类：

```
Outer.Inner inner = new Outer.Inner();
```

由于 static Nested Class 不依赖于外部类的实例对象，所以，static Nested Class 能访问外部类的非 static 成员变量。当在外部类中访问 Static Nested Class 时，可以直接使用 Static Nested Class 的名字，而不需要加上外部类的名字了，在 Static Nested Class 中也可以直接引用外部类的 static 的成员变量，不需要加上外部类的名字。

在静态方法中定义的内部类也是 Static Nested Class，这时候不能在类前面加 static 关键字，静态方法中的 Static Nested Class 与普通方法中的内部类的应用方式很相似，它除了可以直接访问外部类中的 static 的成员变量，还可以访问静态方法中的局部变量，但是，该局部变量前必须加 final 修饰符。

备注：首先根据你的印象说出你对内部类的总体方面的特点：例如，在两个地方可以定义，可以访问外部类的成员变量，不能定义静态成员，这是大的特点。然后再谈一些细节方面的知识，例如，几种定义方式的语法区别，静态内部类，以及匿名内部类。

## 27、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()
        {
            sysout(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。

## 28、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！

## 29、`super.getClass()`方法调用

下面程序的输出结果是多少？

```
import java.util.Date;
public class Test extends Date{
    public static void main(String[] args) {
        new Test().test();
    }

    public void test(){
        System.out.println(super.getClass().getName());
    }
}
```

很奇怪，结果是 `Test`

---

这属于脑筋急转弯的题目，在一个 `qq` 群有个网友正好问过这个问题，我觉得挺有趣，就研究了一下，没想到今天还被你面到了，哈哈。

在 `test` 方法中，直接调用 `getClass().getName()` 方法，返回的是 `Test` 类名

由于 `getClass()` 在 `Object` 类中定义成了 `final`，子类不能覆盖该方法，所以，在

`test` 方法中调用 `getClass().getName()` 方法，其实就是在调用从父类继承的 `getClass()` 方法，等效于调用 `super.getClass().getName()` 方法，所以，`super.getClass().getName()` 方法返回的也应该是 `Test`。

如果想得到父类的名称，应该用如下代码：

```
getClass().getSuperClass().getName();
```

## 30、`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

## 31、`String s = "Hello";s = s + " world!"`;这两行代码执行后，原始的 `String` 对象中的内容到底变了没有？

没有。因为 `String` 被设计成不可变 (`immutable`) 类，所以它的所有对象都是不可变对象。在这段代码中，`s` 原先指向一个 `String` 对象，内容是 `"Hello"`，然后我们对 `s` 进行了 `+操作`，那么 `s` 所指向的那个对象是否发生了改变呢？答案是没有。这时，

---

s 不指向原来那个对象了，而指向了另一个 String 对象，内容为 "Hello world!"，原来那个对象还存在于内存之中，只是 s 这个引用变量不再指向它了。

通过上面的说明，我们很容易导出另一个结论，如果经常对字符串进行各种各样的修改，或者说，不可预见的修改，那么使用 String 来代表字符串的话会引起很大的内存开销。因为 String 对象建立之后不能再改变，所以对于每一个不同的字符串，都需要一个 String 对象来表示。这时，应该考虑使用 StringBuffer 类，它允许修改，而不是每个不同的字符串都要生成一个新的对象。并且，这两种类的对象转换十分容易。

同时，我们还可以知道，如果要使用内容相同的字符串，不必每次都 new 一个 String。例如我们要在构造器中对一个名叫 s 的 String 引用变量进行初始化，把它设置为初始值，应当这样做：

```
public class Demo {  
    private String s;  
    ...  
    public Demo {  
        s = "Initial Value";  
    }  
    ...  
}
```

而非

```
s = new String("Initial Value");
```

后者每次都会调用构造器，生成新对象，性能低下且内存开销大，并且没有意义，因为 String 对象不可改变，所以对于内容相同的字符串，只要一个 String 对象来表示就可以了。也就是说，多次调用上面的构造器创建多个对象，他们的 String 类型属性 s 都指向同一个对象。

上面的结论还基于这样一个事实：对于字符串常量，如果内容相同，Java 认为它们代表同一个 String 对象。而用关键字 new 调用构造器，总是会创建一个新的对象，无论内容是否相同。

至于为什么要把 String 类设计成不可变类，是它的用途决定的。其实不只 String，很多 Java 标准类库中的类都是不可变的。在开发一个系统的时候，我们有时候也需要设计不可变类，来传递一组相关的值，这也是面向对象思想的体现。不可变类有一些优点，比如因为它的对象是只读的，所以多线程并发访问也不会有任何问题。当然也有一些缺点，比如每个不同的状态都要一个对象来代表，可能会造成性能上的问题。所以 Java 标准类库还提供了一个可变版本，即 StringBuffer。

## 32、是否可以继承 String 类？

String 类是 final 类故不可以继承。

### 33、**String s = new String("xyz");**创建了几个**String Object?** 二者之间有什么区别?

两个或一个，“xyz”对应一个对象，这个对象放在字符串常量缓冲区，常量“xyz”不管出现多少遍，都是缓冲区中的那一个。New String 每写一遍，就创建一个新的对象，它一句那个常量“xyz”对象的内容来创建出一个新 String 对象。如果以前就用过‘xyz’，这句代表就不会创建“xyz”自己了，直接从缓冲区拿。

### 34、**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 StringBuffer 来动态构造字符数据。另外，String 实现了 equals 方法，new String("abc").equals(new String("abc")) 的结果为 true，而 StringBuffer 没有实现 equals 方法，所以，new StringBuffer("abc").equals(new StringBuffer("abc")) 的结果为 false。

接着要举一个具体的例子来说明，我们要把 1 到 100 的所有数字拼起来，组成一个串。

```
StringBuffer sbf = new StringBuffer();
for(int i=0;i<100;i++)
{
    sbf.append(i);
}
```

上面的代码效率很高，因为只创建了一个 StringBuffer 对象，而下面的代码效率很低，因为创建了 101 个对象。

```
String str = new String();
for(int i=0;i<100;i++)
{
    str = str + i;
}
```

在讲两者区别时，应把循环的次数搞成 10000，然后用 endTime - beginTime 来比较两者执行的时间差异，最后还要讲讲 StringBuilder 与 StringBuffer 的区别。

String 覆盖了 equals 方法和 hashCode 方法，而 StringBuffer 没有覆盖 equals 方法和 hashCode 方法，所以，将 StringBuffer 对象存储进 Java 集合类中时会出现问题。

### 35、如何把一段逗号分割的字符串转换成一个数组?

如果不查 jdk api，我很难写出来！我可以说说我的思路：

```
1. 用正则表达式，代码大概为：String [] result = orgStr.split(",");  
2. 用 StringTokenizer，代码为：StringTokenizer token = StringTokenizer(orgStr, ",");  
String [] result = new String[token.countTokens()];  
Int i=0;  
while(token.hasNext()) {result[i++]=token.nextToken();}
```

### 36、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。

### 37、下面这条语句一共创建了多少个对象： String s="a"+ "b" + "c" + "d";

答：对于如下代码：

```
String s1 = "a";  
String s2 = s1 + "b";  
String s3 = "a" + "b";  
System.out.println(s2 == "ab");  
System.out.println(s3 == "ab");
```

第一条语句打印的结果为 false，第二条语句打印的结果为 true，这说明 javac 编译可以对字符串常量直接相加的表达式进行优化，不必要等到运行期去进行加法运算处理，而是在编译时去掉其中的加号，直接将其编译成一个这些常量相连的结果。

题目中的第一行代码被编译器在编译时优化后，相当于直接定义了一个“abcd”的字符串，所以，上面的代码应该只创建了一个 String 对象。写如下两行代码，

```
String s = "a" + "b" + "c" + "d";  
System.out.println(s == "abcd");
```

最终打印的结果应该为 true。

### 38、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？

也许你的答案是在 return 之前，但往更细地说，我的答案是在 return 中间执行，请看下面程序代码的运行结果：

```
public class Test {  
  
    /**  
     * @param args add by zxx ,Dec 9, 2008  
     */  
    public static void main(String[] args) {  
        // TODO Auto-generated method stub  
        System.out.println(new Test().test());  
    }  
  
    static int test()  
    {
```

```
int x = 1;
try
{
    return x;
}
finally
{
    ++x;
}
}

-----执行结果 -----
1
```

运行结果是 1，为什么呢？主函数调用子函数并得到结果的过程，好比主函数准备一个空罐子，当子函数要返回结果时，先把结果放在罐子里，然后再将程序逻辑返回到主函数。所谓返回，就是子函数说，我不运行了，你主函数继续运行吧，这没什么结果可言，结果是在说这话之前放进罐子里的。

### 39、下面的程序代码输出的结果是多少？

```
public class smallT
{
    public static void main(String args[])
    {
        smallT t = new smallT();
        int b = t.get();
        System.out.println(b);
    }

    public int get()
    {
        try
        {
            return 1 ;
        }
        finally
        {
            return 2 ;
        }
    }
}
```

返回的结果是 2。

我可以通过下面一个例子程序来帮助我解释这个答案，从下面例子的运行结果中可以发现，try 中的 return 语句调用的函数先于 finally 中调用的函数执行，也就是说 return 语句先执行，finally 语句后执行，所以，返回的结果是 2。Return 并不是让函数马上返回，而是 return 语句执行后，将把返回结果放置进函数栈中，此时函数并不是马上返回，它要执行 finally 语句后才真正开始返回。

在讲解答案时可以用下面的程序来帮助分析：

```
public class Test {

    /**
     * @param args add by zxx ,Dec 9, 2008
     */
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        System.out.println(new Test().test());
    }

    int test()
    {
        try
        {
            return func1();
        }
        finally
        {
            return func2();
        }
    }

    int func1()
    {
        System.out.println("func1");
        return 1;
    }
    int func2()
    {
        System.out.println("func2");
        return 2;
    }
}
```

-----执行结果-----

```
func1
func2
2
```

结论：finally 中的代码比 return 和 break 语句后执行

#### 40、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，例如，一段代码.....

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

#### 41、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。

#### 42、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。

#### 43、Java 中的异常处理机制的简单原理和应用。

异常是指 java 程序运行时（非编译）所发生的非正常情况或错误，与现实生活中的事件很相似，现实生活中的事件可以包含事件发生的时间、地点、人物、情节等信息，可以用一个对象来表示，Java 使用面向对象的方式来处理异常，它把程序中发生的每个异常也都分别封装到一个对象来表示的，该对象中包含有异常的信息。

Java 对异常进行了分类，不同类型的异常分别用不同的 Java 类表示，所有异常的根类为 java.lang.Throwable，下面又派生了两个子类：Error 和 Exception，Error 表示应用程序本身无法克服和恢复的一种严重问题，程序只有死了，例如，说内存溢出和线程死锁等系统问题。Exception 表示程序还能够克服和恢复的问题，其中又分为系统异常和普通异常，系统异常是软件本身缺陷所导致的问题，也就是软件开发人员考虑不周所导致的问题，软件使用者无法克服和恢复这种问题。

但在这种问题下还可以让软件系统继续运行或者让软件死掉，例如，数组脚本越界（`ArrayIndexOutOfBoundsException`），空指针异常（`NullPointerException`）类转换异常（`ClassCastException`）；普通异常是运行环境的变化或异常所导致的问题，是用户能够克服的问题，例如，网络断线，硬盘空间不够，发生这样的异常后，程序不应该死掉。

java 为系统异常和普通异常提供了不同的解决方案，编译器强制普通异常必须 `try..catch` 处理或用 `throws` 声明继续抛给上层调用方法处理，所以普通异常也称为 `checked` 异常，而系统异常可以处理也可以不处理，所以，编译器不强制用 `try..catch` 处理或用 `throws` 声明，所以系统异常也称为 `unchecked` 异常。

提示答题者：就按照三个级别去思考：虚拟机必须宕机的错误，程序可以死掉也可以不死掉的错误，程序不应该死掉的错误；

#### 44、请写出你最常见到的 5 个 runtime exception。

这道题主要考你的代码量到底多大，如果你长期写代码的，应该经常都看到过一些系统方面的异常，你不一定真要回答出 5 个具体的系统异常，但你要能够说出什么是系统异常，以及几个系统异常就可以了，当然，这些异常完全用其英文名称来写是最好的，如果实在写不出，那就用中文吧，有总比没有强！

所谓系统异常，就是……，它们都是 `RuntimeException` 的子类，在 `jdk doc` 中查 `RuntimeException` 类，就可以看到其所有的子类列表，也就是看到了所有的系统异常。我比较有印象的系统异常有：`NullPointerException`、`ArrayIndexOutOfBoundsException`、`ClassCastException`。

#### 45、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？

#### 46、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？

java5 以前，有如下两种：

第一种：

`new Thread(){}.start();` 这表示调用 `Thread` 子类对象的 `run` 方法，`new Thread(){}.` 表示一个 `Thread` 的匿名子类的实例对象，子类加上 `run` 方法后的代码如下：

```
new Thread(){  
    public void run(){  
    }  
}.start();
```

第二种：

`new Thread(new Runnable(){}).` start(); 这表示调用 `Thread` 对象接受的 `Runnable` 对象的 `run` 方法，`new Runnable(){}.` 表示一个 `Runnable` 的匿名子类的实例对象，`Runnable` 的子类加上 `run` 方法后的代码如下：

```
new Thread(new Runnable(){  
    public void run(){  
    }  
}).start();
```

```
public void run() {  
}  
}  
.start();
```

从 java5 开始，还有如下一些线程池创建多线程的方式：

```
ExecutorService pool = Executors.newFixedThreadPool(3)  
for(int i=0;i<10;i++)  
{  
    pool.execute(new Runnable(){public void run(){}});  
}  
Executors.newCachedThreadPool().execute(new Runnable(){public void run(){}});  
Executors.newSingleThreadExecutor().execute(new Runnable(){public void run(){}});
```

有两种实现方法，分别使用 new Thread() 和 new Thread(runnable) 形式，第一种直接调用 thread 的 run 方法，所以，我们往往使用 Thread 子类，即 new SubThread()。第二种调用 runnable 的 run 方法。

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。

#### 47、sleep() 和 wait() 有什么区别？

( 网上的答案：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。)

sleep 就是正在执行的线程主动让出 cpu , cpu 去执行其他线程 , 在 sleep 指定的时间过后 , cpu 才会回到这个线程上继续往下执行 , 如果当前线程进入了同步锁 , sleep 方法并不会释放锁 , 即使当前线程使用 sleep 方法让出了 cpu , 但其他被同步锁挡住了的线程也无法得到执行。 wait 是指在一个已经进入了同步锁的线程内 , 让自己暂时让出同步锁 , 以便其他正在等待此锁的线程可以得到同步锁并运行 , 只有其他线程调用了 notify 方法 ( notify 并不释放锁 , 只是告诉调用过 wait 方法的线程可以去参与获得锁的竞争了 , 但不是马上得到锁 , 因为锁还在别人手里 , 别人还没释放 ) 。如果 notify 方法后面的代码还有很多 , 需要这些代码执行完后才会释放锁 , 可以在 notify 方法后增加一个等待和一些代码 , 看看效果 ) , 调用 wait 方法的线程就会解除 wait 状态和程序可以再次得到锁后继续向下运行。对于 wait 的讲解一定要配合例子代码来说明 , 才显得自己真明白。

```
package com.huawei.interview;

public class MultiThread {

    /**
     * @param args
     */
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        new Thread(new Thread1()).start();
        try {
            Thread.sleep(10);
        } catch (InterruptedException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        new Thread(new Thread2()).start();
    }

    private static class Thread1 implements Runnable
    {

        @Override
        public void run() {
            // TODO Auto-generated method stub
        }
    }
}

//由于这里的Thread1和下面的Thread2内部run方法要用同一对象作为监视器，我们这里不能用this，因为在Thread2里面的this和这个Thread1的this不是同一个对象。我们用MultiThread.class这个字节码对象，当前虚拟机里引用这个变量时，指向的都是同一个对象。
synchronized (MultiThread.class) {

    System.out.println("enter thread1...");

    System.out.println("thread1 is waiting");
    try {

```

//释放锁有两种方式，第一种方式是程序自然离开监视器的范围，也就是离开了 synchronized关键字管辖的代码范围，另一种方式就是在synchronized关键字管辖的代码内部调用监视器对象的wait方法。这里，使用wait方法释放锁。

```
        MultiThread.class.wait();
    } catch (InterruptedException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }

    System.out.println("thread1 is going on...");
    System.out.println("thread1 is being over!");
}
}

private static class Thread2 implements Runnable
{

    @Override
    public void run() {
        // TODO Auto-generated method stub
        synchronized (MultiThread.class) {

            System.out.println("enter thread2...");

            System.out.println("thread2 notify other thread can release wait status..");
//由于notify方法并不释放锁，即使thread2调用下面的sleep方法休息了10毫秒，但thread1仍然不会执行，因为thread2没有释放锁，所以
Thread1无法得不到锁。

            MultiThread.class.notify();

            System.out.println("thread2 is sleeping ten millisecond...");
            try {
                Thread.sleep(10);
            } catch (InterruptedException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }

            System.out.println("thread2 is going on...");
            System.out.println("thread2 is being over!");

        }
    }
}
```

**48、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。

**49. 下面两个方法同步吗？（自己发明）**

```
class Test
{
    synchronized static void sayHello3()
    {
    }

    synchronized void getX(){}
}
```

**50、多线程有几种实现方法?同步有几种实现方法?**

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

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。

Allnotify():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。

**51、启动一个线程是用 run()还是 start()？**

启动一个线程是调用 start() 方法，使线程就绪状态，以后可以被调度为运行状态，一个线程必须关联一些具体的执行代码，run() 方法是该线程所关联的执行代码。

## 52、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**。

## 53、线程的基本概念、线程的基本状态以及状态之间的关系

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。

状态：就绪，运行，**synchronize** 阻塞，**wait** 和 **sleep** 挂起，结束。**wait** 必须在 **synchronized** 内部调用。

调用线程的 **start** 方法后线程进入就绪状态，线程调度系统将就绪状态的线程转为运行状态，遇到 **synchronized** 语句时，由运行状态转为阻塞，当 **synchronized** 获得锁后，由阻塞转为运行，在这种情况下可以调用 **wait** 方法转为挂起状态，当线程关联的代码执行完后，线程变为结束状态。

## 54、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。**Lock** 还有更强大的功能，例如，它的 **tryLock** 方法可以非阻塞方式去拿锁。

举例说明（对下面的题用 **lock** 进行了改写）：

```
package com.huawei.interview;
```

```
import java.util.concurrent.locks.Lock;
import java.util.concurrent.locks.ReentrantLock;

public class ThreadTest {

    /**
     * @param args
     */

    private int j;
    private Lock lock = new ReentrantLock();
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        ThreadTest tt = new ThreadTest();
        for(int i=0;i<2;i++)
        {
            new Thread(tt.new Adder()).start();
            new Thread(tt.new Subtractor()).start();
        }
    }

    private class Subtractor implements Runnable
    {

        @Override
        public void run() {
            // TODO Auto-generated method stub
            while(true)
            {
                /*synchronized (ThreadTest.this) {
                    System.out.println("j---" + j--);
                    //这里抛异常了，锁能释放吗？
                }*/
                lock.lock();
                try
                {
                    System.out.println("j---" + j--);
                }finally
                {
                    lock.unlock();
                }
            }
        }
    }

    private class Adder implements Runnable
    {
```

```
@Override
public void run() {
    // TODO Auto-generated method stub
    while(true)
    {
        /*synchronized (ThreadTest.this) {
        System.out.println("j++=" + j++);
        }*/
        lock.lock();
        try
        {
            System.out.println("j++=" + j++);
        }finally
        {
            lock.unlock();
        }
    }
}
```

**55、设计 4 个线程，其中两个线程每次对 j 增加 1，另外两个线程对 j 每次减少 1。写出程序。**

以下程序使用内部类实现线程，对 j 增减的时候没有考虑顺序问题。

```
public class ThreadTest1
{
private int j;
public static void main(String args[]){
    ThreadTest1 tt=new ThreadTest1();
    Inc inc=tt.new Inc();
    Dec dec=tt.new Dec();
    for(int i=0;i<2;i++){
        Thread t=new Thread(inc);
        t.start();
        t=new Thread(dec);
        t.start();
    }
}
private synchronized void inc(){
    j++;
    System.out.println(Thread.currentThread().getName()+"-inc:"+j);
}
private synchronized void dec(){
    j--;
}
```

```
System.out.println(Thread.currentThread().getName()+"-dec:"+j);  
}  
class Inc implements Runnable{  
    public void run(){  
        for(int i=0;i<100;i++){  
            inc();  
        }  
    }  
}  
class Dec implements Runnable{  
    public void run(){  
        for(int i=0;i<100;i++){  
            dec();  
        }  
    }  
}  
}
```

-----随手再写的一个-----

```
class A  
{  
    JManger j =new JManger();  
    main()  
    {  
        new A().call();  
    }  
  
    void call  
    {  
        for(int i=0;i<2;i++)  
        {  
            new Thread(  
                new Runnable(){ public void run(){while(true){j.accumulate()}}}  
            ).start();  
            new Thread(new Runnable(){ public void run(){while(true){j.sub()}}}).start();  
        }  
    }  
}  
  
class JManger  
{
```

```
private j = 0;

public synchronized void subtract()
{
    j--;
}

public synchronized void accumulate()
{
    j++;
}

}
```

**56、子线程循环 10 次，接着主线程循环 100，接着又回到子线程循环 10 次，接着再回到主线程又循环 100，如此循环 50 次，请写出程序。**

最终的程序代码如下：

```
public class ThreadTest {

    /**
     * @param args
     */
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        new ThreadTest().init();
    }

    public void init()
    {
        final Business business = new Business();
        new Thread(
            new Runnable()
            {

                public void run() {
                    for(int i=0;i<50;i++)
                    {
                        business.SubThread(i);
                    }
                }
            }
        );
    }
}
```

```
    ).start();

    for(int i=0;i<50;i++)
    {
        business.MainThread(i);
    }
}

private class Business
{
    boolean bShouldSub = true;//这里相当于定义了控制该谁执行的一个信号灯
    public synchronized void MainThread(int i)
    {
        if(bShouldSub)
            try {
                this.wait();
            } catch (InterruptedException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }

        for(int j=0;j<5;j++)
        {
            System.out.println(Thread.currentThread().getName() + ":i=" + i +",j=" + j);
        }
        bShouldSub = true;
        this.notify();
    }

    public synchronized void SubThread(int i)
    {
        if(!bShouldSub)
            try {
                this.wait();
            } catch (InterruptedException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }

        for(int j=0;j<10;j++)
        {
            System.out.println(Thread.currentThread().getName() + ":i=" + i +",j=" + j);
        }
        bShouldSub = false;
        this.notify();
    }
}
```

```
    }  
}
```

备注：不可能一上来就写出上面的完整代码，最初写出来的代码如下，问题在于两个线程的代码要参照同一个变量，即这两个线程的代码要共享数据，所以，把这两个线程的执行代码搬到同一个类中去：

```
package com.huawei.interview.lym;  
  
public class ThreadTest {  
  
    private static boolean bShouldMain = false;  
  
    public static void main(String[] args) {  
        // TODO Auto-generated method stub  
        /*new Thread() {  
            public void run()  
            {  
                for(int i=0;i<50;i++)  
                {  
                    for(int j=0;j<10;j++)  
                    {  
                        System.out.println("i=" + i + ",j=" + j);  
                    }  
                }  
            }  
        }.start();*/  
  
        //final String str = new String("");  
  
        new Thread(  
            new Runnable()  
            {  
                public void run()  
                {  
                    for(int i=0;i<50;i++)  
                    {  
                        synchronized (ThreadTest.class) {  
                            if(bShouldMain)  
                            {  
                                try {  
                                    ThreadTest.class.wait();  
                                catch (InterruptedException e) {  
                                    e.printStackTrace();  
                                }  
                            }  
                            for(int j=0;j<10;j++)  
                            {  
                                System.out.println("i=" + i + ",j=" + j);  
                            }  
                        }  
                    }  
                }  
            }  
        ).start();  
    }  
}
```

```

        System.out.println(
            Thread.currentThread().getName() +
            "i=" + i + ",j=" + j);
    }
    bShouldMain = true;
    ThreadTest.class.notify();
}
}
)
.start();

for(int i=0;i<50;i++)
{
    synchronized (ThreadTest.class) {
        if(!bShouldMain)
        {
            try {
                ThreadTest.class.wait();
            catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
        for(int j=0;j<5;j++)
        {
            System.out.println(
                Thread.currentThread().getName() +
                "i=" + i + ",j=" + j);
        }
        bShouldMain = false;
        ThreadTest.class.notify();
    }
}
}

}

```

下面使用 jdk5 中的并发库来实现的：

```

import java.util.concurrent.Executors;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.locks.Lock;
import java.util.concurrent.locks.ReentrantLock;
import java.util.concurrent.locks.Condition;

public class ThreadTest
{
    private static Lock lock = new ReentrantLock();
    private static Condition subThreadCondition = lock.newCondition();

```

```
private static boolean bBshouldSubThread = false;
public static void main(String [] args)
{
    ExecutorService threadPool = Executors.newFixedThreadPool(3);
    threadPool.execute(new Runnable() {
        public void run()
        {
            for(int i=0;i<50;i++)
            {
                lock.lock();
                try
                {
                    if(!bBshouldSubThread)
                        subThreadCondition.await();
                    for(int j=0;j<10;j++)
                    {
                        System.out.println(Thread.currentThread().getName() + ",j=" + j);
                    }
                    bBshouldSubThread = false;
                    subThreadCondition.signal();
                }catch(Exception e)
                {
                }
            finally
            {
                lock.unlock();
            }
        }
    });
}

});  
threadPool.shutdown();
for(int i=0;i<50;i++)
{
    lock.lock();
    try
    {
        if(bBshouldSubThread)
            subThreadCondition.await();
        for(int j=0;j<10;j++)
        {
```

```
        System.out.println(Thread.currentThread().getName() + ", j=" + j);
    }
    bShouldSubThread = true;
    subThreadCondition.signal();
} catch (Exception e)
{
}
finally
{
    lock.unlock();
}
}
}
```

## 57、介绍 Collection 框架的结构

答：随意发挥题，天南海北谁便谈，只要让别觉得你知识渊博，理解透彻即可。

## 58、Collection 框架中实现比较要实现什么接口

comparable/comparator

## 59、ArrayList 和 Vector 的区别

答：

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，也就是说它的方法之间是线程同步的，而 ArrayList 是线程不安全的，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。

## 60、HashMap 和 Hashtable 的区别

（条理上还需要整理，也是先说相同点，再说不同点）

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，在只有一个线程访问的情况下，效率要高于 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 Synchronized 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。

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

## 61、List 和 Map 区别？

一个是存储单列数据的集合，另一个是存储键和值这样的双列数据的集合，List 中存储的数据是有顺序，并且允许重复；Map 中存储的数据是没有顺序的，其键是不能重复的，它的值是可以有重复的。

## 62、List, Set, Map 是否继承自 Collection 接口？

List, Set 是，Map 不是

## 63、List、Map、Set 三个接口，存取元素时，各有什么特点？

这样的题属于随意发挥题：这样的题比较考水平，两个方面的水平：一是要真正明白这些内容，二是要有较强的总结和表述能力。如果你明白，但表述不清楚，在别人那里则等同于不明白。

首先，List 与 Set 具有相似性，它们都是单列元素的集合，所以，它们有一个共同的父接口，叫 Collection。Set 里面不允许有重复的元素，所谓重复，即不能有两个相等（注意，不是仅仅是相同）的对象，即假设 Set 集合中有了一个 A 对象，现在我要向 Set 集合再存入一个 B 对象，但 B 对象与 A 对象 equals 相等，则 B 对象存储不进去，所以，Set 集合的 add 方法有一个 boolean 的返回值，当集合中没有某个元素，此时 add 方法可成功加入该元素时，则返回 true，当集合含有与某个元素 equals 相等的元素时，此时 add 方法无法加入该元素，返回结果为 false。Set 取元素时，没法说取第几个，只能以 Iterator 接口取得所有的元素，再逐一遍历各个元素。

List 表示有先后顺序的集合，注意，不是那种按年龄、按大小、按价格之类的排序。当我们多次调用 add(Obj e) 方法时，每次加入的对象就像火车站买票有排队顺序一样，按先来后到的顺序排序。有时候，也可以插队，即调用 add(int index, Obj e) 方法，就可以指定当前对象在集合中的存放位置。一个对象可以被反复存储进 List 中，每调用一次 add 方法，这个对象就被插入进集合中一次，其实，并不是把这个对象本身存储进了集合中，而是在集合中用一个索引变量指向这个对象，当这个对象被 add 多次时，即相当于集合中有多个索引指向了这个对象，如图 x 所示。List 除了可以以 Iterator 接口取得所有的元素，再逐一遍历各个元素之外，还可以调用 get(index i) 来明确说明取第几个。

Map 与 List 和 Set 不同，它是双列的集合，其中有 put 方法，定义如下：put(obj key, obj value)，每次存储时，要存储一对 key/value 不能存储重复的 key 这个重复的规则也是按 equals 比较相等。取则可以根据 key 获得相应的 value，即 get(Object key) 返回值为 key 所对应的 value。另外，也可以获得所有的 key 的结合，还可以获得所有的 value 的结合，还可以获得 key 和 value 组合成的 Map.Entry 对象的集合。

---

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。

HashSet 按照 hashCode 值的某种运算方式进行存储，而不是直接按 hashCode 值的大小进行存储。例如，"abc" ---> 78, "def" ---> 62, "xyz" ---> 65 在 HashSet 中的存储顺序不是 62, 65, 78，这些问题感谢以前一个叫崔健的学员提出，最后通过查看源代码给他解释清楚，看本次培训学员当中有多少能看懂源码。LinkedHashSet 按插入的顺序存储，那被存储对象的 hashCode 方法还有什么作用呢？学员想想！HashSet 集合比较两个对象是否相等，首先看 hashCode 方法是否相等，然后看 equals 方法是否相等。new 两个 Student 插入到 HashSet 中，看 HashSet 的 size，实现 hashCode 和 equals 方法后再看 size。

同一个对象可以在 Vector 中加入多次。往集合里面加元素，相当于集合里用一根绳子连接到了目标对象。往 HashSet 中却加不了多次的。

#### 64、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。

LinkedList 也是线程不安全的，LinkedList 提供了一些方法，使得 LinkedList 可以被当作堆栈和队列来使用。

#### 65、去掉一个 Vector 集合中重复的元素

```
Vector newVector = new Vector();
For (int i=0;i<vector.size();i++)
{
    Object obj = vector.get(i);
    if (!newVector.contains(obj));
        newVector.add(obj);
}
```

还有一种简单的方式，`HashSet set = new HashSet(vector);`

## 66、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。

## 67、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。

## 68、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。

ArrayList/Vector → List  
→ Collection

HashSet/TreeSet → Set

Properties → HashTable  
→ Map  
TreeMap/HashMap

我记的不是方法名，而是思想，我知道它们都有增删改查的方法，但这些方法的具体名称，我记得不是很清楚，对于 set，大概的方法是 add, remove, contains；对于 map，大概的方法就是 put, remove, contains 等，因为，我只要在 eclipse 下按点操作符，很自然的这些方法就出来了。我记住的一些思想就是 List 类会有 get(int index) 这样的方法，因为它可以按顺序取元素，而 set 类中没有 get(int index) 这样的方法。List 和 set 都可以迭代出所有元素，迭代时先要得到一个 iterator 对象，所以，set 和 list 类都有一个 iterator 方法，用于返回那个 iterator 对象。map 可以返回三个集合，一个是返回所有的 key 的集合，另外一个返回的是所有 value 的集合，再一个返回的 key 和 value 组合成的 EntrySet 对象的集合，map 也有 get 方法，参数是 key，返回值是 key 对应的 value。

## 69、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？

对。

如果对象要保存在 `HashSet` 或 `HashMap` 中，它们的 `equals` 相等，那么，它们的 `hashCode` 值就必须相等。

如果不是要保存在 `HashSet` 或 `HashMap` 则与 `hashCode` 没有什么关系了。这时候 `hashCode` 不等是可以的，例如 `ArrayList` 存储的对象就不用实现 `hashCode`，当然，我们没有理由不实现，通常都会去实现的。

## 70、`TreeSet` 里面放对象，如果同时放入了父类和子类的实例对象，那比较时使用的是父类的 `compareTo` 方法，还是使用的子类的 `compareTo` 方法，还是抛异常！

( 应该是没有针对问题的确切的答案，当前的 `add` 方法放入的是哪个对象，就调用哪个对象的 `compareTo` 方法，至于这个 `compareTo` 方法怎么做，就看当前这个对象的类中是如何编写这个方法的 )

实验代码：

```
public class Parent implements Comparable {
    private int age = 0;
    public Parent(int age) {
        this.age = age;
    }
    public int compareTo(Object o) {
        // TODO Auto-generated method stub
        System.out.println("method of parent");
        Parent o1 = (Parent)o;
        return age>o1.age?1:age<o1.age?-1:0;
    }
}

public class Child extends Parent {

    public Child(){
        super(3);
    }
    public int compareTo(Object o) {

        // TODO Auto-generated method stub
        System.out.println("method of child");
        Child o1 = (Child)o;
        return 1;
    }
}

public class TreeSetTest {
```

```
/*
 * @param args
 */
public static void main(String[] args) {
    // TODO Auto-generated method stub
    TreeSet set = new TreeSet();
    set.add(new Parent(3));
    set.add(new Child());
    set.add(new Parent(4));
    System.out.println(set.size());
}

}
```

### 71、说出一些常用的类，包，接口，请各举 5 个

要让人家感觉你对 java ee 开发很熟，所以，不能仅仅只列 core java 中的那些东西，要多列你在做 ssh 项目中涉及的那些东西。就写你最近写的那些程序中涉及的那些类。

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

常用的包： java.lang java.io java.util  
java.sql , javax.servlet , org.apache.struts.action , org.hibernate

常用的接口： Remote List Map Document

NodeList , Servlet , HttpServletRequest , HttpServletResponse , Transaction (Hibernate) ,  
Session (Hibernate) , HttpSession

### 72、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。

### 73、字节流与字符流的区别

要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);
}

}
```

#### 74、什么是 **java** 序列化，如何实现 **java** 序列化？或者请解释 **Serializable** 接口的作用。

我们有时候将一个 **java** 对象变成字节流的形式传出去或者从一个字节流中恢复成一个 **java** 对象，例如，要将 **java** 对象存储到硬盘或者传送给网络上的其他计算机，这个过程我们可以自己写代码去把一个 **java** 对象变成某个格式的字节流再传输，但是，**jre** 本身就提供了这种支持，我们可以调用 **OutputStream** 的 **writeObject** 方法来做，如果要让 **java** 帮我们做，要被传输的对象必须实现 **Serializable** 接口，这样，**javac** 编译时就会进行特殊处理，编译的类才可以被 **writeObject** 方法操作，这就是所谓的序列化。需要被序列化的类必须实现 **Serializable** 接口，该接口是一个 **mini** 接口，其中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。

例如，在 **web** 开发中，如果对象被保存在了 **Session** 中，**tomcat** 在重启时要把 **Session** 对象序列化到硬盘，这个对象就必须实现 **Serializable** 接口。如果对象要经过分布式系统进行网络传输或通过 **rmi** 等远程调用，这就需要在网络上传输对象，被传输的对象就必须实现 **Serializable** 接口。

#### 75、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。

## 76、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。

## 77、GC 是什么？为什么要有 GC？

GC 是垃圾收集的意思（Garbage 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。

## 78、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。

## 79、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。

## 80、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，assert 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。

```
package com.huawei.interview;

public class AssertTest {

    /**
     * @param args
     */
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        int i = 0;
        for(i=0;i<5;i++)
        {
            System.out.println(i);
        }
        //假设程序不小心多了一句--i;
        --i;
        assert i==5;
    }
}
```

## 81、java 中会存在内存泄漏吗，请简单描述。

所谓内存泄露就是指一个不再被程序使用的对象或变量一直被占据在内存中。java 中有垃圾回收机制，它可以保证一对象不再被引用的时候，即对象编程了孤儿的时候，对象将自动被垃圾回收器从内存中清除掉。由于 Java 使用有向图的方式进行垃圾回收管理，可以消除引用循环的问题，例如有两个对象，相互引用，只要它们和根进程不可达的，那么 GC 也是可以回收它们的，例如下面的代码可以看到这种情况的内存回收：

```
package com.huawei.interview;
```

```
import java.io.IOException;
```

```
public class GarbageTest {
```

```
    /**
     * @param args
     * @throws IOException
     */
```

```

public static void main(String[] args) throws IOException {
    // TODO Auto-generated method stub
    try {
        gcTest();
    } catch (IOException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
    System.out.println("has exited gcTest!");
    System.in.read();
    System.in.read();
    System.out.println("out begin gc!");
    for(int i=0;i<100;i++)
    {
        System.gc();
        System.in.read();
        System.in.read();
    }
}

private static void gcTest() throws IOException {
    System.in.read();
    System.in.read();
    Person p1 = new Person();
    System.in.read();
    System.in.read();
    Person p2 = new Person();
    p1.setMate(p2);
    p2.setMate(p1);
    System.out.println("before exit gctest!");
    System.in.read();
    System.in.read();
    System.gc();
    System.out.println("exit gctest!");
}

private static class Person
{
    byte[] data = new byte[20000000];
    Person mate = null;
    public void setMate(Person other)
    {
        mate = other;
    }
}

```

java 中的内存泄露的情况：长生命周期的对象持有短生命周期对象的引用就很可能发生内存泄露，尽管短生命周期对象已经不

再需要，但是因为长生命周期对象持有它的引用而导致不能被回收，这就是 java 中内存泄露的发生场景，通俗地说，就是程序员可能创建了一个对象，以后一直不再使用这个对象，这个对象却一直被引用，即这个对象无用但是却无法被垃圾回收器回收的，这就是 java 中可能出现内存泄露的情况，例如，缓存系统，我们加载了一个对象放在缓存中（例如放在一个全局 map 对象中），然后一直不再使用它，这个对象一直被缓存引用，但却不再被使用。

检查 java 中的内存泄露，一定要让程序将各种分支情况都完整执行到程序结束，然后看某个对象是否被使用过，如果没有，则才能判定这个对象属于内存泄露。

如果一个外部类的实例对象的方法返回了一个内部类的实例对象，这个内部类对象被长期引用了，即使那个外部类实例对象不再被使用，但由于内部类持久外部类的实例对象，这个外部类对象将不会被垃圾回收，这也会造成内存泄露。

下面内容来自于网上（主要特点就是清空堆栈中的某个元素，并不是彻底把它从数组中拿掉，而是把存储的总数减少，本人写得可以比这个好，在拿掉某个元素时，顺便也让它从数组中消失，将那个元素所在的位置的值设置为 null 即可）：

我实在想不到比那个堆栈更经典的例子了，以至于我还要引用别人的例子，下面的例子不是我想到的，是书上看到的，当然如果没有在书上看到，可能过一段时间我自己也想的到，可是那时我说是我自己想到的也没有人相信的。

```
public class Stack {  
    private Object[] elements=new Object[10];  
    private int size = 0;  
    public void push(Object e) {  
        ensureCapacity();  
        elements[size++] = e;  
    }  
    public Object pop() {  
        if( size == 0)  
  
            throw new EmptyStackException();  
        return elements[--size];  
    }  
    private void ensureCapacity() {  
        if(elements.length == size){  
            Object[] oldElements = elements;  
            elements = new Object[2 * elements.length+1];  
        }  
    }  
}
```

```
System.arraycopy(oldElements, 0, elements, 0, size);  
}  
}  
}  
}
```

上面的原理应该很简单，假如堆栈加了 10 个元素，然后全部弹出来，虽然堆栈是空的，没有我们要的东西，但是这个对象是无法回收的，这个才符合了内存泄露的两个条件：无用，无法回收。

但是就是存在这样的东西也不一定会导致什么样的后果，如果这个堆栈用的比较少，也就浪费了几个 K 内存而已，反正我们的内存都上 G 了，哪里会有什么影响，再说这个东西很快就会被回收的，有什么关系。下面看两个例子。

### 例子 1

```
public class Bad{  
    public static Stack s=Stack();  
    static{  
        s.push(new Object());  
        s.pop(); //这里有一个对象发生内存泄露  
        s.push(new Object()); //上面的对象可以被回收了，等于是自愈了  
    }  
}
```

因为是 static，就一直存在到程序退出，但是我们也可以看到它有自愈功能，就是说如果你的 Stack 最多有 100 个对象，那么最多也就只有 100 个对象无法被回收其实这个应该很容易理解，Stack 内部持有 100 个引用，最坏的情况就是他们都是无用的，因为我们一旦放新的进去，以前的引用自然消失！

内存泄露的另外一种情况：当一个对象被存储进 HashSet 集合中以后，就不能修改这个对象中的那些参与计算哈希值的字段了，否则，对象修改后的哈希值与最初存储进 HashSet 集合中时的哈希值就不同了，在这种情况下，即使在 contains 方法使用该对象的当前引用作为的参数去 HashSet 集合中检索对象，也将返回找不到对象的结果，这也会导致无法从 HashSet 集合中单独删除当前对象，造成内存泄露。

## 82、能不能自己写个类，也叫 `java.lang.String`？

可以，但在应用的时候，需要用自己的类加载器去加载，否则，系统的类加载器永远只是去加载jre.jar包中的那个`java.lang.String`。由于在tomcat的web应用程序中，都是由webapp自己的类加载器先自己加载WEB-INF/classes目录中的类，然后才委托上级的类加载器加载，如果我们在tomcat的web应用程序中写一个`java.lang.String`，这时候Servlet程序加载的就是我们自己写的`java.lang.String`，但是这么干就会出很多潜在的问题，原来所有用了`java.lang.String`类的都将出现问题。

虽然java提供了endorsed技术，可以覆盖jdk中的某些类，具体做法是....。但是，能够被覆盖的类是有限制范围，反正不包括`java.lang`这样的包中的类。

(下面的例如主要是便于大家学习理解只用，不要作为答案的一部分，否则，人家怀疑是题目泄露了) 例如，运行下面的程序：

```
package java.lang;

public class String {

    /**
     * @param args
     */
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        System.out.println("string");
    }
}
```

报告的错误如下：

```
java.lang.NoSuchMethodError: main
Exception in thread "main"
```

这是因为加载了jre自带的`java.lang.String`，而该类中没有main方法。

## 83. Java 代码查错

```
1.
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 variable (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 属于 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.java

```
class 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。

## 二. 算法与编程

1、编写一个程序，将 **a.txt** 文件中的单词与 **b.txt** 文件中的单词交替合并到 **c.txt** 文件中，**a.txt** 文件中的单词用回车符分隔，**b.txt** 文件中用回车或空格进行分隔。

答：

```
package cn.itcast;
```

```
import java.io.File;
import java.io.FileReader;
import java.io.FileWriter;

public class MainClass{
    public static void main(String[] args) throws Exception{
        FileManager a = new FileManager("a.txt",new char[]{'\n'});
        FileManager b = new FileManager("b.txt",new char[]{'\n',' '});
        FileWriter c = new FileWriter("c.txt");
        String aWord = null;
        String bWord = null;
        while((aWord = a.nextWord()) !=null ){
            c.write(aWord + "\n");
            bWord = b.nextWord();
            if(bWord != null)
                c.write(bWord + "\n");
        }

        while((bWord = b.nextWord()) != null){
            c.write(bWord + "\n");
        }
        c.close();
    }
}

class FileManager{

    String[] words = null;
    int pos = 0;
    public FileManager(String filename,char[] seperators) throws Exception{
        File f = new File(filename);
        FileReader reader = new FileReader(f);
        char[] buf = new char[(int)f.length()];
        int len = reader.read(buf);
        String results = new String(buf,0,len);
        String regex = null;
        if(seperators.length >1 ){
            regex = "" + seperators[0] + "|" + seperators[1];
        }else{

```

```
        regex = "" + separators[0];
    }
    words = results.split(regex);
}

public String nextWord(){
    if(pos == words.length)
        return null;
    return words[pos++];
}

