

think in java interview-高级开发人员面试宝典代码示例

下载资源地址为：<http://download.csdn.net/detail/lifetragedy/6379755>

这是think in java interview中的代码示例，包括JAVA基础的数据结构， IO, 核心基础以及设计模式等。

因此我把它称为wallet工程（钱包工程），可以直接导入eclipse工程中去。

作者：lifetragedy 发表于2013/10/10 21:54:52 [原文链接](#)  
阅读：7786 评论：7 [查看评论](#)

think in java interview-高级开发人员面试宝典(十)

今天是“面经”的最后一篇，主要讲的就是一些大公司在招人时最后的那道“智力题”关卡。

## 什么是智力题？

什么样的智力题呢？就是类似于下面这种题目，出个1道到2道让面试官回答一下。

例：

有四个人要在夜里穿过一条悬索桥回到宿营地。可是他们只有一支手电，电池只够再亮17分钟。过桥必须要有手电，否则太危险。桥最多只能承受两个人 同时通过的重量。这四个人的过桥速度都不一样：一个需要1分钟，一个需要2分钟，一个需要5分钟，还有一个需要10分钟。问什么样的组合可以在最短的时间过桥？

说实话对于一般的开发公司或者是企业级开发/研发型队伍，更多的是关注在解决一个实际的工程问题上，可是为什么这些大公司，尤其是“微软”，“eBay”，“百度”会出一些这样的智力问答呢？

主要原因还在于这些公司在招收新人时不希望这个候选人仅仅是一个码农或者只是一个会写“企业业务逻辑”的“业务开发人员”。

它们对程序员的要求是这个程序员具有“搜索算法”的编写、优化能力或者是要求程序员们必须具备跳跃性的、创新思维能力的人，因此这种智力题就成了考核一个程序员是否具备这样的能力的最佳选择了。

我经常看到一些程序写了很好的候选人，智力题做了一塌糊涂，但是是一些程序不咋样的人智力题却做了很好，这样的例子彼彼皆是。其主要问题还是出在我们的大脑平时不擅于用这种“思路”去思考，于是在碰到这样的题目时一下子不习惯而导致的。

仔细分析这些智力题，创新式跳跃式的题种较少，较多的一些智力题其实考查的还是我们平时是否有严谨的、逻辑性的思考问题的习惯，因此在这篇中我会给出大量的智力题，希望通过这些智力题，使得读者们对于这种智力题有一个认识，以及养成一种平时面对类似问题该如何思考的习惯。

### 第一题 就先说上面那道“过河”的智力题吧

有四个人要在夜里穿过一条悬索桥回到宿营地。可是他们只有一支手电，电池只够再亮17分钟。过桥必须要有手电，否则太危险。桥最多只能承受两个人 同时通过的重量。这四个人的过桥速度都不一样：一个需要1分钟，一个需要2分钟，一个需要5分钟，还有一个需要10分钟。问什么样的组合可以在最短的时间过桥？

答案： 1和2一起过(2分钟);1返回(3分钟);5和10一起过(13分钟);2返回(15分钟);1和2一起过(17分钟)。全体安全过桥。

### 第二题

有一个埃及人拾到一枚标有“公元前3世纪”的金币，他问一个考古学家，考古学家说是假币，问为什么？

答案： 因为在那个时候没有公元（西元）的说法，以耶稣出生的那一年定为公元元年（西元元年），在古代，人们怎么可能过早的知道耶稣出生的时间，又怎么可能知道“公元”这一名词呢？所以，在当时，“公元前三世纪”是一个陌生的词，没有人知道它是什么意思，又怎么可能刻在金币上呢？这个金币是伪造的！

### 第三题

一只虫子在1丈深的井底，它白天爬3尺，晚上掉2尺，问它要几天才能爬出井？

答案： 一开始前7天都是白天爬3尺，晚上掉2尺，所以是7尺，第8天白天爬3尺，加上前面7天的7尺就是1丈，已经爬出来了。.....

### 第四题

假设你站在镜子前，抬起左手，抬起右手，看看镜中的自己。当你抬起左手时，镜中的自己抬起的似乎是右手。可是当你仰头时，镜中的自己也在仰头，而不是低头。为什么镜子中的影像似乎颠倒了左右，却没有颠倒上下？

答案： 上下和左右的定义不同，上下是面对称的，左右是旋转对称的（如果两只眼睛是长成一上一下就好了）

### 第五题

一个屋子有一个门(门是关闭的)和3盏电灯。屋外有3个开关，分别与这3盏灯相连。你可以随意操纵这些开关，可一旦你将门打开，就不能变换开关了。如何确定每个开关具体管哪盏灯？

答案： 开两个开关，过一段时间关一个，进去，一个灯亮，两个灯灭，灭的灯有一个是热的。

### 第六题

每个飞机只有一个油箱，飞机之间可以相互加油（注意是相互，没有加油机）一箱油可供一架飞机绕地球飞半圈。

问题： 为使至少一架飞机绕地球一圈回到起飞时的飞机场，至少需要出动几架飞机？（所有飞机从同一机场起飞，而且必须安全返回机场，不允许中途降落，中间没有飞机场）

答案：

最少5架飞机(算上绕地球一圈的那架飞机A)

DD—DD—DD—DD—

A

B

C

D

E

条件已知一箱油可供一架飞机绕地球飞半圈,就是1/2路程

方法如下:

- 1.首先三架飞机A,B,C一起飞,飞到1/8路程,C开始给A,B各加1/8路程的油.C返回花掉1/8路程的油.
2. A,B再继续飞1/8路程,B给A加1/8路程的油,B返回
3. 这时A在2/8路程时,满油,继续飞只能到6/8路程;与此同时派一飞机D从反方向接应A(绕地球一圈),D飞2/8路程,碰到A,给A加1/8路程的油,自己留1/8路程的油.
4. A,D一起飞1/8路程,派E从反方向飞1/8路程接应给A,D加油.与开始情况相似,然后一起返回.

所以加上A一共派5架

### 第七题

有3箱水果，一箱是苹果，一箱是橘子，一箱是两种水果的混装

三个箱子上都贴了标签，但所有的标签都贴错了现在你只拿出一个水果来判断3个箱字里的情况。

答案： 从贴混装箱子里面拿出一个水果，如果是苹果，这一箱就是苹果，那么，贴橘子的就是混装，贴苹果的，就是橘子  
如果从贴混装箱子里面拿出的是一个橘子，那么，这一箱就是橘子，贴橘子的就是苹果，贴苹果的就是混

### 第八题

3 ml 5 ml杯子 各一个，拼出4 ml 的容量。

答案：

3ml装满 倒入5ml，在装满 倒入5ml。现在剩下1ml。

5ml的全倒掉。 3ml里面剩下的1ml倒入5ml里。

再把3ml装满，倒入5ml。

### 第九题

烧一根不均匀的绳，从头烧到尾总共需要1个小时。现在有若干条材质相同的绳子，问如何用烧绳的方法来计时一个小时十五分钟呢？

答案：

一根一头烧，另一根从两头烧，再有一根做参照，两头烧完的记下位置（即烧到这里要半小时），把参照的那根从标记位置处剪开，取其中一段A，一头烧的那根烧完后（就是一个小时后），把A从两头开始烧，烧完后即为十五分钟，加起来共一小时十五分钟。

### 第十题

1元钱一瓶汽水，喝完后两个空瓶换一瓶汽水，问：你有20元钱，最多可以喝到几瓶汽水？

答案： 39瓶。两个空瓶=一瓶汽水（带空瓶）。一空瓶=汽水。因为瓶+水=1元。所以水=0.5元。所以20元相当于40瓶水。因为最后剩一个空瓶不给换。所以能喝39瓶。

### 第十一题

烧一根不均匀的绳要用一个小时，如何用它来判断半个小时？

答案： 两边一起烧。

### 第十二题

为什么下水道的盖子是圆的？

答案： 首先在同等用材的情况下他的面积最大。第二因为如果是方的、长方的或椭圆的，那无聊之徒拎起来它就可以直接扔进地下道啦！但圆形的盖子嘛，就可以避免这种情况了

### 第十三题

有50家人家，每家一条狗。有一天警察通知，50条狗当中有病狗，行为和正常狗不一样。每人只能通过观察别人家的狗来判断自己家的狗是否生病，而不能看自己家的狗，如果判断出自己家的狗病了，就必须当天一枪打死自己家的狗。结果，第一天没有枪声，第二天没有枪声，第三天开始一阵枪响，问：一共死了几条狗？

答案：死了3条（第几天枪响就几条）。

分析：从有一条不正常的狗开始，显然第一天将会听到一声枪响。这里的要点是你只需站在那条不正常狗的主人的角度考虑。

有两条的话思路继续，只考虑有两条不正常狗的人，其余人无需考虑。通过第一天他们了解了对方的信息。第二天杀死自己的狗。换句话说每个人需要一天的时间证明自己的狗是正常的。有三条的话，同样只考虑那三个人，其中每一个人需要两天的时间证明自己的狗是正常的狗。

第十四题

有一座山，山上有座庙，只有一条路可以从山上的庙到山脚，每周一早上8点，有一个聪明的小和尚去山下化缘，周二早上8点从山脚回山上的庙里，小和尚的上下山的速度是任意的，在每个往返中，他总是能在周一和周二的同一钟点到达山路上的同一点。例如，有一次他发现星期一的8点30和星期二的8 点30他都到了山路靠山脚的3/4的地方，问这是为什么？

答案：假定把周一和周二的活动合并为同一天，即某一天早上8点，小和尚A下山，小和尚B上山，则无论小和尚以任意速度上山和上山，小和尚A和小和尚B一定会在山路上的某一点相遇——这一点即是实际上小和尚在周一和周二的同一钟点到达的山路上的同一点。

第十五题

三个小伙子同时爱上了一个姑娘，为了决定他们谁能娶这个姑娘，他们决定用手枪进行一次决斗。小李的命中率是30％，小黄比他好些，命中率是50％，最出色的枪手是小林，他从不失误，命中率是100％。由于这个显而易见的事实，为公平起见，他们决定按这样的顺序：小李先开枪，小黄第二，小林最后。然后这样循环，直到他们只剩下一个人。那么这三个人中谁活下来的机会最大呢？他们都应该采取什么样的策略？

答案： 小林在轮到自己且小黄没死的条件下必杀黄，再跟菜鸟李单挑。

所以黄在林没死的情况下必打林，否则自己必死。

小李经过计算比较（过程略），会决定自己先打小林。

于是经计算，小李有873/2600≈33.6%的生机；

小黄有109/260≈41.9%的生机；

小林有24.5%的生机。

哦，这样，那小李的第一枪会朝天开，以后当然是打敌人，谁活着打谁；

小黄一如既往先打林，小林还是先干掉黄，冤家路窄啊！

最后李，黄，林存活率约38：27：35；

菜鸟活下来抱得美人归的几率大。

李先放一空枪（如果合伙干中林，自己最吃亏）黄会选林打一枪（如不打林，自己肯定先玩完了）林会选黄打一枪（毕竟它命中率高）

李黄对决0.3:0.280.4可能性李林对决0.3:0.60.6可能性成功率0.73

李和黄打林李黄对决0.3:0.40.7\*0.4可能性李林对决0.3:0.7\*0.6\*0.7\*0.6可能性成功率0.64

第十六题

一间囚房里关押着两个犯人。每天监狱都会为这间囚房提供一罐汤，让这两个犯人自己来分。起初，这两个人经常会发生争执，因为他们总是有人认为对方的汤比自己的多。后来他们找到了一个两全其美的办法：一个人分汤，让另一个人先选。于是争端就这么解决了。可是，现在这间囚房里又加进来一个新犯人，现在是三个人来分汤。必须寻找一个新的方法来维持他们之间的和平。该怎么办呢？

答案： 是让甲分汤，分好后由乙和丙按任意顺序给自己挑汤，剩余一碗留给甲。这样乙和丙两人的总和肯定是他们两人可拿到的最大。然后将他们两人的汤混合之后再按两人的方法再次分汤。

第十七题

有7克、2克砝码各一个，天平一只，如何只用这些物品三次将140克的盐 分成50、90克各一份？

答案： 天平一边放7+2=9克砝码，另一边放9克盐。  
天平一边放7克砝码和刚才得到的9克盐，另一边放16克盐。  
天平一边放刚才得到的16克盐和再刚才得到的9克盐，另一边放25克盐。  
这些16+9+25=50克盐，剩下的就是90克盐。

第十八题

你是路易10世的俘虏。他要给自己的城堡增加三个新地牢，让你做一个规划。干得好就释放，干不好就终生监禁。  
小地牢很难设计，要12周，但容易建成，1周即可；中地牢设计要5周，施工要6周；大地牢设计只要1周，但建造要用9周。  
你有一个建筑师和一个建造师，建筑师不会建造而建造师不会设计。  
要建好这三个地牢，你规划的工期是几周？

答案： 第一周设计大地牢。

5周用于设计中地牢。建大地牢（还剩4周）  
12周用于设计小地牢。建大地牢（4周）+建中地牢。

1周用于建小地牢。

这样：1+5+12+1=19。

第十九题

有一个长方形蛋糕，切掉了长方形的一块(大小和位置随意)，你怎样才能直直的一刀下去，将剩下的蛋糕切成大小相等的两块？

答案：将完整的蛋糕的中心与被切掉的那块蛋糕的中心连成一条线。这个方法也适用于立方体！请注意，切掉的那块蛋糕的大小和位置是随意的，不要一心想着自己切生日蛋糕的方式，要跳出这个圈子。

第二十题

你有5瓶药，每个药丸重10克，只有一瓶受到污染的药丸重量发生了变化，每个药丸重9克。给你一个天平，你怎样一次就能测出哪一瓶是受到污染的药呢？

答案：

给5个瓶子标上1、2、3、4、5。

从1号瓶中取1个药丸，2号瓶中取2个药丸，3号瓶中取3个药丸，4号瓶中取4个药丸，5号瓶中取5个药丸。

把它们全部放在天平上称一下重量。

现在用1×10+2×10+3×10+4×10+5×10的结果减去测出的重量。

结果就是装着被污染的药丸的瓶子号码。

第二十一题

某城市发生了一起汽车撞人逃跑事件，该城市只有两种颜色的车, 蓝15%绿85%，事发时有一个人在现场看见了，他指证是蓝车，但是根据专家在现场分析, 当时那种条件能看正确的可能性是80%那么, 肇事的车是蓝车的概率到底是多少？

答案：15%\*80%/(85%×20%+15%\*80%)

第二十二题

有一人有240公斤水，他想运往干旱地区赚钱。他每次最多携带60公斤，并且每前进一公里须耗水1公斤（均匀耗水）。假设水的价格在出发地为0，以后，与运输路程成正比，（即在10公里处为10元/公斤，在20公里处为20元/公斤.....），又假设他必须安全返回，请问，他最多可赚多少钱？

答案：f(x)=(60-2x)\*x,当x=15时，有最大值450。  
450×4

第二十三题

现在共有100匹马跟100块石头，马分3种，大型马；中型马跟小型马。其中一匹大马一次可以驮3块石头，中型马可以驮2块，而小型马2头可以驮一块石头。问需要多少匹大马，中型马跟小型马？（问题的关键是刚好必须是用完100匹马） 6种结果【13】1=5， 2=15， 3=215， 4=2145那么5=?

答案：因为1=5，所以5=1

第二十四题

一楼到十楼的每层电梯门口都放着一颗钻石，钻石大小不一。你乘坐电梯从一楼到十楼，每层楼电梯门都会打开一次，只能拿一次钻石，问怎样才能拿到最大的一颗？

答案：先拿下第一楼的钻石，然后在每一楼把手中的钻石与那一楼的钻石相比较，如果那一楼的钻石比手中的钻石大的话那就把手中的钻石换成那一层的钻石。

第二十五题

1， 11， 21， 1211， 111221，下一个数是什么

公务员考试也经常考到这

解题思路如下：

1

11--- 表示前一个数“1”是 1 个 1；

21--- 表示前一个数“11”是 由 2 个 1 组成；

1211--- 表示前一个数“21”是 由 1 个 2、 1 个 1 组成；

111221--- 即 11 12 21 ， 表示前一个数“1211”是依次由 1 个 1、 1 个 2， 2 个 1组成；

因此，下一次数应该是：312211

312211---即 31 22 11， 表示前一个数“11221”是依次由 3 个 1、 2 个 2， 1 个 1 组成；

第二十六题

有4个小孩看见一块石头正沿着山坡滚下来，便议论开了。

“我看这块石头有17公斤重，”第一个孩子说。

“我说它有26公斤，”第二个孩子不同意地说。

“我看它重21公斤”，第三个孩子说。

“你们都说得不对，我看它的正确重量是20公斤，”第四个孩子争着说。

他们四人争得面红耳赤，谁也不服谁。最后他们把石头拿去称了一下，结果谁也没猜准。其中一个人所猜的重量与石头的正确重量相差2公斤，另外两个人所猜的重量与石头的正确重量之差相同。当然，这里所指的差，不考虑正负号，取绝对值。请问这块石头究竟有多重？

答案： 23公斤。因为23比21多两公斤，26与20公斤都和23公斤相差3公斤。

第二十七题

“有一牧场，已知养牛27头，6天把草吃尽；养牛23头，9天把草吃尽。如果养牛21头，那么几天能把牧场上的草吃尽呢？并且牧场上的草是不断生长的。”

答案：

设牛每天吃掉x，草每天长出y，原来有牧场的草量是a  $a=(27x-y)*6=(23x-y)*9$

可解出 $y=15x,a=72x$ ,所以 $a=(21x-y)*12$ ,所以需要12天。

第二十八题

一只熊从20米深的洞掉下去用2秒钟，这熊什么颜色的？

答案：熊是白色的。

这只熊能在2秒钟内掉进20米的深洞，最主要根据物理学重力加速度原理，可以推断出这是一只在北极的北极熊，北极熊是白色的。

根据 $s=1/2\times gt^2$ ，利用时间和路程求出加速度， $g=10$ ：（只有在南北极g才会等于10，其它地方为9.8，这是物理常识）

然后跟据重力加速度判断位置，应该是南北极。

然后，看看那里会有什么熊，这要用地理生物知识，南极要是有熊的话，企鹅都没了。所以是北极的，北极熊是白色的。

第二十九题

一个经理有三个女儿，三个女儿的年龄加起来等于13，三个女儿的年龄乘起来等于经理自己的年龄，有一个下属已知道经理的年龄，但仍不能确定经理三 个女儿的年龄，这时经理说只有一个女儿的头发是黑的，然后这个下属就知道了经理三个女儿的年龄。请问三个女儿的年龄分别是多少？为什么？

答案： 三女的年龄应该是2、2、9。因为只有 一个孩子黑头发，即只有她长大了，其他两个还是幼年时期即小于3岁，头发为淡色。再结合经理的年龄应该至少大于25。

第三十题

有三个人去住旅馆，住三间房，每一间房\$10元，于是他们一共付给老板\$30， 第二天，老板觉得三间房只需要\$25元就够了于是叫小弟退回\$5给三位客人， 谁知小弟贪心,只退回每人\$1，自己偷偷拿了\$2，这样一来便等于那三位客人每人各花了九元， 于是三个人一共花了\$27，再加上小弟独吞了\$2，总共是\$29。可是当初他们三个人一共付出\$30那么还有\$1呢？

答案：偷换概念转移注意力的题型。事实上3人只付出了27元，老板得了25元，小弟拿了2元。

第三十一题

有两位盲人，他们都各自买了两对黑袜和两对白袜，八对袜了的布质、大小完全相同， 而每对袜了都有一张商标纸连着。两位盲人不小心将八对袜了混在一起。他们每人怎样才能取回黑袜和白袜各两对呢？

答案： 将每对袜子拆开一人一只。

第三十二题

一个家庭有两个小孩，其中有一个是女孩，问另一个也是女孩的概率（假定生男生女的概率一样）？

答案：1/3

因为你知 道一共有两个小孩 其中一个是女孩 而你已知的那个女孩并不知道是她第一个孩子还是第二个孩子所以它的概率是1/3如果题目换成 已知第一个是女孩 那么第二个是女孩的概率就是1/2了

总结：

基本上先写这么多吧，这些智力题覆盖了概率、统筹、排列组合、方程求解、逻辑推理以及在题目中使用偷换概念考核候选人的注意力是否集中等各种手段。

可能还有更多类似的题，不可能收集的太全，在此也只是抛砖引玉提供给读者一个解答这种类型的智力题一些思路 and 参考。

面经至此结束了！

更多的还是要靠候选人自身的不断修炼、不断积累，基础永远是最重要的，一个人不可能是神，不可能什么都会，关键在于这个人的基础，他/她的思路是很重要的，如果基础都没有，上手和人家面试官去侃云、侃nosql（我还碰到过有一个人来侃神经网络计算的结果连字符检索程序都写不出的候选人），那是沙滩上的城堡。

作者：lifetragedy 发表于2013/10/10 9:54:41 [原文链接](#)  
阅读：11143 评论：14 [查看评论](#)

[think in java interview](#)番外篇~谈程序员如何修炼英语

一、程序员对英语能力的重视度和能力要求应该是在各行各业中排在比较靠前的

这样说吧，英语程度的好坏直接影响着一个程序员的编程、开发、创新能力。

道理很简单：

1. 计算机和软件是用英语创造出来的
2. 国内的技术普及度不怎么高，而最前沿最好最全的资料也往往是英语
3. 你在读OpenSource的一些源码时，这些OpenSource也大都来源于英语



因此英语和技术对于一个程序员来说是各占50%这样的一个比重的，英语能力的好坏直接影响到一个程序员的技术能力。

说到这儿其实还是主要源于国内的技术普及度不够而导致的，网上很多东西都是千篇一律、copy & paste，要想学到国外的一些先进思路，你肯定逃不过去读别人的opensource，有些opensource如：JBPM, GUVNOR, APACHE基金会下的一些开源项目（这个是公认的最好的）都是英语的，你肯定要把这些资料看下来吧？

我们在此不谈什么4级，6级，而是把它上升到一个你吃饭的必须手段、技能这个地位来说。

## 二、如何提高英语

英语有多重要，外面说了很多，网上也有很多人说过许多道理，在此我们不会再一一复述了，我们这篇是面经， 主要的是提出问题和解决问题，及What? Why? How? 的一种模式。

