随着各大公司春招的开始，很多小伙伴都行动起来了，我有幸能够加入百度并和大家分享自己的经验心得。由于我面试的都是比较大的公司，所以自然也是做了这方面的准备，因此这篇总结并不一定适合想去创业公司的同学。另外，由于经验本来就是主观性极强的东西，加之笔者水平有限，所以如果有不认可的地方，万望诸君呵呵一笑，抛之脑后。

接下来，我就斗胆分享一下自己在准备和参加面试的过程中的收获、对面试的思考，以及一些可能对大家有用的建议。最后附赠一份大礼包，希望能帮助每位读者找到自己心仪的工作。

什么是面试

有些人可能会把面试看的太重，觉得面试过了就能进入大厂，技术和财富兼得……

我倒是觉得，面试没有这么夸张（抱歉做了一回标题党），它其实是一次你和面试官互相了解的绝佳机会，借此机会你还可以对未来的工作有初步的了解。

面试本身并不能完全评价一人个的实力。面试通过的人，也许只是恰好在面试时遇到了自己熟悉的问题，面试不通过，也有可能是面试官自身的问题，并非每个面试官都具备客观评价别人的能力。

换句话说，面试没通过也许是面试官没有发现你的才华，面试通过了也并不代表你就能胜任工作，因为进入企业之后可不是每天负责回答面试题！

所以从这一点来看，面试有点像相亲。你满意我，我满意你，王八对绿豆——看上眼了，那就一拍即合，否则就分道扬镳。我本人非常希望能够多几轮面试（实际并不总是能做到），这样大家都有充足的时间互相了解，决定去留。

网上某些面经中，介绍了一些“装逼”的方法，还有所谓的“面试技巧”，我是不太认可的。技巧需要有，这是为了让你更好的展示自己，而非坑蒙拐骗，无理取闹，无中生有。我更想展现一个真实的自己，如果面试官不认可，说明我们没有缘分，或者说自己的能力还不够。

面试要准备什么

有一位小伙伴面试阿里被拒后，面试官给出了这样的评价：“……计算机基础，以及编程基础能力上都有所欠缺……”。但这种笼统的回答并非是我们希望的答案，所谓的基础到底指的是什么？

作为一名 iOS 开发者，我所理解的基础是 操作系统、网络和算法这三大块，不同的开发方向可能有不同的侧重，但基础总的来说就是这些。我不推荐通过去网上看教程来学习这些基础知识，因为能用短短几篇文章讲明白的事情不叫基础，至少我没见过写得这么深入浅出的文章。

不知道有多少读者和我一样有过这样的困扰：“我知道某些东西很重要，所以去百度查了资料，但是查到的文章质量很差，正确率没有保证”。这其实是正常的，优秀的文章一般都放在优秀的作者的个人博客上，这恰恰是搜索引擎的盲区，所以一般只能搜到 CSDN、博客园这种地方的文章。自然就无法保证文章质量。

出于这种考虑，我在文章最后的复习资料中，提供了用于学习相关基础知识的书籍，如果您恰好是 iOS 开发者，还可以阅读我收集的一些高质量文章，正确性比较有保证（我写的除外）。

除了准备通用的基础知识以外，简历也是一个很重要的环节。一直很仰慕唐巧老师的猿题库，无奈简历太差，都没有收到面试邀请。后来好好改了简历以后，就没有这种问题了。关于简历的书写，推荐两篇文章：如何写面向互联网公司的求职简历、程序猿简历模板。你也可以参考我的简历，没有亮点，就当是抛砖引玉。

最后，当然是准备好相关岗位的基础知识了。作为 iOS 开发者，虽然 Swift 已经发布了快两年，但是大公司转向 Swift 的动作还不明显，所以 Objective-C 几乎是必备项，Swift 都不一定能算是加分项。iOS 方面的知识也必不可少，虽然招聘信息上写着如果基础扎实，零 iOS 基础也可以，但是现实往往是比较残酷的。

