

目录

1.1	第一章第一节.....	2
1.1.1	复杂度	2
1.1.2	排序	2
1.2	第三章	4
1.2.1	01:26:00: 题目 1——用数组结构实现大小固定的队列和栈	4
1.2.2	01:32:00: 题目 2——用队列结构实现大小固定的队列和栈	4
1.2.3	01:44:50: 实现一个特殊栈，在实现栈的基本功能的基础上，再实现返回栈中最小元素的操作	4
1.2.4	02:04:26	4
1.2.5	02:19:15: 猫狗队列.....	4
1.2.6	02:30:15: 认识哈希函数和哈希表.....	4
1.2.7	03:03:00:设计 RandomPlool	5
1.3	第四章	5
1.3.1	00:06:： 转圈打印矩阵（螺旋）	5
1.3.2	00:20:00: “之”字形打印	6
1.3.3	00:35:00: 在行列都拍好序的矩阵中找数	6
1.3.4	00:47:00: 打印两个有序链表公共部分	6
1.3.5	00:49:30: 判断链表回文结构	6
1.3.6	01:05:00（单项链表 partition）:将单向链表按某值划分左边小，中间相等，右边大形式.....	6
1.3.7	01:20:08: 链表的深复制（hashmap 简单+方法二暂时看不懂）	6
1.3.8	01:40:00 两个单链表相交的一系列问题（单链表最难）	7
1.3.9	02:21:00 布隆过滤器和一致性哈希	8
1.4	第五章	9
1.4.1	00:07:00: 一致性哈希	9
1.4.2	00:49:20: 优先队列应用：随时找到数据流的中位数	9
1.4.3	01:09:10 优先队列应用：最小铜板	10
1.4.4	01:20:36 优先队列应用：项目最大收益	11
1.4.5	01:36:40 二叉树先序、中序、后序非递归实现.....	11
1.4.6	折纸问题（二叉树）	12
1.4.7	打印二叉树 01:25:00	13
1.4.8	02:28:00 找到二叉树的后继节点(中序遍历应用).....	13
1.4.9	02:50:00 在数组中找到一个局部最小值	15
1.5	第六章	15

1.5.1	00: 10:45 并查集 class4_09.....	15
1.5.2	01:09:38 前缀树 class5_01	17
1.5.3	01:44:00 图 class5_Edge-Graph-Node-GraphGenerator	17
1.5.4	02:01:00 图的宽度优先遍历-class5-02BFS.....	19
1.5.5	02:09:30 深度优先遍历 class5-03.....	20
1.5.6	02:21:50 图拓扑排序算法	21
1.5.7	02:41:46 最小生成树 K 算法 class_05	22
1.5.8	02:53:05 最小生成树 P 算法 class_06.....	22

算法：给一个数据源（数据结构），结构之上功能设计流程。

优先队列：性质为堆

1.1 第一章第一节

1.1.1 复杂度

认识时间复杂度

时间复杂度为，一个算法流程中，常数操作数量的指标，这个指标叫做O，big O。具体为，如果常数操作数量的表达式中，只要高阶项，不要低阶项，也不要高阶项系数之后，剩下的部分记为f(N)，那么该算法的时间复杂度为O(f(N))

● 时间复杂度

An^2+bn+c ：不考虑低阶项，和高阶系数，即复杂度为 $O(n^2)$ 【读做 big ON 平方】

二分复杂度： $O(\log N)$ 【即 2 为底的对数函数】

● 空间复杂度

通常为额外空间复杂度，即输入输出不算。只算辅助中间加入的中间数据结构与输入输出之间产生的复杂度

最优解：先满足时间复杂度，使用最小空间复杂度

对数器的应用：暴力方法，产生高数据量的随机数比较测试

贪心算法：提出自己策略，使用对数器

1.1.2 排序

排序稳定性：不是指复杂度忽高忽低。无序序列相同数排序后，相同值的相对位置保持一致。

稳定序排序：冒泡排序、插入排序、归并排序

插入排序：常数项低。排序时数组小于 60 时使用非常合适

稳定性出发：基础类型：快排；自定义类型：归并排序（常数项比快排大）

比较器使用：定义自己比较器

● 排序方法

时间复杂度 $O(N^2)$ ，额外空间复杂度 $O(1)$ ，实现可以做到稳定性

1) 冒泡排序

2) 选择排序

3) 插入排序

$O(N^2)$ -- $O(1)$ 【有限几个变量】

三大排序： $O(N \log N)$

4) 归并排序 $O(N \log N)$

递归（栈）

$$T(n) = 2T\left(\frac{n}{2}\right) + O(N)$$

$$T(n) = aT\left(\frac{n}{b}\right) + O(N^d)$$

$$\begin{matrix} a=2 \\ b=2 \\ d=1 \end{matrix} \quad \begin{cases} \log_b a < d & O(N^d) \\ \log_b a > d & O(N^{\log_b a}) \\ \log_b a = d & O(N^d \log N) \end{cases}$$