有人说了，英语我从来不感兴趣，大学4级过了后基本就丢光了。

也有人说了，英语不是一朝一日练成的，是一个长期的过程我现在怎么来得及哦。

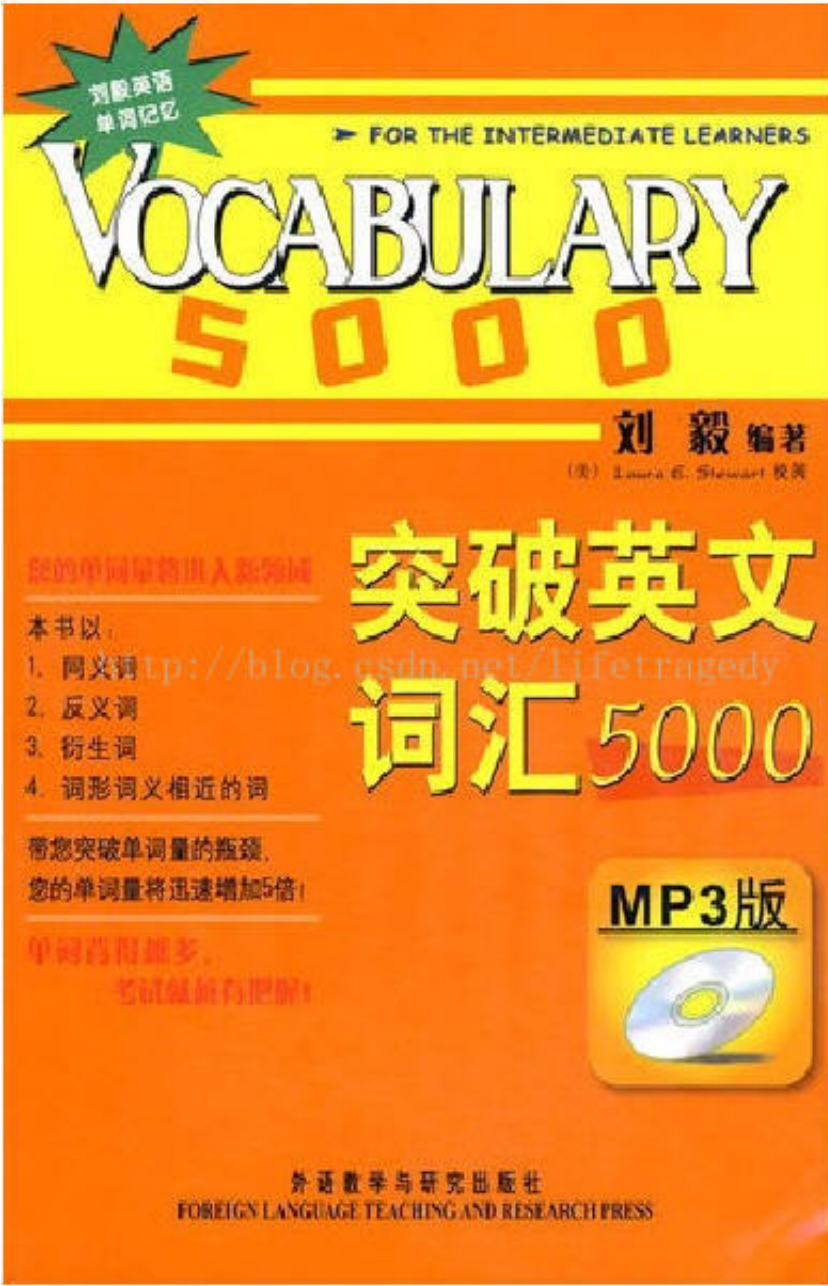
所以接下来我告诉大家的是一条我和我周边一群人近10年来总结的一条学习英语的路线，如果你按着下面这条路线去走，可以做到“娱乐学习两不误，工作中英语得到提升，英语提升后帮助你在工作事业上成长”。

下面我们来看如何把英语从我们的“敌人”变成我们的“朋友”吧。

## 三、背单词原来也可以这么奇乐无穷

先把下面这几本书看完：

### 1. 单词5000



笔者以前看的是录音带版的，现在已经有了MP3版了，真好。

为什么说这本书好呢？

英语逃不过单词，在此笔者极力不推荐大家去背字典，看语法等，而在于迅速有效的去提高自己的英语单词能力。

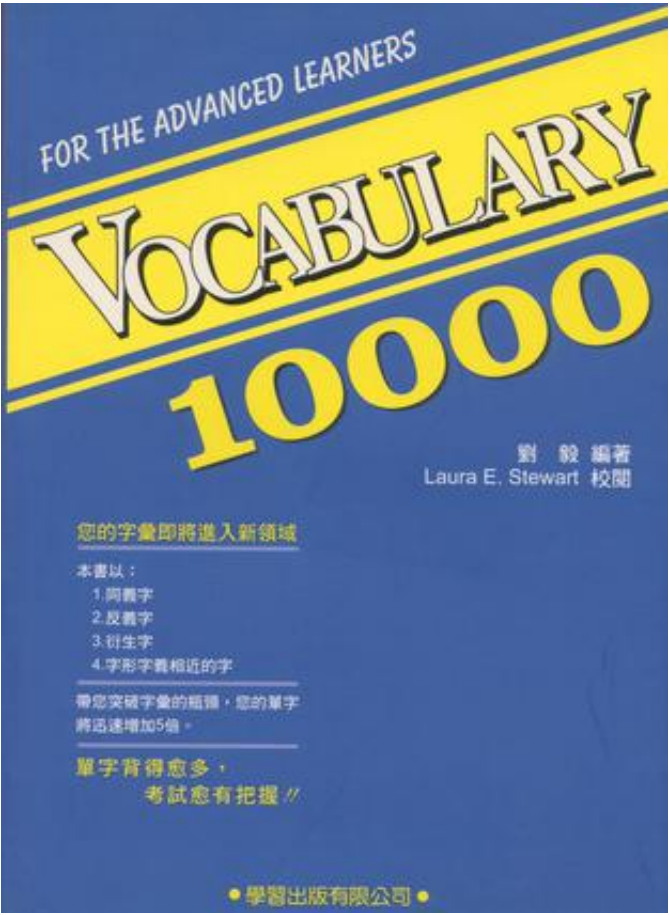
但是背单词这一关你是逃不过去的，大量的单词枯燥又乏味，不过这本Vocabulary 5000可不像字典一样，它是把一个个的生词融入到了一句句的“例句”中，除了在背这个生词外，还同时教会了你这个字到底是怎么用的。

基本推荐是一天2-3篇，然后就是听MP3（录音），上班路上听，洗衣服时听，洗澡时听，坐在马桶上也听，婴儿如何学语？他是先通过“听”。

基本上1-2个月这本书就可以干掉了。

接下来要阅读的就是：

### 2. 单词10000



这个不用多说了，和上一本一样，是上一本的升级版，唯一不同的就是这本书把一个单词列出后还给出了它的同义词，反义词，因此翻一翻，成了10000，同时有10000个例句。

这本书要解决掉的时间稍长一些，2-3个月可以完全解决掉。

这边也必须强调一个听。

就是你每天背完后要听前一天您背过的单词。

这套书的录音非常的棒，是标准美式男女口音的MP3，没有中文，用英语解释英语，而且它的录音是带有“导读模式”的，不是以前我们在学校听到的那种录音，它在录音时还是经过了编排和摸索读者的听英语的习惯的，非常赞。

同时我在这边推荐大家在背这两本书时采用的一种方法，这也是一个学习方法论的介绍吧算：

1. 买一些厚的白板卡片纸，裁剪成7\*5cm左右的一张张小卡片
2. 一面写上英语，一面上上中文
3. 100张或者200张卡片用橡皮筋扎一捆
4. 每天身上带1-2捆，没事时就拿出来，看到的是中文，脑子里背出它的英语和how to spell的，如果你看到的是英语，那么就脑子里默背出它的中文意思吧

喏，就是下面这个样的，这是笔者当初开始强化单词时自己制作的小卡片，我是一个卡片上写2个英语单词，另一面也是两个中文的解析和翻译，100-200张卡片一捆。



2本书小半年，呵呵，如果你真的能够抗下来，你会发觉你的英语能力真的有点小成，不信，我们来测试一下你的水平吧！

#### 四、增加阅读量，增加一些“习惯性用语”

由其是老美，美国的IT技术是世界一流，老美写文档时经常会带有一种特殊的“美式俚语”，如果你不知道这些俚语的用法，有时在看一篇文章或者是论文时你都不知道自己在看点什么，因为是很多老美在自己的文章中使用了大量的俚语，怎么提高这方面的能力呢？

来了，看下面这套书：





这套书有7本，我介章大家在有了单词5000的基础上从第4册开始看到第7册，你会发觉大部分单词和意思你不需字典可以看得懂，这就是vocabulary 5000和10000两本书的好处。

这套书里，每一本书都有许许多多小故事，就和我们的故事会一样，并且都有录音，而且是标准美国现代男女口音，不带有中文的。

这里的每个故事都是由美国人写的，那可不是4级英语里的什么国家教委编写的，不一样的，这几本书里的英语由其是单词方面，你可以大量看到和听到美式英语、俚语的用法。

4-7本书，共4本书，差不多近百篇故事，每篇故事的背面是中文的完全翻译，然后是单词表，这边的单词表，嘿嘿，非常有趣，列出的都是美式俚语的一些用法，这正是我们需要的。

也是天天看，听，刚开始看第4册时会觉得比较简单，每天可以看1-2篇小故事，到了第5册，每天坚持看一篇小故事，要让你的生活中英语占有50%左右的时间，如果你不在外资企业或者对英语要求不高的外资企业，上面这些招数是你自己给自己培养英语环境的必备手段。

## 五、飞速提升你的口语

婴儿学语，通过的就是“先听”，把美国小孩从小放在中国来养，长大后他是标准的美国人长相，说一口标准的当地方言；如果把一个中国小孩放在美国从小养大，他将来就是标准的中国人长相，一口标准的美式英语。

来！

我们打开下面这个网站

<http://www.putclub.com/index.html>



通过点击这个“每日听力”



Monday 星期一

Tuesday 星期二

Wednesday 星期三

Thursday 星期四

Friday 星期五

Saturday 星期六

Sunday 星期日

Special English 特别英语

1530MP3PDFHTMLLRCFeatureMP3PDFHTMLLRCReportMP3PDFHTMLLRCAPMP3PDFHTMLLRC

Special English也就是我们常说的VOA慢速英语或特别英语，比较适合听力新手。节目有美国之音的1530news,report,feature,word-lover,豆知识,AP(美联社)时事一分钟视频新闻。

Standard English 标准新闻节目

BBCMP3PDFHTMLLRCVOAMP3PDFHTMLLRCNPRMP3PDFHTMLLRCNNMP3PDFHTMLLRC

Au-newsMP3PDFHTMLLRC

Standard English 听力训练，整点新闻（标准语速）：节目选自BBC、VOA、NPR(美国国家广播电台)、CNN、科学美国人（60 Second Science），澳洲（ABC新闻），节目有一定的难度,但是由于正式、规范，非常有利大家的提高。

Multi-Topics 标准英语多主题

IV(访谈录)MP3PDFHTMLLRCBC(商业新闻)MP3PDFHTMLLRCKS(万花筒)MP3PDFHTMLLRCIO(国际瞭望)MP3PDFHTMLLRC

TIB(我的信MP3PDFHTMLLRCIJ(科技前沿)MP3PDFHTMLLRC

意(英伦广角)MP3PDFHTMLLRC

初学者请从“**Special English**”开始听起，因为Special English是美国等英语国家为了照顾非英语国家的学生而说的很慢速的英语。

1. 通过下载MP3，每个MP3时间不超过2分钟，它讲的内容或者是一个新闻或者是一篇体育报道，或者是一件奇闻趣事。
2. 每天坚持听3个MP3，然后自己想想每个MP3大致在讲些什么内容，一定要先自己做到在听完后就想一下你听到的是在讲一件什么样的事
3. 然后我们进入下面这个版块

你好：lifetragedy，欢迎登录 退出登录

手机版 网站地图 设为首页

普特英语听力

www.putclub.com

听力VOABBC

雅思四级六级

口语阅读写作

演讲名著老友记

论坛少儿旧版

视听电影音乐

托福专四专八

翻译词汇语法

博客家园模仿

教程初级中级

BEC高考考研

金融外贸新概念

词典百科播客

下载 电台 赖世雄

首页 | 听力资源 | 每日听力 | 网络电台 | 在线词典 | 听力论坛 | 下载频道 | 部落家园 | 在线背单词 | 双语阅读 | 在线听写 | 普特网

VIPABC

学英语 保证有效

每天只要45分钟

免费英语能力测试

保证有效

What's new 上海自由贸易试验区将挂牌

Most popular 美文30篇

你好：lifetragedy，欢迎登录 退出登录

手机版 网站地图 设为首页

普特英语听力

www.putclub.com

听力VOABBC

雅思四级六级

口语阅读写作

演讲名著老友记

论坛少儿旧版

视听电影音乐

托福专四专八

翻译词汇语法

博客家园模仿

教程初级中级

BEC高考考研

金融外贸新概念

词典百科播客

下载 电台 赖世雄

首页 | 听力资源 | 每日听力 | 网络电台 | 在线词典 | 听力论坛 | 下载频道 | 部落家园 | 在线背单词 | 双语阅读 | 在线听写 | 普特网

VIPABC

学英语 保证有效

每天只要45分钟

免费英语能力测试

保证有效

What's new 上海自由贸易试验区将挂牌

Most popular 美文30篇

叫“听力论坛”，进入后

假设我们想知道，我听的是“每日听力”中，Special English中的1530这篇MP3，它到底在讲点什么

Monday 星期一

Tuesday 星期二

Wednesday 星期三

Thursday 星期四

Friday 星期五

Saturday 星期六

Sunday 星期日

Special English 特别英语

1530MP3PDFHTMLLRCFeatureMP3PDFHTMLLRCReportMP3PDFHTMLLRCAPMP3PDFHTMLLRC

Special English也就是我们常说的VOA慢速英语或特别英语，比较适合听力新手。节目有美国之音的1530news,report,feature,word-lover,豆知识,AP(美联社)时事一分钟视频新闻。

流利英语 7天找对感觉  
坚持一个月，听懂VOA

于是我们就可以用“听力论坛”中的搜索功能，去搜完整的“听力记录”，有那种GRE考满分或者是TOFEL考满分或者是在美国的留学生把他听到的完整内容记录下来，并且在“听力论坛”中分享给大家了。

搜索

1530

标题

搜索

高级

帖子

帖子商品分类信息

结果：找到“1530”相关主题 316 个

1234567

下一页

标题

版块

作者

回复/查看

最后发表

2.2【难度27】慢速类听写----SE 1530 News---测试帖子 ... 1 2 3 4 5 6

新手成长摇篮

sfere 2011-3-6

87 / 2634

2013-9-18 04:40

lossofit

重新发布轻量级的人名地名册子，1530的帖子注意了！ ... 1 2 3 4

Special 听力训练

sfere 2011-1-6

59 / 3462

2013-4-3 10:44

wocalei9p1

1530 voa special english原文？？

论坛申诉区

zilaixiong2 2012-10-31

1 / 632

2012-11-2 09:35

qingchengsha

求se 1530 news 的后半部分的英语原文

英语问题答疑区

qqm2010 2010-10-2

1 / 20

2012-8-17 13:21

baneit319

现在的1530新闻怎么找不到以前的音频和文本

英语问题答疑区

christiestone 2012-4-10

0 / 17

2012-4-10 10:06

christiestone

【整理】SENEWS-2008-01-21-1530

Special 听力训练

qingchengsha 2008-1-22

6 / 4401

2011-12-19 14:59

漫夜

【整理福汇总】SE1530NEWS (2011年)

Special 文本集中

Mickey\_lee 2011-1-13

5 / 23697

2011-6-21 03:48

Mickey\_lee

http://blog.csdn.net/rss.html?type=column&column=javainterview

9/46



有疑难，请进-- **【1530News】--新闻突击队**

版主提示：  
一、若是自己的听写稿，请发帖时标注 'Homework'。  
二、若是改稿，请发帖时标注 'on 某某人'并在修改处标红。  
三、为了达到最快的下载速度,推荐使用迅雷高速下载本站音频/视频材料。

第十一期招收新学员报名正在进行中！！

【整理】SENEWS-2008-01-21-1530

*Special thanks to yzmin, cyncina and sfere*blog.csdn.net/lifetragedy

It is 15:30 Universal Time. Here is the news in Special English. I'm Karen Leggett in Washington.

[1] The main electrical power center in the Gaza Strip has closed. Owners say they have no fuel to operate the center. Tens of thousands of Gazans are affected. The fuel supplies halted last Thursday when Israel closed the border around Gaza. United Nation officials, human rights groups and Palestinians have condemned the closure. They say the closure has pushed Gaza closed to a humanitarian disaster. But Israel said the closure was in reaction to rocket attacks from Gaza. Israel says that more than 200 rockets have hit Israel in the past week. Israel also says Hamas officials purposely cut electrical power in Gaza to put blame on Israel.

[2] Stock prices in Europe and Asia fell sharply Monday. In Hong Kong the price of shares fell 5.5%, the biggest drop since the terrorist attacks of September 11th, 2001. There were losses of 3-7% in India, China, Britain, France and Germany. Prices are dropping in part because of concern about the United States economy and fears of recession. American markets are closed Monday for the Martin Luther King Holiday.

OK，先自己听，自己想想：我听到了什么，再看看人家大牛的翻译。

半年，这样坚持下来，你的听力也会上升，你的"说"这道坎自然就跨过了，如果你坚持了1年，那你和同类人相比会迅速脱颖而出。

你的英语面试关，就很容易过了。

## 六、用英语来思考和让英语成为你的朋友

1. 听英语歌曲，不仅仅是要听音乐弦律啥优不优美的，是真的要能够听懂它在唱点啥，同时可以八卦一下歌唱者的一些花边啦、私生活啦，人生经历啦什么的。

2. 做到天天至少看一部美剧

捡自己喜欢看的美剧，内容最好能够覆盖科技、医疗、生活等方面。

Friends就算了，傻叉才会把Friends类为学习英语的首选。

我在这边推荐大家几部片吧

《星际之门》，好家伙10个季+5个季的亚特兰蒂斯番外篇，科学、医疗、自然常识全覆盖到了

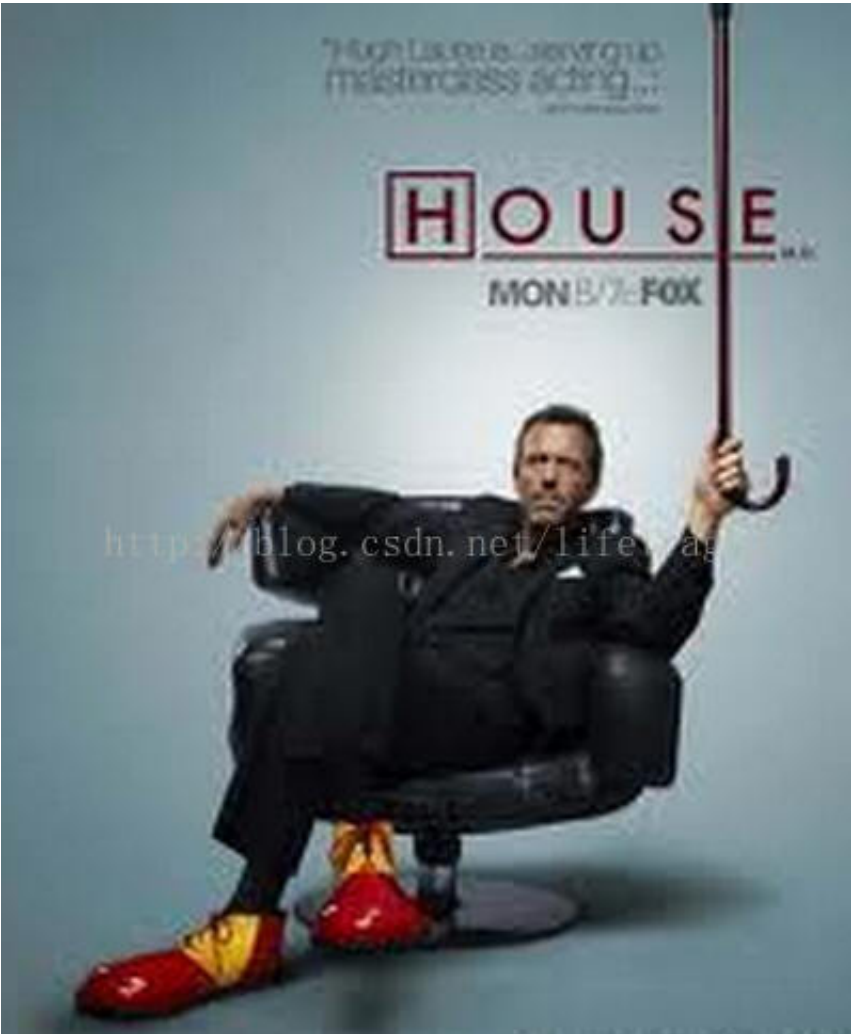


《太空堡垒卡拉狄加》，美国历史上获奖最多，收视率最高的美剧





《豪斯》， 这片从我大学毕业就一直看，看到到去年刚结束，而且这家伙喜欢吃苹果，“被咬掉一口的苹果，有没有，有没有懂的？”，天才豪斯医生啊，天才啊，都喜欢“苹果”。



《super natural》，不用多说了，杀鬼的，又有帅哥，又有MM，美国也是热播



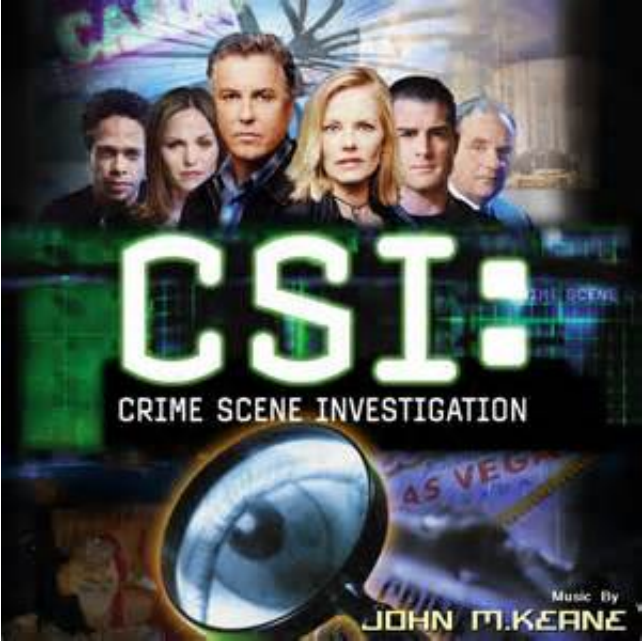
《行尸走肉》，这个不用多说了，目前美国最热



《陨落星辰》，这个是斯皮尔伯格导演的，一级棒，外星人直接把全地球给占领了



《CSI》，这个太长了节奏又快，科技含量也很高的一部美剧



《梅林传奇》，不是梅林午餐肉哦，如果说Friends主要是内容健康，那还不如去看这部梅林呢，那才叫正能量呢





反正，基本上要挑自己喜欢看的内容，集数最好长一点。

还有，不要看了就看了，要一边看一边学，怎么学呢？

因为这些美剧都是目前最火或者曾经最火的美剧，网上有大量的人把每一集都翻译成了字幕，这些字幕基本都是中英语对照着看的，即一行英语，一行中文。

你可以看完后把字幕看一遍，然后再看一遍原剧，看时，拿一张白纸条，把电脑显示器显示字幕的那个部位正好贴掉盖住，这样在收看美剧时全靠听力在看美剧，看到不懂时，看看字幕翻译，或者看看英语字幕是怎么描述的。  
半年有小成，一年有大成。

