## 一、夯实基础

在最初的阶段，算法世界的大门刚刚打开，这个时候迷茫是正常的，解决迷茫的要诀在于少想多做，勇往直前。怀着一颗"千磨万击还坚韧，任尔东西南北风"的恒心，爬上算法的高楼，做到"望尽天涯路"。

从一个算法萌新入门，第一步便在于打牢根基。推荐阅读书籍：

1.**《算法第 4 版》**- Robert Sedgewick

适合初学者入门

2.**《大话数据结构》**- 程杰  
3.**《算法图解》**- Aditya Bhargava

《大话数据结构》和《算法图解》这两本书的特点是有趣、易理解，也非常适合初学者。

4.《**数据结构和算法分析- C 语言描述**》- Mark Allen Weiss

需要有一定 C 语言基础

**进阶**：

1.《**编程珠玑**》- Jon Bentley

本书探讨了程序设计人员面对一系列的实际问题以及解决问题的措施（解决方案的代码以 C/C++ 语言编写）。书中选取了许多具有典型意义的复杂编程和算法问题，并阐述和总结了许多独特精妙的设计原则、思考和解决问题的方法以及实用的程序设计技巧。

2.《**算法导论**》- Cormen,T.H.

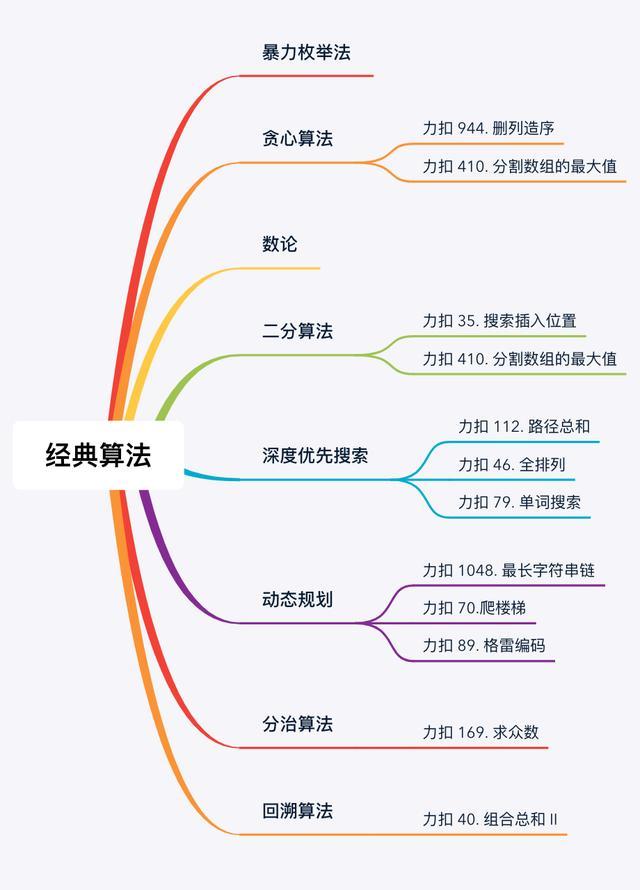
《算法导论》的特点是全面，它是一本算法的百科全书，着重在于开阔算法视野，适合有一定算法基础后再去学习。

入门阶段是看一些天赋的，花费时间因人而异，大约在 3～6 月之间，将上述提到的书籍选择其中一本看完基本就能入门了。在这个阶段中，需要了解几类常用的算法：

1. 常用的数据结构：**数组、字符串、链表、树（如二叉树）等**

2. 常用的算法：**分治、贪心、穷举、动态规划、回溯、二分算法、深度优先搜索等**

可搭配力扣的题目进行练习



其中，暴力枚举、贪心算法容易理解，可以很快上手。数论相关的算法需要用到一些数学技巧，包括位运算、幂函数、求模等等性质。二分算法和深度优先搜索算法相对有些技巧性，好在他们都有固定的模板。另外，不得不提的是，深度优先搜索算法的思想非常重要，而且深度优先搜索是动态规划、分治和回溯的基础，需要重点掌握。  
在此过程中，可以辅以力扣（LeetCode）中的简单题目，它们往往都代表了一类经典算法，如：

[70. 爬楼梯](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/climbing-stairs/)