我的面试经历

扯了这么多，终于进入正题了，分享一下我的面试经历。题目如下，**破折线后面是简单的解决思路**。

百度

一面：约 1.5 小时

首先是四个算法题：

1. 不用临时变量怎么实现 swap(a, b)——用加法或者异或都可以
2. 二维有序数组查找数字——剑指 offer 第 3题
3. 亿级日志中，查找登陆次数最多的十个用户——（不确定对不对，我的思路是）先用哈希表保存登陆次数和ID，然后用红黑树保存最大的十个数。剑指 offer 第 30题
4. 简述排序算法——快排，`partion` 函数的原理，堆排（不稳定），归并排序，基数排序。

然后有一个智力题，没完整的答出来，好像影响不是很大。

最后是 iOS 相关，面试官问的很开放，都是谈谈自己的理解：

1. 说说你对 OC 中 `load` 方法和 `initialize` 方法的异同。——主要说一下执行时间，各自用途，没实现子类的方法会不会调用父类的?
2. 说说你对 block 的理解。—— 三种 block，栈上的自动复制到堆上，block 的属性修饰符是 copy，循环引用的原理和解决方案。
3. 说说你对 runtime 的理解。——主要是方法调用时如何查找缓存，如何找到方法，找不到方法时怎么转发，对象的内存布局。
4. 说说你对 MVC 和 MVVM 的理解。—— MVC 的 C 太臃肿，可以和 V 合并，变成 MVVM 中的 V，而 VM 用来将 M 转化成 V 能用的数据。
5. 说说 UITableView 的调优。——一方面是通过 instruments 检查影响性能的地方，另一方面是估算高度并在 runloop 空闲时缓存。

6. 谈谈你对 ARC 的理解。ARC 是编译器完成的，依靠引用计数，谈谈几个属性修饰符的内存管理策略，什么情况下会内存泄露。

一面的问题非常基础，主要是算法和 Objective-C，因为准备比较充分，基本上答出来 80% 吧。大约一周后突然二面。

二面：约 0.5 小时

二面比较突然，显示简单的自我介绍，然后问了三个问题：

1. 野指针是什么，iOS 开发中什么情况下会有野指针？——野指针是不为 nil，但是指向已经被释放的内存的指针，不知道什么时候会有，如果有知道的读者还望提醒。
2. 介绍 block。—— (接第一问) 我让面试官提示我一下什么时候会有野指针，他说用 block 时，我表示还是不知道，只知道 block 会有循环引用。于是就扯回了一面的问题。
3. 说说你是怎么优化 UITableView 的。——还是一面的问题。。。。。。。。。。。

虽然通过了，但是几乎又问了一遍一面的问题让我感觉对方不太认真。

三面：北京 onsite，约 2.5 小时

首先是给一个小时，手写算法：

1. 给一个字符串，如何判断它是否是合法的 IP 地址，比如 "192.168.1.1" 就是合法的。
2. 说说大数相加的思路，动手写代码实现。

没能写完，主要是大数相加的时候需要考虑负数，耽搁了一点时间。

然后让我简述 TCP 建立和关闭连接时，握手的过程。还问了前者为什么是三次，后者需要四次?

接下来是设计了一个实际场景，为了简化问题，我们考虑这个问题：假设有 10W 条电话号码，如何通过输入电话号码的某一段内容，快速搜索出来。比如输入 `234`，以下两个号码都会显示在搜索结果中：

1. 123456789000
2. 188888823400

其实最简单的解决方案是遍历所有字符串，然后用 KMP 算法。但是这样的问题是需要遍历 10W 个元素，效率比较低。我想到的是办法是使用索引。建立 100 个索引（00 到 99），比如输入 `234` 时只需要在索引 `23` 对应的区域查找即可，可以加快 100 倍速度。但是缺点是插入数据时，需要更新多个索引，数据量会是原来的 10 倍。

目前还没有想到好的解决方案。有大神提醒说用字典树，有空研究一番。

最后问了 OC 的数组中，添加 `nil` 对象会有什么问题。当时没答上来，现在想想很不应该，因为数组是以 `nil` 结尾的，如果添加了 `nil`，后续就不能添加对象了。

网易

笔试

主要是计算机方面的大杂烩，涉及操作系统，网络，移动开发，算法等。难度不大，目测是为了淘汰浑水摸鱼的人，就不列出题目了，算法有三题，直接在线写（木有 IDE 表示很忧伤）：

