PHP 程序员 > 面试笔试真题库

< / > 在这里 /
有技术大牛倾囊相授的面试经验和技巧
< / > 在这里 /
有来自各大名企的面试真题大合集
< / > 在这里 /
有专业团队倾力打造的精炼解析

猿媛之家 / 组编　琉忆 楚秦 等 / 编著

PROGRAMER
> INTERVIEW QUESTIONS LIBRARY

本书聚焦名企面试笔试特点，精选20套真题卷

单选 / 多选 / 填空 / 问答 / 编程 / 开放 / 智力，多题型练习

当你细细品读完本书后，各类企业的offer将任由你挑选

一 书 在 手 / 工 作 不 愁 > .

机械工业出版社
CHINA MACHINE PRESS

PHP 程序员面试笔试真题库

猿媛之家　组编
琉　忆　楚　秦　等编著

机械工业出版社

本书针对当前各大 IT 企业面试笔试中特性与侧重点，精心挑选了近 3 年来约 20 家典型 IT 企业的 PHP 面试笔试真题，这些企业业务包括系统软件、搜索引擎、电子商务、手机 APP、安全关键软件等，所提供的 PHP 面试笔试真题涉及的知识点包括 PHP 语言基础知识、计算机网络、操作系统、数据结构与算法、数据库、设计模式等，非常具有代表性与参考性。同时，本书对这些题目进行了庖丁解牛式的分析与讲解，针对试题中涉及的部分重难点问题，本书都进行了适当地扩展与延伸，力求对知识点的讲解清晰而不紊乱，全面而不啰嗦，使得读者通过本书不仅能够获取到求职的知识，还能更有针对性地进行求职准备，最终获得一份满意的工作。

本书是一本计算机相关专业毕业生面试、笔试的求职用书，同时也适合期望在计算机软、硬件行业大显身手的计算机爱好者阅读。

图书在版编目（CIP）数据

PHP 程序员面试笔试真题库 / 猿媛之家组编；琉忆等编著. —北京：机械工业出版社，2019.3

ISBN 978-7-111-62159-1

Ⅰ. ①P… Ⅱ. ①猿… ②琉… Ⅲ. ①PHP 语言—程序设计—资格考试—习题集 Ⅳ. ①TP312.8-44

中国版本图书馆 CIP 数据核字（2019）第 039072 号

机械工业出版社（北京市百万庄大街 22 号 邮政编码 100037）

策划编辑：尚 晨 责任编辑：尚 晨

责任校对：张艳霞 责任印制：张 博

三河市宏达印刷有限公司印刷

2019 年 3 月第 1 版 • 第 1 次印刷

184mm×260mm • 15.25 印张 • 371 千字

0001—3000 册

标准书号：ISBN 978-7-111-62159-1

定价：59.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

电话服务

服务咨询热线：（010）88361066

读者购书热线：（010）68326294

网络服务

机 工 官 网：www.cmpbook.com

机 工 官 博：weibo.com/cmp1952

金 书 网：www.golden-book.com

教育服务网：www.cmpedu.com

封面无防伪标均为盗版

前　言

《PHP 程序员面试笔试真题库》是继《PHP 程序员面试笔试宝典》《PHP 程序员面试笔试真题与解析》《PHP 程序员面试算法宝典》后的第四本书，这本书与前三本书形成了互补完善的作用。面试笔试宝典以 PHP 面试知识点的讲解为主，真题与解析主要把历年常考的面试题进行汇总讲解，算法宝典则是把面试常考的算法提炼成书进行讲解分析。而真题库则是把历年各大知名互联网公司考的面试笔试题提炼成套题，面试者可以为面试大公司的笔试题提前预热，也可以根据要准备面试的行业来针对性地做笔试题，提高自己的笔试能力。

真题库对比前三本书的特色具有以下几点：

1）行业针对性强。每套真题都摘取自典型互联网公司历年考查的真题，再根据选择题、填空题、简答题、编程题等题型整理成套题，让面试者更加清晰地知道该行业公司对 PHP 程序员的笔试题是如何考查的。

2）讲解分离，更利于自测。本书分为套题和解析两部分，面试者可以先把套题都做完，再对着答案查看解答是否正确，最后再看解析。这本书更像是一本自测的面试书籍。

3）考查率高。书中大部分真题出现在各小公司的笔试中，这不仅是大公司考查的笔试题，也更多地被中小企业所参考，所以即使不是去面试这些大互联网公司，也完全可以运用本书的套题练习，去应对各互联网公司的面试。

由于篇幅有限，部分的互联网公司真题有所删减，想了解更多 PHP 面试、开发的知识可以关注编者博客“琉忆编程库”（www.shuaiqi100.com）、微信公众号和小程序“琉忆编程库”。这几个平台上每天都会更新 PHP 相关内容，以便最大限度地满足读者需求。

由于编者水平有限，书中不足之处在所难免，还望读者见谅。如果发现书中的问题或者有什么疑问，可以通过 QQ 或邮箱联系我，邮箱 yuancoder@foxmail.com 或330168885@qq.com。

最后真诚地祝愿每一位求职者都能一帆风顺、马到成功。

编　者

目　录

真题详解篇

面试笔试经验技巧篇

想找到一份程序员的工作，一点技术都没有显然是不行的，但是，只有技术也是不够的。面试笔试经验技巧篇主要针对程序员面试笔试中遇到的常见问题进行深度解析，并且结合实际情景，给出一个较为合理的参考答案以供读者学习与应用，掌握这些问题的解答精髓，对于求职者大有裨益。

经验技巧 1　如何巧妙地回答面试官的问题?

