

< / > 在这里 / 有面试笔试常见技巧的提炼与总结;
< / > 在这里 / 有面试笔试高频前端知识点的整理与剖析;
< / > 在这里 / 有面试笔试历年前端真题的解答与拓展。

PROGRAMMER

> Interview and written examination

前端程序员 > 面试笔试宝典

猿媛之家 / 组编 平文 等 / 编著



本书覆盖了近**3年**程序员面试笔试中超过**98%**的前端高频知识点

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 > .

前端程序员面试笔试宝典

猿媛之家 组编

平 文 等编著



机械工业出版社

本书是一本前端程序员面试笔试的应试类用书，在内容上，除了讲解如何解答前端程序员面试笔试问题，还引入了相关知识点辅以说明，让读者能够更加容易理解。

本书包括前端程序员面试笔试过程中各类知识点，在题目的广度上，搜集了近3年来多家IT企业针对前端岗位的笔试面试真题。在讲解的深度上，本书由浅入深，庖丁解牛式地分析每一个知识点，并提炼归纳，同时，引入相关知识点，并对其进行深度剖析，让读者不仅能够理解这个知识点，还能在遇到相似问题的时候，也能游刃有余地解决。本书根据知识点进行分类，结构合理，条理清晰，便于读者学习与查阅。

图书在版编目(CIP)数据

前端程序员面试笔试宝典 / 猿媛之家组编；平文等编著. —北京：机械工业出版社，2018.9

ISBN 978-7-111-60747-2

I. ①前… II. ①猿… ②平… III. ①程序设计—资格考试—题解
IV. ①TP311.1-44

中国版本图书馆CIP数据核字(2018)第194494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：时 静 责任编辑：时 静

责任校对：张艳霞 责任印制：张 博

三河市国英印务有限公司印刷

2018年9月第1版·第1次印刷

184mm×260mm·19.75印张·484千字

0001—3000册

标准书号：ISBN 978-7-111-60747-2

定价：69.00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

网络服务

服务咨询热线：(010) 88361066

机工官网：www.cmpbook.com

读者购书热线：(010) 68326294

机工官博：weibo.com/cmp1952

(010) 88379203

教育服务网：www.cmpedu.com

封面无防伪标均为盗版

金书网：www.golden-book.com

前言

计算机技术博大精深，日新月异。Hadoop、GPU 计算、移动互联网、模式匹配、图像识别、神经网络、蚁群算法、大数据、机器学习、人工智能、深度学习等新技术让人眼花缭乱，稍有不慎，就会被时代抛弃。于是，很多 IT 从业者就开始困惑了，不知道从何学起，到底什么才是计算机技术的基石。其实，究其本质，还是最基础的计算机语言知识、数据结构与算法知识。所以，无论是世界级的大型企业还是几个人的小公司，在面试求职者的时候，往往都会考察这些最基础的知识，无论你的研究方向是什么，这些基础知识都是应该熟练掌握的。

本书在写作风格上，推陈出新，对于前端知识点的讲解，不仅有文字描述，更以示例佐证（示例代码可以从 <https://github.com/pwstrick/FrontEndInterviewCode> 下载）。为了能够写出精品书籍，我对每一个技术问题，都反复推敲，与技术大牛一起反复论证其可行性；对每一句话，都咬文嚼字，字斟句酌，所有这些付出，只为了让读者能够对书中技术点放心，文字描述舒心。

本书在选择题目时也下了巨大的工夫：首先，搜集近 3 年来多家 IT 企业的面试笔试真题，包括已经出版的其他著作、技术博客、在线编码平台、刷题网站等，保证所选样本足够大。其次，在选择题目的时候，尽可能不选择那种一眼就能知道结果的简单题，也不选择怪题、偏题、难题。选题原则是选择难度适中或者看上去简单但实际容易被坑的题目，力求遴选出来的真题能够最大限度地帮助读者。在真题的解析上，采用层层递进的写法，先易后难，层层深入，将问题抽丝剥茧，使读者能够跟随书中的思路，一步步找到问题的最优解。