}
```

**2、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。**

( 大家正在做上面这道题，网上迟到的朋友也请做做这道题，找工作必须能编写这些简单问题的代码 !)

**答：listFiles 方法接受一个 FileFilter 对象，这个 FileFilter 对象就是过滤的策略对象，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");
                }
            }
        );
        for (File file : files) {
            String fileName = file.getName();
            String jadName = fileName.substring(0, fileName.length() - 4) + ".jad";
            File jadFile = new File("jad/" + jadName);
            try (InputStream in = new FileInputStream(file);
                 OutputStream out = new FileOutputStream(jadFile)) {
                byte[] buffer = new byte[1024];
                int len;
                while ((len = in.read(buffer)) != -1) {
                    out.write(buffer, 0, len);
                }
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }
}
```

```

        }

    );

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 implements 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();  
}
```

3、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，但要保证汉字不被截取半个，如“我 ABC”，4，应该截取“我 AB”，输入“我 ABC 汉 DEF”，6，应该输出“我 ABC”，而不是“我 ABC+汉的半个”。

答：

首先要了解中文字符有多种编码及各种编码的特征。

假设 n 为要截取的字节数。

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

```
String str = "我 a 爱中华 abc 我爱传智 def";  
  
String str = "我 ABC 汉";  
int num = trimGBK(str.getBytes("GBK"), 5);  
System.out.println(str.substring(0, num) );  
}  
  
public static int trimGBK(byte[] buf, int n){  
    int num = 0;  
    boolean bChineseFirstHalf = false;  
    for(int i=0;i<n;i++){  
        {  
            if(buf[i]<0 && !bChineseFirstHalf){  
                bChineseFirstHalf = true;  
            }else{  
                num++;  
                bChineseFirstHalf = false;  
            }  
        }  
    }  
    return num;  
}
```

#### 4、有一个字符串，其中包含中文字符、英文字符和数字字符，请统计和打印出各个字符的个数。

答：哈哈，其实包含中文字符、英文字符、数字字符原来是出题者放的烟雾弹。

```
String content = "中国 aadf 的 111 萨 bbb 菲的 zz 萨菲";  
HashMap map = new HashMap();  
for(int i=0;i<content.length();i++)  
{  
    char c = content.charAt(i);  
    Integer num = map.get(c);  
    if(num == null)  
        num = 1;  
    else  
        num = num + 1;  
    map.put(c, num);  
}  
for(Map.EntrySet entry : map)  
{  
    system.out.println(entry.getKey() + ":" + entry.getValue());  
}
```

估计是当初面试的那个学员表述不清楚，问题很可能是：

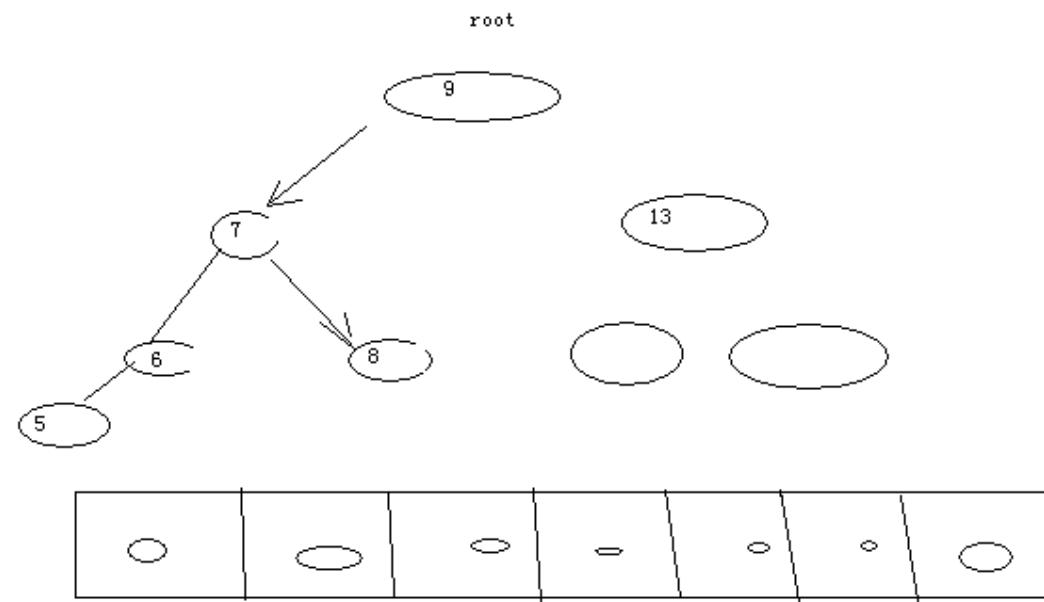
如果一串字符如“aaaabbc中国1512”要分别统计英文字母的数量，中文字母的数量，和数字字符的数量，假设字符串中没有中文字符、英文字母、数字字符之外的其他特殊字符。

```
int engishCount;
int chineseCount;
int digitCount;
for(int i=0;i<str.length;i++)
{
    char ch = str.charAt(i);
    if(ch>='0' && ch<='9')
    {
        digitCount++;
    }
    else if((ch>='a' && ch<='z') || (ch>='A' && ch<='Z'))
    {
        engishCount++;
    }
    else
    {
        chineseCount++;
    }
}
System.out.println(.....);
```

## 5、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;  
}  
  
}
```

**6、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);
        }
    }
}

```

## 7、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

## 8、递归算法题 1

一个整数，大于 0，不用循环和本地变量，按照  $n, 2n, 4n, 8n$  的顺序递增，当值大于 5000 时，把值按照指定顺序输出来。

例：n=1237

则输出为：

```
1237,  
2474,  
4948,  
9896,  
9896,
```

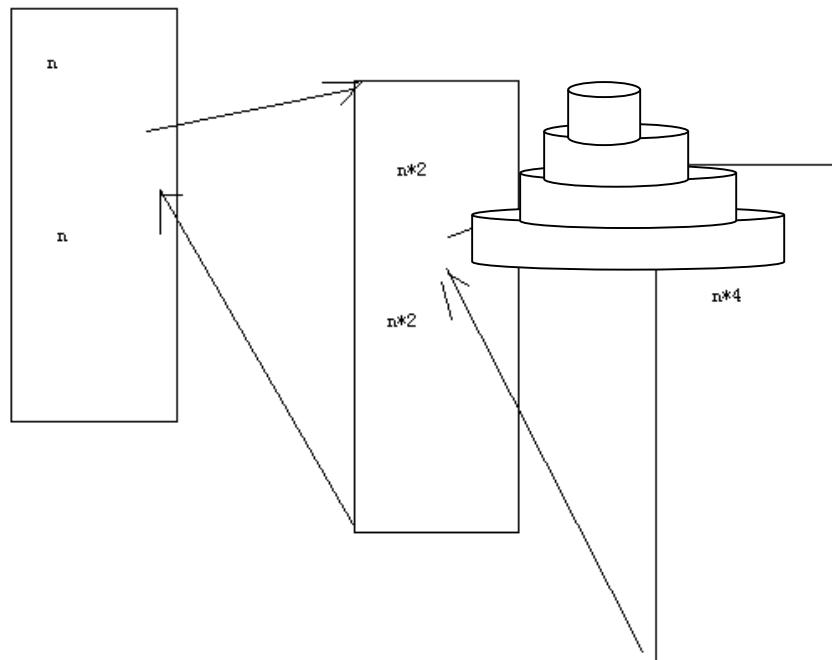
4948 ,

2474 ,

1237 ,

提示：写程序时，先致谢按递增方式的代码，写好递增的以后，再增加考虑递减部分。

```
public static void doubleNum(int n)
{
    System.out.println(n);
    if(n<=5000)
        doubleNum(n*2);
    System.out.println(n);
}
    Gaibaota(N) = Gaibaota(N-1)
```



## 9、递归算法题 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);
}
```

## 10、排序都有哪几种方法？请列举。用 JAVA 实现一个快速排序。

本人只研究过冒泡排序、选择排序和快速排序，下面是快速排序的代码：

```
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;
            }
        }while(i<=j);
    }
}
```

```
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]+" ");
}
}
}
```

## 11、有数组 a[n]，用 java 代码将数组元素顺序颠倒

```
//用下面的也可以
//for(int i=0, int j=a.length-1;i<j;i++, j--) 是否等效于 for(int i=0;i<a.length/2;i++)呢？

import java.util.Arrays;

public class SwapDemo{

    public static void main(String[] args) {
        int [] a = new int[]{
            (int)(Math.random() * 1000),
            (int)(Math.random() * 1000),
            (int)(Math.random() * 1000),
            (int)(Math.random() * 1000),
            (int)(Math.random() * 1000)
        }
    }
}
```

```
    System.out.println(a);
    System.out.println(Arrays.toString(a));
    swap(a);
    System.out.println(Arrays.toString(a));
}

public static void swap(int a[]){
    int len = a.length;
    for(int i=0;i<len/2;i++){
        int tmp = a[i];
        a[i] = a[len-1-i];
        a[len-1-i] = tmp;
    }
}
```

**12.** 金额转换，阿拉伯数字的金额转换成中国传统的形式如：（¥1011）→（一千零一拾壹元整）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', '貳', '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();
    }
}

```

### 三. html&JavaScript&ajax 部分

#### 1. 判断第二个日期比第一个日期大

如何用脚本判断用户输入的字符串是下面的时间格式 2004-11-21 必须要保证用户的输入是此格式，并且是时间，比如说月份不大于 12 等等，另外我需要用户输入两个，并且后一个要比前一个晚，只允许用 JAVASCRIPT，请详细帮助作答，，，

//这里可用正则表达式判断提前判断一下格式，然后按下提取各时间字段内容

```

<script type="text/javascript">
window.onload = function()
{
    //这么写是为了实现 js 代码与 html 代码的分离，当我修改 js 时，不能影响 html 代码。
    document.getElementById("frm1").onsubmit =
        function()
    {
        var d1 = this.d1.value;
        var d2 = this.d2.value;

        if(!verifyDate (d1)) {alert("第一个日期格式不对");return false;}
        if(!verifyDate (d2)) {alert("第二个日期格式不对");return false;}
        if(!compareDate(d1,d2)) {alert("第二个日期比第一日期小");return false;}
    };
}

function compareDate(d1,d2)
{
    var arrayD1 = d1.split("-");
    var date1 = new Date(arrayD1[0],arrayD1[1],arrayD1[2]);
    var arrayD2 = d2.split("-");
    var date2 = new Date(arrayD2[0],arrayD2[1],arrayD2[2]);
}

```

```

        if(date1 > date2) return false;
        return true;
    }

    function verifyDate(d)
    {
        var datePattern = /^(\d{4}-(0?[1-9]|1[0-2])-(0?[1-9]|[1-2]\d|3[0-1]))$/;
        return datePattern.test(d);
    }
</script>

<form id="frm1" action="xxx.html">
<input type="text" name="d1" />
<input type="text" name="d2" />
<input type="submit"/>
</form>

```

## 2. 用 **table** 显示 **n** 条记录，每 **3** 行换一次颜色，即 **1, 2, 3** 用红色字体，**4, 5, 6** 用绿色字体，**7, 8, 9** 用红颜色字体。

```

<body>
<table id="tbl">
<tr><td>1</td></tr>
<tr><td>2</td></tr>
<tr><td>3</td></tr>
<tr><td>4</td></tr>
<tr><td>5</td></tr>
<tr><td>6</td></tr>
<tr><td>7</td></tr>
<tr><td>8</td></tr>
<tr><td>9</td></tr>
<tr><td>10</td></tr>
</table>
</body>
<script type="text/javascript">
window.onload=function()
{
    var tbl = document.getElementById("tbl");
    rows = tbl.getElementsByTagName("tr");
    for(i=0;i<rows.length;i++)
    {
        var j = parseInt(i/3);

```

```
        if(j%2==0) rows[i].style.backgroundColor="#f00";
        else rows[i].style.backgroundColor="#0f0";
    }
}
</script>
```

### 3、HTML 的 form 提交之前如何验证数值文本框的内容全部为数字？否则的话提示用户并终止提交？

```
<form onsubmit='return chkForm(this)'>
<input type="text" name="d1"/>
<input type="submit"/>
</form>
<script type="text/javascript" />
function chkForm(this)
{
    var value = thist.d1.value;
    var len = value.length;
    for(var i=0;i<len;i++)
    {
        if(value.charAt(i)>"9" || value.charAt(i)<"0")
        {
            alert("含有非数字字符");
            return false;
        }
    }
    return true;
}
</script>
```

### 4、请写出用于校验 HTML 文本框中输入的内容全部为数字的 javascript 代码

```
<input type="text" id="d1" onblur="chkNumber (this)"/>
<script type="text/javascript" />
function chkNumber(eleText)

{
    var value = eleText.value;
    var len = value.length;
    for(var i=0;i<len;i++)
    {
        if(value.charAt(i)>"9" || value.charAt(i)<"0")
        {
```

```
        alert("含有非数字字符");

        eleText.focus();
        break;
    }
}

</script>
```

除了写完代码，还应该在网页上写出实验步骤和在代码中加入实现思路，让面试官一看就明白你的意图和检查你的结果。

## 5、说说你用过那些 **ajax** 技术和框架，说说它们的区别

## 四. Java web 部分

### 1、**Tomcat** 的优化经验

答: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、HTTP 请求的 **GET** 与 **POST** 方式的区别

答 :servlet 有良好的生存期的定义，包括加载和实例化、初始化、处理请求以及服务结束。这个生存期由 javax.servlet.Servlet 接口的 init, service 和 destroy 方法表达。

### 3、解释一下什么是 **servlet**;

答 :servlet 有良好的生存期的定义，包括加载和实例化、初始化、处理请求以及服务结束。这个生存期由 javax.servlet.Servlet 接口的 init, service 和 destroy 方法表达。

### 4、说一说 **Servlet** 的生命周期?

答 :servlet 有良好的生存期的定义，包括加载和实例化、初始化、处理请求以及服务结束。这个生存期由 javax.servlet.Servlet 接口的 init, service 和 destroy 方法表达。

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。

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。

## 5、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 {  
    }  
}
```

## 6、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。

## 7、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()。

## 8、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

## 9、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。

## 10、`request.getAttribute()` 和 `request.getParameter()` 有何区别？

## 11. jsp 有哪些内置对象?作用分别是什么? 分别有什么方法?

答:JSP 共有以下 9 个内置的对象 :

`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

`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` 可以贮存用户的状态信息

`application` 表示一个 `javax.servlet.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

## 12. JSP 有哪些动作?作用分别是什么?

(这个问题似乎不重要,不明白为何有此题)

答:JSP 共有以下 6 种基本动作

jsp:include : 在页面被请求的时候引入一个文件。

jsp:useBean : 寻找或者实例化一个 JavaBean。

jsp:setProperty : 设置 JavaBean 的属性。

jsp:getProperty : 输出某个 JavaBean 的属性。

jsp:forward : 把请求转到一个新的页面。

jsp:plugin : 根据浏览器类型为 Java 插件生成 OBJECT 或 EMBED 标记

## 13、JSP 的常用指令

isErrorPage(是否能使用 Exception 对象), isELIgnored(是否忽略表达式)

## 14. JSP 中动态 INCLUDE 与静态 INCLUDE 的区别?

答:动态 INCLUDE 用 jsp:include 动作实现

<jsp:include page=included.jsp flush=true />它总是会检查所含文件中的变化,适合用于包含动态页面,并且可以带参数 静态 INCLUDE 用 include 伪码实现,定不会检查所含文件的变化,适用于包含静态页面 <%@ include file=included.htm %>

## 15、两种跳转方式分别是什么?有什么区别?

(下面的回答严重错误,应该是想问 forward 和 sendRedirect 的区别,毕竟出题的人不是专业搞文字艺术的人,可能表达能力并不见得很强,用词不一定精准,加之其自身的技术面也可能存在一些问题,不一定真正将他的意思表达清楚了,严格意思上来讲,一些题目可能根本就无人能答,所以,答题时要掌握主动,只要把自己知道的表达清楚就够了,而不要去推敲原始题目具体含义是什么,不要一味想着是在答题)

答:有两种,分别为:

```
<jsp:include page=included.jsp flush=true>
<jsp:forward page= nextpage.jsp/>
```

前者页面不会转向 include 所指的页面,只是显示该页的结果,主页面还是原来的页面。执行完后还会回来,相当于函数调用。并且可以带参数.后者完全转向新页面,不会再回来。相当于 go to 语句。

## 16、页面间对象传递的方法

request , session , application , cookie 等

## 17、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。

## 18、MVC 的各个部分都有那些技术来实现?如何实现?

答:MVC 是 Model - View - Controller 的简写。Model 代表的是应用的业务逻辑(通过 JavaBean , EJB 组件实现) , View 是应用的表示面(由 JSP 页面产生), Controller 是提供应用的处理过程控制(一般是一个 Servlet ) , 通过这种设计模型把应用逻辑 , 处理过程和显示逻辑分成不同的组件实现。这些组件可以进行交互和重用。

## 19、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;  
}
```

**20.** 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**

## 五. 数据库部分

**1、用两种方式根据部门号从高到低，工资从低到高列出每个员工的信息。**

```
employee:  
    eid,ename,salary,deptid;  
    select * from employee order by deptid desc,salary
```

**2、列出各个部门中工资高于本部门的平均工资的员工数和部门号，并按部门号排序**

创建表：

```
mysql> create table employee921(id int primary key auto_increment,name varchar(50),salary bigint,deptid int);
```

插入实验数据：

```
mysql> insert into employee921 values(null,'zs',1000,1),(null,'ls',1100,1),(null,'ww',1100,1),(null,'zl',900,1),(null,'zl',1000,2),(null,'zl',900,2),(null,'zl',1000,2),(null,'zl',1100,2);
```

编写 sql 语句：

```
( ) select avg(salary) from employee921 group by deptid;  
( ) mysql> select employee921.id,employee921.name,employee921.salary,employee921.deptid tid from employee921 where salary > (select avg(salary) from employee921 where deptid = tid);  
效率低的一个语句，仅供学习参考使用（在 group by 之后不能使用 where，只能使用 having，在 group by 之前可以使用 where，即表示对过滤后的结果分组）：
```

```
mysql> select employee921.id,employee921.name,employee921.salary,employee921.deptid tid from employee921 where salary > (select avg(salary) from employee921 group by deptid having deptid = tid);  
( ) select count(*) ,tid  
      from (  
        select employee921.id,employee921.name,employee921.salary,employee921.deptid tid  
        from employee921
```

```

where salary >
    (select avg(salary) from employee921 where deptid = tid)
) as t
group by tid ;

```

另外一种方式：关联查询

```

select a.ename,a.salary,a.deptid
from emp a,
     (select deptd,avg(salary) avgsal from emp group by deptid ) b
where a.deptid=b.deptid and a.salary>b.avgsal;

```

### 3、存储过程与触发器必须讲，经常被面试到？

```

create procedure insert_Student (_name varchar(50),_age int ,out _id int)
begin
    insert into student value(null,_name,_age);
    select max(stuId) into _id from student;
end;

```

```

call insert_Student('wfz',23,@id);
select @id;

```

```

mysql> create trigger update_Student BEFORE update on student FOR EACH ROW
-> select * from student;

```

触发器不允许返回结果

```

create trigger update_Student BEFORE update on student FOR EACH ROW
insert into student value(null,'zxx',28);

```

mysql 的触发器目前不能对当前表进行操作

```

create trigger update_Student BEFORE update on student FOR EACH ROW
delete from articles where id=8;

```

这个例子不是很好，最好是用删除一个用户时，顺带删除该用户的所有帖子

这里要注意使用 OLD.id

触发器用处还是很多的，比如校内网、开心网、Facebook，你发一个日志，自动通知好友，其实就是在增加日志时做一个后触发，再向通知表中写入条目。因为触发器效率高。而 UCH 没有用触发器，效率和数据处理能力都很低。

存储过程的实验步骤：

```
mysql> delimiter |
mysql> create procedure insertArticle_Procedure (pTitle varchar(50),pBid int,out
pId int)
-> begin
-> insert into article1 value(null,pTitle,pBid);
-> select max(id) into pId from article1;
-> end;
-> |
Query OK, 0 rows affected (0.05 sec)

mysql> call insertArticle_Procedure('传智播客',1,@pid);
-> |
Query OK, 0 rows affected (0.00 sec)

mysql> delimiter ;
mysql> select @pid;
+-----+
| @pid |
+-----+
| 3    |
+-----+
1 row in set (0.00 sec)

mysql> select * from article1;
+----+-----+----+
| id | title      | bid  |
+----+-----+----+
| 1  | test       | 1    |
| 2  | chuanzhiboke | 1    |
| 3  | 传智播客     | 1    |
+----+-----+----+
3 rows in set (0.00 sec)
```

### 触发器的实验步骤：

```
create table board1(id int primary key auto_increment,name varchar(50),ar
ticleCount int);

create table article1(id int primary key auto_increment,title varchar(50)
,bid int references board1(id));
```

```
delimiter |

create trigger insertArticle_Trigger after insert on article1 for each row
begin
    -> update board1 set articleCount=articleCount+1 where id= NEW.bid;
    -> end;
    -> |

delimiter ;
```

```
insert into board1 value (null,'test',0);
```

```
insert into article1 value(null,'test',1);
```

还有，每插入一个帖子，都希望将版面表中的最后发帖时间，帖子总数字段进行同步更新，用触发器做效率就很高。下次课设计这样一个案例，写触发器时，对于最后发帖时间可能需要用 `declare` 方式声明一个变量，或者是用 `NEW.posttime` 来生成。

#### 4、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）：

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唯一地区分。通常需要为表加上一个列，以存储各个实例的唯一标识。这个唯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，以存储各个实例的唯一标识。简而言之，第二范式就是非主属性非部分依赖于主关键字。

第三范式的要求如下：

满足第三范式（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。

## 5、说出一些数据库优化方面的经验？

用 PreparedStatement 一般来说比 Statement 性能高：一个 sql 发给服务器去执行，涉及步骤：语法检查、语义分析，编译，缓存

```
"inert into user values(1,1,1)" ->二进制  
"inert into user values(2,2,2)" ->二进制  
"inert into user values(?, ?, ?)" ->二进制
```

有外键约束会影响插入和删除性能，如果程序能够保证数据的完整性，那在设计数据库时就去掉外键。（比喻：就好比免检产品，就是为了提高效率，充分相信产品的制造商）

（对于 hibernate 来说，就应该有一个变化：employee->Deptment 对象，现在设计时就成了 employee->deptid）

看 mysql 帮助文档子查询章节的最后部分，例如，根据扫描的原理，下面的子查询语句要比第二条关联查询的效率高：

```
1. select e.name,e.salary where e.managerid=(select id from employee where name='zxx');  
  
2. select e.name,e.salary,m.name,m.salary from employees e,employees m where  
e.managerid = m.id and m.name='zxx';
```

表中允许适当冗余，譬如，主题帖的回复数量和最后回复时间等

将姓名和密码单独从用户表中独立出来。这可以是非常好的一对一对的案例哟！

sql 语句全部大写，特别是列名和表名都大写。特别是 sql 命令的缓存功能，更加需要统一大小写，sql 语句→发给 oracle

服务器→语法检查和编译成为内部指令→缓存和执行指令。根据缓存的特点，不要拼凑条件，而是用?和 PreparedStatement

还有索引对查询性能的改进也是值得关注的。

备注：下面是关于性能的讨论举例

4 航班 3 个城市

$m \times n$

```
select * from flight,city where flight.startcityid=city.cityid and city.name='beijing';
```

$m + n$

```
select * from flight where startcityid = (select cityid from city where cityname='beijing');
```

```
select flight.id,'beijing',flight.flightTime from flight where startcityid = (select cityid from city where cityname='beijing')
```

## 6、union 和 union all 有什么不同？

假设我们有一个表 Student，包括以下字段与数据：

```
drop table student;
create table student
(
    id int primary key,
    namenvarchar2(50) not null,
    score number not null
);
insert into student values(1,'Aaron',78);
insert into student values(2,'Bill',76);
insert into student values(3,'Cindy',89);
insert into student values(4,'Damon',90);
insert into student values(5,'Ella',73);
insert into student values(6,'Frado',61);
insert into student values(7,'Gill',99);
insert into student values(8,'Hellen',56);
insert into student values(9,'Ivan',93);
insert into student values(10,'Jay',90);
```

commit;

Union 和 Union All 的区别。

```
select *  
from student  
where id < 4  
union  
select *  
from student  
where id > 2 and id < 6
```

结果将是

1	Aaron	78
2	Bill	76
3	Cindy	89
4	Damon	90
5	Ella	73

如果换成 Union All 连接两个结果集，则返回结果是：

1	Aaron	78
2	Bill	76
3	Cindy	89
3	Cindy	89
4	Damon	90
5	Ella	73

可以看到，Union 和 Union All 的区别之一在于对重复结果的处理。

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。如：

```
select * from gc_dfys  
union  
select * from ls_jg_dfys
```

这个 SQL 在运行时先取出两个表的结果，再用排序空间进行排序删除重复的记录，最后返回结果集，如果表数据量大的话可能会导致用磁盘进行排序。

而 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，

## 7. 分页语句

### 取出 sql 表中第 31 到 40 的记录 (以自动增长 ID 为主键)

sql server 方案 1：

```
select top 10 * from t where id not in (select top 30 id from t order by id) order by id
```

sql server 方案 2：

```
select top 10 * from t where id in (select top 40 id from t order by id) order by id desc
```

mysql 方案：select \* from t order by id limit 30,10

oracle 方案：select \* from (select rownum r,\* from t where r<=40) where r>30

-----待整理进去的内容-----

```
pageSize=20;  
pageNo = 5;
```

### 1. 分页技术 1 (直接利用 sql 语句进行分页，效率最高和最推荐的)

```
mysql:sql = "select * from articles limit " + (pageNo-1)*pageSize + "," + pageSize;  
oracle: sql = "select * from " +  
              "(select rownum r,* from " +  
               "(select * from articles order by postime desc)" +  
               "where rownum<= " + pageNo*pageSize +") tmp " +  
               "where r>" + (pageNo-1)*pageSize;
```

注释：第 7 行保证 rownum 的顺序是确定的，因为 oracle 的索引会造成 rownum 返回不同的值

简评提示：没有 order by 时，rownum 按顺序输出，一旦有了 order by，rownum 不按顺序输出了，这说明 rownum 是排序前的编号。如果对 order by 从句中的字段建立了索引，那么，rownum 也是按顺序输出的，因为这时候生成原始的查询结果集时会参照索引表的顺序来构建。

```
sqlserver:sql = "select top 10 * from id not id(select top " + (pageNo-1)*pageSize + "id from articles)"
```

```
DataSource ds = new InitialContext().lookup(jndiurl);  
Connection cn = ds.getConnection();  
// "select * from user where id=?" --->binary directive  
PreparedStatement pstmt = cn.prepareStatement(sql);
```

```
ResultSet rs = pstmt.executeQuery()
while(rs.next())
{
    out.println(rs.getString(1));
}
```

## 2.不可滚动的游标

```
pageSize=20;
pageNo = 5;
cn = null
stmt = null;
rs = null;
try
{
    sqlserver:sql = "select * from articles";

    DataSource ds = new InitialContext().lookup(jndiurl);
    Connection cn = ds.getConnection();
    //"select * from user where id=?" --->binary directive
    PreparedStatement pstmt = cn.prepareStatement(sql);
    ResultSet rs = pstmt.executeQuery()
    for(int j=0;j<(pageNo-1)*pageSize;j++)
    {
        rs.next();
    }

    int i=0;

    while(rs.next() && i<10)
    {
        i++;
        out.println(rs.getString(1));
    }
}
catch(){}
finally
{
    if(rs!=null)try{rs.close();}catch(Exception e){}
    if(stmt....)
    if(cn....)
}
```

### 3. 可滚动的游标

```
pageSize=20;
pageNo = 5;
cn = null
stmt = null;
rs = null;
try
{
sqlserver:sql = "select * from articles";

DataSource ds = new InitialContext().lookup(jndiurl);
Connection cn = ds.getConnection();
//"select * from user where id=?" --->binary directive
PreparedStatement pstmt = cn.prepareStatement(sql,ResultSet.TYPE_SCROLL_INSENSITIVE,...);
//根据上面这行代码的异常 SQLFeatureNotSupportedException , 就可判断驱动是否支持可滚动游标

ResultSet rs = pstmt.executeQuery()
rs.absolute((pageNo-1)*pageSize)
int i=0;
while(rs.next() && i<10)
{
    i++;
    out.println(rs.getString(1));
}
}
catch(){}
finally
{
    if(rs!=null)try{rs.close();}catch(Exception e){}
    if(stmt....)
    if(cn....)
}
```

### 8. 用一条 SQL 语句 查询出每门课都大于 80 分的学生姓名

	name	kecheng	fenshu
张三		语文	81
张三		数学	75
李四		语文	76

李四	数学	90
王五	语文	81
王五	数学	100
王五	英语	90

准备数据的 sql 代码：

```
create table score(id int primary key auto_increment,name varchar(20),subject varchar(20),score int);
insert into score values
(null,'张三','语文',81),
(null,'张三','数学',75),
(null,'李四','语文',76),
(null,'李四','数学',90),
(null,'王五','语文',81),
(null,'王五','数学',100),
(null,'王五 ','英语',90);
```

提示：当百思不得其解时，请理想思维，把小变成大做，把大变成小做，

答案：

A: select distinct name from score where name not in (select distinct name from score where score<=80)

B:select distince name t1 from score where 80< all (select score from score where name=t1);

## 9.所有部门之间的比赛组合

一个叫 department 的表，里面只有一个字段 name，一共有 4 条纪录，分别是 a, b, c, d，对应四个球对，现在四个球对进行比赛，用一条 sql 语句显示所有可能的比赛组合。

答：select a.name, b.name  
from team a, team b

where a.name < b.name

## 10.每个月份的发生额都比 101 科目多的科目

请用 SQL 语句实现：从 TestDB 数据表中查询出所有月份的发生额都比 101 科目相应月份的发生额高的科目。请注意：TestDB 中有很多科目，都有 1 - 12 月份的发生额。

AccID：科目代码，Occmonth：发生额月份，DebitOccur：发生额。

数据库名：JcyAudit，数据集：Select \* from TestDB

准备数据的 sql 代码：

```
drop table if exists TestDB;
create table TestDB(id int primary key auto_increment, AccID varchar(20), Occmonth date, DebitOccur bigint);
insert into TestDB values
(null,'101','1988-1-1',100),
(null,'101','1988-2-1',110),
(null,'101','1988-3-1',120),
(null,'101','1988-4-1',100),
(null,'101','1988-5-1',100),
(null,'101','1988-6-1',100),
(null,'101','1988-7-1',100),
(null,'101','1988-8-1',100);
```

--复制上面的数据，故意把第一个月份的发生额数字改小一点

```
insert into TestDB values
(null,'102','1988-1-1',90),
(null,'102','1988-2-1',110),
(null,'102','1988-3-1',120),
(null,'102','1988-4-1',100),
(null,'102','1988-5-1',100),
(null,'102','1988-6-1',100),
(null,'102','1988-7-1',100),
(null,'102','1988-8-1',100);
```

--复制最上面的数据，故意把所有发生额数字改大一点

```
insert into TestDB values
(null,'103','1988-1-1',150),
(null,'103','1988-2-1',160),
(null,'103','1988-3-1',180),
(null,'103','1988-4-1',120),
```

```
(null,'103','1988-5-1',120),
(null,'103','1988-6-1',120),
(null,'103','1988-7-1',120),
(null,'103','1988-8-1',120);
```

--复制最上面的数据，故意把所有发生额数字改大一点

```
insert into TestDB values
(null,'104','1988-1-1',130),
(null,'104','1988-2-1',130),
(null,'104','1988-3-1',140),
(null,'104','1988-4-1',150),
(null,'104','1988-5-1',160),
(null,'104','1988-6-1',170),
(null,'104','1988-7-1',180),
(null,'104','1988-8-1',140);
```

--复制最上面的数据，故意把第二个月份的发生额数字改小一点

```
insert into TestDB values
(null,'105','1988-1-1',100),
(null,'105','1988-2-1',80),
(null,'105','1988-3-1',120),
(null,'105','1988-4-1',100),
(null,'105','1988-5-1',100),
(null,'105','1988-6-1',100),
(null,'105','1988-7-1',100),
(null,'105','1988-8-1',100);
```

答案：

```
select distinct AccID from TestDB
where AccID not in
    (select TestDB.AccIDfrom TestDB,
        (select * from TestDB where AccID='101') as db101
    where TestDB.Occmonth=db101.Occmonth and TestDB.DebitOccur<=db101.DebitOccur
    );
```

## 11.统计每年每月的信息

year	month	amount
1991	1	1.1
1991	2	1.2
1991	3	1.3
1991	4	1.4

---

1992	1	2. 1
1992	2	2. 2
1992	3	2. 3
1992	4	2. 4

查成这样一个结果

```
year m1    m2    m3    m4
1991 1. 1  1. 2  1. 3  1. 4
1992 2. 1  2. 2  2. 3  2. 4
```

提示：这个与工资条非常类似，与学生的科目成绩也很相似。

准备sql语句：

```
drop table if exists sales;
create table sales(id int auto_increment primary key, year varchar(10), month varchar(10), amount float(2, 1));
insert into sales values
(null, '1991', '1', 1.1),
(null, '1991', '2', 1.2),
(null, '1991', '3', 1.3),
(null, '1991', '4', 1.4),
(null, '1992', '1', 2.1),
(null, '1992', '2', 2.2),
(null, '1992', '3', 2.3),
(null, '1992', '4', 2.4);
```

答案一、

```
select sales.year ,
(select t.amount from sales t where t.month='1' and t.year= sales.year) '1',
(select t.amount from sales t where t.month='2' and t.year= sales.year) '2',
(select t.amount from sales t where t.month='3' and t.year= sales.year) '3',
(select t.amount from sales t where t.month='4' and t.year= sales.year) as '4'
from sales group by year;
```

## 12.显示文章标题，发帖人、最后回复时间

表：id, title, postuser, postdate, parentid

准备 sql 语句：

```

drop table if exists articles;
create table articles(id int auto_increment primary key,title varchar(50), postuser varchar(10),
postdate datetime,parentid int references articles(id));
insert into articles values
(null,'第一条','张三','1998-10-10 12:32:32',null),
(null,'第二条','张三','1998-10-10 12:34:32',null),
(null,'第一条回复 1','李四','1998-10-10 12:35:32',1),
(null,'第二条回复 1','李四','1998-10-10 12:36:32',2),
(null,'第一条回复 2','王五','1998-10-10 12:37:32',1),
(null,'第一条回复 3','李四','1998-10-10 12:38:32',1),
(null,'第二条回复 2','李四','1998-10-10 12:39:32',2),
(null,'第一条回复 4','王五','1998-10-10 12:39:40',1);

```

答案：

```

select a.title,a.postuser,
(select max(postdate) from articles where parentid=a.id) reply
from articles a where a.parentid is null;

```

注释：子查询可以用在选择列中，也可用于 where 的比较条件中，还可以用于 from 从句中。

### 13.删除除了 id 号不同，其他都相同的学生冗余信息

2. 学生表 如下：

id号	学号	姓名	课程编号	课程名称	分数
1	2005001	张三	0001	数学	69
2	2005002	李四	0001	数学	89
3	2005001	张三	0001	数学	69

A: delete from tablename where id号 not in(select min(id号) from tablename group by 学号,姓名,课程编号,课程名称,分数)

实验：

```
create table student2(id int auto_increment primary key,code varchar(20),name varchar(20));
```

```
insert into student2 values(null,'2005001','张三'),(null,'2005002','李四'),(null,'2005001','张三');
```

//如下语句，mysql 报告错误，可能删除依赖后面统计语句，而删除又导致统计语句结果不一致。

```
delete from student2 where id not in(select min(id) from student2 group by name);
```

//但是，如下语句没有问题：

```
select * from student2 where id not in(select min(id) from student2 group by name);
```

//于是，我想先把分组的结果做成虚表，然后从虚表中选出结果，最后再将结果作为删除的条件数据。

```
delete from student2 where id not in(select mid from (select min(id) mid  
from student2 group by name) as t);
```

或者：

```
delete from student2 where id not in(select min(id) from (select * from s  
tudent2) as t group by t.name);
```

## 14.航空网的几个航班查询题：

表结构如下：

```
flight{flightID,StartCityID ,endCityID,StartTime}  
city{cityID, CityName}
```

实验环境：

```
create table city(cityID int auto_increment primary key,cityName varchar(20));  
create table flight (flightID int auto_increment primary key,  
    StartCityID int references city(cityID),  
    endCityID int references city(cityID),  
    StartTime timestamp);
```

//航班本来应该没有日期部分才好，但是下面的题目当中涉及到了日期

```
insert into city values(null,'北京'),(null,'上海'),(null,'广州');  
insert into flight values  
(null,1,2,'9:37:23'),(null,1,3,'9:37:23'),(null,1,2,'10:37:23'),(null,2,3,'10:37:23');
```

1、查询起飞城市是北京的所有航班，按到达城市的名字排序

参与运算的列是我起码能够显示出来的那些列，但最终我不一定把它们显示出来。各个表组合出来的中间结果字段中必须包含所

有运算的字段。

```
select * from flight f,city c
where f.endcityid = c.cityid and startcityid =
(select c1.cityid from city c1 where c1.cityname = "北京")
order by c.cityname asc;
```

```
mysql> select flight.flightid,'北京' startcity, e.cityname from flight,city e where
flight.endcityid=e.cityid and flight.startcityid=(select cityid from city where
cityname='北京');
```

```
mysql> select flight.flightid,s.cityname,e.cityname from flight,city s,city e where
flight.startcityid=s.cityid and s.cityname='北京' and flight.endCityId=e.cityID
order by e.cityName desc;
```

2、查询北京到上海的所有航班纪录（起飞城市，到达城市，起飞时间，航班号）

```
select c1.CityName,c2.CityName,f.StartTime,f.flightID
from city c1,city c2,flight f
where f.StartCityID=c1.cityID
and f.endCityID=c2.cityID
and c1.cityName='北京'
and c2.cityName='上海'
```

3、查询具体某一天（2005-5-8）的北京到上海的航班次数

```
select count(*) from
(select c1.CityName,c2.CityName,f.StartTime,f.flightID
from city c1,city c2,flight f
where f.StartCityID=c1.cityID
and f.endCityID=c2.cityID
and c1.cityName='北京'
and c2.cityName='上海'
and 查帮助获得的某个日期处理函数(startTime) like '2005-5-8%'
```

mysql 中提取日期部分进行比较的示例代码如下：

```
select * from flight where date_format(starttime,'%Y-%m-%d')='1998-01-02'
```

### 15.查出比经理薪水还高的员工信息:

```

Drop table if not exists employees;
create table employees(id int primary key auto_increment,name varchar(50)
,salary int,managerid int references employees(id));
insert into employees values (null,'lhm',10000,null), (null,'zxx',15000,1
),(null,'flx',9000,1),(null,'tg',10000,2),(null,'wzg',10000,3);

```

wzg 大于 flx, lhm 大于 zxx

解题思路：

根据 sql 语句的查询特点，是逐行进行运算，不可能两行同时参与运算。

涉及了员工薪水和经理薪水，所有，一行记录要同时包含两个薪水，所有想到要把这个表自关联组合一下。

首先要组合出一个包含有各个员工及该员工的经理信息的长记录，譬如，左半部分是员工，右半部分是经理。而迪卡尔积会组合出很多垃圾信息，先去除这些垃圾信息。

```
select e.* from employees e,employees m where e.managerid=m.id and e.salary>m.salary;
```

### 16、求出小于 45 岁的各个老师所带的大于 12 岁的学生人数

数据库中有 3 个表 teacher 表，student 表，tea\_stu 关系表。

teacher 表 teaID name age

student 表 stuID name age

teacher\_student 表 teaID stuID

要求用一条 sql 查询出这样的结果

1. 显示的字段要有老师 name, age 每个老师所带的学生人数

2 只列出老师 age 为 40 以下，学生 age 为 12 以上的记录

预备知识：

1.sql 语句是对每一条记录依次处理，条件为真则执行动作 ( select,insert,delete,update )

2.只要是迪卡尔积，就会产生“垃圾”信息，所以，只要迪卡尔积了，我们首先就要想到清除“垃圾”信息

实验准备：

```

drop table if exists tea_stu;
drop table if exists teacher;
drop table if exists student;
create table teacher(teaID int primary key,name varchar(50),age int);
create table student(stuID int primary key,name varchar(50),age int);
create table tea_stu(teaID int references teacher(teaID),stuID int references student(stuID));

```

```
insert into teacher values(1,'zxx',45), (2,'lhm',25) , (3,'wzg',26) , (4,'tg',27);
insert into student values(1,'wy',11), (2,'dh',25) , (3,'ysq',26) , (4,'mxc',27);
insert into tea_stu values(1,1), (1,2), (1,3);
insert into tea_stu values(2,2), (2,3), (2,4);
insert into tea_stu values(3,3), (3,4), (3,1);
insert into tea_stu values(4,4), (4,1), (4,2) , (4,3);
```

结果 : 2→3,3→2,4→3

解题思路 : ( 真面试答题时 , 也要写出每个分析步骤 , 如果纸张不够 , 就找别人要 )

1 要会统计分组信息 , 统计信息放在中间表中 :

```
select teaid,count(*) from tea_stu group by teaid;
```

2 接着其实应该是筛除掉小于 12 岁的学生 , 然后再进行统计 , 中间表必须与 student 关联才能得到 12 岁以下学生和把该学生记录从中间表中剔除 , 代码是 :

```
select tea_stu.teaid,count(*) total from student,tea_stu
where student.stuid=tea_stu.stuid and student.age>12 group by tea_stu.teaid
```

3.接着把上面的结果做成虚表与 teacher 进行关联 , 并筛除大于 45 的老师

```
select teacher.teaid,teacher.name,total from teacher ,(select tea_stu.tea
id,count(*) total from student,tea_stu where student.stuid=tea_stu.stuid and stu
dent.age>12 group by tea_stu.teaid) as tea_stu2 where teacher.teaid=tea_stu2.tea
id and teacher.age<45;
```

## 17.求出发帖最多的人:

```
select authorid,count(*) total from articles
group by authorid
having total=
(select max(total2) from (select count(*) total2 from articles group by authorid) as t);

select t.authorid,max(t.total) from
(select authorid,count(*) total from articles ) as t
```

这条语句不行 , 因为 max 只有一列 , 不能与其他列混淆。

```
select authorid,count(*) total from articles
group by authorid having total=max(total)也不行。
```

## 18、一个用户表中有一个积分字段，假如数据库中有 100 多万个用户，若要在每年第一天凌晨将积分清零，你将考虑什么，你将想什么办法解决？

```
alter table drop column score;
alter table add column score int;
```

可能会很快 , 但是需要试验 , 试验不能拿真实的环境来操刀 , 并且要注意 ,

这样的操作时无法回滚的，在我的印象中，只有 insert update delete 等 DML 语句才能回滚，

对于 create table, drop table , alter table 等 DDL 语句是不能回滚。

解决方案一，update user set score=0;

解决方案二，假设上面的代码要执行好长时间，超出我们的容忍范围，那我就 alter table user drop column score; alter table user add column score int。

下面代码实现每年的那个凌晨时刻进行清零。

```
Runnable runnable =  
    new Runnable(){  
        public void run(){  
            clearDb();  
            schedule(this,new Date(new Date().getYear()+1,0,0));  
        }  
    };  
  
schedule(runnable,  
    new Date(new Date().getYear()+1,0,1));
```

## 19、一个用户具有多个角色，请查询出该表中具有该用户的所有角色的其他用户。

```
select count(*) as num,tb.id  
from  
tb,  
(select role from tb where id=xxx) as t1  
where  
tb.role = t1.role and tb.id != t1.id  
group by tb.id  
having  
num = select count(role) from tb where id=xxx;
```