在程序员面试中，求职者不可避免地需要回答面试官各种刁钻、犀利的问题，这时不能简单地回答“是”或者“不是”，而应该具体分析“是”或者“不是”的理由。

那么，面对面试官提出的各类问题，如何才能条理清晰地回答呢？如何才能让自己的回答令面试官满意呢？

谈话是一种艺术，回答问题也是一种艺术。同样的问题，不同的回答方式，往往会产生不同的效果，甚至是截然不同的效果。在此，编者提出以下几点建议，供读者参考。

首先，回答问题务必谦虚谨慎。既不能让面试官觉得自己很自卑，唯唯诺诺，也不能让面试官觉得自己清高自负，而应该通过问题的回答表现出自己自信从容、不卑不亢的一面。例如，当面试官提出“你在项目中起到了什么作用”的问题时，如果求职者回答：我完成了团队中最难的工作，此时就会给面试官一种居功自傲的感觉，而如果回答：我完成了文件系统的构建工作，这个工作被认为是整个项目中最具有挑战性的一部分内容，因为它几乎无法重用以前的框架，需要重新设计。这种回答不仅不傲慢，反而有理有据，更能打动面试官。

其次，回答面试官的问题时，不要什么都说，要适当地留有悬念。人一般都有猎奇的心理，面试官自然也不例外，而且，人们往往对好奇的事情更有兴趣、更加偏爱，也更加记忆深刻。所以，在回答面试官问题时，切记说关键点而非细节，说重点而非和盘托出，通过关键点，吸引面试官的注意力，等待他们继续“刨根问底”。例如，当面试官对你的简历中一个算法问题有兴趣，希望了解时，可以作如下回答：我设计的这种查找算法，对于 80%以上的情况，都可以将时间复杂度从 O(n)降低到 O(logn)，如果您有兴趣，我可以详细给您分析具体的细节。

最后，回答问题要条理清晰、简单明了，最好使用“三段式”方式。所谓“三段式”，有点类似于中学作文中的写作风格，包括“场景/任务”“行动”“结果”三部分内容。以面试官提的问题“你在团队建设中，遇到的最大挑战是什么”为例，第一步，分析场景/任务：在我参与的一个 ERP 项目中，我们团队一共四个人，除了我以外的其他三个人中，两个人能力很给力，人也比较好相处，但有一个人却不太好相处，每次我们小组讨论问题的时候，他都不太爱说话，也很少发言，分配给他的任务也很难完成。第二步，分析行动：为了提高团队的综合实力，我决定找个时间和他好好单独谈一谈。于是我利用周末时间，约他一起吃饭，吃饭的时候，顺便讨论了一下我们的项目，我询问了一些项目中他遇到的问题，通过他的回答，我发现他并不懒，也不糊涂，只是对项目不太了解，缺乏经验，缺乏自信而已，所以越来越孤立，越来越不愿意讨论问题。为了解决这个问题，我尝试着把问题细化到他可以完成的程度，从而建立起他的自信心。第三步，分析结果：他是小组中水平最弱的人，但是，慢慢地，他的技术变得越来越厉害了，也能够按时完成安排给他的工作了，人也越来越自信了，也越来越喜欢参与我们的讨论，并发表自己的看法，我们也都愿意与他一起合作了。“三段式”回答的一个最明显的好处就是条理清晰，既有描述，也有结果，有理有据，让面试官一目了然。

回答问题的技巧，是一门大的学问。求职者完全可以在平时的生活中加以练习，提高自己与人沟通的技能，等到面试时，自然就得心应手了。

经验技巧 2 如何回答技术性的问题？

程序员面试中，面试官会经常询问一些技术性的问题，有的问题可能比较简单，都是历年的笔试面试真题，求职者在平时的复习中会经常遇到，应对自然不在话下。但有的题目可能比较难，来源于 Google、Microsoft 等大企业的题库或是企业自己为了招聘需要设计的题库，求职者可能从来没见过或者从来都不能完整地、独立地想到解决方案，而这些题目往往又是企业比较关注的。