假设你正在爬楼梯。需要 *n* 阶你才能到达楼顶。  
每次你可以爬 1 或 2 个台阶。你有多少种不同的方法可以爬到楼顶呢？

**动态规划** 算法的经典题目，通过此题目可以了解状态、边界条件、状态转移方程等基本概念。

[112. 路径总和](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/path-sum/)

给定一个二叉树和一个目标和，判断该树中是否存在根节点到叶子节点的路径，这条路径上所有节点值相加等于目标和。

**深度优先算法** 的入门题目，递归实现和迭代实现都不难，可以学习到深度优先算法的层层嵌套搜索、找到答案或到达边界停止的基本解题思路。

[35. 搜索插入位置](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/search-insert-position/)

给定一个排序数组和一个目标值，在数组中找到目标值，并返回其索引。如果目标值不存在于数组中，返回它将会被按顺序插入的位置。

**二分算法** 的典型题目，使用二分算法的解题模板可以轻松解决，二分算法的算法思想清晰明确，一通百通。

[169. 求众数](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/majority-element/)

给定一个大小为 *n* 的数组，找到其中的众数。众数是指在数组中出现次数**大于** ⌊ n/2 ⌋ 的元素。  
你可以假设数组是非空的，并且给定的数组总是存在众数。

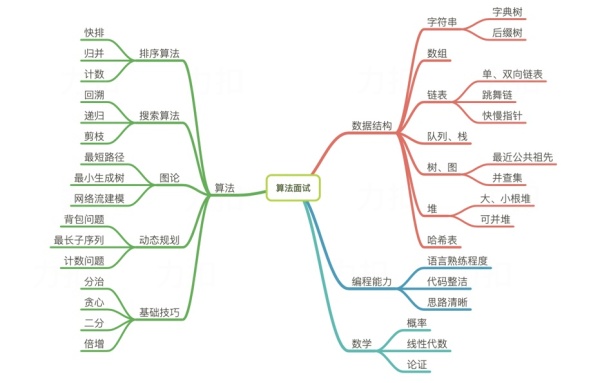
**分治算法** 的简单题目，如果我们知道数组左边一半和右边一半的众数，我们就可以用线性时间知道全局的众数是哪个。这道题妙就妙在可以有多种解题方式，让初学者至少可以写出暴力枚举算法 AC 题目，然后再逐步深入，优化算法。

[944. 删列造序](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/delete-columns-to-make-sorted/)

给定由 N 个小写字母字符串组成的数组 A，其中每个字符串长度相等。  
选取一个删除索引序列，对于 A 中的每个字符串，删除对应每个索引处的字符。 所余下的字符串行从上往下读形成列。  
假设，我们选择了一组删除索引 D，那么在执行删除操作之后，A 中所剩余的每一列都必须是 **非降序** 排列的，然后请你返回 D.length 的最小可能值。

这是一道 **贪心算法** 的简单题目，贪心算法理解简单，上手容易，适合作为初学者掌握的第一个算法。

时间充裕的同学可以按照下图进行系统性地学习：



## 二、融会贯通

学习算法理论如同阅读了一本武功秘籍，然而仅仅掌握理论是不够的，接下来就要进入到实际练习阶段。

实战练习非常重要，不经过实战练习，理论仅仅是纸上谈兵。比如，不经过大量练习，永远不会知道二分算法是多么容易出现死循环。一个边界条件控制不好，程序就会显示无情的"Time Limit Exceeded"。在 20 分钟的调试后，或许仅仅是将 while (left <= right) 改为了 while (left < right) 。程序员说到底也是手艺人，这一个字符的改动，正是"台上一分钟，台下十年功"的体现，需要在大量的练习中才能理解两者之间的不同作用。

再比如，动态规划算法中，递归的函数就像是《盗梦空间》中的"梦中梦"，一层套一层，又渐次展开，很难整体把控。在不断的练习后，才会知道，动态规划算法的重点是抓住动态转移方程，只处理两个状态之间的过渡和边界条件，慢慢"大事化小，小事化了"。

这一阶段花费的时间将会很长很长，伴随着不断地摔倒、爬起，你会对每类算法逐渐融会贯通。好在这一阶段是不看天赋只看勤奋的，每次从坑里爬起，都是献给成长的一份力量。  
推荐的进阶书籍有《编程珠玑》，本书探讨了程序设计人员面对一系列的实际问题以及解决问题的措施（解决方案的代码以 C/C++ 语言编写）。书中选取了许多具有典型意义的复杂编程和算法问题，并阐述和总结了许多独特精妙的设计原则、思考和解决问题的方法以及实用的程序设计技巧。