## 20. xxx 公司的 sql 面试

Table **EMPLOYEES** Structure:

```
EMPLOYEE_ID      NUMBER      Primary Key,  
FIRST_NAME       VARCHAR2(25),  
LAST_NAME        VARCHAR2(25),  
Salary number(8,2),  
HiredDate DATE,
```

---

Departmentid number(2)

Table **Departments** Structure:

```
Departmentid number(2)      Primary Key,
DepartmentName  VARCHAR2(25).
```

(2) 基于上述 EMPLOYEES 表写出查询 :写出雇用日期在今年的 ,或者工资在 [1000,2000] 之间的 ,或者员工姓名( last\_name )以 'Obama' 打头的所有员工 , 列出这些员工的全部个人信息。 ( 4 分 )

```
select * from employees
where Year(hiredDate) = Year(date())
or (salary between 1000 and 200)
or left(last_name,3)='abc';
```

(3) 基于上述 EMPLOYEES 表写出查询 :查出部门平均工资大于 1800 元的部门的所有员工 , 列出这些员工的全部个人信息。 ( 4 分 )

```
mysql> select id,name,salary,deptid did from employee1 where (select avg(salary)
from employee1 where deptid = did) > 1800;
```

(4) 基于上述 EMPLOYEES 表写出查询 :查出个人工资高于其所在部门平均工资的员工 , 列出这些员工的全部个人信息及该员工工资高出部门平均工资百分比。 ( 5 分 )

```
select employee1.*, (employee1.salary-t.avgSalary)*100/employee1.salary
from employee1,
(select deptid,avg(salary) avgSalary from employee1 group by deptid) as t
where employee1.deptid = t.deptid and employee1.salary>t.avgSalary;
```

## 21、注册 **Jdbc** 驱动程序的三种方式

## 22、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();
            }
        }
    }
}

```

## 23、JDBC 中的 PreparedStatement 相比 Statement 的好处

答：一个 sql 命令发给服务器去执行的步骤为：语法检查，语义分析，编译成内部指令，缓存指令，执行指令等过程。

select \* from student where id =3----缓存-->xxxxx 二进制命令

select \* from student where id =3----直接取→xxxxx二进制命令

select \* from student where id =4--- →会怎么干？

如果当初是 select \* from student where id =?--- →又会怎么干？

上面说的是性能提高

可以防止 sql 注入。

## 24. 写一个用 **jdbc** 连接并访问 **oracle** 数据的程序代码

## 25、**Class.forName** 的作用?为什么要用?

答：按参数中指定的字符串形式的类名去搜索并加载相应的类，如果该类字节码已经被加载过，则返回代表该字节码的 Class 实例对象，否则，按类加载器的委托机制去搜索和加载该类，如果所有的类加载器都无法加载到该类，则抛出 ClassNotFoundException。加载完这个 Class 字节码后，接着就可以使用 Class 字节码的 newInstance 方法去创建该类的实例对象了。

有时候，我们程序中所有使用的具体类名在设计时（即开发时）无法确定，只有程序运行时才能确定，这时候就需要使用 Class.forName 去动态加载该类，这个类名通常是在配置文件中配置的，例如，spring 的 ioc 中每次依赖注入的具体类就是这样配置的，jdbc 的驱动类名通常也是通过配置文件来配置的，以便在产品交付使用后不用修改源程序就可以更换驱动类名。

## 26、大数据量下的分页解决方法。

答：最好的办法是利用 sql 语句进行分页，这样每次查询出的结果集中就只包含某页的数据内容。再 sql 语句无法实现分页的情况下，可以考虑对大的结果集通过游标定位方式来获取某页的数据。

sql 语句分页，不同的数据库下的分页方案各不一样，下面是主流的三种数据库的分页 sql：

sql server:

```
String sql =
"select top " + pageSize + " * from students where id not in" +
"(select top " + pageSize * (pageNumber-1) + " id from students order by id)" +
"order by id";
```

mysql:

```
String sql =
"select * from students order by id limit " + pageSize*(pageNumber-1) + "," + pageSize;
```

oracle:

```
String sql =
"select * from " +
(select *,rownum rid from (select * from students order by postime desc) where rid<=" +
pagesize*pagenumber + ") as t" +
"where t>" + pageSize*(pageNumber-1);
```

## 27、用 JDBC 查询学生成绩单，把主要代码写出来（考试概率极大）。

```
Connection cn = null;
PreparedStatement pstmt =null;
ResultSet rs = null;
try
{
    Class.forName(driveClassName);
    cn = DriverManager.getConnection(url,username,password);
    pstmt = cn.prepareStatement("select score.* from score ,student " +
        "where score.stuId = student.id and student.name = ?");
    pstmt.setString(1,studentName);
    ResultSet rs = pstmt.executeQuery();
    while(rs.next())
    {
        System.out.println(rs.getInt("subject") + " " + rs.getFloat("score"));
    }
} catch (Exception e) {e.printStackTrace();}
finally
{
    if(rs != null) try{ rs.close() }catch(exception e){}
    if(pstmt != null) try{pstmt.close()}catch(exception e){}
    if(cn != null) try{ cn.close() }catch(exception e){}
}
```

## 28、这段代码有什么不足之处？

```
try {
Connection conn = ...;
Statement stmt = ...;
```

```
ResultSet rs = stmt.executeQuery("select * from table1");

while(rs.next()) {

}

} catch(Exception ex) {
}
```

答：没有 finally 语句来关闭各个对象，另外，使用 finally 之后，要把变量的定义放在 try 语句块的外面，以便在 try 语句块之外的 finally 块中仍可以访问这些变量。

## 29、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。  
实现方式，返回的 Connection 是原始 Connection 的代理，代理 Connection 的 close 方法不是真正关连接，而是把它代理的 Connection 对象还回到连接池中。

## 30、为什么要用 ORM? 和 JDBC 有何不一样?

orm 是一种思想，就是把 object 转变成数据库中的记录，或者把数据库中的记录转变成 object，我们可以用 jdbc 来实现这种思想，其实，如果我们的项目是严格按照 oop 方式编写的话，我们的 jdbc 程序不管是有意还是无意，就已经在实现 orm 的工作了。

现在有许多 orm 工具，它们底层调用 jdbc 来实现了 orm 工作，我们直接使用这些工具，就省去了直接使用 jdbc 的繁琐细节，提高了开发效率，现在用的较多的 orm 工具是 hibernate。也听说一些其他 orm 工具，如 topLink, ojb 等。

## 六. XML 部分

### 1、xml 有哪些解析技术?区别是什么?

答:有 DOM, SAX, STAX 等

DOM:处理大型文件时其性能下降的非常厉害。这个问题是由 DOM 的树结构所造成的，这种结构占用的内存较多，而且 DOM 必须

在解析文件之前把整个文档装入内存，适合对 XML 的随机访问 SAX：不现于 DOM，SAX 是事件驱动型的 XML 解析方式。它顺序读取 XML 文件，不需要一次全部装载整个文件。当遇到像文件开头，文档结束，或者标签开头与标签结束时，它会触发一个事件，用户通过在其回调事件中写入处理代码来处理 XML 文件，适合对 XML 的顺序访问

STAX：Streaming API for XML (StAX)

讲解这些区别是不需要特别去比较，就像说传智播客与其他培训机构的区别时，我们只需说清楚传智播客有什么特点和优点就行了，这就已经间接回答了彼此的区别。

## 2、你在项目中用到了 xml 技术的哪些方面？如何实现的？

答：用到了数据存储，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储在 XML 文件中。

## 3、用 jdom 解析 xml 文件时如何解决中文问题？如何解析？

答：看如下代码，用编码方式加以解决

```
package test;
import java.io.*;
public class DOMTest
{
    private String inFile = "c:\\people.xml"
    private String outFile = "c:\\people.xml"
    public static void main(String args[])
    {
        new DOMTest();
    }
    public DOMTest()
    {
        try
        {
            javax.xml.parsers.DocumentBuilder builder =
            javax.xml.parsers.DocumentBuilderFactory.newInstance().newDocumentBuilder();
            org.w3c.dom.Document doc = builder.newDocument();
            org.w3c.dom.Element root = doc.createElement("老师");
            org.w3c.dom.Element wang = doc.createElement("王");
            org.w3c.dom.Element liu = doc.createElement("刘");
        }
    }
}
```

```
wang.appendChild(doc.createTextNode("我是王老师"));

root.appendChild(wang);
doc.appendChild(root);

javax.xml.transform.Transformer transformer =
javax.xml.transform.TransformerFactory.newInstance().newTransformer();
transformer.setOutputProperty(javax.xml.transform.OutputKeys.ENCODING, "gb2312");
transformer.setOutputProperty(javax.xml.transform.OutputKeys.INDENT, "yes");
transformer.transform(new javax.xml.transform.dom.DOMSource(doc),
new
javax.xml.transform.stream.StreamResult(outFile));
}

catch (Exception e)
{
System.out.println (e.getMessage());
}
}
}
```

#### 4、编程用 JAVA 解析 XML 的方式。

答:用 SAX 方式解析 XML , XML 文件如下 :

```
<?xml version=1.0 encoding=gb2312?>
<person>
<name>王小明</name>

<college>信息学院</college>
<telephone>6258113</telephone>
<notes>男, 1955 年生, 博士, 95 年调入海南大学</notes>
</person>
```

事件回调类 SAXHandler.java

```
import java.io.*;
import java.util.Hashtable;
import org.xml.sax.*;
public class SAXHandler extends HandlerBase
{
private Hashtable table = new Hashtable();
private String currentElement = null;
private String currentValue = null;
public void setTable(Hashtable table)
{
```

```
this.table = table;
}
public Hashtable getTable()
{
return table;
}
public void startElement(String tag, AttributeList attrs)
throws SAXException
{
currentElement = tag;
}
public void characters(char[] ch, int start, int length)
throws SAXException
{
currentValue = new String(ch, start, length);
}
public void endElement(String name) throws SAXException
{
if (currentElement.equals(name))
table.put(currentElement, currentValue);
}

}
```

JSP 内容显示源码, SaxXml.jsp:

```
<HTML>
<HEAD>
<TITLE>剖析 XML 文件 people.xml</TITLE>
</HEAD>
<BODY>
<%@ page errorPage=ErrPage.jsp
contentType=text/html;charset=GB2312 %>
<%@ page import=java.io.* %>
<%@ page import=java.util.Hashtable %>
<%@ page import=org.w3c.dom.* %>
<%@ page import=org.xml.sax.* %>
<%@ page import=javax.xml.parsers.SAXParserFactory %>
<%@ page import=javax.xml.parsers.SAXParser %>
<%@ page import=SAXHandler %>
<%
File file = new File(c:\people.xml);
```

```
FileReader reader = new FileReader(file);
Parser parser;
SAXParserFactory spf = SAXParserFactory.newInstance();
SAXParser sp = spf.newSAXParser();
SAXHandler handler = new SAXHandler();
sp.parse(new InputSource(reader), handler);
Hashtable hashTable = handler.getTable();

out.println(<TABLE BORDER=2><CAPTION>教师信息表</CAPTION>);

out.println(<TR><TD>姓名</TD> + <TD> +
(String)hashTable.get(new String(name)) + </TD></TR>);

out.println(<TR><TD>学院</TD> + <TD> +
(String)hashTable.get(new String(college))+</TD></TR>);

out.println(<TR><TD>电话</TD> + <TD> +
(String)hashTable.get(new String(telephone)) + </TD></TR>);

out.println(<TR><TD>备注</TD> + <TD> +
(String)hashTable.get(new String(notes)) + </TD></TR>);

out.println(</TABLE>);

%>

</BODY>
</HTML>
```

## 5、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: 不同于 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)

## 七. 流行的框架与新技术

## 1、谈谈你对 **Struts** 的理解。

答：

1. struts 是一个按 MVC 模式设计的 Web 层框架，其实它就是一个大大的 servlet，这个 Servlet 名为 ActionServlet，或是 ActionServlet 的子类。我们可以在 web.xml 文件中将符合某种特征的所有请求交给这个 Servlet 处理。这个 Servlet 再参照一个配置文件（通常为 /WEB-INF/struts-config.xml）将各个请求分别分配给不同的 action 去处理。

一个扩展知识点：struts 的配置文件可以有多个，可以按模块配置各自的配置文件，这样可以防止配置文件的过度膨胀；

2. ActionServlet 把请求交给 action 去处理之前，会将请求参数封装成一个 formbean 对象（就是一个 java 类，这个类中的每个属性对应一个请求参数），封装成一个什么样的 formbean 对象呢？看配置文件。

3. 要说明的是，ActionServlet 把 formbean 对象传递给 action 的 execute 方法之前，可能会调用 formbean 的 validate 方法进行校验，只有校验通过后才将这个 formbean 对象传递给 action 的 execute 方法，否则，它将返回一个错误页面，这个错误页面由 input 属性指定，（看配置文件）作者为什么将这里命名为 input 属性，而不是 error 属性，我们后面结合实际的运行效果进行分析。

4. action 执行完后要返回显示的结果视图，这个结果视图是用一个 ActionForward 对象来表示的，actionforward 对象通过 struts-config.xml 配置文件中的配置关联到某个 jsp 页面，因为程序中使用的是在 struts-config.xml 配置文件为 jsp 页面设置的逻辑名，这样可以实现 action 程序代码与返回的 jsp 页面名称的解耦。

你对 struts 可能还有自己的应用方面的经验，那也要一并说出来。

## 2、谈谈你对 **Hibernate** 的理解。

答：

1. 面向对象设计的软件内部运行过程可以理解成就是在不断创建各种新对象、建立对象之间的关系，调用对象的方法来改变各个对象的状态和对象消亡的过程，不管程序运行的过程和操作怎么样，本质上都是要得到一个结果，程序上一个时刻和下一个时刻的运行结果的差异就表现在内存中的对象状态发生了变化。

2. 为了在关机和内存空间不够的状况下，保持程序的运行状态，需要将内存中的对象状态保存到持久化设备和从持久化设备中恢复出对象的状态，通常都是保存到关系数据库来保存大量对象信息。从 Java 程序的运行功能上来讲，保存对象状态的功能相比系统运行的其他功能来说，应该是一个很不起眼的附属功能，java 采用 jdbc 来实现这个功能，这个不起眼的功能却要编写大量的代码，而做的事情仅仅是保存对象和恢复对象，并且那些大量的 jdbc 代码并没有什么技术含量，基本上是采用一套例行公事的标准代码模板来编写，是一种苦活和重复性的工作。

3. 通过数据库保存 java 程序运行时产生的对象和恢复对象，其实现了 java 对象与关系数据库记录的映射关系，称为 ORM（即 Object Relation Mapping），人们可以通过封装 JDBC 代码来实现这种功能，封装出来的产品称之为 ORM 框架，Hibernate 就是其中的一种流行 ORM 框架。使用 Hibernate 框架，不用写 JDBC 代码，仅仅是调用一个 save 方法，就可以将对象保存到关系数据库中，仅仅是调用一个 get 方法，就可以从数据库中加载出一个对象。

4. 使用 Hibernate 的基本流程是：配置 Configuration 对象、产生 SessionFactory、创建 session 对象，启动事务，完成 CRUD 操作，提交事务，关闭 session。

5. 使用 Hibernate 时，先要配置 hibernate.cfg.xml 文件，其中配置数据库连接信息和方言等，还要为每个实体配置相应的 hbm.xml 文件，hibernate.cfg.xml 文件中需要登记每个 hbm.xml 文件。

6. 在应用 Hibernate 时，重点要了解 Session 的缓存原理，级联，延迟加载和 hql 查询。

### 3、AOP 的作用。

### 4、你对 Spring 的理解。

1. Spring 实现了工厂模式的工厂类（在这里有必要解释清楚什么是工厂模式），这个类名为 BeanFactory（实际上是一个接口），在程序中通常 BeanFactory 的子类 ApplicationContext。Spring 相当于一个大的工厂类，在其配置文件中通过 <bean> 元素配置用于创建实例对象的类名和实例对象的属性。

2. Spring 提供了对 IOC 良好支持，IOC 是一种编程思想，是一种架构艺术，利用这种思想可以很好地实现模块之间的解耦。IOC 也称为 DI（Dependency Injection），什么叫依赖注入呢？

譬如，Class Programmer

```
{  
    Computer computer = null;  
    public void code()  
    {  
        //Computer computer = new IBMComputer();  
        //Computer computer = beanfactory.getComputer();  
        computer.write();  
    }  
    public void setComputer(Computer computer)  
    {  
        this.computer = computer;  
    }  
}
```

另外两种方式都由依赖，第一个直接依赖于目标类，第二个把依赖转移到工厂上，第三个彻底与目标和工厂解耦了。在 spring

的配置文件中配置片段如下：

```
<bean id="computer" class="cn.itcast.interview.Computer">
</bean>

<bean id="programmer" class="cn.itcast.interview.Programmer">
    <property name="computer" ref="computer"></property>
</bean>
```

3. Spring 提供了对 AOP 技术的良好封装，AOP 称为面向切面编程，就是系统中有很多各不相干的类的方法，在这些众多方法中要加入某种系统功能的代码，例如，加入日志，加入权限判断，加入异常处理，这种应用称为 AOP。实现 AOP 功能采用的是代理技术，客户端程序不再调用目标，而调用代理类，代理类与目标类对外具有相同的方法声明，有两种方式可以实现相同的方法声明，一是实现相同的接口，二是作为目标的子类在，JDK 中采用 Proxy 类产生动态代理的方式为某个接口生成实现类，如果要为某个类生成子类，则可以用 CGLIB。在生成的代理类的方法中加入系统功能和调用目标类的相应方法，系统功能的代理以 Advice 对象进行提供，显然要创建出代理对象，至少需要目标类和 Advice 类。Spring 提供了这种支持，只需要在 spring 配置文件中配置这两个元素即可实现代理和 aop 功能，例如，

```
<bean id="proxy" type="org.springframework.aop.ProxyBeanFactory">
    <property name="target" ref=""></property>
    <property name="advisor" ref=""></property>
</bean>
```

## 5、谈谈 Struts 中的 Action servlet。

### 6、Struts 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

- 一、 转到展示层时，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
- 二、 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 用到的所有资源都必需统一同步，这个就引起了线程安全的问题。
- 三、 测试不方便。Struts 的每个 Action 都同 Web 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。
- 四、 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 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。
- 五、 对 Servlet 的依赖性过强。Struts 处理 Action 时必需要依赖 HttpServletRequest 和 HttpServletResponse，所有它摆脱不了 Servlet 容器。
- 六、 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。
- 七、 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。
- 八、 对 Action 执行前和后的处理。Struts 处理 Action 的时候是基于 class 的 hierarchies，很难在 action 处理前和后进行操作。
- 九、 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

## 7、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。

## 8、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，但是应用群众众多，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。

（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；）

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。

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。

**9、hibernate 中的 update() 和 saveOrUpdate() 的区别， session 的 load() 和 get() 的区别。**

**10、简述 Hibernate 和 JDBC 的优缺点？如何书写一个 one to many 配置文件。**

**11、iBatis 与 Hibernate 有什么不同？**

相同点：屏蔽 jdbc api 的底层访问细节，使用我们不用与 jdbc api 打交道，就可以访问数据。

jdbc api 编程流程固定，还将 sql 语句与 java 代码混杂在了一起，经常需要拼凑 sql 语句，细节很繁琐。

ibatis 的好处：屏蔽 jdbc api 的底层访问细节；将 sql 语句与 java 代码进行分离；提供了将结果集自动封装称为实体对象和对象的集合的功能，queryForList 返回对象集合，用 queryForObject 返回单个对象；提供了自动将实体对象的属性传递给 sql 语句的参数。

Hibernate 是一个全自动的 orm 映射工具，它可以自动生成 sql 语句，ibatis 需要我们自己在 xml 配置文件中写 sql 语句，hibernate 要比 ibatis 功能负责和强大很多。因为 hibernate 自动生成 sql 语句，我们无法控制该语句，我们就无法去写特定的高效率的 sql。对于一些不太复杂的 sql 查询，hibernate 可以很好帮我们完成，但是，对于特别复杂的查询，hibernate 就很难适应了，这时候用 ibatis 就是不错的选择，因为 ibatis 还是由我们自己写 sql 语句。

**12、写 Hibernate 的一对多和多对一双向关联的 orm 配置？**

**9、hibernate 的 inverse 属性的作用？**

解决方案一，按照 Object[] 数据取出数据，然后自己组 bean

解决方案二，对每个表的 bean 写构造函数，比如表一要查出 field1, field2 两个字段，那么有一个构造函数就是 Bean (type1, filed1, type2, field2)，然后在 hql 里面就可以直接生成这个 bean 了。

### 13、在 DAO 中如何体现 DAO 设计模式?

解决方案一，按照 Object[] 数据取出数据，然后自己组 bean

解决方案二 对每个表的 bean 写构造函数 比如表一要查出 field1,field2 两个字段 那么有一个构造函数就是 Bean(type1 filed1,type2 field2) ，然后在 hql 里面就可以直接生成这个 bean 了。

### 14、spring+Hibernate 中委托方案怎么配置?

解决方案一，按照 Object[] 数据取出数据，然后自己组 bean

解决方案二 对每个表的 bean 写构造函数 比如表一要查出 field1,field2 两个字段 那么有一个构造函数就是 Bean(type1 filed1,type2 field2) ，然后在 hql 里面就可以直接生成这个 bean 了。

### 15、spring+Hibernate 中委托方案怎么配置?

解决方案一，按照 Object[] 数据取出数据，然后自己组 bean

解决方案二 对每个表的 bean 写构造函数 比如表一要查出 field1,field2 两个字段 那么有一个构造函数就是 Bean(type1 filed1,type2 field2) ，然后在 hql 里面就可以直接生成这个 bean 了。

### 16. hibernate 进行多表查询每个表中各取几个字段，也就是说查询出来的结果集没有一个实体类与之对应如何解决;

解决方案一，按照 Object[] 数据取出数据，然后自己组 bean

解决方案二 对每个表的 bean 写构造函数 比如表一要查出 field1,field2 两个字段 那么有一个构造函数就是 Bean(type1 filed1,type2 field2) ，然后在 hql 里面就可以直接生成这个 bean 了。

### 17.介绍一下 Hibernate 的二级缓存

按照以下思路来回答：(1)首先说清楚什么是缓存，(2)再说有了 hibernate 的 Session 就是一级缓存，即有了一级缓存，为什么还要有二级缓存，(3)最后再说如何配置 Hibernate 的二级缓存。

(1) 缓存就是把以前从数据库中查询出来和使用过的对象保存在内存中(一个数据结构中)，这个数据结构通常是或类似

HashMap，当以后要使用某个对象时，先查询缓存中是否有这个对象，如果有则使用缓存中的对象，如果没有则去查询数据库，并将查询出来的对象保存在缓存中，以便下次使用。下面是缓存的伪代码：

引出 hibernate 的第二级缓存，用下面的伪代码分析了 Cache 的实现原理

Dao

```
{  
    hashmap map = new map();  
    User getUser(integer id)  
    {  
        User user = map.get(id)  
        if(user == null)  
        {  
            user = session.get(id);  
            map.put(id,user);  
        }  
        return user;  
    }  
}
```

Dao

```
{  
    Cache cache = null  
    setCache(Cache cache)  
    {  
        this.cache = cache  
    }  
  
    User getUser(int id)  
    {  
        if(cache!=null)  
        {  
            User user = cache.get(id);  
            if(user ==null)  
            {  
                user = session.get(id);  
                cache.put(id,user);  
            }  
            return user;  
        }  
  
        return session.get(id);  
    }  
}
```

```
    }  
}
```

(2) Hibernate 的 Session 就是一种缓存，我们通常将之称为 Hibernate 的一级缓存，当想使用 session 从数据库中查询出一个对象时，Session 也是先从自己内部查看是否存在这个对象，存在则直接返回，不存在才去访问数据库，并将查询的结果保存在自己内部。由于 Session 代表一次会话过程，一个 Session 与一个数据库连接相关连，所以 Session 最好不要长时间保持打开，通常仅用于一个事务当中，在事务结束时就应关闭。并且 Session 是线程不安全的，被多个线程共享时容易出现问题。通常只有那种全局意义上的缓存才是真正的缓存应用，才有较大的缓存价值，因此，Hibernate 的 Session 这一级缓存的缓存作用并不明显，应用价值不大。Hibernate 的二级缓存就是要为 Hibernate 配置一种全局缓存，让多个线程和多个事务都可以共享这个缓存。我们希望的是一个人使用过，其他人也可以使用，session 没有这种效果。

(3) 二级缓存是独立于 Hibernate 的软件部件，属于第三方的产品，多个厂商和组织都提供有缓存产品，例如，EHCache 和 OSCache 等等。在 Hibernate 中使用二级缓存，首先就要在 hibernate.cfg.xml 配置文件中配置使用哪个厂家的缓存产品，接着需要配置该缓存产品自己的配置文件，最后要配置 Hibernate 中的哪些实体对象要纳入到二级缓存的管理中。明白了二级缓存原理和有了这个思路后，很容易配置起 Hibernate 的二级缓存。扩展知识：一个 SessionFactory 可以关联一个二级缓存，也即一个二级缓存只能负责缓存一个数据库中的数据，当使用 Hibernate 的二级缓存后，注意不要有其他的应用或 SessionFactory 来更改当前数据库中的数据，这样缓存的数据就会与数据库中的实际数据不一致。

**18、Spring 的依赖注入是什么意思？给一个 Bean 的 message 属性，字符串类型，注入值为 "Hello" 的 XML 配置文件该怎么写？**

**19、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。

## 20、什么是 spring 的 IOC AOP

## 21、STRUTS 的工作流程！

## 22、spring 与 EJB 的区别！！

## 八. 软件工程与设计模式

### 1、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。

### 2、j2ee 常用的设计模式？说明工厂模式。

总共 23 种，分为三大类：创建型，结构型，行为型

我只记得其中常用的 6、7 种，分别是：

创建型（工厂、工厂方法、抽象工厂、单例）

结构型（包装、适配器，组合，代理）

行为（观察者，模版，策略）

然后再针对你熟悉的模式谈谈你的理解即可。

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。

### 3、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。

## 九. j2ee 部分

### 1、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。

## 2、应用服务器与 WEB SERVER 的区别？

应用服务器：Weblogic、Tomcat、JBoss

WEB SERVER：IIS、Apache

## 3、应用服务器有那些？

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

#### 4、J2EE 是什么？

答：J2EE 是 Sun 公司提出的多层 (multi-tiered), 分布式 (distributed), 基于组件 (component-base) 的企业级应用模型 (enterprise 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 (client tier) 组件, web 层和组件, Business 层和组件, 企业信息系统 (EIS) 层。

一个另类的回答：j2ee 就是增删改查。

#### 5、J2EE 是技术还是平台还是框架？ 什么是 J2EE

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。

#### 6、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 直接与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。

## 7、如何给 **weblogic** 指定大小的内存?

(这个问题不作具体回答，列出来只是告诉读者可能会遇到什么问题，你不需要面面俱到，什么都精通。)

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

## 8、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。

## 9、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。

## 10、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。

## 11、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//helloworld.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。

## 12、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.xml CMP 实体 Bean 一般还需要 weblogic-cmp-rdbms-jar.xml

## 13、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。

## 14、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

### 十. EJB 部分

#### 1、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。

#### 2、简要讲一下 EJB 的 7 个 Transaction Level?

#### 3、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 必须被部署在诸如 Websphere、

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。

#### 4、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)，开放组 (X/Open) 的 XA 接口。

#### 5、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。

#### 6、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。

#### 7、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

## 8、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。

## 十一. webservice 部分

### 1、WEB SERVICE 名词解释。JSWDL 开发包的介绍。JAXP、JAXM 的解释。SOAP、UDDI,WSDL 解释。

Web Service Web 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，以便别的企业能够发现的访问协议的实现标准。

### 2、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。

### 3. Linux

### 4、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。

## 5. 问得稀里糊涂的题

## 6、四种会话跟踪技术

会话作用域 Servlets JSP 页面描述

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

## 7、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

## 十二. 电话面试等

### 1、请用英文简单介绍一下自己。

4、WEB SERVICE 名词解释。JSWDL 开发包的介绍。JAXP、JAXM 的解释。SOAP、UDDI,WSDL 解释。

### 2、请把 <http://tomcat.apache.org/> 首页的这一段话用中文翻译一下?

Apache Tomcat is the servlet container that is used in the official Reference Implementation for the [Java Servlet](#) and [JavaServer Pages](#) technologies. The Java Servlet and JavaServer Pages specifications are developed by Sun under the [Java Community Process](#).

Apache Tomcat is developed in an open and participatory environment and released under the [Apache Software License](#). Apache Tomcat is intended to be a collaboration of the best-of-breed developers from around the world. We invite you to participate in this open development project. To learn more about getting involved, [click here](#).

Apache Tomcat powers numerous large-scale, mission-critical web applications across a diverse range of industries and organizations. Some of these users and their stories are listed on the [PoweredBy](#) wiki page.

### 3、美资软件公司 JAVA 工程师电话面试题目

1. Talk about overriding, overloading.
2. Talk about JAVA design patterns you known.
3. Talk about the difference between LinkList, ArrayList and Victor.
4. Talk about the difference between an Abstract class and an Interface.
5. Class a = new Class(); Class b = new Class();  
if(a == b) returns true or false, why?
6. Why we use StringBuffer when concatenating strings?
7. Try to explain Singleton to us? Is it thread safe? If no, how to make it thread safe?
8. Try to explain Ioc?
9. How to set many-to-many relationship in Hibernate?
10. Talk about the difference between INNER JOIN and LFET JOIN.
11. Why we use index in database? How many indexes is the maximum in one table as your suggestion?
12. When ‘Final’ is used in class, method and property, what dose it mean?
13. Do you have any experience on XML? Talk about any XML tool you used , e.g. JAXB, JAXG.
14. Do you have any experience on Linux?
15. In OOD what is the reason when you create a Sequence diagram?

- 1，堆和栈的区别，有一个64k的字符串，是放到堆上，还是放到栈上，为什么？
- 2，什么时候用到接口，什么时候用到抽象类，二者区别
- 3，有一个100万的数组，里边有两个是重复的，如何设计算法找到。
- 4，设计数据库时，n维，如何设计。

例如[省份][城市][网吧]，这是三维关系，它的表也应该有三个，网吧有外键引用城市，城市有外键应用省份，这个规律就是下层的要有一外键去引用上层。

## 第6章 面霸 - C, C++, 嵌入式

Void \*p = malloc  
请计算  
sizeof(A)

11. If i=5, what will be the result of:

```
do
{
    cout<<(--i)--<<" ";
} while(i>=2 && i<5);
A. 4 3 2 1      B. 4 2 1      C. 4 2      D. It won't enter the loop
```

12. In which header file is the function isalpha()?

A. string.h B. stdio.h C. conio.h D. ctype.h

13. Which one is correct?

- A. int a; a=new int(20); X
- B. int a; a=new int[20]; X
- C. int \*a; a=new int(20) X
- D. int \*a; a=new int[20]; ✓

14. How do you access the last cell of "int arr[123]"?

A. arr B. arr[122] C. arr[123] D. arr[124]

15. What does the following do:

void afunction(int \*x)

```
{  
    x=new int;  
    *x=12;  
}
```

int main()

{

```
    int v=10;  
    afunction(&v);  
    cout<<v<<endl;  
    return 0;
```

- A. Outputs 10 B. Outputs 12 C. Outputs the address of v D. Complier error

## 二、填空题(16-20)

16. int a 与 “零值” 比较的 if 语句为: if(a==0) if(a!=0)  
则 bool flag 与 “零值” 比较的 if 语句为 if(flag) if(!flag)

17. 以下为 Windows NT 下的 32 位 C++ 程序, 请计算 sizeof 的值:

void Func(char str[100])

{

请计算

sizeof(str)= 4

# — 信核数据笔试题

## B 卷

### 一、单选题(1-15)

(1) 1. If you have "int main(int argc,char\* argv[])" and invoke your program this way: prog.exe

a.out -l 2 -g -x 3 4 What will be the value of argc?

- A.7      B.3      C.2      D.0

(2) 2. If we have:

int f(int x)

```
{
    if(x>2)      4+f(3)+3+f(2)
        return x + f(-x);  5+f(4)
    else
        return 0;
}
```

What will be the result of: cout<<f(5);

- A.6      B.9      C.10      D.12

(3) 3. What will be the result of: cout<<(5<<3);?

- A.53      B.40      C.35      D.0

$$\begin{array}{r} 101000 \\ 321684 \quad 21 \\ \hline 321684 = 40 \end{array}$$

(4) 4. What is the default access permission for members in a struct?

- A. public      B. protected      C. private      D. depends

(5) 5. Which is not valid in C?

- A. class aclass{public :int x;};      B. /\*A comment\*/ ✓  
C. print("Hi!");      D. char x=12;

(6) 6. What does strcat(an\_array,"This"); do?

- A. Copies "This" into an\_array      B. Adds "This" to the end of an\_array  
C. Compares an\_array and "This"      D. Adds "This" to the front of an\_array

(7) 7. Evaluate the following :22%5

- A.0      B.1      C.2      D.4

(8) 8. If all is successful, what should main return?

- A.-1      B.0      C.1      D.void

(9) 9. Which is not a valid keyword:

- A. public      B. protected      C. private      D.guarded

(10) 10. What header file allows file I/O with streams?

- A. fstream      B. fileio.h      C.iostream      D. ifstream

6. 标准模板库由哪些部分组成？它们之间的关系是怎么样的？

7. TCP 和 UDP 两种协议各有什么特点？它们各自应用于哪些方面？

2.

8. 除了 C/C++ 语言外，你还知道哪些语言？

Java C# delphi

9. 软件测试有哪些方法？你是如何理解单元测试的？

黑盒 白盒

10. 使用模板完成二分法查找函数 template<class T>

T\* bsearch(const T& key, T\* p, size\_t count)

19. 构造函数为什么不能是虚拟的

~~虚函数和构造函数都在子类函数表中，而构造函数只在创建对象时调用一次，这个时候虚函数的指针还没有指向函数表~~

20. 软件开发过程中，概要设计主要的目的是什么？

二. 简答

1. 使用 MFC AppWizard 创建一个名称为 Test 多文档程序，AppWizard 自动创建了哪些主要的类？每个类的主要作用是什么？这些类之间的关系如何？

2. 列举关键字 static 在 C++ 中作用。

~~在类的对像没有被创建时就可以使用 static  
修饰的变量或函数，是基于类而不仅仅是对象。~~

3. 请简单说明 C++ 编译器可能如何实现多态，在你所知道的 C++ 编译器中哪些使用了这些方法？

~~内联函数~~

~~虚基类~~

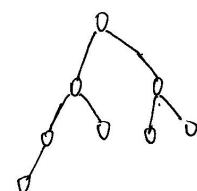
4. 在 win32 编程中，线程之间同步的方法有哪些？进程之间同步呢？

~~信号量~~

~~临界资源~~

5. 完全平衡二叉树的定义是什么？当向完全平衡二叉树中插入一个节点，以至于它不再平衡时，简单说明如何使它重新平衡。

~~真插入？~~



```
    printf("%d,", CA::m_Num); 3
    printf("%d", sizeof(CA));
}
```

2 3 3 8

10. 有如下具有异常规格描述的函数 f,

```
void f() throw(std::bad_alloc, std::bad_cast);
```

写出调用函数 f, 并捕获其抛出异常的代码

```
try {
    f();
} catch (exception &x) {
}
```

11. 表达式  $a * (b + c) - d / e$  的逆波兰表达式(后缀表达式)为 \_\_\_\_\_

2

12. 相比 C 语言,C++语言最重要的特点是 继承、封装、多态。

13. 在 COM 编程中, 所有的 COM 接口都必须直接或间接从接口 \_\_\_\_\_ 继承而来, 该接口提供了 \_\_\_\_\_、\_\_\_\_\_ 和 \_\_\_\_\_ 三个方法。

14. 在 Win32 编程中, 用来表示窗口对象的数据类型是 \_\_\_\_\_, 在一个窗口的生命周期中它的处理过程收到的第一个消息是 \_\_\_\_\_, 当窗口需要被重新绘制时它通常会收到 \_\_\_\_\_ 消息。

15. 在 MFC 中, 如果一个类想使用 MFC 提供的消息映射进行消息处理, 则它必须从类继承, 除了实现了消息映射功能之外, 该类还提供了实现 \_\_\_\_\_ 功能的支持。

16. 在 Win32 API 中, 函数 SendMessage 和 PostMessage 的区别是 \_\_\_\_\_

17. 在一名为 Test.dll 的动态链接库文件中, 有一个名为 TestFunc 的导出函数, 其原型如下:

```
int __stdcall TestFunc(void);
```

请写出动态加载该 DLL 并调用 TestFunc 函数的代码

18. win32 下进程间的通讯方法有(尽可能多的列举)

```

void f2()
{
    printf("CB::f2()\n");
}
};

void main()
{
    CB b;
    CA& a = b;
    b.f1();
    b.f2();
    a.f1();
    a.f2();
}

```

CB::f1()

CB::f2()

CA::f1()

CA::f2()

9. 写出在win32环境下面代码的执行结果

```

class CA
{
public:
    CA() : p(0)
    {
        m_Num++;
    }
    CA(const CA& a)
    {
    }
    ~CA()
    {
        m_Num--;
    };
    const CA& operator=(const CA& a)
    {
        m_Num++;
        return *this;
    }
};

public:
    char* p;
    static int m_Num;
};

int CA::m_Num = 0;

```

```

int main()
{
    CA a;
    CA b = a;
    printf("%d, ", CA::m_Num); 2
}

    CA c;
    printf("%d, ", CA::m_Num); 3

```

```

        mov v, ebx
    }
    printf("在CPU中有%d个核心\n", _____);
    return 0;
}

```

5. 执行下面代码后，打印结果是

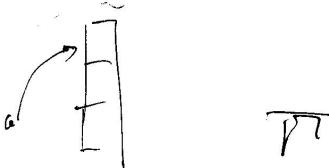
```
#include "stdio.h"
int main()
```

```
{
    int a = 5;
    a += (++a);   a = a + (++a) +      a = 6+6=12
    printf("%d", a);
    a += (a++);
    printf("%d", a);
    a = a + (a++)
}
```

12, 25

$8 + 6 + 6$

$a = a + (a++) + (a++)$



PT

6. 写出浮点数 x 与 0 比较的语句

$x > 0 \quad (x - 0.0000)^+$   $x < 0 \quad (0.0000 - x)^-$

7. 函数 Add 定义如下：

```
int Add(int a,int b)
{
    return a + b;
}
```

请定义一个函数指针变量 pFunc，使其指向 Add

int (\*pFunc)(int, int)

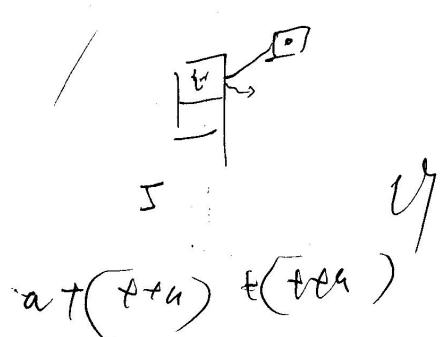
8. 写出下面程序执行的结果

```

class CA
{
public:
    virtual void f1() const
    {
        printf("CA::f1()\n");
    }
    void f2()
    {
        printf("CA::f2()\n");
    }
};

class CB : public CA
{
public:
    virtual void f1() const
    {
        printf("CB::f1()\n");
    }
};

```



$a + (t+a) + (t+a)$

中控技术研发中心软件工程师笔试试卷

注意事项:

- 1、本试卷共 6 页，考试时间 1 小时。
- 2、请写上姓名，考试结束后，将试卷、草稿纸一并交回。

姓名: 于月光 成绩: \_\_\_\_\_

1. 解释如下一行代码的含义:

int (\*a)[10];

a是一个包含有10个整形元素的数组的指针

2. 变量 a 的定义如下，请定义一个指针变量 p 指向该变量:

int const a;

const int \* p = &a;

3. 写出下面代码的执行结果

```
#include <stdio.h>
void swap(int a, int b)
{
    int tmp = a;
    a = b;
    b = tmp;
}
int main()
{
    int a = 10;
    int b = 20;
    swap(a, b);
    printf(" %d, %d", a, b); 10, 20
    return 0;
}
```

4. 在 Intel 的 x86 多核心 CPU 中，常常使用 cpuid 指令来检测物理 CPU 中有几个核，在寄存器 eax 等于 0x01 时调用 cpuid 指令，指令执行完成后，寄存器 ebx 中 bit16~bit23 存放着该 cpu 中核心的个数，填写下面程序的空白，以打印出你机器上 cpu 中核心的个数。

```
#include "stdio.h"
int main()
{
    unsigned long v;
    __asm
    {
        mov eax, 1
        cpuid
    }
    ebx
```



10. 写一个函数找出一个整数数组中，第二大的数（编码实现）

11. 写一个排序算法（编码实现）

附加题：比较 C++ 中的 4 种类型转换方式？



### C++笔试 A 卷

1. 什么是“引用”？声明和使用“引用”要注意哪些问题？

2. “引用”与指针的区别是什么？

3. #include<file.h> 与 #include "file.h" 的区别？

4. 面向对象的三个基本特征，并简单叙述之？

5. struct 和 class 的区别？

6. `#define DOUBLE(x) x + x;`  
`i = 5 * DOUBLE(5);`  
i 是多少？  
30

7. 简述数组与指针的区别？

8. 类成员函数的重载、重写(overried, 有的书也叫做“覆盖”)和隐藏的区别？

9. There are two int variables: a and b, don't use if, ?, :, switch or other judgement statements, find out the biggest one of the two numbers.

$$i = a - b$$



## 同花顺软件开发工程师测试考题

### 说明:

- 1、本考题答题时间为 100 分钟;
- 2、本考题重点考察基础的编程知识和解决问题的能力。

### 测试考题:

1. 在 C++ 语言中 什么函数不能声明为虚函数? 为什么?
- (2) 冒泡排序算法和快速排序算法的时间复杂度分别是什么?  
 $O(N^2)$      $O(N \log N)$
3. C 语言中 局部变量能否和全局变量重名? 如果能, 如何解决冲突? 若不能, 为什么?
- (4) Heap 和 Stack 的有什么区别?
5. 请简述数组和链表数据结构的特点以及适用场合。
6. 写出 float x 与 “零值” 比较的 if 语句。(任何编程语言都可以)
7. 写一个函数, 输出指定数字的斐波那契数。(可以不考虑溢出问题, 但请注意函数性能)  
(斐波那契数列的排列是: 1, 1, 2, 3, 5, 8, 13 即后一个数等于前面两个数的和)
8. 请编写字符串拷贝函数 (不能调用 C/C++字符串函数库)  
函数原型: `char* strcpy(char* strDest, const char* strSrc)`
9. 请用你熟悉的开发语言实现一个单例模式 Singleton。(此题为选做题)
10. 请描述下你对 MVC 模式的理解, 以及 MVC 模式的优点和不足?
11. 请估算一下在大学期间您总共写过多少行代码? 如果不包括课后作业和老师布置的编程任务以为写过多少行代码? 你认为最得意的代码是什么? 请简单描述一下。
12. 请估算每天使用同花顺的用户有多少, 需要多少台服务器来支撑这些用户。你是如何估算的?
13. 如果有一天, 领导给你分配了一个你工作职责之外的任务, 这个时候你会怎么做? (此题为选做题)
14. 大部分的搜索引擎主页都千篇一律, 如果需要你设计出一个与众不同但是同样能够吸引客户的搜索引擎主页, 你会怎么做?
15. 简单写一下你最近关注的和 IT 行业相关的新闻、技术、产品、人物等。(此题为选做题)
16. 假如有一天你一个人在家里。想象一下下面的事情同时发生: 两个月大的孩子在哭闹, 煤气炉上开水开了, 衣服没收天开始下大雨, 门外有人敲门, 此刻电话响了。你该怎么做? 请写出你处理这些事情的先后顺序。

(反面还有试题)

支持环保, 请不要在试卷上涂画, 以便循环利用 ☺