写作的过程是一个自我提高、自我认识的过程，很多知识，只有你深入理解与剖析后，才能领悟其中的精髓，掌握其中的技巧，程序员求职面试也不例外。本书不仅具备了其他书籍分析透彻，代码清晰合理等优点，还具备以下两个方面的优势：

第一，知识覆盖面全。本书涉及知识点覆盖了前端程序员面试笔试中至少 98% 的基础知识点。

第二，讲解详细，剖析深入。本书对每个知识点都进行了详细阐述，知识点介绍完以后还会针对部分真题进行深度分析与讲解，图文并茂，生动形象。

我在相关技术岗位上已经有多年的经验。一直想把自己的心得分享给大家，机缘巧合促成了这本书的撰写。在此，要感谢与出版本书有关的人，因为有你们，我才能坚持下来，完成整本书的编写。

首先，感谢机械工业出版社时静老师给我写作的机会。其次，要感谢的是楚秦，他不但让我加入到这项工作中，还审阅了整本书，并对文字和代码进行了修改和润色，为改进本书提供了许多宝贵的建设性意见，这些建议与意见极大地提高了本书的质量。

除此之外，也感谢那些给予我热情帮助的人，从他们那里亦得到了很多非常好的建议，这些人是（按姓氏拼音排列）：陈安阳、陈曼杰、陈涛、潘义璠、沈哲俊、王春明、王汝婷、夏丽、赵茹林、周捷、周山。还要特别感谢周晶撰写序。

最后，我要感谢我的家人，他们是我生命中最重要的人，感谢他们对我的理解和鼓励。尤其要感谢我的爱妻，一直陪伴在我身边，在我感到困难的时候支持我、鼓励我，为我营造了一个安心、舒适的写作环境，让我有信心完成整本书的写作。

其实，本书不仅可以作为程序员求职的应试类书籍，还可以作为前端程序员的教辅书籍进行学习。

书中的很多思想、方法对于提高对前端知识的理解大有裨益，无论你是本科生还是研究生，无论你是低年级学生还是高年级学生，无论你对计算机底层知识或是当前的计算机前沿知识是否了解，都不影响你学好本书。

本书中有部分思想来源于网络上的无名英雄，无法追踪到原始出处，在此对这些幕后英雄致以最高的敬意。没有学不好的学生，只有教不好的老师，希望无论是什么层次的学生，都能毫无障碍地看懂书中所讲内容。如果读者存在求职困惑或是对书中的内容存在异议，都可以通过 yuancoder@foxmail.com 联系作者。

平 文
于上海松江

目 录

序

前言

面试笔试经验技巧篇

经验技巧 1	如何巧妙地回答面试官的问题？	2
经验技巧 2	如何回答技术性的问题？	2
经验技巧 3	如何回答非技术性问题？	4
经验技巧 4	如何回答快速估算类问题？	5
经验技巧 5	如何回答算法设计问题？	5
经验技巧 6	如何回答系统设计题？	7
经验技巧 7	如何解决求职中的时间冲突问题？	9
经验技巧 8	如果面试问题曾经遇到过，是否要告诉面试官？	10
经验技巧 9	被企业拒绝后是否可以再申请？	10
经验技巧 10	如何应对自己不会回答的问题？	11
经验技巧 11	如何应对面试官的“激将法”语言？	11
经验技巧 12	如何处理与面试官持不同观点这个问题？	12
经验技巧 13	职场暗语有哪些？	12
经验技巧 14	当前市场对前端工程师的需求如何？待遇如何？	15
经验技巧 15	前端工程师未来的发展方向如何？	15
经验技巧 16	前端工程师有哪些可供选择的职业发展道路？	16
经验技巧 17	企业在招聘时，对前端工程师通常有何要求？前端工程师的日常工作是什么？	16
经验技巧 18	要想成为一名出色的前端工程师，需要掌握哪些必备的知识？有哪些好的书籍或网站可供推荐学习？	17

面试笔试技术攻克篇

第 1 章	HTML 基础	20
1.1	HTML	20
1.2	HTML5	21
1.2.1	HTML5 新特性	21
1.2.2	Web App、Hybrid App 和 Native App	21
1.3	DOCTYPE	23
1.3.1	语法	23
1.3.2	常用声明	23

