方法论

五毒神掌

第一遍 5分钟想不出直接看答案
理解背诵最优解 比较
第二遍 第二天 自己做一遍（考试）
第三遍 第四天 做一遍
第四遍 一周后 做一遍
第五遍 面试/考试前 恢复记忆性训练

刷题4步

1.看清楚题，跟面试官确认你的理解
2.想出所有的解法，比较空间时间复杂度，并给出最优解
3.coding ,more coding!
4.test and check

自顶向下编程

先写好主干逻辑，再在后面补全具体细节

vs code使用技巧

1.**Shift+Alt+鼠标左键**
2.将自动高亮的变量、字符一次性替换：双击变量，右键'更改所有匹配项'。或者双击变量，**Ctrl+F2**
3.**删除当前字符串中当前光标之后的内容：Ctrl+Delete**
4.上下移动一行： **Alt+Up 或 Alt+Down**
向上向下复制一行： Shift+Alt+Up或Shift+Alt+Down
在当前行下边插入一行Ctrl+Enter
在当前行上方插入一行Ctrl+Shift+Enter
光标移动到行首：Home
光标移动到行尾：End
回退上一个光标操作Ctrl+U
选中当前行Ctrl+i（双击）
选择从光标到行尾Shift+End
选择从行首到光标处Shift+Home
选择单词 ctrl+d

代码规范

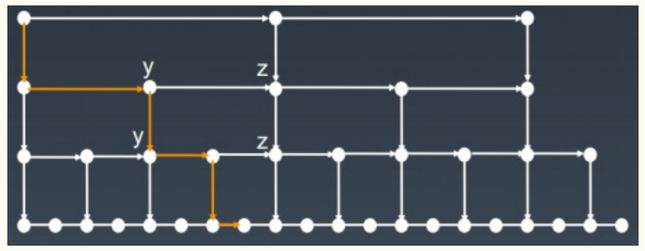
运算符、if 后面加空格、花括弧用c++风格
专有名词前加空格

数组、链表和跳表

数组：prepend O(n) n-1
append O(1)
search O(1)
insert O(n)
delete O(n)
实现：略，已掌握
链表：prepend O(1)
append O(1)
search O(n)
insert O(1)
delete O(1)
实现：略，已掌握
跳表：（单/双向、循环）
思想：增加维数，以空间换时间
使用前提：线性表有序
时间复杂度：O(logn)，因为高度为O(logn)
这里最高层索引有两个，第k层索引有n/2^k，所以n/2^h=2，所以h=log2(n)-1
查找、添加、删除操作都可以在对数期望时间下完成（但是维护成本高，要改索引跨度等）

空间复杂度：n(1/2+1/4+1/8+.....) O(n)

应用：有序集



（续跳表）
实现构思：
1.具体实现中跳表由于节点的增删不会太工整，节点的跨数会变

2.score 值可重复。
对比一个元素需要同时检查它的 score 和 memeber 。
每个节点带有**高度为 1 层的后退指针**，用于从表尾方向向表头方向迭代。

3.核心数据结构：

```
typedef struct zskiplist {
    // 头节点，尾节点
    struct zskiplistNode *header, *tail;
    // 节点数量
    unsigned long length;
    // 目前表内节点的最大层数
    int level;
} zskiplist;
typedef struct zskiplistNode {
    // member 对象
    robj *obj;
    // 分值
    double score;
    // 后退指针
    struct zskiplistNode *backward;
    // 层
    struct zskiplistLevel {
        // 前进指针
        struct zskiplistNode *forward;
        // 这个层跨越的节点数量
        unsigned int span;
    } level[];
} zskiplistNode;
```

数据结构

一维

数组、链表
高级：跳表、栈、队列（循环队列、优先队列、双端队列）

二维

树、图
高级：平衡树、二叉搜索树、红黑树、堆、并查集、字典树（字典树没接触过）

特殊

位运算 — 布隆过滤器
缓存 — LRU cache

栈和队列

1.队列（普通）
插入、删除O(1)、搜索O(n)
2.栈（普通）
插入、删除O(1)、搜索O(n)
3.双端队列
4.优先队列

算法

最基本的元素

分支(if else) 循环(for while) 递归
最终所有的复杂的算法都可以找到当中重复的部分去破解

搜索

BFS DFS

查找

二分查找

贪心

动态规划

数学/几何

复杂度分析

时间复杂度：主要是递归那里：看递归树
e.g.斐波那契数列，用递归去求第n个数，复杂度o(2^n)，递归树当中有很多重复节点
和主定理：
（二分搜索）T(n)=T(n/2)+O(1) O(logn)
（二叉树遍历） T(n)=2T(n/2)+O(1) O(n)
（最优排序矩阵搜索） T(n)=2T(n/2) +O(logn) O(n)
（归并排序）T(n)=2T(n/2)+O(n) O(nlogn)

空间复杂度：数组看个数、递归看树高