```
for (u = NULL; p; p = p->next)
    first_insert((3));

return u;
}

void print_link(NODE *p)
{
    for( ; (4)
        printf( "%d\t", p->val);

    printf( "\n");
}

void free_link(NODE *p)
{
    NODE *u;
    while (p != NULL)
    {
        u = p->next;
        free(p);
        (5);
    }
}

void main()
{
    NODE *link1, *link2;
    int i;
    link1 = NULL;
    for (i = 1; i <= 10; i++)
        first_insert(&link1, i);

    link2 = reverse_copy(link1);
    print_link(link1);
    free_link(link1);
    print_link(link2);
    free_link(link2);
}
```

2. 阅读下列函数说明和 C 代码，填空 (n) 处的字句。

● [函数说明]

函数 strcmp() 是比较两个字符串 s 和 t 的大小。若  $s < t$  函数返回负数；若  $s = t$  函数返回 0；若  $s > t$  函数返回正数。

[函数]

```
int strcmp(char *s, char *t)
{
    while (*s && *t && (1))
    {
        s++;
        t++;
    }
    return (2);
}
```

3. 阅读下列程序说明和 C 代码，填空 (n) 处的字句。

● [程序说明]

本程序中的函数 first\_insert() 的功能是在已知链表的首表元之前插入一个指定值的表元；函数 reverse\_copy() 的功能是按已知链表复制出一个新链表，但新链表的表元链接顺序与已知链表的表元链接顺序相反；函数 print\_link() 用来输出链表中各表元的值；函数 free\_link() 用来释放链表全部表元空间。

[程序]

```
#include <stdio.h>
#include <malloc.h>
```

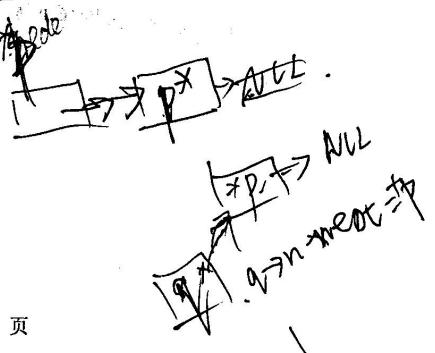
```
typedef struct node
{
    int val;
    struct node *next;
} NODE;
```

```
void first_insert(NODE **p, int v)
```

```
{
    NODE *q = (NODE *) malloc(sizeof(NODE));
    q->val = v;
    (1) q->next = *p
    *p = q (2) next
}
```

```
NODE *reverse_copy(NODE *p)
```

```
{
    NODE *u;
```



## 招聘试题

招聘职位：软件开发

1. 请问运行 Test 函数会有什么样的结果？

```
void GetMemory(char *p)
{
    p = (char *)malloc(100);
}
void Test(void)
{
    char *str = NULL;
    GetMemory(str);
    strcpy(str, "hello world");
    printf(str);
}
```

请问运行 Test 函数会有什么样的结果?  
答：

```
char *GetMemory(void)
{
    char p[] = "hello world";
    return p;
}
void Test(void)
{
    char *str = NULL;
    str = GetMemory();
    printf(str);
}
```

请问运行 Test 函数会有什么样的结果?  
答：

```
Void GetMemory2(char **p, int num)
{
    *p = (char *)malloc(num);
}
void Test(void)
{
    char *str = NULL;
    GetMemory2(&str, 100);
    strcpy(str, "hello");
    printf(str);
}
```

请问运行 Test 函数会有什么样的结果?  
答：

```
void Test(void)
{
    char *str = (char *) malloc(100);
    strcpy(str, "hello" );
    free(str);
    if(str != NULL)
    {
        strcpy(str, "world" );
        printf(str);
    }
}
```

请问运行 Test 函数会有什么样的结果?  
答：

for(k=0;k<4;k++) printf("%d ",v[k]);  
}

A)0 1 2 3      B)0 1 1 8

C)5 6 7 8

D)0 2 4 6

④ 15 下面程序输出结果是

main ()

{

Int k = 0; char c = 'A';

Do{

Switch(c++)

{

Case 'A': k++; break;

Case 'B': k--;

Case 'C': k+=2; break;

Case 'D': k=k%2; continue;

Case 'E': k=k\*10; break;

Default: k=k/3;

}

k++;

}while(c<'G');

Printf("k=%d\n", k);

}

A k=3; B k=4; C k=2; D k=0

## 二、简答题

1. 请列举 ASP.NET 4.0 框架能支持的几种开发语言。

答：C# 编程语言，HTML 语言，VBScript 编程语言

2. 在 VS2010 提供的高级开发技术中，微软为 UI 的开发提供了一种具有超强震撼力的技术 WPF，请简要描述其主要特性。

答：提供了统一的编程模型、语言和框架，能做到分离界面设计人员与开发人员的工作，并且提供全新的多媒体交互用户界面。

```
int a[][4]={1,2,3,4,5,6,7,8,9,10,11,12},(*p)[4];
p=a;
```

```
printf("%d\n",*(p+2));
```

- A、1              B、5              C、3              D、7

(C) 9. C 语言规定：在一个源程序中， main 函数的位置是

- A、必须在最开始              B、必须在系统调用的函数的后面  
 C、可以任意              D、必须在最后

(A) 10. 一个 C 程序的执行是从

- A、本程序的 main 函数开始，到 main 函数结束  
 B、本程序的第一个函数开始，到本程序文件的最后一个函数结束  
 C、本程序的 main 函数开始，到本程序文件的最后一个函数结束  
 D、本程序文件的第一个函数开始，到本程序 main 函数结束

(C) 11. 判断 char 型变量 ch 是否为大写字母的正确表达式

- A) 'A'<=ch<='Z'              B) (ch>='A')and (ch <='z')  
 C) (ch >= 'A')&&(ch <='Z')              D) ('A'<=ch )||(‘Z’>=ch)

(B) 12. 以下程序的输出结果是

```
main()
{
    int i, k, a[10], p[3]; k=5;
    for(i=0;i<10;i++) a[i]=i;
    for(i=0;i<3;i++) p[i]=a[i*(i+1)];
    for(i=0;i<3;i++) k+=p[i]*2;
    printf("%d\n",k);
}
```

- A)20              B)21              C)22              D)23

(C) 13. 下面程序的运行结果是

```
main()
{
    char ch[7]={"65ab21"};
    int i,s=0;
    for(i=0;ch[i]>='0'&&ch[i]<='9';i+=2) s=10*s+ch[i]-'0';
    printf("%d\n",s);
}
```

- A)12ba56              B)6521              C)6              D)62

(B) 14. 以下程序运行后的结果是

```
main()
{ char s[]="12345678"; int v[4]={0,1,1,0},k,i;
for (k=0; s[k];k++)
{
    switch (s[k])
    {case'1': i=0;        case'2': i=1;
     case'3': i=2;        case'4': i=3;
    }
    v[i]++;
}
}
```

(高科工资) 软件工程师笔试题

姓名 苏文华

电话 13675880691

一、选择题

1. 下述循环的循环次数是

```
int k=2;
while (k>0) printf ("%d", k);
k--; printf ("\n");
```

- A、无限次      B、0 次      C、1 次      D、2 次

2. 在下列选项中，没有构成死循环的程序段是

```
A、int i=100;
while (1)
{i=i%100+1;
if (i>100) break;
}
```

```
C、int k=1000;
do {++k;} while (k>=10000);      while (s); --s;
```

3. 若有以下定义和语句，且  $0 \leq i < 10$ ，则对数组元素的错误引用是

```
int a[]={0, 1, 2, 3, 4, 5, 6, 7, 8, 9}, *p, i;
p=a;
```

- A、\*(a+i)      B、a[p-a]      C、p+i      D、\*(&a[i])

4. 下面能正确将字符串"Boy"进行完整赋值操作的语句是

```
A、char s[3]={'B', 'o', 'y'};
C、char s[3]={"Boy"};
B、char s[ ]="Boy";
D、char s[3];
```

s[0]='B'; s[1]='o'; s[2]='y';

5. 以下程序段的输出结果是

```
char arr[ ]="ABCD";
char*ptr;
for (ptr=arr; ptr<arr+4; ptr++)
printf ("%s\n", ptr);
```

- A、ABCD      B、A      C、D      D、ABCD  
BCD  
CD  
D

6. C 语言的 auto 型变量是

- A、存储在动态存储区中。  
B、存储在静态存储区中。  
C、存储在计算机 CPU 的寄存器中。  
D、存储在外存储器中。

7. 测试文件是否结束函数的函数名是

- A、feof      B、EOF      C、eof      D、FEOF

8. 下面程序段的输出结果是

杭州贵仁科技 c/c++笔试题

2、请解释 SQL 语句 begin transaction、commit transaction、rollback transaction 的作用。

3、按数据结构划分，SQL Server 是属于型数据库。

4、DML 的功能是？

5、数据库应用中，编程方法有哪三种？

6、某校学生成绩表 t\_student，有 6 个字段 (i\_id:一个整数，代表这个学生的唯一 ID, c\_name: 学生的姓名, c\_class:所在的班级, c\_teacher:所在班的班主任姓名, c\_subject 考试科目, i\_score:考试成绩)。这种设计不太合理，你会怎么设计？

7、从表 t\_table1 (有字段 i\_id,c\_name,c\_sex,i\_score) 中取出 c\_sex 为男，i\_score 列前十名的 c\_name 字段。

8、有学生信息表 t\_students(i\_id: 学生 ID, c\_name: 姓名, c\_class:班级),学生成绩表 t\_scores(i\_student\_id:学生 ID, i\_subject:科目编号, i\_score:考试成绩)，试编写一条 SQL 语句取回每个 9501 班学生的姓名和个人总分。

9、请列举出数据库编程中连接 SQL Server 数据库的三种不同技术：\_\_\_\_\_、\_\_\_\_\_、\_\_\_\_\_。

10、某用户管理系统的某一客户端需要打开一个后台数据库表 T1，用来增加、删除、修改用户信息；另一客户端同时打开 T1 表，可对所有用户进行实时查询操作。第二个客户端所用的 Recordset 的游标类型应该是\_\_\_\_\_ (Dynamic, Forward Only, Keyset, Static)，锁定类型应该是\_\_\_\_\_ (Batch Optimistic, Optimistic, Pessimistic, Read Only)。

杭州贵仁科技 c/c++笔试题

请问运行 Test 函数会有什么样的结果?

答:

```
4、void Test(void)
{ char *str = (char *) malloc(100);
  strcpy(str, "hello");
  free(str);
  if(str != NULL)
  {
    strcpy(str, "world");
    printf(str);
  }
}
```

请问运行 Test 函数会有什么样的结果?

答:

五、编写 strcpy 函数

已知 strcpy 函数的原型是 `char *strcpy(char *strDest, const char *strSrc);` 其中 strDest 是目的字符串, strSrc 是源字符串。

1、不调用 C++/C 的字符串库函数, 请编写函数 strcpy

1、strcpy 能把 strSrc 的内容复制到 strDest, 为什么还要 char \* 类型的返回值?

六、编写类 String 的构造函数、析构函数和赋值函数

已知类 String 的原型为:

```
class String
{public:
  String(const char *str = NULL); // 普通构造函数
  String(const String &other); // 拷贝构造函数
  String(void); // 析构函数
  String & operator =(const String &other); // 赋值函数
private:
  char *m_data; // 用于保存字符串
};
```

请编写 String 的上述 4 个函数。

七、数据库

1、请列出你所知道的五种关系数据库。

杭州贵仁科技 c/c++笔试题

8、简述进程间通信的几种方式

9、简述 TCP, UDP 通信的区别

四、有关内存的思考题

1、void GetMemory(char \*p)

```
{  
    p = (char *)malloc(100);  
}  
void Test(void)  
{  
    char *str = NULL;  
    GetMemory(str);  
    strcpy(str, "hello world");  
    printf(str);}
```

请问运行 Test 函数会有什么样的结果?

答:

2、char \*GetMemory(void)

```
{  
    char p[] = "hello world";  
    return p;  
}  
void Test(void)  
{  
    char *str = NULL;  
    str = GetMemory();  
    printf(str);}
```

请问运行 Test 函数会有什么样的结果?

答:

3、Void GetMemory2(char \*\*p, int num)

```
{  
    *p = (char *)malloc(num);  
}  
void Test(void)  
{  
    char *str = NULL;  
    GetMemory2(&str, 100);  
    strcpy(str, "hello");  
    printf(str);  
}
```

杭州贵仁科技 c/c++笔试题

一、请填写 BOOL, float, 指针变量 与 “零值” 比较的 if 语句。

(提示: 这里 “零值” 可以是 0, 0.0, FALSE 或者 “空指针”。例如 int 变量 n 与 “零值” 比较的 if 语句为: if (n == 0)、if (n != 0) 以此类推。)

1、请写出 BOOL flag 与 “零值” 比较的 if 语句:

2、请写出 float x 与 “零值” 比较的 if 语句:

3、请写出 char \*p 与 “零值” 比较的 if 语句:

二、以下为 Windows NT 下的 32 位 C++程序, 请计算 sizeof 的值

char str[] = "Hello"; char \*p = str; int n = 10;

请计算

1、sizeof(str) =

2、sizeof(p) =

3、sizeof(n) =

void Func ( char str[100])

{

请计算

sizeof(str) =

}

void \*p = malloc( 100 );

请计算 sizeof(p) =

三、简答题 (25 分)

1、头文件中的 ifndef/define/endif 的作用是什么?

2、#include <filename.h> 和 #include "filename.h" 有什么区别?

3、const 与 static 有什么用途?

4、在 C++ 程序中调用被 C 编译器编译后的函数, 为什么要加 extern “C” 声明?

5、面向对象的三个基本特征, 并简单叙述之?

6、重载 (overload)和重写(overried, 有的书也叫做“覆盖”)的区别?

7、.#define DOUBLE(x) x+x , i = 5\*DOUBLE(5); i 是多少?

\_Node;

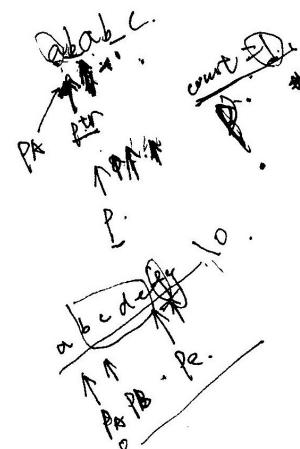
有两个双向循环链表A、B，知道其头指针为：pHeadA, pHeadB，现需将两上链表中date 值相同的结点删除，即该date值的结点同时存在于A、B两链表中，将该date值的所有结点删除。  
请描述算法并编程实现。

八、请编程实现 strcmp 函数。

```
strcmp(char *str1, char *str2) {
    for(;; str1++, str2++) {
        if(*str1 != *str2)
            return (*str1 - *str2);
    }
}
```

九、一语句实现 x 是否为 2 的若干次幂的判断。

十、给一个字符串，例如 “ababc” 要求返回 “ab”。因为 “ab” 连续重复出现且最长。  
请描述算法，并用 C/C++语言写一函数完成该算法，给出复杂度。若要求返回重复出  
现次数最多的子串，请描述算法。





柯林思力

可以

一、一个父类写了一个 **virtual** 函数，如果子类覆盖它的函数不加 **virtual**，也能实现多态？  
在子类的空间里，有没有父类的这个函数，或者父类的私有变量？

有 - 父类

二、完成字符串拷贝可以使用 **sprintf**、**strcpy** 及 **memcpy** 函数，请问这些函数有什么区别，你喜欢使用哪个，为什么？

三、引用与指针有什么区别？

四、全局变量和局部变量在内存中是否有区别？如果有，是什么区别？

五、**char \* const p;**   **char const \* p;**   **const char \*p;**      上述三个有什么区别？

六、假设现有一个单向的链表，但是只知道只有一个指向该节点的指针 **p**，并且假设这个节点不是尾节点，试描述算法并编程实现删除此节点。



~~p = p -> next~~

~~p1 = p~~

~~p = p -> next~~

~~delete(p)~~

七、有双向循环链表结点：

```
typedef struct node
{
    int date;
    struct node *front, *next;
```

} Node;

有两个双向循环链表A、B，知道其头指针为：pHeadA, pHeadB，现需将两上链表中date 值相同的结点删除，即该date值的结点同时存在于A、B两链表中，将该date值的所有结点删除。  
请描述算法并编程实现。

八、请编程实现 strcmp 函数。

```
char * strcmp(char * str1, char * str2) {
    assert(str1 != NULL || str2 != NULL);
    for (; str1 == str2; str1++, str2++);
    return str1 - str2;
}
```

九、一语句实现 x 是否为 2 的若干次幂的判断。

```
int n=8
if (2<<x) == n
```

十、给一个字符串、例如 “ababc” 要求返回 “ab”。因为 “ab” 连续重复出现且最长。  
请描述算法，并用 C/C++语言写一函数完成该算法，给出复杂度。另若要求返回重复出  
现次数最多的子串，请描述算法。

一、一个父类写了一个 **virtual** 函数，如果子类覆盖它的函数不加 **virtual**，也能实现多态？

在子类的空间里，有没有父类的这个函数，或者父类的私有变量？

二、完成字符串拷贝可以使用 **sprintf**、**strcpy** 及 **memcpy** 函数，请问这些函数有什么区别，你喜欢使用哪个，为什么？

三、引用与指针有什么区别？

引用必须有初值，一经定义不能改变。

指针不必有初值，可以

四、全局变量和局部变量在内存中是否有区别？如果有，是什么区别？

五、**char \* const p;**   **char const \* p;**   **const char \*p;**       上述三个有什么区别？

六、假设现有一个单向的链表，但是只知道只有一个指向该节点的指针 **p**，并且假设这个节点不是尾节点，试描述算法并编程实现删除此节点。

七、有双向循环链表结点：

```
typedef struct node
{
    int date;
    struct node *front, *next;
```

3、需要将 RGB 格式的彩色图像先转换成黑白图像。

图像转换的公式如下：

$$Y = 0.299 * R + 0.587 * G + 0.114 * B;$$

图像尺寸 640\*480\*24bit，RGB 图像已经按照 RGBRGB 顺序排列的格式，放在内存里面了。

请编程描述上面这个公式。要求 CPU 运算速度最快。

C++软件工程师面试笔试题

2、在一次扑克牌游戏中，采用单幅扑克牌，每次重新洗牌后都从整副牌中抽出一部分出来，第一次抽出部分记为 firstPork，记录下 firstPork 的字符和花色；然后将该抽出部分重新放入扑克牌中，再次对整副牌进行洗牌，重新洗牌后再从整副牌中抽出一部分出来，第二次抽出部分记为 secondPork，记录下 secondPork 的字符和花色，然后将该抽出部分重新放入扑克牌中，再次对整副牌进行洗牌，在进行第三次操作，第三次抽出部分记为 thirdPork，记录下 thirdPork 的字符和花色。编程如下：

```
#define NUM 100 //预定义字符串的个数
```

```
typedef struct PORK{
```

```
    unsigned char letter; //字符  
    unsigned char suit; //花色
```

```
}PORK;//扑克牌
```

```
struct PORK firstPork[num]; //第一堆扑克牌
```

```
struct PORK secondPork[num]; //第二堆扑克牌
```

```
struct PORK thirdPork[num]; //第三堆扑克牌
```

```
char suit_all[4]={'S','H','C','D'}; //每张扑克牌可能出现的四种花色
```

```
char letter_all[13]={'A','K','Q','J','T','9','8','7','6','5','4','3','2'}; //每张扑克牌可能出现的 13 种字符  
加入三次抽牌的示例如下：
```

```
firstPork[11].letter={'A','K','Q','J','T','9','8','7','6','5','4'};
```

```
firstPork[11].suit={'S','H','C','D','S','H','C','D','S','H','C'};
```

```
secondPork[13].letter={'7','6','5','4','3','2','J','K','A','T','9','J','A','Q','T'};
```

```
secondPork[13].suit={'D','S','H','C','D','S','H','C','D','S','H','C','D'};
```

```
thirdPork[14].letter={'5','4','3','2','J','K','A','T','9','J','A','J','Q','T'};
```

```
thirdPork[14].suit={'H','C','D','S','H','C','D','S','H','C','D','S','H','C'};
```

请用链表处理，根据重复部分及其重复部分的方向将 firstPork, secondPork, thirdPork 三个字符串合成一个字符串。



Uwaysoft Technical Development Co., Ltd.

2. 打印出 1000 以内的所有的“水仙花数”中最大一个数，并算出各个位的和值，所谓“水仙花数”是指一个三位数，其各位数字立方和等于该数本身。例如：153 是一个“水仙花数”，因为  $153=1^3+5^3+3^3$ 。使用 C/C++ 完成。

```
#include <stdab.h>
#include <math.h>
int main()
{
    int i, sum=0, j=0;
    int ary[999] = {0};
    for (j=100; j<1000; j++)
    {
        int one, ten, has;
        int temp = j;
        one = temp % 10;
        temp = temp / 10;
        ten = temp % 10;
        has = temp / 10;
        temp = one * one * one + ten * ten * ten
               + has * has * has;
        if (temp == j)
            ary[j] = i;
        sum += one;
    }
    printf("max=%d, sum-one=%d\n", ary[999], sum);
}
```

写个求二叉树深度的算法：

```
int BTreDepth(TreeNode *bt)
{
    if (bt == NULL)
    {
        left = 1 + BTreDepth(bt->left);
        right = 1 + BTreDepth(bt->right);
        return (left > right) ? left : right;
    }
    else
        return 0;
}
```

8. 写一个函数，找出一个字符串中最长的重复子串。“t1t1”结果就是 t1. “cabcabca”结果就是 cab 或者 abc 或者 bca。

函数 char \* FoundLong(const char \*str, char \*result)

```

Void GetMemory2(char **p, int num)
{
  *p = (char *)malloc(num);
}
void Test(void)
{
  char *str = NULL;
  GetMemory(&str, 100);
  strcpy(str, "hello");
  printf(str);
}
  
```

请问运行 Test 函数会有什么样的结果?

```

void Test(void)
{
  char *str = (char *) malloc(100);
  strcpy(str, "hello");
  free(str);
  if(str != NULL)
  {
    strcpy(str, "world");
  }
  printf(str);
}
  
```

请问运行 Test 函数会有什么样的结果?

#### 五、编写 strcpy 函数 (10 分)

已知 strcpy 函数的原型是

char \*strcpy(char \*strDest, const char \*strSrc);

其中 strDest 是目的字符串, strSrc 是源字符串

(1) 不调用 C++/C 的字符串库函数, 请编写函数 strcpy

(2) strcpy 能把 strSrc 的内容复制到 strDest, 为什么还要 char \* 类型的返回值?

#### 六、编写类 String 的构造函数、析构函数和赋值函数 (25 分)

已知类 String 的原型为:

```

class String
{
public:
  String(const char *str = NULL); // 普通构造函数
  String(const String &other); // 拷贝构造函数
  ~String(void); // 析构函数
  String & operator =(const String &other); // 赋值函数
private:
  char *m_data; // 用于保存字符串
  
```

}

```
// 第二个
if (condition)
{
    for (i=0; i<N; i++)
        DoSomething();
}
else
{
    for (i=0; i<N; i++)
        DoOtherthing();
}
```

优点：  
缺点：

优点：  
缺点：

#### 四、有关内存的思考题（20分）

```
void GetMemory(char *p)
{
    p = (char *)malloc(100);
}

void Test(void)
{
    char *str = NULL;
    GetMemory(str);
    strcpy(str, "hello world");
    printf(str);
}
```

请问运行 Test 函数会有什么样的结果？

```
char *GetMemory(void)
{
    char p[] = "hello world";
    return p;
}

void Test(void)
{
    char *str = NULL;
    str = GetMemory();
    printf(str);
}
```

崩溃

请问运行 Test 函数会有什么样的结果？

Win32 API —— BOOL ShowWindow(HWND hWnd, int nCmdShow);  
问题：为什么 CWnd:: ShowWindow 函数少了一个参数也能实现相同的功能

答：

这体现了类对于变量和函数共同封装，CWnd 类中为记录窗口句柄定义了类的成员变量，因此在调用 ShowWindow 函数时，不用传入 HWND 参数，而是直接使用对象中保存的窗口句柄。

### 三、程序题（共 30 分）

1、(14 分) 在计费系统的预处理程序中，对话单进行格式转换时，需要使用 strcpy 函数已知 strcpy，此函数的原型是

char \*strcpy(char \*strDest, const char \*strSrc);

其中 strDest 是目的字符串，strSrc 是源字符串。编写 strcpy 函数

- (1) 不调用 C++/C 的字符串库函数，请编写函数 strcpy
- (2) strcpy 能把 strSrc 的内容复制到 strDest，为什么还要 char \* 类型的返回值？

2、(16 分) 在电信业务的后台处理程序中，经常会涉及到处理字符串，除了用 char \*处理字符串之外，C++还为我们提供了封装了的字符串类 string，其本质也是用一个动态数组来保存字符串，类 String 的原型为：

```
class String
{
public:
    String(const char *str = NULL); // 普通构造函数
    String(const String &other); // 拷贝构造函数
    ~String(void); // 析构函数
    String & operator =(const String &other); // 赋值函数
private:
    char *m_data; // 用于保存字符串
};
```

请编写 String 的上述 4 个函数普通构造函数、拷贝构造函数、析构函数和赋值函数。

15938700170

```

{
  char *pstr = strfun();
  printf("%s\n", pstr); //printf 语句 2
}
问题 1：printf 语句 1 和 printf 语句 2 哪个能在屏幕上正在打印出来？
问题 2：如果不能正常在屏幕上打印出字符串，请说明原因。
问题 3：如果不修改 strfun 的声明，请问该如何修改上述程序的错误。
答：
问题 1：语句 1 可以正常打印，语句 2 不能正常打印；
问题 2：语句 2 使用的指针所指向的内存空间 str[20]，在函数 strfun 返回时已经被释放了；
问题 3：可以将函数 strfun 中的语句 char str[20]; 改为 char *str = new char[20];

```

4、(7 分) 下面是交换两个 double 型数据的函数，

```

swap( double* p1, double* p2 )
{
  double *p;
  *p = *p1;
  *p1 = *p2;
  *p2 = *p;
}
main()
{
  double a = 0.1;
  double b = 0.2;
  swap( &a, &b );
}

```

请找出上述代码的错误，指出错误的原因，并改正。

答：

函数 swap 中混淆了 double 型指针与 double 型变量的差别，对于一个未初始化的指针访问其内存空间是非常危险的。对 swap 函数修改如下

```

swap( double* p1, double* p2 )
{
  double p;
  p = *p1;
  *p1 = *p2;
  *p2 = p;
}

```

5、(7 分) 如果 Win32 程序的消息处理函数的定义由

```

LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam,
LPARAM lParam)

```

修改为

```

LRESULT CALLBACK WndProc(MSG msg)
//MSG 结构体中的成员变量包含有 hWnd、message、wParam 和 lParam 等

```

这种修改可行吗？为什么？

答：

不能修改，因为 WndProc 是回调函数，操作系统预先定义了函数的输入参数和返回值。

6、(5 分) MFC 中，CWnd 类作为所有窗体封装类的根类，它定义的一些成员函数名称与 Win32 API 函数名称完全相同，且功能也完全相同，如 ShowWindow、UpdateWindow 等。但函数参数却不同，如：

```

CWnd 类 —— BOOL ShowWindow( int nCmdShow );

```

```
void Derived::eval();  
  
* 8、已知下列类层次结构，他们都定义了缺省的构造函数（ BC ）
```

```
class X{};  
class A{};  
class B: public A{};  
class C: private B{};  
class D:public X,public C{};  
D * pd = new D;  
对于下列哪些转换是不允许的?  
A. X * px = pd;  
B. B * pb = pd;  
C. A * pa = pd;  
D. C * pc = pd;
```

9、在对语音话单文件进行取话单字段信息操作时，需要用 lseek 函数，在 lseek 函数的 (int filedes, off\_t offset, int whence) 参数中，若 whence 参数为 SEEK\_CUR，则 offset 参数代表下列的那个含义（ B ）

- A. 将该文件的位移量设置为距文件开始处 offset 个字节。
- B. 将该文件的位移量设置为其当前值加 offset，offset 可为正或负。
- C. 将该文件的位移量设置为文件长度加 offset，offset 可为正或负。
- D. 将该文件的位移量设置文件起始位置。

10、在打开一个 ASCII 文本格式的语音清单文件时，需要用到 open 函数，在 open 函数的 oflag 参数中，下面的那个选项代表“若此文件不存在则创建它”的含义（ B ）

- A. O\_APPEND
- B. O\_CREAT
- C. O\_EXCL
- D. O\_TRUNC

## 二、问答题（共 38 分）

1、(5 分) 头文件中的 ifndef/define/endif 有什么作用？

答：防止该头文件被重复引用，避免变量、类型等被重新定义。

2、(6 分) const 有什么用途？（请至少说明两种）

答：

(1) 可以定义 const 常量

(2) const 可以修饰函数的参数、返回值，甚至函数的定义体。被 const 修饰的东西都受到强制保护，可以预防意外的变动，能提高程序的健壮性。

3、(8 分) 如下的字符串函数，用于生成一个字符串 “连接号码异常”，并返回它的指针

```
char* strfun()  
{  
    char str[20];  
    strcpy(str, "连接号码异常");  
    printf("%s \n", str); //printf 语句 1  
    return str;  
}  
main()
```

```
ListNode* find_midlist(ListNode* head)
{
    //请填充
}
```

六、 编写一个函数，作用是把一个 char 组成的字符串循环右移 n 个。比如原来是

“abcdefghi”如果 n=2，移位后应该是“hiabcdefgh”

函数头是这样的：

```
//pStr 是指向以'0'结尾的字符串的指针
//steps 是要求移动的 n
void LoopMove ( char * pStr, int steps )
{
    //请填充...
}
```

}

七、 写一个函数返回  $1+2+3+\dots+n$  的值（假定结果不会超过长整型变量的范围）

$\Delta = i+i+1+i+2+\dots$

~~rest = rest -~~

八、 你如何理解 MVC。简单举例来说明其应用。

## 陆虎科技 C/C++岗位 笔试题

完成时间：20—30分钟（不超过30分钟）

姓名： 结果：建议复试 淘汰

一、 static有什么用途？

二、 分别给出 BOOL, int, float, 指针变量 与“零值”比较的 if 语句（假设变量名为 var）

三、 请写出下列代码的输出结果内容

```
main()
{
    int a, b, c, d;
    a=10;
    b=a++; b=(0
    c=++a; c=1)
    d=10*a++; d>(10
    printf("b, c, d: %d, %d, %d", b, c, d);
    return 0;
}
```

四、 下面代码是否存在问题，如有请指出。

```
void swap(int *p1, int *p2)
{
    int *p; 写错了
    *p = *p1;
    *p1 = *p2;
    *p2 = *p;
}
```

五、 以下给出链表结点的数据结构，编写函数find\_midlist返回链表的中间元素

```
typedef struct _list_node
{
    double keyVal;
    struct _list_node *next;
} ListNode;
```

Return 0;

}

2. 以下程序的功能是输出字母和它的 ASCLL 码, 请填写完整。

```
#include<iostream>
#include <cstdio>
Using namespace std;
Int main()
{
    Char c=a;
    Int l;
    _____;
    _____;
    _____;
    Return 0;
}
```

五. 基本操作 (第一题 10 分, 第二、三题 20 分, 总分 50 分)

1. 编写一个简单的 C++ 程序, 已知输入三个数, a、b、c, 要求将三个数按从大到小的顺序输出。
2. 从键盘输入一个十进制的整数, 将其转换为八进制、十六进制整数, 并输出转换结果。
3. 从键盘输入半径, 然后计算并输出球踢得表面积和体积。

等\_\_\_\_\_;

3. 所谓符号常量，是指用标示符代表一个常量，符号常量名一定为大写，以表示同变量名相区别\_\_\_\_\_；
4. 对于任意一个变量，一旦被指定为某一确定类型后，该变量在程序运行时所占用的存储空间的多少和所能参加的运算类型便已确定\_\_\_\_\_；
5. 若 x 为整形变量，j 为实型变量，当执行 `x=(int) j;` 语句后，j 也变为整形变量\_\_\_\_\_；
6. 由于计算机的计算精度很高，所以在 C++ 中计算 `1.0/3*3` 的结果肯定等于 `1_____;`
7. 一个整数的补码和其相对应的负数的补码是相同的\_\_\_\_\_；
8. 一个浮点型数据的表示形式和在内存中的存放形式都只有一种，那就是指数形式\_\_\_\_\_。
9. 因为逗号表达式的值等于其最后一个表达式的值，因此计算逗号表达式时，直接看最后一个表达式即可\_\_\_\_\_。
10. 自加运算符“++”的前置和后置都是使原有变量的值加 1，因此它们在表达式中运算时结果往往也是一样的\_\_\_\_\_。 X

#### 四. 程序补写（每小题 5 分，共 10 分）

1. 序的功能是实现大小写字母的转换，请填写完整程序。

```
#include<iostream>
#include <cstdio>
Using namespace std;
Int main()
{
    Char c1,c2;
    _____;
    _____;
    _____;
    _____;
    _____;
```

1. 标准 C++ 规定 main() 必须声明为 \_\_\_\_\_ 类型。
2. C++ 的标准库函数和类必须在命名空间 \_\_\_\_\_ 中进行声明。
3. C++ 中，输入和输出流对象分别是 \_\_\_\_\_ 和 \_\_\_\_\_。
4. C++ 的主要特点表现在两个方面：一是 \_\_\_\_\_；二是支持 \_\_\_\_\_ 的方法。
5. 在 C++ 中，输入/输出流有一些控制符，其中，其中用于设置输出宽度的是 \_\_\_\_\_；用于设置填充字符的是 \_\_\_\_\_；用于设置浮点数精度的是 \_\_\_\_\_。
6. 下列语句的执行结果是 \_\_\_\_\_。

i=3;

Cout<<++i; 4

Cout<<i++; 4

Cout<<i; 5 8, 8x7+7.63

7. 若采用十进制的表示形式，则 077 为 \_\_\_\_\_，0111 为 \_\_\_\_\_，  
0x29 为 \_\_\_\_\_，0xab 为 \_\_\_\_\_。

8. 若 a=4，逻辑表达式 !a 的值为 \_\_\_\_\_；若 a=4、b=5，a&&b 的值为 \_\_\_\_\_；a||b 的值为 \_\_\_\_\_；!a||b 的值为 \_\_\_\_\_；而表达式 4&&0||2 的值为 \_\_\_\_\_。

9. 设 a、b、c 为整形变量，且 a=2，b=3，c=4，当执行语句 a\*=16+(b++)-(++c)；后，a 的值是 \_\_\_\_\_。 19-5=14

10. 若有定义语句：

Int x=3, y=2; float a=2.5, b=3.5;

则下列表达式的值是 \_\_\_\_\_。

$$(x+y) \% 2 + (int) a / (int) b$$

$$9 \% 2 = 1 + 1$$

### 三. 判断题题 (每小题 1 分, 共 10 分)

1. 在 C++ 源程序中，注释使用越多，编译后的可执行程序的执行效率越低 \_\_\_\_\_；
2. 在 C++ 程序中，不能使用 C 语言程序中的函数和命令，比如 printf 和 scanf

7. 若  $x$ 、 $i$ 、 $j$  和  $k$  都是 int 型变量，计算以下表达式后， $x$  的值是（）。

$$x = (i=4, j=16, k=32)$$

- A. 4  
B. 16  
C. 32  
D. 52

8. 若有以下定义语句：

```
Int a=7; float x=2.5, y=4.7;
```

则表达式  $x+a \% 3 * (\text{int})(x+y) \% 2 / 4$  的值是（）。

- A. 2.500000  
B. 2.750000  
C. 3.500000  
D. 不确定

9. 下面的程序运行结果是（）。

```
#include<iostream>
```

```
Using namespace std;
```

```
Int main()
```

```
{
```

```
Int a=2;
```

```
a%=-4-1;
```

```
Cout << a << " ";
```

```
a+=a*-a=a*-3;
```

```
Cout << a;
```

```
Return 0;
```

$$a = a \% 3 = ?$$

$$\begin{aligned} a &= a - a \\ &= 2 - 3 \\ &= -1 \end{aligned}$$

- A. 2,0  
B. 1,0  
C. -1,12  
D. 2,12

10. 下列关于赋值运算的叙述中，正确的是（）。

- A. 把一个浮点型的赋给整形变量时，舍弃浮点型数据的小数部分。  
 B. 将字符型数据赋给整形变量时，将其放在整形变量存储单元最高 8 位。  
 C. 将一个整形数据赋给字符型数据时，将其高 8 位原样送到字符型变量。  
 D. 将无符号型数据赋给长整形时，只需原样送入，高位补 1

二. 填空题。（每小题 1 分，共 10 分）

18. 小明再次做一个界面元素非常多的界面，他完成绘制代码后，却发现界面非常“闪铄”，  
请您帮他分析一下原因，并给出相应的解决方案。

答案：

19. 某赌场老板设赌，使用三个骰子。规则如下：每人押 1-6 中的一个数字，如有一个骰子  
出现这个数字则该人赢 1 倍(即收回 2 倍押金)，如有两个骰子出现这个数字则该人赢 2  
倍(即收回 3 倍押金)，如有三个骰子出现这个数字则该人赢 3 倍(即收回 4 倍押金)。问  
此赌局老板赢钱的概率？

答案：(包含计算过程)

20. 您比较感兴趣的几种棋牌类游戏是什么？请选取一种游戏，详细说明游戏的规则。

14. 某公司网用户无法访问外部游戏服务器 61.172.241.123 , 管理人员在 windows 操作系统下可以使用\_\_\_\_\_判断故障发生在公司网内还是公司网外。

- A. ping 61.172.241.123
- B. tracert 61.172.241.123
- C. netstat 61.172.241.123
- D. arp 61.172.241.123

答案:

15. http 协议是基于 TCP 协议还是 UDP 协议?

16. 小明是个初级程序员,一天领导给他一个任务,要求他做一个基于对话框的应用程序,上面有很多的控件 : CTreeCtrl, CEdit, CButton, CRichEdit, CStatic, CSliderCtrl,CProgressCtrl CListCtrl 小明兴冲冲的布置好所有的界面,结果发现这个程序运行起来没有出现界面就自动退出了,请你帮忙他分析一下出现了什么问题?

答案:

17. 小明的 Bug 终于找到了,领导要求他背景要是一副美丽的风景图,他不会做,请你帮他解决.

答案:

请用 函数重载， 模板来实现相应的代码：  
答案：

13. 已知 VectorA 和 VectorB，其中 VectorA 表示 玩家当前的扑克，VectorB 代表玩家选中，即将打出去的扑克，请完成下面的函数，将 VectorA 中去掉 VectorB 的扑克

```
#include <vector>
Void OnCardOut(vector <int>& vectorA, vector <int>& vectorB)
{
```

```
}
```

D. 该类的静态数据成员变量的值不可修改

答案:

10. 构造函数可否是虚函数, 为什么? 析构函数呢, 可否是纯虚的呢?

答案:

11. 列举几个 stl 中的容器类, 以及简要说明

12. 已知有 3 个 类, 定义如下:

```
class Gameabc1
{
public:
    int m_id;
    char m_username[100];
};

class Gameabc2
{
public:
    int m_id;
    char m_username[100];
    char m_nickname[50];
};

class Gameabc3
{
public:
    int m_id;
    char m_username[100];
    __int64 sr;
};
```

现要求 分别将 3 个类 按照 [类的长度(2 个字节),类的 m\_id,类的内容] 放入 BYTE Buffer[1024];

7. 在配置为 cpu P4 3.0, 显卡 MX440, 分辨率 1024\*768 的机器上, 游戏以 FPS 60 进行全屏半透明渲染, 用 GDI 实现 和用 D3D 实现, 在程序上有什么区别? 对玩家来说, 最大的区别是什么?

答案:

8. 已知 3 个类 O、P 和 Q, 类 O 中定义了一个私有方法 F1、一个公有方法 F2 和一个受保护的方法 F3: 类 P 和类 Q 是类 O 的派生类, 其继承方式如下所示:

```
class P : protected O {…};  
class Q : public O {…};
```

关于方法 F1 的描述中正确的是\_\_(1)\_\_; 关于方法 F2 的描述中正确的是\_\_(2)\_\_;  
关于方法 F3 的描述中正确的是\_\_(3)\_\_。

- (1) A. 方法 F1 无法被访问      (B) 只有在类 O 内才能访问方法 F1  
    C. 只有在类 P 内才能访问方法 F1      D. 只有在类 Q 内才能访问方法 F1  
(2) A. 类 O、P 和 Q 的对象都可以访问方法 F2      B. 类 P 和 Q 的对象都可以访问方法 F2  
    C. 类 O 和 Q 的对象都可以访问方法 F2      D. 只有在类 P 内才能访问方法 F2  
(3) A. 类 O、P 和 Q 的对象都可以访问方法 F3      B. 类 O、P 和 Q 的对象都不可以访问方法 F3  
    C. 类 O 和 Q 的对象都可以访问方法 F3      D. 类 P 和 Q 的对象都可以访问方法 F3。

答案: (1)                        (2)                        (3)

9. 下列关于一个类的静态成员的描述中, 正确的是(多选)\_\_\_\_\_。

- A. 该类的对象共享其静态成员变量的值
- B. 静态成员变量可被该类的所有方法访问
- C. 该类的静态方法只能访问该类的静态成员变量

```
temp = pai[m];
pai[m] = pai[n];
pai[n] = temp;
}
int playerPai[4][5];
memcpy(playerPai,pai,sizeof(playerPai));
其中 playerPai 代表 4 个玩家的 5 张牌
```

答案：

6. 写一个排序算法，将上题中的 `pai[54]` 这个数组 按 大到小排列

答案：

2. 在 windos xp sp2 32 位系统下, 请写出下面的输出结果:

```
char strHomeUrl[] = "http://www.gameabc.com";
static char *p = strHomeUrl;
__int64 len = sizeof(p);
cout<< sizeof(strHomeUrl) << " " << sizeof(p) << " " << sizeof(len);
```

答案: 23 4 8

3. 请说出 #include<AsdeGameEngine.h> 和 #include "AsdeGameEngine.h" 的区别

答案:

4. 使用 strcpy 时必须注意什么才能避免缓冲区溢出的问题

5. 下面的服务端代码如果作为梭哈的洗牌发牌算法存在什么问题? 能否被玩家猜出其他玩家的底牌?

如果不能, 请说明理由,

如果能, 请说明理由, 写出如果猜其他玩家的底牌的方法, 并且给出改进方案。

```
 srand(time(NULL));
int pai[54];
for(int i=0;i<54;i++)
{
    pai[i]=i+1;
}
for (int i=0;i<54;i++)
{
    int m,n,temp;
    m = rand()%54;
    n = rand()%54;
```

## 边锋研发人员试题

基本资料:

姓名: 性别:

学历: 专业:

联系方式(手机号码):

★请附上详细的解题过程

### 简要说明

本试题重点将考察研发工程师的经验、能力、知识面及逻辑推理能力，将做为入职的重要参考，请谨慎答题。(即使不能完整做答，也请写上思路作为参考。)

1. 下面这段程序有问题吗？如果有问题，请指出问题在哪里？

如果没问题请写出下面这段程序的输出结果：

```
int c = 100;           100 100
c += (c++);           C = C + (C++)
201 cout << c << endl;
c=100;                101 101
c+=(++c);            C = C + (C++)
202 cout << c << endl;
c=100;
(++c) +=(c++);       101 + 101 = 202
203 cout << c << endl; 203
c=100;
(++c) +=(++c);       102 + 102
204 cout << c << endl;
```



答案：

};  
请编写 String 的上述 4 个函数。

七、附加题

- 1、请列举 5 个最常见 Windows GDI 函数; (5 分)
- 2、请列举 5 个最常见的 STL 容器? (5 分)

- (3) 三拖二牌型根据 3 同张的点数确定大小;
- (4) 四拖一牌型根据 4 同张的点数确定大小;
- (5) 炸弹中 3 个 A 可以带或者不带一张都为最大的炸弹。

**8、胜负判定以及得分规则**

对手牌:剩 16 张, 得 50\*基础分。(全关)

对手牌:剩 15 张, 得 40\*基础分。

对手牌:剩 14 张, 得 30\*基础分。

对手牌: 剩 13 张, 得 26\*基础分

对手牌剩 12 张及 12 张以下, 得剩余牌张数\*基础分。

**9、逃跑问题:**

逃跑后按约定的等级的最大可能输的值扣, 50\*基础分, 扣掉茶水费, 把扣的银子全部给对方。

**关牌规则:**

**【游戏简介】**

关牌游戏，一副牌去掉大小王，红桃 2 方块 2 草花 2 和方块 A，共计 48 张牌，2 个人玩，每人发 16 张，余下的牌不用。

**【游戏规则】**

1、游戏者：2 人

2、游戏目的：每位游戏者都要想方设法将自己手中的牌尽快打出去，谁先出完谁胜利。

3、关张牌数：当一方出完牌时，对家所剩的牌数。

4、使用牌数：游戏使用 48 张牌，扣除两张王牌，3 张 2 和 1 张 A。牌分三份，一份留底不用，每位游戏者 16 张牌。

5、出牌：

第一盘游戏都由首先抓到黑桃 3 的游戏者先出牌（如双方没有黑桃 3 则黑桃 4 先出，依此类推，若都没有黑桃，则系统重新自动发牌），他可以出一张牌，也可以出几张牌，只要是正确的牌型就可以了；

6、牌型：

(1) 单张（任意一张牌）；

(2) 对子（两张牌点相同的牌，两张牌的花色可以不同）；如：33、55

(3) 三同张（三张牌点相同的牌，三张牌的花色可以不同）；如：333、555

(4) 姐妹对（点数相连的 2 副对起）；如：3344、334455、44556677

(5) 连三同张（点数相连的 2 副三同张牌起）；如：333444、555666777

(6) 三拖二（一副三同张加一副对起）；如：33344、55533、3334448899

(7) 四拖一（一副四同张加一单张）；如：33334、55553

(8) 顺子（五张或五张以上牌点连续的牌，花色不限）；3、4、5、6、7、8、9、10、J、Q、K、A

(9) 炸弹（四张牌点相同的牌）如 3333、4444

三个 A 炸弹用，也可拖一单张依然是最大的炸弹

7、牌的大小顺序：

本游戏的牌点由大到小排列为：2、A、K、Q、J、10、9、8、7、6、5、4、3；

注意：

(1) 2 不在顺子中；

(2) 单张、对、三同张、姐妹对、连三同张、顺子、炸弹等牌型，直接根据牌点确定大小，但要求出牌的牌型必须相同；

### 一、关牌 AI 设计

根据关牌游戏规则，设计一份自动出牌的 AI 算法，要求思路清晰、条理明确、结构严谨，以流程图和伪代码形式表述清楚即可，不须实现详细细节。

注：生成 AI 出牌数据时，可以用到的牌数据只能是玩家自己能知道的牌，包括双方已出的牌、自家手里的牌，不能使用对家手里的牌。

```
// 关牌AI
class CGPAI
{
public:
    CGPAI();
    virtual ~CGPAI();

    // 事件接口
public:
    // 游戏开始
    void OnGameStart();

    // 对某玩家发牌( 玩家座位, 牌数据 )
    void OnTakeFirst( SHORT nSeat, const TCARDS& tCards );

    // 某玩家出牌( 玩家座位, 牌数据 )
    void OnPlay( SHORT nSeat, const TCARDS& tCards );

    // 游戏结束
    void OnGameEnd();

    // 获取AI数据接口
public:
    // 获取某玩家AI出牌数据( 玩家数据, 回传牌数据 )
    BOOL GetAIPlay( SHORT nSeat, TCARDS& tCards );

    // 数据
protected:
};

};
```

正泰中自

软件面试题目 研发中心

1. 请写出 float x 与“零值”比较的 if 语句

if ( $x > 0.00001 \text{ } \&\& \text{ } x < -0.00001$ )

2. #include <filename.h> 和 #include "filename.h" 有什么区别?

3. 什么是 STL, 举出你都用过 STL 中的哪些模板类?

STL 标准模板库

4. 堆和栈有什么区别? 在 C++ 函数中定义的局部变量内存空间属于堆还是栈?

栈为全局分配，函数运行时自动分配，成员手动分配

5. 请不用第三个变量实现两个变量的值交换(如变量 a = 1, 变量 b = 2, 交换后是 a=2, b=1)?

a = a+b;  
a<sup>1</sup> = b;  
b = a-b;  
b<sup>1</sup> = a;  
a = a-b;  
a<sup>1</sup> = b;

6. 四个数字 5, 5, 5, 1, 每个数字只用一次, 加减乘除后得 24, 请写出此算式。

5×5-1<sup>5</sup>    5×(5-1/5)

7. WINDOWS 操作系统都是基于消息的, 请写出你对消息机制的理解?

8. WIN32 应用程序创建的几个步骤是依次是(从创建到显示)?

9. 什么是 MFC, 写出 MFC 中你最熟悉的类(如 CWnd) ?

## ▲ 杭州华银视讯科技有限公司

as("Got a valid pointer");

### TypeDef

15. TypeDef 在 C 语言中频繁用以声明一个已经存在的数据类型的同义字。也可

以用预处理器做类似的事。例如，思考一下下面的例子：

```
#define dPS struct s *
```

```
typedef struct s * tPS;
```

### 晦涩的语法

16. C 语言同意一些令人震惊的结构,下面的结构是合法的吗,如果是它做些什么?

