先导篇

《91 天学算法》的主要目的是**帮助算法初学者入门**，以解决诸如面试中的算法问题和部分设计问题。

> 一很多设计问题都可以抽象为纯粹的算法问题。

相比于传统的算法入门书籍和训练营，我们的特色是：

- 内容完整，几乎覆盖了所有算法面试的考点。
- 语言通俗易懂，讲解丰富，对初学者友好。部分小专题篇幅达到了 2 万多字，同时有大量的图画和题目，力求大家真正理解。
- 很多题目都提供了多种语言（Python，Java，CPP，JS 等），方便不同语言的人进行学习。
- 侧重刷题。通过结合力扣题目，真正做到大家学习认真完成之后可以解决大部分的力扣题目，做到在面试中迎刃有余。

91 算法共分为三篇， 基础篇 ， 进阶篇 和 专题篇 。

让你：

- 显著提高你的刷题效率，让你少走弯路
- 掌握常见面试题的思路和解法
- 掌握常见套路，了解常见算法的本质，横向对比各种题目
- 纵向剖析一道题，多种方法不同角度解决同一题目

第一阶段基础篇(30 天)。预计五个子栏目，每个子栏目 6 天。到时候发讲义给大家，题目的话天一道。 **讲义的内容大概是我在下方讲义部分放出的链接那样哦。**

91 天学算法主要讲什么？

- 基础数据结构
- 常见的基础算法，几乎可以覆盖到面试的 80% 的考点。如果大家把这 80% 学精了，那么其他的 20 % 也是手到擒来。后续会陆续更新更多内容
- 面试题。每天更新一道题，这些题都是经过讲师精挑细选的，质量很高。并且题目之间梯度把控合理，难度循序渐进。

什么人适合这门课?

- 准备算法面试的人
- 想要提高程序员内功的人
- 。。。

那什么样的人不需要学呢?

- 得过 ACM 区域赛冠亚军
- 力扣竞赛排名前 500
- 平时的 OJ 题目困难都可以做出来（可能不是特别顺）
- 。。。

规则

大家的问题，打卡题目，讲义都在这里更新哦，冲鸭 。91 天见证更好的自己！不过要注意一周不打卡会被强制清退。

需要提前准备些什么?

- 数据结构与算法的基础知识。 推荐看一下大学里面的教材讲义，或者看一些入门的图书，视频等，比如《图解算法》，邓俊辉的《数据结构与算法》免费视频课程。总之， 至少 你要知道有哪些常见的数据结构与算法以及他们各自的特点。
- 有 Github 账号，且会使用 Github 常用操作。 比如提 issue，留言等。
- 有 LeetCode 账号，且会用其提交代码。

> 语言不限，大家可以用自己喜欢的任何语言。同时我也希望你不要纠结于语言本身。

这里建议大家结合我的新书《算法通关之路》来进行学习，效果会更好。图书介绍以及购买方式：https://leetcode-solution.cn/book-intro

图书大纲：

具体形式是什么样的?

- 总共三个大的阶段（基础篇，专题篇，进阶篇）
- 每个大阶段划分为几个小阶段
- 每个小阶段的时间内，每天都会出关于这个阶段的题目，并提供官方题解

比如：

- 第一个大阶段是基础
- 基础中第一个小阶段是 数组，栈和队列 。
- 数组，栈和队列 正式开始前，会开放对应讲义，大家可以提前预习。
- 之后的每天都会围绕 数组，栈和队列 出一道题，并给出解答。大家可以在出题当天在**打卡标签**打卡。

> 大家遇到问题可以在群里回答，对于比较好的问题，会记录起来供大家查看。

如何打卡?

- 每天都会有一道题目，大家可以在打卡标签中看到。地址：https://leetcode-solution.cn/91?tab=sign
- 大家在下面留言即可

> 想要坚持打卡抽奖的小伙伴注意了，必须当天打卡才算打卡哦。 不是当天的话需要补签卡，补签卡需要连续打卡一周才可以获得一张的

打卡格式：

- 思路
- 代码
- 复杂度分析（时间和空间）

一个例子：

```
### 思路

（此处撰写思路）

### 代码


```java（此处换成你的语言，比如js，py 等）
（此处撰写代码）


```

复杂度分析
- 时间复杂度：O(N)，其中 N 为数组长度。
- 空间复杂度：O(1)
```

环境准备

由于打卡都在 LeetCode 和 BinarySearch 这样的 OJ 平台，因此大家不必准备本地环境。

同时为了大家更好的学习，分享，debug 代码。我推荐两个在线的 repl 工具给大家。

- https://replit.com
- https://glot.io

建议大家使用第一个，功能更多，体验更好。不过如果你不喜欢花里胡哨的功能，追求简洁或者你不想注册（replit 需要注册登录）的可以使用第二个。

平时大家做题卡住，也可以使用 replit 的调试功能，甚至可以直接将代码分享给其他人（下面《如何提问》部分会提到）。

使用在线工具还有一个好处，就是更贴近面试。有时候面试是白板，有时候面试是一个在线编辑器。不管是什么，使用这种工具都比使用力扣更贴近，因为至少需要你自己准备测试用例。

如何提问

大家不懂的都可以在群里提问。不过为了更优效率，建议大家做到以下几点：

- 是否有其他人问过同样的问题，且得到了解决（可以尝试搜索聊天记录）
- 提供题目的信息，包括不限于题目描述，题目地址
- 描述你的思考过程，你的期望是什么。
- 最好附上 repl 链接，大家点开链接就是你的代码，环境都准备好了。大家不需要自己准备环境。

比如：

```
今天的打卡题 xxxx 部分理解。

链接：xxxxx
题目描述 xxxxx


我的想法是：xxxx
我期望的答案是：xxxx（或者我的代码是xxxx）
这里是我的 repl link：xxxxx
请教大佬帮忙解决
```

面试技巧

我在网上找到一份**《Interview Cheat Sheet》**，这个 PDF 列举了面试的**模板步骤**。，详细指示了如何一步步完成面试。

这个 pdf 开头就提到了好的代码三个标准：

1. 可读性

2. 时间复杂度

3. 空间复杂度

这写的太好了。

紧接着，列举了 15 算法面试的步骤。比如步骤一：**当面试官提问完后，你需要先下来关键点（之后再下面写注释和代码）** 看完我的感受就是，**面试只要按照这个来做，成功率蹭蹭提升**

这里正好有一个视频，是面试 Google 的视频，基本上就是按照这 15 条来的， 你可以拿这个当实际的例子，按照这个来面试就行了。 当视频里的面试者提问**有没有小数，有没有负数，同一数字是否可以用两次的时候，我就知道了我和他的差距！** 。顺便提一句，这个视频的算法题是两数和，大家不需要太担心题目太难。这样大家可以集中精力学习**别人是怎么面试的**

另外我推荐大家参考一下我们的明星学员**易潇**的经验总结。

她的上岸经历可以给大家一些参考。下面是关于她的一些 link

- 《技术面试原来不止考技术？》
- 个人采访
- Github 主页

上一页 下一页