基本这些就是我提高英语的方法，它可以让你在非英语环境内，每天你的生活也能够有50%被英语占据。

# 贵在坚持！！！！

作者：lifetragedy 发表于2013/9/26 15:55:59 [原文链接](#)  
阅读：9346 评论：16 [查看评论](#)

## think in java interview-高级开发人员面试宝典(九)

### 模拟面试

今天来谈一下出于面试官的角度，他是会如何去考虑给一个候选人面试的。

下面先来看一份简历的摘要。

熟练掌握**SPRING, STRUTS1, 2, HIBERNATE**;

**J2EE**方面**5**年工作经验;

**ORACLE**使用经验**48**个月;

熟练掌握**JAVA**缓存机制;

了解**工作流、ERP、EAI**工作机制并有实际项目经验;

老实说，一般面试官看到上面的简历就会想让这个人来面试了，因为真的你SSH，ORACLE，ERP或者是EAI等工作过5年很厉害了已经，于是面试官在F2F（Face To Face）面试前会先准备一些题目。

一般他会做一些下面的准备，大家来看看这些题目，然后自己不用GOOGLE和书，试着去回答一下，就当是一次面试。

### 进入面试官的头脑

面试官会想，这个人J2EE有5年工作经验，JAVA一定很熟，我会先从JAVA方面入手。

#### 1. CORE JAVA的一些基础是肯定逃不掉的，如

a) HashMap, TreeMap的机制、源码片段这个候选人肯定应该能够分析、了解和掌握了。

b) Comparable与Comparator接口的使用。

c) stack的原理，如：计算 <{[<>]}这样的左边括号和右边的括号是否匹配，就可能通过stack的pop, push原理去实现。

d) 单向链表，双向链表的完型填空。

e) 序列化与反序列化的具体用法

f) hashCode Equals的原理，怎么写，算法，为什么要hashCode Equals，两个对象如果Equals了是不是hashCode一定相等，反之呢？

g) GOF设计模式考个2-3道，主要是问候选人碰到这样的情形用什么设计模式：装饰模式和工厂模式和候选人搞一下脑子，什么是ITERATOR模式。

让候选人自己写一个ITERATOR看看，去一个饭店点菜，每个饭店里都有一个菜单，然后顾客根据这个菜单点菜，这是一个什么设计模式？能否写一个菜单和顾客点菜的这样的一个设计模式来呢（享元设计模式）。

h) JAVA垃圾回收机制

i) 从《深入理解JAVA虚拟机》这本书里抽一道关于JDK平台的架构问题如：JAVA的新生代和老年代怎么安排他们的比例较好，新生代里放的是什么，老年代里放的又是什么？

j) 最后来两道数据结构，排序是肯定要考的，如果这个候选人开价较高，给他一道归并排序，归并排序归并排序归并排序，有没有，大家还记得吗？呵呵呵！

除了排序后，给候选人来一道较难的，如：约瑟夫环问题，汉诺塔问题，傅立叶级数问题，求极限问题，迷宫问题（有如何生成迷宫和走出迷宫两种问题）。

## 2. 多线程

a) 两个线程间如何通讯（不是WAIT, NOTIFY这种简单的哦），比如说一个线程如何传一个含有复杂结构的对象到另一个线程，对于批量数据如：读一个含有数百万行记录的.txt文本文件看看候选人有什么实际能力。

b) 线程锁的机制， synchronized关键字的用法，作用在method上。作用在method内以 synchronized(this){}开头的用法，synchronized(Object)的用法；  
public synchronized static method(){}又是如何使用的它和public synchronized method()有何区别？

在JDK1.6中有什么好的方法可以代替synchronized，lock关键字的使用。

c) 阻塞队列的作用？JDK1.5后的阻塞队列是怎么用的，用阻塞队列来模拟一个可以执行100个TASK的TASK POOL，即我有一个工作池，里面可以同时处理100条任务。

此时我有一个客户端，不断的往工作池里加任务，能否给出你的核心设计，拿支mark笔在黑板上画出。

同时请给出你认为比较优化的task pool的设计，并且给面试官解释一下你的设计。

## 3. 问点J2EE问题吧

a) 候选人用过Spring，Spring中AOP, IOC举个实际应用场景吧

b) 我在SPRING里把一个PROPERTIES文件注入，然后我在SPRING框架Coding时只要通过get properties文件中的一个key就可以得到这个value，相当于我不需要每次在读写properties文件时要输入或算出properties文件当前所在的入径，通过spring的context就可以得到这个properties文件的内容了。

c) 在spring中，一个类child继承自一个abstract的parent类，该child类要作为@Service到处注入的，在spring的配置文件如何写法？

d) 切面，SPRING的事务管理用过吧？用过，说一下SPRING中“事务的传播性”，有没有，有没有！事务切在Dao层，切在Service层，为什么要切在Dao层，为什么要切在Service层。

e) 除了事务你还作过哪些Spring方面的AOP编程？不要和我说对操作的method进行日志记录这个例子，因为SPRING出来时就是拿这两个例子来讲它的特性，是个JAVA程序员都知道。

你可以说说看Spring Batch里的“event” “flow”，不知大家看过没；

你可以说说看Spring与Ehcache的结合，利用AOP来模拟Hibernate里的二有缓存；

f) Spring JdbcTemplate的用法，这个肯定要考到的吧，对吧

g) 说一下jsp中有哪些“隐式对象”，以及他们的用法

h) servlet的生命周期

i) Struts1和Struts2的核心区别

j) Strutss2中常用的几个拦截器的用法（面试官可是会问里面的关键方法的使用的哦，因此你光会背STURTS2里有几个拦截器也是没有用的）

k) 改变一个request中的值，然后把改变的值再放回该request

L) 你说你用过缓存，如果我的应用布署在一个集群运用中，你的缓存是怎么设计的？每个集群节点一个缓存？那你有没有全局缓存，全局缓存你是怎么用在集群上的？

## 4. 再问一些Hibernate吧

a) 哦，你用过Hibernate，和我说一下Hibernate里我要对一张表进行锁定是怎么做的

b) 说一下Hibernate里evict, flush两个关键方法的区别

c) 用Hibernate插入1万条数据，怎么做，有什么好的优化方法吗

d) Hibernate的sessionFactory是怎么管理的，ThreadLocal的SessionFactory和非ThreadLocal的SessionFactory。

e) Hibernate和Spring结合后为什么我们就不用手动关闭SessionFactory了，不要和我说因为Spring帮我们关了，废话，是个人都知道，但原理是什么？是Spring里在哪边并且是怎么去帮我们关的？

f) Hibernate里 lazy loading的用法，由其是fetch关键字是怎么用的

## 5. 哦！！还有人撑到现在？没有say 88？好，下面继续

基本上，我们的问题问到这边，有些本来很自信的候选人，应该有几个已经动怒了。

这些候选人本来准备好的面试问题，结果在我们这边面到现在这个地步已经一塌糊涂了，这时可以问一下他们有什么问题，或者问一下他为什么要离开上家公司，然后基本可以和他们say 88了。

## 6. 从简历上看oracle用了48个月，4年（我还看到过用了6年，7年的呢），我们问点Oracle相关的问题吧

a) 什么叫数据库设计范式，来，把5个范式说一下，然后给你几个例题，让你说出，这样的设计是符合第一、第二、第三还是第四、第五范式，光背可没有用哦，呵呵呵。

b) 说过了数据库设计范式，什么叫反范式呢？

c) 做了48个月的Oracle了，对于日志归档，flashrecovery，SGA, PGA的设置管理及原理因该很知道了

e) 在plsql中选取100万条数据写成txt文件怎么做？只依赖于plsql

f) orace中的索引有几种，分别应用在何种场景

g) 给出一个sql，请用plsql的分析器即analyze来分析一下这条SQL的性能如何，有哪几个关键指标是你需要看的，什么叫hashjoin, 什么叫nestedloop

h) 用exist, not exist代替in的写法

i) 从我上次写的面试宝典第6天和第7天的那近百道SQL里选2道让候选人写写看，以此来看候选人的SQL能力

j) ORACLE中的bitmap 和 btree索引，说一下原理

k) 有没有必要对FK建索引啊？如果建和不建分别在ORACLE中会是什么样啊？

## 7. 智力题（这块我们放在“第十天”中来讲）

## 结束语：

基本到此为止，相信50个候选人里可能真正能够满足：

第一部分CORE JAVA答对70%以上

第二部分多线程答对60%以上

第三和第四部分合起来 答对70%以上

第六部分能够答对50%以上（但2道SQL题必须有一道是全答出，另一道复杂的可以写出思路 and 核心语句，不一定全答出）

第七部分智力题中，能够给出自己的想法，思路的，不一定得出最后答案（因为这种题很特别，有些候选人没有这种思考习惯或者是准备过的是基本答不出的）。

另外对于一些500强内的公司，候选人面临的还可能是第一部分、第二部分、第三、第四部分是**全英语**问答！

能够满足上述条件的大概人不多，50个里可能有1，2个满足吧，这点我没有夸张，这1，2个人，才是真正符合**JAVA**高级程序员资格的（当然，我指的是真的搞纯技术的一些公司、职位或者是部门），我的“面经”也是对于真正搞技术的，纯技术开发人员而言的，是这样的。

因此高级程序员不是这么好做的，面试也不是这么好准备的，不是靠一、两天突击背背维基百科上根据名词搜到的理论就行的，是要靠长期的积累，准备的。

在此我也只是给出一个准备的范围，要准备哪些东西，哪些方面是大家平时容易疏忽的。

今天就到这吧，第十天将集中探讨一些和任何语言无关的智力题，一些知名企业是必考这些的。

作者：lifetragedy 发表于2013/9/22 16:32:48 [原文链接](#)  
阅读：14661 评论：42 [查看评论](#)

### think in java interview-高级开发人员面试宝典(八)

面经出了7套，收到许多读者的Email，有许多人说了，这些基础知识是不是为了后面进一步的“通向架构师的道路”做准备的？

对的，你们没有猜错，就是这样的，我一直在酝酿后面的“通向架构师的道路”如何开章。

说实话，我已经在肚子里准备好的后面的“通向架构师的道路”的内容自己觉得如果一下子全拿出来的话，很多人吃不消，因为架构越来越复杂，用到的知识越来越多，而且很多都是各知识点的混合应用。

所以，先以这几套面经来铺路，我们把基础打实了，才能把大楼造的更好。因为，一个架构师首先他是一个程序员，他的基础知识必须非常的扎实，API对于架构师来说已经不太需要eclipse的code insight(即在eclipse编辑器里打一个小点点就可以得到后面的函数)，尤其是一些常用的JAVA API来说，是必须熟记于心的。

下面我们继续来几天面经，顺便便复习一下JAVA和数据库的一些基础。

## Java IO流的复习

大家平时J2EE写多了，JAVA的IO操作可能都已经生疏了，面试时如果来上这么几道，是不是有点“其实这个问题很简单，可是我就是想不起来”的感觉啊？

呵呵！

JAVA的IO操作太多，我这边挑腾讯，盛大和百度的几道面试题，并整理出答案来供大家参考。



## InputFromConsole

这个最简单不过了，从console接受用户输入的字符，如和用户有交互的命令行。

如果你不复习的话，嘿嘿，还真答不出，来看：

```
package org.sky.io;

public class InputFromConsole {

    /**
     * @param args
     */
    public static void main(String[] args) throws Exception {
        int a = 0;
        byte[] input = new byte[1024];
        System.in.read(input);
        System.out.println("your input is: " + new String(input));
    }

}
```

## ListDir

列出给出路径下所有的目录，包括子目录

```
package org.sky.io;

import java.io.*;

public class ListMyDir {

    /**
     * @param args
     */
    public static void main(String[] args) {
        String fileName = "D:" + File.separator + "tomcat2";
        File f = new File(fileName);
        File[] fs = f.listFiles();
        for (int i = 0; i < fs.length; i++) {
            System.out.println(fs[i].getName());
        }
    }

}
```

哦，上面这个程序只列出了一层目录，我们想连子目录一起List出来怎么办？

## ListMyDirWithSubDir

```
package org.sky.io;

import java.io.*;

public class ListMyDirWithSubDir {

    /**
     * @param args
     */
    public void print(File f) {
        if (f != null) {
            if (f.isDirectory()) {
                File[] fileArray = f.listFiles();
                if (fileArray != null) {
                    for (int i = 0; i < fileArray.length; i++) {
                        print(fileArray[i]);
                    }
                }
            } else {
                System.out.println(f);
            }
        }
    }

    public static void main(String[] args) {
        String fileName = "D:" + File.separator + "tomcat2";
        File f = new File(fileName);
        ListMyDirWithSubDir listDir = new ListMyDirWithSubDir();
        listDir.print(f);
    }

}
```

## InputStreamDemo

从外部读入一个文件

```
package org.sky.io;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.InputStream;
import java.io.OutputStream;

public class InputStreamDemo {
    public void readFile(String fileName) {
        File srcFile = new File(fileName);
        InputStream in = null;
        try {
            in = new FileInputStream(srcFile);
            byte[] b = new byte[(int) srcFile.length()];
            for (int i = 0; i < b.length; i++) {
                b[i] = (byte) in.read();
            }
            System.out.println(new String(b));
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            try {
                if (in != null) {
                    in.close();
                    in = null;
                }
            }
        }
    }
}
```

```
        }
    } catch (Exception e) {
    }
}

public static void main(String[] args) {
    InputStreamDemo id = new InputStreamDemo();
    String src = "D:" + File.separator + "hello.txt";
    id.readFile(src);
}
}
```

## OutputStreamDemo

讲完了InputStream来讲OutputStream，输出内容至外部的一个文件

```
package org.sky.io;

import java.io.*;

public class OutputStreamDemo {

    public void writeWithByte() {
        String fileName = "D:" + File.separator + "hello.txt";
        OutputStream out = null;
        File f = new File(fileName);
        try {
            out = new FileOutputStream(f, true);
            String str = "    [Publicity ministry of Shanghai Municipal committee of CPC]";
            byte[] b = str.getBytes();
            out.write(b);
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            try {
                if (out != null) {
                    out.close();
                    out = null;
                }
            } catch (Exception e) {
            }
        }
    }

    public void writeWithByteArray() {
        String fileName = "D:" + File.separator + "hello.txt";
        OutputStream out = null;
        File f = new File(fileName);
        try {
            out = new FileOutputStream(f, true);
            String str = "    [hello with byte yi ge ge xie]";
            byte[] b = str.getBytes();
            for (int i = 0; i < b.length; i++) {
                out.write(b[i]);
            }
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            try {
                if (out != null) {
                    out.close();
                    out = null;
                }
            } catch (Exception e) {
            }
        }
    }

    public static void main(String[] args) {
        OutputStreamDemo od = new OutputStreamDemo();
        od.writeWithByte();
        od.writeWithByteArray();
    }
}
```

这个Demo里分别用了“writeWithByte”和 “writeWithByteArray”两种方法，注意查看

## CopyFile

我们讲完了InputStream和OutputStream，我们就可以自己实现一个File Copy的功能了，来看

```
package org.sky.io;

import java.io.*;

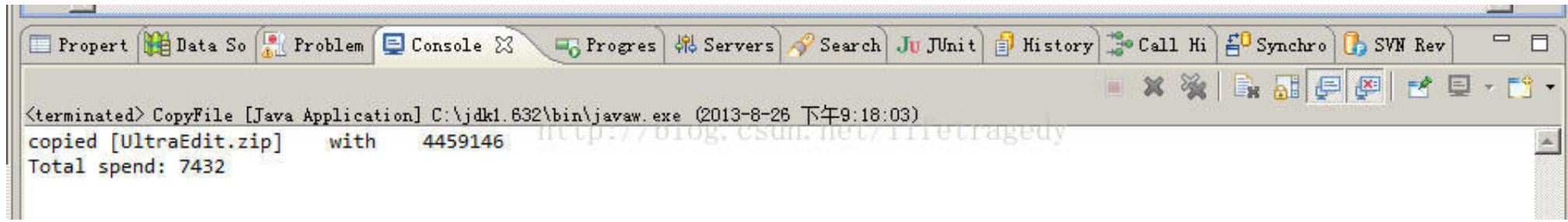
public class CopyFile {

    public void copy(String src, String des) {
        File srcFile = new File(src);
        File desFile = new File(des);
        InputStream in = null;
        OutputStream out = null;
        try {
            in = new FileInputStream(srcFile);
            out = new FileOutputStream(desFile);
            byte[] b = new byte[(int) srcFile.length()];
            for (int i = 0; i < b.length; i++) {
                b[i] = (byte) in.read();
            }
            out.write(b);
            System.out.println("copied [" + srcFile.getName() + "]    with    "
                               + srcFile.length());
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            try {
                if (out != null) {
                    out.close();
                    out = null;
                }
            } catch (Exception e) {
            }
            try {
                if (in != null) {
                    in.close();
                    in = null;
                }
            } catch (Exception e) {
            }
        }
    }

    public static void main(String[] args) {
        CopyFile cp = new CopyFile();
    }
}
```

```
String src = "D:" + File.separator + "UltraEdit.zip";
String des = "D:" + File.separator + "UltraEdit_Copy.zip";
long sTime = System.currentTimeMillis();
cp.copy(src, des);
long eTime = System.currentTimeMillis();
System.out.println("Total spend: " + (eTime - sTime));
}
}
```

运行后显示:



来看我们被Copy的这个文件的大小:

UltraEdit.zip	2011/9/26 10:37	WinZip File	4,355 KB
---------------	-----------------	-------------	----------

也不大，怎么用了7秒多？

原是我们没有使用 **Buffer** 这个东西，即缓冲，性能会相差多大呢？来看

## BufferInputStreamDemo

```
package org.sky.io;
```

```
import java.io.*;
```

```
public class BufferInputStreamDemo {

    /**
     * @param args
     */
    public void copy(String src, String des) {
        File srcFile = new File(src);
        File desFile = new File(des);
        BufferedInputStream bin = null;
        BufferedOutputStream bout = null;
        try {
            bin = new BufferedInputStream(new FileInputStream(srcFile));
            bout = new BufferedOutputStream(new FileOutputStream(desFile));
            byte[] b = new byte[1024];
            while (bin.read(b) != -1) {
                bout.write(b);
            }
            bout.flush();
            System.out.println("copied [" + srcFile.getName() + "] with " + " "
                               + srcFile.length());
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            try {
                if (bout != null) {
                    bout.close();
                    bout = null;
                }
            } catch (Exception e) {
            }
            try {
                if (bin != null) {
                    bin.close();
                    bin = null;
                }
            } catch (Exception e) {
            }
        }
    }

    public static void main(String[] args) {
        BufferInputStreamDemo bd = new BufferInputStreamDemo();
        String src = "D:" + File.separator + "UltraEdit.zip";
        String des = "D:" + File.separator + "UltraEdit_Copy.zip";
        long sTime = System.currentTimeMillis();
        bd.copy(src, des);
        long eTime = System.currentTimeMillis();
        System.out.println("Total spend: " + (eTime - sTime));
    }
}
```

我们Copy同样一个文件，用了多少时间呢？来看！



丫只用了14毫秒，CALL！！！

## ByteArrayDemo

来看看使用ByteArray输出文件吧

```
package org.sky.io;
```

```
import java.io.*;
```

```
public class ByteArrayDemo {

    /**
     * @param args
     */
    public void testByteArray() {
        String str = "HOLLYJESUS";
        ByteArrayInputStream input = null;
        ByteArrayOutputStream output = null;
        try {
            input = new ByteArrayInputStream(str.getBytes());
            output = new ByteArrayOutputStream();
            int temp = 0;
            while ((temp = input.read()) != -1) {
                char ch = (char) temp;
                output.write(Character.toLowerCase(ch));
            }
        }
    }
}
```



```
        }
        String outStr = output.toString();
        input.close();
        output.close();
        System.out.println(outStr);
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        try {
            if (output != null) {
                output.close();
                output = null;
            }
        } catch (Exception e) {
        }
        try {
            if (input != null) {
                input.close();
                input = null;
            }
        } catch (Exception e) {
        }
    }
}

public static void main(String[] args) {
    ByteArrayDemo bd = new ByteArrayDemo();
    bd.testByteArray();
}
}
```

## RandomAccess

有种输出流叫**Random**，你们还记得吗？学习时记得的，工作久了，**HOHO**，忘了，它到底有什么特殊的地方呢？来看：

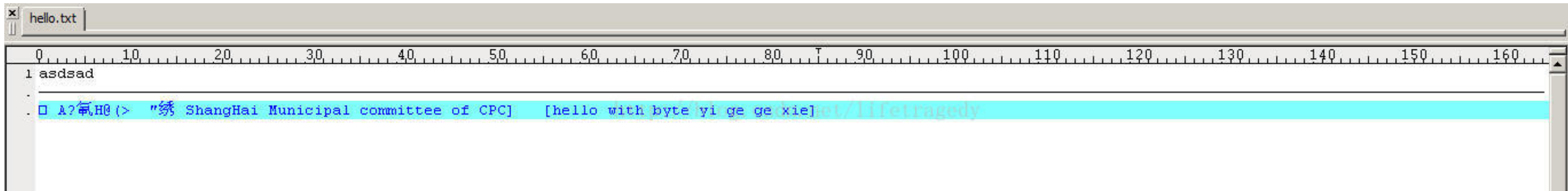
```
package org.sky.io;

import java.io.*;

public class RandomAccess {
    public void writeToFile() {
        String fileName = "D:" + File.separator + "hello.txt";
        RandomAccessFile randomIO = null;
        try {
            File f = new File(fileName);
            randomIO = new RandomAccessFile(f, "rw");
            randomIO.writeBytes("asdsad");
            randomIO.writeInt(12);
            randomIO.writeBoolean(true);
            randomIO.writeChar('A');
            randomIO.writeFloat(1.21f);
            randomIO.writeDouble(12.123);
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            try {
                if (randomIO != null) {
                    randomIO.close();
                    randomIO = null;
                }
            } catch (Exception e) {
            }
        }
    }

    public static void main(String[] args) {
        RandomAccess randomA = new RandomAccess();
        randomA.writeToFile();
    }
}
```