如何能够回答好这些技术性的问题呢？编者建议：会做的一定要拿满分，不会做的一定要拿部分分。即对于简单的题目，求职者要努力做到完全正确，毕竟这些题目，只要复习得当，完全回答正确一点问题都没有（编者认识的一个朋友据说把《编程之美》《编程珠玑》《程序员面试笔试宝典》上面的技术性题目与答案全都背得滚瓜烂熟了，后来找工作简直成了“offer 杀器”，完全无解了）；对于难度比较大的题目，不要惊慌，也不要害怕，即使无法完全做出来，也要努力思考问题，哪怕是半成品也要写出来，至少要把自己的思路表达给面试官，让面试官知道你的想法，而不是完全回答不会或者放弃，因为面试官很多时候除了关注你的独立思考问题的能力以外，还会关注你技术能力的可塑性，观察求职者是否能够在别人的引导下去正确地解决问题，所以，对于你不会的问题，他们很有可能会循序渐进地启发你去思考，通过这个过程，让他们更加了解你。

一般而言，在回答技术性问题时，求职者大可不必胆战心惊，除非是没学过的新知识，否则，一般都可以采用以下 6 个步骤来分析解决。

（1）勇于提问

面试官提出的问题，有时候可能过于抽象，让求职者不知所措，或者无从下手，所以，对于面试中的疑惑，求职者要勇敢地提出来，多向面试官提问，把不明确或二义性的情况都问清楚。不用担心你的问题会让面试官烦恼，影响你的面试成绩，相反还会对面试结果产生积极影响：一方面，提问可以让面试官知道你在思考，也可以给面试官一个心思缜密的好印象；另一方面，方便后续自己对问题的解答。

例如，面试官提出一个问题：设计一个高效的排序算法。求职者可能丈二和尚摸不到头脑，排序对象是链表还是数组？数据类型是整型、浮点型、字符型还是结构体类型？数据基本有序还是杂乱无序？数据量有多大，1000 以内还是百万以上个数？此时，求职者大可以将自己的疑问提出来，问题清楚了，解决方案也自然就出来了。

（2）高效设计

对于技术性问题，如何才能打动面试官？完成基本功能是必须的，仅此而已吗？显然不是，完成基本功能顶多只能算及格水平，要想达到优秀水平，至少还应该考虑更多的内容。以排序算法为例：时间是否高效？空间是否高效？数据量不大时也许没有问题，如果是海量数据呢？是否考虑了相关环节，例如数据的“增删改查”？是否考虑了代码的可扩展性、安全性、完整性以及鲁棒性？如果是网站设计，是否考虑了大规模数据访问的情况？是否需要考虑分布式系统架构？是否考虑了开源框架的使用？

（3）伪代码先行

有时候实际代码会比较复杂，上手就写很有可能会漏洞百出、条理混乱，所以，求职

者可以首先征求面试官的同意，在编写实际代码前，写一个伪代码或者画好流程图，这样做往往会让思路更加清晰明了。

切记在写伪代码前要告诉面试官，他们很有可能对你产生误解，认为你只会纸上谈兵，实际编码能力却不行。只有征得了他们的允许，方可先写伪代码。

（4）控制节奏

如果是算法设计题，面试官都会给求职者一个时间限制用以完成设计，一般为 20 分钟左右。完成得太慢，会给面试官留下能力不行的印象，但完成得太快，如果不能保证百分百正确，也会给面试官留下毛手毛脚的印象，速度快当然是好事情，但只有速度，没有质量，速度快根本不会给面试加分。所以，编者建议，回答问题的节奏最好不要太慢，也不要太快，如果实在是完成得比较快，也不要急于提交给面试官，最好能够利用剩余的时间，认真仔细地检查一些边界情况、异常情况及极性情况等，看是否均能满足要求。

（5）规范编码

回答技术性问题时，多数都是纸上写代码，离开了编译器的帮助，求职者要想让面试官对自己的代码一看即懂，除了字迹要工整，不能龙飞凤舞以外，最好是能够严格遵循编码规范：函数变量命名、换行缩进、语句嵌套和代码布局等，同时，代码设计应该具有完整性，保证代码能够完成基本功能、输入边界值能够得到正确地输出、对各种不合规范的非法输入能够做出合理的错误处理，否则，写出的代码即使无比高效，面试官也不一定看得懂或者看起来非常费劲，这些对面试成功都是非常不利的。

（6）精心测试

在软件界，有一句真理：任何软件都有 bug。但不能因为如此就纵容自己的代码，允许错误百出。尤其是在面试过程中，实现功能也许并不十分困难，困难的是在有限的时间内设计出的算法，各种异常是否都得到了有效的处理，各种边界值是否都在算法设计的范围内。

测试代码是让代码变得完备的高效方式之一，也是一名优秀程序员必备的素质之一。所以，在编写代码前，求职者最好能够了解一些基本的测试知识，做一些基本的单元测试、功能测试、边界测试以及异常测试。

在回答技术性问题时，注意在思考问题的时候，千万别一句话都不说，面试官面试的时间是有限的，他们希望在有限的时间内尽可能地去了解求职者，如果求职者坐在那里一句话不说，不仅会让面试官觉得求职者技术水平不行，而且会认为求职者思考问题能力以及沟通能力可能都存在问题。