```
int a = 5, b = 7, c;
```

c = a++ + b;

4

地址：杭州市莫干山路 989 号万马集团内  
传真：0571-88932680  
E-MAIL：[service@wanin.net](mailto:service@wanin.net)

邮编：310011  
电话：0571-88932681/82/83/84/85  
网址：<http://www.wanin.net>



杭州华银视讯科技有限公司

```
printf(" Area = %f", area);  
  
return area;  
  
}
```

### 代码例子 (Code examples)

12. 下面的代码输出是什么, 为什么?

```
void foo(void)  
{  
    unsigned int a = 6;  
  
    int b = -20;  
  
    (a+b > 6) puts("> 6") : puts("<= 6");  
}
```

13. 评价下面的代码片断:

```
unsigned int zero = 0; ~ 230  
  
unsigned int compzero = 0xFFFF;  
  
/*1's complement of zero */
```

### 动态内存分配 (Dynamic memory allocation)

14. 尽管不像非嵌入式计算机那么常见, 嵌入式系统还是有从堆 (heap) 中动态分配内存的过程的。那么嵌入式系统中, 动态分配内存可能发生的问题是什么?

下面的代码片段的输出是什么, 为什么?

```
char *ptr;  
if ((ptr = (char *)malloc(0)) == NULL)  
    puts("Got a null pointer");  
else
```

地址: 杭州市莫干山路 989 号万马集团内

传真: 0571-88932680

E-MAIL: [service@wanin.net](mailto:service@wanin.net)

邮编: 310011

电话: 0571-88932681/82/83/84/85

网址: <http://www.wanin.net>



杭州华银视讯科技有限公司

```
const int a;  
int const a;  
const int *a;  
int * const a;  
int const * a const;
```

### Volatile

8. 关键字 volatile 有什么含意 并给出三个不同的例子。

### 位操作 (Bit manipulation)

9. 嵌入式系统总是要用户对变量或寄存器进行位操作。给定一个整型变量 a, 写两段代码, 第一个设置 a 的 bit 3, 第二个清除 a 的 bit 3。在以上两个操作中, 要保持其它位不变。

### 访问固定的内存位置 (Accessing fixed memory locations)

10. 嵌入式系统经常具有要求程序员去访问某特定的内存位置的特点。在某工程中, 要求设置一绝对地址为 0x67a9 的整型变量的值为 0xaa66。编译器是一个纯粹的 ANSI 编译器。写代码去完成这一任务。

### 中断 (Interrupts)

11. 中断是嵌入式系统中重要的组成部分, 这导致了很多编译开发商提供一种扩展—让标准 C 支持中断。具代表事实是, 产生了一个新的关键字 \_interrupt。下面的代码就使用了 \_interrupt 关键字去定义了一个中断服务子程序(ISR), 请评论一下这段代码的。

```
_interrupt double compute_area (double radius)  
{  
    double area = PI * radius * radius;
```

地址: 杭州市莫干山路 989 号万马集团内

传真: 0571-88932680

E-MAIL: [service@wanin.net](mailto:service@wanin.net)

邮编: 310011

电话: 0571-88932681/82/83/84/85

网址: <http://www.wanin.net>



杭州华银视讯科技有限公司

## 嵌入式 C 语言工程师招聘笔试题目(1)

### 预处理器 (Preprocessor)

1. 用预处理指令#define 声明一个常数，用以表明 1 年中有多少秒（忽略闰年问题）
2. 写一个“标准”宏 MIN，这个宏输入两个参数并返回较小的一个。
3. 预处理器标识#error 的目的是什么？

### 死循环 (Infinite loops)

4. 嵌入式系统中经常要用到无限循环，你怎么样用 C 编写死循环呢？

### 数据声明 (Data declarations)

5. 用变量 a 给出下面的定义

- a) 一个整型数 (An integer)
- b) 一个指向整型数的指针 (A pointer to an integer)
- c) 一个指向指针的指针，它指向的指针是指向一个整型数 (A pointer to a pointer to an integer)
- d) 一个有 10 个整型数的数组 (An array of 10 integers)
- e) 一个有 10 个指针的数组，该指针是指向一个整型数的 (An array of 10 pointers to integers)
- f) 一个指向有 10 个整型数数组的指针 (A pointer to an array of 10 integers)
- g) 一个指向函数的指针，该函数有一个整型参数并返回一个整型数 (A pointer to a function that takes an integer as an argument and returns an integer)
- h) 一个有 10 个指针的数组，该指针指向一个函数，该函数有一个整型参数并返回一个整型数 (An array of ten pointers to functions that take an integer argument and return an integer)

### Static

6. 关键字 static 的作用是什么？

限制变量的作用域  
设置变量的存储域

### Const

7. 关键字 const 是什么含意？

下面的声明都是什么意思？

地址：杭州市莫干山路 989 号万马集团内  
传真：0571-88932680  
E-MAIL：[service@wanin.net](mailto:service@wanin.net)

邮编：310011  
电话：0571-88932681/82/83/84/85  
网址：<http://www.wanin.net>



医疗软件

用专注的心，做专业的软件

7. 请简要回答如下问题：

- (1) 你选择软件开发职位的原因？
- (2) 你五年内的职业规划是什么？请进行描述，并解释。
- (3) 谈谈你对医疗软件的理解。



用专注的心，做专业的软件

6. 选答题（二选一）

- (1) 应聘 VC++/MFC 开发职位请答此题

使用 MFC 开发软件时，主要有哪几个最基本的类？分别起到什么作用？

- (2) 应聘.NET 开发职位请答此题

简单谈谈你对.NET 中 Delegate 的理解。

VB .NET 或 C# 支持多继承么？如果支持，如何实现？



用专注的心，做专业的软件

5. 设计一个对话框，用来对需要保护的单、多个文件进行加密处理（可用中文）：
  - (1) 请设计出该对话框功能；
  - (2) 写出各功能的优先级，解释为什么这样设定；
  - (3) 对于上述各功能，你准备如何测试？写出测试要点；
  - (4) 设计用于实现上述对话框各种功能的类，说明每个类的关键属性和方法，描述类之间的关系。

用专注的心，做专业的软件

4. 选择“BOOK”作为对象：

- (1) 请为它设计一个合适的类数据集；
- (2) 为该类提供适当的构造函数集、析构函数，并且重载其赋值运算符“=”。

用专注的心，做专业的软件

3. 请用 C/C++、VB.NET 或 C#编写程序，找出两个整型数组中所有相同的元素，并按照升序输出。

用专注的心，做专业的软件

2. 32位系统下，C程序，请计算如下代码片段中 sizeof()的值。

```
char str[] = "http://www.ibegroup.com/";
char *p = str;
int n = 10;
struct
{
    char str[12];
    int n;
    byte bt;
    double f;
} strData;
```

sizeof(str) = \_\_\_\_\_  
sizeof(p) = \_\_\_\_\_  
sizeof(n) = \_\_\_\_\_  
sizeof(strData) = \_\_\_\_\_

```
void Foo ( char str[100])
{
    sizeof(str) = _____
}
```

```
void *pMemory = malloc( 100 );
sizeof(pMemory) = _____
```



用专注的心，做专业的软件

## 杭州维科软件工程有限责任公司

### 软件工程师笔试试题

(时间: 90 分钟)

姓名 \_\_\_\_\_

联系电话 \_\_\_\_\_

原工作单位(毕业学校) \_\_\_\_\_

应聘岗位 \_\_\_\_\_ (VC++、VB、C#)

- 以下术语，请任选其中不少于两个，描述其英文全称及其概念，并举例说明其在实际中的应用： MFC, COM, XML, UML, WPF, ASP, SOAP

## 选做题

### 一. 数据库部分

1. 描述用 sqlplus, 执行一条查询语句, sqlplus 与 oracle 需要做的处理过程  
比如 :
2. 说明在数据库编程中, 比如 java 中常用 setString, setxxx 等方法。与把变量拼接成 sql 查询语句有什么不同?
3. 要操作 3 张表 (简单操作 select, update, insert), 用存储过程高, 还是用程序直接调用高? 为什么?

### 二. 网络部分

1. 请写出一个简单的 socket server 从 listen 到获得 client 的数据所用的函数
2. 如果要发送 1M 的文件, 用 udp 与 tcp, http 分别实现, 需要注意什么?

### 三. 非技术部分

四个小孩捉迷藏, 其中一个不小心打碎了一只杯子。凯特说:“是汤姆打碎的”。汤姆说:“是苯尼打碎的”。苯尼说:“不是我打碎的”珍妮说:“汤姆说是我打碎的, 他说谎”。他们四个人中其实有三个人说了谎, 只有一个人讲了真话, 那么究竟是谁打碎了杯子呢?

```
{int i, a[] = { 11, 9, 2, 5, 3, 7 } ;
s(a,6) ;
for ( i = 0 ; i < 6 ; i++ ) printf( "%4d", a[i] ) ;
printf( "\n" ) ;
}
```

### 5. 程序填空

设一个环上有编号为 0~n-1 的 n 粒不同颜色的珠子（每粒珠子颜色用字母表示，n 粒珠子的颜色由输入的字符串表示）。将环中某两粒珠子间剪开，环上珠子形成一个序列，然后按以下规则从序列中取走珠子：首先从序列左端取走所有连续同色珠子；然后从序列右端在剩下珠子中取走所有连续同色珠子，两者之和为该剪开处可取走珠子的粒数。在不同位置剪开，能取走的珠子数不尽相同。

本程序所求的是在环上哪个位置剪开，按上述规则可取走的珠子粒数最多。程序中用数组存储字符串。例如，10 粒珠子颜色对应字符串为“aaabbbadcc”，从 0 号珠子前剪开，序列为 aaabbbadcc，从左端取走 3 粒 a 色珠子，从右端取走 2 粒 c 色珠子，共取走 5 粒珠子。若在 3 号珠子前剪开，即 bbbadccaaa 共可取走 6 粒珠子。

```
#include
int count(char*s, int start, int end)
{ int i, c = 0, color = s[start], step = ( start > end ) ? -1 : 1;
  for ( i = start; s[i] == color ; i += step ) {
    if ( (step > 0 && i > end) || __1__ ) break;
    __2__
  }
  return c ;
}
void main()
{
  char t, s[120]; int i, j, c, len, maxc, cut=0 ;
  printf( "请输入环上代表不同颜色珠子字符串: " );
  scanf( "%s", s );
  len = strlen(s) ;
  for ( i = maxc = 0 ; i < len ; i++ ) { /*尝试不同的剪开方式*/
    c = count(s, 0, len-1) ;
    if ( c < len ) c += count( __3__ ); SL
    if ( c > maxc ) { cut = i ; maxc = c; }
    /*数组 s 的元素循环向左移动一个位置*/
    t = s[0] ;
    for ( j = 1; j < len ; j++ ) __4__ ;
    __5__ ;
  }
  printf( "在第 %d 号珠子前面剪开，可以取走 %d 个珠子.\n", cut, maxc ) ;
}
```

## 必做题

1. char a[] = "hello world";  
char \*p = a;  
cout << sizeof(a) << endl;  
cout << sizeof(p) << endl;  
输出结果是?

180  
770

2. 请写出 str 的值, 如果有错误请指明原因

```
void GetMemory(char *p, int num)
{
    p = (char *)malloc(sizeof(char) * num);
}
void GetMemory(char **p, int num)
{
    *p = (char *)malloc(sizeof(char) * num);
}
void Test(void)
{
    char *str = NULL;
    GetMemory(str, 100);
    strcpy(str, "hello");
}
```

3. 请列举出你在 C++ 编程过程中用到的 stl 对象类型, 说明其应用场合。

比如: vector, hash\_map, list

4. 以下程序的输出结果是 9,11,2,5,3,7, 函数 s( int b[], int n) 的功能是\_\_\_\_\_

```
#include
void s( int b[ ] , int n )
{ int i , j , t , flg ;
  for ( i = 0 ; i < n-1 ; i++ ) {
    for ( flg = 0, j = 0 ; j < n-i-1 ; i++ )
      if ( b[j] > b[j+1] ) {
        t = b[j] ; b[j] = b[j+1] ; b[j+1] = t ;
        flg = 1
      }
    if ( !flg ) break
  }
}
main( )
```



创新·专业·安全

核新同花顺软件开发工程师测试考题--A 卷

17. 共有 1000 瓶汽水，每喝完后一瓶得到的一个空瓶子，每 3 个空瓶子又能换 1 瓶汽水，喝掉以后又得到一个空瓶子，问总共能喝多少瓶汽水，最后还剩余多少个空瓶子？

18. 哪个国家的人养鱼（此题为选做题）

在一条街上，有 5 座房子，喷了 5 种颜色。每个房子里住着不同国家的人。每个人喝不同的饮料，抽不同牌子的香烟，养不同的宠物。

请你根据下面的提示说出：哪个国家的人养鱼？

- 1)、英国人住红色房子。
- 2)、瑞典人养狗。
- 3)、丹麦人喝茶。
- 4)、绿色房子在白色房子左面。
- 5)、绿色房子主人喝咖啡。
- 6)、抽 Pall Mall 香烟的人养鸟。
- 7)、黄色房子主人抽 Dunhill 香烟。
- 8)、住在中间房子的人喝牛奶。
- 9)、挪威人住第一间房。
- 10)、抽 Blends 香烟的人住在养猫的人隔壁。
- 11)、养马的人住抽 Dunhill 香烟的人隔壁。
- 12)、抽 Blue Master 的人喝啤酒。
- 13)、德国人抽 Prince 香烟。
- 14)、挪威人住蓝色房子隔壁。
- 15)、抽 Blends 香烟的人有一个喝水的邻居。



核新同花顺软件开发工程师测试考题—A 卷

## 同花顺软件开发工程师测试考题

### 说明：

- 1、本考题答题时间为 100 分钟；
- 2、本考题重点考察基础的编程知识和解决问题的能力。

### 测试考题：

1. 在 C++ 语言中 什么函数不能声明为虚函数？为什么？
2. 冒泡排序算法和快速排序算法的时间复杂度分别是什么？
3. C 语言中 局部变量能否和全局变量重名？如果能，如何解决冲突？若不能，为什么？
4. Heap 和 Stack 的有什么区别？
5. 请简述数组和链表数据结构的特点以及适用场合。
6. 写出 float x 与 “零值” 比较的 if 语句。(任何编程语言都可以)
7. 写一个函数，输出指定数字的斐波那契数。(可以不考虑溢出问题，但请注意函数性能)  
(斐波那契数列的排列是：1, 1, 2, 3, 5, 8, 13 即后一个数等于前面两个数的和)
8. 请编写字符串拷贝函数（不能调用 C/C++字符串函数库）  
函数原型：char\* strcpy(char\* strDest, const char\* strSrc)
9. 请用你熟悉的开发语言实现一个单例模式 Singleton。(此题为选做题)
10. 请描述下你对 MVC 模式的理解，以及 MVC 模式的优点和不足？
11. 请估算一下在大学期间您总共写过多少行代码？如果不包括课后作业和老师布置的编程任务以为写过多少行代码？你认为最得意的代码是什么？请简单描述一下。
12. 请估算每天使用同花顺的用户有多少，需要多少台服务器来支撑这些用户。你是如何估算的？
13. 如果有一天，领导给你分配了一个你工作职责之外的任务，这个时候你会怎么做？(此题为选做题)
14. 大部分的搜索引擎主页都千篇一律，如果需要你设计出一个与众不同但是同样能够吸引客户的搜索引擎主页，你会怎么做？
15. 简单写一下你最近关注的和 IT 行业相关的新闻、技术、产品、人物等。(此题为选做题)
16. 假如有一天你一个人在家里。想象一下下面的事情同时发生：两个月大的孩子在哭闹，煤气炉上开水开了，衣服没收天开始下大雨，门外有人敲门，此刻电话响了。你该怎么做？请写出你处理这些事情的先后顺序。  
(反面还有试题)

支持环保，请不要在试卷上涂画，以便循环利用 ☺

## C++模拟试卷---- (标准 C++部分)

总分 100 分 合格分 60 分

本次考试为 C++概述和数据类型、表达式、基本运算

选择题。(每小题 2 分, 总分 20 分)

1. C++是在 (C) 语言的基础上设计出来的。  
A. B 语言      B. D 语言  
C. C 语言      D. FORTRAN
2. 一个 C++程序的执行是从 ( )。  
A. 本程序的 main () 函数开始, 到 main () 函数结束。  
B. 本程序文件的第一个函数开始, 到本程序文件的最后一个函数结束。  
C. 本程序饿 main () 函数开始, 到本程序的最后一个函数结束。  
D. 本程序文件的第一个函数开始, 到本程序 main () 函数结束。
3. 下列叙述中, 正确的是 (A)。  
A. C++程序中注释部分可以出现在程序中的任意合适地方。  
B. 花括号 ‘{’ 和 ‘}’ 只能作为函数体的定界符。  
C. 构成 C++程序的基本单位是函数, 所有函数名都可以由用户命名。  
D. 分号是 C++语句之间的分隔符, 不是语句的一部分。
4. C++规定, 在一个源程序中, main () 函数的位置 ( )。  
A. 必须在最开始      B. 必须在系统调用函数的后面  
C. 可以任意      D. 必须在最后
5. 以下属于 C++基本特征的是 ( )。  
A. C++的输入和输出不是靠输入/输出语句来完成的。  
B. C++的每个语句结束后必须换行。  
C. C++的一个语句只能写在一行上面。  
D. C++的注释可以用 “/#.....#/” 符号来表示。
6. 下列表达式中, 不能正确表达代数式  $2xy/7z$  的是 ( )。  
A.  $2*x*y/7/z$       B.  $2*x*y/(7*z)$   
C.  $2*x/7*y/z$       D.  $2*x*y/7*z$

(C-STYL)  
C/C++开发笔试题

1. 写一个常用的 Singleton 模式的类框架，比如 Logger，可以方便地调用 log\_printf() 这样的函数打印调试信息。不需要写出成员函数实现代码。

2. .h 头文件中的 ifndef/define/endif 的作用？

3. 有如下结构体定义：

```
typedef struct _mystruct_
{
    char id;
    int birthday;
    unsigned char name[60];
    char hash[16];
} mystruct_t;
```

这个定义存在什么问题，需要如何优化？

重载是一种动态机制，即代码通过参数的类型或个数不同实现的修改机制，是一种静态的修改机制

4. 重载 (overload) 和重写 (override) 的区别？

重写是一种动态绑定的修改机制，即在父类和子类中同名方法（如成员函数）有不同的实现代码，执行的是哪个代码根据运行实际情况而定

5. 简要描述一下 MFC 的事件驱动模型、状态机模型、和 MVC 模型的概念，以游戏开发为例说说上述开发模型是如何使用的。

根据运行实际情况而定

6. 堆和栈有什么区别？如何决定变量或对象的存放位置？

在传统的 C 中堆和栈共享是一块物理内存，堆主要用于动态分配内存，从堆栈内存的低端向上分配，而栈主要是用来传递函数参数，返回值和局部变量内存分配，是从堆栈内存的高端向下分配。

7. 实现快速排序算法。快排压栈和出栈：堆是动态分配，比如用 new, malloc 分配，需要手工释放，不然会导致 memory leak。  
快排是静态分配，比如数组调用是局部的，堆栈，但堆栈能自动释放

8. 不使用库函数实现 strcat 功能或 strcpy 功能（任选一题），注意函数声明。

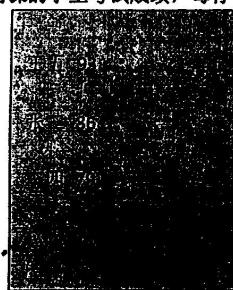
9. 实现一个 FIFO 的 Stack 类，实现 void push(char ch) 和 char pop() 两个接口，并且具备自动扩容功能。

10. 简单说明什么是虚函数和纯虚函数，分别用于什么场合？

1.已知文件 In.txt，该文件记录某班级某门课的学生考试成绩，每行记录一条信息，格式为：姓名+空格+成绩+回车换行，比如



in.txt (示例)



## out.txt (示例)

请编程实现对考试成绩按照从高到低的顺序排序，把结果输出到文件 out.txt。函数声明如下，fin 为输入文件，fout 为输出文件，成功返回 0，失败返回-1。

2.一般的应用网络协议数据由帧头和帧数据体组成，帧头会描述帧数据的长度。

请使用已经实现的数据接收函数 `recvData` 和帧数据处理函数 `processFrame`，完成连续接收数据并封装成帧数据进行处理的函数 `recvAndProcessFrame`。

```
Cin>x;
For (i=2;i<=x-1;i++)
If (x%i==0) _____;
    If (i==x) cout<<x<<" is a prime number." ;
    Else cout<<x<<" is not a prime number." ;
Return 0;
}
```

- A、goto      B、break      C、continue      D、switch

20. 有以下程序段：

```
Int n=0, p;
Do
{
    Cin>>p;
    N++;
}while (p!=12345&&n<3);
此处 do...while 循环的结束条件是 ()。
(A) p 的值不等于 12345 并且 n 的值小于 3
(B) p 的值等于 12345 并且 n 的值大于等于 3
(C) p 的值不等于 12345 或者 n 的值小于 3
(D) p 的值等于 12345 或者 n 的值大于等于 3
```

# xionyu

杭州信宇数码科技有限公司 (c++笔试题)

一、填空题 (4\*7=28)

1. 求下面函数的返回值 () ~~0+2~~ ~~X~~ ~~1110110~~ ~~1110110~~

```
1. int func(x)
2. {
3.     int countx = 0;
4.     while(x) {
5.         if (x & 1) countx++;
6.         x = x << 1; x = 0
7.     }
8.     return countx;
9. }
10. }
```

2. 运行结果: () 3 | 4. ✓

```
// test2
union V {
    struct X {
        unsigned char s1:2;
        unsigned char s2:3;
        unsigned char s3:3;
    } x;
    unsigned char c;
} v;

v.c = 100;
printf("%d", v.x.s3); 4
}
```

3. i 最后等于多少? ()

```
int i = 1;
int j = i++; j=1, i=2 i=i+j
if((i>j++) && (i++ = j)) i+=j; i=5
4. unsigned short array[] = {1, 2, 3, 4, 5, 6, 7};
int i = 3;
*(array + i) = ? 4
```

```
struct Node
{
    int data;
    Node *next;
};

typedef struct Node Node;
```

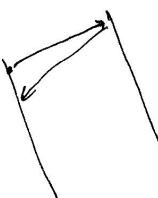
已知链表结果如上，(1)已知链表的头结点 head,写一个函数把这个链表逆序 ( Intel)

Node \* ReverseList(Node \*head) //链表逆序

(2)已知两个链表 head1 和 head2 各自有序, 请把它们合并成一个链表依然有序。(保留所有  
结点, 即便大小相同) Node \* Merge(Node \*head1, Node \*head2)

(1). Node \* ReverseList(Node \*head) {  
 Node \*p = new Node;  
 if (p != NULL){  
 p = ~~p~~ head->next;

(2). Node \* Merge(Node \*head1, Node \*head2) {



24. 阅读下面程序，指出下面程序实现的功能

```
vector<string> svec;
svec.reserve(1024);
string text_word;
while(cin>>text_word)
    svec.push_back(text_word);
svec.resize(svec.size() + svec.size() / 2);
```

定义了一个名为svec，值为string类型的vector容器；svec的可接受容量为1024，若用户输入一个字符串，则在vector容器中加入这个字符串，并将容器大小扩充为原来的一半。

25. 阅读下面代码，写出程序的运行结果

```
#include<iostream>
class A
{
public:
    A(){cout<<"A::A() called.\n";}
    virtual ~A(){cout<<"A::~A() called.\n";}
}
class B: public A
{
public:
    B(int i)
    {
        cout<<"B::B() called.\n";
        buf=new char[i];
    }
    virtual ~B()
    {
        delete []buf;
        cout<<"B::~B() called.\n";
    }
private:
    char* buf;
}
void fun(A*a)
{
    delete a;
}
void main()
{
    A *a=new B(15);
    fun(a);
}
```

A::A() called  
A::A() called  
B::B() called  
B::~B() called  
A::~A() called.

```
struct Node
{
    int data;
    Node* next;
}
```

```
}
```

void \*p = malloc(100);  
请计算  
sizeof(p)= 4

18. 通过基类的引用或指针调用虚函数时，默认实参为在 子类 声明中指定的值。

19. 构造函数与析构函数除功能不同外，在定义形式上，它们的区别还包括：

- (1) 构造函数名与类名相同，而析构函数名是在类名前加一个~
- (2) 析构函数没有参数
- (3) 析构函数没有初始化列表

20. 举出 C++ 中两种 代码复用 的方式：\_\_\_\_\_、\_\_\_\_\_

### 三、程序阅读题(21-25)

21. 阅读下面代码，写出运行结果

```
#include<stdio.h>
void main()
{
    char a='a', b='j';
    float x;
    x=(b-a)/('F'-'A');
    printf("%d\n", (int)(3.14*x));
}
```

3

22. 有如下代码，指出该代码的作用并解释该处是否是 assert 宏的一种恰当应用，并作简要分析。

```
string s;
while(cin>>s)
{
    assert(cin);
    //process s
}
```

23. 下面函数运行时，将会存在什么潜在的问题？

String &processText()

```
string text;
while(cin>>text){/*...*/};
//...
return text;
}
```

当输入一个空字符串，就什么也不做，函数返回这个字符串。

## 研发工程师编程基础测试 ( C++ )

1、数值计算。找出一个整数数组中子数组之和的最大值，例如：数组 [1, -2, 3, 5, -1]，返回 8；数组 [1, -23, -8, 5, 1]，返回 6。

2、字符串操作。把一个英语句子中的单词顺序颠倒后输出。例如：“how are you”，返回 “you are how”。

3、指针操作。用二叉树进行排序。

4、使用工具 Visual Studio，创建一个 C++Windows 程序，在主窗口中显示 “Hello Word”，用户可以用鼠标拖动 “Hello Word” 在主窗口中移动。

重点解惑：

### 1. 指针和引用的区别

指针指向一块内存，它的内容是指向内存的地址；引用是某内存的别名

引用使用是无需解引用，指针需解引用

引用不能为空，指针可以为空

引用在定义时被初始化一次，之后不可变；指针可变

程序为指针变量分配内存区域，而引用不需要分配内存区域

### 2. memcpy 和 strcpy 的区别

memcpy 用来内存拷贝的，它有指定的拷贝数据长度，它可以拷贝任何数据类型的对象

strcpy 它只能去拷贝字符串，它遇到 '\0' 结束拷贝

### 3. new 和 malloc 的区别，free 和 delete 的区别

malloc 与 free 是 C++/C 语言的标准库函数，new/delete 是 C++ 的运算符。它们都可用于申请动态内存和释放内存。

对于非内部数据类型的对象而言，光用 malloc/free 无法满足动态对象的要求。对象在创建的同时要自动执行构造函数，对象在消亡之前要自动执行析构函数。由于 malloc/free 是库函数而不是运算符，不在编译器控制权限之内，不能够把执行构造函数和析构函数的任务强加于 malloc/free。

因此 C++ 语言需要一个能完成动态内存分配和初始化工作的运算符 `new`，以及一个能完成清理与释放内存工作的运算符 `delete`。注意 `new/delete` 不是库函数。

#### 4. `struct` 和 `class` 的区别

##### 1. 成员变量

结构在默认情况下的成员是公共 (`public`) 的，

而类在默认情况下的成员是私有 (`private`) 的。

##### 2. 存储

`struct` 保证成员按照声明顺序在内存中存储。`class` 不保证等等

##### 3. 继承

```
struct A { };
```

```
class B : A{ }; //private 继承
```

```
struct C : B{ }; //public 继承
```

这是由于 `class` 默认是 `private`，`struct` 默认是 `public`。

#### 5. `struct` 与 `union` 的区别。(一般假定在 32 位机器上)

1. 一个 `union` 类型的变量，所有成员变量共享一块内存，该内存的大小由这些成员变量中长度最大的一个来决定，`struct` 中成员变量内存都是独立的

2. `union` 分配的内存是连续的，而 `struct` 不能保证分配的内存是连续的

#### 6. 队列和栈有什么区别？

队列先进先出，栈后进先出

#### 7. 指针在 16 位机、32 位机、64 位机分别占用多少个字节

16 位机 2 字节

32 位机 4 字节

64 位机 8 字节

#### 8. 如何引用一个已经定义过的全局变量？

`extern`

可以用引用头文件的方式，也可以用 `extern` 关键字，如果用引用头文件方式来引用某个在头文件中声明的全局变量，假定你将那个变量写错了，那么在编译期间会报错，如果你用 `extern` 方式引用时，假定你犯了同样的错误，那么在编译期间不会报错，而在连接期间报错

9. 全局变量可不可以定义在可被多个.c 文件包含的头文件中？为什么？

可以，在不同的 C 文件中以 static 形式来声明同名全局变量。

可以在不同的 C 文件中声明同名的全局变量，前提是其中只能有一个 C 文件中对此变量赋初值，此时连接不会出错

10. 语句 for( ; 1 ; )有什么问题？它是什么意思？

for( ; 1 ; )和 while(1)相同。

11. do.....while 和 while.....do 有什么区别？

前一个循环一遍再判断，后一个判断以后再循环

12. 请写出下列代码的输出内容

```
#include<stdio.h>
main()
{
    int a,b,c,d;
    a=10;
    b=a++;
    c=++a;
    d=10*a++;
    printf("b , c , d : %d , %d , %d" , b , c , d );
    return 0;
}
10 , 12 , 120
```

13. 对于一个频繁使用的短小函数，在 C 语言中应用什么实现，在 C++ 中应用什么实现？

C 用宏定义，C++ 用 inline

14. main 函数执行以前，还会执行什么代码？

全局对象的构造函数会在 main 函数之前执行。

15. main 主函数执行完毕后，是否可能会再执行一段代码，给出说明？

可以，可以用 on\_exit 注册一个函数，它会在 main 之后执行 int fn1(void), fn2(void), fn3(void), fn4 (void);

```
void main( void )
{
    String str("zhanglin");
    on_exit( fn1 );
    on_exit( fn2 );
    on_exit( fn3 );
    on_exit( fn4 );
```

```
    printf( "This is executed first.\n" );
}
int fn1()
{
    printf( "next.\n" );
    return 0;
}
int fn2()
{
    printf( "executed " );
    return 0;
}
int fn3()
{
    printf( "is " );
    return 0;
}
int fn4()
{
    printf( "This " );
    return 0;
}
```

The `on_exit` function is passed the address of a function (`func`) to be called when the program terminates normally. Successive calls to `on_exit` create a register of functions that are executed in LIFO (last-in-first-out) order. The functions passed to `on_exit` cannot take parameters.

This is executed next.

#### 16. 局部变量能否和全局变量重名？

能，局部会屏蔽全局。要用全局变量，需要使用"::"

局部变量可以与全局变量同名，在函数内引用这个变量时，会用到同名的局部变量，而不会用到全局变量。对于有些编译器而言，在同一个函数内可以定义多个同名的局部变量，比如在两个循环体内都定义一个同名的局部变量，而那个局部变量的作用域就在那个循环体内

#### 17. 描述内存分配方式以及它们的区别？

1. 从静态存储区域分配。内存在程序编译的时候就已经分配好，这块内存在程序的整个运行期间都存在。例如全局变量，`static` 变量。

2. 在栈上创建。在执行函数时，函数内局部变量的存储单元都可以在栈上创建，函数执行结束时这些存储单元自动被释放。栈内存分配运算内置于处理器的指令集。

- 
3. 从堆上分配，亦称动态内存分配。程序在运行的时候用 `malloc` 或 `new` 申请任意多少的内存，程序员自己负责在何时用 `free` 或 `delete` 释放内存。动态内存的生存期由程序员决定，使用非常灵活，但问题也最多。
18. 类成员函数的重载、覆盖和隐藏区别？
1. 成员函数被重载的特征：
- (1) 相同的范围（在同一个类中）；
  - (2) 函数名字相同；
  - (3) 参数不同；
  - (4) `virtual` 关键字可有可无。
2. 覆盖是指派生类函数覆盖基类函数，特征是：
- (1) 不同的范围（分别位于派生类与基类）；
  - (2) 函数名字相同；
  - (3) 参数相同；
  - (4) 基类函数必须有 `virtual` 关键字。
3. “隐藏”是指派生类的函数屏蔽了与其同名的基类函数，规则如下：
- (1) 如果派生类的函数与基类的函数同名，但是参数不同。此时，不论有无 `virtual` 关键字，基类的函数将被隐藏（注意别与重载混淆）。
  - (2) 如果派生类的函数与基类的函数同名，并且参数也相同，但是基类函数没有 `virtual` 关键字。此时，基类的函数被隐藏（注意别与覆盖混淆）
19. `static` 有什么用途？（请至少说明两种）
- 1. 限制变量的作用域
  - 2. 设置变量的存储域
20. 请说出 `const` 与 `#define` 相比，有何优点？
- 1. `const` 常量有数据类型，而宏常量没有数据类型。编译器可以对前者进行类型安全检查。而对后者只进行字符替换，没有类型安全检查，并且在字符替换可能会产生意想不到的错误。
  - 2. 有些集成化的调试工具可以对 `const` 常量进行调试，但是不能对宏常量进行调试。
- 堆栈溢出一般是由什么原因导致的？
- 没有回收垃圾资源

21. 简述数组与指针的区别？

数组要么在静态存储区被创建（如全局数组），要么在栈上被创建。指针可以随时指向任意类型的内存块。

(1) 修改内容上的差别

```
char a[] = "hello";
a[0] = 'x';

char *p = "world"; // 注意 p 指向常量字符串
p[0] = 'x'; // 编译器不能发现该错误，运行时错误
```

(2) 用运算符 `sizeof` 可以计算出数组的容量（字节数）。`sizeof(p)`, `p` 为指针得到的是一个指针变量的字节数，而不是 `p` 所指的内存容量。C++/C 语言没有办法知道指针所指的内存容量，除非在申请内存时记住它。注意当数组作为函数的参数进行传递时，该数组自动退化为同类型的指针。

```
char a[] = "hello world";
char *p = a;

cout << sizeof(a) << endl; // 12 字节
cout << sizeof(p) << endl; // 4 字节
```

计算数组和指针的内存容量

```
void Func(char a[100])
{
    cout << sizeof(a) << endl; // 4 字节而不是 100 字节
}
```

22. There are two int variables: `a` and `b`, don't use "if", "? :", "switch" or other judgement statements, find out the biggest one of the two numbers.