1.3.3	浏览器渲染模式	24
1.4	XHTML	25
1.4.1	XHTML 规范	25
1.4.2	HTML 与 XHTML 的区别	25
1.5	语义化	26
1.5.1	语义化的元素	26
1.5.2	微格式	27
1.6	HTML 实体	28
1.6.1	HTML 实体的定义	28
1.6.2	可转义的字符或符号	29
第 2 章	HTML 元素和高级功能	30
2.1	元素基础	30
2.1.1	元素的分类	30
2.1.2	元素属性	31
2.2	应用 CSS 样式	32
2.2.1	内联样式	32
2.2.2	内嵌样式	32
2.2.3	外部样式	33
2.3	嵌入 JavaScript	33
2.3.1	内联脚本	33
2.3.2	外部脚本	34
2.3.3	元素属性	35
2.4	meta 元素	36
2.4.1	charset	36
2.4.2	name	36
2.4.3	http-equiv	37
2.5	超链接	38
2.5.1	href	38
2.5.2	target	38
2.5.3	其他属性	39
2.6	图像	39
2.6.1	属性	40
2.6.2	分区响应图	40
2.6.3	插图元素	41
2.7	收集用户数据	42
2.7.1	文本	42
2.7.2	日期与数值	43
2.7.3	按钮和其他	44
2.7.4	给表单控件分组	45
2.8	表格	46
2.8.1	表格的组成	46
2.8.2	属性	47
2.9	iframe	48
2.9.1	iframe 属性	48

2.9.2	iframe 用途	49
2.10	多媒体	50
2.10.1	使用多媒体元素的优势	50
2.10.2	video	51
2.10.3	audio	52
2.11	绘图	53
2.11.1	位图图像与矢量图形	53
2.11.2	canvas	53
2.11.3	SVG	54
2.12	数据存储	55
2.12.1	Cookie	55
2.12.2	Web 存储	56
2.12.3	userData	56
第 3 章	CSS 基础	57
3.1	CSS3	57
3.1.1	CSS3 新特性	57
3.1.2	渐进增强	58
3.2	盒模型	59
3.2.1	盒模型	60
3.2.2	box-sizing	60
3.2.3	盒子的显示类型	61
3.3	元素盒类型	62
3.3.1	list-item	62
3.3.2	表格相关的属性值	63
3.3.3	run-in	63
3.3.4	inline-block	63
3.3.5	伸缩盒	64
3.4	BFC	65
3.4.1	创建 BFC	65
3.4.2	BFC 的用途	66
3.5	使用 CSS 选择器	68
3.5.1	基本选择器	68
3.5.2	关系选择器	69
3.5.3	伪选择器	69
3.5.4	选择器分组	70
3.6	内容生成	71
3.6.1	计数器	71
3.6.2	引用属性值和图像	72
3.6.3	添加文本	73
3.7	层叠	73
3.8	单位	76
3.8.1	绝对长度单位	76
3.8.2	相对长度单位	76
3.8.3	其他单位	78

3.9	百分数	78
3.9.1	定位	79
3.9.2	宽和高	79
3.9.3	外边距和内边距	80
3.9.4	边框圆角和位移	80
3.9.5	字体大小	81
3.10	颜色	82
3.10.1	颜色名称	82
3.10.2	RGB 颜色	82
3.10.3	HSL 颜色	83
3.10.4	Web 安全色	83
第 4 章	CSS 属性	84
4.1	浮动	84
4.1.1	浮动范围	84
4.1.2	创建 BFC	85
4.1.3	负外边距	85
4.1.4	清除浮动	86
4.2	定位	88
4.2.1	相对定位	88
4.2.2	绝对定位	89
4.2.3	固定定位	89
4.2.4	偏移属性	90
4.2.5	z-index	91
4.3	边框	92
4.3.1	外观	92
4.3.2	宽度	92
4.3.3	颜色	93
4.3.4	圆角	93
4.3.5	阴影	95
4.3.6	outline	95
4.4	文本属性	96
4.4.1	overflow	96
4.4.2	text-decoration	97
4.4.3	white-space	97
4.4.4	文本换行	98
4.5	字体	99
4.5.1	字体系列	100
4.5.2	Web 字体	100
4.6	垂直对齐	102
4.6.1	行内非替换元素	102
4.6.2	行内替换元素	103
4.6.3	垂直对齐	104
4.7	背景	105
4.7.1	起始点和裁剪背景区	105