其实，在面试时，求职者往往会存在一种思想误区，把技术性面试的结果看得太重要了。面试过程中的技术性问题，结果固然重要，但也并非最重要的内容，因为面试官看重的不仅仅是最终的结果，还包括求职者在解决问题的过程中体现出来的逻辑思维能力以及分析问题的能力。所以，求职者在与面试官的博弈中，要适当地提问，通过提问获取面试官的反馈信息，并抓住这些有用的信息进行辅助思考，从而博得面试官的认可，进而提高面试的成功率。

经验技巧 3　如何回答非技术性问题？

评价一个人的能力，除了专业能力，还有一些非专业能力，如智力、沟通能力和反应能力等，所以在 IT 企业招聘过程的笔试面试环节中，并非所有的笔试内容都是 C/C++、数据结

构与算法及操作系统等专业知识，也包括其他一些非技术类的知识，如智力题、推理题和作文题等。技术水平测试可以考查一个求职者的专业素养，而非技术类测试则更加强调求职者的综合素质，包括数学分析能力、反应能力、临场应变能力、思维灵活性、文字表达能力和性格特征等内容。考查的形式多种多样，与公务员考查相似，主要包括行测（占大多数）、性格测试（大部分都有）、应用文和开放问题等内容。

每个人都有自己的答题技巧，答题方式也各不相同，以下是一些相对比较好的答题技巧（以行测为例）：

1）合理有效的时间管理。由于题目的难易程度不同，所以不要对所有题目都“绝对的公平”、都“一刀切”，要有轻重缓急，最好的做法是不按顺序回答。行测中有各种题型，如数量关系、图形推理、应用题、资料分析和文字逻辑等，而不同的人擅长的题型是不一样的，因此应该首先回答自己最擅长的问题。例如，如果对数字比较敏感，那么就先答数量关系。

2）注意时间的把握。由于题量一般都比较大，可以先按照总时间/题数来计算每道题的平均答题时间，如 10s，如果看到某一道题 5s 后还没思路，则马上放弃。在做行测题目的时候，以在最短的时间内拿到最多分为目标。

3）平时多关注图表类题目，培养迅速抓住图表中各个数字要素间相互逻辑关系的能力。

4）做题要集中精力，只有集中精力、全神贯注，才能将自己的水平最大限度地发挥出来。

5）学会关键字查找，通过关键字查找，能够提高做题效率。

6）提高估算能力，很多时候估算能够极大地提高做题速度，同时保证正确率。

除了行测以外，一些企业非常相信个人性格对职位匹配的影响，所以都会引入相关的性格测试题用于测试求职者的性格特性，看其是否适合所投递的职位。大多数情况下，只要按照自己的真实想法选择就行了，不要弄巧成拙，因为测试是为了得出正确的结果，所以大多测试题前后都有相互验证的题目。如果求职者自作聪明，选择该职位可能要求的性格选项，则很可能导致测试前后不符，这样很容易让企业发现你是个不诚实的人，从而首先予以筛除。

经验技巧 4　如何回答系统设计题？

应届生在面试的时候，偶尔也会遇到一些系统设计题，而这些题目往往只是测试一下求职者的知识面，或者测试求职者对系统架构方面的了解，一般不会涉及具体的编码工作。虽然如此，对于此类问题，很多人还是感觉难以应对，也不知道从何说起。

如何应对此类题目呢？在正式介绍基础知识之前，首先罗列几个常见的系统设计相关的面试笔试题：

1）设计一个 DNS 的 Cache 结构，要求能够满足每秒 5000 次以上的查询，满足 IP 数据的快速插入，查询的速度要快（题目还给出了一系列的数据，比如站点数总共为 5000 万、IP 地址有 1000 万等）。

2）有 N 台机器，M 个文件，文件可以以任意方式存放到任意机器上，文件可任意分割成若干块。假设这 N 台机器的宕机率小于 1/3，想在宕机时可以从其他未宕机的机器中完整导出这 M 个文件，求最好的存储与分割策略。

3）假设有三十台服务器，每台服务器上面都存有上百亿条数据（有可能重复），如何找出这三十台机器中，根据某关键字，重复出现次数最多的前 100 条？要求使用 Hadoop 来实现。

4）设计一个系统，要求写速度尽可能快，并说明设计原理。

5）设计一个高并发系统，说明架构和关键技术要点。

6）有 25T 的 log(query->queryinfo)，log 在不断地增长，设计一个方案，给出一个 query 能快速返回 queryinfo。

以上所有问题中凡是不涉及高并发的，基本可以采用 Google 的三个技术解决，即 GFS、MapReduce 和 Bigtable，这三个技术被称为“Google 三驾马车”，Google 只公开了论文而未公开源代码，开源界对此非常有兴趣，仿照这三篇论文实现了一系列软件，如 Hadoop、HBase、HDFS 及 Cassandra 等。