1. 很长一道题，读了很久才读懂，目测是 DFS，但是最后没时间了，写了个思路。
2. 把 "www.zhidao.baidu.com" 这样的字符串改成 "com/baidu/zhidao/www"。——老题目了，剑指 offer 的，两次逆序排列即可。
3. 求数组中和为某个值的所有子数组，比如数组是 `[5,5,10,2,3]` 一共有四个子数组的和是 15，比如 `[5,10]`，`[5,10]`，`[10,2,3]`，`[5,5,2,3]`。这个就是简单的递归了，分两种情况，当前位置的数字在子数组中，以及不在子数组中。

一面

全部是 iOS 题，可能是觉得算法已经面过了：

1. 介绍 block。——我提到栈上的 block 在 ARC 下会自动复制到堆上，面试官问我从 iOS 4 还是 5 开始支持这一特性，表示不知道，我又不是学 OC 历史的，后来想想可能是公司内部老项目有这个坑。
2. ARC 会对代码做什么优化？——比如 `NSString *s2 = s1; s2 = nil` 这样的语句，可能就不会有 `retain` 和 `release` 方法了。
3. 介绍一下 MVVM 和 RAC。——可能是我简历的某个角落写了用过 RAC，被挖出来了，大概谈了一下，结果面试官问我数据的双向绑定怎么做，`bind` 函数了解过么，果断说已经忘了😂😂😂
4. 介绍自己用过哪些开源库。——Masonry 和 SnapKit，AFNetWorking，MKNetworkKit，Alamofire，Mantle，SDWebImage
5. 如果让你写，你能实现么？——当然不能，不然还要实习？
6. 读过某个库的源码么？——扯了一点 SDWebImage，后来被告知这个库用了 runloop 来保证滑动是加载数据的流畅性，自己看了源码后表示没有发现，唯一用到 runloop 地方是保证后台线程一直跑，也有可能是我理解错了，如果错误欢迎指正。
7. SDWebImage 下载了图片后为什么要解码？——当时蒙住了，面试官很 nice 的解释了一下，说是要把 png 文件建立一个什么内存映射，目前还不太懂，有空研究一下。

本来以为面的这么差肯定是挂了，没想到还是过了一面。过了不到一个小时，HR 电话打过来，约了两天后二面。

二面

纯数学和算法：

1. 下面这段代码的输出结果是：

```
int main() {
    int a[5]={1,2,3,4,5};
    int *ptr=(int *)(&a+1);
    printf("%d,%d",*(a+1),*(ptr-1));
}
```

答案是 2 和 5。`a` 是指向数组开头元素的指针，`a + 1` 就是指向下一个元素的指针，所以星号求值以后是 2。`&a` 相当于是数组的指针，`&a + 1` 是数组后面一个数组的指针，然后转换成 `int *` 类型是 5 这个数字后面的一个数字的指针。再减一就是指向 5 的指针，所以星号求值以后是 5。

2. 某个地方天气有如下规律：如果第一天和第二天都不下雨，则第三天下雨的概率为30%；如果第一天和第二天中有任 意一天下雨,则第三天下雨的概率为60%。问如果周一周二都没下雨，那么周四下雨的概率为___。

   简单的概率题，答案是：`30% * 60% + 70% * 30% = 39%`

3. 某痴迷扑克的小团体喜欢用23456789TJQKA来计数，A后面是22,23,...,2A,32,...,AA,222,... 依次类推。请用C/C++或Java写个程序，将用字符串表示这种计数法转换成字符串表示的10进制整数。其中，该计数法的2就对应于十进制的2，之后依次递增。C/C++函数接口： char* pokToDec(char *)

   我的解决思路是进制转换，类似于 16 进制转换 10 进制这种，最后再把数字转成 `char *` 类型。

然后好像没结果了，可能是编程实现太渣了?

其他我知道的面试题

阿里一面：

1. `MVC` 具有什么样的优势，各个模块之间怎么通信，比如点击 Button 后 怎么通知 Model?
2. 两个无限长度链表（也就是可能有环） 判断有没有交点
3. `UITableView` 的相关优化
4. `KVO` 、`Notification` 、`delegate` 各自的优缺点，效率还有使用场景
5. 如何手动通知 `KVO`
6. Objective-C 中的 `copy` 方法
7. runtime 中， `SEL` 和 `IMP` 的区别
8. `autoreleasepool` 的使用场景和原理
9. `RunLoop` 的实现原理和数据结构，什么时候会用到
10. `block` 为什么会有循环引用
11. 使用 `GCD` 如何实现这个需求：A、B、C 三个任务并发，完成后执行任务 D。
12. `NSOperation` 和 `GCD` 的区别
13. `CoreData` 的使用，如何处理多线程问题
14. 如何设计图片缓存?