它输出后的文件是怎么样子的呢？



## PipeStream

这个流很特殊，我们在线程操作时，两个线程都在运行，这时通过发送一个指令让某个线程**do something**，我们在以前的**jdk1.4**中为了实现这样的功能，使用的就是这个**PipeStream**

先来看两个类，一个叫**SendMessage**，即发送一个指令。一个叫**ReceiveMessage**，用于接受指令。

### SendMessage

```
package org.sky.io;

import java.io.*;

public class SendMessage implements Runnable {
    private PipedOutputStream out = null;

    public PipedOutputStream getOut() {
        return this.out;
    }

    public SendMessage() {
        this.out = new PipedOutputStream();
    }

    public void send() {
        String msg = "start";
        try {
            out.write(msg.getBytes());
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            try {
                if (out != null) {
                    out.close();
                    out = null;
                }
            }
        }
    }
}
```

```
        }
        } catch (Exception e) {
        }
    }

    public void run() {
        try {
            System.out.println("waiting for signal...");
            Thread.sleep(2000);
            send();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

ReceiveMessage

```
package org.sky.io;

import java.io.*;

public class ReceiveMessage implements Runnable {
    private PipedInputStream input = null;

    public PipedInputStream getInput() {
        return this.input;
    }

    public ReceiveMessage() {
        this.input = new PipedInputStream();
    }

    private void receive() {
        byte[] b = new byte[1000];
        int len = 0;
        String msg = "";
        try {
            len = input.read(b);
            msg = new String(b, 0, len);
            if (msg.equals("start")) {
                System.out
                    .println("received the start message, receive now can do something.....");
                Thread.interrupted();
            }
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            try {
                if (input != null) {
                    input.close();
                    input = null;
                }
            } catch (Exception e) {
            }
        }
    }

    public void run() {
        try {
            receive();
        } catch (Exception e) {
        }
    }
}
```

如何使用这两个类呢？

TestPipeStream

```
package org.sky.io;

public class TestPipeStream {

    /**
     * @param args
     */
    public static void main(String[] args) {
        SendMessage send = new SendMessage();
        ReceiveMessage receive = new ReceiveMessage();
        try {
            send.getOutputStream().connect(receive.getInputStream());
            Thread t1 = new Thread(send);
            Thread t2 = new Thread(receive);
            t1.start();
            t2.start();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

注意这边有一个send.getOutputStream().connect(receive.getInputStream());

这个方法就把两个线程“connect”起来了。

Serializable的IO操作

把一个类序列化到磁盘上，怎么做？

先来看我们要序列化的一个Java Bean

Person

```
package org.sky.io;

import java.io.Serializable;

public class Person implements Serializable {

    private String name = "";
    private String age = "";
    private String personId = "";
```

```
public String getName() {
    return name;
}

public void setName(String name) {
    this.name = name;
}

public String getAge() {
    return age;
}

public void setAge(String age) {
    this.age = age;
}

public String getPersonId() {
    return personId;
}

public void setPersonId(String personId) {
    this.personId = personId;
}

public String getCellPhoneNo() {
    return cellPhoneNo;
}

public void setCellPhoneNo(String cellPhoneNo) {
    this.cellPhoneNo = cellPhoneNo;
}

private String cellPhoneNo = "";
```

下面来看序列化的操作

## SerializablePersonToFile

```
package org.sky.io;
```

```
import java.io.*;
import java.util.*;
```

```
public class SerializablePersonToFile {

    /**
     * @param args
     */
    private List<Person> initList() {
        List<Person> userList = new ArrayList<Person>();
        Person loginUser = new Person();
        loginUser.setName("sam");
        loginUser.setAge("30");
        loginUser.setCellPhoneNo("13333333333");
        loginUser.setPersonId("111111111111111111");
        userList.add(loginUser);
        loginUser = new Person();
        loginUser.setName("tonny");
        loginUser.setAge("31");
        loginUser.setCellPhoneNo("14333333333");
        loginUser.setPersonId("111111111111111111");
        userList.add(loginUser);
        loginUser = new Person();
        loginUser.setName("jim");
        loginUser.setAge("28");
        loginUser.setCellPhoneNo("15333333333");
        loginUser.setPersonId("111111111111111111");
        userList.add(loginUser);
        loginUser = new Person();
        loginUser.setName("Simon");
        loginUser.setAge("30");
        loginUser.setCellPhoneNo("17333333333");
        loginUser.setPersonId("111111111111111111");
        userList.add(loginUser);
        return userList;
    }

    private void serializeFromFile() {
        FileInputStream fs = null;
        ObjectInputStream ois = null;
        try {
            fs = new FileInputStream("person.txt");
            ois = new ObjectInputStream(fs);
            List<Person> userList = (ArrayList<Person>) ois.readObject();
            for (Person p : userList) {
                System.out.println(p.getName() + " " + p.getAge() + " "
                    + p.getCellPhoneNo() + " " + p.getCellPhoneNo());
            }
        } catch (Exception ex) {
            ex.printStackTrace();
        } finally {
            try {
                if (ois != null) {
                    ois.close();
                }
            } catch (Exception e) {
            }
            try {
                if (fs != null) {
                    fs.close();
                }
            } catch (Exception e) {
            }
        }
    }

    private void serializeToFile() {
        List<Person> userList = new ArrayList<Person>();
        userList = initList();
        FileOutputStream fs = null;
        ObjectOutputStream os = null;
        try {
            fs = new FileOutputStream("person.txt");
            os = new ObjectOutputStream(fs);
            os.writeObject(userList);
        } catch (Exception ex) {
            ex.printStackTrace();
        } finally {
            try {
                if (os != null) {
                    os.close();
                }
            } catch (Exception e) {
            }
            try {
                if (fs != null) {
                    fs.close();
                }
            } catch (Exception e) {
            }
        }
    }
}
```



```
    }

    public static void main(String[] args) {
        SerializablePersonToFile sf = new SerializablePersonToFile();
        sf.serializeToFile();
        sf.serializeFromFile();
    }
}
```

这边先把Person输出到Person.txt，再从Person.txt里反序列化出这个Person的Java Bean。

先讲这么些吧！

Java的流操作还有很多，这些是经常会被面试到的，很基础，因此经常被考到。

以前听一个读IT的同学说过，这些IO操作，就算没有Eclipse编辑器的话，用文本编辑器也应该能够写出来，你写不出只能代表你的基础太弱了。

作者：lifetragedy 发表于2013/8/26 22:05:25 [原文链接](#)  
阅读： 11797 评论： 37 [查看评论](#)

## think in java interview-高级开发人员面试宝典(七)

上两周研发任务太紧了，所以耽搁了一下，我们继续我们的面试之旅。

下面是一个基于图书系统的15道SQL问答，供大家参考

- 问题描述：
- 本题用到下面三个关系表：
- CARD 借书卡。 CNO 卡号，NAME 姓名，CLASS 班级
- BOOKS 图书。 BNO 书号，BNAME 书名,AUTHOR 作者，PRICE 单价， QUANTITY 库存册数
- BORROW 借书记录。 CNO 借书卡号，BNO 书号，RDATE 还书日期
- 备注：限定每人每种书只能借一本；库存册数随借书、还书而改变。
- 要求实现如下15个处理：
1. 写出建立BORROW表的SQL语句，要求定义主码完整性约束和引用完整性约束。
  2. 找出借书超过5本的读者,输出借书卡号及所借图书册数。
  3. 查询借阅了“水浒”一书的读者，输出姓名及班级。
  4. 查询过期未还图书，输出借阅者（卡号）、书号及还书日期。
  5. 查询书名包括“网络”关键词的图书，输出书号、书名、作者。
  6. 查询现有图书中价格最高的图书，输出书名及作者。
  7. 查询当前借了“计算方法”但没有借“计算方法习题集”的读者，输出其借书卡号，并按卡号降序排序输出。
  8. 将“C01”班同学所借图书的还期都延长一周。
  9. 从BOOKS表中删除当前无人借阅的图书记录。
  10. 如果经常按书名查询图书信息，请建立合适的索引。
  11. 在BORROW表上建立一个触发器，完成如下功能：如果读者借阅的书名是"数据库技术及应用", 就将该读者的借阅记录保存在BORROW\_SAVE表中（注ORROW\_SAVE表结构同BORROW表）。
  12. 建立一个视图，显示"力01"班学生的借书信息（只要求显示姓名和书名）。
  13. 查询当前同时借有“计算方法”和“组合数学”两本书的读者，输出其借书卡号，并按卡号升序排序输出。
  14. 假定在建BOOKS表时没有定义主码，写出为BOOKS表追加定义主码的语句。
  15. 对CARD表做如下修改：
    - a. 将NAME最大列宽增加到10个字符（假定原为6个字符）。
    - b. 为该表增加1列NAME（系名），可变长，最大20个字符。

1. 写出建立BORROW表的SQL语句，要求定义主码完整性约束和引用完整性约束

一实现代码：

```
CREATE TABLE BORROW (
    CNO int FOREIGN KEY REFERENCES CARD (CNO),
    BNO int FOREIGN KEY REFERENCES BOOKS (BNO),
    RDATE datetime,
    PRIMARY KEY (CNO, BNO))
```

2. 找出借书超过5本的读者,输出借书卡号及所借图书册数

一实现代码：

```
SELECT CNO, 借图书册数=COUNT(*)
FROM BORROW
GROUP BY CNO
HAVING COUNT(*)>5
```

3. 查询借阅了“水浒”一书的读者，输出姓名及班级

一实现代码：

```
SELECT * FROM CARD c
WHERE EXISTS (
    SELECT * FROM BORROW a,BOOKS b
    WHERE a.BNO=b.BNO
    AND b.BNAME='N' '水浒'
    AND a.CNO=c.CNO)
```

4. 查询过期未还图书，输出借阅者（卡号）、书号及还书日期

一实现代码：

```
SELECT * FROM BORROW
WHERE RDATE<GETDATE()
```

5. 查询书名包括“网络”关键词的图书，输出书号、书名、作者

一实现代码：

SELECT BNO,BNAME,AUTHOR FROM BOOKS

WHERE BNAME LIKE N' %网络%

6. 查询现有图书中价格最高的图书，输出书名及作者

—实现代码：

SELECT BNO,BNAME,AUTHOR FROM BOOKS

WHERE PRICE=(  
SELECT MAX(PRICE) FROM BOOKS)

7. 查询当前借了“计算方法”但没有借“计算方法习题集”的读者，输出其借书卡号，并按卡号降序排序输出

—实现代码：

SELECT a.CNO

FROM BORROW a,BOOKS b

WHERE a.BNO=b.BNO AND b.BNAME=N' 计算方法'  
AND NOT EXISTS(  
SELECT \* FROM BORROW aa,BOOKS bb  
WHERE aa.BNO=bb.BNO  
AND bb.BNAME=N' 计算方法习题集'  
AND aa.CNO=a.CNO)

ORDER BY a.CNO DESC

8. 将“C01”班同学所借图书的还期都延长一周

—实现代码：

UPDATE b SET RDATE=DATEADD(Day,7,b.RDATE)

FROM CARD a,BORROW b

WHERE a.CNO=b.CNO  
AND a.CLASS=N' C01'

9. 从BOOKS表中删除当前无人借阅的图书记录

—实现代码：

DELETE A FROM BOOKS a

WHERE NOT EXISTS(  
SELECT \* FROM BORROW  
WHERE BNO=a.BNO)

10. 如果经常按书名查询图书信息，请建立合适的索引

—实现代码：

CREATE CLUSTERED INDEX IDX\_BOOKS\_BNAME ON BOOKS(BNAME)

11. 在BORROW表上建立一个触发器，完成如下功能：如果读者借阅的书名是“数据库技术及应用”，就将该读者的借阅记录保存在BORROW\_SAVE表中（注ORROW\_SAVE表结构同BORROW表）

—实现代码：

CREATE TRIGGER TR\_SAVE ON BORROW

FOR INSERT,UPDATE

AS

IF @@ROWCOUNT>0  
INSERT BORROW\_SAVE SELECT i.\*

FROM INSERTED i,BOOKS b

WHERE i.BNO=b.BNO  
AND b.BNAME=N' 数据库技术及应用'

12. 建立一个视图，显示“力01”班学生的借书信息（只要求显示姓名和书名）

—实现代码：

CREATE VIEW V\_VIEW

AS

SELECT a.NAME,b.BNAME

FROM BORROW ab,CARD a,BOOKS b

WHERE ab.CNO=a.CNO  
AND ab.BNO=b.BNO  
AND a.CLASS=N' 力01'

13. 查询当前同时借有“计算方法”和“组合数学”两本书的读者，输出其借书卡号，并按卡号升序排序输出

—实现代码：

SELECT a.CNO

FROM BORROW a,BOOKS b

WHERE a.BNO=b.BNO  
AND b.BNAME IN(N' 计算方法',N' 组合数学')

GROUP BY a.CNO

HAVING COUNT(\*)=2

ORDER BY a.CNO DESC

14. 假定在建BOOKS表时没有定义主码，写出为BOOKS表追加定义主码的语句

—实现代码：

ALTER TABLE BOOKS ADD PRIMARY KEY (BNO)

15.1 将NAME最大列宽增加到10个字符（假定原为6个字符）

—实现代码:

```
ALTER TABLE CARD ALTER COLUMN NAME varchar(10)
```

15.2 为该表增加1列NAME（系名），可变长，最大20个字符

—实现代码:

```
ALTER TABLE CARD ADD 系名 varchar(20)
```

问题描述:

为管理岗位业务培训信息，建立3个表:

S (S#,SN,SD,SA) S#,SN,SD,SA 分别代表学号、学员姓名、所属单位、学员年龄

C (C#,CN ) C#,CN 分别代表课程编号、课程名称

SC ( S#,C#,G ) S#,C#,G 分别代表学号、所选修的课程编号、学习成绩

作者: lifetragedy 发表于2013/8/25 17:42:26 [原文链接](#)  
阅读: 8617 评论: 3 [查看评论](#)

think in java interview~高级开发人员面试宝典(六)

写了这么多JAVA基础，来点SQL吧！

一般面试时考SQL，主要就是考你“**统计分析**”这一块，下面我们来看面试官经常采用的手段。

由**4**张简单的不能再简单的表，演变出**50**道**SQL**

哈哈哈哈哈，够这个面试官面个**15**，**20**个人，不带重复的了，而且每个**SQL**你真的不动动脑子还写不出呢，你别不服气，下面开始。

表结构:

表**Student**

(S#,Sname,Sage,Ssex) 学生表

S#	student_no
Sage	student_age
Ssex	student_sex

表**Course**

(C#,Cname,T#) 课程表

C#	course_no
Cname	course_name
T#	teacher_no

表**SC**（学生与课程的分**数mapping**表）

(S#,C#,score) 成绩表

S#	student_no
C#	course_no
score	分数啦

表**Teacher**

(T#,Tname) 教师表

T#	teacher_no
Tname	teacher_name

**50**道问题开始

**1**、查询“**001**”课程比“**002**”课程成绩高的所有学生的学号:

```
select a.S# from (select s#,score from SC where C#='001') a,(select s#,score
from SC where C#='002')
```

```
where a.score>b.score and a.s#=b.s#;
```

**2**、查询平均成绩大于**60**分的同学的学号和平均成绩:

```
select S#,avg(score)
from sc
```



```
group by S# having avg(score) >60;
```

3、查询所有同学的学号、姓名、选课数、总成绩；

```
select Student.S#,Student.Sname,count(SC.C#),sum(score)
from Student left Outer join SC on Student.S#=SC.S#
group by Student.S#,Sname
```

4、查询姓“李”的老师的个数；

```
select count(distinct(Tname))
from Teacher
where Tname like '李%';
```

5、查询没学过“叶平”老师课的同学的学号、姓名；

```
select Student.S#,Student.Sname
from Student
where S# not in (select distinct( SC.S#) fromSC,Course,Teacher where SC.C#=Course.C#and Teacher.T#=Course.T# andTeacher.Tname='叶平');
```

6、查询学过“001”并且也学过编号“002”课程的同学的学号、姓名；

```
select Student.S#,Student.Sname fromStudent,SC where Student.S#=SC.S# andSC.C#='001'and exists( Select * from SC as SC_2 where SC_2.S#=SC.S# and SC_2.C#='002');
```

7、查询学过“叶平”老师所教的所有课的同学的学号、姓名；

```
select S#,Sname
from Student
where S# in (select S# from SC,Course ,Teacher where SC.C#=Course.C# andTeacher.T#=Course.T# and Teacher.Tname='叶平'group by S# having count(SC.C#)=(select count(C#) fromCourse,Teacher whereTeacher.T#=Course.T# and Tname='叶平'));
```

8、查询课程编号“002”的成绩比课程编号“001”课程低的所有同学的学号、姓名；

```
Select S#,Sname from (select Student.S#,Student.Sname,score ,(select score from SC SC_2 where SC_2.S#=Student.S#and SC_2.C#='002') score2
from Student,SC where Student.S#=SC.S# andC#='001') S_2 where score2 <score;
```

9、查询所有课程成绩小于60分的同学的学号、姓名；

```
select S#,Sname
from Student
where S# not in (select Student.S# fromStudent,SC where S.S#=SC.S# andscore>60);
```

10、查询没有学全所有课的同学的学号、姓名；

```
select Student.S#,Student.Sname
from Student,SC
whereStudent.S#=SC.S# group by Student.S#,Student.Sname having count(C#) <(select count(C#) from Course);
```

11、查询至少有一门课与学号为“1001”的同学所学相同的同学的学号和姓名；

```
select S#,Sname from Student,SC whereStudent.S#=SC.S# and C# in select C# from SC where S#='1001';
```

12、查询至少学过学号为“001”同学所有一门课的其他同学学号和姓名；

```
select distinct SC.S#,Sname
from Student,SC
where Student.S#=SC.S# and C# in(select C# from SC where S#='001');
```

13、把“SC”表中“叶平”老师教的课的成绩都更改为此课程的平均成绩；

```
update SC set score=(select avg(SC_2.score)
from SC SC_2
where SC_2.C#=SC.C# ) fromCourse,Teacher where Course.C#=SC.C# andCourse.T#=Teacher.T# and Teacher.Tname='叶平');
```

14、查询和“1002”号的同学学习的课程完全相同的其他同学学号和姓名；

```
select S# from SC where C# in(select C# from SC where S#='1002')
group by S# having count(*)=(select count(*) from SC where S#='1002');
```

15、删除学习“叶平”老师课的SC表记录；

```
DelectSC
from course ,Teacher
where Course.C#=SC.C# and Course.T#=Teacher.T# and Tname='叶平';
```

16、向SC表中插入一些记录，这些记录要求符合以下条件：没有上过编号“003”课程的同学学号、2号课的平均成绩；

```
Insert SC select S#,'002',(Select avg(score)
from SC where C#='002') from Student where S# notin (Select S# from SC where C#='002');
```

17、按平均成绩从高到低显示所有学生的“数据库”、“企业管理”、“英语”三门的课程成绩，按

如下形式显示： 学生ID,,数据库,企业管理,英语,有效课程数,有效平均分

```
SELECT S# as 学生ID
,(SELECT score FROM SC WHERE SC.S#=t.S#AND C#='004') AS 数据库
,(SELECT score FROM SC WHERE SC.S#=t.S#AND C#='001') AS 企业管理
,(SELECT score FROM SC WHERE SC.S#=t.S#AND C#='006') AS 英语
,COUNT(*) AS 有效课程数, AVG(t.score) AS 平均成绩
FROM SC AS t
GROUP BY S#
ORDER BY avg(t.score)
```

18、查询各科成绩最高和最低的分：以如下形式显示：课程ID，最高分，最低分

```
SELECT L.C# AS 课程ID,L.score AS 最高分,R.score AS 最低分
FROM SC L ,SC AS R
WHERE L.C# = R.C# and
L.score = (SELECT MAX(IL.score)
FROM SC ASIL,Student AS IM
WHERE L.C# =IL.C# and IM.S#=IL.S#
GROUP BYIL.C#)
AND
R.Score= (SELECT MIN(IR.score)
FROM SC ASIR
WHERE R.C# =IR.C#
GROUP BY IR.C#
);
```

19、按各科平均成绩从低到高和及格率的百分数从高到低顺序

```
SELECT t.C# AS 课程号,max(course.Cname)AS 课程名,isnull(AVG(score),0) AS 平均成绩
,100 * SUM(CASE WHEN isnull(score,0)>=60 THEN 1 ELSE 0 END)/COUNT(*) AS 及格百分数
FROM SC T,Course
where t.C#=course.C#
GROUP BY t.C#
ORDER BY 100* SUM(CASE WHEN isnull(score,0)>=60 THEN 1 ELSE 0 END)/COUNT(*) DESC
```

20、查询如下课程平均成绩和及格率的百分数(用"1行"显示):

企业管理（001），马克思（002），OO&UML（003），数据库（004）

```
SELECT SUM(CASE WHEN C# ='001' THEN score ELSE 0 END)/SUM(CASE C# WHEN '001' THEN 1 ELSE 0 END) AS 企业管理平均分
,100 * SUM(CASE WHEN C# = '001' AND score >= 60 THEN 1 ELSE 0 END)/SUM(CASE WHEN C# = '001' THEN 1 ELSE 0 END) AS 企业管理及格百分数
,SUM(CASE WHEN C# = '002' THEN score ELSE 0 END)/SUM(CASE C# WHEN '002' THEN 1 ELSE 0 END) AS 马克思平均分
,100 * SUM(CASE WHEN C# = '002' AND score >= 60 THEN 1 ELSE 0 END)/SUM(CASE WHEN C# = '002' THEN 1 ELSE 0 END) AS 马克思及格百分数
,SUM(CASE WHEN C# = '003' THEN score ELSE 0 END)/SUM(CASE C# WHEN '003' THEN 1 ELSE 0 END) AS UML平均分
,100* SUM(CASE WHEN C# = '003' AND score >= 60 THEN 1 ELSE 0 END)/SUM(CASE WHEN C# = '003' THEN 1 ELSE 0 END) AS UML及格百分数
,SUM(CASE WHEN C# = '004' THEN score ELSE 0 END)/SUM(CASE C# WHEN '004' THEN 1 ELSE 0 END) AS 数据库平均分
,100 * SUM(CASE WHEN C# = '004' AND score >= 60 THEN 1 ELSE 0 END)/SUM(CASE WHEN C# = '004' THEN 1 ELSE 0 END) AS 数据库及格百分数
FROM SC
```

21、查询不同老师所教不同课程平均分从高到低显示

```
SELECT max(Z.T#) AS 教师ID,MAX(Z.Tname) AS 教师姓名,C.C# AS 课程 I D,MAX(C.Cname) AS 课程名称,AVG(Score) AS 平均成绩
FROM SC AS T,Course AS C ,Teacher AS Z
where T.C#=C.C# and C.T#=Z.T#
GROUP BY C.C#
ORDER BY AVG(Score) DESC
```