应用：求小和，降序队

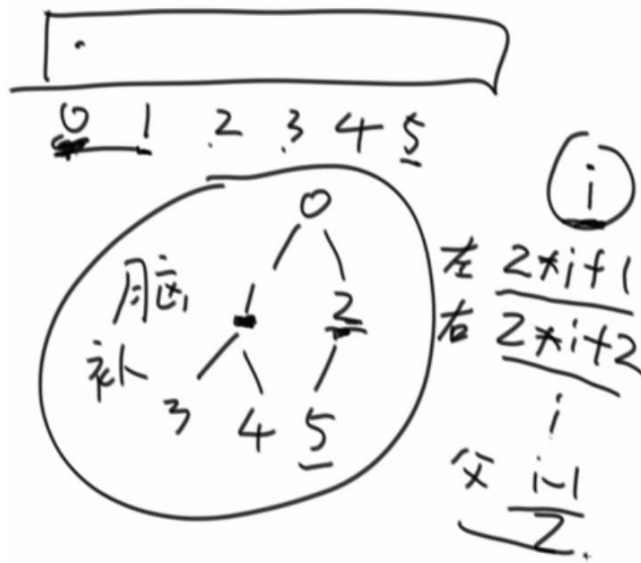
论文：归并排序内部缓存法

5) 快速排序 $O(\log N)$

应用：荷兰国旗问题

论文：0-1Stable：难，实现稳定快排【奇数左偶数右】

6) 堆排序 $O(1)$ ：第二章 01:56



完全二叉树+大根堆

缺点（工程上用的少）：不稳定，常数项比较大

7) 桶排序：第三章 00:33

1.2 第三章

1.2.1 01:26:00：题目 1——用数组结构实现大小固定的队列和栈

1.2.2 01:32:00：题目 2——用队列结构实现大小固定的队列和栈

1.2.3 01:44:50：实现一个特殊栈，在实现栈的基本功能的基础上，再实现返回栈中最小元素的操作

实现一个特殊的栈，在实现栈的基本功能的基础上，再实现返回栈中最小元素的操作。

【要求】

1. pop、push、getMin操作的时间复杂度都是 $O(1)$ 。

2. 设计的栈类型可以使用现成的栈结构。

1.2.4 02:04:26

如何仅用队列结构实现栈结构？

如何仅用栈结构实现队列结构？

1.2.5 02:19:15：猫狗队列

原则 1：弹出栈为空的时候，压入栈压入数据

原则 2：要导入要全部导完

1.2.6 02:30:15：认识哈希函数和哈希表

- 1) 输入很大（无穷大），输出固定【非随机，无任何随机因素】
- 2) 相同输入相同输出

- 3) 不同输入也会有相同输出（哈希碰撞）
- 4) 最重要特性：输出域具有几乎均匀分布性（即离散型很好）。——此特性越好，代表哈希函数越好

应用：打乱输入规律

Hashmap：增删改查（时间复杂度 $O(1)$ ），常数项大（算的时候代价大）

1.2.7 03:03:00:设计 RandomPool

设计RandomPool结构

【题目】

设计一种结构，在该结构中有如下三个功能：

insert(key)：将某个key加入到该结构，做到不重复加入。

delete(key)：将原本在结构中的某个key移除。

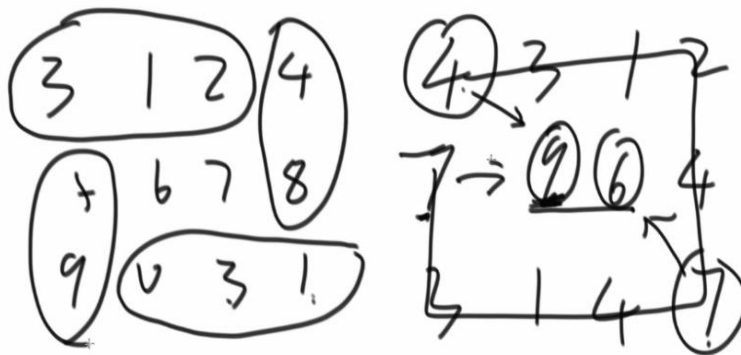
getRandom()：等概率随机返回结构中的任何一个key。

【要求】

Insert、delete和getRandom方法的时间复杂度都是 $O(1)$ 。

1.3 第四章

1.3.1 00:06:: 转圈打印矩阵（螺旋）



1.3.2 00:20:00: “之”字形打印

1.3.3 00:35:00: 在行列都拍好序的矩阵中找数

1.3.4 00:47:00: 打印两个有序链表公共部分

1.3.5 00:49:30: 判断链表回文结构

判断一个链表是否为回文结构

【题目】

给定一个链表的头节点head，请判断该链表是否为回文结构。

例如：

1->2->1，返回true。

1->2->2->1，返回true。

15->6->15，返回true。

1->2->3，返回false。

进阶：

如果链表长度为N，时间复杂度达到 $O(N)$ ，额外空间复杂度达到 $O(1)$ 。

1.3.6 01:05:00（单项链表 partition）：将单向链表按某值划分左边小，中间相等，右边大形式

1.3.7 01:20:08: 链表的深复制（hashmap 简单+方法二暂时看不懂）

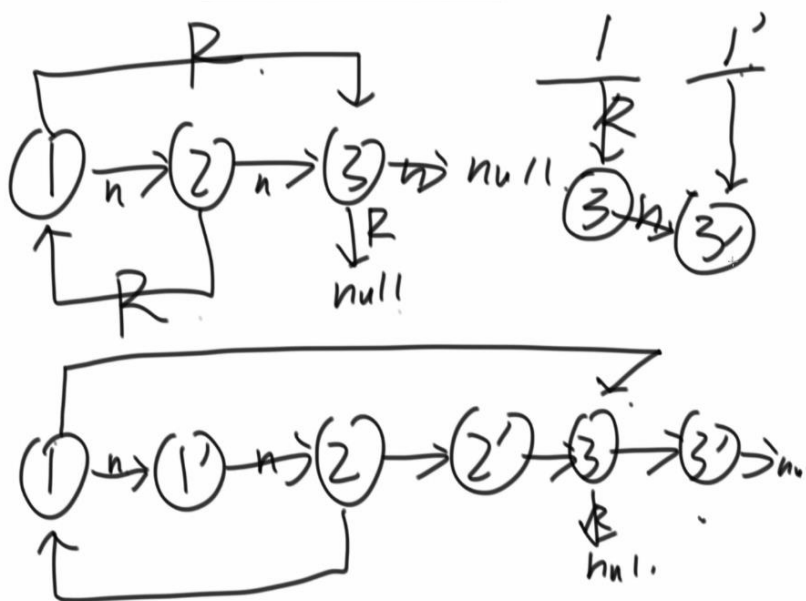
题目十三

Node类中的value是节点值，next指针和正常单链表中next指针的意义一样，都指向下一个节点，rand指针是Node类中新增的指针，这个指针可能指向链表中的任意一个节点，也可能指向null。

给定一个由Node节点类型组成的无环单链表的头节点head，请实现一个函数完成这个链表中所有结构的复制，并返回复制的新链表的头节点。

进阶：不使用额外的数据结构，只用有限几个变量，且在时间复杂度为 $O(N)$ 内完成原问题要实现的函数。

01:34:10 不用 hash 表实现上面的问题



1.3.8 01:40:00 两个单链表相交的一系列问题（单链表最难）

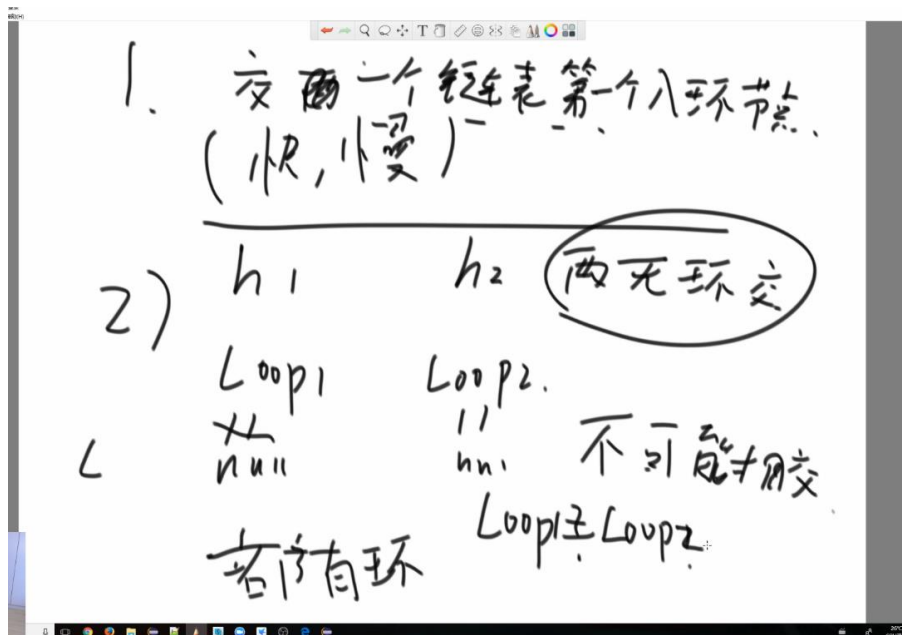
题目十四

两个单链表相交的一系列问题

【题目】

在本题中，单链表可能有环，也可能无环。给定两个单链表的头节点 head1 和 head2，这两个链表可能相交，也可能不相交。请实现一个函数，如果两个链表相交，请返回相交的第一个节点；如果不相交，返回 null 即可。