在这个阶段，可以尝试练习力扣上的中等题目，中等题目基本上也只会使用一种算法，加上一些特殊的限制，好比让你在学习了直拳的理论后衍生出左勾拳和右勾拳。推荐练习题目有：

[1048. 最长字符串链](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/longest-string-chain/)

给出一个单词列表，其中每个单词都由小写英文字母组成。  
如果我们可以在 word1 的任何地方添加一个字母使其变成 word2，那么我们认为 word1 是 word2 的前身。例如，"abc" 是 "abac" 的前身。  
词链是单词 [word\_1, word\_2, ..., word\_k] 组成的序列，k >= 1，其中 word\_1 是 word\_2 的前身，word\_2 是 word\_3 的前身，依此类推。  
从给定单词列表 words 中选择单词组成词链，返回词链的最长可能长度。

分析题目可知，要求出答案必须遍历所有可能的词链，动态规划算法在其中起备忘录的作用，用于记录已经算过的答案，减少计算次数。

[47. 全排列 II](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/permutations-ii/)

给定一个可包含重复数字的序列，返回所有不重复的全排列。

这道题是 [46. 全排列](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/permutations/) 的加强版，全排列 I 的题目是：给定一个 **没有重复** 数字的序列，返回其所有可能的全排列。使用深度优先搜索算法即可解决。本题在其基础上加强了难度，有两种方法可解。第一种方法最简单，直接用全排列 I 的答案去重即可，第二种方法是先将数组排序，全排列时遇到重复数字则跳过，这样的剪枝优化可以减少遍历次数，提高算法效率。

[40. 组合总和 II](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/combination-sum-ii/)

给定一个数组 candidates 和一个目标数 target ，找出 candidates 中所有可以使数字和为 target 的组合。  
candidates 中的每个数字在每个组合中只能使用一次。

深度优先搜索算法衍生出来的回溯算法，同样用到 47 题的剪枝优化思想：相同数字只允许递归第一个。

[89. 格雷编码](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/gray-code/)

格雷编码是一个二进制数字系统，在该系统中，两个连续的数值仅有一个位数的差异。  
给定一个代表编码总位数的非负整数 n，打印其格雷编码序列。格雷编码序列必须以 0 开头。

**动态规划** 算法的实际应用之一。

[79. 单词搜索](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/word-search/)

给定一个二维网格和一个单词，找出该单词是否存在于网格中。  
单词必须按照字母顺序，通过相邻的单元格内的字母构成，其中“相邻”单元格是那些水平相邻或垂直相邻的单元格。同一个单元格内的字母不允许被重复使用。

深度优先搜索的中级应用，使用单独数组标记已使用过的元素，这也是 DFS 中较为常见的做法，难点在于将标记数组复原的时机，需要反复练习，熟练掌握。

当你把每一类算法的中等题目刷起来得心应手时，不妨开始尝试困难题目的练习。困难题目总是融合两种或两种以上算法，或是加深难度的经典算法，如二维甚至三维动态规划。练习困难题目好比同时用上左勾拳和扫堂腿，不仅让思维酣畅淋漓，在每次 AC 之后还会带来无与伦比的成就感。推荐练习题目有：

[679. 24 点游戏](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/24-game/)

你有 4 张写有 1 到 9 数字的牌。你需要判断是否能通过 \*，/，+，-，(，) 的运算得到 24。

只有 4 张牌，且只能执行 4 种操作。即使所有运算符都不进行交换，最多也只有 12 \* 6 \* 2 \* 4 \* 4 \* 4 = 9216 种可能性，这使得我们可以尝试所有这些可能，如果用深度优先搜索算法则需要费一番功夫。

[124. 二叉树中的最大路径和](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/binary-tree-maximum-path-sum/)

给定一个**非空**二叉树，返回其最大路径和。  
本题中，路径被定义为一条从树中任意节点出发，达到任意节点的序列。该路径**至少包含一个**节点，且不一定经过根节点。

首先，考虑实现一个简化的函数：计算每个节点及其子树对路径和的最大贡献。再考虑第二点：最大路径不一定包括根节点。这意味着我们在每一步都检查哪种选择更好：是继续当前路径或者以当前节点作为最高节点计算新的路径。

[410. 分割数组的最大值](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/split-array-largest-sum/)