22、查询如下课程成绩第3名到第6名的学生成绩单:

企业管理（001），马克思（002），UML（003），数据库（004）

[学生ID],[学生姓名],企业管理,马克思,UML,数据库,平均成绩

```
SELECT DISTINCT top 3
SC.S# As 学生学号,
Student.Sname AS 学生姓名 ,
T1.score AS 企业管理,
T2.score AS 马克思,
T3.score AS UML,
```

```
T4.score AS 数据库,

ISNULL(T1.score,0) + ISNULL(T2.score,0) + ISNULL(T3.score,0) + ISNULL(T4.score,0) as 总分

FROM Student,SC LEFT JOIN SC AS T1

ON SC.S# = T1.S# AND T1.C# = '001'

LEFT JOIN SC AS T2

ON SC.S# = T2.S# AND T2.C# = '002'

LEFT JOIN SC AS T3

ON SC.S# = T3.S# AND T3.C# = '003'

LEFT JOIN SC AS T4

ON SC.S# = T4.S# AND T4.C# = '004'

WHERE student.S#=SC.S# and

ISNULL(T1.score,0) + ISNULL(T2.score,0) + ISNULL(T3.score,0) + ISNULL(T4.score,0)

NOT IN

(SELECT

DISTINCT

TOP 15 WITH TIES

ISNULL(T1.score,0) + ISNULL(T2.score,0) + ISNULL(T3.score,0) + ISNULL(T4.score,0)

FROM sc

LEFT JOIN sc AS T1

ON sc.S# = T1.S# AND T1.C# = 'k1'

LEFT JOIN sc AS T2

ON sc.S# = T2.S# AND T2.C# = 'k2'

LEFT JOIN sc AS T3

ON sc.S# = T3.S# AND T3.C# = 'k3'

LEFT JOIN sc AS T4

ON sc.S# = T4.S# AND T4.C# = 'k4'

ORDER BY ISNULL(T1.score,0) + ISNULL(T2.score,0) + ISNULL(T3.score,0) + ISNULL(T4.score,0) DESC);
```

23、统计列印各科成绩,各分数段人数:课程ID, 课程名称, [100-85], [85-70], [70-60], [ <60]

```
SELECT SC.C# as 课程ID, Cname as 课程名称

,SUM(CASE WHEN score BETWEEN 85 AND 100 THEN 1 ELSE 0 END) AS [100 - 85]

,SUM(CASE WHEN score BETWEEN 70 AND 85 THEN 1 ELSE 0 END) AS [85 - 70]

,SUM(CASE WHEN score BETWEEN 60 AND 70 THEN 1 ELSE 0 END) AS [70 - 60]

,SUM(CASE WHEN score < 60 THEN 1 ELSE 0 END) AS [60 -]

FROM SC,Course

where SC.C#=Course.C#

GROUP BY SC.C#,Cname;
```

24、查询学生平均成绩及其名次

```
SELECT 1+(SELECT COUNT( distinct 平均成绩)

FROM (SELECT S#,AVG(score) AS 平均成绩

FROM SC

GROUP BY S#

) AS T1

WHERE 平均成绩 > T2.平均成绩) as 名次,

S# as 学生学号,平均成绩

FROM (SELECT S#,AVG(score) 平均成绩

FROM SC

GROUP BY S#

) AS T2

ORDER BY 平均成绩 desc;
```

25、查询各科成绩前三名的记录:(不考虑成绩并列情况)

```
SELECT t1.S# as 学生ID,t1.C# as 课程ID,Score as 分数

FROM SC t1

WHERE score IN (SELECT TOP 3 score

FROM SC

WHERE t1.C#= C#

ORDER BY score DESC

)

ORDER BY t1.C#;
```

26、查询每门课程被选修的学生数

```
select c#,count(S#) from sc group by C#;
```

27、查询出只选修了一门课程的全部学生的学号和姓名

```
select SC.S#,Student.Sname,count(C#) AS 选课数

from SC ,Student

where SC.S#=Student.S# group by SC.S# ,Student.Sname having count(C#)=1;
```

28、查询男生、女生人数



```
Select count(Ssex) as 男生人数 from Student group by Ssex having Ssex='男';

Select count(Ssex) as 女生人数 from Student group by Ssex having Ssex='女';
```

29、查询姓“张”的学生名单

```
SELECT Sname FROM Student WHERE Sname like '张%';
```

30、查询同名同性学生名单，并统计同名人数

```
select Sname,count(*) from Student group by Sname having count(*)>1;
```

31、1981年出生的学生名单(注：Student表中Sage列的类型是datetime)

```
select Sname, CONVERT(char (11),DATEPART(year,Sage)) as age
from student
where CONVERT(char (11),DATEPART(year,Sage))='1981';
```

32、查询每门课程的平均成绩，结果按平均成绩升序排列，平均成绩相同时，按课程号降序排列

```
Select C#,Avg(score) from SC group by C# order by Avg(score),C# DESC ;
```

33、查询平均成绩大于85的所有学生的学号、姓名和平均成绩

```
select Sname,SC.S#,avg(score)
from Student,SC
where Student.S#=SC.S# group by SC.S#,Sname having avg(score)>85;
```

34、查询课程名称为“数据库”，且分数低于60的学生姓名和分数

```
Select Sname,isnull(score,0)
from Student,SC,Course
where SC.S#=Student.S# and SC.C#=Course.C# and Course.Cname='数据库' and score <60;
```

35、查询所有学生的选课情况；

```
SELECT SC.S#,SC.C#,Sname,Cname
FROM SC,Student,Course
where SC.S#=Student.S# and SC.C#=Course.C# ;
```

36、查询任何一门课程成绩在70分以上的姓名、课程名称和分数；

```
SELECT distinct student.S#,student.Sname,SC.C#,SC.score
FROM student,Sc
WHERE SC.score>=70 AND SC.S#=student.S#;
```

37、查询不及格的课程，并按课程号从大到小排列

```
select c# from sc where scor e <60 order by C# ;
```

38、查询课程编号为003且课程成绩在80分以上的学生的学号和姓名；

```
select SC.S#,Student.Sname from SC,Student where SC.S#=Student.S# and Score>80 and C#='003';
```

39、求选了课程的学生人数

```
select count(*) from sc;
```

40、查询选修“叶平”老师所授课程的学生中，成绩最高的学生姓名及其成绩

```
select Student.Sname,score
from Student,SC,Course C,Teacher
where Student.S#=SC.S# and SC.C#=C.C# and C.T#=Teacher.T# and Teacher.Tname='叶平' and SC.score=(select max(score)from SC where C#=C.C# );
```

41、查询各个课程及相应的选修人数

```
select count(*) from sc group by C#;
```

42、查询不同课程成绩相同的学生的学号、课程号、学生成绩

```
select distinct A.S#,B.score from SC A ,SC B where A.Score=B.Score and A.C# <>B.C# ;
```

43、查询每门功成绩最好的前两名

```
SELECT t1.S# as 学生ID,t1.C# as 课程ID,Score as 分数
FROM SC t1
WHERE score IN (SELECT TOP 2 score
FROM SC
WHERE t1.C#= C#
ORDER BY score DESC
)
ORDER BY t1.C#;
```

44、统计每门课程的学生选修人数（超过10人的课程才统计）。要求输出课程号和选修人数，查询结果按人数降序排列，查询结果按人数降序排列，若人数相同，按课程号升序排列

```
select C# as 课程号,count(*) as 人数
from sc
group by C#
order by count(*) desc,c#
```

45、检索至少选修两门课程的学生学号

```
select S#
```



```
from sc
group by s#
having count(*) >= 2
```

46、查询全部学生都选修的课程的课程号和课程名

```
select C#,Cname
from Course
where C# in (select c# from sc group by c#)
```

47、查询没学过“叶平”老师讲授的任一门课程的学生姓名

```
select Sname from Student where S# not in (select S# from Course,Teacher,SC where Course.T#=Teacher.T# and SC.C#=course.C# and Tname='叶平');
```

48、查询两门以上不及格课程的同学的学号及其平均成绩

```
select S#,avg(isnull(score,0)) from SC where S# in (select S# from SC where score <60 group by S# having count(*)>2)group by S#;
```

49、检索“004”课程分数小于60，按分数降序排列的同学学号

```
select S# from SC where C#='004' and score <60 order by score desc;
```

50、删除“002”同学的“001”课程的成绩

```
delete from Sc where S#='001' and C#='001';
```

作者：lifetragedy 发表于2013/8/12 22:12:01 [原文链接](#)  
阅读：11879 评论：26 [查看评论](#)

think in java interview-高级开发人员面试宝典(五)

这次开始我们来点洋文吧。

有些基础，大家可能用中文知道如何表示，但是面试官如果让你用全英语表达你就不知道如何去说了，那么下面我们将给出对于一些常用的JAVA基础知识的英语问答以及相关的答案。

大家可以看一下如何用英语去回答这些基础的问题，找一下感觉。

Overriding & Overloading

Overriding - same method names with same arguments and same return types associated in a class and its subclass.

Overloading - same method name with different arguments, may or may not be same return type written in the same class itself.

What's the difference between an interface and an abstract class?

- An abstract class may contain code in method bodies, which is not allowed in an interface. With abstract classes, you have to inherit your class from it and Java does not allow multiple inheritance. On the other hand, you can implement multiple interfaces in your class.
- What’s the difference between Hashtable and Hashmap
- They Both provide key-value access to data.
- The key difference between the two is that access to the Hashtable is synchronized on the table while access to the HashMap isn't. You can add it, but it isn't there by default.
- Another difference is that iterator in the HashMap is fail-safe while the enumerator for the Hashtable isn't.
- And, a third difference is that HashMap permits null values in it, while Hashtable doesn't.

What do you understand by Synchronization?

Synchronization is a process of controlling the access of shared resources by the multiple threads in such a manner that only one thread can access one resource at a time. In non synchronized multithreaded application, it is possible for one thread to modify a shared object while another thread is in the process of using or updating the object's value. Synchronization prevents such type of data corruption.

Synchronized statement and Synchronized block

Synchronized methods are methods that are used to control access to an object. A thread only executes a synchronized method after it has acquired the lock for the method’sobject or class. Synchronized statements are similar to synchronized methods, always use synchronized(this) to instead of.

What is difference between jdbc3.0 and jdbc2.0? What’s the new features in jdbc3.0?

这道题可见老外的面试对基础是多么的重视，回答时掌握好6点

The JDBC 3.0 API introduces new material and changes in these areas:

1. Savepoint support

Added the Savepoint interface, which contains new methods to set, release, or roll back a transaction to designated savepoints.

2. Reuse of prepared statements by connection pools

Added the ability for deployers to control how prepared statements are pooled and reused by connections.

3. Connection pool configuration Defined a number of properties for the ConnectionPoolDataSource interface.

These properties can be used to describe how pooledConnection objects created by DataSource objects should be pooled.

4. Retrieval of parameter metadata

Added the interface ParameterMetaData, which describes the number, type and properties of parameters to prepared statements.

5. Retrieval of auto-generated keys

Added a means of retrieving values from columns containing automatically generated values.

6. Ability to have multiple open ResultSet objects

Added the new method getMoreResults(int), whichtakes an argument that specifies whether ResultSet objects returned by a Statement object should be closed before returning any subsequent ResultSet objects.

## What's the difference between the methods sleep() and wait()

简单了不能再简单的一道题了，可是用英语如何回答呢？来！

The code sleep(1000); puts thread aside for exactly one second. The code wait(1000), causes a wait of up to one second. A thread could stop waiting earlier if it receives the notify() or notifyAll() call. The method wait() is defined in the class Object and the method sleep() is defined in the class Thread

## What is oops

OOPs is the new concept of programming parallel to Procedure oriented programming

There are three main principals of oops which are called Polymorphism, Inheritance and Encapsulation

It help in programming approach in order to built robust user friendly and efficient softwares and provide the efficient way to maintain real world softwares..

## What is Authentication & Authorization

这道题很猛，来自于Oracle的一次面试。

Fundamentally, authentication is performed by a series of Spring Security filter (implementations of J2EE Servlet Filters) chains, linked together. Each element in a given chain has a dedicated responsibility, while each chain is responsible for accomplishing high-level goals towards the handling of a request. Ultimately, these chains must prepare a request to fulfill a single contract enforced by the last link in the primary security filter chain.

### Authentication

An authentication system is how you identify yourself to the computer. The goal behind an authentication system is to verify that the user is actually who they say they are.

There are many ways of authenticating a user. Any combination of the following are good examples.

#### Password based authentication

Requires the user to know some predetermined quantity (their password).

- Advantages: Easy to impliemnt, requires no special equipemnt.
- Disadvantages: Easy to forget password. User can tell another user their password. Password can be written down. Password can be reused.

#### Device based authentication

Requires the user to posses some item such as a key, mag strip, card, s/key device, etc.

- Advantages: Difficult to copy. Cannot forget password. If used with a PIN is near useless if stolen.
- Disadvantages: Must have device to use service so the user might forget it at home. Easy target for theft. Still doesn't actually actively identify the user.

#### Biometric Authentication

- My voice is my passport. Verify me. This is from the movie sneakers and demonstrates one type of biometric authentication device. It identifies some physical characteristic of the user that cannot be seperated from their body.

#### Retina Scanners

- Advantages: Accurately identifies the user when it works.
- Disadvantages: New technology that is still evolving. Not perfect yet.

#### Hand Scanners

- Advantages: Difficult to separate from the user. Accurately identifies the user.
- Disadvantages: Getting your hand stolen to break into a vault sucks a lot more than getting your ID card stolen.

### Authorization

Once the system knows who the user is through authentication, authorization is how the system decides what the user can do.

A good example of this is using group permissions or the difference between a normal user and the superuser on a unix system.

There are other more compicated ACL (Access Control Lists) available to decide what a user can do and how they can do it. Most unix systems don't impliment this very well (if at all.)

## What is generic type?

中文回答一下，谁都可以答得出，够简单吧，试着用英语回答呢？

When you take an element out of a Collection, you must cast it to the type of element that is stored in the collection. Besides being inconvenient, this is unsafe. The compiler does not check that your cast is the same as the collection's type, so the cast can fail at run time.

Generics provides a way for you to communicate the type of a collection to the compiler, so that it can be checked. Once the compiler knows the element type of the collection, the compiler can check that you have used the collection consistently and can insert the correct casts on values being taken out of the collection.

## What is autoboxing and unboxing?

很多人中文可能都不一定回答得出这道题，自动装箱，拆箱，忘了吧？

**Autoboxing**, introduced in Java 5, is the automatic conversion the Java compiler makes between the primitive (basic) types and their corresponding object wrapper classes (eg, int and Integer, double and Double, etc). The underlying code that is generated is the same, **but autoboxing** provides a sugar coating that avoids the tedious and hard-to-read casting typically required by Java Collections, which can not be used with primitive types.

## What is equals() and hashCode()

又来了，具体原理和算法看：[hashCode & equals之5重天](#)

Java.lang.Object has methods called hasCode() and equals(). These methods play a significant role in the real time application. However its use is not always common to all applications. In some case these methods are overridden to perform certain purpose. In this article I will explain you some concept of these methods and why it becomes necessary to override these methods.

#### hashCode()

As you know this method provides the has code of an object. Basically the default implementation of hashCode() provided by Object is derived by mapping the memory address to an integer value. If look into the source of Object class , you will find the following code for the hashCode.

```
public native int hashCode();
```

It indicates that hashCode is the native implementation which provides the memory address to a certain extent. However it is possible to override the hashCode method in your implementation class.

#### equals()

This particular method is used to make equal comparison between two objects. There are two types of comparisons in Java. One is using “= =” operator and another is “equals()”. I hope that you know the difference between this two. More specifically the “.equals()” refers to equivalence relations. So in broad sense you say that two objects are equivalent they satisfy the “equals()” condition. If you look into the source code of Object class you will find the following code for the equals() method

### 下面开始来几道J2EE的面试题

### What is the Lazy Loading

Lazy loading in Hibernate means fetching and loading the data,

only when it is needed, from a persistent storage like a database.

Lazy loading improves the performance of data fetching and significantly reduces the memory footprint.

In Hibernate, there are two main components which can be lazy loaded.

They are namely Entities and Collections.  
An Entity represents a relational data in database, whereas a collection represents collection of children for an entity.

### What is MVC

Model–view–controller (MVC) is an architecturalpattern used in softwareengineering. Successful use of the pattern isolates business logic from user interface considerations, resulting in an application where it is easier to modify either the visual appearance of theapplication or the underlying business rules without affecting the other

- Model
- Business logic goes here
- View
- Presentation logic goes here
- Controller
- Application logic goes here

### What is Spring

Springfeature:  
The Spring Framework comprisesseveral modules that provide a range of services:

- Inversion of Control container: configuration of application components and lifecycle management of Java objects
- Aspect-oriented programming: enables implementation of cross-cutting routines
- Data access: working with relational database management systems on the Java platform using JDBC and object-relational mapping tools
- Transaction management: unifies several transaction management APIs and coordinates transactions for Java objects
- Model-view-controller: an HTTP and Servlet-based framework providing hooks for extension and customization
- Remote Access framework: configurative RPC-style export and import of Java objects over networks supporting RMI, CORBA and HTTP-based protocols including web services (SOAP)
- Batch processing: a framework for high-volume processing featuring reusable functions including logging/tracing, transaction management, job processing statistics, job restart, skip, and resource management
- Authentication and authorization: configurable security processes that support a range of standards, protocols, tools and practices via the Spring Security sub-project (formerly Acegi).
- Remote Management: configurative exposure and management of Java objects for local or remote configuration via JMX
- Messaging: configurative registration of message listener objects for transparent message consumption from message queues via JMS, improvement of message sending over standard JMS APIs
- Testing: support classes for writing unit tests and integratio

### Why use hibernate

1. Hibernate generates very efficient queries very consistently  
Hibernateemploys very aggressive, and very intelligent first and second level cachingstrategy.  
Thisis a major factor in acheiving the high scalability.
2. Cross-Database You only have to change the databasedialect.

### What is dependency-injection

Software components (Clients), are often a part of a set of collaborating components which depend upon other components (Services) to successfully complete their intended purpose. In many scenarios, they need to know “which” components to communicate with, “where” to locate them, and “how” to communicate with them. When the way such services can be accessed is changed, such changes can potentially require the source of lot of clients to be changed.  
One way of structuring the code is to let the clients embed the logic of locating and/or instantiating the services as a part of their usual logic. Another way to structure the code is to have the clients declare their dependency on services, and have some "external" piece of code assume the responsibility of locating and/or instantiating the services and simply supplying the relevant service references to the clients when needed.

### What's difference between criterial query and hql:

Criteria is an object-oriented API, while HQL means string concatenation. That means all of the benefits of object-orientedness apply:  
1. All else being equal, the OO version is somewhat less prone to error. Any old string could get appended into the HQL query, whereas only valid Criteria objects can make it into a Criteria tree. Effectively, the Criteria classes are more constrained.  
2. With auto-complete, the OO is more discoverable (and thus easier to use, for me at least). You don't necessarily need to remember which parts of the query go where; the IDE can help you  
3. You also don't need to remember the particulars of the syntax (like which symbols go where). All you need to know is how to call methods and create objects.  
Since HQL is very much like SQL (which most devs know very well already) then these "don't have to remember" arguments don't carry as much weight. If HQL was more different, then this would be more importatnt.

### Pls tell me the advantage points or new features of Spring FrameWork?

The Spring Framework comprises several modules that provide a range of services:

**Inversion of Control container:** configuration of application components and lifecycle management of Java objects

**Aspect-oriented programming:** enables implementation of cross-cutting routines  
Data access: working with relational database management systems on the Java platform using JDBC and object-relational mapping tools

**Transaction management:** unifies several transaction management APIs and coordinates transactions for Java objects  
**Model-view-controller:** an HTTP and Servlet-based framework providing hooks for extension and customization

**Remote Access framework:** configurative RPC-style export and import of Java objects over networks supporting RMI, CORBA and HTTP-based protocols including web services (SOAP)

**Batch processing:** a framework for high-volume processing featuring reusable functions including logging/tracing, transaction management, job processing statistics, job restart, skip, and resource management

**Authentication and authorization:** configurable security processes that support a range of standards, protocols, tools and practices via the Spring Security sub-project (formerly Acegi).  
Remote Management: configurative exposure and management of Java objects for local or remote configuration via JMX

**Messaging:** configurative registration of message listener objects for transparent message consumption from message queues via JMS, improvement of message sending over standard JMS APIs

**Testing:** support classes for writing unit tests and integratio

**What is spring transaction managerment?**  
Spring provides a consistent abstraction for transaction management. This abstraction is one of the most important of Spring's abstractions, and delivers the following benefits:  
Provides a consistent programming model across different transaction APIs such as JTA, JDBC, Hibernate, iBATIS Database Layer and JDO.  
Provides a simpler, easier to use, API for programmatic transaction management than most of these transaction APIs  
Integrates with the Spring data access abstraction  
Supports Spring declarative transaction management

### the difference between BeanFactory and ApplicationContext ?

Two of the most fundamental and important packages in Spring are the org.springframework.beans and org.springframework.context packages. Code in these packages provides the basis for Spring's Inversion of Control (alternately called Dependency Injection) features. The BeanFactory provides an advanced configuration mechanism capable of managing beans (objects) of any nature, using potentially any kind of storage facility. The ApplicationContext builds on top of the BeanFactory (it's a subclass) and adds other functionality such as easier integration with Springs AOP features, message resource handling (for use in internationalization), event propagation, declarative mechanisms to create the ApplicationContext and optional parent contexts, and application-layer specific contexts such as the WebApplicationContext, among other enhancements.

Tell me what is AOP?

1. **Aspect**: A modularization of a concern that cuts across multiple objects. Transaction management is a good example of a crosscutting concern in J2EE applications.