```
((a + b) + abs(a - b)) / 2
```

23. 冒泡排序算法的时间复杂度是什么？

$O(n^2)$

24. 什么函数不能声明为虚函数？

构造函数 (Constructor)

25. 变量在内存中存放的位置

全局变量 全局静态区

全局静态变量 全局静态区

全局常量

有初始化 代码区

无初始化	全局静态区
局部变量	堆栈区
局部静态变量	静态区
局部常量	堆栈区
new 和 malloc 分配空间	堆区

## 26. 进程间通信方式

管道（有名管道，无名管道），共享内存，消息队列，信号量，socket 通信

## 27. 线程同步方式

临界区：通过对多线程的串行化来访问公共资源或一段代码，速度快，适合控制数据访问

互斥量：为协调共同对一个共享资源的单独访问而设计

信号量（PV 操作）：为控制一个具有有限数量用户资源而设计

事件：用来通知线程有一些事件已

## 28. 进程和线程的区别

资源：进程是拥有资源的一个独立单位，线程是不拥有资源。

调度：线程作为调度和分配的基本单位，进程是作为资源的基本单位

并发性：进程之间可以有并发性进行，同一个进程中的多个线程是可以并发执行

系统开销：进程在创建和撤销的时候，由于系统要分配和回收资源，导致系统的开销明显大于线程

一个进程可以拥有多个线程。

## 29. 局部变量和全局变量能否重名

能，局部屏蔽全局。在 C++ 里使用全局，需要使用 “::”。在 C 语言里，extern

## 30. 虚函数和纯虚函数的区别

虚函数必须实现，纯虚函数没有实现

虚函数在子类里可以不重载，但是纯虚函数必须在每一个子类里去实现

在动态内存分配的时候，析构函数必须是虚函数，但没有必要是纯虚函数

## 31. 面向对象的三大特性（四大特性）

封装、继承、多态（抽象）

封装：把客观事物封装成抽象的类，并且类可以把自己的数据和方法只让可信的类或者对象操作，对不可信的进行信息隐藏

**继承**：子类可以拥有父类的属性和方法，但父类没有子类的属性和方法

**多态**：允许将子类类型的指针赋值给父类类型的指针

实现多态，有二种方式，覆盖，重载

覆盖，是指子类重新定义父类的虚函数的做法

重载，是指允许存在多个同名函数，而这些函数的参数表不同（或许参数个数不同，或许参数类型不同，或许两者都不相同）

32. vi 编辑器打开时跳到指定的行

vi +5000 filename

33. int 型在 Touble C 里占多少个字节

2 个字节

34. 判断一个单链表是否有环

两个指针指向链表头，一个指针每次走一步，另一个指针每次走两步，若有一个指针先指向为 NULL 表示这个链表无环。若两个指针重合表示链表有环

35. 刷新缓冲区方式？

换行刷新缓冲区

printf("\n");

使用函数刷新缓冲区

fflush(stdout);

程序结束刷新缓冲区

return 0;

36. 类和对象的两个基本概念什么？

对象就是对客观事物在计算机中的抽象描述。

类就是对具体相似属性和行为的一组对象的统一描述。

类的包括：类说明和类实现两大部分：

类说明提供了对该类所有数据成员和成员函数的描述。

类实现提供了所有成员函数的实现代码。

37. 数据库三范式

第一范式：没有重复的列

第二范式：非主属的部分依赖于主属部分

### 第三范式：属性部分不依赖于其他非主属部分

38. ASSERT( )是什么用的

是在调试程序使用的一个宏，括号里面要满足，如果不满足，程序将报告错误，并将终止执行。

39. 如果只想让程序有一个实例运行，不能运行两个。像 winamp 一样，只能开一个窗口，怎样实现？

用内存映射或全局原子（互斥变量）查找窗口句柄

FindWindow，互斥，写标志到文件或注册表，共享内存

40. 如何截取键盘的响应，让所有的 ‘a’ 变成 ‘b’ ？

键盘钩子 SetWindowsHookEx

41. 网络编程中设计并发服务器，使用多进程与 多线程，请问有什么区别？

1. 进程：子进程是父进程的复制品。子进程获得父进程数据空间、堆和栈的复制品。

2. 线程：相对与进程而言，线程是一个更加接近与执行体的概念，它可以与同进程的其他线程共享数据，但拥有自己的栈空间，拥有独立的执行序列。

两者都可以提高程序的并发度，提高程序运行效率和响应时间。

线程和进程在使用上各有优缺点：线程执行开销小，但不利于资源管理和保护；而进程正相反。同时，线程适合于在 SMP 机器上运行，而进程则可以跨机器迁移。

## 第 7 章 编程

### 字符串实现

#### strcat

```
char *strcat(char *strDes, const char *strSrc)
{
    assert(strDes != NULL) && (strSrc != NULL);
    char *address = strDes;
    while (*strDes != '\0')
        ++ strDes;
    while ((*strDes ++ = *strSrc ++) != '\0')
        NULL;
    return address;
}
```

### strncat

```
char *strncat(char *strDes, const char *strSrc, int count)
{
    assert(strDes != NULL && strSrc != NULL);
    char *address = strDes;
    while (*strDes != '\0')
        ++ strDes;
    while (count -- && *strSrc != '\0' )
        *strDes ++ = *strSrc++;
    *strDes = '\0';
    return address;
}
```

### strcmp

```
int strcmp(const char *s, const char *t)
{
    assert(s != NULL && t != NULL);
    while (*s && *t && *s == *t)
    {
        ++ s;
        ++ t;
    }
    return (*s - *t);
}
```

### strncmp

```
int strncmp(const char *s, const char *t, int count)
{
    assert(s != NULL && t != NULL);
    while (*s && *t && *s == *t && count --)
    {
        ++ s;
        ++ t;
    }
    return (*s - *t);
}
```

### strcpy

```
char *strcpy(char *strDes, const char *strSrc)
{
    assert(strDes != NULL && strSrc != NULL);
```

```
char *address = strDes;
while ((*strDes ++ = *strSrc ++) != '\0')
    NULL;
return address;
}
```

### **strncpy**

```
char *strncpy(char *strDes, const char *strSrc, int count)
{
    assert(strDes != NULL && strSrc != NULL);
    char *address = strDes;
    while (count -- && *strSrc != '\0')
        *strDes ++ = *strSrc++;
    return address;
}
```

### **strlen**

```
int strlen(const char *str)
{
    assert(str != NULL);
    int len = 0;
    while (*str ++ != '\0')
        ++ len;
    return len;
}
```

### **strpbrk**

```
char *strpbrk(const char *strSrc, const char *str)
{
    assert((strSrc != NULL) && (str != NULL));
    const char *s;
    while (*strSrc != '\0')
    {
        s = str;
        while (*s != '\0')
        {
            if (*strSrc == *s)
                return (char *) strSrc;
            ++ s;
        }
    }
}
```

```
    ++ strSrc;
}
return NULL;
}

strstr
char *strstr(const char *strSrc, const char *str)
{
    assert(strSrc != NULL && str != NULL);
    const char *s = strSrc;
    const char *t = str;
    for (; *t != '\0'; ++ strSrc)
    {
        for (s = strSrc, t = str; *t != '\0' && *s == *t; ++s, ++t)
            NULL;
        if (*t == '\0')
            return (char *) strSrc;
    }
    return NULL;
}
```

### **strcspn**

```
int strcspn(const char *strSrc, const char *str)
{
    assert((strSrc != NULL) && (str != NULL));
    const char *s;
    const char *t = strSrc;
    while (*t != '\0')
    {
        s = str;
        while (*s != '\0')
        {
            if (*t == *s)
                return t - strSrc;
            ++ s;
        }
        ++ t;
    }
    return 0;
}
```

### strspn

```
int strspn(const char *strSrc, const char *str)
{
    assert(strSrc != NULL) && (str != NULL);
    const char *s;
    const char *t = strSrc;
    while (*t != '\0')
    {
        s = str;
        while (*s != '\0')
        {
            if (*t == *s)
                break;
            ++ s;
        }
        if (*s == '\0')
            return t - strSrc;
        ++ t;
    }
    return 0;
}
```

### strrchr

```
char *strrchr(const char *str, int c)
{
    assert(str != NULL);
    const char *s = str;
    while (*s != '\0')
        ++ s;
    for (-- s; *s != (char) c; -- s)
        if (s == str)
            return NULL;
    return (char *) s;
}
```

### strrev

```
char* strrev(char *str)
{
    assert(str != NULL);
    char *s = str, *t = str, c;
    while (*t != '\0')
```

```
    ++ t;
    for (-- t; s < t; ++ s, -- t)
    {
        c = *s;
        *s = *t;
        *t = c;
    }
    return str;
}
```

### strnset

```
char *strnset(char *str, int c, int count)
{
    assert(str != NULL);
    char *s = str;
    for (; *s != '\0' && s - str < count; ++ s)
        *s = (char) c;
    return str;
}
```

### strset

```
char *strset(char *str, int c)
{
    assert(str != NULL);
    char *s = str;
    for (; *s != '\0'; ++ s)
        *s = (char) c;
    return str;
}
```

### strtok

```
char *strtok(char *strToken, const char *str)
{
    assert(strToken != NULL && str != NULL);
    char *s = strToken;
    const char *t = str;
    while (*s != '\0')
    {
        t = str;
        while (*t != '\0')
        {
```

```
    if (*s == *t)
    {
        *(strToken + (s - strToken)) = '\0';
        return strToken;
    }
    ++ t;
}
++ s;
}
return NULL;
}
```

### strupr

```
char *strupr(char *str)
{
    assert(str != NULL);
    char *s = str;
    while (*s != '\0')
    {
        if (*s >= 'a' && *s <= 'z')
            *s -= 0x20;
        s++;
    }
    return str;
}
```

### strlwr

```
char *strlwr(char *str)
{
    assert(str != NULL);
    char *s = str;
    while (*s != '\0')
    {
        if (*s >= 'A' && *s <= 'Z')
            *s += 0x20;
        s++;
    }
    return str;
}
```

### **memcpy**

```
void *memcpy(void *dest, const void *src, int count)
{
    assert((dest != NULL) && (src != NULL));
    void *address = dest;
    while (count --)
    {
        *(char *) dest = *(char *) src;
        dest = (char *) dest + 1;
        src = (char *) src + 1;
    }
    return address;
}
```

### **memccpy**

```
void *memccpy(void *dest, const void *src, int c, unsigned int count)
{
    assert((dest != NULL) && (src != NULL));
    while (count --)
    {
        *(char *) dest = *(char *) src;
        if (* (char *) src == (char) c)
            return ((char *) dest + 1);
        dest = (char *) dest + 1;
        src = (char *) src + 1;
    }
    return NULL;
}
```

### **memchr**

```
void *memchr(const void *buf, int c, int count)
{
    assert(buf != NULL);
    while (count --)
    {
        if (* (char *) buf == c)
            return (void *) buf;
        buf = (char *) buf + 1;
    }
    return NULL;
}
```

### **memcmp**

```
int memcmp(const void *s, const void *t, int count)
{
    assert((s != NULL) && (t != NULL));
    while (* (char *) s && * (char *) t && * (char *) s == * (char *) t && count --)
    {
        s = (char *) s + 1;
        t = (char *) t + 1;
    }
    return (* (char *) s - * (char *) t);
}
```

### **memmove**

```
void *memmove(void *dest, const void *src, int count)
{
    assert(dest != NULL && src != NULL);
    void *address = dest;
    while (count --)
    {
        * (char *) dest = * (char *) src;
        dest = (char *) dest + 1;
        src = (const char *) src + 1;
    }
    return address;
}
```

### **memset**

```
void *memset(void *str, int c, int count)
{
    assert(str != NULL);
    void *s = str;
    while (count --)
    {
        * (char *) s = (char) c;
        s = (char *) s + 1;
    }
    return str;
}
```

### **strup**

```
char *strup(const char *strSrc)
{
    assert(strSrc != NULL);
    int len = 0;
    while (*strSrc ++ != '\0')
        ++ len;
    char *strDes = (char *) malloc (len + 1);
    while ((*strDes ++ = *strSrc ++) != '\0')
        NULL;
    return strDes;
}
```

### **strchr\_**

```
char *strchr_(char *str, int c)
{
    assert(str != NULL);
    while ((*str != (char) c) && (*str != '\0'))
        str++;
    if (*str != '\0')
        return str;
    return NULL;
}
```

### **strchr**

```
char *strchr(const char *str, int c)
{
    assert(str != NULL);
    for (; *str != (char) c; ++ str)
        if (*str == '\0')
            return NULL;
    return (char *) str;
}
```

### **atoi**

```
int atoi(const char* str)
{
    int x=0;
    const char* p=str;
    if(*str=='-' || *str=='+')
    {
```

```
    str++;
}
while (*str!=0)
{
    if ((*str>'9') || (*str<'0'))
    {
        break;
    }
    x=x*10+(*str-'0');
    str++;
}
if (*p=='-')
{
    x=-x;
}
return x;
}
```

### itoa

```
char* itoa(int val,char* buf,unsigned int radix)
{
    char *bufptr;
    char *firstdig;
    char temp;
    unsigned int digval;
    assert(buf != NULL);
    bufptr = buf;
    if (val < 0)
    {
        *bufptr++ = '-';
        val = (unsigned int) (-(int)val);
    }
    firstdig = bufptr;
    do
    {
        digval =(unsigned int) val % radix;    val /= radix;
        if (digval > 9)
        {
            *bufptr++ = (char) (digval - 10 + 'a');
        }
        else
        {
```

```

        *bufptr++ = (char)(digval + '0');

    }
} while(val > 0);

*bufptr-- = '\0'; //设置字符串末尾，并将指针指向最后一个字符

do //反转字符
{
    temp = *bufptr;    *bufptr = *firstdig;  *firstdig = temp;
    --bufptr; ++firstdig;
} while(firstdig < bufptr);
return buf;
}

```

## String 实现

已知 String 原型为：

```

class String
{
public:
    //普通构造函数
    String(const char *str = NULL)
    //拷贝构造函数
    String(const String &other)
    //析构函数
    ~String(void);
    //赋值函数

    String & operator=(String &other) //oh,原题目打错了, string 可是一个关键字
private:
    char* m_str;
    unsigned m_uCount;
};

```

分别实现以上四个函数

```

//普通构造函数
String::String(const char* str)
{
    if(str==NULL)          //如果 str 为 NULL, 存空字符串
    {

```

```
m_str = new char[1];    //分配一个字节
*m_str = '\0';          //赋一个'\0'
}else
{
    m_str = new char[strlen(str) + 1];//分配空间容纳 str 内容
    strcpy(m_str, str);           //复制 str 到私有成员 m_str 中
}
//析构函数
String::~String()
{
    if(m_str!=NULL)    //如果 m_str 不为 NULL , 释放堆内存
    {
        delete [] m_str;
        m_str = NULL;
    }
}
//拷贝构造函数
String::String(const String &other)
{
    m_str = new char[strlen(other.m_str)+1]; //分配空间容纳 str 内容
    strcpy(m_str, other.m_str);           //复制 other.m_str 到私有成员 m_str 中
}
//赋值函数
String & String::operator=(String &other)
{
    if(this == &other)                //若对象与 other 是同一个对象 , 直接返回本身
    {
        return *this
    }
    delete [] m_str;                //否则 , 先释放当前对象堆内存
    m_str = new char[strlen(other.m_str)+1]; //分配空间容纳 str 内容
    strcpy(m_str, other.m_str);           //复制 other.m_str 到私有成员 m_str 中
    return *this;
}
```

}

### 编写一个二分查找的功能函数

```
int BSearch(elemtype a[], elemtype x, int low, int high)
/*在下届为 low , 上界为 high 的数组 a 中折半查找数据元素 x*/
{
    int mid;
    if (low>high)
        return -1;
    mid=(low+high)/2;
    if (x==a[mid])
        return mid;
    if (x<a[mid])
        return (BSearch(a,x,low,mid-1));
    else
        return (BSearch(a,x,mid+1,high));
}
```

### 2) 非递归方法实现：

```
int BSearch(elemtype a[], keytype key, int n)
{
    int low,high,mid;
    low=0;high=n-1;
    while (low<=high)
    {
        mid=(low+high)/2;
        if (a[mid].key==key)
            return mid;
        else if (a[mid].key<key)
            low=mid+1;
        else
            high=mid-1;
    }
    return -1;
}
```

### 字符串逆序

#### 方法一

```
#include <stdio.h>
#include <string.h>
```

```
void main()
{
    char str[]="hello,world";
    int len=strlen(str);
    char t;
    int i;
    for(i=0; i<len/2; i++)
    {
        t=str[i];
        str[i]=str[len-i-1];
        str[len-i-1]=t;
    }
    printf("%s\n",str);
    return 0;
}
```

## 方法二

```
#include <stdio.h>
int main(){
    char* src = "hello,world";
    int len = strlen(src);

    char* dest = (char*)malloc(len+1); //要为\0分配一个空间
    char* d = dest;

    char* s = &src[len-1]; //指向最后一个字符
    while( len-- != 0 )
        *d++=*s--;
    *d = 0; //尾部要加\0
    printf("%s\n",dest);
    free(dest); // 使用完，应当释放空间，以免造成内存泄露
    return 0;
}
```

## 排序

### 冒泡排序

```
void bubble_sort(int a[],int n)
{
    int i,j;
    for(i=0;i<n-1;i++)
    {
```

```
bool x=ture;
for(j=0;j<n-1-i;j++)
{
    int temp;
    if(a[j]>a[j+1])
    {
        temp=a[j];
        a[j]=a[j+1];
        a[j+1]=temp;
        x=false;
    }
    if(x) break;
}
}
```

时间复杂度  $O(N^2)$

### 选择排序

```
void select_sort(int a[],int n)
{
    int i,j;
    for(i=0;i<n-1;i++)
    {
        int min=i;
        for(j=i+1;j<n;j++)
        {
            if(a[j]<a[min])
                min=j;
            if(min!=i)
            {
                int temp=a[j];
                a[j]=a[min];
                a[min]=temp;
            }
        }
    }
}
```

时间复杂度  $O(N^2)$

## 插入排序

```
void insert_sort(int a[],int n)
{
    int i,j;
    for(i=1;i<n;i++)
    {
        int x=a[i];
        for(j=i;j>0&&x<a[j-1];j--)
            a[j]=a[j-1];
        a[j]=x;
    }
}
```

时间复杂度  $O(N^2)$

## 快速排序

```
void quick_sort(int a[],int ileft,int iright)
{
    int iPivot=(left+right)/2;
    int nPivot=a[iPivot];
    for(int i=ileft,j=iright;i<j;)
    {
        while(!(i>=iPivot||nPivot<a[i]))
            i++;
        if(i<iPivot)
        {
            a[iPivot]=a[i];
            iPivot=i;
        }
        while(!(j<=iPivot||nPivot>a[j]))
            j--;
        if(j>iPivot)
        {
            a[iPivot]=a[j];
            iPivot=j;
        }
    }
    a[iPivot]=nPivot;
    if(iPivot-ileft>1)
        quick_sort(a,ileft,iPivot-1);
    if(iright-iPivot>1)
```

```
    quick_sort(a, iPivot+1, iright);  
}
```

时间复杂度  $O(N \log N)$

## 链表

### 单链表

### 双链表

### 循环链表

#### 单链表逆置

```
void reverse(link *head)  
{  
    link *p, *s, *t;  
    p = head;  
    s = p->next;  
    while(s->next!=NULL)  
    {  
        t = s->next;  
        s->next = p;  
        p = s;  
        s = t;  
    }  
    s->next = p;  
    head->next->next = NULL; //尾指针置为空  
    head->next = s; //赋值到头指针后一位  
}
```

#### 链表合并

```
Node * Merge(Node *head1 , Node *head2)  
{  
    if ( head1 == NULL)  
        return head2 ;  
    if ( head2 == NULL)  
        return head1 ;  
    Node *head = NULL ;  
    Node *p1 = NULL;
```

```
Node *p2 = NULL;
if ( head1->data < head2->data )
{
    head = head1 ;
    p1 = head1->next;
    p2 = head2 ;
} else
{
    head = head2 ;
    p2 = head2->next ;
    p1 = head1 ;
}
Node *pcurrent = head ;
while ( p1 != NULL && p2 != NULL)
{
    if ( p1->data <= p2->data )
    {
        pcurrent->next = p1 ;
        pcurrent = p1 ;
        p1 = p1->next ;
    } else
    {
        pcurrent->next = p2 ;
        pcurrent = p2 ;
        p2 = p2->next ;
    }
}
if ( p1 != NULL )
    pcurrent->next = p1 ;
if ( p2 != NULL )
    pcurrent->next = p2 ;
return head ;
}
```

递归方式：

```
Node * MergeRecursive(Node *head1 , Node *head2)
{
    if ( head1 == NULL )
        return head2 ;
    if ( head2 == NULL)
        return head1 ;
```

```
Node *head = NULL ;
if ( head1->data < head2->data )
{
    head = head1 ;
    head->next = MergeRecursive(head1->next,head2);
}
else
{
    head = head2 ;
    head->next = MergeRecursive(head1,head2->next);
}
return head ;
}
```

## 写一个 **Singleton** 模式

```
#include<iostream>
using namespace std;
class Singleton
{
private:
    static Singleton* _instance;
protected:
    Singleton()
    {
        cout<<"Singleton"<<endl;
    }
public:
    static Singleton* Instance()
    {
        if(NULL==_instance)
        {
            _instance=new Singleton();
        }
        return _instance;
    }
};
static Singleton* Singleton::_instance=NULL;
int main()
{
    Singleton * s =Singleton::Instance();
    Singleton * s1=Singleton::Instance();
```

}

如何对 **String** 类型数据的某个字符进行访问？

```
#include<iostream>
using namespace std;
int main()
{
    string s="abcdefg";
    const char *c=s.c_str();
    while(*c]!='\0')
    {
        printf("%c", *c++);
    }
}
```

文件加密、解密

### 1. 加密（**encryption**）：

```
#include<stdio.h>
void encryption(char *ch)
{
    (*ch) ^= 0xFF; // 算法可自行修改调整，使用 AES 加密算法
}
int main(int argc,char *argv[])
{
    if(argc<2)
    {
        printf("参数不足");
        return -1;
    }
    // 文件的打开 ( fopen 函数 )
/*
    r   read  只读
    w   write 只写
    a   append 追加
    t   text   文本文件，可省略不写
    b   banary 二进制文件

```

+ 读和写

```
*/  
a.out      a.c      b.txt  
argv[0]    argv[1]   argv[2]
```

```
FILE* fpr=NULL;  
FILE* fpw=NULL;  
//文件打开失败返回一个空指针值 NULL  
if(NULL==(fpr=fopen(argv[1],"r"))){printf("%m\n");return -1;}  
if(NULL==(fpw=fopen(argv[2],"w+"))){printf("%m\n");return -1;}  
char ch;  
while((ch=fgetc(fpr))!=EOF)  
{  
    //putchar(ch);  
    encryption(&ch); //加密函数  
    printf("%c",ch); //加密后字符显示  
    fputc(ch,fpw); //存放进文件  
}  
printf("\n 文件加密成功！\n");  
  
//文件的关闭(fclose 函数)  
fclose(fpr);  
fclose(fpw);  
}
```

**2.解密 (decryption):**

```
#include<stdio.h>  
#include<time.h>  
  
void show()  
{  
    time_t start=time(NULL);  
    while(start==time(NULL));  
}  
void decryption(char ch)
```

```
{  
    (*ch) ^=0xFF; //算法可自行修改调整  
}  
int main(int argc,char *argv[]){  
    if(argc<2){  
        printf("参数不足");  
        return -1;  
    }  
    //文件的打开 ( fopen 函数 )  
/*  
    r   read  只读  
    w   write 只写  
    a   append 追加  
    t   text   文本文件 , 可省略不写  
    b   banary 二进制文件  
    +   读和写  
*/  
FILE* fpr=NULL;  
FILE* fpw=NULL;  
//文件打开失败返回一个空指针值 NULL  
if(NULL==(fpr=fopen(argv[1],"r"))){printf("%m\n");return -1;}  
if(NULL==(fpw=fopen(argv[2],"w+"))){printf("%m\n");return -1;}  
char ch;  
printf("开始解密 ! \n");  
while((ch=fgetc(fpr))!=EOF){  
    show();  
    ch=decryption(ch); //解密函数  
    printf("%c",ch); //解密后字符显示  
    fputc(ch,fpw); //存放进文件  
    fflush(stdout); //刷新显示  
}
```

```
printf("\n 文件解密成功！\n");

//文件的关闭(fclose 函数)
fclose(fpr);
fclose(fpw);
}
```

### 斐波那契数列 (**Fibonacci sequence**)

```
int Funct( int n )
{
    if( n==0 || n==1 ) return 1;
    retrurn Funct(n-1) + Funct(n-2);
}
```

1、引用与指针有什么区别？

- 答： 1) 引用必须被初始化，指针不必。  
2) 引用初始化以后不能被改变，指针可以改变所指的对象。  
3) 不存在指向空值的引用，但是存在指向空值的指针。

2、全局变量和局部变量在内存中是否有区别？如果有，是什么区别？

答： 全局变量储存在静态数据库，局部变量在堆栈

3、\_STDC\_ 和 \_cplusplus 宏的使用

答： #if \_\_cplusplus

```
#include<iostream>
```

```
Using namespace std;
```

```
int main()
```

```
{  
    cout << "c++" << endl;  
}  
  
#elif __STDC__  
  
#include<stdio.h>  
  
int main()  
{  
    printf("C\n");  
}  
  
#endif
```

#### 4、怎样判断大端和小端？

答：大端格式：在这种格式中，字数据的高字节存储在低地址中，而字数据的低字节则存放在高地址中

小端格式：与大端存储格式相反，在小端存储格式中，低地址中存放的是字数据的低字节，高地址存放的是字数据的高字节

```
#include <stdio.h>  
  
int main() {  
    union un {  
        short s;  
        char c[2];  
    } u;
```

```
u. s=0x0102;  
  
if(u. c[0]==1&&u. c[1]==2) {  
    printf("大端\n");  
}  
else if(u. c[0]==2&&u. c[1]==1) {  
    printf("小端\n");  
}  
}
```

5、以下代码中的两个 sizeof 用法有问题吗？

```
void UpperCase( char str[] ) // 将 str 中的小写字母转换成大写字母  
{  
    for( size_t i=0; i<sizeof(str)/sizeof(str[0]); ++i )  
        if( 'a'<=str[i] && str[i]<='z' )  
            str[i] -= ('a'-'A');  
    }  
  
char str[] = "aBcDe";  
  
cout << "str 字符长度为: " << sizeof(str)/sizeof(str[0]) << endl;  
  
UpperCase( str );  
  
cout << str << endl;
```

答：函数内的 sizeof 有问题。根据语法，sizeof 如用于数组，只能测出静态数组的大小，无法检测动态

---

分配的或外部数组大小。函数外的 str 是一个静态定义的数组，因此其大小为 6，函数内的 str 实际只是一个指向字符串的指针，没有任何额外的与数组相关的信息，因此 sizeof 作用于上只将其当指针看，一个指针为 4 个字节，因此返回 4。

## 6、什么是预编译？何时需要预编译：

答： 1、总是使用不经常改动的大型代码体。

2、程序由多个模块组成，所有模块都使用一组标准的包含文件和相同的编译选项。在这种情况下，可以将所有包含文件预编译为一个预编译头。

## 7、在 main 函数的 return 0; 代码之后还有什么程序运行么？

答： #include <stdlib.h>

```
int atexit(void (*function)(void));
```

此函数用以释放栈空间，保存环境等善后工作，可以自行根据需要重写此函数。

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

void bye(void) {
    printf("that was all, folks!\n");
}

int main() {
    long a;
    int i;
    i=atexit(bye);
    if(i!=0){
        fprintf(stderr,"cannot set exit function!\n");
    }
}
```

```
        return EXIT_FAILURE;  
    }  
    return EXIT_SUCCESS;  
}
```

#### 8、const 和#define 的区别，优缺点

答：#define 是一种宏，是一种代替的意思；而 const 是标准的 C++ 定义符，更能够体现定义的意思。

#define 没有类型信息，并且有全局的意义，在 c++ 中不被推荐使用；const 有类型信息，并且容易设定有效范围，是推荐的使用方式

#### 9、如何引用一个已经定义过的全局变量？

答：extern

可以用引用头文件的方式，也可以用 extern 关键字，如果用引用头文件方式来引用某个在头文件中声明的全局变理，假定你将那个变写错了，那么在编译期间会报错，如果你用 extern 方式引用时，假定你犯了同样的错误，那么在编译期间不会报错，而在连接期间报错

#### 10、局部变量能否和全局变量重名？

答：能，局部会屏蔽全局。要用全局变量，需要使用"::"

局部变量可以与全局变量同名，在函数内引用这个变量时，会用到同名的局部变量，而不会用到全局变量。对于有些编译器而言，在同一个函数内可以定义多个同名的局部变量，比如在两个循环体内都定义一个同名的局部变量，而那个局部变量的作用域就在那个循环体内

#### 11、重载 覆盖 隐藏的区别和执行方式

答：成员函数被重载的特征

- (1) 相同的范围（在同一个类中）；
- (2) 函数名字相同；
- (3) 参数不同；
- (4) `virtual` 关键字可有可无。

覆盖是指派生类函数覆盖基类函数，特征是

- (1) 不同的范围（分别位于派生类与基类）；
- (2) 函数名字相同；
- (3) 参数相同；
- (4) 基类函数必须有 `virtual` 关键字。

“隐藏”是指派生类的函数屏蔽了与其同名的基类函数，规则如下

- (1) 如果派生类的函数与基类的函数同名，但是参数不同。此时，不论有无 `virtual` 关键字，基类的函数将被隐藏（注意别与重载混淆）。
- (2) 如果派生类的函数与基类的函数同名，并且参数也相同，但是基类函数没有 `virtual` 关键字。此时，基类的函数被隐藏（注意别与覆盖混淆）

3 种情况怎么执行：1。重载：看参数 2。隐藏：用什么就调用什么 3。覆盖：调用派生类

## 12、c 语言局部优先问题

```
#include<stdio.h>
```

```
int a=100;
```

```
int main()
```

```
{
```

```
int a=10;  
//在此添加代码修改全局变量 a 的值  
}  
  
#include<stdio.h>  
  
int a=100;  
  
int main()  
{  
    int a=10;  
    {  
        extern int a;  
        a=5;  
    }  
}
```

13、用两个栈实现一个队列的功能？要求给出算法和思路！

答：设 2 个栈为 A, B，一开始均为空。

入队：

将新元素 push 入栈 A；

出队：

- (1) 判断栈 B 是否为空；
- (2) 如果不为空，则将栈 A 中所有元素依次 pop 出并 push 到栈 B；
- (3) 将栈 B 的栈顶元素 pop 出；

14、对于一个频繁使用的短小函数,在 C 语言中应用什么实现,在 C++中应用什么实现?

答: c 用宏定义, c++用 inline

15、以下三个有什么区别 ?

```
char * const p  
char const * p  
const char *p
```

char \* const p; //常量指针 , p 的值不可以修改

char const \* p ; //指向常量的指针 , 指向的常量值不可以改

const char \*p ; //和 char const \*p

```
char str1[] = "abc";  
char str2[] = "abc";  
  
const char str3[] = "abc";  
const char str4[] = "abc";
```

```
const char *str5 = "abc";  
const char *str6 = "abc";
```

```
char *str7 = "abc";  
char *str8 = "abc";
```

```
cout << ( str1 == str2 ) << endl;  
cout << ( str3 == str4 ) << endl;  
cout << ( str5 == str6 ) << endl;
```

---

```
cout << ( str7 == str8 ) << endl;
```

结果是：0 0 1 1

答：str1,str2,str3,str4 是数组变量，它们有各自的内存空间；  
而 str5,str6,str7,str8 是指针，它们指向相同的常量区域

## 16、虚继承

第一种情况：

```
class a
{
    virtual void func();
};
```

```
class b:public virtual a
```

```
{           //注意：b类中没有继承a类的成员
    virtual void foo();
};
```

第二种情况：

```
class a
{
    virtual void func();
};
```

```
class b:public a
{
    virtual void foo();
};
```

第三种情况：

```
class a
{
    virtual void func();
    char x;
};
```

```
class b:public virtual a
{
    virtual void foo();
};
```

第四种情况：

```
class a
{
    virtual void func();
    char x;
};
```

```
class b:public a
{
    virtual void foo();
};
```

对这四种情况分别求 sizeof(a) ,sizeof(b).

Linux 操作系统下运行结果:

第一种: 4, 4

第二种: 4, 4

第三种: 8, 12

第四种: 8, 8

17、多态实现的条件和机制?

答: 条件 a>继承

b>虚函数

c>指向子类的父类指针或者引用

机制:虚函数表

18、static 全局变量与普通的全局变量有什么区别? static 局部变量和普通局部变量有什么区别? static 函数与普通函数有什么区别?

答: 全局变量(外部变量)的说明之前再冠以 static 就构成了静态的全局变量。全局变量本身就是静态存储方式, 静态全局变量当然也是静态存储方式。这两者在存储方式上并无不同。这两者的区别虽在于

非静态全局变量的作用域是整个源程序，当一个源程序由多个源文件组成时，非静态的全局变量在各个源文件中都是有效的。而静态全局变量则限制了其作用域，即只在定义该变量的源文件内有效，在同一源程序的其它源文件中不能使用它。由于静态全局变量的作用域局限于一个源文件内，只能为该源文件内的函数公用，因此可以避免在其它源文件中引起错误。

从以上分析可以看出，把局部变量改变为静态变量后是改变了它的存储方式即改变了它的生存期。把全局变量改变为静态变量后是改变了它的作用域，限制了它的使用范围。

static 函数与普通函数作用域不同。仅在本文件。只在当前源文件中使用的函数应该说明为内部函数 (static)，内部函数应该在当前源文件中说明和定义。对于可在当前源文件以外使用的函数，应该在一个头文件中说明，要使用这些函数的源文件要包含这个头文件

static 全局变量与普通的全局变量有什么区别：static 全局变量只初使化一次，防止在其他文件单元中被引用；

static 局部变量和普通局部变量有什么区别：static 局部变量只被初始化一次，下一次依据上一次结果值；

static 函数与普通函数有什么区别：static 函数在内存中只有一份，普通函数在每个被调用中维持一份拷贝

## 19、进程和线程的差别。

答：线程是指进程内的一个执行单元，也是进程内的可调度实体。

与进程的区别：

(1) 调度：线程作为调度和分配的基本单位，进程作为拥有资源的基本单位

(2) 并发性：不仅进程之间可以并发执行，同一个进程的多个线程之间也可并发执行

- (3)拥有资源：进程是拥有资源的一个独立单位，线程不拥有系统资源，但可以访问隶属于进程的资源。
- (4)系统开销：在创建或撤消进程时，由于系统都要为之分配和回收资源，导致系统的开销明显大于创建或撤消线程时的开销。

## 20、试算运行结果：

```
main()
```

```
{
```

```
    int a[5]={1,2,3,4,5};
```

```
    int *ptr=(int *)(&a+1);
```

```
    printf("%d,%d",*(a+1),*(ptr-1));
```

```
}
```

输出：2,5

$*(a+1)$  就是  $a[1]$ ,  $*(ptr-1)$ 就是  $a[4]$ ,执行结果是 2, 5

$\&a+1$  不是首地址+1， 系统会认为加一个  $a$  数组的偏移， 是偏移了一个数组的大小（本例是 5 个 int）

```
int *ptr=(int *)(&a+1);
```

则  $ptr$  实际是  $\&(a[5])$ ,也就是  $a+5$

原因如下：

$\&a$  是数组指针， 其类型为  $int (*)[5]$ ;

而指针加 1 要根据指针类型加上一定的值，

不同类型的指针+1 之后增加的大小不同

a 是长度为 5 的 int 数组指针，所以要加  $5 * \text{sizeof}(\text{int})$

所以 ptr 实际是 a[5]

但是 prt 与  $(\&a+1)$  类型是不一样的(这点很重要)

所以 prt-1 只会减去  $\text{sizeof}(\text{int}^*)$

a, &a 的地址是一样的，但意思不一样，a 是数组首地址，也就是 a[0] 的地址，&a 是对象（数组）首地址，

a+1 是数组下一元素的地址，即 a[1], &a+1 是下一个对象的地址，即 a[5].

## 21、tcp 和 udp 通讯的区别

答：tcp 协议效率低，但是安全性好

udp 协议效率高，但是安全性低

## 22、虚函数和纯虚函数的区别

答：1 虚函数可以存在具体类中，也可以在抽象类中，设计的目的，是为了在子类重写这个函数，具有不同父类的功能。

2 纯虚函数只能存在于抽象类中，它设计的目的就是为了在子类对它进行实现，该子类才成为可以构造对象的具体类，否则也是抽象类

3 如果虚基类里有多个纯虚函数，子类继承的时候必须对每个纯虚函数都实现，才能成为可以实例化对象的类。

4 对于父类里的虚函数，子类可以继承过来不加修改，也可以修改之后实例化对象，但是纯虚函数必须都实现才能实例化对象。

## 23、简述链表和数组的区别.

答：从逻辑结构来看

数组必须事先固定的长度（元素个数），不能适应数据动态的增减的情况。当数据增加时，可能超出原先定义的元素个数；当数据减少时，造成内存浪费

链表动态的进行存储分配，可以适应数据动态地增减的情况，且可以方便的插入、删除数据项。

从内存存储来看

数组从栈中分配空间，对于程序员方便快捷，但是自由度小

链表从堆中分配空间，自由度大但是申请管理比较麻烦

## 24、如果 Win32 程序的消息处理函数的定义由

LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)  
修改为

LRESULT CALLBACK WndProc(MSG msg)

//MSG 结构体中的成员变量包含有 hWnd、message、wParam 和 lParam 等

这种修改可行吗？为什么？

答:不能修改,WndProc 是回调函数,操作系统预先定义了函数的输入参数和返回值

25、MFC 中， CWnd 类作为所有窗体封装类的根类，它定义的一些成员函数名称与 Win32 API 函数名称完全相同，且功能也完全相同，如 ShowWindow、UpdateWindow 等。但函数参数却不同，如：

CWnd 类 —— BOOL ShowWindow( int nCmdShow );

Win32 API —— BOOL ShowWindow(HWND hWnd, int nCmdShow);

问题：为什么 CWnd:: ShowWindow 函数少了一个参数也能实现相同的功能

答:这体现了累对变量和函数的共同封装。CWnd 类中为记录窗口句柄定义了类的成员变量，因此在调用 ShowWindow 函数时不用传入 HWND 参数，而是直接使用对象中保存的窗口句柄

## 二 程序题:

1、在计费系统的预处理程序中，对话单进行格式转换时，需要使用 strcpy 函数已知 strcpy, 此函数的

原型是

```
char *strcpy(char *strDest, const char *strSrc);
```

其中 strDest 是目的字符串， strSrc 是源字符串。编写 strcpy 函数

(1) 不调用 C++/C 的字符串库函数，请编写函数 strcpy

```
char *mystrcpy(char *s1,const char *s2/*,size_t n*/)
```

```
{  
    char *t=s1;  
    while((*s1++=*s2++)!='\0');  
    /*while(1)  
    {  
        *s1=*s2;  
        if(*s1=='\0')break;  
        s1++;  
        s2++;  
    }*/  
    return t;  
}
```

(2) strcpy 能把 strSrc 的内容复制到 strDest，为什么还要 char \* 类型的返回值？

答：是为了防止读取指针 strDest 和 strSrc 时发生错误，如果上面两个指针无效（是空指针或者是野指针），返回 NULL 表示函数失败

2、在电信业务的后台处理程序中，经常会涉及到处理字符串，除了用 `char *` 处理字符串之外，C++ 还为我们提供了封装了的字符串类 `string`，其本质也是用一个动态数组来保存字符串，类 `String` 的原型为：

```
class String
{
public:
    String(const char *str = NULL); // 普通构造函数
    String(const String &other); // 拷贝构造函数
    ~String(void); // 析构函数
    String & operator =(const String &other); // 赋值函数
private:
    char *m_data; // 用于保存字符串
};
```

请编写 `String` 的上述 4 个函数普通构造函数、拷贝构造函数、析构函数和赋值函数。

```
String::String(const char* str=NULL) {
    if(!str) {
        m_data=new char[1];
        *m_data=' \0' ;
    } else {
        int length=strlen(str);
        m_data=new char[length+1];
```

```
        strcpy(m_data, str);

    }

String::~String() {
    delete m_data;
}

String::String(const String& other) {
    int length=strlen(other.m_data);
    m_data=new char[length+1];
    strcpy(m_data, other.m_data);
}

String::String& operator=(const String& other) {
    if(this==&other)
        return *this;
    delete m_data;
    int length=strlen(other.m_data);
    m_data=new char[length+1];
    strcpy(m_data, other.m_data);
    return *this;
}
```

### 3、写一个单链表的创建，插入，删除

```
#include<iostream>
#include<stdio.h>
using namespace std;
typedef struct student{
    int data;
    struct student* next;
}node;
node *creat() {
    node *head, *p1, *p2;
    head = node* malloc(sizeof(node));
    p1 = head;
    int x, flag = 1;
    while(flag) {
        printf("please input a num\n");
        scanf("%d\n", &x);
        if(x != 0) {
            p2 = (node*) malloc(sizeof(node));
            p2->data = x;
            p1->next = p2;
            p1 = p2;
        }
    }
}
```

```
    }

    else

        flag = 0;

    }

    head = head ->next;

    p1 ->next = NULL;

    return head;

}

node* del(node *head, int num) {

    node *p1, *p2;

    p1 = head;

    while(num != p1 ->data) {

        p2 = p1;

        p1 = p1 ->next;

    }

    if(num == p1 ->data) {

        if(p1 == head) {

            head = p1 -> next;

            free(p1);

        }

        else
```

```
p2 ->next = p1 ->next;  
}  
  
else  
printf("could not found %d\n", num);  
return head;  
}  
  
node* insert(node* head, int num) {  
    node *p0, *p1, *p2;  
    p1 = head;  
    p0 = (node*) malloc(sizeof(node));  
    p0 ->data = num;  
    while(p0 ->data > p1 ->data && p1 ->next != NULL) {  
        p2 = p1;  
        p1 = p1 ->next;  
    }  
    if(p0 ->data <= p1 ->data) {  
        if(head == p1) {  
            p0 ->next = p1;  
            head = p0;  
        }  
        else {
```

```
p2 ->next = p0;  
  
p0 ->next = p1;  
  
}  
  
}  
  
else{  
  
    p1 ->next = p0;  
  
    p0 ->next = NULL;  
  
}  
  
return head;  
}  
  
int main() {  
  
    node *head, stud;  
  
    int n, del_num, insert_num;  
  
    head = creat();  
  
    printf("\nplease input the del_num:");  
  
    scanf("%d\n", del_num);  
  
    head = del(head, del_num);  
  
    printf("\nplease input the insert_num:");  
  
    scanf("%d\n", insert_num);  
  
    head = insert(head, insert_num);  
  
    return 0;
```

}

#### 4、写一个快速排序

```
#include <iostream>
using namespace std;
void QuickSort(int nData[],int iLeft,int iRight){
    int iPivot=(iLeft+iRight)/2;
    int nPivot=nData[iPivot];
    for(int i=iLeft,j=iRight;i<j;){
        while(!(i>iPivot||nData[i]>nPivot))
            i++;
        if(i<iPivot){
            nData[iPivot]=nData[i];
            iPivot=i;
        }
        while(!(j<iPivot||nData[j]<nPivot))
            j--;
        if(j>iPivot){
            nData[iPivot]=nData[j];
            iPivot=j;
        }
    }
}
```

```
nData[iPivot]=nPivot;  
if((iPivot-iLeft)>1)  
    QuickSort(nData,iLeft,iPivot-1);  
if((iRight-iPivot)>1)  
    QuickSort(nData,iPivot+1,iRight);  
}
```

5、编写一个函数，把一个整数以十六进制形式输出.

```
#include <stdio.h>  
  
void changeDecToHex(int nDec) {  
    char str[100];  
    int i=0, j=0, temp;  
    while (nDec) {  
        temp=nDec%16;  
        if (temp<10)  
            str[i]=temp+48;  
        else  
            Str[i]=temp-10+65;  
        nDec=nDec/16;  
        i++;  
    }
```

```
printf("the Hex new string is:\n");

for(j=i-1;j>=0;j--)
    printf("%c", str[j]);
printf("\n");

}

int main() {
    int nDec=0;
    scanf("%d", &nDec);
    changeDecToHex(nDec);
}
```

### 经典嵌入式面试题

C 语言测试是招聘嵌入式系统程序员过程中必须而且有效的方法。这些年，我既参加也组织了许多这种测试，在这过程中我意识到这些测试能为带面试者和被面试者提供许多有用信息，此外，撇开面试的压力不谈，这种测试也是相当有趣的。

从被面试者的角度来讲，你能了解许多关于出题者或监考者的情况。这个测试只是出题者为显示其对 ANSI 标准细节的知识而不是技术技巧而设计吗？这个愚蠢的问题吗？如要你答出某个字符的 ASCII 值。这些问题着重考察你的系统调用和内存分配策略方面的能力吗？这标志着出题者也许花时间在微机上而不上在嵌入式系统上。如果上述任何问题的答案是“是”的话，那么我知道我得认真考虑我是否应该去做这份工作。

从面试者的角度来讲，一个测试也许能从多方面揭示应试者的素质：

最基本的，你能了解应试者 C 语言的水平。不管怎么样，看一下这人如何回答他不会的问题也是满有趣。应试者是以好的直觉做出明智的选择，还是只是瞎蒙呢？当应试者在某个问题上卡住时是找借口呢，还是表现出对问题的真正的好奇心，把这看成学习的机会呢？我发现这些信息与他们的测试成绩一样有用。

有了这些想法，我决定出一些真正针对嵌入式系统的考题，希望这些令人头痛的考题能给正在找工作的人一点帮住。这些问题都是我这些年实际碰到的。其中有些题很难，但它们应该都能给你一点启迪。

这个测试适于不同水平的应试者，大多数初级水平的应试者的成绩会很差，经验丰富的程序员应该有很好的成绩。为了让你能自己决定某些问题的偏好，每个问题没有分配分数，如果选择这些考题为你所用，请自行按你的意思分配分数。

### 预处理器（Preprocessor）

1. 用预处理指令`#define` 声明一个常数，用以表明 1 年中有多少秒（忽略闰年问题）

```
#define SECONDS_PER_YEAR (60 * 60 * 24 * 365)UL
```

我在这想看到几件事情：

?; `#define` 语法的基本知识（例如：不能以分号结束，括号的使用，等等）

?; 懂得预处理器将为你计算常数表达式的值，因此，直接写出你是如何计算一年中有多少秒而不是计算出实际的值，是更清晰而没有代价的。

?; 意识到这个表达式将使一个 16 位机的整型数溢出-因此要用到长整型符号 `L`,告诉编译器这个常数是的长整型数。

?; 如果你在你的表达式中用到 `UL`（表示无符号长整型），那么你有了一个好的起点。记住，第一印象很重要。

2. 写一个"标准"宏 `MIN`，这个宏输入两个参数并返回较小的一个。

---

```
#define MIN(A,B) ( (A) <= (B) ? (A) : (B) )
```

这个测试是为下面的目的而设的：

?; 标识#define 在宏中应用的基本知识。这是很重要的，因为直到嵌入(inline)操作符变为标准 C 的一部分，宏是方便产生嵌入代码的唯一方法，对于嵌入式系统来说，为了能达到要求的性能，嵌入代码经常是必须的方法。

?; 三重条件操作符的知识。这个操作符存在 C 语言中的原因是它使得编译器能产生比 if-then-else 更优化的代码，了解这个用法是很重要的。

?; 懂得在宏中小心地把参数用括号括起来

?; 我也用这个问题开始讨论宏的副作用，例如：当你写下面的代码时会发生什么事？`least = MIN(*p++, b);`

### 3. 预处理器标识#error 的目的是什么？

`#error` 停止编译并显示错误信息

如果你不知道答案，请看参考文献 1。这问题对区分一个正常的伙计和一个书呆子是很有用的。只有书呆子才会读 C 语言课本的附录去找出象这种问题的答案。当然如果你不是在找一个书呆子，那么应试者最好希望自己不要知道答案。

死循环（Infinite loops）

### 4. 嵌入式系统中经常要用到无限循环，你怎么样用 C 编写死循环呢？

这个问题用几个解决方案。我首选的方案是：

```
while(1)
{
}
```

一些程序员更喜欢如下方案：

```
for(;;)
```

{  
}

这个实现方式让我为难，因为这个语法没有确切表达到底怎么回事。如果一个应试者给出这个作为方案，我将用这个作为一个机会去探究他们这样做的基本原理。如果他们的基本答案是：“我被教着这样做，但从没有想到过为什么。”这会给我留下一个坏印象。

第三个方案是用 `goto`

`Loop:`

...