要求：如果链表1的长度为N，链表2的长度为M，时间复杂度请达到 $O(N+M)$ ，额外空间复杂度请达到 $O(1)$ 。



1.3.9 02:21:00 布隆过滤器和一致性哈希

1. 爬虫
2. 黑名单问题
3. 单样本量大
4. 空间要求
5. 允许有一定的失误差

集合概念, 允许有一定的失误差 (正常的被过滤, 另可错杀也不可放过一个):
很少空间

布隆器空间 m (向上取整) 计算公式:

$$m = - \frac{n * \ln P}{(\ln 2)^2}$$

K 个哈希函数 (向上取整)

$$K = \ln 2 * \frac{m}{n} \approx 0.7 * \frac{m}{n}$$

失误差 $P(\text{真}) < P$:

$$P_{\text{真}} = (1 - e^{-\frac{n * K}{m}})^K \quad (P_{\text{真}} < P)$$

1.4 第五章

1.4.1 00:07:00: 一致性哈希

- 1) 哈希离散性：当数量大的时候才适用，才能保证环分配均匀
解决：虚拟节点计算应用，
- 2) 应用于分布式系统

1.4.2 00:49:20: 优先队列应用：随时找到数据流的中位数

优先级队列，加入弹出都是 $O(\log N)$

题目一

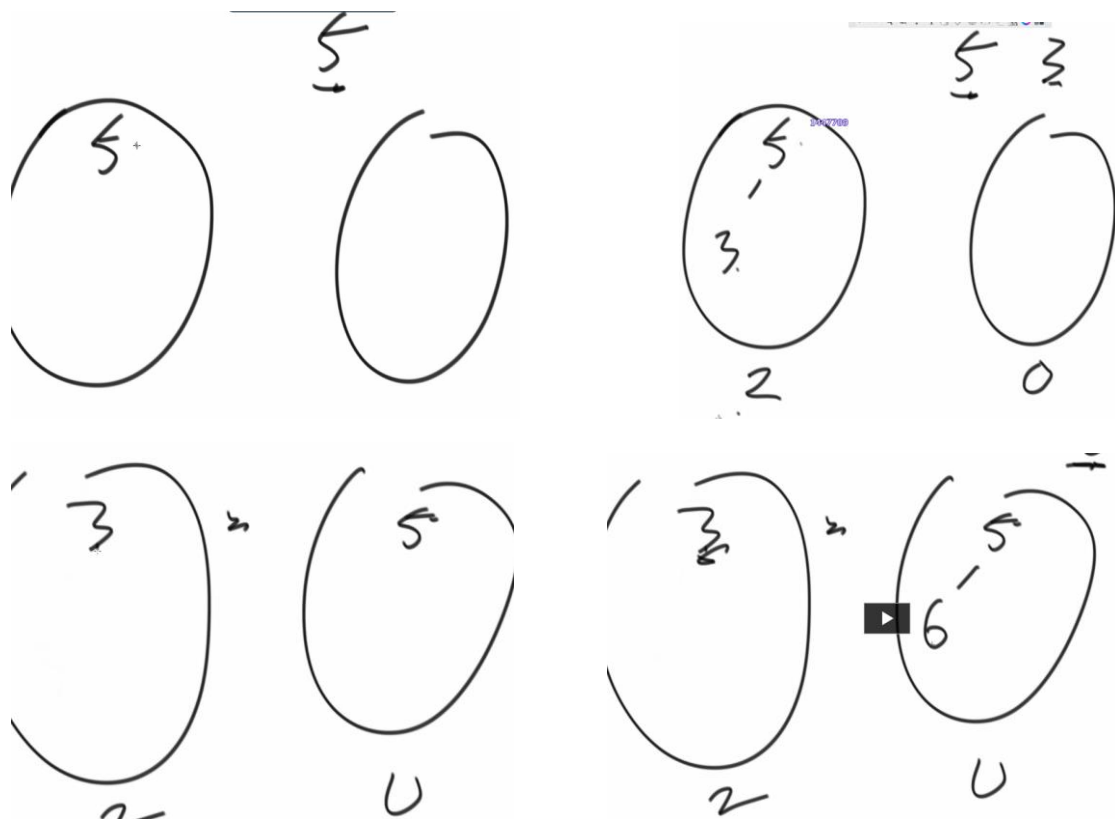
随时找到数据流的中位数

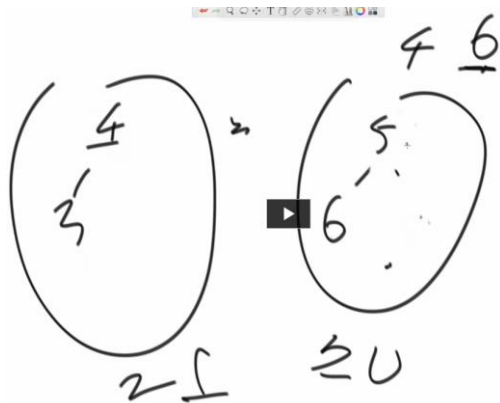
【题目】

有一个源源不断地吐出整数的数据流，假设你有足够的空间来保存吐出的数。请设计一个名叫MedianHolder的结构，MedianHolder可以随时取得之前吐出所有数的中位数。

【要求】

1. 如果MedianHolder已经保存了吐出的N个数，那么任意时刻将一个新数加入到MedianHolder的过程，其时间复杂度是 $O(\log N)$ 。
2. 取得已经吐出的N个数整体的中位数的过程，时间复杂度为 $O(1)$ 。





1.4.3 01:09:10 优先队列应用：最小铜板

- 1) 小根堆
- 2) 弹出两个数，放回小根堆
- 3) 重复 1) 和 2)

题目二

一块金条切成两半，是需要花费和长度数值一样的铜板的。比如长度为20的金条，不管切成长度多大的两半，都要花费20个铜板。一群人想整分整块金条，怎么分最省铜板？

例如，给定数组 {10, 20, 30}，代表一共三个人，整块金条长度为10+20+30=60。金条要分成10, 20, 30三个部分。

如果，

先把长度60的金条分成10和50，花费60

再把长度50的金条分成20和30，花费50

一共花费110铜板。

但是如果，

先把长度60的金条分成30和30，花费60

再把长度30金条分成10和20，花费30

一共花费90铜板。

输入一个数组，返回分割的最小代价。



1.4.4 01:20:36 优先队列应用：项目最大收益

题目三

输入：

参数1，正数数组costs

参数2，正数数组profits

参数3，正数k

参数4，正数m

costs[i]表示i号项目的花费

profits[i]表示i号项目在扣除花费之后还能挣到的钱(利润)