给定一个非负整数数组和一个整数 *m*，你需要将这个数组分成 *m* 个非空的连续子数组。设计一个算法使得这 *m* 个子数组各自和的最大值最小。

二分算法和贪心算法的综合练习，仔细分析可知其单调关系：数组和的最大值越小，分组数越大。并且数组和的范围是可以确定的。根据此特性，可以将题目转换为：当子数组的和最大为 maxSum 时，至少需要分多少组，能否在最多 m 组的限制范围内完成分割。在每次分割时，采用贪心策略，尽可能多的放置元素，直到一组放不下，再另起一组。如果满足分割条件，记录当前值，利用二分法，缩小子数组总和。否则扩大子数组总和，直到找到最佳答案。

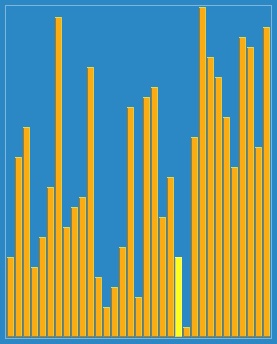
## 三、推陈出新

事实上，大量程序员停留在第二重境界就无法再进一步。当提到某一类算法时，你可以说："我知道"、"我会用"、"踩过坑"，但能说出"我完全理解其思想"、甚至"我能想办法改进"的人却很少很少。这一步仿佛武学中的攻守之道，当你掌握到这一层，便可不再拘泥于一刀一剑、一招一式，如金书中所说：飞花摘叶皆可伤人、草木竹石均可为剑。

开创算法的过程是艰难又孤独的。每一个经典算法的诞生都伴随着"一将功成万骨枯"。比如现在我们在很多语言中都可以直接调用Collection.sort()实现快速排序，而在快速排序算法出现之前，曾有一段时间仅有冒泡、选择、插入三种排序算法。直到1959年，希尔提出"希尔排序"算法，或许现在知道此算法的人已经很少了。但它是首个突破了复杂度的排序算法，它的基本算法思想如下：

选择一个增量序列t1，t2，…，tk，其中 ti > tj， tk = 1； 按增量序列个数k，对序列进行k 趟排序； 每趟排序，根据对应的增量ti，将待排序列分割成若干长度为 m 的子序列，分别对各子表进行直接插入排序。仅增量因子为1 时，整个序列作为一个表来处理，表长度即为整个序列的长度。

动图演示如下：



希尔排序算法较为晦涩难懂，而且并不是最优的排序算法，现在已经被后来的快速排序算法给淘汰了。然而不可否认希尔对排序算法的演进具有开创性贡献，在攀越算法高峰的路上，每一步都走得战战兢兢，我们只有铭记这些伟大的引路人，以此激励自己不断前行。

算法世界不尽完美。不仅有经典算法在前奠基，后起之秀遗传算法、深度学习算法也熠熠生辉。算法世界还有许多"所罗门王的宝藏"，一直静静地守候在"灯火阑珊处"，等待着人们去发掘。

## 四、学习方法

我给大家整理了一个学习计划，可以保存下图进行学习：



现在网上有很多资源、博客、论坛可供我们更方便地学习知识片段。然而这种类似兵来将挡、水来土掩般的学习方法虽然有用，却并不特别的好。这里推荐大家在网上寻找一些系统的学习教程，以帮助自己由浅入深，一路成长。

## 五、真题演练

力扣将 Top Interview Questions 里比较新的题目按照类别进行了整理，以供大家按模块练习。

### 模拟

* [134. 加油站](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/gas-station/)
* [146. LRU缓存机制](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/lru-cache/)
* [202. 快乐数](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/happy-number/)
* [289. 生命游戏](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/game-of-life/)
* [371. 两整数之和](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/sum-of-two-integers/)
* [412. Fizz Buzz](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/fizz-buzz/)

### 数组

* [152. 乘积最大子序列](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/maximum-product-subarray/)
* [169. 求众数](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/majority-element/)
* [189. 旋转数组](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/rotate-array/)
* [217. 存在重复元素](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/contains-duplicate/)
* [283. 移动零](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/move-zeroes/)
* [384. 打乱数组](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/shuffle-an-array/)
* [350. 两个数组的交集 II](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/intersection-of-two-arrays-ii/)
* [334. 递增的三元子序列](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/increasing-triplet-subsequence/)
* [240. 搜索二维矩阵 II](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/search-a-2d-matrix-ii/)
* [238. 除自身以外数组的乘积](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/product-of-array-except-self/)

