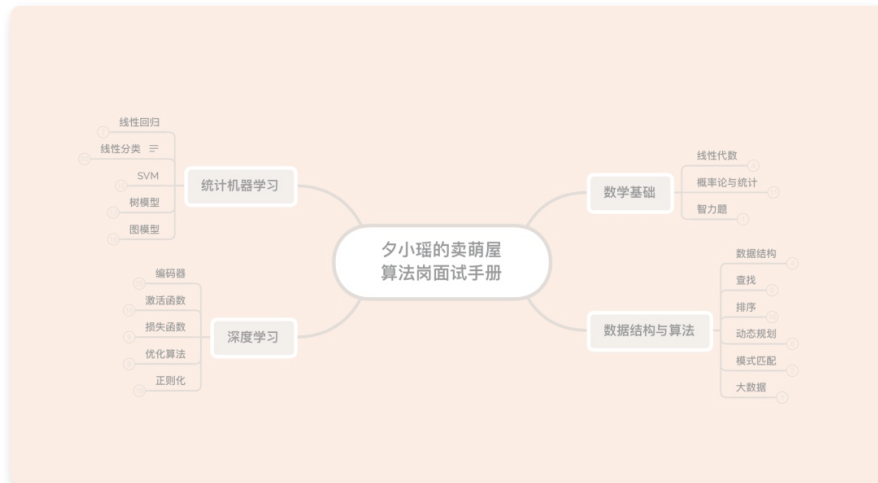


算法工程师思维导图—数据结构与算法

原创 rumor酱 夕小瑶的卖萌屋 1周前

来自专辑

卖萌屋@算法工程师思维导图



卖萌屋的妹子们（划掉）作者团整理的**算法工程师思维导图**，求职/自我提升/查漏补缺神器。该手册一共分为**数据结构与算法**、**数学基础**、**统计机器学习**和**深度学习**四个部分。

点击[这里](#)查看具体使用指南。该手册有两种获取方式：

- 公众号后台回复【**思维导图**】获取完整手册（Xmind脑图源文件，学习起来更方便(๑ • _ •)๑）
- 点击 [卖萌屋@算法工程师思维导图](#) 专辑，查看对应部分内容。

下面是**数据结构与算法**部分~

数据结构

• 栈

■ 二维矩阵/直方图最大矩形面积

矩阵最大面积：<https://leetcode-cn.com/problems/maximal-rectangle/solution/zu-i-da-ju-xing-by-leetcode/>

直方图dp

■ 最小栈

<https://leetcode-cn.com/problems/min-stack/solution/xiang-xi-tong-su-de-si-lu-fe-n-xi-duo-jie-fa-by-38/>

• 链表

■ LRU：双向链表

<https://leetcode-cn.com/problems/lru-cache/Least Recently Used>

- 寻找重复数-环的入口

<https://leetcode-cn.com/problems/find-the-duplicate-number/>

快慢指针，相遇后，从头发起一个指针，按相同速度走，相遇即使环的入口

- 并查集

<http://www.cnblogs.com/cyjb/p/UnionFindSets.html>

- 字节跳动大闯关

https://blog.csdn.net/sinat_27705993/article/details/82053102

求不同并查集的个数 多一个count，每union一次count就减1

- 岛屿个数

<https://leetcode-cn.com/problems/number-of-islands/solution/dfs-bfs-bing-cha-ji-python-dai-ma-java-dai-ma-by-l/>

多开一个放0的岛

- 树

- 二叉树前序/中序/后序

前序：= 自顶向下

```
curr=stack.pop()
print(curr.val)
stack.push(curr.right)
stack.push(curr.left)
```

中序：加一个记录

```
curr = stack.pop()
if curr in cache:
    res.append(curr.val)
    continue
cache.add(curr)
stack.append(curr.right)
stack.append(curr)
stack.append(curr.left)
```