k表示你不能并行、只能串行的最多做k个项目

m表示你初始的资金

说明：你每做完一个项目，马上获得的收益，可以支持你去做下一个项目。

输出：

你最后获得的最大钱数。

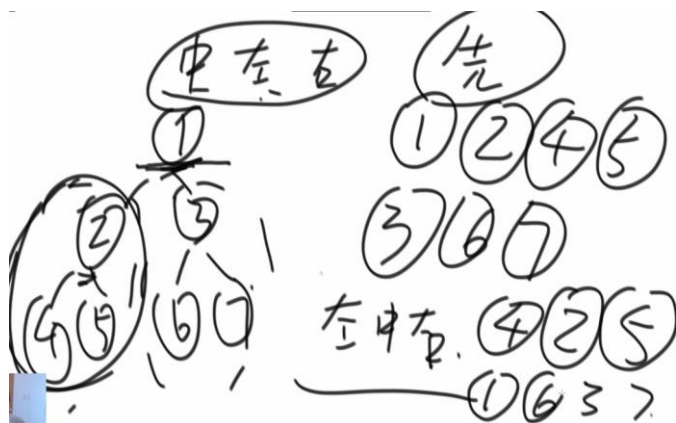


1.4.5 01:36:40 二叉树先序、中序、后序非递归实现

先序遍历：先中再左再右



中序遍历：左中右



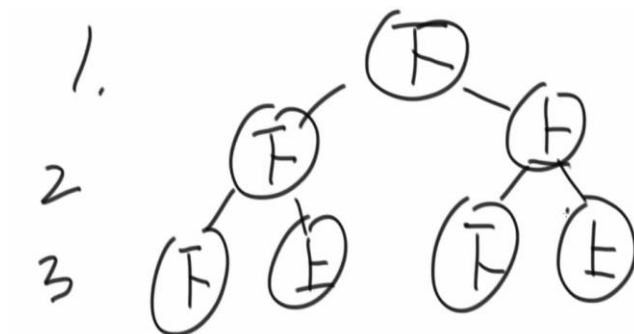
后序遍历：左右中



452

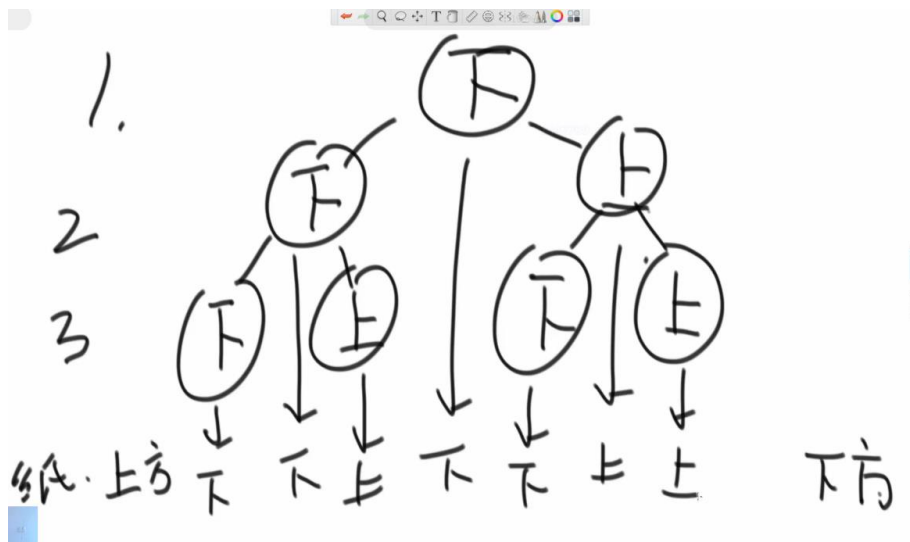
6731

1.4.6 折纸问题（二叉树）



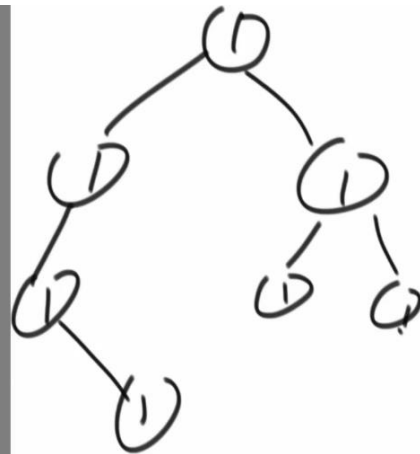
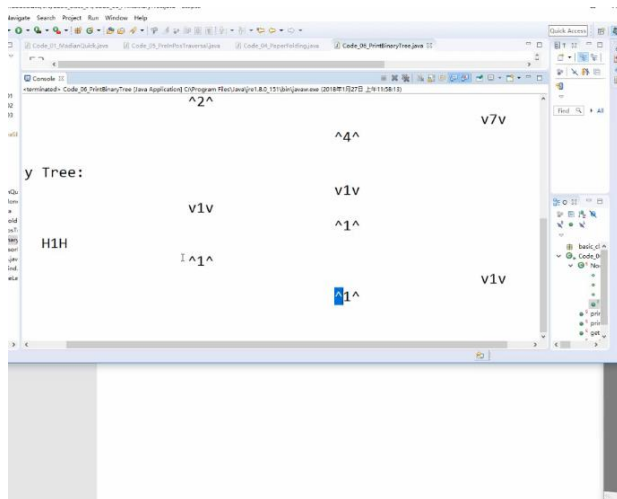
纸.上方

下下



(等价：中序遍历)

1.4.7 打印二叉树 01:25:00



1.4.8 02:28:00 找到二叉树的后继节点(中序遍历应用)