4.7.2	背景图像尺寸	107
4.7.3	背景图像附着	108
4.7.4	背景图像定位	109
4.7.5	background	110
4.8	变形、过渡和动画	111
4.8.1	变形	111
4.8.2	过渡	112
4.8.3	动画	114
4.9	媒体查询	117
4.9.1	使用方法	117
4.9.2	媒体类型	117
4.9.3	媒体特性	118
4.9.4	操作符	119
4.9.5	支持度	119
第5章	CSS 应用	121
5.1	CSS Hack	121
5.1.1	浏览器前缀	121
5.1.2	条件注释	121
5.1.3	CSS 属性级前缀	122
5.2	布局	123
5.2.1	浮动布局	123
5.2.2	定位布局	124
5.2.3	流式布局	124
5.2.4	弹性布局	125
5.2.5	多列布局	125
5.2.6	等高布局	127
5.3	CSS Reset	129
5.3.1	全局重置	129
5.3.2	Reset.css	129
5.3.3	Normalize.css	130
5.4	伸缩盒布局	131
5.4.1	主轴和侧轴	132
5.4.2	对齐方式	133
5.4.3	伸缩性	135
5.4.4	显示顺序	137
5.4.5	新旧版本属性对照	137
5.5	居中	138
5.5.1	水平居中	139
5.5.2	垂直居中	140
5.6	CSS 预处理器	142
5.6.1	变量与运算	142
5.6.2	选择器嵌套	143
5.6.3	控制语句	143
5.6.4	混合和函数	144

5.6.5	继承	144
5.7	Bootstrap	145
5.7.1	栅格系统	146
5.7.2	排版	148
5.7.3	颜色	148
5.7.4	表格	149
5.7.5	表单	150
5.7.6	组件	151
第6章	计算机网络	153
6.1	TCP/IP	153
6.1.1	协议	153
6.1.2	TCP/IP	154
6.2	HTTP	155
6.2.1	URI 和 URL	156
6.2.2	HTTP 协议	156
6.2.3	HTTP 报文	157
6.2.4	HTTP 首部	158
6.2.5	缓存	160
6.3	RESTful 架构风格	162
6.3.1	REST	162
6.3.2	约束条件	162
6.4	TCP	163
6.4.1	连接管理	163
6.4.2	确认应答	164
6.4.3	窗口控制	166
6.4.4	重传控制	166
6.5	HTTPS	167
6.5.1	加密	168
6.5.2	数字签名	169
6.5.3	数字证书	169
6.5.4	安全通信机制	170
6.6	HTTP/2.0	171
6.6.1	二进制分帧层	171
6.6.2	多路通信	172
6.6.3	请求优先级	173
6.6.4	服务器推送	173
6.6.5	首部压缩	173
第7章	JavaScript 语言	175
7.1	JavaScript 概述	175
7.1.1	ECMAScript	175
7.1.2	DOM	176
7.1.3	BOM	176
7.2	基本语法	177
7.2.1	字符集	177

7.2.2	标识符、关键字和保留字	177
7.2.3	数据类型	178
7.2.4	运算符	179
7.2.5	表达式	181
7.2.6	语句	181
7.3	数字和字符串	183
7.3.1	数字	183
7.3.2	字符串	185
7.4	强制类型转换	188
7.4.1	转换为数字	188
7.4.2	转换为字符串	190
7.4.3	转换为布尔值	191
7.4.4	相等运算符	191
7.5	全局对象	192
7.5.1	全局对象	193
7.5.2	包装对象	193
7.5.3	处理 URI 的全局函数	194
7.6	对象 (Object)	195
7.6.1	创建	195
7.6.2	原型和原型链	195
7.6.3	属性	196
7.6.4	可扩展性	200
7.7	JSON	202
7.7.1	语法	203
7.7.2	序列化	204
7.7.3	解析	205
7.8	日期和时间 (Date)	206
7.8.1	UTC 和 GMT	206
7.8.2	构造函数	206
7.8.3	静态方法	207
7.8.4	日期格式化方法	208
7.8.5	其他日期方法	208
7.9	正则表达式 (RegExp)	209
7.9.1	创建	209
7.9.2	语法	210
7.9.3	String 中的方法	212
7.9.4	RegExp 中的方法	214
7.10	数组 (Array)	215
7.10.1	创建	215
7.10.2	数组操作	215
7.10.3	数组方法	216
7.10.4	类数组对象和字符串	220
7.11	函数 (Function)	221
7.11.1	作用域	222