2. **Join point**

3. **Advice**

4. **Pointcut**

5. **Introduction**

6. **Target object**

7. **Weaving**

How many Types of advice are there in Spring Framework:

- Before advice
- After returning advice
- After throwing advice
- After (finally) advice
- Around advice

每篇不宜写过多，先写这么多吧，下次继续，嘿嘿！！！！

作者: lifetragedy 发表于2013/8/10 17:07:31 [原文链接](#)  
阅读: 10641 评论: 14 [查看评论](#)

think in java interview-高级开发人员面试宝典(四)

算出num个数内的质数

质数即大于1的一个自然数，这个数可以被1和自身整除，如算出20之内的质数，它们有2，3，5，7，11，13，17，19这样的数字。这道题也是面试过程中笔试常问的一道题。

这道题的其目的在于：

1. 看笔试者的数学还记不记得
2. 看笔试者平时的算法

因此答题有两种。

第一种，通用做法

```
public class prime {
    public static boolean isPrime(int num) {
        for (int i = 2; i <= Math.sqrt(num); i++) {
            if (num % i == 0) {
                return false;
            }
        }
        return true;
    }

    public static void main(String[] args) {
        for (int j = 2; j <= 20; j++) {
            if (isPrime(j)) {
                System.out.println(j + " is a prime");
            }
        }
    }
}
```

这种做法是对每个2--num个数内的数，进行扫描，需要用到2个循环，其重复度高，效率低下，因此我们还有一种更先进的做法。

``筛选"法

```
public class Prime2 {

    /**
     * @param args
     */
    public static void main(String[] args) {

        int n = 20;

        int[] array = new int[n];

        for (int i = 2; i < n; i++) {

            array[i] = i;

        }

        for (int i = 2; i < n; i++) {

            if (array[i] != 0) {

                int j, temp;

                temp = array[i];

                for (j = 2 * temp; j < n; j = j + temp) {

                    array[j] = 0;

                }

                System.out.println("\n");

            }

        }

    }
}
```

它的原理就是把所有的偶数去掉后即留下质数。



## 约瑟夫环算法

假设有**20**个人手拉手围成一圈，顺时针开始报号，报到**3**的人出圈，然后继续往后报，报到**3**的人出圈，依次把所有报到**3**的人都踢出圈，最后剩下一人也踢出圈，问先后被踢出圈的那些人原来是圈内的几号？

不难不难，关键记住这个环的算法（有人看到这个“环”字，要笑了，打住，别想歪了，俗人！）

环的算法即闭合算法，永远依次**1，2，3，4，5...20...1**再**2,3,4,5...20**这样一直转下去，假设要让**20**以内的**20**个数以环型转起来，它的核心算法如下：

```
int flag=0;
while(true){
    flag=(flag+1)%n;
}
```

即“取模运算”，循环继续。

有了核心算法，我们的问题就可以解决了，来：

```
public class JosephCircle {

    /**
     * @param args
     */
    public void josephCircle(int n,int k){
        int flag=0;
        boolean[] kick = new boolean[n];
        //set kick flag to false;
        for(int i=0;i<n-1;i++){
            kick[flag]=false;
        }
        int counter=0;
        int accumulate=0;
        while(true){
            if(!kick[flag]){
                accumulate++;
                if(counter==n-1){
                    System.out.println("kick last person===="+(flag+1));
                    break;
                }
                if(accumulate==k){
                    kick[flag]=true;
                    System.out.println("kick person===="+(flag+1));
                    accumulate=0;
                    counter++;
                }
            }
            flag=(flag+1)%n;
        }
    }

    public static void main(String[] args) {
        JosephCircle jCircle = new JosephCircle();
        jCircle.josephCircle(20, 3);
    }
}
```

## 求Missed Number，找丢失的那个数

上次有朋友在美国去yahoo面试，说一道题把它给整了，题目如下：

说有一数组如 int[]{0,1,2} 这样的一个数组，这个数组的第一个必须从0开始，以次+1列出，该数组内最后一个数是这个数组的长度，因此：

int[]{1,2}, missed number为0

int[]{0,1,2}, missed number为3

int[]{0,2}, missed number为1

我朋友和我说这看上去有点像等差数列，我当时就在MSN里和他说：等差数你个头啊！

然后我朋友说，他用了两个循环嵌套也搞不定

我和他说：两个循环你个头啊

他在MSN上又要和我说什么，我还没等他把它打出来直接就是“你个头啊”回过去了。

大家看，这个找missed number是很好玩的一个东西，如果你想着用什么循环，什么算法，什么数据结构，我一律在这边回“你个头啊”，为什么，这么简单的东西，直接套公式啊，唉。。。

```
public class MissedNumber {

    public int findMissedOne(int[] numArray) {
        int sum = 0;
        int idx = -1;
        for (int i = 0; i < numArray.length; i++) {
            if (numArray[i] == 0) {
                idx = i;
            } else {
                sum += numArray[i];
            }
        }

        // the total sum of numbers between 1 and arr.length.
        int total = (numArray.length + 1) * numArray.length / 2;
        int missedNumber = total - sum;
        return missedNumber;
    }
}
```

来个测试类

```
public class TestMissedNumber {

    protected MissedNumber mNum = new MissedNumber();

    @Test
    public void testFindMissedOne() {
        int[] testArray = new int[] {0,2};
        int missedNumber = mNum.findMissedOne(testArray);
        System.out.println(missedNumber);
    }
}
```

简单吧，数组内的数的总和减去（（数组长度+1）\*数组长度/2），即为missed number

所以你就循环求一个数组内各数的和即可鸟。

## 硬币组合问题

这道题老套了，即2角，5角，1角硬币，问有多少种组合可得到1块钱（一块钱？一块钱以前能够买奶油雪糕，一块钱以前可以买一个铅笔盒，一块钱以前可以吃到大排面，现在能干吗？）

```
public class CentsCombinations {
    public static void main(String[] args) {
        int i, j, k, total;
        System.out.println("1 cent+" 2 cents+" 5 cents");
        for (i = 0; i <= 10; i++)
            for (j = 0; j <= 5; j++)
                for (k = 0; k <= 2; k++) {
                    total = i * 1 + j * 2 + k * 5;
                    if (total > 10)
                        break;
                    if (10 == total)
                        System.out.println("    " + i + ",    " + j + ",    " + k);
                }
            }
    }
```

## 在List内去除重复值的问题

说有一个ArrayList{1,2,3,5,1,2,7,5,3,3,4,5,6,9,11,11}，它里面有许多重复数，我要求去除所有的重复数据，得到一个不重复的List。

有的人碰到这个问题，循环2个，嵌套，还有用二分算法的（这种人已经算好的了）。

要什么循环啊？

这道题考的是你对JDK的API是否熟悉，来看下面答案，如果以前没碰到过这个问题的人看完下面的答案后直接就吐血了

```
public class DuplicateValue {

    public void removeDuplicateValue() {
        List myList = new ArrayList();
        myList.add(1);
        myList.add(2);
        myList.add(1);
        myList.add(3);
        myList.add(4);
        myList.add(5);
        myList.add(6);
        myList.add(5);

        Set myset = new HashSet(myList);
        myList = new ArrayList(myset);
        Iterator it = myList.iterator();
        while (it.hasNext()) {
            System.out.println(""+it.next());
        }
    }

    public static void main(String[] args) {
        DuplicateValue dv = new DuplicateValue();
        dv.removeDuplicateValue();
    }
}
```

## 一道题即考了冒泡又考了二分

说有一数组int[] {1,2,2,4,6,8,9,5,3,7,5}，里面可能还有很多，想要知道“两两相加等于10”的这样的数有几对，可以重复如：2+6, 6+2

上手就是冒泡

```
public Vector<SumPossibleBean> checkSumPossible(ArrayList<Integer> intList,
        int sumResult) {
    Vector<SumPossibleBean> possibleList = new Vector<SumPossibleBean>();
    int size = intList.size();
    for (int i = 0; i < size; i++) {
        for (int j = i + 1; j < size; j++) {
            if ((intList.get(i).intValue() + intList.get(j).intValue()) == sumResult) {

                SumPossibleBean spBean = new SumPossibleBean();
                spBean.setFirst(intList.get(i).intValue());
                spBean.setSecond(intList.get(j).intValue());
                possibleList.addElement(spBean);
            }
        }
    }
    return possibleList;
}
```

不错，两个循环嵌套，当考官问你，有没有更好的算法时，70%以上的人基本歇菜！！！

唉哎。。。二分来一个吗，以前在学校怎么学的？

```
public Vector<SumPossibleBean> checkSumPossibleBinSearch(
        ArrayList<Integer> intList, int sumResult) {
    Vector<SumPossibleBean> possibleList = new Vector<SumPossibleBean>();
    int size = intList.size();
    Collections.sort(intList);
    for (int i = 0, j = (intList.size() - 1); i < j;) {
        Integer s = intList.get(i);
        Integer e = intList.get(j);
        if ((s + e) == sumResult) {
            SumPossibleBean spBean = new SumPossibleBean();
            spBean.setFirst(intList.get(i).intValue());
            spBean.setSecond(intList.get(j).intValue());
            possibleList.addElement(spBean);
            i++;
            j--;
        }
    }
    return possibleList;
}
```



```
        } else if ((s + e) > sumResult) {
            j--;
        } else if ((s + e) < sumResult) {
            i++;
        }
    }
    return possibleList;
}
```

算法讲一些，讲多了枯燥，来几道问答题，如果你没准备过的话也一样可以让你爽到吐血。

## Collection 和 Collections的区别

**Collections**是个**java.util**下的类，它包含有各种有关集合操作的静态方法。  
**Collection**是个**java.util**下的接口，它是各种集合结构的父接口。

吐血了吗？没吐，OK，下面这道如果在面试时问到一定让你吐，而架构师的人**50%**以上被问到过这道题

## 什么时候用assert

断言是一个包含布尔表达式的语句，在执行这个语句时假定该表达式为 **true**。如果表达式计算为 **false**，那么系统会报告一个 **AssertionError**。它用于调试目的：

```
assert(a > 0); // throws an AssertionError if a <= 0
```

断言可以有两种形式：

```
assert Expression1 ;
assert Expression1 : Expression2 ;
```

**Expression1** 应该总是产生一个布尔值。  
**Expression2** 可以是得出一个值的任意表达式。这个值用于生成显示更多调试信息的 **String** 消息。

断言在默认情况下是禁用的。要在编译时启用断言，需要使用 **source 1.4** 标记：

```
javac -source 1.4 Test.java
```

要在运行时启用断言，可使用 **-enableassertions** 或者 **-ea** 标记。

要在运行时选择禁用断言，可使用 **-da** 或者 **-disableassertions** 标记。

要系统类中启用断言，可使用 **-esa** 或者 **-dsa** 标记。还可以在包的基础上启用或者禁用断言。

可以在预计正常情况下不会到达的任何位置上放置断言。断言可以用于验证传递给私有方法的参数。不过，断言不应该用于验证传递给公有方法的参数，因为不管是否启用了断言，公有方法都必须检查其参数。不过，既可以在公有方法中，也可以在非公有方法中利用断言测试后置条件。另外，断言不应该以任何方式改变程序的状态。

## 企业级开发写多了，忘了**JAVA**是一门计算机语言，计算机语言最早是用来做运算的

来一道

问：**Math.round(11.5)**等於多少？**Math.round(-11.5)**等於多少

**Math.round(11.5)**返回（long）12

**Math.round(-11.5)**返回（long）-11；

再来一道

问：**short s1 = 1; s1 = s1+ 1;**有什么错？**short s1 = 1; s1 += 1;**有什么错

**short s1 = 1; s1 = s1 + 1;**有错，s1是short型，s1+1是int型,不能显式转化为short型。可修改为**s1 =(short)(s1 + 1)**。**short s1 = 1; s1 += 1**正确。

## 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。

## **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。

## **abstract**的**method**是否可同时是**static**,是否可同时是**native**，是否可同时是**synchronized**？

都不能

## 可变类与不可变类的区别

所谓不可变类：

是指当创建了这个类的实例后，就不允许修改它的属性值。在JDK的基本类库中，所有基本类型的包装类，如Integer和Long类，都是不可变类，java.lang.String也是不可变类。

不可变类：

当你获得这个类的一个实例引用时，你不可以改变这个实例的内容。不可变类的实例一旦创建，其内在成员变量的值就不能被修改。

如何创建一个不可变类？

这道题90%以上的人都会被挂或者只回答一半对，来看：

- 1. 所有成员都是private
- 2. 不提供对成员的改变方法，例如：setXXXX
- 3. 确保所有的方法不会被重载。手段有两种：使用final Class(强不可变类)，或者将所有类方法加上final(弱不可变类)。
- 4. 如果某一个类成员不是原始变量(primitive)或者不可变类，必须通过在成员初始化(in)或者get方法(out)时通过深度clone方法，来确保类的不可变。

四点中少一点，回答失败，来看一个例子：

```
public final class MyImmutableWrong {
    private final int[] myArray;

    public MyImmutableWrong(int[] anArray) {
        this.myArray = anArray; // wrong
    }

    public String toString() {
        StringBuffer sb = new StringBuffer("Numbers are: ");
        for (int i = 0; i < myArray.length; i++) {
            sb.append(myArray[i] + " ");
        }
        return sb.toString();
    }
}
```

以上不可变类书写错误，为什么？ 违反了第四点

正确的写法：

```
public final class MyImmutableCorrect {
    private final int[] myArray;

    public MyImmutableCorrect(int[] anArray) {
        this.myArray = anArray.clone();
    }

    public String toString() {
        StringBuffer sb = new StringBuffer("Numbers are: ");
        for (int i = 0; i < myArray.length; i++) {
            sb.append(myArray[i] + " ");
        }
        return sb.toString();
    }
}
```

记住，不可变类的写法，我说过90%的人碰到这个问题会挂，另10%回答出。

但 是这10%里90%的人只回答出第1点，第2点和第3点，对于第四点即：

如果某一个类成员不是原始变量(primitive)或者不可变类，必须通过在成员初始化(in)或者get方法(out)时通过深度clone方法，来确保类的不可变。

则都没回答上来。

由此可判断这个面试者是真正写过还是只是为了应付面试而背原理。

所以，不要把面试当儿戏，每一次面试可能就是对自己这段时间的一个检查，看看自己平时缺了什么，JAVA是一门体系化的语言，要想面试的好，不只是死记硬背，是真的平时要化时间自己去补一些基础的。

上手就学SSH，JSP能连个数据库满天飞，这就是JAVA了？这就是中国IT了。。。这是错误的。

今天暂写这么多，下次继续。

作者：lifetragedy 发表于2013/8/7 11:00:06 原文链接  
阅读：12343 评论：32 查看评论

think in java interview-高级开发人员面试宝典(三)

收集自Oracle公司的10次（60道）电话面试全部问答（英语）

Q: What environment variables do I need to set on my machine in order to be able to run Java programs?  
A: CLASSPATH and PATH are the two variables.

Q: Can an application have multiple classes having main method?  
A: Yes it is possible. While starting the application we mention the class name to be run. The JVM will look for the Main method only in the class whose name you have mentioned. Hence there is not conflict amongst the multiple classes having main method.

Q: Can I have multiple main methods in the same class?  
A: No the program fails to compile. The compiler says that the main method is already defined in the class.

Q: Do I need to import java.lang package any time? Why ?  
A: No. It is by default loaded internally by the JVM.

Q Can I import same package/class twice? Will the JVM load the package twice at runtime?  
A: One can import the same package or same class multiple times. Neither compiler nor JVM complains abt it. And the JVM will internally load the class only once no matter how many times you import the same class.

Q: What are Checked and UnChecked Exception?  
A: A checked exception is some subclass of Exception (or Exception itself), excluding class RuntimeException and its subclasses. Making an exception checked forces client programmers to deal with the possibility that the exception will be thrown. eg, IOException thrown by java.io.FileInputStream's read() method? Unchecked exceptions are RuntimeException and any of its subclasses. Class Error and its subclasses also are unchecked. With an unchecked exception, however, the compiler doesn't force client programmers either to catch the exception or declare it in a throws clause. In fact, client programmers may not even know that the exception could be thrown. eg, StringIndexOutOfBoundsException thrown by String's charAt() method? Checked exceptions must be caught at compile time. Runtime exceptions do not need to be. Errors often cannot be.

Q: What is Overriding?  
A: When a class defines a method using the same name, return type, and arguments as a method in its superclass, the method in the class overrides the method in the superclass. When the method is invoked for an object of the class, it is the new definition of the method that is called, and not the method definition from superclass. Methods may be overridden to be more public, not more private.

Q: What are different types of inner classes?  
A: Nested top-level classes, Member classes, Local classes, Anonymous classes

Nested top-level classes- If you declare a class within a class and specify the static modifier, the compiler treats the class just like any other top-level class.

Any class outside the declaring class accesses the nested class with the declaring class name acting similarly to a package. eg, outer.inner. Top-level inner classes implicitly have access only to static variables. There can also be inner interfaces. All of these are of the nested top-level variety.

Member classes - Member inner classes are just like other member methods and member variables and access to the member class is restricted, just like methods and variables. This means a public member class acts similarly to a nested top-level class. The primary difference between member classes and nested top-level classes is that member classes have access to the specific instance of the enclosing class.

Local classes - Local classes are like local variables, specific to a block of code. Their visibility is only within the block of their declaration. In order for the class to be useful beyond the declaration block, it would need to implement a more publicly available interface. Because local classes are not members, the modifiers public, protected, private, and static are not usable.

Anonymous classes - Anonymous inner classes extend local inner classes one level further. As anonymous classes have no name, you cannot provide a constructor.

Q: Are the imports checked for validity at compile time? e.g. will the code containing an import such as java.lang.ABCD compile?

A: Yes the imports are checked for the semantic validity at compile time. The code containing above line of import will not compile. It will throw an error saying, can not resolve symbol  
symbol : class ABCD  
location: package io  
import java.io.ABCD;

Q: Does importing a package import the subpackages as well? e.g. Does importing com.MyTest.\* also import com.MyTest.UnitTests.\*?

A: No you will have to import the subpackages explicitly. Importing com.MyTest.\* will import classes in the package MyTest only. It will not import any class in any of its subpackage.

Q: What is the difference between declaring a variable and defining a variable?

A: In declaration we just mention the type of the variable and its name. We do not initialize it. But defining means declaration + initialization.  
e.g String s; is just a declaration while String s = new String ("abcd"); Or String s = "abcd"; are both definitions.

Q: What is the default value of an object reference declared as an instance variable?

A: null unless we define it explicitly.

Q: Can a top level class be private or protected?

A: No. A top level class can not be private or protected. It can have either "public" or no modifier. If it does not have a modifier it is supposed to have a default access. If a top level class is declared as private the compiler will complain that the "modifier private is not allowed here". This means that a top level class can not be private. Same is the case with protected.

Q: What type of parameter passing does Java support?

A: In Java the arguments are always passed by value .

Q: Primitive data types are passed by reference or pass by value?

A: Primitive data types are passed by value.

Q: Objects are passed by value or by reference?

A: Java only supports pass by value. With objects, the object reference itself is passed by value and so both the original reference and parameter copy both refer to the same object .

Q: What is serialization?

A: Serialization is a mechanism by which you can save the state of an object by converting it to a byte stream.

Q: How do I serialize an object to a file?

A: The class whose instances are to be serialized should implement an interface Serializable. Then you pass the instance to the ObjectOutputStream which is connected to a FileOutputStream. This will save the object to a file.

Q: Which methods of Serializable interface should I implement?

A: The Serializable interface is an empty interface, it does not contain any methods. So we do not implement any methods.

Q: How can I customize the serialization process? i.e. how can one have a control over the serialization process?

A: Yes it is possible to have control over serialization process. The class should implement Externalizable interface. This interface contains two methods namely readExternal and writeExternal. You should implement these methods and write the logic for customizing the serialization process.

Q: What is the common usage of serialization?

A: Whenever an object is to be sent over the network, objects need to be serialized. Moreover if the state of an object is to be saved, objects need to be serialized.

Q: What is Externalizable interface?

A: Externalizable is an interface which contains two methods readExternal and writeExternal. These methods give you a control over the serialization mechanism. Thus if your class implements this interface, you can customize the serialization process by implementing these methods.

Q: When you serialize an object, what happens to the object references included in the object?

A: The serialization mechanism generates an object graph for serialization. Thus it determines whether the included object references are serializable or not. This is a recursive process. Thus when an object is serialized, all the included objects are also serialized along with the original object.

Q: What one should take care of while serializing the object?

A: One should make sure that all the included objects are also serializable. If any of the objects is not serializable then it throws a NotSerializableException.

Q: What happens to the static fields of a class during serialization?

A: There are three exceptions in which serialization does not necessarily read and write to the stream. These are  
1. Serialization ignores static fields, because they are not part of any particular state.  
2. Base class fields are only handled if the base class itself is serializable.  
3. Transient fields.

Q: Does Java provide any construct to find out the size of an object?

A: No there is no sizeof operator in Java. So there is no direct way to determine the size of an object directly in Java.

Q: Give a simplest way to find out the time a method takes for execution without using any profiling tool?

A: Read the system time just before the method is invoked and immediately after method returns. Take the time difference, which will give you the time taken by a method for execution.

To put it in code...

```
long start = System.currentTimeMillis ();  
method ();  
long end = System.currentTimeMillis ();
```

```
System.out.println ("Time taken for execution is " + (end - start));
```

Remember that if the time taken for execution is too small, it might show that it is taking zero milliseconds for execution. Try it on a method which is big enough, in the sense the one which is doing considerable amount of processing.

Q: What are wrapper classes?

A: Java provides specialized classes corresponding to each of the primitive data types. These are called wrapper classes. They are e.g. Integer, Character, Double etc.

Q: Why do we need wrapper classes?

A: It is sometimes easier to deal with primitives as objects. Moreover most of the collection classes store objects and not primitive data types. And also the wrapper classes provide many utility methods also. Because of these reasons we need wrapper classes. And since we create instances of these classes we can store them in any of the collection classes and pass them around as a collection. Also we can pass them around as method parameters where a method expects an object.

Q: What are checked exceptions?

A: Checked exceptions are those which the Java compiler forces you to catch. e.g. IOException are checked Exceptions.

Q: What are runtime exceptions?



A: Runtime exceptions are those exceptions that are thrown at runtime because of either wrong input data or because of wrong business logic etc. These are not checked by the compiler at compile time.

Q: What is the difference between error and an exception?

A: An error is an irrecoverable condition occurring at runtime. Such as OutOfMemory error. These JVM errors and you can not repair them at runtime. While exceptions are conditions that occur because of bad input etc. e.g. FileNotFoundException will be thrown if the specified file does not exist. Or a NullPointerException will take place if you try using a null reference. In most of the cases it is possible to recover from an exception (probably by giving user a feedback for entering proper values etc.).

Q: How to create custom exceptions?

A: Your class should extend class Exception, or some more specific type thereof.

Q: If I want an object of my class to be thrown as an exception object, what should I do?

A: The class should extend from Exception class. Or you can extend your class from some more precise exception type also.

Q: If my class already extends from some other class what should I do if I want an instance of my class to be thrown as an exception object?

A: One can not do anything in this scenario. Because Java does not allow multiple inheritance and does not provide any exception interface as well.

Q: How does an exception permeate through the code?

A: An unhandled exception moves up the method stack in search of a matching When an exception is thrown from a code which is wrapped in a try block followed by one or more catch blocks, a search is made for matching catch block. If a matching type is found then that block will be invoked. If a matching type is not found then the exception moves up the method stack and reaches the caller method. Same procedure is repeated if the caller method is included in a try catch block. This process continues until a catch block handling the appropriate type of exception is found. If it does not find such a block then finally the program terminates.