在 Google 这些技术还未出现之前，企业界在设计大规模分布式系统时，采用的架构往往是 database+sharding+cache，现在很多公司（比如 taobao、weibo.com）仍采用这种架构。在这种架构中，仍有很多问题值得去探讨。如采用什么数据库，是 SQL 界的 MySQL 还是 NoSQL 界的 Redis/TFS，两者有何优劣？采用什么方式 sharding（数据分片），是水平分片还是垂直分片？据网上资料显示，weibo.com 和 taobao 图片存储中曾采用的架构是 Redis/MySQL/TFS+sharding+cache，该架构解释如下：前端 cache 是为了提高响应速度，后端数据库则用于数据永久存储，防止数据丢失，而 sharding 是为了在多台机器间分摊负载。最前端由大块大块的 cache 组成，要保证至少 99%（该数据在 weibo.com 架构中的是自己猜的，而 taobao 图片存储模块是真实的）的访问数据落在 cache 中，这样可以保证用户访问速度，减少后端数据库的压力。此外，为了保证前端 cache 中的数据与后端数据库中的数据一致，需要有一个中间件异步更新（为什么使用异步？理由简单：同步代价太高。异步有缺点，如何弥补?）数据，这个有些人可能比较清楚，新浪有个开源软件叫 Memcachedb（整合了 Berkeley DB 和 Memcached），正是为完成此功能而设计。另外，为了分摊负载压力和海量数据，会将用户微博信息经过分片后存放到不同节点上（称为“Sharding”）。

这种架构优点非常明显：简单，在数据量和用户量较小的时候完全可以胜任。但缺点是扩展性和容错性太差，维护成本非常高，尤其是数据量和用户量暴增之后，系统不能通过简单地增加机器解决该问题。

鉴于此，新的架构应运而生。新的架构仍然采用 Google 公司的架构模式与设计思想，以下将分别就此内容进行分析。

GFS：GFS 是一个可扩展的分布式文件系统，用于大型的、分布式的、对大量数据进行访问的应用。它运行于廉价的普通硬件上，提供容错功能。现在开源界有 HDFS（Hadoop Distributed File System），该文件系统虽然弥补了数据库+sharding 的很多缺点，但自身仍存在一些问题，比如：由于采用 master/slave 架构，因此存在单点故障问题；元数据信息全部存放在 master 端的内存中，因而不适合存储小文件，或者说如果存储大量小文件，那么存储的总数据量不会太大。

MapReduce：MapReduce 是针对分布式并行计算的一套编程模型。其最大的优点是：编程接口简单、自动备份（数据默认情况下会自动备三份）、自动容错和隐藏跨机器间的通信。在 Hadoop 中，MapReduce 作为分布计算框架，而 HDFS 作为底层的分布式存储系统，但 MapReduce 不是与 HDFS 耦合在一起的，完全可以使用自己的分布式文件系统替换掉 HDFS。当前 MapReduce 有很多开源实现，如 Java 实现 Hadoop MapReduce、C++实现 Sector/sphere 等，甚至有些数据库厂商将 MapReduce 集成到数据库中了。

BigTable：BigTable 俗称“大表”，是用来存储结构化数据的。编者觉得，BigTable 之所以在开源界最火爆，是因为其开源实现最多，包括 HBase、Cassandra 和 levelDB 等，使用也非常广泛。

除了 Google 的这“三驾马车”以外，还有其他一些技术可供学习与使用：

Dynamo：亚马逊的 key-value 模式的存储平台，可用性和扩展性都很好，采用 DHT（Distributed Hash Table）对数据分片，解决单点故障问题。在 Cassandra 中，也借鉴了该技术，在“BT”和“电驴”这两种下载引擎中，也采用了类似算法。

虚拟节点技术：该技术常用于分布式数据分片中。具体应用场景是：有一大块数据（可能 TB 级或者 PB 级），需按照某个字段（key）分片存储到几十（或者更多）台机器上，同时想尽量负载均衡且容易扩展。传统的做法是：Hash(key) mod N，这种方法最大的缺点是不容易扩展，即增加或者减少机器均会导致数据全部重分布，代价太大。于是新技术诞生了，其中一种是上面提到的 DHT，现在已经被很多大型系统采用，还有一种是对“Hash(key) mod N”的改进：假设要将数据分布到 20 台机器上，传统做法是 Hash(key) mod 20，而改进后，N 取值要远大于 20，比如 20000000，然后采用额外一张表记录每个节点存储的 key 的模值，比如：

node1：0～1000000

node2：1000001～2000000

……

这样，当添加一个新的节点时，只需将每个节点上部分数据移动给新节点，同时修改一下该表即可。

Thrift：Thrift 是一个跨语言的 RPC 框架。分别解释“RPC”和“跨语言”如下：RPC 是远程过程调用，其使用方式与调用一个普通函数一样，但执行体发生在远程机器上；跨语言是指不同语言之间进行通信，比如 C/S 架构中，Server 端采用 C++编写，Client 端采用 PHP 编写，怎样让两者之间通信，Thrift 是一种很好的方式。