### 链表

* [138. 复制带随机指针的链表](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/copy-list-with-random-pointer/)
* [141. 环形链表](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/linked-list-cycle/)
* [148. 排序链表](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/sort-list/)
* [160. 相交链表](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/intersection-of-two-linked-lists/)
* [206. 反转链表](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/reverse-linked-list/)
* [234. 回文链表](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/palindrome-linked-list/)
* [237. 删除链表中的节点](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/delete-node-in-a-linked-list/)
* [328. 奇偶链表](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/odd-even-linked-list/)

### 堆

* [155. 最小栈](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/min-stack/)
* [215. 数组中的第K个最大元素](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/kth-largest-element-in-an-array/)
* [295. 数据流的中位数](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/find-median-from-data-stream/)
* [378. 有序矩阵中第K小的元素](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/kth-smallest-element-in-a-sorted-matrix/)
* [347. 前K个高频元素](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/top-k-frequent-elements/)

### 栈

* [150. 逆波兰表达式求值](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/evaluate-reverse-polish-notation/)
* [227. 基本计算器 II](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/basic-calculator-ii/)
* [341. 扁平化嵌套列表迭代器](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/flatten-nested-list-iterator/)

### 哈希 / Map

* [171. Excel表列序号](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/excel-sheet-column-number/)
* [454. 四数相加 II](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/4sum-ii/)
* [380. 常数时间插入、删除和获取随机元素](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/insert-delete-getrandom-o1/)

### 队列

* [239. 滑动窗口最大值](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/sliding-window-maximum/)

### 树

* [230. 二叉搜索树中第K小的元素](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/kth-smallest-element-in-a-bst/)
* [236. 二叉树的最近公共祖先](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/lowest-common-ancestor-of-a-binary-tree/)
* [297. 二叉树的序列化与反序列化](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/serialize-and-deserialize-binary-tree/)

### 线段树

* [218. 天际线问题](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/the-skyline-problem/)

### 排序

* [179. 最大数](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/largest-number/)
* [324. 摆动排序 II](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/wiggle-sort-ii/)

### 二分检索

* [162. 寻找峰值](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/find-peak-element/)
* [287. 寻找重复数](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/find-the-duplicate-number/)
* [315. 计算右侧小于当前元素的个数](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/count-of-smaller-numbers-after-self/)

### 滑动窗口

* [395. 至少有K个重复字符的最长子串](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/longest-substring-with-at-least-k-repeating-characters/)

### 动态规划

* [124. 二叉树中的最大路径和](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/binary-tree-maximum-path-sum/)
* [128. 最长连续序列](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/longest-consecutive-sequence/)
* [198. 打家劫舍](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/house-robber/)
* [279. 完全平方数](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/perfect-squares/)
* [300. 最长上升子序列](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/longest-increasing-subsequence/)
* [322. 零钱兑换](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/coin-change/)
* [329. 矩阵中的最长递增路径](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/longest-increasing-path-in-a-matrix/)

### 图论

* [127. 单词接龙](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/word-ladder/)
* [200. 岛屿的个数](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/number-of-islands/)
* [207. 课程表](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/course-schedule/)
* [210. 课程表 II](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/course-schedule-ii/)

### 数学 & 位运算

* [136. 只出现一次的数字](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/single-number/)
* [149. 直线上最多的点数](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/max-points-on-a-line/)
* [166. 分数到小数](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/fraction-to-recurring-decimal/)
* [172. 阶乘后的零](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/factorial-trailing-zeroes/)
* [190. 颠倒二进制位](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/reverse-bits/)
* [191. 位1的个数](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/number-of-1-bits/)
* [204. 计数质数](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/count-primes/)
* [268. 缺失数字](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/missing-number/)
* [326. 3的幂](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/power-of-three/)

### 字符串

* [125. 验证回文串](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/valid-palindrome/)
* [131. 分割回文串](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/palindrome-partitioning/)
* [139. 单词拆分](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/word-break/)
* [140. 单词拆分 II](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/word-break-ii/)
* [208. 实现 Trie (前缀树)](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/implement-trie-prefix-tree/)
* [212. 单词搜索 II](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/word-search-ii/)
* [242. 有效的字母异位词](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/valid-anagram/)
* [387. 字符串中的第一个唯一字符](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/first-unique-character-in-a-string/)
* [344. 反转字符串](https://link.zhihu.com/?target=https%3A//leetcode-cn.com/problems/reverse-string/)