Q: What are the different ways to handle exceptions?

A: There are two ways to handle exceptions,

1. By wrapping the desired code in a try block followed by a catch block to catch the exceptions. and
2. List the desired exceptions in the throws clause of the method and let the caller of the method hadle those exceptions.

Q: What is the basic difference between the 2 approaches to exception handling.

1> try catch block and

2> specifying the candidate exceptions in the throws clause?

When should you use which approach?

A: In the first approach as a programmer of the method, you urself are dealing with the exception. This is fine if you are in a best position to decide should be done in case of an exception. Whereas if it is not the responsibility of the method to deal with it's own exceptions, then do not use this approach. In this case use the second approach. In the second approach we are forcing the caller of the method to catch the exceptions, that the method is likely to throw. This is often the approach library creators use. They list the exception in the throws clause and we must catch them. You will find the same approach throughout the java libraries we use.

Q: Is it necessary that each try block must be followed by a catch block?

A: It is not necessary that each try block must be followed by a catch block. It should be followed by either a catch block OR a finally block. And whatever exceptions are likely to be thrown should be declared in the throws clause of the method.

Q: If I write return at the end of the try block, will the finally block still execute?

A: Yes even if you write return as the last statement in the try block and no exception occurs, the finally block will execute. The finally block will execute and then the control return.

Q: If I write System.exit (0); at the end of the try block, will the finally block still execute?

A: No in this case the finally block will not execute because when you say System.exit (0); the control immediately goes out of the program, and thus finally never executes.

Q: How are Observer and Observable used?

A: Objects that subclass the Observable class maintain a list of observers. When an Observable object is updated it invokes the update() method of each of its observers to notify the observers that it has changed state. The Observer interface is implemented by objects that observe Observable objects.

Q: What is synchronization and why is it important?

A: With respect to multithreading, synchronization is the capability to control the access of multiple threads to shared resources. Without synchronization, it is possible for one thread to modify a shared object while another thread is in the process of using or updating that object's value. This often leads to significant errors.

Q: How does Java handle integer overflows and underflows?

A: It uses those low order bytes of the result that can fit into the size of the type allowed by the operation.

Q: Does garbage collection guarantee that a program will not run out of memory?

A: Garbage collection does not guarantee that a program will not run out of memory. It is possible for programs to use up memory resources faster than they are garbage collected. It is also possible for programs to create objects that are not subject to garbage collection

Q: What is the difference between preemptive scheduling and time slicing?

A: Under preemptive scheduling, the highest priority task executes until it enters the waiting or dead states or a higher priority task comes into existence. Under time slicing, a task executes for a predefined slice of time and then reenters the pool of ready tasks. The scheduler then determines which task should execute next, based on priority and other factors.

Q: When a thread is created and started, what is its initial state?

A: A thread is in the ready state after it has been created and started.

Q: What is the purpose of finalization?

A: The purpose of finalization is to give an unreachable object the opportunity to perform any cleanup processing before the object is garbage collected.

Q: What is the Locale class?

A: The Locale class is used to tailor program output to the conventions of a particular geographic, political, or cultural region.

Q: What is the difference between a while statement and a do statement?

A: A while statement checks at the beginning of a loop to see whether the next loop iteration should occur. A do statement checks at the end of a loop to see whether the next iteration of a loop should occur. The do statement will always execute the body of a loop at least once.

Q: What is the difference between static and non-static variables?

A: A static variable is associated with the class as a whole rather than with specific instances of a class. Non-static variables take on unique values with each object instance.

Q: How are this() and super() used with constructors?

A: This() is used to invoke a constructor of the same class. super() is used to invoke a superclass constructor.

Q: What are synchronized methods and synchronized statements?

A: Synchronized methods are methods that are used to control access to an object. A thread only executes a synchronized method after it has acquired the lock for the method's object or class. Synchronized statements are similar to synchronized methods. A synchronized statement can only be executed after a thread has acquired the lock for the object or class referenced in the synchronized statement.

Q: What is daemon thread and which method is used to create the daemon thread?

A: Daemon thread is a low priority thread which runs intermittently in the back ground doing the garbage collection operation for the java runtime system. setDaemon method is used to create a daemon thread.

Q: Can applets communicate with each other?

A: At this point in time applets may communicate with other applets running in the same virtual machine. If the applets are of the same class, they can communicate via shared static variables. If the applets are of different classes, then each will need a reference to the same class with static variables. In any case the basic idea is to pass the information back and forth through a static variable.

An applet can also get references to all other applets on the same page using the getApplets() method of java.applet.AppletContext. Once you get the reference to an applet, you can communicate with it by using its public members.

It is conceivable to have applets in different virtual machines that talk to a server somewhere on the Internet and store any data that needs to be serialized there. Then, when another applet needs this data, it could connect to this same server. Implementing this is non-trivial.

Q: What are the steps in the JDBC connection?

A: While making a JDBC connection we go through the following steps :

Step 1 : Register the database driver by using :

Class.forName(\" driver classss for that specific database\");

Step 2 : Now create a database connection using :

```
Connection con = DriverManager.getConnection(url,username,password);
```

Step 3: Now Create a query using :

```
Statement stmt = Connection.Statement("\select * from TABLE NAME\");
```

Step 4 : Exceute the query :

```
stmt.exceuteUpdate();
```

Q: How does a try statement determine which catch clause should be used to handle an exception?

A: When an exception is thrown within the body of a try statement, the catch clauses of the try statement are examined in the order in which they appear. The first catch clause that is capable of handling the exceptionis executed. The remaining catch clauses are ignored.

Q: Can an unreachable object become reachable again?

A: An unreachable object may become reachable again. This can happen when the object's finalize() method is invoked and the object performs an operation which causes it to become accessible to reachable objects.

Q: What method must be implemented by all threads?

A: All tasks must implement the run() method, whether they are a subclass of Thread or implement the Runnable interface.

Q: What are synchronized methods and synchronized statements?

A: Synchronized methods are methods that are used to control access to an object. A thread only executes a synchronized method after it has acquired the lock for the method's object or class. Synchronized statements are similar to synchronized methods. A synchronized statement can only be executed after a thread has acquired the lock for the object or class referenced in the synchronized statement.

Q: What is Externalizable?

A: Externalizable is an Interface that extends Serializable Interface. And sends data into Streams in Compressed Format. It has two methods, writeExternal(ObjectOuput out) and readExternal(ObjectInput in)

Q: What modifiers are allowed for methods in an Interface?

A: Only public and abstract modifiers are allowed for methods in interfaces.

Q: What are some alternatives to inheritance?

A: Delegation is an alternative to inheritance. Delegation means that you include an instance of another class as an instance variable, and forward messages to the instance. It is often safer than inheritance because it forces you to think about each message you forward, because the instance is of a known class, rather than a new class, and because it doesn't force you to accept all the methods of the super class: you can provide only the methods that really make sense. On the other hand, it makes you write more code, and it is harder to re-use (because it is not a subclass).

Q: What does it mean that a method or field is "static"?

A: Static variables and methods are instantiated only once per class. In other words they are class variables, not instance variables. If you change the value of a static variable in a particular object, the value of that variable changes for all instances of that class. Static methods can be referenced with the name of the class rather than the name of a particular object of the class (though that works too). That's how library methods like System.out.println() work out is a static field in the

Q: What is the difference between preemptive scheduling and time slicing?

A: Under preemptive scheduling, the highest priority task executes until it enters the waiting or dead states or a higher priority task comes into existence. Under time slicing, a task executes for a predefined slice of time and then reenters the pool of ready tasks. The scheduler then determines which task should execute next, based on priority and other factors.

Q: What is the catch or declare rule for method declarations?

A: If a checked exception may be thrown within the body of a method, the method must either catch the exception or declare it in its throws clause.

作者: lifetragedy 发表于2013/8/5 13:13:42 [原文链接](#)

阅读: 13338 评论: 13 [查看评论](#)

## think in java interview-高级开发人员面试宝典(二)

从现在开始，以样题的方式一一列出各种面试题以及点评，考虑到我在前文中说的，对于一些大型的外资型公司，你将会面临全程英语面试，因此我在文章中也会出现许多全英语样题。

这些题目来自于各个真实的公司，公司名我就不一一例举了，是本人一直以来苦心收藏的。

## 一个JAVA 的MAIN方法引发的一场血案

Q: What if the main method is declared as private?

A: The program compiles properly but at run time it will give "Main method not public." message.

Q: What if the static modifier is removed from the signature of the main method?

A: Program compiles. But at run time throws an error "NoSuchMethodError".

Q: What if I write static public void instead of public static void?

A: Program compiles and runs properly.

Q: What if I do not provide the String array as the argument to the method?

A: Program compiles but throws a run time error "NoSuchMethodError".

Q: What is the first argument of the String array in main method?

A: The String array is empty. It does not have any element. This is unlike C/C++（读作plus plus） where the first element by default is the program name.

Q: If I do not provide any arguments on the command line, then the String array of Main method will be empty or null?

A: It is empty. But not null.

Q: How can one prove that the array is notnull but empty using one line of code?

A: Print args.length. It will print 0. That means it is empty. But if it would have been null then it would have thrown a NullPointerException on attempting to print args.length.

仔细看完后有人直接吐血了，拿个eclipse，这几个问题全部模拟一边就可以了，即无算法也不需要死记硬背

有人会说了，唉，我怎么写了5年的JAVA怎么就没记得多写多看，多想想这个public static void main(String[] args)方法呢？唉。。。

再来！！！

## hashcode & equals之5重天

## 何时需要重写equals()

当一个类有自己独特的“逻辑相等”概念（不同于对象身份的概念）。

## 如何覆写equals()和hashCode

### 覆写equals方法

- 1 使用instanceof操作符检查“实参是否为正确的类型”。
- 2 对于类中的每一个“关键域”，检查实参中的域与当前对象中对应的域值。
3. 对于非float和double类型的原语类型域，使用==比较；
- 4 对于对象引用域，递归调用equals方法；
- 5 对于float域，使用Float.floatToIntBits(afloat)转换为int，再使用==比较；
- 6 对于double域，使用Double.doubleToLongBits(adouble)转换为int，再使用==比较；
- 7 对于数组域，调用Arrays.equals方法。

### 覆写hashCode

1. 把某个非零常数值，例如17，保存在int变量result中；
2. 对于对象中每一个关键域f（指equals方法中考虑的每一个域）：
- 3, boolean型，计算(f? 0 : 1)；
4. byte,char,short型，计算(int)；
5. long型，计算(int)(f ^ (f>>>32))；
6. float型，计算Float.floatToIntBits(afloat)；
7. double型，计算Double.doubleToLongBits(adouble)得到一个long，再执行[2.3]；
8. 对象引用，递归调用它的hashCode方法；
9. 数组域，对其中每个元素调用它的hashCode方法。
10. 将上面计算得到的散列码保存到int变量c，然后执行result=37\*result+c；
11. 返回result。

举个例子：

```
public class MyUnit {
    private short ashort;
    private char achar;
    private byte abyte;
    private boolean abool;
    private long along;
    private float afloat;
    private double adouble;
    private Unit aObject;
    private int[] ints;
    private Unit[] units;

    public boolean equals(Object o) {
        if (!(o instanceof MyUnit))
            return false;
        MyUnit unit = (MyUnit) o;
        return unit.ashort == ashort
            && unit.achar == achar
            && unit.abyte == abyte
            && unit.abool == abool
            && unit.along == along
            && Float.floatToIntBits(unit.afloat) == Float
                .floatToIntBits(afloat)
            && Double.doubleToLongBits(unit.adouble) == Double
                .doubleToLongBits(adouble)
            && unit.aObject.equals(aObject)
            && equalsInts(unit.ints)
            && equalsUnits(unit.units);
    }

    private boolean equalsInts(int[] aints) {
        return Arrays.equals(ints, aints);
    }

    private boolean equalsUnits(Unit[] aUnits) {
        return Arrays.equals(units, aUnits);
    }

    public int hashCode() {
        int result = 17;
        result = 31 * result + (int) ashort;
        result = 31 * result + (int) achar;
        result = 31 * result + (int) abyte;
        result = 31 * result + (abool ? 0 : 1);
        result = 31 * result + (int) (along ^ (along >>> 32));
        result = 31 * result + Float.floatToIntBits(afloat);
        long tolong = Double.doubleToLongBits(adouble);
        result = 31 * result + (int) (tolong ^ (tolong >>> 32));
        result = 31 * result + aObject.hashCode();
        result = 31 * result + intsHashCode(ints);
        result = 31 * result + unitsHashCode(units);
        return result;
    }

    private int intsHashCode(int[] aints) {
        int result = 17;
        for (int i = 0; i < aints.length; i++)
            result = 31 * result + aints[i];
        return result;
    }

    private int unitsHashCode(Unit[] aUnits) {
        int result = 17;
        for (int i = 0; i < aUnits.length; i++)
            result = 31 * result + aUnits[i].hashCode();
        return result;
    }
}
```

当改写equals()的时候，总是要改写hashCode()

根据一个类的equals方法（改写后），两个截然不同的实例有可能在逻辑上是相等的，但是，根据Object.hashCode方法，它们仅仅是两个对象。因此，违反了“相等的对象必须具有相等的散列码”。

两个对象如果equals那么这两个对象的hashCode一定相等，如果两个对象的hashCode相等那么这两个对象是否一定equals?



回答是不一定，这要看这两个对象有没有重写Object的hashCode方法和equals方法。如果没有重写，是按Object默认的方式去处理。

试想我有一个桶，这个桶就是hashCode，桶里装的是西瓜我们认为西瓜就是object，有的桶是一个桶装一个西瓜，有的桶是一个桶装多个西瓜。

比如String重写了Object的hashCode和equals，但是两个String如果hashCode相等，那么equals比较肯定是相等的，但是“==”比较却不一定相等。如果自定义的对象重写了hashCode方法，有可能hashCode相等，equals却不一定相等，“==”比较也不一定相等。

此处考的是你对object的hashCode的意义的真正的理解！！如果作为一名高级开发人员或者是架构师，必须是要有这个概念的，否则，直接ban掉了。

为什么我们在定义hashCode时如： h = 31\*h + val[off++]; 要使用31这个数呢？

```
public int hashCode() {
    int h = hash;
    int len = count;
    if (h == 0 && len > 0) {
        int off = offset;
        char val[] = value;
        for (int i = 0; i < len; i++) {
            h = 31*h + val[off++];
        }
        hash = h;
    }
    return h;
}
```

来看一段hashCode的覆写案例：

其实上面的实现也可以总结成数数里面下面这样的公式：

$$s[0]*31^{(n-1)} + s[1]*31^{(n-2)} + \cdots + s[n-1]$$

我们来看这个要命的31这个系数为什么总是在里面乘啊乘的？为什么不适用32或者其他数字？

大家都知道，计算机的乘法涉及到移位计算。当一个数乘以2时，就直接拿该数左移一位即可！选择31原因是因为31是一个素数！

所谓素数：

质数又称素数

素数在使用的时候有一个作用就是如果我用一个数字来乘以这个素数，那么最终的出来的结果只能被素数本身和被乘数还有1来整除！如：我们选择素数3来做系数，那么3\*n只能被3和n或者1来整除，我们可以很容易的通过3n来计算出这个n来。这应该也是一个原因！

在存储数据计算hash地址的时候，我们希望尽量减少有同样的hash地址，所谓“冲突”。

31是个神奇的数字，因为任何数n \* 31就可以被JVM优化为 (n << 5) -n, 移位和减法的操作效率要比乘法的操作效率高得多，对左移现在很多虚拟机里面都有做相关优化，并且31只占用5bits！

hashCode & equals基本问到第三问，很多人就已经挂了，如果问到了为什么要使用31比较好呢，90%的人无法回答或者只回答出一半。

此处考的是编程者在平时开发时是否经常考虑“性能”这个问题。

## comparable接口与comparator

两种比较接口分析

前者应该比较固定，和一个具体类相绑定，而后者比较灵活，它可以被用于各个需要比较功能的类使用。

一个类实现了 Comparable 接口表明这个类的对象之间是可以相互比较的。如果用数学语言描述的话就是这个类的对象组成的集合中存在一个全序。这样，这个类对象组成的集合就可以使用 Sort 方法排序了。

- 而 Comparator 的作用有两个：
1. 如果类的设计师没有考虑到 Compare 的问题而没有实现 Comparable 接口，可以通过 Comparator 来实现比较算法进行排序；
  2. 为了使用不同的排序标准做准备，比如：升序、降序或其他什么序。

在 “集合框架” 中有两种比较接口： Comparable 接口和 Comparator 接口。

Comparable 是通用的接口，用户可以实现它来完成自己特定的比较，而 Comparator 可以看成一种算法的实现，在需要容器集合实现比较功能的时候，来指定这个比较器，这可以看成一种设计模式，将算法和数据分离。

来看样例：

PersonBean类

```
public class PersonBean implements Comparable<PersonBean> {
    public PersonBean(int age, String name) {
        this.age = age;
        this.name = name;
    }

    int age = 0;
    String name = "";

    public int getAge() {
        return age;
    }

    public void setAge(int age) {
        this.age = age;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public boolean equals(Object o) {
        if (!(o instanceof PersonBean)) {
            return false;
        }
        PersonBean p = (PersonBean) o;
        return (age == p.age) && (name.equals(p.name));
    }

    public int hashCode() {
        int result = 17;
        result = 31 * result + age;
        result = 31 * result + name.hashCode();
        return result;
    }

    public String toString() {
        return (age + " [" + name + "]");
    }

    public int compareTo(PersonBean person) {
        int cop = age - person.getAge();
        if (cop != 0)
            return cop;
        else
            return name.compareTo(person.name);
    }
}
```

```
}

AlphDesc类

import java.util.Comparator;

public class AlphDesc implements Comparator<PersonBean> {

    public int compare(PersonBean personA, PersonBean personB) {
        int cop = personA.age - personB.age;
        if (cop != 0)
            return cop;
        else
            return personB.getName().compareTo(personA.getName());
    }

}
```

TestComparable类

```
import java.util.*;

public class TestComparable {

    /**
     * @param args
     */
    public void compare() {
        PersonBean[] p = { new PersonBean(20, "Tom"),
                            new PersonBean(20, "Jeff"),
                            new PersonBean(30, "Mary"),
                            new PersonBean(20, "Ada"),
                            new PersonBean(40, "Walton"),
                            new PersonBean(61, "Peter"),
                            new PersonBean(20, "Bush") };
        System.out.println("before sort:\n" + Arrays.toString(p));
        AlphDesc desc = new AlphDesc();
        Arrays.sort(p, desc);
        System.out.println("after sort:\n" + Arrays.toString(p));
    }

    public static void main(String[] args) {
        TestComparable tc = new TestComparable();
        tc.compare();
    }

}
```

java实现shallow clone(浅克隆) 与深克隆(deep clone)

克隆就是复制一个对象的副本, 但一个对象中可能有基本数据类型, 如:int, long, float 等, 也同时含有非基本数据类型如(数组, 集合等)被克隆得到的对象基本类型的值修改了, 原对象的值不会改变. 这种适合shadow clone(浅克隆). 但如果你要改变一个非基本类型的值时, 原对象的值却改变了,. 比如一个数组, 内存中只copy他的地址, 而这个地址指向的值并没有 copy. 当clone时, 两个地址指向了一个值, 这样一旦这个值改变了, 原来的值当然也变了, 因为他们共用一个值., 这就必须得用深克隆(deep clone) 以下举个例子, 说明以上情况.

被克隆类:ShadowClone.java

```
public class ShadowClone implements Cloneable
{
    // 基本类型
    private int a;
    // 非基本类型
    private String b;
    // 非基本类型
    private int[] c;
    // 重写Object.clone() 方法, 并把protected改为public
    @Override
    public Object clone()
    {
        ShadowClone sc = null;
        try
        {
            sc = (ShadowClone) super.clone();
        } catch (CloneNotSupportedException e)
        {
            e.printStackTrace();
        }
        return sc;
    }
    public int getA()
    {
        return a;
    }
    public void setA(int a)
    {
        this.a = a;
    }
    public String getB()
    {
        return b;
    }
    public void setB(String b)
    {
        this.b = b;
    }
    public int[] getC()
    {
        return c;
    }
    public void setC(int[] c)
    {
        this.c = c;
    }
}
```

测试类Test.java

```
public class Test
{
    public static void main(String[] args) throws CloneNotSupportedException
    {
        ShadowClone c1 = new ShadowClone();
        //对c1赋值
        c1.setA(100) ;
        c1.setB("clone1") ;
        c1.setC(new int[] {1000}) ;

        System.out.println("克隆前: c1.a="+c1.getA() );
        System.out.println("克隆前: c1.b="+c1.getB() );
        System.out.println("克隆前: c1.c[0]="+c1.getC()[0]);
        System.out.println("-----") ;

        //克隆出对象c2, 并对c2的属性A, B, C进行修改

        ShadowClone c2 = (ShadowClone) c1.clone();

        //对c2进行修改
        c2.setA(50) ;
        c2.setB("clone2");
        int []a = c2.getC() ;
        a[0]=500 ;
        c2.setC(a);

        System.out.println("克隆后: c1.a="+c1.getA() );
        System.out.println("克隆后: c1.b="+c1.getB() );
        System.out.println("克隆后: c1.c[0]="+c1.getC()[0]);
        System.out.println("-----") ;

        System.out.println("克隆后: c2.a=" + c2.getA());
        System.out.println("克隆后: c2.b=" + c2.getB());
        System.out.println("克隆后: c2.c[0]=" + c2.getC()[0]);
    }
}
```

```
    }
}
```

结果：  
克隆前：c1.a=100  
克隆前：c1.b=clone1  
克隆前：c1.c[0]=1000

克隆后：c1.a=100  
克隆后：c1.b=clone1  
克隆后：c1.c[0]=500

克隆后：c2.a=50  
克隆后：c2.b=clone2  
克隆后：c2.c[0]=500

问题出现了,我指修改了克隆后的对象c2.c的值,但c1.c的值也改变了,与c2的值相等.  
以下针对浅克隆得出结论:基本类型是可以被克隆的,但引用类型只是copy地址,并没有copy这个地址指向的对象的值,这使得两个地址指向同一值,修改其中一个,当然另一个也就变了.  
由此可见,浅克隆只适合克隆基本类型,对于引用类型就不能实现克隆了.