题目七

在二叉树中找到一个节点的后继节点

【题目】

现在有一种新的二叉树节点类型如下：

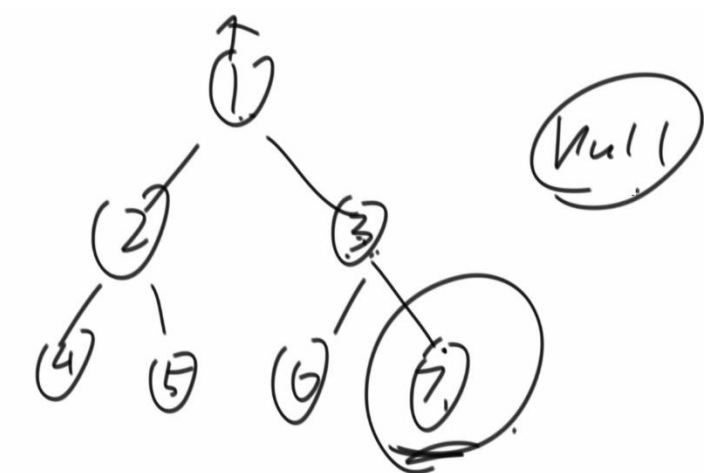
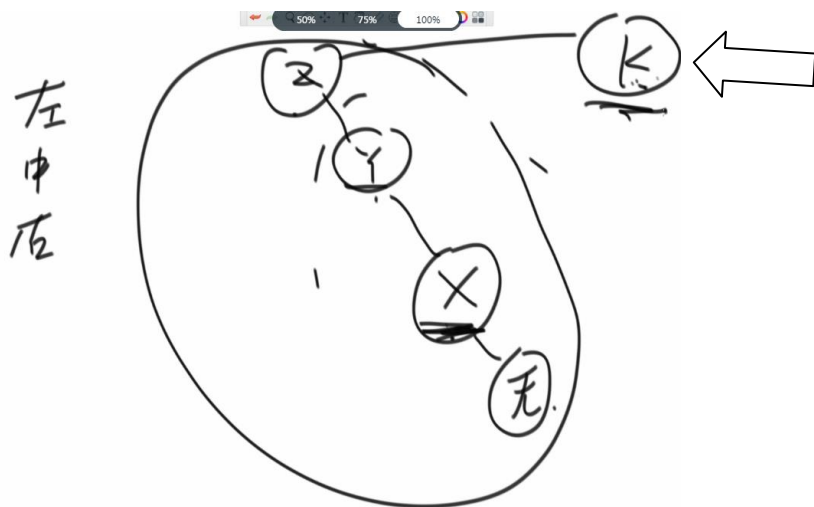
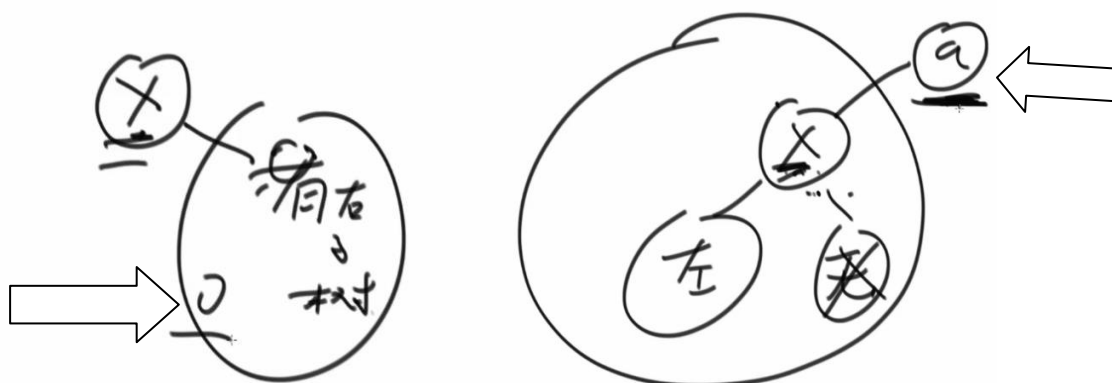
```
public class Node {
    public int value;
    public Node left;
    public Node right;
    public Node parent;

    public Node(int data) {
        this.value = data;
    }
}
```

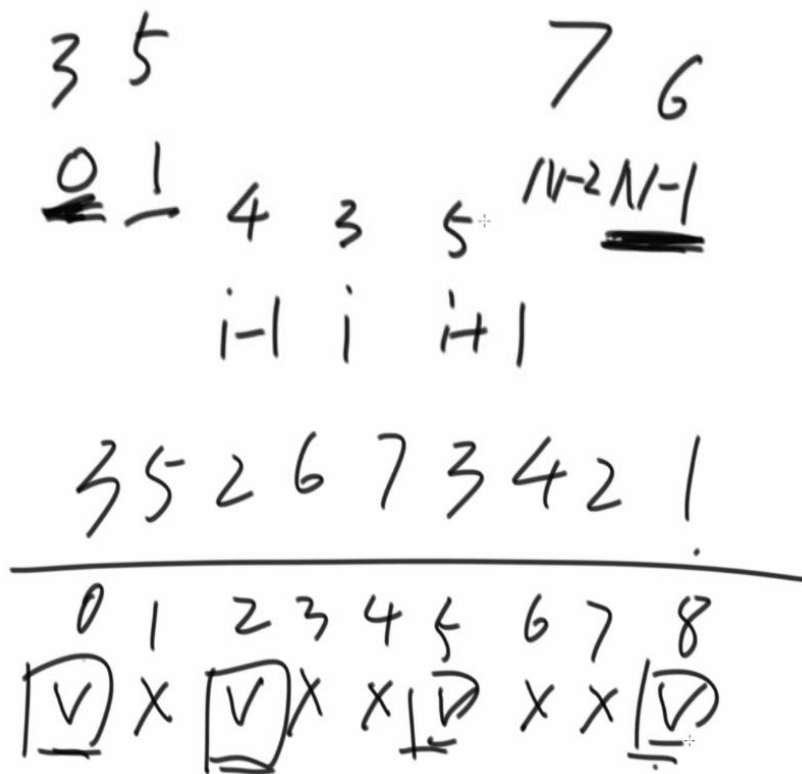
该结构比普通二叉树节点结构多了一个指向父节点的parent指针。假设有一棵Node类型的节点组成的二叉树，树中每个节点的parent指针都正确地指向自己的父节点，头节点的parent指向null。只给一个在二叉树中的某个节点node，请实现返回node的后继节点的函数。在二叉树的中序遍历的序列中，node的下一个节点叫作node的后继节点。

左 中 右

(左) 中 右



1.4.9 02:50:00 在数组中找到一个局部最小值



1.5 第六章

2018 年 06 月 09 日

1.5.1 00: 10:45 并查集 class4_09

应用图比较多，相当有用，面试图比较少。

元素个数 N 、查询此数+合并次数= N 或以上

单次查询和单次合并平均复杂度为：

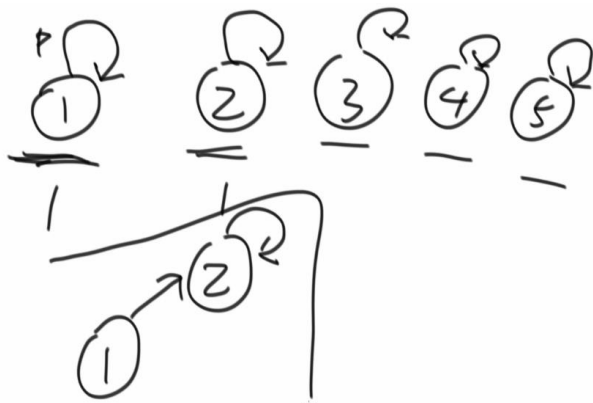
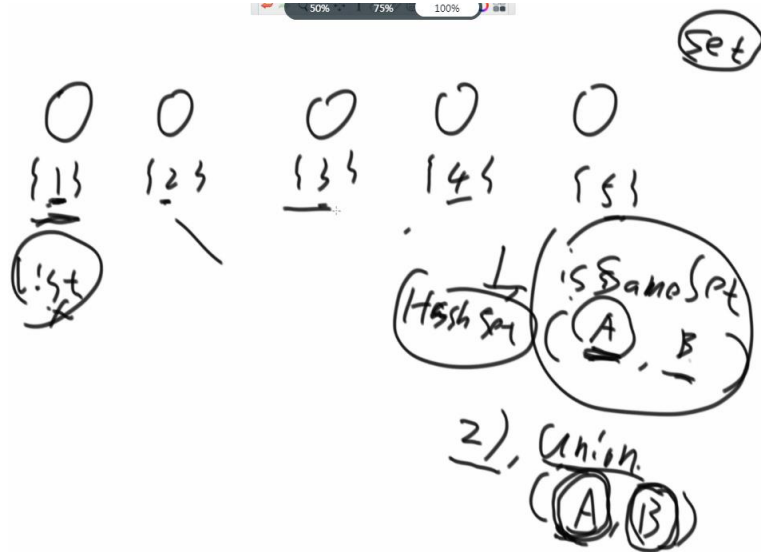
时间复杂度： $O(1)$