后序：同中序 = 自底向上

- 二叉树的最近公共祖先

三个条件满足两个就是True：1.左子树包含p1或p2 2.右子树包含p1或p2 3.自己是p1或p2

- 二分查找树

AVL：<https://mp.weixin.qq.com/s/dYP5-fM22BgM3viWg4V44A>

为啥有了BST和AVL还需要红黑树？<https://zhuanlan.zhihu.com/p/72505589>

AVL每次进行插入/删除节点的时候，几乎都会破坏平衡树的第二个规则，进

而我们都需要通过左旋和右旋来进行调整，使之再次成为一颗符合要求的平衡树 如果在那种插入、删除很频繁的场景中，平衡树需要频繁着进行调整，这会使平衡树的性能大打折扣，为了解决这个问题，于是有了红黑树

- 二叉树中的最大路径和

<https://leetcode-cn.com/problems/binary-tree-maximum-path-sum/>

- 完全二叉树插入

<https://blog.csdn.net/psc0606/article/details/48742239>

利用完全二叉树的性质，找到要插入的位置，先判断左子树的最右结点与右子树的最右结点高度，如果相等，只需要插入到左子树即可，否则插入右子树

- 完全二叉树节点数

<https://leetcode-cn.com/problems/count-complete-tree-nodes/solution/er-fen-cha-zhao-by-xu-yuan-shu/>

• 哈希表

- 连续子数组和为k的倍数

<https://leetcode-cn.com/problems/continuous-subarray-sum/solution/lian-xu-de-zi-shu-zu-he-by-leetcode/>

- 和为K的子数组

<https://leetcode-cn.com/problems/subarray-sum-equals-k/>

连续区间和为K，用字典存储累计和

- 求最长非重复字符串长度

https://blog.csdn.net/zd_nupt/article/details/82669299

做个hash_table表，记录每个字符的位置。碰到重复的就求两个重复字符之间的距离

- 前缀和+哈希表

查找

• 二分查找

- 查找最小值/翻转点 只判断mid-right是否被翻转

- 查找固定target 1.判断mid-right是否被翻转，找到升序的方向 2.跟升序区间的left/right和mid比看存在不在，不在就搜索另一个空间

- 存在重复，寻找最小值 当num[mid]==num[right]时，right-=1 因为左闭右开，mid和right中存在别的值 [0,1,1,1] [1,1,0,1] [1, 0, 1, 1, 1] [1, 1, 1, 1] 特殊情况下时间复杂度为O(N)

- bug-free写法：左闭右开，先写排除中位数的逻辑

<https://www.zhihu.com/question/36132386/answer/97729337> lower/upper bound

- 旋转数组

- 寻找峰值

左闭右开, 往高的地方走 <https://leetcode-cn.com/explore/learn/card/binary-search/210/template-ii/841/>

- 寻找重复数

<https://leetcode-cn.com/problems/find-the-duplicate-number/solution/xun-zhao-zhong-fu-shu-by-leetcode/>

- 双数组中位数

<https://leetcode-cn.com/problems/median-of-two-sorted-arrays/solution/xiang-xi-tong-su-de-si-lu-fen-xi-duo-jie-fa-by-w-2/>

- 找出第K小的距离对

<https://leetcode-cn.com/problems/find-k-th-smallest-pair-distance/solution/hei-ming-dan-zhong-de-sui-ji-shu-by-leetcode/>

二分查找 + 双指针

- 阶乘函数后K个零

<https://leetcode-cn.com/problems/preimage-size-of-factorial-zeroes-function/solution/jie-cheng-han-shu-hou-kge-ling-by-leetcode/>

- 乘法表中第k小的数

<https://leetcode-cn.com/problems/kth-smallest-number-in-multiplication-table/>

给定高度 m 、宽度 n 的一张 $m * n$ 的乘法表, 以及正整数 k , 你需要返回表中第 k 小的数字。

- BFS

- 迷宫中的最短路径

https://blog.csdn.net/qq_28468707/article/details/102786710

- 字符串A和B的最小相似度

<https://leetcode-cn.com/problems/k-similar-strings/solution/xiang-si-du-wei-k-de-zi-fu-chuan-by-leetcode/>