前面的几道题均可以映射到以上几个系统的某个模块中，如：

1）关于高并发系统设计，主要有几个关键技术点，如缓存、索引、数据分片及锁粒度尽可能小。

2）题目 2 涉及现在通用的分布式文件系统的副本存放策略。一般是将大文件切分成小的 block（如 64MB）后，以 block 为单位存放三份到不同的节点上，这三份数据的位置需根据网络拓扑结构配置。一般而言，如果不考虑跨数据中心，可以这样存放，两个副本存放在同一个机架的不同节点上，而另外一个副本存放在另一个机架上，这样从效率和可靠性上都是最优的（这个 Google 公布的文档中有专门的证明，有兴趣的可参阅一下）。如果考虑跨数据中心，可将两份存在一个数据中心的不同机架上，另一份放到另一个数据中心。

3）题目 4 涉及 BigTable 的模型。主要思想是将随机写转化为顺序写，进而大大提高写速度。具体是，由于磁盘物理结构的独特设计，其并发的随机写（主要是因为磁盘寻道时间长）非常慢，考虑到这一点，在 BigTable 模型中，首先会将并发写的大批数据放到一个内存表（称为“memtable”）中，当该表大到一定程度后，会顺序写到一个磁盘表（称为“SSTable”）中，这种写法是顺序写，效率极高。此时可能有读者问，随机读可不可以这样优化？答案是看情况。通常而言，如果读并发度不高，则不可以这么做，因为如果将多个读重新排列组合后再执行，系统的响应时间太长，用户可能接受不了，而如果读并发度极高，也许可以采用类似机制。

经验技巧 5 如何解决求职中的时间冲突问题?

对于求职者而言，求职季就是一个赶场季，一天少则几家、十几家企业入校招聘，多则几十家、上百家企业招兵买马。企业多，选项自然也多，这固然是一件好事情，但由于招聘企业实在是太多，自然而然会导致另外一个问题：同一天企业扎堆面试，且都是自己心仪或欣赏的大企业。如果不能够提前掌握企业的宣讲时间、地点，是很容易迟到或错过的。但有时候即使掌握了宣讲时间、笔试和面试时间，还是有可能错过，为什么呢？时间冲突，人不可能具有分身术，也不可能同一时间做两件不同的事情，所以，很多时候就必须有所取舍了。

到底该如何取舍呢？该如何应对这种时间冲突的问题呢？在此，编者将自己的一些想法和经验分享出来，供读者参考：

1）如果多家心仪企业的校园宣讲时间发生冲突（前提是只宣讲，不笔试，否则请看后面的建议），此时最好的解决方法是和同学或朋友商量好，各去一家，然后大家进行信息共享。

2）如果多家心仪企业的笔试时间发生冲突，此时只能选择其一，毕竟企业的笔试时间都是考虑到了成百上千人的安排，需要提前安排考场、考务人员和阅卷人员等，不可能为了某一个人而轻易改变。所以，最好选择自己更有兴趣的企业参加笔试。

3）如果多家心仪企业的面试时间发生冲突，不要轻易放弃。对于面试官而言，面试任何人都是一样的，因为面试官谁都不认识，而面试时间也是灵活性比较大的，一般可以通过电话协商。求职者可以与相关工作人员（一般是企业的 HR）进行沟通，以某种理由（例如学校的事宜、导师的事宜或家庭的事宜等，前提是必须能够说服人，不要给出的理由连自己都说服不了）让其调整时间，一般都能协调下来。但为了保证协调的成功率，一般要接到面试通知后第一时间联系相关工作人员变更时间，这样他们协调起来也更方便。

正如世界上没有能够包治百病的药物一样，以上这些建议在应用时，很多情况下也做不到全盘兼顾，当必须进行多选一的时候，求职者就要对此进行评估了，评估的项目可以包括：对企业的中意程度、获得 offer 的概率及去工作的可能性等。评估的结果往往具有很强的参考性，求职者依据评估结果做出的选择一般也会比较合理。

经验技巧 6 在被企业拒绝后是否可以再申请?

很多企业为了能够在一年一度的招聘季节中，提前将优秀的程序员招至自己的麾下，往往会先下手为强。他们通常采取的措施有以下两种：第一种，招聘实习生；第二种，多轮招聘。

招聘开始后，往往是几家欢喜几家愁，提前拿到企业绿卡的，于是对酒当歌、欢天喜地，而没有被选上的，担心从此与这家企业无缘了，于是整日囧字写在脸上，忧心忡忡，感叹生不逢时。难道一次失望的表现就永远会被企业拉入黑名单了吗？难道一次失败的经历就会永远被记录在个人历史的耻辱柱上了吗？

答案当然是否定的，对心仪的女孩表白，即使第一次被拒绝了，都还可以一而再再而三地表白呢！多次表白后成功的案例比比皆是，更何况是求职找工作。一般而言，企业是不会记仇的，尤其是知名的大企业，对此都会有明确表示。如果在企业的实习生招聘或在企业