15. 有没有自己设计过网络控件?

阿里二面:

1. 怎么判断某个 `cell` 是否显示在屏幕上
2. 进程和线程的区别
3. `TCP` 与 `UDP` 区别
4. `TCP` 流量控制
5. 数组和链表的区别
6. `UIView` 生命周期
7. 如果页面 A 跳转到 页面 B，A 的 `viewDidDisappear` 方法和 B 的 `viewDidAppear` 方法哪个先调用?
8. `block` 循环引用问题
9. `ARC` 的本质
10. `RunLoop` 的基本概念，它是怎么休眠的?
11. `Autoreleasepool` 什么时候释放，在什么场景下使用?
12. 如何找到字符串中第一个不重复的字符
13. 哈希表如何处理冲突

面试收获

1. 算法题怎么答

面试官可能会问到你闻所未闻的算法，这时候你不应该自己瞎想，而是先和面试官把问题讨论清楚。要知道，通过沟通弄明白复杂的问题也是一种能力，在和面试官交流的过程中，不仅仅可以搞清楚题目真正的意思是什么，还可以展现自己良好的交流沟通能力。所以千万不要因为紧张或者害羞而浪费这次大好的机会。

有些题目似曾相识，但是暂时没有思路。这时候不妨告诉面试官，给我一些时间思考这个题。然后不要急，不要慌，就当他不存在，拿出纸和笔慢慢算（这充分说明了面试戴耳机的重要性）。你一定要坚定一个信念：“任何一道稍微有难度的算法题，除非做过，否则一定是需要时间想的”。所以，合理的安排思考时间吧。如果十几分钟都想不出来，可以直接放弃。

有时候面试官会要求在线编程，相信我，他不会无聊到盯着你的代码看的，面试官一般都很忙，他也有自己的工作要完成，所以你就当是用自己的 IDE 就好。在线编程往往是一个中等难度的问题，所以不要自己吓唬自己。同时要注意代码格式的规范，适当的注释，提前编写好测试用例等，即使没有解决问题，也至少要把自己良好的编程习惯展示给面试官。

2. 遇到不会的问题怎么处理

这个问题有可能是面试官故意说得含糊不清，考察你的交流能力，也有可能是无意的，或者是你的理解方式出现了偏差。不管是以上哪种问题，你都应该先和面试官交流，直到你搞懂了面试官要问你什么，而不是按照自己的理解说了一堆无用的东西。

举个例子，面试官可能会问了一道算法题：“如何判断两个无限长度的链表是否有交点？”。对于“无限长度”可以有不同的理解，如果真的是有无穷多个节点，那显然这个问题是无法解决的。但如果链表仅仅是有环，那么还是可以解决的。如果面试官的本意是链表有环，但你错误的理解成了无穷多个节点，那么必然会导致无法回答这个问题。而且这并非能力不足，而是属于交流沟通方面的失误，这也正是我想分享的“技巧”。

还有一些问题，虽然你没有接触过，但是由于对类似的问题或者情况有过思考，所以可以合理假设。比如面试官问 “ARC 会对代码做什么样的优化？”。我们知道 ARC 的本质就是在合适的地方插入 `retain` 和 `release` 等方法，那么就应该从这个角度出发去思考问题。

显然分别执行 `retain` 和 `release` 操作是没有必要的，那么就可以构造出相应的例子：

```
NSString *s1 = @"hello";
NSString *s2 = s1;
NSString *s2 = nil;
```

由于这种问题我们没有真正实践过，所以可以委婉的告诉面试官：“根据我的推理，可能会有……”。

3. 遇到真的不会的问题怎么处理

遇到不会的问题果断承认啊。如果是基本问题，比如问你哈希表怎么实现，你说不会，那么这次面试可能就悬了。如果是有一定难度的问题，那么你承认不会，也是一种明智之举，毕竟人无完人，一个问题不会并不能全盘否定一个人的能力。

但是比较糟糕的一种情况是，面试者由于过分紧张，担心答不上面试官的问题会有严重后果，所以尝试着去敷衍面试官。比如：“我猜是 xxx 吧”，“我觉得可能是 ……”，更有甚者直接装逼：“这个我试过，不就是 xxx 么”。要知道，此时的你，由于紧张，在心态上已经输给了面试官，更何况面试官问你的问题一定是他有把握的，你觉得这时候你负隅顽抗会有几成胜算呢?