- 抖音红人

DFS: <https://blog.csdn.net/anlian523/article/details/82557468>

BFS: <https://blog.csdn.net/u014253011/article/details/82556976>

对于每个用户, 遍历粉丝数 (记录visited)

- DFS

- 八皇后

<https://blog.csdn.net/handsomekang/article/details/41308993>

一行一行依次遍历(从上往下), 决定放在哪列(从左往右), 这样就不用判断行冲突, 只需要判断列冲突和主斜线副斜线冲突。

对角线 \Rightarrow 斜率为1 $\Rightarrow \text{abs}(A[i]-A[j]) == \text{abs}(i-j)$

- 全排列

<https://leetcode-cn.com/problems/permutations/solution/>

有重复数字的全排列 sort: <https://leetcode-cn.com/problems/permutations-ii/>

- 复原IP地址

<https://blog.csdn.net/OneDeveloper/article/details/84946233>

dfs, 如果加完3个"."了则判断是否符合条件, 否则继续加 (start, start+3)

- 连通岛屿个数

字节-部门合并: https://blog.csdn.net/zd_nupt/article/details/82669299

dfs: 每次遍历到1, 则把联通的岛置为0

- 双指针

- 两数之和

<https://leetcode-cn.com/problems/two-sum-ii-input-array-is-sorted/solution/liang-shu-zhi-he-ii-shu-ru-you-xu-shu-zu-by-leetco/>

在已排序的数组中找到两个数, 和为target

双指针暴力求解 n^2 字典求解 时间 n , 空间 n

我们使用两个指针, 初始分别位于第一个元素和最后一个元素位置, 比较这两个元素之和与目标值的大小。如果和等于目标值, 我们发现了这个唯一解。如果比目标值小, 我们将较小元素指针增加一。如果比目标值大, 我们将较大指针减小一。移动指针后重复上述比较知道找到答案。时间 n , 空间1

- 数组中两数相减的最大值

<https://blog.csdn.net/fkyly/article/details/83930343>

非排序数组中两个数相减 (前面减后面) 的最大值。 $i < j, \max(a[i] - a[j])$

if $a[i] - a[j] > 0$: $j++$ else: $i = j, j++$

- 滑动窗口

- 最小覆盖子串

<https://leetcode-cn.com/problems/minimum-window-substring/solution/zui-xiao-fu-gai-zi-chuan-by-leetcode-2/>

- 和为K的子数组

<https://blog.csdn.net/a546167160/article/details/94401251>

当区间和等于target, 再向后遍历, 可以 $i+$ 或 $j+$, 但是 $j+$ 可能会越界, 因此选择 $i+$

- 乘积小于K的子数组

<https://leetcode-cn.com/problems/subarray-product-less-than-k/solution/cheng-ji-xiao-yu-k-de-zi-shu-zu-by-leetcode/>

排序

- 插入

- 插入排序：稳定

把后面的某个一次次插到前面，再管后面的，第一次确定的位置可能不是最终位置

- 希尔排序

- 选择

- 选择排序

每次选择最小的放到前面

- 堆排

- 交换

- 快速选择

- 冒泡排序：稳定

把某个确定好，再管其他的，第一次确定的位置是最终位置

- 快速排序

https://blog.csdn.net/qq_36528114/article/details/78667034

快排优化：

1. 在个数小于N时使用插入排序
2. 尾递归优化，减少递归栈的深度
3. 加入三取样切分 //省去了对重复元素的比较，性能接近线性

- 归并排序：稳定

原地归并：直接把合适的片段swap过去

<https://blog.csdn.net/xiaolewennofollow/article/details/50896881>

两个片段的交换需要三次逆转：分别逆转[1, i]和[i+1, n] 再逆转[1, n]

大数据归并应用较多

- 数组中的逆序对

<https://leetcode-cn.com/problems/shu-zu-zhong-de-ni-xu-dui-lcof/>

- 区间和的个数在某个区间

<https://leetcode-cn.com/problems/count-of-range-sum/>

- 基数排序：稳定

- 链表排序

<https://www.cnblogs.com/TenosDolt/p/3666585.html>

<https://leetcode-cn.com/problems/sort-list/solution/sort-list-gui-bing-pai-xu-lian-biao-by-jyd/>