`goto Loop;`

应试者如给出上面的方案，这说明或者他是一个汇编语言程序员（这也许是好事）或者他是一个想进入新领域的 BASIC/FORTRAN 程序员。

数据声明（Data declarations）

5. 用变量 `a` 给出下面的定义

a) 一个整型数（An integer） `int a`

b) 一个指向整型数的指针（A pointer to an integer） `int *a`

c) 一个指向指针的指针，它指向的指针是指向一个整型数（A pointer to a pointer to an integer） `r *(int *a) int**a`

d) 一个有 10 个整型数的数组（An array of 10 integers） `int a[10]`

e) 一个有 10 个指针的数组，该指针是指向一个整型数的。（An array of 10 pointers to integers） `int *a[10]`

f) 一个指向有 10 个整型数数组的指针（A pointer to an array of 10 integers） `* (int a[10]) int (*a)[10]`

g) 一个指向函数的指针，该函数有一个整型参数并返回一个整型数（A pointer to a function that takes an integer as an argument and returns an integer） `fun(*int a) int (*max_function)(int a)`

h) 一个有 10 个指针的数组，该指针指向一个函数，该函数有一个整型参数并返回一个整型数（An array of ten pointers to functions that

---

take an integer argument and return an integer ) **fun(\*int a[10])**

**int (\*a[10])(int)**

答案是：

- a) int a; // An integer
- b) int \*a; // A pointer to an integer
- c) int \*\*a; // A pointer to a pointer to an integer
- d) int a[10]; // An array of 10 integers

e) int \*a[10]; // An array of 10 pointers to integers

等价于 `int *(a[10]);`

f) int (\*a)[10]; // A pointer to an array of 10 integers

g) int (\*max\_function)(int a); // A pointer to a function a that takes an integer argument and returns an integer

h) int (\*a[10])(int); // An array of 10 pointers to functions that take an integer argument and return an integer

人们经常声称这里有几个问题是那种要翻一下书才能回答的问题，我同意这种说法。当我写这篇文章时，为了确定语法的正确性，我的确查了一下书。但是当我被面试的时候，我期望被问到这个问题（或者相近的问题）。因为在被面试的这段时间里，我确定我知道这个问题的答案。应试者如果不知道所有的答案（或至少大部分答案），那么也就没有为这次面试做准备，如果该面试者没有为这次面试做准备，那么他又能为什么出准备呢？

## 6. 关键字 **static** 的作用是什么？

在 C 语言中，关键字 **static** 有三个明显的作用：

一旦声明为静态变量，在编译时刻开始永远存在，不受作用域范围约束，但是如果是局部静态变量，则此静态变量只能在局部作用域内使用，超出范围不能使用，但是它确实还占用内存，还存在。

?; 在模块内（但在函数体外），一个被声明为静态的变量可以被模块内所用函数访问，但不能被模块外其它函数访问。它是一个本地的全局变量。

?; 在模块内，一个被声明为静态的函数只可被这一模块内的其它函数调用。那就是，这个函数被限制在声明它的模块的本地范围内使用。大多数应试者能正确回答第一部分，一部分能正确回答第二部分，很少人能懂得第三部分。这是一个应试者的严重的缺点，因为他显然不懂得本地化数据和代码范围的好处和重要性。

## 7. 关键字 **const** 有什么含意？

总结：1) 只读。2) 使用关键字 **const** 也许能产生更紧凑的代码。3) 使编译器很自然地保护那些不希望被改变的参数，防止其被无意的代码修改。

我只要一听到被面试者说：“**const** 意味着常数”，我就知道我正在和一个业余者打交道。去年 Dan Saks 已经在他的文章里完全概括了 **const** 的所有用法，因此 **ESP**(译者：**Embedded Systems Programming**)的每一位读者应该非常熟悉 **const** 能做什么和不能做什么。如果你从没有读到那篇文章，只要能说出 **const** 意味着“只读”就可以了。尽管这个答案不是完全的答案，但我接受它作为一个正确的答案。(如果你想知道更详细的答案，仔细读一下 **Saks** 的文章吧。)

如果应试者能正确回答这个问题，我将问他一个附加的问题：

下面的声明都是什么意思？

```
const int a;  
int const a;  
const int *a;  
int * const a;  
int const * a const;
```

\*\*\*\*\*

前两个的作用是一样，**a** 是一个常整型数。

第三个意味着 **a** 是一个指向常整型数的指针(也就是，整型数是不可修改的，但指针可以)。

第四个意思 **a** 是一个指向整型数的常指针(也就是说，指针指向的整型数是可以修改的，但指针是不可修改的)。

最后一个意味着 **a** 是一个指向常整型数的常指针（也就是说，指针指向的整型数是不可修改的，同时指针也是不可修改的）。

如果应试者能正确回答这些问题，那么他就给我留下了一个好印象。

顺带提一句，也许你可能会问，即使不用关键字 **const**，也还是能很容易写出功能正确的程序，那么我为什么还要如此看重关键字 **const** 呢？我也如下的几下理由：

?; 关键字 **const** 的作用是为给读你代码的人传达非常有用的信息，实际上，声明一个参数为常量是为了告诉了用户这个参数的应用目的。如果你曾花很多时间清理其它人留下的垃圾，你就会很快学会感谢这点多余的信息。（当然，懂得用 **const** 的程序员很少会留下的垃圾让别人来清理的。）

?; 通过给优化器一些附加的信息，使用关键字 **const** 也许能产生更紧凑的代码。

?; 合理地使用关键字 **const** 可以使编译器很自然地保护那些不希望被改变的参数，防止其被无意的代码修改。简而言之，这样可以减少 **bug** 的出现。

## 8. 关键字 **volatile** 有什么含意？并给出三个不同的例子。

一个定义为 **volatile** 的变量是说这变量可能会被意想不到地改变，这样，编译器就不会去假设这个变量的值了。精确地说就是，优化器在用到这个变量时必须每次都小心地重新读取这个变量的值，而不是使用保存在寄存器里的备份。

下面是 **volatile** 变量的几个例子：

?; 并行设备的硬件寄存器（如：状态寄存器）

?; 一个中断服务子程序中会访问到的非自动变量(Non-automatic variables)

?; 多线程应用中被几个任务共享的变量

回答不出这个问题的人是不会被雇佣的。我认为这是区分 C 程序员和嵌入式系统程序员的最基本的问题。搞嵌入式的家伙们经常同硬

件、中断、RTOS 等等打交道，所有这些都要求用到 **volatile** 变量。

不懂得 **volatile** 的内容将会带来灾难。

假设被面试者正确地回答了这是问题（嗯，怀疑是否会是这样），我将稍微深究一下，看一下这家伙是不是真正懂得 **volatile** 完全的重要性。

?; 一个参数既可以是 **const** 还可以是 **volatile** 吗？解释为什么。

?; 一个指针可以是 **volatile** 吗？解释为什么。

?; 下面的函数有什么错误：int square(**volatile int** \*ptr){return \*ptr \*  
\*ptr;}

下面是答案：

?; 是的。一个例子是只读的状态寄存器。它是 **volatile** 因为它可能被意想不到地改变。它是 **const** 因为程序不应该试图去修改它。

?; 是的。尽管这并不很常见。一个例子是当一个中断服务子程序修改一个指向一个 **buffer** 的指针时。

?; 这段代码有点变态。这段代码的目的是用来返回指针\*ptr 指向值的平方，但是，由于\*ptr 指向一个 **volatile** 型参数，编译器将产生类似下面的代码：

```
int square(volatile int *ptr)
{
int a,b;
a = *ptr;
b = *ptr;
return a * b;
}
```

由于\*ptr 的值可能被意想不到地该变，因此 a 和 b 可能是不同的。结果，这段代码可能返不是你所期望的平方值！正确的代码如下：

```
long square(volatile int *ptr)
{
int a;
```

```
a = *ptr;  
return a * a;  
}
```

### 位操作 (Bit manipulation)

9. 嵌入式系统总是要用户对变量或寄存器进行位操作。给定一个整型变量 **a**, 写两段代码, 第一个设置 **a** 的 bit 3, 第二个清除 **a** 的 bit 3。在以上两个操作中, 要保持其它位不变。

对这个问题有三种基本的反应

?; 不知道如何下手。该被面者从没做过任何嵌入式系统的工作。

?; 用 **bit fields**。**Bit fields** 是被扔到 C 语言死角的东西, 它保证你的代码在不同编译器之间是不可移植的, 同时也保证了你的代码是不可重用的。我最近不幸看到 Infineon 为其较复杂的通信芯片写的驱动程序, 它用到了 **bit fields** 因此完全对我无用, 因为我的编译器用其它的方式来实现 **bit fields** 的。从道德讲: 永远不要让一个非嵌入式的家伙粘实际硬件的边。

?; 用 **#defines** 和 **bit masks** 操作。这是一个有极高可移植性的方法, 是应该被用到的方法。最佳的解决方案如下:

```
#define BIT3 (0x1 << 3)  
static int a;  
void set_bit3(void) {a |= BIT3;}  
void clear_bit3(void) {a &= ~BIT3;}
```

一些人喜欢为设置和清除值而定义一个掩码同时定义一些说明常数, 这也是可以接受的。我希望看到几个要点: 说明常数、**|=**和**&=****~**操作。

### 访问固定的内存位置 (Accessing fixed memory locations)

10. 嵌入式系统经常具有要求程序员去访问某特定的内存位置的特点。在某工程中, 要求设置一绝对地址为 0x67a9 的整型变量的值为 0xaa55。

编译器是一个纯粹的 ANSI 编译器。写代码去完成这一任务。

这一问题测试你是否知道为了访问一绝对地址把一个整型数强制转换 (typecast) 为一指针是合法的。这一问题的实现方式随着个人风格不同而不同。典型的类似代码如下：

```
int *ptr;  
ptr = (int *)0x67a9;  
*ptr = 0xaa55;
```

A more obscure approach is:

一个较晦涩的方法是：

```
*(int * const)(0x67a9) = 0xaa55;
```

即使你的品味更接近第二种方案，但我建议你在面试时使用第一种方案。

## 中断 (Interrupts)

11. 中断是嵌入式系统中重要的组成部分，这导致了很多编译开发商提供一种扩展——让标准 C 支持中断。具代表事实是，产生了一个新的关键字 `_interrupt`。下面的代码就使用了 `_interrupt` 关键字去定义了一个中断服务子程序(ISR)，请评论一下这段代码的。

```
_interrupt double compute_area (double radius)  
{  
    double area = PI * radius * radius;  
    printf("\nArea = %f", area);  
    return area;  
}
```

这个函数有太多的错误了，以至让人不知从何说起了：

?; ISR 不能返回一个值。如果你不懂这个，那么你不会被雇用的。

?; ISR 不能传递参数。如果你没有看到这一点，你被雇用的机会等同第一项。

?; 在许多的处理器/编译器中，浮点一般都是不可重入的。有些处理

器/编译器需要让额处的寄存器入栈，有些处理器/编译器就是不允许在 ISR 中做浮点运算。此外，ISR 应该是短而有效率的，在 ISR 中做浮点运算是不明智的。

?; 与第三点一脉相承，printf()经常有重入和性能上的问题。如果你丢掉了第三和第四点，我不会太为难你的。不用说，如果你能得到后两点，那么你的被雇用前景越来越光明了。

\*\*\*\*\*

### 代码例子 (Code examples)

12. 下面的代码输出是什么，为什么？

```
void foo(void)
{
    unsigned int a = 6;
    int b = -20;
    (a+b > 6) ? puts("> 6") : puts("<= 6");
}
```

这个问题测试你是否懂得 C 语言中的整数自动转换原则，我发现有些开发者懂得极少这些东西。不管如何，这无符号整型问题的答案是输出是 ">6"。

原因是当表达式中存在有符号类型和无符号类型时所有的操作数都自动转换为无符号类型。

因此-20 变成了一个非常大的正整数，所以该表达式计算出的结果大于 6。这一点对于应当频繁用到无符号数据类型的嵌入式系统来说是非常重要的。如果你答错了这个问题，你也就到了得不到这份工作的边缘。

13. 评价下面的代码片断：

```
unsigned int zero = 0;
unsigned int compzero = 0xFFFF; //1's complement of zero
```

对于一个 int 型不是 16 位的处理器来说，上面的代码是不正确的。

应编写如下：

---

```
unsigned int compzero = ~0;
```

这一问题真正能揭露出应试者是否懂得处理器字长的重要性。在我的经验里，好的嵌入式程序员非常准确地明白硬件的细节和它的局限，然而 PC 机程序往往把硬件作为一个无法避免的烦恼。

到了这个阶段，应试者或者完全垂头丧气了或者信心满满志在必得。如果显然应试者不是很好，那么这个测试就在这里结束了。但如果显然应试者做得不错，那么我就扔出下面的追加问题，这些问题是比较难的，我想仅仅非常优秀的应试者能做得不错。提出这些问题，我希望更多看到应试者应付问题的方法，而不是答案。不管如何，你就当是这个娱乐吧...

#### 动态内存分配 (Dynamic memory allocation)

14. 尽管不像非嵌入式计算机那么常见，嵌入式系统还是有从堆 (**heap**) 中动态分配内存的过程的。那么嵌入式系统中，动态分配内存可能发生的问题是什么？

这里，我期望应试者能提到内存碎片，碎片收集的问题，变量的持行时间等等。这个主题已经在 **ESP** 杂志中被广泛地讨论过了（主要是 **P.J. Plauger**，他的解释远远超过我这里能提到的任何解释），所有回过头看一下这些杂志吧！让应试者进入一种虚假的安全感觉后，我拿出这么一个小节目：

下面的代码片段的输出是什么，为什么？

```
char *ptr;  
if ((ptr = (char *)malloc(0)) == NULL) puts("Got a null pointer");  
else puts("Got a valid pointer");
```

这是一个有趣的问题。最近在我的一个同事不经意把 0 值传给了函数 **malloc**，得到了一个合法的指针之后，我才想到这个问题。这就是上面的代码，该代码的输出是“**Got a valid pointer**”。我用这个来开始讨论这样的一问题，看看被面试者是否想到库例程这样做是正确。得到正确的答案固然重要，但解决问题的方法和你做决定的基本原理更重

---

要些。

15 **Typedef** 在 C 语言中频繁用以声明一个已经存在的数据类型的同义字。也可以用预处理器做类似的事。例如，思考一下下面的例子：

```
#define dPS struct s *
typedef struct s * tPS;
```

以上两种情况的意图都是要定义 **dPS** 和 **tPS** 作为一个指向结构 **s** 指针。哪种方法更好呢？（如果有的话）为什么？这是一个非常微妙的问题，任何人答对这个问题（正当的原因）是应当被恭喜的。答案是：**typedef** 更好。思考下面的例子：

```
dPS p1,p2;
tPS p3,p4;
```

第一个扩展为 **struct s \* p1, p2;**

上面的代码定义 **p1** 为一个指向结构的指针，**p2** 为一个实际的结构，这也许不是你想要的。

第二个例子正确地定义了 **p3** 和 **p4** 两个指针。

晦涩的语法

16 . C 语言同意一些令人震惊的结构，下面的结构是合法的吗，如果是它做些什么？

```
int a = 5, b = 7, c;
c = a+++b;
```

这个问题将做为这个测验的一个愉快的结尾。不管你相不相信，上面的例子是完全合乎语法的。问题是编译器如何处理它？水平不高的编译作者实际上会争论这个问题，根据最处理原则，编译器应当能处理尽可能所有合法的用法。因此，上面的代码被处理成：**c = a++ + b;** 因此，这段代码执行后 **a = 6, b = 7, c = 12**。

---

如果你知道答案，或猜出正确答案，做得好。如果你不知道答案，我也不把这个当作问题。我发现这个问题的最大好处是这是一个关于代码编写风格，代码的可读性，代码的可修改性的好的话题。

好了，伙计们，你现在已经做完所有的测试了。这就是我出的 C 语言测试题，我怀着愉快的心情写完它，希望你以同样的心情读完它。如果是认为这是一个好的测试，那么尽量都用到你的找工作的过程中去吧。

**memcmp**

Compare characters in two buffers.

比较两块内存中的字符

`int memcmp( const void *buf1, const void *buf2, size_t count );`

**memset**

Sets buffers to a specified character

将内存块设置为指定的字符

**sprintf**

Write formatted data to a string

将格式化的数据写到字符串

## 一、单项选择题

1、如下哪一个命令可以帮助你知道 shell 命令的用法 ( A )

- A. man                    B. pwd                    C. help                    D. more

2、Linux 分区类型默认的是 : ( B )

- A. vfat                    B. ext2/ext3                    C. swap                    D. dos

3、在大多数 Linux 发行版本中，以下哪个属于块设备 ( B )

A. 串行口      B. 硬盘      C. 虚拟终端      D. 打印机

4、下面哪个命令行可用来马上重新启动正在运行的 Linux 系统？( D )

A. restart --delay=0    B. reboot -w  
C. halt -p            D. shutdown -r now

5、在 Linux 系统，默认的 shell 是什么 ( A )

A.bash      B.ash      C.csh      D.gnush

6、下面哪条命令可用来确保文件“myfile”存在 ( B )

A. cp myfile /dev/null    B. touch myfile  
C. create myfile        D. mkfile myfile

7、 LILO 的配置文件是 : ( B )

A. /etc/conf            B. /etc/lilo.conf  
C. /proc/kcore        D. /usr/local/

8、用“useradd jerry”命令添加一个用户，这个用户的主目录是什么 ( A )

A./home/jerry            B./bin/jerry  
C./var/jerry            D./etc/jerry

9、Linux 文件权限一共 10 位长度，分成四段，第三段表示的内容是 ( D )

A.文件类型            B.文件所有者的权限  
C.文件所有者所在组的权限    D.其他用户的权限

10、某文件的组外成员的权限为只读；所有者有全部权限；组内的权限为读与写，则该文件的权限为 ( D )

A.467            B.674            C.476            D.764

11、不是 shell 具有的功能和特点的是 ( A )

A.管道            B.输入输出重定向  
C.执行后台进程    D.处理程序命令

12、如何从当前系统中卸载一个已装载的文件系统 ( A )

A. umount            B. dismount  
C. mount -u        D. 从 /etc/fstab 中删除这个文件系统项

13、你用 vi 编辑器编写了一个脚本文件 shell.sh ,你想将改文件名称修改为 shell2.sh ,下列命令( B )

可以实现。

- A. cp shell.sh shell2.sh      B. mv shell.sh shell2.sh  
C. ls shell.sh >shell2.sh      D. ll shell.sh >shell2.sh

14、在/home/stud1/wang 目录下有一文件 file , 使用 ( D ) 可实现在后台执行命令 , 此命令将 file 文件中的内容输出到 file.copy 文件中。

- A. cat file >file.copy      B. cat file file.copy  
C. &cat file file.copy      D. &cat file >file.copy

15、字符设备文件类型的标志是 ( B )

- A. p      B. c      C. s      D. l

16、删除文件命令为 ( D )

- A. mkdir      B. rmdir      C. mv      D. rm

17、( B ) 命令可更改一个文件的权限设置 ?

- A. attrib      B. chmod      C. change      D. file

18、用命令 ls -al 显示出文件 ff 的描述如下所示 , 由此可知文件 ff 的类型为 ( A )

-rwxr-xr-- 1 root root 599 Cec 10 17:12 ff

- A. 普通文件      B. 硬链接      C. 目录      D. 符号链接

19、系统中有用户 user1 和 user2 , 同属于 users 组。在 user1 用户目录下有一文件 file1 , 它拥有 644 的权限 , 如果 user2 用户想修改 user1 用户目录下的 file1 文件 , 应拥有 ( B ) 权限。

- A. 744      B. 664      C. 646      D. 746

20、在指令系统的各种寻址方式中 , 获取操作数最快的方式是 ( 1 -B ); 若操作数的地址包含在指令中 , 则属于 ( 2-A ) 方式。

- ( 1 )      A、直接寻址      B、立即寻址      C、寄存器寻址      D、间接寻址  
( 2 )      A、直接寻址      B、立即寻址      C、寄存器寻址      D、间接寻址

21、在 CPU 和物理内存之间进行地址转换时 ,( B ) 将地址从虚拟 ( 逻辑 ) 地址空间映射到物理地址空间。

- A、TCB      B、MMU      C、CACHE      D、DMA

22、Linux 将存储设备和输入 / 输出设备均看做文件来操作 ,( C ) 不是以文件的形式出现。

- 
- A. 目录
  - B. 软链接
  - C. i 节点表
  - D. 网络适配器

23、关于文件系统的安装和卸载，下面描述正确的是 ( A )。

- A. 如果光盘未经卸载，光驱是打不开的
- B. 安装文件系统的安装点只能是 /mnt 下
- C. 不管光驱中是否有光盘，系统都可以安装 CD-ROM 设备
- D. mount /dev/fd0 /floppy 此命令中目录 /floppy 是自动生成的

24、为了查看 Linux 启动信息，可以用 ( B ) 命令

- A. cat /etc/lilo.conf
- B. dmesg
- C. cat /proc/cpuinfo
- D. lilo

25、用下列 ( A ) 命令查看 Linux 使用了多少内存

- A. cat /proc/meminfo
- B. cat /bin/meminfo
- C. vi /proc/meminfo
- D. vi /user/local/meminfo

26、下列 ( D ) 设备是字符设备。

- A. hdc
- B. fd0
- C. hda1
- D. tty1

27、下列说法正确的是 ( D )

- A. ln -s a.txt b.txt，作用是制作文件 b.txt 的符号链接，其名称为 a.txt
- B. df 命令可以查看当前目录占用磁盘空间的大小
- C. comm 命令打印两个文本文件中的相同的内容
- D. rm 命令可以用来删除目录

28、有如下的命令：\$ dd if=f1 of=f2。其中 if=f1 表示 ( A )

- A. 以 f1 作为源文件，代替标准输入
- B. 以 f1 作为目标文件，代替标准输出
- C. 当条件满足 f1 的时候，执行真正的拷贝
- D. 拷贝的过程中，不转化文件

29、为了查找出当前用户运行的所有进程的信息，我们可以使用（B）命令：

- A. ps -a      B. ps -u      C. ls -a      D. ls -l

30、为保证在启动服务器时自动启动 DHCP 进程，应对（ B ）文件进行编辑。

- A、/etc/rc.d/rc.inet2                          B、/etc/rc.d/rc.inet1  
C、/etc/dhcpd.conf                                D、/etc/rc.d/rc.S

31、（ D ）设备是字符设备。

- A、hdc                                            B、fd0                                    C、hda1                                    D、tty1

32、文件 exer1 的访问权限为 rw-r--r--，现要增加所有用户的执行权限和同组用户的写权限，下列命令正确的是（ A ）。

- A、chmod a+x g+w exer1                    B、chmod 765 exer1  
C、chmod o+x exer1                            D、chmod g+w exer1

33、删除当前目录 abc 以及下面的所有子目录和文件，并不要求提示任何确认信息的命令是（ B ）

- A. del abc\\*.\*      B. rm -rf abc      C. rmdir abc      D. rm -r abc\ \*.\*

34、如果忘记了 ls 命令的用法，可以采用（ C ）命令获得帮助

- a. ?ls      b.help ls      c.man ls      d.get ls

35、在安装开始前，用光盘启动系统，想要进入字符界面安装，需要输入的命令是（ C ）

- a.linux doc      b.linux      c.linux text      d.linux note

36、要给文件 file1 加上其他人可执行属性的命令是（ C ）

- a.chmod a+x      b.chown a+x      c.chmod o+x      d.chown o+x

37、怎样新建一个新文件：( A )

- a.touch hello.c      b.mk hello.c      c.rm hello.c      d.new hello.c

38、在 bash 命令中，当用（ B ）参数时，表示 bash 是交互的。

- A、-c      B、-i      C、-s      D、-d

39、重定向的符号">"表示：( C )

- A、输出追加      B、输入追加      C、输出重定向，原来的文件会被改写      D、管道

40、linux 系统能够直接读取的分区类型是 ( D )

- a.ntfs      b.fat16      c.fat32      d.ext3

41、下列提法中，属于 ifconfig 命令作用范围的是 ( B )。

- A、编译源程序                          B、配置网卡的 IP 地址  
C、配置系统内核                          D、加载网卡到内核中

42、下列对 shell 变量 FRUIT 操作，正确的是 ( C )

- A、为变量赋值：\$FRUIT=apple                  B、显示变量的值：fruit=apple  
C、显示变量的值：echo \$FRUIT                    D、判断变量是否有值：[ -f "\$FRUIT" ]

43、一般可以用 ( C ) 实现自动编译。

- A、gcc      B、gdb \*      C、make      D、vi

44、处理机主要由处理器、存储器和总线组成，总线包括 ( D )。

- A、数据总线、串行总线、逻辑总线、物理总线  
B、并行总线、地址总线、逻辑总线、物理总线  
C、并行总线、串行总线、全双工总线  
D、数据总线、地址总线、控制总线

45、假设当前目录下有文件 Makefile，下面是其内容：

```
pr1: prog.o subr.o
gcc -o pr1 prog.o subr.o
prog.o: prog.c prog.h
gcc -c -l prog.o prog.c
subr.o: subr.c
gcc -c -o subr.o subr.c
clear:
rm -f pr1*.o
```

现在执行命令 make clear，实际执行的命令是 ( A )：

- A. rm -f pr1\*.o  
B. gcc -c -l prog.o prog.c

---

```
C. gcc -c -o subr.o subr.c
```

D. 都执行

46、Linux 将存储设备和输入/输出设备均看做文件来操作，下列选项 (c) 不是以文件的形式出现。

## A. 目录

## B. 软链接

### C. i 节点表

#### D. 网络适配器

47、有如下的命令：\$dd if=f1 of=f2。其中 if=f1 表示（ A ）

#### A. 以 f1 作为源文件，代替标准输入

B. 以 `f1` 作为目标文件，代替标准输出

c. 当条件满足  $f_1$  的时候，执行真正的拷贝

D. 拷贝的过程中，不转化文件

48. 文件之间可以建立两种链接关系：软链接和硬链接，硬链接的特点是（c）

#### A. 等同于文件复制操作

B. 类似于文件复制，但新的链接文件并不占用文件磁盘存储空间

C. 删除源文件，将使其他链接文件失效

D. 可以对目录文件名建立硬链接

49. 下面哪一个选项不是 linux 系统的进程类型 ( D )

#### A. 交互进程

### B. 批处理进程

### c. 守护进程

#### D. 就绪进程

50、下面(B)特性不符合嵌入式操作系统特点。

A、实时性 B、不可定制

C、微型化 D、易移植

51、下面关于 C 语言程序的描述，正确的是（ C ）。

- A、总是从第一个定义的函数开始执行
- B、要调用的函数必须在 main() 函数中定义
- C、总是从 main() 函数开始执行
- D、main() 函数必须放在程序的开始

52、在 FTP 协议中，控制连接是由 ( B ) 主动建立的。

- A、服务器端
- B、客户端
- C、操作系统
- D、服务提供商

53、以下叙述中，不符合 RISC 指令系统特点的 ( B )。

- A、指令长度固定，指令种类少
- B、寻址方式种类丰富，指令功能尽量增强
- C、设置大量通用寄存器，访问存储器指令简单
- D、选取使用频率较高的一些简单指令

54、当我们与某远程网络连接不上时，就需要跟踪路由查看，以便了解在网络的什么位置出现了问题，满足该目的的命令是 ( C )。

- A、ping
- B、ifconfig
- C、traceroute
- D、netstat

55. 下列哪种文件系统的写入是 LINUX 所不能完全支持的 : D

- A. FAT
- B. UFS
- C. JFS
- D. NTFS

56. LINUX 支持网络文件系统 NFS, 下列哪个命令实现了将位于 192.168.1.4 机器上的 /opt/sirnfs 目录挂载到本机 /mnt/sirnfs 下 : A

- A . mount -t nfs 192.168.1.4:/opt/sirnfs /mnt/sirnfs
- B . mount -t nfs /mnt/sirnfs 192.168.1.4:/opt/sirnfs
- C . mount nfs -t 192.168.1.4:/opt/sirnfs /mnt/sirnfs
- D . mount nfs -t /mnt/sirnfs 192.168.1.4:/opt/sirnfs

57、同 CISC 相比，下面哪一项不属于 RISC 处理器的特征\_ D

- A、采用固定长度的指令格式，指令规整、简单、基本寻址方式有 2 ~ 3 种。
- B、减少指令数和寻址方式，使控制部件简化，加快执行速度。
- C、数据处理指令只对寄存器进行操作，只有加载/存储指令可以访问存储器，以提高指令的执行效率，同时简化处理器的设计。
- D、RISC 处理器都采用哈佛结构

58、在下列 ARM 处理器的各种模式中，\_ D \_\_ 模式有自己独立的 R8-R14 寄存器。

- A、系统模式 (System)、
- B、终止模式 (Abort)
- C、中断模式 (IRQ)
- D、快中断模式 (FIQ)

59、按照 ARM 过程调用标准 (APCS)，栈指针使用 \_ B \_\_ 寄存器，

- A、R0
- B、R13
- C、R14
- D、R15

60、在 ARM 体系结构中，\_ C \_\_ 寄存器作为连接寄存器，当进入子程序时或者处理器响应异常的时候，用来保存 PC 的返回值；\_ C \_\_ 寄存器作为处理器的程序计数器指针。

- A、R0, R14
- B、R13, R15
- C、R14, R15
- D、R14, R0

61、在 ARM 体系结构中，要从主动用户模式 (User) 切换到超级用户模式 (Supervisor)，应采用何种方法？C

- A、直接修改 CPU 状态寄存器 (CPSR) 对应的模式
- B、先修改程序状态备份寄存器 (SPSR) 到对应的模式，再更新 CPU 状态
- C、使用软件中断指令 (SWI)
- D、让处理器执行未定义指令

62、下面关于 MMU 和 Linux 描述错误的是：C

- A、MMU是内存管理单元 Memory Management Unit 的缩写  
B、uClinux可以运行在有 MMU 的处理器上  
C、Linux 内核功能强大，内存管理功能丰富，即使在没有 MMU 的处理器上，也可以通过软件实现地址映射。  
D、Linux 系统正是利用 MMU，才能使得各个进程有独立的寻址空间

63、DNS 域名系统主要负责主机名和 ( A ) 之间的解析。

- A、IP 地址                      B、MAC 地址  
C、网络地址                      D、主机别名

64、在 vi 编辑器中的命令模式下，重复上一次对编辑的文本进行的操作，可使用 ( C ) 命令。

- A、上箭头                      B、下箭头                      C、<.>                      D、<\*>

65、进程有三种状态：( C )。

- A、准备态、执行态和退出态                      B、精确态、模糊态和随机态  
C、运行态、就绪态和等待态                      D、手工态、自动态和自由态

66、下列变量名中有效的 shell 变量名是 ( C )。

- A、-1-time                      B、\_2\$3  
C、bo\_chuang\_1                      D、2009file

67、文件系统的主要功能是 ( A )。

- A、实现对文件的按名存取                      B、实现虚拟存储  
C、提高外存的读写速度                              D、用于保存系统文档

68、在 ARM Linux 体系中，用来处理外设中断的异常模式是 C

- A、软件中断 ( SWI )                      B、未定义的指令异常  
C、中断请求 ( IRQ )                              D、快速中断请求 ( FIQ )

69、在 Linux 系统中，驱动程序注册中断处理程序的函数是 B

- A、trap\_init                      B、request\_irq

C、enable\_irq                            D、register\_irq

70、在 ARM Linux 系统中，中断处理程序进入 C 代码以后，ARM 的处于 A 工作模式

- A、超级用户 ( SVC )                    B、中断 ( IRQ )  
C、快速中断 ( IRQ )                    D、和进入中断之前的状态有关系

71、在 ARM 体系构建的嵌入式系统中，由电平模式触发的中断，其对应的中断标准应该在何时被清除？A

- A、当中断处理程序结束以后，才可以清除  
B、进入相应的中断处理程序，即可以清除  
C、产生 IRQ 中断的时候，处理器自动清除  
D、任何时候都可以清除

72、在操作系统中，Spooling 技术是用一类物理设备模拟另一类物理设备的技术，实现这种技术的功能模块称做 (        B        )。

- A、可林斯系统                            B、斯普林系统  
C、图灵机系统                            D、虚拟存储系统

73、通过修改下面文件哪个文件，可以设定开机时候自动安装的文件系统 ( C )

- A. /etc/mta                              B. /etc/fastboot  
C. /etc/fstab                            D. /etc/inetd.conf

74、下面关于 Shell 的说法，不正确的是： ( D )

- A. 操作系统的外壳  
B. 用户与 Linux 内核之间的接口程序  
C. 一个命令语言解释器  
D. 一种和 C 类似的程序语言

75、init 可执行文件通常存放在 ( C ) 目录中。

- A . /etc                                    B . /boot  
C . /sbin                                    D . /root

76、假设 root 用户执行“init 0”命令，系统将会 ( B )。

- A . 暂停      B . 关机      C . 重新启动      D . 初始化

77、嵌入式系统应用软件一般在宿主机上开发，在目标机上运行，因此需要一个 ( B ) 环境。

- A、交互操作系统      B、交叉编译  
C、交互平台      D、分布式计算

78、已知有变量 data1 定义如下：C

```
union data
{ int i;
  char ch;
  float f;
} data1;
```

则变量 data1 所占的内存存储空间可表示为。

- A、sizeof(int)      B、sizeof(char)  
C、sizeof(float)      D、sizeof(int)+sizeof(char)+sizeof(float)

79、软件开发模型给出了软件开发活动各阶段之间的关系，( D ) 不是软件开发模型。

- A、瀑布模型      B、螺旋模型  
C、原型模型      D、程序模型

80、实时操作系统 ( RTOS ) 内核与应用程序之间的接口称为 ( C )。

- A、输入/输出接口      B、文件系统  
C、API      D、图形用户接口

81、在操作系统中，除赋初值外，对信号量仅能操作的两种原语是 ( C )。

- A、存操作、取操作      B、读操作、写操作  
C、P 操作、V 操作      D、输入操作、输出操作

82、在下列 ARM 处理器的各种模式中，只有 A 模式不可以自由地改变处理器的工作模式。

- A、用户模式 ( User )      B、系统模式 ( System )

C、终止模式(Abort)      D、中断模式(IRQ)

83、32位体系结构的ARM处理器有\_B\_种不同的处理器工作模式，和\_B\_个主要用来标识CPU的工作状态和程序的运行状态的状态寄存器。

- A、7、7      B、7、6      C、6、6      D、6、7

84、已知Linux系统中的唯一一块硬盘是第一个IDE接口的master设备，该硬盘按顺序有3个主分区和一个扩展分区，这个扩展分区又划分了3个逻辑分区，则该硬盘上的第二个逻辑分区在Linux中的设备名称是(D)

- A. /dev/hda2      B. /dev/hda3  
C. /dev/hda5      D. /dev/hda6

85、为了查看Linux启动信息，可以用：(B)

- A. cat /etc/lilo.conf      B. dmesg      C. cat/proc/cpuinfo      D. lilo

86、某文件的组外成员的权限为只写；所有者有读写权限；组内的权限为只读，则该文件的权限为(B)

- A 467      B 642      C 476      D 764

87、下面哪个命令行可用来马上重新启动正在运行的Linux系统？(D)

- A. restart --delay=0      B. reboot -w  
C. halt -p      D. shutdown -r now

88、在bash命令中，当用(B)参数时，表示bash是交互的。

- A、-c      B、-i      C、-s      D、-d

89、重定向的符号">>"表示：(A)

- A、输出追加      B、输入追加      C、输出重定向，原来的文件被改写      D、管道

90、Linux文件权限一共10位长度，分成四段，第一段表示的内容是(A)

- A 文件类型      B 文件所有者的权限  
C 文件所有者所在组的权限      D 其他用户的权限

91、(B)命令可更改一个文件的权限设置？

- A. attrib      B. chmod      C. change      D. file

92、你用 vi 编辑器编写了一个脚本文件 shell.sh ,你想将该文件名称修改为 shell2.sh ,下列命令( B )可以实现。

- A. cp shell.sh shell2.sh
- B. mv shell.sh shell2.sh
- C. ls shell.sh >shell2.sh
- D. ll shell.sh >shell2.sh

93、在使用 GCC 编译器的过程中 ,以下 ( B ) 选项可用来指定生成的目标文件名

- A . -c
- B . -o
- C . -S
- D . -E

94、假设当前目录下有文件 Makefile ,下面是其内容 :

```
pr1: prog.o subr.o
gcc -o pr1 prog.o subr.o
prog.o: prog.c prog.h
gcc -c -l prog.o prog.c
subr.o: subr.c
gcc -c -o subr.o subr.c
clear:
rm -f pr1*.o
```

现在执行命令 make subr.o ,实际执行的命令是 ( C ):

- A. gcc -o pr1 prog.o subr.o
- B. gcc -c -l prog.o prog.c
- C. gcc -c -o subr.o subr.c
- D. 都执行

95、为了使用生成的目标文件能够用于 gdb 调试 ,在编译时 GCC 应使用 ( C ) 选项。

- A . -c
- B . -w
- C . -g
- D . -o

96、存盘并退出 vi 的指令是 ( D )。

- A、q
- B、q!
- C、w
- D、wq

97. 下列关于 /etc/fstab 文件描述 ,正确的是 ( D )。

- A. fstab 文件只能描述属于 linux 的文件系统

B. CD\_ROM 和软盘必须是自动加载的

C. fstab 文件中描述的文件系统不能被卸载

D 启动时按 fstab 文件描述内容加载文件系统

98. ARM 嵌入式系统中，PC 指向的是正在 ( C ) 的指令地址。

A 执行      B 译码      C 取指      D 都不是

99. ARM 系统处理 16-bit 数据时，对应的数据类型是 ( B )。

A Byte      B Halfword      C Word      D 三者都不是

100. 实时系统是指 ( B )

A 响应快的系统      B 时间约束的系统      C 单任务系统      D 内核小的系统

101. 下面属于 blob 运行过程第一阶段的是 ( C )

- A 外围的硬件初始化 ( 串口 , USB 等 );
- B 根据用户选择，进入命令行模块或启动 kernel。
- C 寄存器的初始化
- D 堆栈的初始化

答案：C 第一阶段的代码在 start.s 中定义，大小为 1KB，它包括从系统上电后在 0x00000000 地址开始执行的部分。这部分代码运行在 Flash 中，它包括对 S3C44B0 的一些寄存器的初始化和将 Blob 第二阶段代码从 Flash 拷贝到 SDRAM 中。

102. 下列几种流行的嵌入式 GUI 中，没有采用分层设计的一种是： B

A. MiniGUI      B. Qt/Embedded      C. Nano-X Window      D. OpenGUI

103. Qt/Embedded 的底层图形引擎基于一下哪种接口技术： A

A. framebuffer      B. GAL      C. IAL      D. GFX

104. 在 Linux 使用 GCC 编译器时有如下命令： Gcc-g test.c -o test，其中参数-g 的作用是 (D)

A .生成目标文件 test.o      B. 生成汇编文件 test.s      C .进行预编译      D .包含调试信息

105. LINUX 支持网络文件系统 NFS，下列哪个命令实现了将位于 192.168.1.4 机器上的 /opt/sirnfs

目录挂载到本机 /mnt/sirnfs 下：A

- A .mount -t nfs 192.168.1.4:/opt/sirnfs /mnt/sirnfs
- B .mount -t nfs /mnt/sirnfs 192.168.1.4:/opt/sirnfs
- C mount nfs -t 192.168.1.4:/opt/sirnfs /mnt/sirnfs
- D mount nfs -t /mnt/sirnfs 192.168.1.4:/opt/sirnfs

106、同 CISC 相比，下面哪一项不属于 RISC 处理器的特征 D

- A、采用固定长度的指令格式，指令规整、简单、基本寻址方式有 2 ~ 3 种。
- B、减少指令数和寻址方式，使控制部件简化，加快执行速度。
- C、数据处理指令只对寄存器进行操作，只有加载/存储指令可以访问存储器，以提高指令的执行效率，同时简化处理器的设计。
- D、RISC 处理器都采用哈佛结构

107、32 位数 0x12345678 用小端格式表示，则在 AXD 调试器下观察数据在内存中分布的情况是 ( B )

- A 12 34 56 78      B 78 56 34 12      C 21 43 65 87      D 87 65 43 21

108、RISC 是指 ( C )

- A 复杂指令计算机      B 并行机      C 精简指令计算机      D 多处理器计算机

109、在 ARM 体系结构中，C 寄存器作为连接寄存器，当进入子程序时或者处理器响应异常的时候，用来保存 PC 的返回值；C 寄存器作为处理器的程序计数器指针。

- A、R0 , R14      B、R13 , R15
- C、R14 , R15      D、R14 , R0

110、在 ARM 体系结构中，要从主动用户模式 ( User ) 切换到超级用户模式 ( Supervisor )，应采用何种方法？C

- A、直接修改 CPU 状态寄存器 ( CPSR ) 对应的模式
- B、先修改程序状态备份寄存器 ( SPSR ) 到对应的模式，再更新 CPU 状态

C、使用软件中断指令 ( SWI )

D、让处理器执行未定义指令

111、表达式  $A \oplus B$  实现的功能是 ( C )

A 逻辑与      B 逻辑非      C 逻辑异或      D 逻辑或

112、嵌入式系统的开发通常是在交叉开发环境实现的，交叉开发环境是指 ( A )

A 在宿主机上开发，在目标机上运行      B 在目标机上开发，在宿主机上运行

C 在宿主机上开发，在宿主机上运行      D 在目标机上开发，在目标机上运行

113、在 ARM 系统结构中，MMU 映射最小的单元空间是\_\_D\_\_

A、64KB      B、16KB      C、4KB      D、1KB

114、在 ARM Linux 启动的过程中，开启 MMU 的时候，如何实现从实地址空间到虚拟地址空间的过度？D

A、开启 MMU，在内存中创建页表（映射内核到 3G 以上的虚拟地址空间）并继续运行。

B、开启 MMU，在内存中创建页表（映射内核到 3G 以上的虚拟地址空间），跳转到虚拟地址空间继续运行。

C、在内存中创建页表（映射内核到 3G 以上的虚拟地址空间），开启 MMU，跳转到虚拟地址空间继续运行。

D、在内存中创建页表（映射内核到 3G 以上的虚拟地址空间，同时把内核所在的前 1MB 空间到和其实地址相同的虚拟地址空间），开启 MMU，跳转到虚拟地址空间继续运行。

115、在 ARM 体系中，MMU 的第一级描述符有\_\_项，每个描述符占用\_\_字节

A、1024，32      B、4096，4

C、4096，4      D、1024，32

答案：C ( B 和 C 一样的，A 和 D 是一样的 )

116、在 ARM 体系中，下面 MMU 的一级描述符中，是页描述符的是\_\_A\_\_

A、0xA0000C0E      B、0xA0000C0F

C、0x00000000      D、0xC0000C01

117、在 ARM Linux 体系中，用来处理外设中断的异常模式是\_\_C\_\_

- A、软件中断 ( SWI )      B、未定义的指令异常  
C、中断请求 ( IRQ )      D、快速中断请求 ( FIQ )

118 、指令 ADD R2, R1, R1, LSR #2 中 , LSR 的含义是 ( B )。

- A 逻辑左移      B 逻辑右移      C 算术右移      D 循环右移

119、以下 ARM 异常中 , 优先级最高的是 ( D )。

- A Data abort      B FIQ      C IRQ      D Reset

120、指令 LDR R0, [R4] 对源操作数的寻址方式是 ( A )

- A 寄存器间接寻址      B 寄存器寻址      C 立即数寻址      D 相对寻址

121、在 Linux 2.4 或者 2.6 内核中 , 和 ARM 体系结构相关的中断处理程序的 C 代码在源码树的   B   文件中

- A、kernel/irq.c  
B、arch/arm/kernel/irq.c  
C、arch/arm/mach/irq.c  
D、arch/arm/kernel/entry-armv.S