那如何实现克隆引用对象呢,以下提供一种方法.            用序列化与反序列化实现深克隆(deep copy)

被克隆对象.DeepClone.java

```
import java.io.Serializable;
//要实现深克隆必须实现Serializable接口
public class DeepClone implements Serializable
{
    private int a;
    private String b;
    private int[] c;
    public int getA()
    {
        return a;
    }
    public void setA(int a)
    {
        this.a = a;
    }
    public String getB()
    {
        return b;
    }
    public void setB(String b)
    {
        this.b = b;
    }
    public int[] getC()
    {
        return c;
    }
    public void setC(int[] c)
    {
        this.c = c;
    }
}
```

测试类Test.java

```
import java.io.ByteArrayInputStream;
import java.io.ByteArrayOutputStream;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
public class Test
{
    public static void main(String[] args) throws CloneNotSupportedException
    {
        Test t = new Test();
        DeepClone dc1 = new DeepClone();
        // 对dc1赋值
        dc1.setA(100);
        dc1.setB("clone1");
        dc1.setC(new int[] { 1000 });
        System.out.println("克隆前: dc1.a="+ dc1.getA());
        System.out.println("克隆前: dc1.b="+ dc1.getB());
        System.out.println("克隆前: dc1.c[0]="+ dc1.getC()[0]);
        System.out.println("-----");
        DeepClone dc2 = (DeepClone) t.deepClone(dc1);
        // 对c2进行修改
        dc2.setA(50);
        dc2.setB("clone2");
        int[] a = dc2.getC();
        a[0] = 500;
        dc2.setC(a);
        System.out.println("克隆前: dc1.a="+ dc1.getA());
        System.out.println("克隆前: dc1.b="+ dc1.getB());
        System.out.println("克隆前: dc1.c[0]="+ dc1.getC()[0]);
        System.out.println("-----");
        System.out.println("克隆后: dc2.a="+ dc2.getA());
        System.out.println("克隆后: dc2.b="+ dc2.getB());
        System.out.println("克隆后: dc2.c[0]="+ dc2.getC()[0]);
    }
    // 用序列化与反序列化实现深克隆
    public Object deepClone(Object src)
    {
        Object o = null;
        try
        {
            if (src != null)
            {
                ByteArrayOutputStream baos = new ByteArrayOutputStream();
                ObjectOutputStream oos = new ObjectOutputStream(baos);
                oos.writeObject(src);
                oos.close();
                ByteArrayInputStream bais = new ByteArrayInputStream(baos
                    .toByteArray());
                ObjectInputStream ois = new ObjectInputStream(bais);
                o = ois.readObject();
                ois.close();
            }
        } catch (IOException e)
        {
            e.printStackTrace();
        } catch (ClassNotFoundException e)
        {
            e.printStackTrace();
        }
        return o;
    }
}
```

结果：  
克隆前：dc1.a=100  
克隆前：dc1.b=clone1  
克隆前：dc1.c[0]=1000

克隆后：dc2.a=50  
克隆后：dc2.b=clone2  
克隆后：dc2.c[0]=500

总结：  
当克隆的对象只有基本类型,不含引用类型时,可以用浅克隆实现.

当克隆的对象含有引用类型时,必须使用深克隆实现.



## 谈谈final，finally，finalize的区别

**final**修饰符（关键字）

如果一个类被声明为**final**，意味着它不能再派生出新的子类，不能作为父类被继承。因此一个类不能既被声明为 **abstract**的，又被声明为**final**的。将变量或方法声明为**final**，可以保证它们在使用中不被改变。被声明为**final**的变量必须在声明时给定初值，而在以后的引用中只能读取，不可修改。被声明为**final**的方法也同样只能使用，不能重载**finally**？再异常处理时提供 **finally** 块来执行任何清除操作。如果抛出一个异常，那么相匹配的 **catch** 子句就会执行，然后控制就会进入 **finally** 块（如果有的话）。

**finalize**方法名

**Java** 技术允许使用 **finalize**（）方法在垃圾收集器将对象从内存中清除出去之前做必要的清理工作。这个方法是由垃圾收集器在确定这个对象没有被引用时对这个对象调用的。它是在 **Object** 类中定义的，因此所有的类都继承了它。子类覆盖 **finalize**（）方法以整理系统资源或者执行其他清理工作。**finalize**（）方法是在垃圾收集器删除对象之前对这个对象调用的。

## Java对象的强、软、弱和虚引用

### 强引用

```
Object o=new Object();
Object o1=o;
```

上面代码中第一句是在**heap**堆中创建新的**Object**对象通过**o**引用这个对象，第二句是通过**o**建立**o1**到**new Object()**这个**heap**堆中的对象的引用，这两个引用都是强引用.只要存在对**heap**中对象的引用，**gc**就不会收集该对象.如果通过如下代码：

```
o=null;
o1=null;
```

如果显式地设置**o**和**o1**为**null**，或超出范围，则**gc**认为该对象不存在引用，这时就可以收集它了。可以收集并不等于就会一会被收集，什么时候收集这要取 决于**gc**的算法，这要就带来很多不确定性。例如你就想指定一个对象，希望下次**gc**运行时把它收集了，那就没办法了，有了其他的三种引用就可以做到了。其他 三种引用在不妨碍**gc**收集的情况下，可以做简单的交互。

**heap**中对象有强可及对象、软可及对象、弱可及对象、虚可及对象和不可到达对象。应用的强弱顺序是强、软、弱、和虚。对于对象是属于哪种可及的对象，由他的最强的引用决定。如下：

```
String abc=new String("abc"); //1
SoftReference<String> abcSoftRef=new SoftReference<String>(abc); //2
WeakReference<String> abcWeakRef = new WeakReference<String>(abc); //3
abc=null; //4
abcSoftRef.clear(); //5
```

第一行在**heap**中对中创建内容为“abc”的对象，并建立**abc**到该对象的强引用,该对象是强可及的。

第二行和第三行分别建立对**heap**中对象的软引用和弱引用，此时**heap**中的对象仍是强可及的。

第四行之后**heap**中对象不再是强可及的，变成软可及的。同样第五行执行之后变成弱可及的。

### SoftReference（软引用）

软引用是主要用于内存敏感的高速缓存。在**jvm**报告内存不足之前会清除所有的软引用，这样一来**gc**就有可能收集软可及的对象，可能解决内存吃紧问题，避免内存溢出。什么时候会被收集取决于**gc**的算法和**gc**运行时可用内存的大小。当**gc**决定要收集软引用是执行以下过程,以上面的**abc SoftRef**为例：

- 1、首先将**abcSoftRef**的**referent**设置为**null**，不再引用**heap**中的**new String("abc")**对象。
- 2、将**heap**中的**new String("abc")**对象设置为可结束的(**finalizable**)。
- 3、当**heap**中的**new String("abc")**对象的**finalize()**方法被运行而且该对象占用的内存被释放， **abcSoftRef**被添加到它的**ReferenceQueue**中。

注:对**ReferenceQueue**软引用和弱引用可以有可无，但是虚引用必须有，参见：

```
Reference(T paramT, ReferenceQueue<? super T>paramReferenceQueue)
```

被 **Soft Reference** 指到的对象，即使没有任何 **Direct Reference**，也不会被清除。

一直到 **JVM** 内存不足且 没有 **Direct Reference** 时才会清除，**SoftReference** 是用来设计 **object-cache** 之用的。

如此一来 **SoftReference** 不但可以把对象 **cache** 起来，也不会造成内存不足的错误（**OutOfMemoryError**）。我觉得 **Soft Reference** 也适合拿来实作 **pooling** 的技巧。

```
A obj = new A();
SoftReference sr = new SoftReference(obj);
```

```
//引用时
if(sr!=null){
    obj = sr.get();
}else{
    obj = new A();
    sr = new SoftReference(obj);
}
```

### 弱引用

当**gc**碰到弱可及对象，并释放**abcWeakRef**的引用，收集该对象。但是**gc**可能需要对此运用才能找到该弱可及对象。通过如下代码可以了明了的看出它的作用：

```
String abc=new String("abc");
WeakReference<String> abcWeakRef = new WeakReference<String>(abc);
abc=null;
System.out.println("before gc: "+abcWeakRef.get());
System.gc();
System.out.println("after gc: "+abcWeakRef.get());
```

运行结果：

```
before gc: abc
after gc: null
```

**gc**收集弱可及对象的执行过程和软可及一样，只是**gc**不会根据内存情况来决定是不是收集该对象。

如果你希望能随时取得某对象的信息，但又不想影响此对象的垃圾收集，那么你应该用 **Weak Reference** 来记住此对象，而不是用一般的 **reference**。

```
A obj = new A();

WeakReference wr = new WeakReference(obj);

obj = null;

//等待一段时间，obj对象就会被垃圾回收
...

if (wr.get()==null) {
    System.out.println("obj 已经被清除了 ");
} else {
    System.out.println("obj 尚未被清除，其信息是 "+obj.toString());
}
...
```

在此例中，透过 **get()** 可以取得此 **Reference** 的所指到的对象，如果返回值为 **null** 的话，代表此对象已经被清除。

这类的技巧，在设计 **Optimizer** 或 **Debugger** 这类的程序时经常会用到，因为这类程序需要取得某对象的信息，但是不可以 影响此对象的垃圾收集。

### PhantomRefrence（虚引用）

虚顾名思义就是没有的意思，建立虚引用之后通过**get**方法返回结果始终为**null**，通过源代码你会发现，虚引用通向会把引用的对象写进**referent**，只是**get**方法返回结果为**null**。先看一下和**gc**交互的过程在说一下他的作用。

- 1 不把**referent**设置为**null**，直接把**heap**中的**new String("abc")**对象设置为可结束的(**finalizable**)。

2 与软引用和弱引用不同，先把PhantomReference对象添加到它的ReferenceQueue中，然后在释放虚可及的对象。

你会发现在收集heap中的new String("abc")对象之前，你就可以做一些其他的事情。通过以下代码可以了解他的作用。

```
import java.lang.ref.PhantomReference;
import java.lang.ref.Reference;
import java.lang.ref.ReferenceQueue;
import java.lang.reflect.Field;

public class Test {
    public static boolean isRun = true;

    public static void main(String[] args) throws Exception {
        String abc = new String("abc");
        System.out.println(abc.getClass() + "@" + abc.hashCode());
        final ReferenceQueue referenceQueue = new ReferenceQueue<String>();
        new Thread() {
            public void run() {
                while (isRun) {
                    Object o = referenceQueue.poll();
                    if (o != null) {
                        try {
                            Field rereferent = Reference.class
                                .getDeclaredField("referent");
                            rereferent.setAccessible(true);
                            Object result = rereferent.get(o);
                            System.out.println("gc will collect:"
                                + result.getClass() + "@"
                                + result.hashCode());
                        } catch (Exception e) {
                            e.printStackTrace();
                        }
                    }
                }
            }
        }.start();
        PhantomReference<String> abcWeakRef = new PhantomReference<String>(abc,
            referenceQueue);
        abc = null;
        Thread.currentThread().sleep(3000);
        System.gc();
        Thread.currentThread().sleep(3000);
        isRun = false;
    }
}
```

结果为：

```
class java.lang.String@96354
gc will collect:class java.lang.String@96354
```

### 为什么需要使用软引用

首先，我们看一个雇员信息查询系统的实例。

我们将使用一个Java语言实现的雇员信息查询系统查询存储在磁盘文件或者数据库中的雇员人事档案信息。

作为一个用户，我们完全有可能需要回头去查看几分钟甚至几秒钟前查看过的雇员档案信息(同样，我们在浏览WEB页面的时候也经常会使用“后退”按钮)。

这时我们通常会有两种程序实现方式：

一种是把过去查看过的雇员信息保存在内存中，每一个存储了雇员档案信息的Java对象的生命周期贯穿整个应用程序始终；

另一种是当用户开始查看其他雇员的档案信息的时候，把存储了当前所查看的雇员档案信息的Java对象结束引用，使得垃圾收集线程可以回收其所占用的内存空间，当用户再次需要浏览该雇员的档案信息的时候，重新构建该雇员的信息。

很显然，第一种实现方法将造成大量的内存浪费，而第二种实现的缺陷在于即使垃圾收集线程还没有进行垃圾收集，包含雇员档案信息的对象仍然完好地保存在内存中，应用程序也要重新构建一个对象。

我们知道，访问磁盘文件、访问网络资源、查询数据库等操作都是影响应用程序执行性能的重要因素，如果能重新获取那些尚未被回收的Java对象的引用，必将减少不必要的访问，大大提高程序的运行速度。

## 在J2EE的bean设计时我们对有些bean需要实现Serializable，为什么？

实现了Serializable接口的对象，可将它们转换成一系列字节，并可在以后完全恢复回原来的样子。这一过程亦可通过网络进行。这意味着序列化机制能自动补偿操作系统间的差异。换句话说，可以先在Windows机器上创建一个对象，对其序列化，然后通过网络发给一台Unix机器，然后在那里准确无误地重新“装配”。不必关心数据在不同机器上如何表示，也不必关心字节的顺序或者其他任何细节。

## 每一篇不宜写得过长，下篇继续

作者：lifetragedy 发表于2013/8/5 0:43:28 [原文链接](#)  
阅读：34464 评论：64 [查看评论](#)

### think in java interview-高级开发人员面试宝典(一)

## “生死九重门”

无论你是在职，非在职，高级工程师，工程师，架构师，如果你正在面试阶段，请看完此文！

相信这篇文章对你的职业生涯和价值观会造成重大的改变！

如果你是一名PM或者是管理者正在物色合适的开发人选，那么我相信这篇文章同样会在在你考虑如何挑选技术型人材上给你带来重大的帮助。

## 本系列不适合想去应聘PM，管理路线的人士！！！！

作为一名技术型人材由其是程序员，用什么可以恒量自己是否合格或者我们怎么去恒量一个程序员是否质深、是否合格？

比如说：他可以适合一般的软件工程师岗位，还是适合高级工程师岗位，还是架构师、系统分析员这样的岗位呢？

作为一名JAVA开发人员来说，JAVA涉及到的面太广了，我们就拿企业级开发人员即J2EE开发人员的基本功来说事吧！

在面试前先问自己5个问题：

**1.** 我想要什么

**2.** 我会什么

**3.** 目前的市场需要什么

**4.** 目前我还缺什么

5. 如何去补缺

一个**J2EE**高级开发人员需要具备些什么技能？

或许你会说：我会SSH，我会工作流，我会jQuery, ExtJS，我参加过数个大型项目，我带过**5**，**6**个人的队，我有SCJP, SCEA证书，我工作经验达**5**年。

我会告诉你，其实你是**Nothing**，**Nothing**！！！

我这个**Nothing**的前提是指好的公司，好的项目组招人时的要求！！为什么这么说？

这样告诉你吧，越是好的公司，越是好的项目组在招开发人员时他们面的东西越是底层。

本人结合以前在公司负责**JAVA**开发人员的招聘与策划，和对新进**JAVA**人员的培训经验即自己本身近**10**年来**50**多次的面试经历总结,发觉：

越是工作年限长的人，技术越是倒退

尤其到了**3**年，**5**年，**7**年的工作经验的高级开发工程师群体们，往往发现自己在择业上碰到了一个瓶颈，上不上去，跳槽时工资能涨个**1000**，**2000**已经到顶了，几乎不太可能达到他们本身期望的的**30%-50%**的涨幅。**这些原因主要源自于“基础”**。

大部分人基础全忘了，或者以前做过几年程序，但是后面转向管理，TL等角色后，平时开发的时间下降到只占到本身工作的**30-50%**这样的量了；

或者有很多一批人，或者由于工作、项目等其它的原因，没有接触过一些正规的，系统化的知识；

或者平时不善于总结，只求我用技术时就google, copy & paste，忘记了代码基本功了；

您别不信，来，我们就来试一下

不许查网络，你现在就处在面试官面前，面试官问你这么一个问题：

面试官： 你有没有平时自己覆写过**hashCode**, **equals**两个方法？

**回答：**我写过的（可能很多人以前用的是**eclipse**或者是**myeclipse**里的自动生成器

**面试官：**怎么写？

**回答：**。。。。。。（大致讲出来什么原理）

面试官：你为什么覆写**hashCode & equals**方法

**回答：**（如果准备过的人能够回答的出这个问题）

面试官：如果说两个对象**equals**它们的**hashCode**是否一定相同？如果两个对象的**hashCode**相同它们一定**equals**吗？

**回答：****70%**以上的人会挂在这个问题上

面试官：你知道在写**hashCode**时为什么要使用一个**31**这样的数，我用**100**，**11**，**17**是不是一样呢？有什么不一样的地方呢？

到此为之，基本都挂了（如果在没有网络的情况下）

你看到这边或许会说：**come one**，有eclipse代码生成器，这个没必要。

**OK**，我们再来：

面试官：**Oracle**中的**index**有**bitmap**, **btree**两种索引，请比较它们

回答：**70-80**说自己**Oracle**用了**3**，**4**年的人首先就不知道这个**bitmap**和**btree**是什么，更无从谈起比较了

以上两道问题，可以作为一个面试官判断你是否在一直写代码和是否真正熟悉**oracle**的标准，可能因为这两道题，你就挂了，面试官不愿意再和你谈下去了。

真的，这两道问题其实基础了不能再基础了，就和你不会**1+1=2**一样，你说“我熟练掌握四则运算”是一个道理。

记住，越是好的公司，越是好的项目，面试的问题就越是底层，**IBM**, **eBay**, **HP**, 百度，**MS**，1号店，阿里巴巴，**Oracle**，**starcite**等都是这样。

一般，一个高级**JAVA**开发人员的面试，其实是分成**3**个部分

一、基础考核

二、**J2EE**相关知识和框架以及**J2EE**相关性能调优等的考核

三、综合考核，即给你**1-2**个**CASE**,让你说说你的设计和想法

其中，最难过的就是这个基础。

这个基础中，其实再可以分为四个部分的考核：

第一部分：笔试，**45-60**分钟笔试卷，基本都是多选

第二部分：**SQL**部分，一般是数据库基础，**SQL**统计函数等的写法是肯定逃不掉的

第三部分：然后是**1-2**道的问答类也可以是设计模式方面的题

第四部分：放入一道智力题（**ebay**, **Oracle**, **MS**, 百度）很喜欢干这种勾当

有的公司也会把这四部分都放在笔试题里，有的则会加上多线程，算法变成**6**道问答直接让面试者在面试时给面试者一些白纸，然后就此一问一答。

像一些外资类公司，会加入一道“电话面试”关，一般在**30**分钟，过了电话面试再去他们公司本部，然后就是**3**轮的**f2f**的面试，最后再和主管谈，HR谈，一般**6**轮面试是需要的。

这边请记住，这**6**轮面试**9**成的可能性都是全英语面试。。。。。

很多搞**IT**的，首先英语本身没有重视，有的不能够很好的交流，基础又丢了不少，那基本这样的面试都会被挂，这也是为什么有些人总觉得这些大公司怎么怎么好，怎么怎么难进！

其实不然，掌握了技巧，要通过这样的面试，易反掌，因为你发觉吗，一般人都是伤在基础部分的面试，这关过了基本后面都很好过的，所以我们把基础部分**4**个部分的面试，即：

**JAVA**基础+**SQL**+设计模式+**2EE**+算法+智力题，总结成“**6**道题”，别看看这些都是基础，如果真的让你处在面试官面前时，你不能查**GOOGLE**,没有**eclipse**，相信**90%**的人都会挂掉。

这“**6**道题”其实就是考核一个高级开发人员是否真正适合高级开发人员岗位的标准，也是一个真正做技术的人必须要具备的能力，没有了这个基础，你的一切，你的工作经验，你的简历再漂亮，也都是浮云。

有些人会说，我能达这**6**个部分的**50%**左右的能力，有些人能达到**70%**,那些面试官就会觉得：这个人我可以用了，只是招进来后砍砍价吧。

试问，你愿意被人砍价吗？对吧，被人砍掉**2-3K**，你不疯了，如果你工资本身就**4**，**50**万被砍个**2**，**3K**也无所谓，大部分人都是**5K**冲**8K**，更多的是**8K**冲**13**，**15**或者是**12**，**15**想冲**20k**的人，那么，我告诉你，你必须，对，是必须这“**6**道题”至少，这边看清了哦，是至少能够回答到**90%**，如果是英语面试你还要准备全英语面试哦！！！

那么，我们说为什么一个企业要这么去考核一个开发人员呢？



首先，对于企业来说，招一个高级开发人员，是不容易的，中国人，做什么都喜欢一哄而起，96年房地产发展时，短短1，2年内到处都是房地产公司，设计所，设计公司，CAD盛行一时。

2000年初，IT开始盛行，这个更猛，到处都是IT公司，都是IT人员，学个IT就高档了，学个IT就高技术了，搞得连在6层楼居民住宅中租个2室户也能开个IT公司了。

从业人数一庞大，就造成了鱼龙混杂，层次参差不齐。

于是SSH满天飞，大有

“练好STRUTS,HIBERNATE,SPRING,货卖帝王家”的感觉，嘿嘿，你别不喜爱看，现在的市场还真TMD的是这个样。

但是，事实摆在那边，你要想进一个好的公司，要被一个好的项目组，你的工作想要有价值，可以学到真正知识、锻炼到能力的项目、TEAM中去，你就得过这6道关，所以，无论你觉得现在如何，或者觉得不爱看我写的这些或者觉得“我这样挺好呗”，但你就是不能否认这6道关是摆在那边的，它是你提高一个层次的拦路虎，不打掉它，你永远上升不了。

啊。。。或许这就是许多人到了后来选择走管理的道路的原因吧，这也许是中国IT至所以还停留在初级阶段的原因吧，唉。。。中国式的一夜暴富梦想充斥着搞IT人们的头脑，建筑在沙滩上的城堡总有一天会倒塌。。。。。

come on MAN!! 又来了，老套了，不好意思，发发牢骚。

下面我们就要讲如何来过这六道关，一道道我们来过，任何一个以技术为终生目标的开发人员，这些基础，其实不是通过看了我这篇文章，您临终抱一下佛脚就可以抱得出的，我告诉你，抱不出的，就算你死记硬背了我在这系列文章中提到的近百道例题，你如果不是真正把它们做为你一直需要修练的基本功的话你到头来还是要露馅的。

这就和我上面用1+1=2都不知道何谈四则运算是一个道理，这就和练武功一样，奇材很少，所有人都是从练马步开始的。

下面我们将慢慢来谈“面经”。


我这一阵超忙，小孩小照顾起来不易，工作上又是做的研发，没有时间概念，如果更新不及时，还往大家抱歉！

作者：lifetragedy 发表于2013/8/4 22:44:22 [原文链接](#)  
阅读：21321 评论：48 [查看评论](#)

公司简介 | 广告服务 | 银行汇款帐号 | 联系方式 | 版权声明 | 法律顾问 | 问题报告

北京创新乐知信息技术有限公司 版权所有, 京 ICP 证 070598 号

世纪乐知(北京)网络技术有限公司 提供技术支持

 Email: [webmaster@csdn.net](mailto:webmaster@csdn.net)

Copyright © 1999-2011, CSDN.NET, All Rights Reserved