空间复杂度： $O(1)$

使下面两次操作代价最低

isSameSet

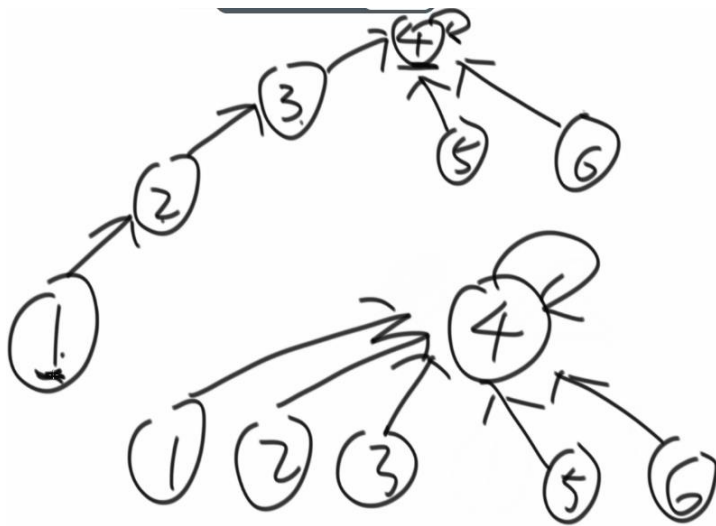
union (合并)



查询一个数字所在集合：一直向上找，找到集合的“代表点”



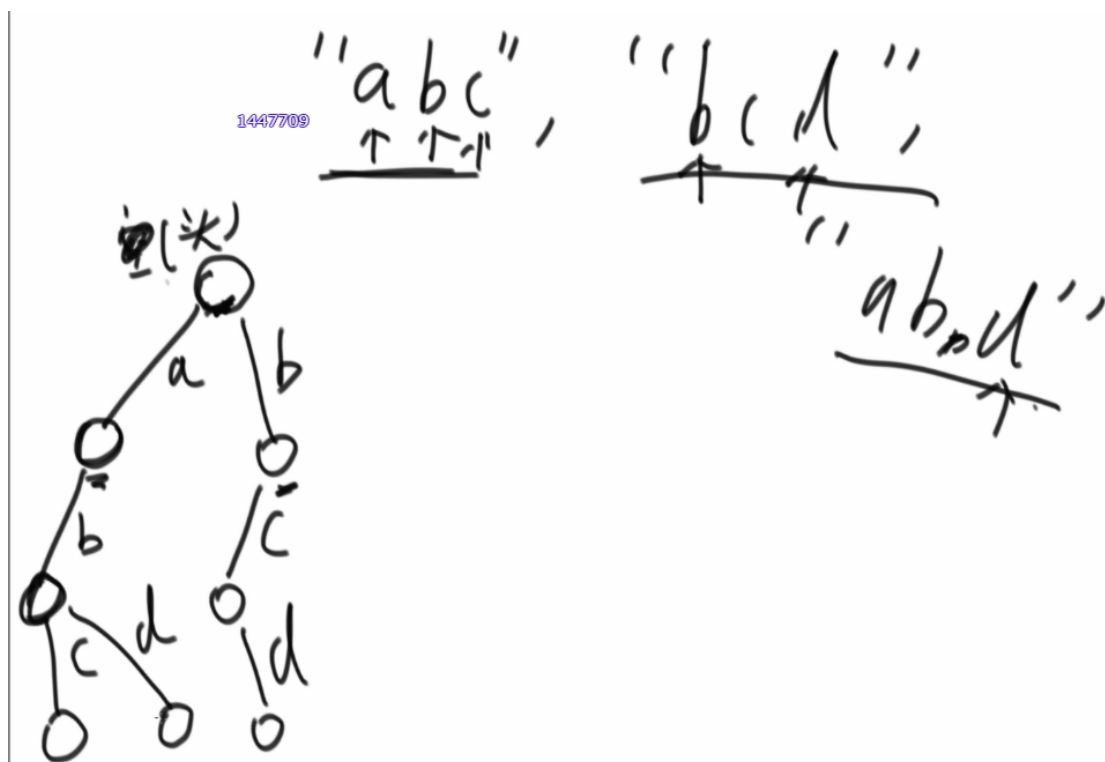
集合合并，较小的集合挂在较大集合代表点下面



查询完后，所有数指向集合代表点

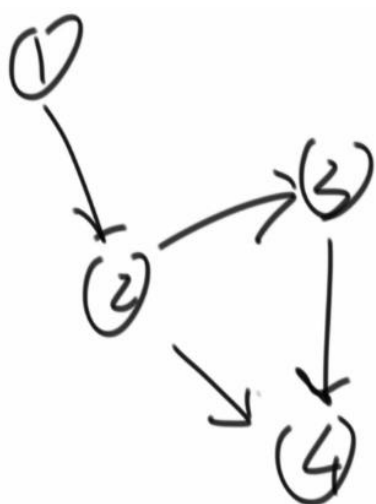
1.5.2 01:09:38 前缀树 class5_01

应用：查询字符串出现的次数



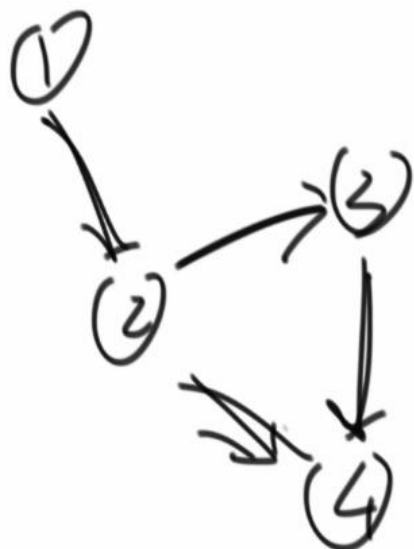
1.5.3 01:44:00 图 class5_Edge-Graph-Node-GraphGenerator