以前的招聘中不幸被 pass 掉了，一般是不会被拉入企业的黑名单的。在下一次招聘中，和其他求职者具有相同的竞争机会(有些企业可能会要求求职者等待半年到一年时间才能再应聘该企业，但上一次求职的糟糕表现不会被计入此次招聘中)。

对心仪的对象表白被拒绝了，不是一样还可以继续表白吗？也许是在考验，也许是在等待，也许真的是拒绝，但无论出于什么原因，此时此刻都不要对自己丧失信心。工作也是如此，以编者身边的很多同学和朋友为例，很多人最开始被一家企业拒绝了，过了一段时间，又发现他们已成为该企业的员工。所以，即使被企业拒绝了也不是什么大不了的事情，以后还有机会的，有志者自有千计万计，无志者只感千难万难，关键是看你愿意成为什么样的人了。

经验技巧 7　如何应对自己不会回答的问题？

在面试的过程中，求职者对面试官提出的问题并不是每个问题都能回答上来。计算机技术博大精深，很少有人能对计算机技术的各个分支学科了如指掌，而且抛开技术层面的问题，在面试那种紧张的环境中，回答不上来的情况也容易出现。面试的过程是一个和面试官“斗智斗勇”的过程，遇到自己不会回答的问题时，错误的做法是保持沉默或者支支吾吾、不懂装懂，硬着头皮胡乱说一通，这样会使面试气氛很尴尬，很难再往下继续进行。

其实面试遇到不会的问题是一件很正常的事情，没有人是万事通，即使对自己的专业有相当的研究与认识，也可能会在面试中遇到感觉没有任何印象、不知道如何回答的问题。在面试中遇到实在不懂或不会回答的问题，正确的办法是本着实事求是的原则，态度诚恳，告诉面试官不知道答案。例如，“对不起，不好意思，这个问题我回答不出来，我能向您请教吗?”

征求面试官的意见时可以说说自己的个人想法，如果面试官同意听了，就将自己的想法说出来，回答时要谦逊有礼，切不可说个没完。然后应该虚心地向面试官请教，表现出强烈的学习欲望。

所以，遇到自己不会的问题时，正确的做法是：“知之为知之，不知为不知”，不懂就是不懂，不会就是不会，一定要实事求是，坦然面对。最后也能给面试官留下诚实、坦率的好印象。

经验技巧 8　如何应对面试官的“激将法”语言？

“激将法”是面试官用以淘汰求职者的一种惯用方法，它是指面试官采用怀疑、尖锐或咄咄逼人的交流方式来对求职者进行提问的方法。例如，“我觉得你比较缺乏工作经验”“我们需要活泼开朗的人，你恐怕不合适”“你的教育背景与我们的需求不太适合”“你的成绩太差”“你的英语没过六级”“你的专业和我们不对口”“为什么你还没找到工作”“你竟然有好多门课不及格”等，很多求职者遇到这样的问题，会很快产生我是来面试而不是来受侮辱的想法，往往会被“激怒”，于是奋起反抗。千万要记住，面试的目的是要获得工作，而不是要与面试官争个高低，也许争辩取胜了，却失去了一份工作。所以对于此类问题求职者应该进行巧妙的回答，一方面化解不友好的气

氛，另一方面得到面试官的认可。

具体而言，受到这种“激将”时，求职者首先应该保持清醒的头脑，企业让你来参加面试，说明你已经通过了他们第一轮的筛选，至少从简历上看，已经表明你符合求职岗位的需要，企业对你还是感兴趣的。其次，做到不卑不亢，不要被面试官的思路带走，要时刻保持自己的思路和步调。此时可以换一种方式，如介绍自己的经历、工作和优势，来表现自己的抗压能力。

针对面试官提出的非名校毕业的问题，可以巧妙地回答：比尔·盖茨也并非毕业于哈佛大学，但他一样成为世界首富，成为举世瞩目的人物。针对缺乏工作经验的问题，可以回答：每个人都是从没经验变为有经验的，如果有幸最终能够成为贵公司的一员，我将很快成为一个经验丰富的人。针对专业不对口的问题，可以回答：专业人才难得，复合型人才更难得，在某些方面，外行的灵感往往超过内行，他们一般没有思维定式，没有条条框框。面试官还可能提问：你的学历对我们来讲太高了。此时也可以很巧妙地回答：今天我带来的 3 张学历证书，您可以从中挑选一张您认为合适的，其他两张，您就不用管了。针对性格内向的问题，可以回答：内向的人往往具有专心致志、锲而不舍的品质，而且我善于倾听，我觉得应该把发言机会更多地留给别人。

面对面试官的“挑衅”行为，如果求职者回答得结结巴巴，或者无言以对，抑或怒形于色、据理力争，那就掉进了对方所设的陷阱，所以当求职者碰到此种情况时，最重要的一点就是保持头脑冷静，不要过分较真，以一颗平淡的心对待。

经验技巧 9　如何处理与面试官持不同观点这个问题?