7.11.2	函数基本概念	223
7.11.3	闭包	227
7.11.4	函数式编程	228
7.12	this	230
7.13	即时函数	232
7.13.1	块级作用域	233
7.13.2	循环	233
7.13.3	可读性和 undefined	233
7.13.4	类库封装	234
7.14	检测类型和对象	234
7.14.1	检测数据的类型	235
7.14.2	对象之间的关联性	236
7.15	Node.js	237
7.15.1	安装 Node	237
7.15.2	npm	238
第 8 章	客户端中的 JavaScript	239
8.1	BOM	239
8.1.1	Window	239
8.1.2	Location	242
8.1.3	Navigator	243
8.1.4	History	245
8.2	DOM	247
8.2.1	节点	247
8.2.2	操作元素	250
8.3	元素的属性、尺寸和坐标	256
8.3.1	特性和属性	256
8.3.2	尺寸	258
8.3.3	坐标	261
8.4	控制 CSS 样式	264
8.4.1	内联样式	264
8.4.2	获取 CSS 属性	264
8.4.3	设置 CSS 属性	265
8.4.4	读写 CSS 类	265
8.4.5	计算样式	266
8.5	事件	267
8.5.1	术语	267
8.5.2	注册事件	268
8.5.3	事件传播	271
8.5.4	事件对象	272
8.5.5	模拟事件	273
8.6	表单	275
8.6.1	表单	275
8.6.2	单选框和复选框	277
8.6.3	选择框	278

8.6.4	上传按钮	280
8.7	Ajax	282
8.7.1	1 级 XMLHttpRequest	282
8.7.2	2 级 XMLHttpRequest	285
8.7.3	跨域通信	288
8.8	jQuery	290
8.8.1	jQuery 基础	290
8.8.2	操作元素	291
8.8.3	元素的属性	295
8.8.4	事件	298
8.8.5	Ajax	299
8.8.6	动画	300
8.8.7	工具函数	301

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 18 个常见问题进行深度解析，并且结合实际情景，给出了较为合理的参考答案以供读者学习与应用。掌握这 18 个问题的解答精髓，对求职者大有裨益。

经验技巧 1

如何巧妙地回答面试官的问题?

所谓“来者不善，善者不来”，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。回答面试官的问题千万不能简单地回答“是”或者“不是”，而应该具体分析“是”或者“不是”的原因。

回答面试官的问题是一门很深的学问。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不至于撞上枪口呢？如何才能让自己的回答结果令面试官满意呢？

谈话是一门艺术，回答问题也是一门艺术，同样的话，不同的回答方式，往往会产生不同的效果，甚至是截然相反的效果。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自负，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 $O(\log n)$ ，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，其他三个人中，两个人能力很强，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，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，有根有据，让面试官一目了然。

回答问题的技巧是一门大学问。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题真题，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”，完全就是一个 Bug，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六个步骤来分析解决。

（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，这样做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，解决方案自然也就出来了。

（2）高效设计

对于技术性问题，如何才能打动面试官？完成基本功能是必需的，仅此而已吗？显然不是，完成基本功能顶多算及格水平，要想达到优秀水平，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，求职者可以首先征得面试官的同意，在编写实际代码前，写一段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，否则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 20min 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，认真检查一些边界情况、异常情况、极值情况等，看是否也能满足要求。

（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，最好严格遵循编码规范，注意函数变量命名、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