- 拓扑排序

<https://www.cnblogs.com/fengziwei/p/7875355.html>

- 字典序

- 下一个排列

<https://leetcode-cn.com/articles/next-permutation/>

- 字典序的第K小数字

<https://leetcode-cn.com/problems/k-th-smallest-in-lexicographical-order/>

- 字典序排数-先序遍历

<https://leetcode-cn.com/problems/lexicographical-numbers/>

- 按字典序排在最后的子串

<https://leetcode-cn.com/problems/last-substring-in-lexicographical-order/>

- TopK问题

- 移除K位数字得到最小结果-栈

给定一个以字符串表示的非负整数 num, 移除这个数中的 k 位数字, 使得剩下的数字最小。 <https://leetcode-cn.com/problems/remove-k-digits/>

- 数组中前K个高频元素

<https://leetcode-cn.com/problems/top-k-frequent-elements/solution/leetcode-di-347-hao-wen-ti-qian-k-ge-gao-pin-yuan-/>

哈希统计频率+topK排序

堆排/快速选择/桶排

- 查找和最小的K对数字

<https://leetcode-cn.com/problems/find-k-pairs-with-smallest-sums/>

动态规划

- 编辑距离

<https://zhuanlan.zhihu.com/p/80682302>

- 最长回文子序列/子串

最长回文子序列: bbbab -> bbbb

<https://leetcode-cn.com/problems/longest-palindromic-subsequence/solution/dong-tai-gui-hua-si-yao-su-by-a380922457-3/>

- 最长回文子串: bbbab -> bbb <https://leetcode-cn.com/problems/longest-palindromic-substring/solution/zui-chang-hui-wen-zi-chuan-by-leetcode/>

- LCS

公共子串: 要求元素相邻: 矩阵最长对角线

- LIS

1. 排序后求LCS，时间 $O(n^2)$ ，空间 $O(n)$

2. $dp[i]$ 存储 $A[:i]$ 的LIS，每个 i 和前面的下标对比，时间 $O(n^2)$ ，空间 $O(n)$ for j in $[i-1, 0]$: if $A[j] < A[i]$: $dp[i] = \max(dp[i], dp[j]+1)$

3. $dp[i]$ 存储LIS为 $i+1$ 时最大的值，最后 $\text{len}(dp)$ 即为答案，时间 $O(n \log n)$ ，空间 $O(n)$ 二分法查找插入 dp 的位置

- 最大子序和

最大和包含当前和不包含： $\text{sum}[i] = \max(\text{sum}[i-1] + a[i], a[i])$

- 背包问题

https://blog.csdn.net/stack_queue/article/details/53544109

- 最短路径

Dijkstra：单源&边权非负

<https://www.jianshu.com/p/ff6db00ad866>

Floyd：全源&负环，任意两点间的最短路径，时间复杂度为 $O(N^3)$ ，空间复杂度为 $O(N^2)$

Bellmanford：单源

<https://blog.csdn.net/lpjishu/article/details/52413812>

Johnson：全源&非负环

- 股票问题

<https://leetcode-cn.com/problems/best-time-to-buy-and-sell-stock/solution/yi-ge-fang-fa-tuan-mie-6-dao-gu-piao-wen-ti-by-l-3/>

模式匹配

- 单模式单匹配：KMP

字符串匹配，返回第一个匹配的位置

<https://www.zhihu.com/question/21923021/answer/281346746>

- 多模式单匹配：Trie

- 多模式多匹配：AC自动机

大数据

- 蓄水池抽样法

选 k 个，新旧元素被选中第概率都是 k/n 第 $k+1$ 个以 $k/(k+1)$ 被选中，之前在水池里的被替换概率为 $k/(k+1) * 1/k = 1/(k+1)$ 则旧元素留下的概率为 $k/(k+1)$ ，与新元素相等

公众号后台回复【**思维导图**】获取完整手册（Xmind脑图源文件，学习起来更方便(๑•̀•́)๑