在面试的过程中，求职者所持有的观点不可能与面试官一模一样，在对某个问题的看法上，很有可能两个人相去甚远。当与面试官持不同观点时，有的求职者自作聪明，立马就反驳面试官，例如，“不见得吧!”“我看未必”“不会”“完全不是这么回事!”“这样的说法未必全对”等，其实，虽然也许确实不像面试官所说的，但是太过直接的反驳往往会导致面试官心里的不悦，最终的结果很可能是“逞一时之快，失一份工作”。

就算与面试官持不一样的观点，也应该委婉地表达自己的真实想法，因为我们不清楚面试官的度量，碰到心胸宽广的面试官还好，万一碰到了“小心眼”的面试官，他和你较真起来，吃亏的还是自己。

所以回答此类问题的最好方法往往是应该先赞同面试官的观点，给对方一个台阶下，然后再说明自己的观点，用“同时”“而且”过渡，千万不要说“但是”，一旦说了“但是”“却”就容易把自己放在面试官的对立面去。

经验技巧 10　什么是职场暗语?

随着求职大势的变迁发展，以往常规的面试套路，因为过于单调、简明，已经被众多“面试达人”们挖掘出了各种“破解秘诀”，形成了类似“求职宝典”的各类“面经”。所谓“道高一尺，魔高一丈”，面试官们也纷纷升级面试模式，为求职者们制作了更为隐蔽、间接、含混、“下套”的面试题目，让那些早已流传开来的“面试攻略”毫无用武之地，

一些蕴涵丰富信息但以更新面目出现的问话屡屡“秒杀”求职者，让求职者一头雾水，掉进了陷阱里面还以为吃到肉了，例如，“面试官从头到尾都表现出对我很感兴趣的样子，营造出马上就要录用我的氛围，为什么我最后还是被拒了？”“为什么 HR 会问我一些与专业、能力根本无关的怪问题，我感觉回答得也还行，为什么最后还是被拒了？”其实，这都是没有听懂面试“暗语”，没有听出面试官“弦外之音”的表现。“暗语”已经成为一种测试求职者心理素质、挖掘求职者内心真实想法的有效手段。理解这些面试中的暗语，对于求职者而言，不可或缺。

以下是一些常见的面试暗语，求职者一定要弄清楚其中蕴含的深意，不然可能“躺着也中枪”，最后只能铩羽而归。

（1）我不是人力资源的，你别拘束，咱们就当是聊天，随便聊聊

一般来说，能当面试官的人都是久经沙场的老将，不太好对付。表面上彬彬有礼，看上去笑眯眯、很和气的样子，说起话来可能偶尔还带点小结巴，但没准儿一肚子“坏水”，巴不得下个套把你套进去。所以，作为求职者，千万不能被眼前的这种“假象”所迷惑，而应该时刻保持高度警觉，面试官不经意间问出来的问题，看似随意，很可能是他最想知道的。所以千万不要把面试过程当作聊天，不要把面试官提出的问题当作是普通问题，而应该对每一个问题都仔细思考，认真回答，切忌不经过大脑的随意接话和回答。

（2）是否可以谈谈你的要求和打算

面试官在翻阅了求职者的简历后，说出这句话，很有可能是对求职者有兴趣，此时求职者应该尽量全方位地表现个人水平与才能，但也不能自卖自夸引起对方的反感。

（3）面试时只是“例行公事”式的问答

如果面试时只是“例行公事”式的问答，没有什么激情或者主观性的赞许，此时希望就很渺茫了；但如果面试官对你的专长问得很细，而且表现出一种极大的关注与热情，那么此时希望会很大。作为求职者，一定要抓住机会，将自己最好的一面展示在面试官面前。

（4）你好，请坐

简单的一句话，从面试官口中说出来其含义就大不同了。一般而言，面试官说出此话，求职者回答“你好”或“您好”不重要，重要的是求职者是否“礼貌回应”和“坐不坐”。有的求职者的回应是“你好”或“您好”后直接落座，也有求职者回答“你好，谢谢”或“您好，谢谢”后落座，还有求职者一声不吭就坐下去，极个别求职者回答“谢谢”但不坐下来。前两种方法都可接受，但后两者都不可接受。通过问候语，可以体现一个人的基本修养，直接影响在面试官心目中的第一印象。

（5）面试官向求职者探过身去

在面试的过程中，面试官会有一些肢体语言，了解这些肢体语言对于了解面试官的心理情况以及面试的进展情况非常重要。例如当面试官向求职者探过身去时，一般表明面试官对求职者很感兴趣；当面试官打呵欠或者目光呆滞、游移不定，甚至打开手机看时间或打电话、接电话时，一般表明面试官此时有了厌烦的情绪；而当面试官收拾文件或从椅子上站起来，一般表明此时面试官打算结束面试。针对面试官的肢体语言，求职者也应该迎合他们：当面试官很感兴趣时，应该继续陈述自己的观点；当面试官厌烦时，此时最好停下来，询问面试官是否愿意再继续听下去；当面试官打算结束面试，领会其用意，并准备好收场白，尽快地