Week2

- 哈希表 映射 集合
  - 基础知识
    - 哈希表
      - 时间复杂度
        - 平均情况
          - 查询 O(1)
          - 插入 O(1)
          - 删除 O(1)
        - 最坏情况(哈希函数选择有问题)
          - 查询 O(n)
          - 插入 O(n)
          - 删除 O(n)
    - 集合
      - set
        - Java 中 set 是通过 map 进行内部实现的
    - 映射
      - map
        - HashMap
          - 通过哈希表实现，时间复杂度参考哈希表
        - TreeMap
          - 通过红黑树实现，平均和最坏的时间复杂度均为 O(logn)
  - 相关习题
    - 整体思路
      - 通过Map来存储中间
    - 两数之和
      - KEY
        - target - nums[k]
      - VALUE
        - k
    - 字母异位词
      - 字母异位词判定
        - KEY
          - 字符
        - VALUE
          - 第一个字符串添加词频
          - 第二个字符串删除词频
      - 字母异位词分组
        - KEY
          - 排序结果
        - VALUE
          - 排序后为KEY的原字符串列表
- 树 二叉树 二叉搜索树
  - 基础知识
    - 二叉树
    - 二叉搜索树
      - 左子树所有节点小于根节点
      - 右子树所有节点大于根节点
      - 中序遍历是升序排列
      - 时间复杂度
        - 平均情况
          - 查找 O(logn)
          - 插入 O(logn)
          - 删除 O(logn)
        - 最坏情况()
          - 查询 O(n)
          - 插入 O(n)
          - 删除 O(n)
          - 未进行平衡的二叉树
  - 相关习题
    - 二叉树遍历
      - 递归遍历 =
        - 判断是否为空
        - 根据前中后序的顺序递归或访问
      - 迭代遍历 =
        - 构建空栈 将root节点入栈
        - 当栈不为空循环
          - 出栈判断是否为空
            - 为空则访问下一个元素
            - 不为空按照递归的反顺序入栈
              - 左右节点不为空直接入栈
              - 根节点入栈后再推入一个空指针
    - 多叉树遍历
      - 注意：多叉树只有前序和后序
      - 递归遍历 =
        - 调用自节点的时候遍历列表
      - 迭代遍历 =
        - 入栈时反向遍历列表
- 堆 二叉堆 图
  - 基础知识
    - 图
    - 堆
      - 时间复杂度
        - 查找 O(1)
        - 插入
          - 二叉堆 O(logn)
          - 斐波那契堆 O(1)
        - 删除 O(1)
    - 二叉堆
      - 通过完全二叉搜索树得到
      - 根节点的值大于自节点的值
  - 相关习题
    - 利用堆维护最大或最小的k个数
      - 前 K 个高频元素 =
      - 最小的k个数 =
    - 利用堆存储某一中间状态，状态要求返回极值
      - 丑数 II =
        - 返回最小值需以限定的因子数
      - 滑动窗口最大值 =
        - 不断弹出最大值直到在滑动窗口内部

XMind:ZEN | 试用模式