所以，面试官问我“堆排序”的细节时，由于我当时忘了堆排序是怎么实现的，所以我直接告诉他我记不清了。另一个主动认输的例子是面试官问我 RAC 如何实现双向绑定，我告诉他这个是我当时学习的时候写过的 demo，因为不常用，已经只记得一些简单的概念了。

最后，还需要保持一个平稳的心态：“面试时尽力就好，遇到自己不会的问题也是正常情况”。如果面试者顺利答对了所有问题，难免会让面试官感到一丝尴尬，面试者也有可能会产生一些别的情绪。所以，我们要做的只是把自己的能力展示给面试官，做到不骄不躁。

4. 准备杀手锏

除了能够回答上面试官的问题以外，我建议自己准备一两个杀手锏级别的话题。所谓的杀手锏，至少具备以下几个特征：

- 你亲自动手试验过。所谓实践是检验真理的唯一标准，数据是不会说谎的。
- 问题有足够的深度。一面的面试官可能是你的直接上司，二面一般就是更改级别的。你的深度一定要**远超**其他面试者，让一面面试官觉得自己没有十足把握，让二面面试官觉得这是一个好话题，自己的手下

都不一定能有这么独到深刻的见解。
- 你对这个问题理解的足够深入，无论是广度还是深度都达到一定水平。

以 iOS 中的 `UITableView` 的调优为例，我自认为对它有一定的理解，同为 iOS 开发者的读者可以阅读这篇文章：UIKit性能调优实战讲解，同时我还仔细研究了 sunnyxx 大神的 优化UITableViewCell高度计算的那些事。

这一类的话题通常需要仔细研究官方文档，iOS 开发者还可以观看 WWDC 视频，然后花上充足的时间去总结。比如我写 iOS自定义转场动画实战讲解 这篇文章就花了至少三天时间，包括大年初一一整天。

由于此类话题数量不多，所以准备一个或数个即可，面试时可以有意识的将面试官引导到这些话题上去，从而充分的展示自己。

5. 心态

通常情况下，面试结果都会在 1 - 3 天内知道。有的面试官会当场告诉你通过了，有的公司面试结束后几个小时就能出结果。

但有些时候，由于某些原因（我也不清楚。。。。可能是比较忙？），你迟迟无法获知面试结果。这时候你可以选择耐心等待，获知直接给 HR or 内推者发送邮件。一般来说面试结束后三天还没收到通知，你可以发送邮件询问或者再等等。

复习资料

对于读到这一段的读者，为了感谢你耐心的听我废话了这么久，送上一波精心整理的干货和资料。不敢说完全没有错，但是应该比自己去查要靠谱得多。主要涉及算法、网络、操作系统、Objective-C 和 iOS 五个方面。如果你不是 iOS 开发者，相信前三部分的资料也或多或少能够帮上你。

算法

这一部分的内容主要分为以下几个部分：字符串、数组与查找、链表、树以及其他基础问题。

总的来说，算法问题可以分为以下三类：

1. 基础问题：即使是新手，一眼看过去就有思路，只是实现的时候需要注意细节。
2. 普通问题：这些问题通常属于以上分类中的某一类，需要面试者掌握一些常见的思路，比如递归、动态规划、BFS/DFS、双指针、二分搜索 等。或者是直接考察数据结构的使用，如：哈希、栈和队列、链表等，如果具备了这些基础知识，此类题目通常能够比较快速的解决。
3. 进阶问题：这些题的解题思路和普通问题相似，但是需要你事先有对应的知识积累，否则难以直接看出问题的本质。
4. 疑难杂题：这类问题比较奇怪，解决它以后并不能给别的题目太多帮助，如果时间紧张可以暂时放弃。

一般来说，一类问题难度不大，面试前简单复习一下，面试时小心仔细，全面思考即可。二三类问题是面试重点，需要提前准备。第四类问题通常出现较少，即使不会做，对最终评价的负面影响也不会有前三类那么