122、以下关于 init 进程 , 描述不正确的是 : ( A )

- A. 一个通用进程  
B. 可以产生新的进程  
C. 在某些程序退出的时候能重起它们  
D. 负责在系统启动的时候运行一系列程序和脚本文件

123、哈佛结构和冯诺依曼结构的区别是 ( A )

- A 指令和数据分开存储      B 不需要程序计数器      C 统一编址      D 单一数据总线

124、 fstab 文件存放在 ( A ) 目录中。

- A . /etc      B . /boot  
C . /sbin      D . /root

125、Linux 系统运行级别 5 工作在 ( D ) 状态。

- A . 单用户字符模式
- B . 多用户字符模式
- C . 单用户图形模式
- D . 多用户图形模式

126、下面关于 Shell 的说法，不正确的是： ( D )

- A. 操作系统的外壳
- B. 用户与 Linux 内核之间的接口程序
- C. 一个命令语言解释器
- D. 一种和 C 类似的程序语言

127、init 启动进程需要读取 ( A ) 配置文件：

- A. /etc/inittab
- B. /sbin/init
- C. /etc/sysvinit
- D. /bin/sh

128、启动 init 进程前，不需要经过 ( D ) 步骤。

- A . 加载内核
- B . 检测内存
- C . 加载文件系统
- D . 启动网络支持

129、RISC 是指 ( C )

- A 复杂指令计算机
- B 并行机
- C 精简指令计算机
- D 多处理器计算机

130、波特率 9600bps 是指数据每秒传输 ( B )

- A 9600 个字节
- B 9600 个比特
- C 9600 个字
- D 9600 个字符

131、ARM9 和 ARM7 的重要区别是 ( A )

- A ARM9 带有 MMU 功能
- B ARM9 支持 Thumb 指令集
- C ARM9 带有 Cache 功能
- D ARM9 是哈佛结构

132、32 位体系结构的 ARM 处理器有  B  种不同的处理器工作模式，和  B  个主要用来标识 CPU 的工作状态和程序的运行状态的状态寄存器。

- A、7、7      B、7、6  
C、6、6      D、6、7

133、在安装 Linux 的过程中的第五步是让用户选择安装方式，如果用户希望安装部分组件（软件程序），并在选择好后让系统自动安装，应该选择的选项是 D。

- A) full      B) expert      C) newbie      D) menu

134、当系统工作负载增加时，CPU 的 A 将占很大比重

- A) 用户时间      B) 系统时间      C) 空闲时间、      D) 进程时间

135、fsck 对文件系统的检查最先是从文件系统的 C 开始的

- A) MBR      B) 磁盘块      C) 超级块      D) 块链表

如果数据的存储格式是大端模式，32bit 宽的数 0x12345678 在大端模式下的 CPU 内存中的存放（假设从地址 0x4000 开始）。内存地址为 0x4001 的内容是（A）。

- A、0x34      B、0x56  
C、0x23      D、0x78

136、关于 RISC 指令系统描述不正确的是（A）。

- A、指令条数多      B、指令长度固定  
C、指令格式种类少      D、寻址方式种类少

137、对 ARM7 微处理器说法不正确的是（D）。

- A、兼容 16 位的 Thumb 指令集      B、集成式 RISC 内核  
C、集成了 ICE-RT 逻辑      D、哈佛体系结构

138、在寄存器间接寻址方式中，指定寄存器中存放的是（B）。

- A、操作数      B、操作数地址  
C、转移地址      D、地址偏移量

139、Samba 服务器的进程由 B 两部分组成。

- A) named 和 sendmail      B) smbd 和 nmbd      C) bootp 和 dhcpcd      D) httpd 和 squid

140、为保证在启动服务器时自动启动 DHCP 进程，应对 B 文件进行编辑。

- A) /etc/rc.d/rc.inet2
- B) /etc/rc.d/rc.inet1
- C) /etc/dhcpd.conf
- D) /etc/rc.d/rc.S

141、在配置代理服务器时，若设置代理服务器的工作缓存为 64MB，配置行应为 D 。

- A) cache 64MB
- B) cache\_dir ufs /usr/local/squid/cache 10000 16 256
- C) cache\_mngr 64MB
- D) cache\_mem 64MB

142、安全管理涉及的问题包括保证网络管理工作可靠进行的安全问题和保护网络用户及网络管理对象问题。

C 属于安全管理的内容。

- A) 配置设备的工作参数
- B) 收集与网络性能有关的数据
- C) 控制和维护访问权限
- D) 监测故障

143、B 命令是在 vi 编辑器中执行存盘退出。

- A) q
- B) wq
- C) q!
- D) WQ

144、下列关于/etc/fstab 文件描述，正确的是 D 。

- A) fstab 文件只能描述属于 linux 的文件系统
- B) CD\_ROM 和软盘必须是自动加载的
- C) fstab 文件中描述的文件系统不能被卸载
- D) 启动时按 fstab 文件描述内容加载文件系统

145、D 设备是字符设备。

- A) hdc
- B) fd0
- C) hda1
- D) tty1

146、已知有如下程序：

```
#include <stdio.h>
void main() {
    int a[5]={1,2,3,4,5};
    int * p = (int *)(&a+1);
    printf("%d",p[-1]);
}
```

那么，输出结果为 ( B )

- A、该程序不可执行，无输出结果      B、5  
C、1      D、不确定的随机值

147、终止一个前台进程可能用到的命令和操作是 (B)。

- A、kill      B、`<ctrl>+c`  
C、shut down      D、halt

148、B 目录存放着 Linux 的源代码。

- A) /etc      B) /usr/src      C) /usr      D) /home

149、关于文件系统的安装和卸载，下面描述正确的是 A。

- A) 如果光盘未经卸载，光驱是打不开的      B) 安装文件系统的安装点只能是/mnt 下  
C) 不管光驱中是否有光盘，系统都可以安装 CD-ROM 设备  
D) `mount /dev/fd0 /floppy` 此命令中目录/floppy 是自动生成的

150、文件 exer1 的访问权限为 `rw-r--r--`，现要增加所有用户的执行权限和同组用户的写权限，下列命令正确的是 A。

- A) chmod a+x g+w exer1      B) chmod 765 exer1  
C) chmod o+x exer1      D) chmod g+w exer1

151、有关归档和压缩命令，下面描述正确的是 C。

- A) 用 `uncompress` 命令解压缩由 `compress` 命令生成的后缀为.zip 的压缩文件  
B) `unzip` 命令和 `gzip` 命令可以解压缩相同类型的文件  
C) `tar` 归档且压缩的文件可以由 `gzip` 命令解压缩  
D) `tar` 命令归档后的文件也是一种压缩文件

152、不是 shell 具有的功能和特点的是 C。

- A) 管道      B) 输入输出重定向      C) 执行后台进程      D) 处理程序命令

153、(D) 设备是字符设备。

- A、hdc      B、fd0

C、hdal D、tty1。

154、具有很多 C 语言的功能，又称过滤器的是 C 。

- A) csh                    B) tcsh                    C) awk                    D) sed

155、局域网的网络地址 192.168.1.0/24，局域网络连接其它网络的网关地址是 192.168.1.1。主机 192.168.1.20 访问 172.16.1.0/24 网络时，其路由设置正确的是 B。

- A) route add -net 192.168.1.0 gw 192.168.1.1 netmask 255.255.255.0 metric 1
  - B) route add -net 172.16.1.0 gw 192.168.1.1 netmask 255.255.255.255 metric 1
  - C) route add -net 172.16.1.0 gw 172.16.1.1 netmask 255.255.255.0 metric 1
  - D) route add default 192.168.1.0 netmask 172.168.1.1 metric 1

156、不需要编译内核的情况是 D。

- A) 删除系统不用的设备驱动程序时      B) 升级内核时      C) 添加新硬件时      D) 将网卡激活

157、内核不包括的子系统是 D。

- A) 进程管理系统      B) 内存管理系统      C) I/O 管理系统      D) 硬件管理系统

158、以下叙述中，不符合 RISC 指令系统特点的是 B。

- A) 指令长度固定，指令种类少
  - B) 寻址方式种类丰富，指令功能尽量增强
  - C) 设置大量通用寄存器，访问存储器指令简单
  - D) 选取使用频率较高的一些简单指令

159、系统中有用户 user1 和 user2 ,同属于 users 组。在 user1 用户目录下有一文件 file1 ,它拥有 644 的权限，如果 user2 用户想修改 user1 用户目录下的 file1 文件，应拥有 B 权限。

- A) 744                      B) 664                      C) 646                      D) 746

160、下列对 shell 变量 FRUIT 操作，正确的是： C。

- A) 为变量赋值 : \$FRUIT=apple
  - B) 显示变量的值 : fruit=apple
  - C) 显示变量的值 : echo \$FRUIT
  - D) 判断变量是否有值 : [ -f "\$FRUIT" ]

161、一般可以用 C 实现自动编译。

- A) gcc      B) gdb \*      C) make      D) vi

162、通常所说的 32 位微处理器是指 C。

- A) 地址总线的宽度为 32 位      B) 处理的数据长度只能为 32 位  
C) CPU 字长为 32 位      D) 通用寄存器数目为 32 个

163、在 32 位处理器上，假设栈顶指针寄存器的当前值为 0x00FFFFE8，那么在执行完指令“push eax”（eax 为 32 位寄存器）后，栈指针的当前值应为 A

- A) 0x00FFFFE4      B) 0x00FFFFE6      C) 0x00FFFFEA      D) 0x00FFFFEC

164、有若干并发进程均将一个共享变量 count 中的值加 1 一次，那么有关 count 中的值说法正确的是：  
\_\_\_\_\_。（C）

- A、肯定有不正确的结果  
B、肯定有正确的结果  
C、若控制这些并发进程互斥执行 count 加 1 操作，count 中的值正确  
D、A, B, C 均不对

165、使用 vim 作为文本编辑器，在指令模式下要将光标移动到文档的最后一行的命令是 (C)。

- A、0      B、\$  
C、G      D、GG

166、已知某用户 stud1，其用户目录为 /home/stud1。如果当前目录为 /home，进入目录 /home/stud1/test 的命令是 ( )。

- A、cd test      B、cd /stud1/test  
C、cd stud1/test      D、cd home

167、如果想配置一台匿名 ftp 服务器，应修改 c 文件。

- A) /etc/gateway      B) /etc/ftpservers  
C) /etc/ftpusers      D) /etc/inetd.conf

168、要配置 NFS 服务器，在服务器端主要配置 c 文件。

- A) /etc/rc.d/rc.inet1      B) /etc/rc.d/rc.M  
C) /etc/exports      D) /etc/rc.d/rc.S

169、Linux 将存储设备和输入/输出设备均看做文件来操作，c 不是以文件的形式出现。

- A) 目录      B) 软链接      C) i 节点表      D) 网络适配器

170. Linux 文件权限一共 10 位长度，分成四段，第三段表示的内容是 C。

- A) 文件类型      B) 文件所有者的权限  
C) 文件所有者所在组的权限      D) 其他用户的权限

171. 一个文件名字为 rr.Z，可以用来解压缩的命令是： D。

- A) tar      B) gzip      C) compress      D) uncompress

172. 在使用 ln 建立链接时，为了跨越不同的文件系统，需要使用 (B)。

- A. 普通链接      B. 硬链接  
C. 特殊链接      D. 软链接

173. Samba 服务器的进程由 (B) 两部分组成。

- A、named 和 sendmail      B、smbd 和 nmbd  
C、bootp 和 dhcpcd      D、httpd 和 squid

174. PV 操作是在 (D) 上的操作。

- A、临界区      B、进程  
C、缓冲区      D、信号量

175. 在 TCP/IP 模型中，应用层包含了所有的高层协议，在下列的一些应用协议中， B 是能够实现本地与远程主机之间的文件传输工作。

- A) telnet      B) FTP      C) SNMP      D) NFS

176. 当我们与某远程网络连接不上时，就需要跟踪路由查看，以便了解在网络的什么位置出现了问题，满足该目的的命令是 C。

- A) ping      B) ifconfig      C) traceroute      D) netstat

177. DNS 域名系统主要负责主机名和 A 之间的解析。

- A) IP 地址      B) MAC 地址      C) 网络地址      D) 主机别名

178. 关于 Qt 说法不正确的是 (C)

A、是跨平台的 C++ 图形用户界面库      B、Qt Embedded ( Qttopia ) 基于 Framebuffer

C、Qt 的各元件通信是基于 callback 的      D、可以同几种 Java 虚拟机集成

179、暂停当前（前台）任务并放到后台去的命令是（ A ）。

A、<CTRL> + Z      B、<CTRL> + C

C、<CTRL> + P      D、&

180、关于 RISC 指令系统描述不正确的是（ C ）。

A、优先选取使用频率最高的一些指令      B、避免使用复杂指令

C、不需要一个复杂的编译器      D、寻址方式种类少

181、启动 samba 服务器进程，可以有两种方式：独立启动方式和父进程启动方式，其中前者是在 C 文件中以独立进程方式启动。

A) /usr/sbin/smbd      B) /usr/sbin/nmbd      C) rc.samba      D) /etc/inetd.conf

182、进程有三种状态： C 。

A) 准备态、执行态和退出态      B) 精确态、模糊态和随机态

C) 运行态、就绪态和等待态      D) 手工态、自动态和自由态

183、Samba 服务器的配置文件是 D 。

A) httpd.conf      B) inetd.conf      C) rc.samba      D) smb.conf

184、字符设备文件类型的标志是 B 。

A) p      B) c      C) s      D) l

185、下列变量名中有效的 shell 变量名是： C 。

A) -2-time      B) \_2\$3      C) trust\_no\_1      D) 2004file

186、以下叙述中正确的是 C 。

A) 宿主机与目标机之间只需要建立逻辑连接即可

B) 在嵌入式系统中，调试器与被调试程序一般位于同一台机器上

C) 在嵌入式系统开发中，通常采用的是交叉编译器

D) 宿主机与目标机之间的通信方式只有串口和并口两种

187、文件系统的主要功能是 A。

- A) 实现对文件的按名存取
- B) 实现虚拟存储
- C) 提高外存的读写速度
- D) 用于保存系统文档

188、以下做法不利于嵌入式应用软件的移植的是 D。

- A) 在软件设计上，采用层次化设计和模块化设计
- B) 在软件体系结构上，在操作系统和应用软件之间引入一个虚拟机层，把一些通用的、共性的操作系统的 API 接口函数封装起来
- C) 将不可移植的部分局域化，集中在某几个特定的文件之中
- D) 在数据类型上，尽量直接使用 C 语言的数据类型

189、对 ARM 处理器说法不正确的是 ( D )。

- A、小体积、低功耗、低成本、高性能
- B、支持 Thumb (16 位) /ARM (32 位) 双指令集
- C、只有 Load/Store 指令可以访问存储器
- D、寻址方式多而复杂

190、嵌入式微控制器相比嵌入式微处理器的最大特点 ( B )。

- A、体积大大减小
- B、单片化
- C、功耗低
- D、成本高

191. c-shell 中变量名 ignoreeof 表示 C

- A) 执行之前显示每一条命令
- B) 使文件名结束
- C) 必须用 logout 注销而不是 ^D
- D) 禁止文件名扩展

192. 在某嵌入式操作系统中，若 P、V 操作的信号量 S 的初值为 2，当前值为 -1，则表示等待信号量 S 的任务个数为 B。

- A) 0
- B) 1
- C) 2
- D) 3

193、在字符界面环境下注销 LINUX，可用 ( C ) 命令。

- A. exit 或 quit
- B. quit 或 ctrl+D
- C. exit 或 ctrl+D
- D. 以上都可

194. 用下列 ( A ) 命令查看 Linux 使用了多少内存。

- A. cat /proc/meminfo
- B. cat /bin/meminfo
- C. vi /proc/meminfo
- D. vi /user/local/meminfo

195. LINUX 支持网络文件系统 NFS, 下列哪个命令实现了将位于 192.168.1.4 机器上的 /opt/sirnfs 目录挂载到本机/mnt/sirnfs 下： D

- A .mount nfs -t /mnt/sirnfs 192.168.1.4:/opt/sirnfs
- B .mount -t nfs /mnt/sirnfs 192.168.1.4:/opt/sirnfs
- C .mount nfs -t 192.168.1.4:/opt/sirnfs /mnt/sirnfs
- D .mount -t nfs 192.168.1.4:/opt/sirnfs /mnt/sirnfs

196、下面哪条命令可用来确保文件“file1”存在 ( B )

- A. cp file1 /dev/null
- B. touch file1
- C. create file1
- D. mkfile file1

197、在安装开始前，用光盘启动系统，想要进入字符界面安装，需要输入的命令是 ( C )

- A.linux doc
- B.linux
- C.linux text
- D.linux note

198、操作系统中同时存在着多个进程，它们 ( C )

- A、不能共享系统资源
- B、不能调用同一段程序代码

C、可以共享所有的系统资源      D、可以共享允许共享的系统资源

199、在变址寻址方式中，操作数的有效地址等于（ C ）

- A、变址寄存器内容+形式地址（位移量）
- B、程序计数器内容+形式地址
- C、基址寄存器内容+形式地址
- D、堆栈指示器内容+形式地址

200、下列文件系统中，采用了 inode 来标识文件的是（ D ）

- a.ntfs
- b.fat16
- c.fat32
- d.ext3

201、Linux 文件权限一共 10 位长度，分成四段，第一段表示的内容是（ A ）

- A 文件类型
- B 文件所有者的权限
- C 文件所有者所在组的权限
- D 其他用户的权限

202、对于所有用户具有读的文件权限，而文件主同时具有执行权限的文件权限是（ B ）

- a.655
- b.544
- c.644
- d.540

203、在使用 GCC 编译器的过程中，如果只想生成目标文件而不进行连接，需要使用选项（ C ）

- A . -S
- B . -O
- C . -C
- D . -E

204、Linux 将存储设备和输入/输出设备均看做文件来操作，下列选项（ C ）不是以文件的形式出现。

- A. 目录
- B. 软链接
- C. i 节点表
- D. 硬链接

205、Qt/Embedded 的底层图形引擎基于一下哪种接口技术： A

- A . framebuffer
- B . GAL
- C . IAL
- D . GFX

206、同 CISC 相比，下面哪一项不属于 RISC 处理器的特征 D \_\_\_\_\_

A、采用固定长度的指令格式，指令规整、简单、基本寻址方式有 2 ~ 3 种。

- B、减少指令数和寻址方式，使控制部件简化，加快执行速度。
- C、数据处理指令只对寄存器进行操作，只有加载/存储指令可以访问存储器，以提高指令的执行效率，同时简化处理器的设计。
- D、RISC处理器都采用哈佛结构

207、在给定文件中查找与设定条件相符字符串命令为（A）

- A、grep                    B、gzip
- C、find                    D、sort

208、Linux系统中的设备可分为三类：字符设备、块设备和网络设备，其中不是基于文件系统访问的设备是（C）。

- A. 字符设备              B. 块设备
- C. 网络设备              D. 字符和块设备

209、中断向量是指（C）。

- A、中断断点的地址        B、中断向量表起始地址
- C、中断处理程序入口地址    D、中断返回地址

210、（B）不是进程和程序的区别。

- A. 程序是一组有序的静态指令，进程是一次程序的执行过程
- B. 程序只能在前台运行，而进程可以在前台或后台运行
- C. 程序可以长期保存，进程是暂时的
- D. 程序没有状态，而进程是有状态的

211、在ARM系统结构中，MMU映射最大的单元空间是\_\_\_\_A\_\_\_\_

- A、1MB                    B、128KB                    C、64KB                    D、4KB

212、下面哪一个选项不是linux系统的进程类型（C）

- A. 交互进程

B. 批处理进程

C. 就绪进程

D. 守护进程

213. 如果 Boot Loader、内核、启动参数以及其他系统映像四部分在固态存储设备上分别独立存放，则其存储结构的分配顺序应当是：D\_\_\_\_\_。

A . 文件系统、内核、启动参数、Bootloader

B . 启动参数、Bootloader、内核、文件系统

C . Bootloader、内核、启动参数、文件系统

D . Bootloader、启动参数、内核、文件系统

214. Boot Loader 的 stage2 通常使用 C 语言实现，以完成复杂的功能，并增加可读性和可移植性，以下哪一步骤属于 stage2 的内容：D\_\_\_\_\_

A . 为加载 Boot Loader 的 stage2 准备 RAM 空间

B . 设置好堆栈

C . 硬件设备初始化

D . 将 kernel 映像和根文件系统映像从 flash 上读到 RAM 空间中

215、执行以下程序段

```
MOV SP, #3AH
MOV A, #20H
MOV B, #30H
PUSH ACC
PUSH B
POP ACC
POP B
```

后，A 和 B 的内容是 ( B )

A、20H , 30H

B、30H , 20H

C、3AH , 30H

D、3AH , 3AH

216、请选择正确的命令 ( B ) , 完成加载 NFS Server "svr.server.net"的 /home/nfs 到 /home2。

- A、 mount -t nfs svr.server.net:/home/nfs /home2
- B.、 mount -t -s nfs svr.server.net /home/nfs /home2
- C.、 nfsmount svr.server.net:/home/nfs /home2
- D、 nfsmount -s svr.server.net /home/nfs /home2

217、( D )设备是字符设备。

- A、 hdc
- B、 fd0
- C、 hda1
- D、 tty1

218、下面 ( D ) 命令可以列出当前动态加载的模块清单 , 会把当前插入的所有内核模块都列出来。

- A、 insmod
- B、 rmmod
- C、 dmesg
- D、 lsmod

219、在 Linux 2.4 或者 2.6 内核中 , 和 ARM 体系结构相关的中断处理程序的 C 代码在源码树的  B  文件中

- A、 kernel/irq.c
- B、 arch/arm/kernel/irq.c
- C、 arch/arm/mach/irq.c
- D、 arch/arm/kernel/entry-armv.S

220、通过修改下面文件哪个文件 , 可以设定开机时候自动安装的文件系统 ( C )

- A. /etc/mta
- B. /etc/fastboot
- C. /etc/fstab
- D. /etc/inetd.conf

221、下面关于 Shell 的说法 , 不正确的是 : ( D )

- A. 操作系统的外壳
- B. 用户与 Linux 内核之间的接口程序
- C. 一个命令语言解释器

D. 一种和 C 类似的程序语言

222、下面关于 Shell 的说法，不正确的是：( D )

- A. 操作系统的外壳
- B. 用户与 Linux 内核之间的接口程序
- C. 一个命令语言解释器
- D. 一种和 C 类似的程序语言

223、下面对于 Bootloader 的描述不正确的是 ( C )

- A、是上电后运行的第一个程序
- B、改变系统时钟
- C、Bootloader 的两种模式对开发人员没有意义
- D、向内核传递启动参数

224、符号“|”在 shell 命令中表示：( D )

- A、输出追加
- B、输入追加
- C、输出重定向，原来的文件被改写
- D、管道

225、某文件的组外成员的权限为只读；所有者有读执行权限；组内的权限为只写，则该文件的权限为 ( D )

- A 467
- B 642
- C 476
- D 524

226、在 ARM Linux 体系中，用来处理外设中断的异常模式是\_C\_\_\_\_\_

- A、软件中断 ( SWI )
- B、未定义的指令异常
- C、中断请求 ( IRQ )
- D、快速中断请求 ( FIQ )

227、在 Linux 系统中，驱动程序注册中断处理程序的函数是\_B\_\_\_\_\_

- A、trap\_init
- B、request\_irq
- C、enable\_irq
- D、register\_irq

228、未定义指令异常的 C 处理函数在 ( C ) 文件中定义。

- A、arch/arm/kernel/traps.c
- B、arch/arm/mm/fault.c
- C、arch/arm/mm/irq.c
- D、arch/arm/calls.S

229、在 ARM 体系构建的嵌入式系统中，由电平模式触发的中断，其对应的中断标准应该在何时被清除？A

- A、当中断处理程序结束以后，才可以清除

- B、进入相应的中断处理程序，即可以清除
- C、产生 IRQ 中断的时候，处理器自动清除
- D、任何时候都可以清除

230、仅当前一个命令执行出错时才执行后一条命令，需要采取的操作是：( C )

- A. command1 && command2
- B. command1 XOR command2
- C. command1 || command2
- D. command1 << command2

231、如果要将文件名 file1 修改为 file2，下列命令 ( B ) 可以实现。

- A. cp file1 file2
- B. mv file1 file2
- C. ls file1 >file2
- D. ll file1 >file2

232、在使用 GCC 编译器的过程中，以下 ( B ) 选项可用来指定生成的目标文件名

- A . -c
- B . -o
- C . -S
- D . -E

233、为了使用生成的目标文件能够用于 gdb 调试，在编译时 GCC 应使用 ( C ) 选项。

- A . -c
- B . -w
- C . -g
- D . -o

234、不存盘退出 vi 的指令是 ( B )。

- A、q
- B、q!
- C、w
- D、wq

235. 下列关于 /etc/fstab 文件描述，正确的是 ( D )。

- A. fstab 文件只能描述属于 linux 的文件系统
- B. CD\_ROM 和软盘必须是自动加载的
- C. fstab 文件中描述的文件系统不能被卸载
- D 启动时按 fstab 文件描述内容加载文件系统

236. 下列哪个命令以文本菜单方式界面配置内核选项： A

A Make menuconfig    B make xconfig    C make config    D make mrproper

237. 如果 Boot Loader、内核、启动参数以及其他系统的映像四部分在固态存储设备上分别独立存放，则其存储结构的分配顺序应当是：\_\_\_\_D\_\_\_\_。

- A. 文件系统、内核、启动参数、Bootloader
- B. 启动参数、Bootloader、内核、文件系统
- C. Bootloader、内核、启动参数、文件系统
- D. Bootloader、启动参数、内核、文件系统

238. Boot Loader 的 stage2 通常使用 C 语言实现，以完成复杂的功能，并增加可读性和可移植性，以下哪一步骤属于 stage2 的内容：\_\_\_\_D\_\_\_\_

- A . 为加载 Boot Loader 的 stage2 准备 RAM 空间
- B . 设置好堆栈
- C . 硬件设备初始化
- D . 将 kernel 映像和根文件系统映像从 flash 上读到 RAM 空间中

239. 下列几种流行的嵌入式 GUI 中，没有采用分层设计的一种是： B

- A. MiniGUI
- B. Qt/Embedded
- C. Nano-X Window
- D. OpenGUI

240. 在使用文件通配符对文件名操作时 ? 号表示 ( A )

- A. 只与一个任意的字符匹配
- B. 只与一个任意的字母匹配
- C. 只与一个任意的数字匹配
- D. 匹配于任意字符的组合

241. 同 CISC 相比，下面哪一项不属于 RISC 处理器的特征 \_\_\_\_D\_\_\_\_\_

- A、采用固定长度的指令格式，指令规整、简单、基本寻址方式有 2 ~ 3 种。
- B、减少指令数和寻址方式，使控制部件简化，加快执行速度。
- C、数据处理指令只对寄存器进行操作，只有加载/存储指令可以访问存储器，以提高指令的执行效率，同时简

化处理器的设计。

D、RISC 处理器都采用哈佛结构

242、已知 Linux 系统中的唯一一块硬盘是第一个 IDE 接口的 master 设备，该硬盘按顺序有 3 个主分区和一个扩展分区，这个扩展分区又划分了 3 个逻辑分区，则该硬盘上的第二个逻辑分区在 Linux 中的设备名称是（ D ）

- A. /dev/hda2
- B. /dev/hda3
- C. /dev/hda5
- D. /dev/hda6

243、为了查看 Linux 启动信息，可以用：( B )

- A、cat /etc/lilo.conf
- B、dmesg
- C、cat/proc/cpuinfo
- D、lilo

244、在下列 ARM 处理器的各种模式中，\_\_D\_\_ 模式有自己独立的 R8-R14 寄存器。

- A、系统模式 (System)、
- B、终止模式 (Abort)
- C、中断模式 (IRQ)
- D、快中断模式 (FIQ)

245、按照 ARM 过程调用标准 (APCS)，栈指针使用 \_\_B\_\_ 寄存器，

- A、R0
- B、R13
- C、R14
- D、R15

246、在 ARM 体系结构中，\_\_C\_\_ 寄存器作为连接寄存器，当进入子程序时或者处理器响应异常的时候，用来保存 PC 的返回值；\_\_C\_\_ 寄存器作为处理器的程序计数器指针。

- A、R0 , R14
- B、R13 , R15
- C、R14 , R15
- D、R14 , R0

247、在 ARM 体系结构中，要从主动用户模式 (User) 切换到超级用户模式 (Supervisor)，应采用何种方

法 ? C

- A、直接修改 CPU 状态寄存器 ( CPSR ) 对应的模式
- B、先修改程序状态备份寄存器 ( SPSR ) 到对应的模式，再更新 CPU 状态
- C、使用软件中断指令 ( SWI )
- D、让处理器执行未定义指令

248、在 ARM 系统结构中，MMU 映射最小的单元空间是 \_\_ D \_\_

- A、64KB
- B、16KB
- C、4KB
- D、1KB

249、在 ARM Linux 启动的过程中，开启 MMU 的时候，如何实现从实地址空间到虚拟地址空间的过度 ? D

- A、开启 MMU，在内存中创建页表（映射内核到 3G 以上的虚拟地址空间）并继续运行。
- B、开启 MMU，在内存中创建页表（映射内核到 3G 以上的虚拟地址空间），跳转到虚拟地址空间继续运行。
- C、在内存中创建页表（映射内核到 3G 以上的虚拟地址空间），开启 MMU，跳转到虚拟地址空间继续运行。
- D、在内存中创建页表（映射内核到 3G 以上的虚拟地址空间，同时把内核所在的前 1MB 空间到和其实地址相同的虚拟地址空间），开启 MMU，跳转到虚拟地址空间继续运行。

250、在 Linux 2.4 或者 2.6 内核中，和 ARM 体系结构相关的中断处理程序的 C 代码在源码树的 \_\_ B \_\_ 文件中

- A、kernel/irq.c
- B、arch/arm/kernel/irq.c
- C、arch/arm/mach/irq.c
- D、arch/arm/kernel/entry-armv.S

251、启动 init 进程前，不需要经过 ( D ) 步骤。

- A . 加载内核
- B . 检测内存
- C . 加载文件系统
- D . 启动网络支持

252、能在 Linux 下，用 mkfs.jffs2 命令创建 JFFS2 文件系统（基本上是使用 JFFS2 的 Ramdisk ），关

于 `mkfs.jffs2` 下列说法错误的是：( C )

- A . -e 选项确定闪存的擦除扇区大小 ( 通常是 64 千字节 )
- B . -p 选项用来在映像的剩余空间用零填充。
- C . -f 选项用于输出文件，通常是 JFFS2 文件系统映像
- D . 一旦创建了 JFFS2 文件系统，它就被装入闪存中适当的位置 ( 引导装载程序告知内核查找文件系统的地址 ) 以便内核能挂载它。

253、切换用户登录的命令是：( B )

- A. ps
- B. su
- C. kill
- D. changeuser

254、文件之间可以建立两种链接关系：软链接和硬链接，硬链接的特点是 ( C )

- A. 等同于文件复制操作
- B. 类似于文件复制，但新的链接文件并不占用文件磁盘存储空间
- C. 删除源文件，将使其他链接文件失效
- D. 可以对目录文件名建立硬链接

255、在使用文件通配符对文件名操作时 ? 号表示 ( A )

- A. 只与一个任意的字符匹配
- B. 只与一个任意的字母匹配
- C. 只与一个任意的数字匹配
- D. 匹配于任意字符的组合

256、Linux 文件权限一共 10 位长度，分成四段，第二段表示的内容是 ( B )

- A. 文件类型
- B. 文件所有者的权限
- C. 文件所有者所在组的权限
- D. 其他用户的权限

257、对于所有用户都只能读的文件权限是 ( B )

- a. 777
- b. 444
- c. 644
- d. 640

258、在 vi 编辑器的命令模式中，删除一行的命令是 ( B )

- a. yy
- b. dd
- c. pp
- d. xx

259、在使用 GCC 编译器的过程中，如果只想生成目标文件而不进行连接，需要使用选项 ( A )

A . -C      B . -O      C . -S      D . -E

260、如果 Boot Loader、内核、启动参数以及其他系统映像四部分在固态存储设备上分别独立存放，则其存储结构的分配顺序应当是：D。

- A. 文件系统、内核、启动参数、Bootloader
- B. 启动参数、Bootloader、内核、文件系统
- C. Bootloader、内核、启动参数、文件系统
- D. Bootloader、启动参数、内核、文件系统

261、Boot Loader 的 stage2 通常使用 C 语言实现，以完成复杂的功能，并增加可读性和可移植性，以下哪一步骤属于 stage2 的内容：D

- A. 为加载 Boot Loader 的 stage2 准备 RAM 空间
- B. 设置好堆栈
- C. 硬件设备初始化
- D. 将 kernel 映像和根文件系统映像从 flash 上读到 RAM 空间中

262、下面属于 blob 运行过程第一阶段的是 ( C )

- A. 外围的硬件初始化 (串口, USB 等);
- B. 根据用户选择，进入命令行模块或启动 kernel。
- C. 寄存器的初始化
- D. 堆栈的初始化

答案：C 第一阶段的代码在 start.s 中定义，大小为 1KB，它包括从系统上电后在 0x00000000 地址开始执行的部分。这部分代码运行在 Flash 中，它包括对 S3C44B0 的一些寄存器的初始化和将 Blob 第二阶段代码从 Flash 拷贝到 SDRAM 中。

263、下列几种流行的嵌入式 GUI 中，没有采用分层设计的一种是： B

- A. MiniGUI
- B. Qt/Embedded
- C. Nano-X Window
- D. OpenGUI

264、Qt/Embedded 的底层图形引擎基于一下哪种接口技术： A

A . framebuffer    B . GAL    C . IAL    D . GFX

265、在 ARM 系统结构中，MMU 映射最大的单元空间是 A \_\_\_\_\_

A、1MB              B、128KB              C、64KB              D、4KB

266、在 ARM 系统结构中，MMU 映射最小的单元空间是 D \_\_\_\_\_

A、64KB              B、16KB              C、4KB              D、1KB

267、在 ARM Linux 启动的过程中，开启 MMU 的时候，如何实现从实地址空间到虚拟地址空间的过度？D

- A、开启 MMU，在内存中创建页表（映射内核到 3G 以上的虚拟地址空间）并继续运行。
- B、开启 MMU，在内存中创建页表（映射内核到 3G 以上的虚拟地址空间），跳转到虚拟地址空间继续运行。
- C、在内存中创建页表（映射内核到 3G 以上的虚拟地址空间），开启 MMU，跳转到虚拟地址空间继续运行。
- D、在内存中创建页表（映射内核到 3G 以上的虚拟地址空间，同时把内核所在的前 1MB 空间到和其实地址相同的虚拟地址空间），开启 MMU，跳转到虚拟地址空间继续运行。

268、在 ARM 体系中，MMU 的第一级描述符有       项，每个描述符占用       字节

A、1024 , 32              B、4096 , 4

C、4096 , 4              D、1024 , 32

答案：C (B 和 C 一样的，A 和 D 是一样的)

269、在 ARM 体系中，下面 MMU 的一级描述符中，是节描述符的是 A \_\_\_\_\_

A、0xA0000C0E              B、0xA0000C0F

C、0x00000000              D、0xC0000C01

270、在 Linux 2.4 或者 2.6 内核中，和 ARM 体系结构相关的中断处理程序的 C 代码在源码树的 B \_\_\_\_\_ 文件中

A、kernel/irq.c

B、arch/arm/kernel/irq.c

C、arch/arm/mach/irq.c

D、arch/arm/kernel/entry-armv.S

271、下面关于 Shell 的说法，不正确的是：( D )

- A. 操作系统的外壳
- B. 用户与 Linux 内核之间的接口程序
- C. 一个命令语言解释器
- D. 一种和 C 类似的程序语言

272、以下关于 init 进程，描述不正确的是：( A )

- A. 一个通用进程
- B. 可以产生新的进程
- C. 在某些程序退出的时候能重起它们
- D. 负责在系统启动的时候运行一系列程序和脚本文件

273、在 Linux 系统中，下列哪个命令可以用来加载模块 ( A )

- A.insmod
- B.load
- C.init
- D.installmod

274、可加载模块一般位于系统的 ( B ) 目录下：

- A. /lib/modules
- B. /lib/modules/x.y.z ( x.y.z 是内核的版本号 )
- C. /usr/lib
- D. /usr/local/lib

275、下列设备中 ( D ) 是字符设备。

- A hdc
- B fd0
- C hda1
- D tty1

276、下列哪个命令以文本菜单方式界面配置内核选项： A

- A Make menuconfig
- B make xconfig
- C make config
- D make mrproper

277、如果 Boot Loader、内核、启动参数以及其他的数据映像四部分在固态存储设备上分别独立存放，则其存储结构的分配顺序应当是： B \_\_\_\_\_。

- A . 文件系统、内核、启动参数、Bootloader

B . 启动参数、Bootloader、内核、文件系统

C . Bootloader、内核、启动参数、文件系统

D . Bootloader、启动参数、内核、文件系统

278、Boot Loader 的 stage2 通常使用 C 语言实现，以完成复杂的功能，并增加可读性和可移植性，以下哪一步骤属于 stage2 的内容：\_\_D\_\_

A . 为加载 Boot Loader 的 stage2 准备 RAM 空间

B . 设置好堆栈

C . 硬件设备初始化

D . 将 kernel 映像和根文件系统映像从 flash 上读到 RAM 空间中

279、Linux 分区默认的文件系统的类型是：( B )

A. vfat      B. ext2/ext3      C. swap      D. dos

280、在下列 ARM 处理器的各种模式中，\_\_D\_\_ 模式有自己独立的 R8-R14 寄存器。

A、系统模式 (System)、

B、终止模式 (Abort)

C、中断模式 (IRQ)

D、快中断模式 (FIQ)

281、按照 ARM 过程调用标准 (APCS)，栈指针使用 \_\_B\_\_ 寄存器，

A、R0      B、R13      C、R14      D、R15

282、下面关于 MMU 和 Linux 描述错误的是：C

A、MMU 是内存管理单元 Memory Management Unit 的缩写

B、uClinux 可以运行在有 MMU 的处理器上

C、Linux 内核功能强大，内存管理功能丰富，即使在没有 MMU 的处理器上，也可以通过软件实现地址映射。

D、Linux 系统正是利用 MMU，才能使得各个进程有独立的寻址空间

283、在 ARM 系统结构中，MMU 映射最大的单元空间是 \_\_A\_\_

- A、1MB      B、128KB      C、64KB      D、4KB

284、在 Linux 系统中，驱动程序注册中断处理程序的函数是 B

- A、trap\_init      B、request\_irq  
C、enable\_irq      D、register\_irq

285、在 ARM Linux 系统中，中断处理程序进入 C 代码以后，ARM 的处于 A 工作模式

- A、超级用户 (SVC)      B、中断 (IRQ)  
C、快速中断 (IRQ)      D、和进入中断之前的状态有关系

286、在 ARM 体系构建的嵌入式系统中，由电平模式触发的中断，其对应的中断标准应该在何时被清除？A

- A、当中断处理程序结束以后，才可以清除  
B、进入相应的中断处理程序，即可以清除  
C、产生 IRQ 中断的时候，处理器自动清除  
D、任何时候都可以清除

287、下面哪一个选项不是 linux 系统的进程类型 (D)

- A. 交互进程  
B. 批处理进程  
C. 守护进程  
D. 就绪进程

288、可加载模块一般位于系统的 (B) 目录下：

- A. /lib/modules  
B. /lib/modules/x.y.z (x.y.z 是内核的版本号)  
C. /usr/lib  
D. /usr/local/lib

289、在 Linux 下，用 mkfs.jffs2 命令创建 JFFS2 文件系统（基本上是使用 JFFS2 的 Ramdisk），关于 mkfs.jffs2 下列说法错误的是：(C)

- A . -e 选项确定闪存的擦除扇区大小 ( 通常是 64 千字节 )  
B . -p 选项用来在映像的剩余空间用零填充。  
C . -f 选项用于输出文件 , 通常是 JFFS2 文件系统映像  
D . 一旦创建了 JFFS2 文件系统 , 它就被装入闪存中适当的位置 ( 引导装载程序告知内核查找文件系统的地址 ) 以便内核能挂载它。

290、在下列 ARM 处理器的各种模式中 , 只有  A  模式不可以自由地改变处理器的工作模式。

- A、 用户模式 ( User )  
B、 系统模式 ( System )  
C、 终止模式 ( Abort )  
D、 中断模式 ( IRQ )

291、在 CPU 和物理内存之间进行地址转换时 ,  B  将地址从虚拟 ( 逻辑 ) 地址空间映射到物理地址空间。

- A ) TCB                    B ) MMU                    C ) CACHE                    D ) DMA

292、嵌入式系统由硬件部分和软件部分构成 , 以下不属于嵌入式系统软件的是  C  。

- A) 系统内核              B) 驱动程序              C) FPGA 编程软件              D) 嵌入式中

293、如果我们需要设置一个文件 , 使它们作为可执行文件运行时 , 该进程是作为文件所有者的权限 , 此时我们需要额外设置该文件的  C

- A ) seg-GID 位            B ) 粘滞位              C ) set-UID 位              D ) UMASK

294、NFS 服务器通过调用 /etc/rc.d/init.d 中的 portmap 和 nfs 脚本启动 , 启动后它将通过寻找本地服务器的  D  文件 , 向网络上的子机提供 NFS 文件共享服务

- A) /etc/hosts              B) /etc/inittab  
C) /etc/inet.d              D) /etc/exports

295、  B  不是进程和程序的区别。

- A ) 程序是一组有序的静态指令 , 进程是一次程序的执行过程  
B ) 程序只能在前台运行 , 而进程可以在前台或后台运行

C) 程序可以长期保存，进程是暂时的

D) 程序没有状态，而进程是有状态的

296、终止一个前台进程可能用到的命令和操作 B。

A) kill                  B) <CTRL>+C                  C) shut down                  D) halt

297、为了得到所有的命令行输入的参数，我们可以使用变量： B

A) \$#                  B) \$@                  C) \$0                  D) \$!

298、通过修改文件 C，可以设定开机时候自动安装的文件系统

A) /etc/mtab                  B) /etc/fastboot  
C) /etc/fstab                  D) /etc/inetd.conf

299、以下叙述中，不符合 RISC 指令系统特点的是 B。

A) 指令长度固定，指令种类少  
B) 寻址方式种类丰富，指令功能尽量增强  
C) 设置大量通用寄存器，访问存储器指令简单  
D) 选取使用频率较高的一些简单指令

300、下列提法中，不属于 ifconfig 命令作用范围的是 D。

A) 配置本地回环地址                  B) 配置网卡的 IP 地址  
C) 激活网络适配器                  D) 加载网卡到内核中

301、下列不是 Linux 系统进程类型的是 D。

A) 交互进程    B) 批处理进程    C) 守护进程    D) 就绪进程

302、在日常管理中，通常 CPU 会影响系统性能的情况是： A。

A) CPU 已满负荷地运转                  B) CPU 的运行效率为 30%  
C) CPU 的运行效率为 50%                  D) CPU 的运行效率为 80%

303、WWW 服务器是在 Internet 上使用最为广泛，它采用的是 B 结构。

A) 服务器/工作站                  B) B/S                  C) 集中式                  D) 分布式

304、NFS 是 C 系统。

- 
- A) 文件      B) 磁盘      C) 网络文件      D) 操作

305、关闭 linux 系统（不重新启动）可使用命令 B。

- A) Ctrl+Alt+Del      B) halt      C) shutdown -r now      D) reboot

306、在 vi 编辑器中的命令模式下，键入 B 可在光标当前所在行下添加一行。

- A) "a"      B) "o"      C) "I"      D) A

307、在 vi 编辑器中的命令模式下，重复上一次对编辑的文本进行的操作，可使用 C 命令。

- A) 上箭头      B) 下箭头      C) <. >;      D) <\*>;