大。

如果时间充裕，我建议阅读**《剑指 Offer》**这本书并配合 Leetcode 来巩固知识，在我的面试过程中，出现很多书上的原题或者变体，我自认为没有因为算法而影响任何一次面试的成绩。如果时间紧张，你也可以只完成我列出的一些经典题目，在“【】”中标记了我对此题类型的分类，**如果加星号表示此题在实际面试中出现过。**

> PS: 最近有小伙伴被问到了哈希表的实现。这可以理解为算法，也可以归类为计算机基础知识。总的来说你至少需要明白哈希值的特点和两种解决冲突的方式：拉链式和开放寻址。

字符串

1. 【3】最长回文子串
2. 【3】最长无重复子串
3. 【1*】字符串转数字
4. 【4】KMP 算法
5. 【2】字符串全排列
6. 【2*】翻转字符串

动态规划

1. 【2】背包问题
2. 【3】连续子数组的最大和
3. 【4】实现简单的正则表达式匹配

数组

1. 【3】求两个等长、有序数组的中位数（二分法）
2. 【4】求两个不等长、有序数组的中位数
3. 【3】旋转数组求最小值、【3】旋转数组求查找某个值是否存在（二分法）
4. 【4*】每行从左到右，每列从上到下递增的二维数组中，判断某个数是否存在（剑指 offer 第 3 题）
5. 【3*】数组中出现次数超过一半的数字
6. 【3*】第 k 大的数（拓展：最大的 k 个数）
7. 【3*】有序数组中某个数字出现的次数（提示：利用二分搜索）

链表

1. 【2】反转链表（使用递归和迭代两种解法，了解头插法）
2. 【3】删除链表的当前节点
3. 【3】删除倒数第 k 个节点
4. 【1】两个有序链表合并
5. 【4】复杂链表的复制
6. 【2*】判断链表是否有环
7. 【3*】两个链表的第一个公共节点（提示：考虑链表有环的情况）
8. 【3】删除链表中重复节点

树

1. 【3】根据中序和后序遍历结果重建二叉树、【3】根据中序和前序遍历结果重建二叉树
2. 【2】翻转二叉树
3. 【2】从上往下打印二叉树 (BFS 的思想)
4. 【3】判断某个数组是不是二叉树的后序遍历结果 (剑指 offer 第 24 题)
5. 【3】二叉树中和为某个值的路径
6. 【3*】二叉树中某个节点的下一个节点 (强烈推荐准备一下，剑指 offer 第 58 题)

栈

1. 【2】用两个栈实现队列、【2】用两个队列实现栈
2. 【2】实现一个栈，可以用常数级时间找出栈中的最小值
3. 【3】判断栈的压栈、弹栈序列是否合法（剑指offer 第 22 题）

排序

了解以下排序的时间、空间复杂度，是否稳定，实现原理

1. 归并排序、拓展：求数组中的逆序对个数
2. 快速排序 重点：`partion` 函数的实现
3. 堆排序
4. 数组元素值域已知时，考虑 基数排序 和 桶排序

位运算

1. 【2】给一个十进制数字，求它的二进制表示中，有多少个 1 (n &= n - 1)
2. 【3】给一个数组，所有数字都出现了偶数次，只有一个出现了一次，找出这个数
3. 【4】给一个数组，所有数字都出现了三次，只有一个出现了一次，找出这个数
4. 【3】给一个数组，所有数组都出现了偶数次，只有两个数字出现了一次，找出这两个数

网络层

根据不同的面试岗位，侧重点略有不同。对 iOS 和 Android 开发者来说，HTTP 考的略少，以 TCP 和 UDP 为主。其实 UDP 基本上只会考察和 TCP 的区别。

当然还有一些常见的基础问题，比如 Cookie 和 Session 的考察，POST 和 GET 的考察，HTTPS 的简单了解等。这些问题在我的博客中都有简单的总结。

总结了一些资料，数字序号越大的资料表示篇幅更长，耗时更久，难度更大，讲解更细致。破折线后表示预计需要多久可以读完。

1. 【博客】我的六篇总结————不到一周
2. 【书】图解 TCP/IP————半个月
3. 【书】TCP/IP 详解————没读过，感觉至少需要一个月

4. 【书】TCP/IP 协议簇－－－－没读过，感觉至少需要一个月

光读书是没有用的，一问到实际问题很容易懵逼，以下是我总结的一些问题：

1. 简介 TCP 和 UDP 区别，他们位于哪一层?
2. 路由器和交换机的工作原理大概是什么，他们分别用到什么协议，位于哪一层?
3. 描述TCP 协议三次握手，四次释放的过程。
4. TCP 协议是如何进行流量控制，拥塞控制的?
5. 为什么建立连接时是三次握手，两次行不行？如果第三次握手失败了怎么处理
6. 关闭连接时，第四次握手失败怎么处理?
7. 你怎么理解分层和协议?
8. HTTP 请求中的 GET 和 POST 的区别，Session 和 Cookie 的区别。
9. 谈谈你对 HTTP 1.1，2.0 和 HTTPS 的理解。

操作系统与编译

我被问到的操作系统问题很少，所以仅仅总结了一些自认为比较重要的问题。关于这一部分的知识，推荐阅读**《程序员的自我修养》**，如果时间有限，你可以阅读我的**《程序员的自我修养读书笔记》**，并思考这些问题：

1. 源代码是怎么变成可执行文件的，每一步的作用是什么？（预编译，词法分析，语法分析，语义分析，中间语言生成目标代码生成，汇编，链接）
2. 应用层、API、运行库、系统调用、操作系统内核之间的关系是什么?
3. 虚拟内存空间是什么，为什么要有虚拟内存空间。
4. 静态链接和动态链接分别表示什么，大概是怎么实现的?
5. 可执行文件的结构如何？（分为哪些段）
6. 它是怎么装载进内存的，为什么要分段，分页，页错误是什么?
7. 进程的内存格局是怎样的？（堆、栈、全局/静态区，代码区，常量区）
8. 堆和栈的区别，函数调用和栈的关系
9. 进程和线程的区别
10. 异步和同步，串行，并发，并行的区别
11. 多并发任务，仅多线程能加快速度么（不能，会变慢，有线程切换的开销）
12. 多个线程之间可以共享那些数据
13. 进程之间如何通信
14. 介绍几种锁，他们的用途和区别

关于多线程相关的，推荐阅读这篇文章的前面一小部分－－**《iOS多线程编程－－GCD与NSOperation总结》**

关于操作系统和编译方面的文章，除了读原书和我的读书笔记外，还可以参考这篇文章－－**《修改一个数字破解Mac上的应用》**

OC

首先两本必备的神书一定是要读完的。一本是讲 OC 的《Effective Objective-C 2.0》，中文名叫：“编写高质量 iOS 与 OS X 代码的 52 个有效方法”。另一本书叫：《Objective-C 高级编程》。前者讲解 OC 中各种细节，后者主要讲了 ARC、Block 和 GCD。

光是读书，思考不够，很容易在面试时被问懵逼，所以建议一遍尝试回答面试真题，一边阅读以下总结性的文章，重要性不分先后：

1. 检测内存泄露
2. KVO与KVC原理、KVO、Notification、Delegate优缺点、最推荐的官方文档
3. GCD 与 NSOperation
4. Runtime
5. block
6. atomic 线程安全、@synchronized
7. 对象的深浅复制
8. 招聘一个靠谱的iOS
9. 消息传递机制
10. 深入理解Objective-C：Category

强烈推荐第八篇文章，做完这上面的题目基本上可以应付大多数 OC 方面的问题了。

iOS 开发

1. RunLoop
2. Cell 图片异步加载优化
3. iOS 函数式编程的实现 && 响应式编程概念
4. 内存恶鬼drawRect
5. UIKit 性能调优(主要是UITableView)
6. 优化UITableViewCell高度计算的那些事
7. 高性能图片架构与设计
8. 轻量化视图控制器
9. UIView的生命周期
10. 高效设置圆角
11. 事件的传递和响应机制
12. ReactiveCocoa 和 MVVM 入门

其中需要重点了解 `runloop`，它不仅仅是简单的“跑圈”的 概念，很多问题其实都与它有关，建议认真阅读 ibireme 大神的总结

其他面经

1. 我是如何同时拿到阿里和腾讯offer的
2. 大三学生拿到阿里,百度实习offer面试经验分享
3. 2016年1月TX电面题