邻接图—有向图



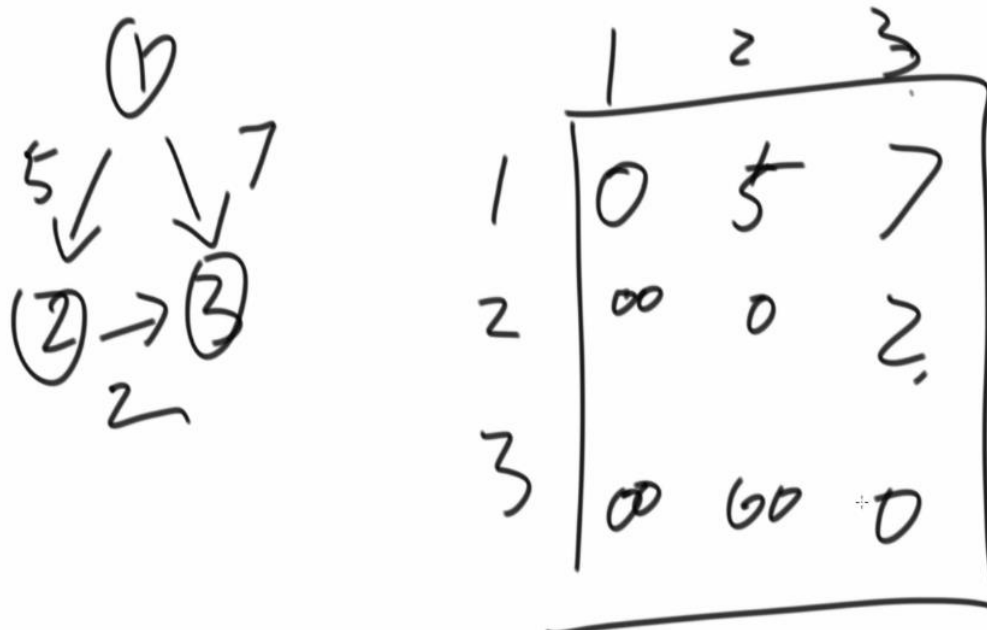
① : ②
 ② : ③ ④
 ③ : ④
 ④ : null

邻接图-无向图

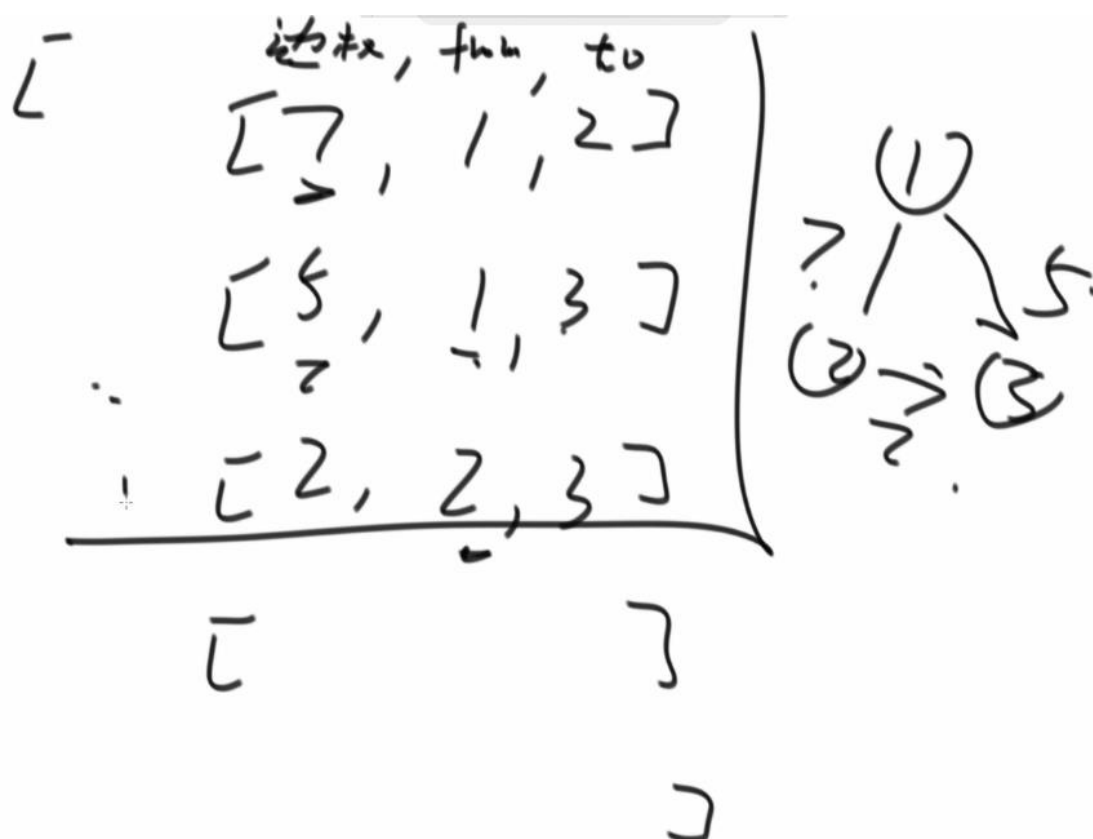


① : ②
 ② : ① ③ ④
 ③ : ④
 ④ : null

邻接矩阵:

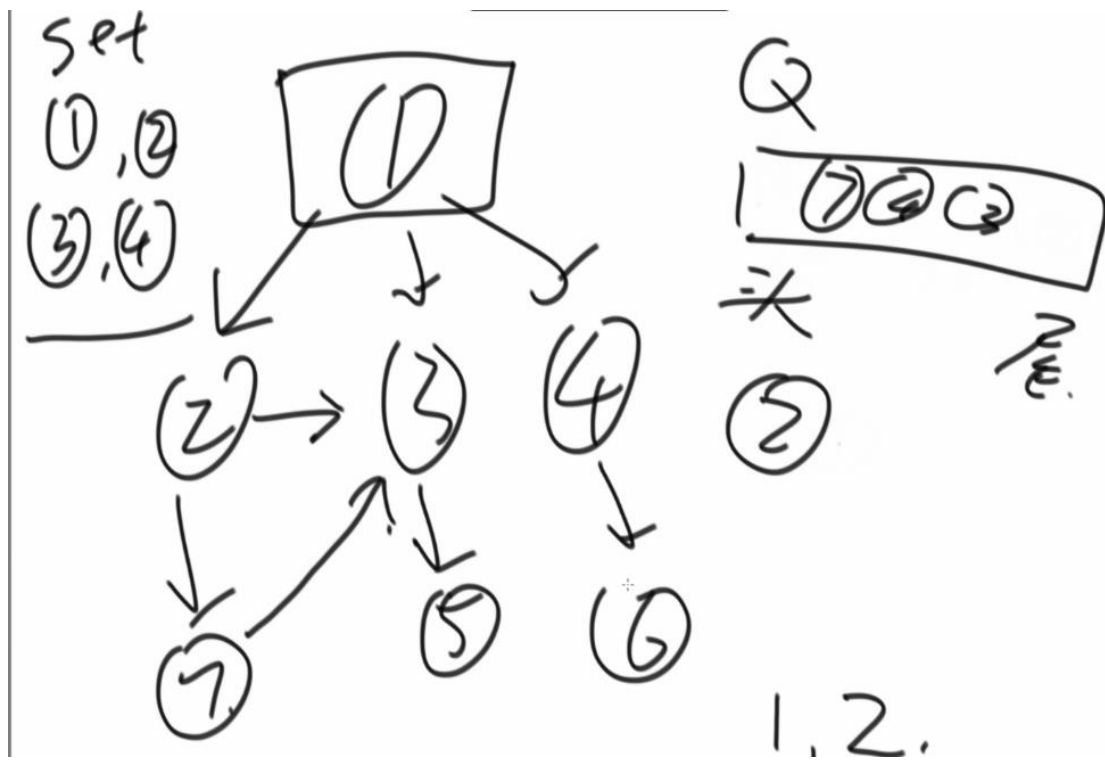


面试中表达图:



1.5.4 02:01:00 图的宽度优先遍历-class5-02BFS

队列应用



逐级打印离 1 近的点:

1

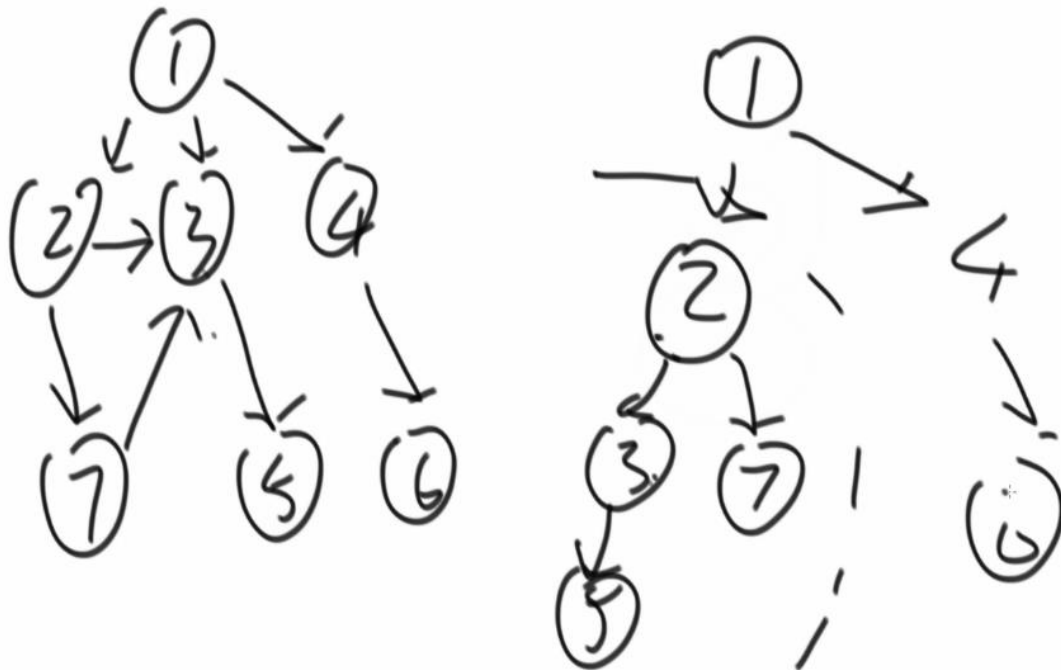
2,3,4

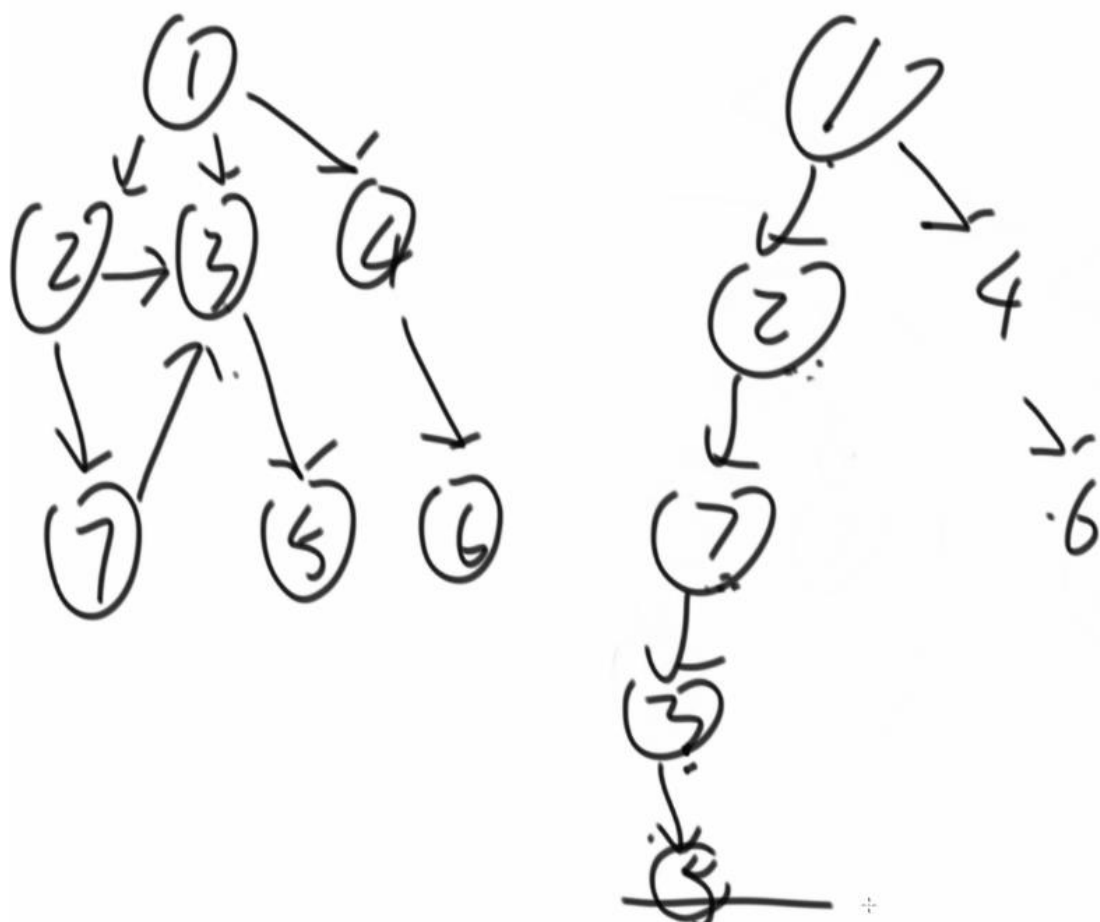
5,6,7

1.5.5 02:09:30 深度优先遍历 class5-03

栈应用

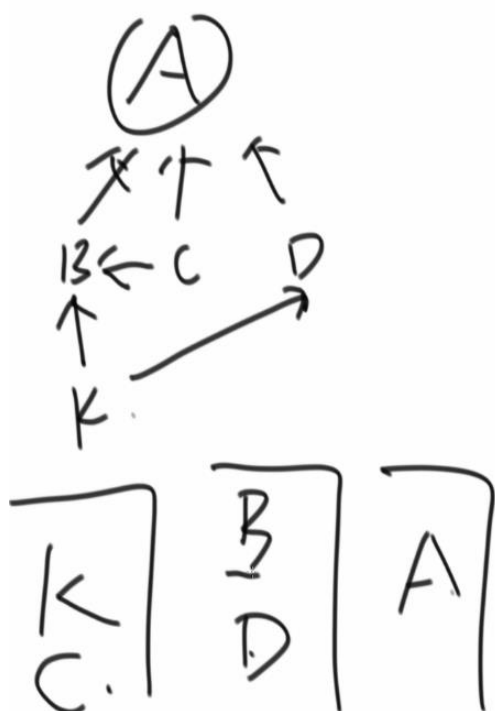
一条路走尽头才继续下一条





1.5.6 02:21:50 图拓扑排序算法

决定一件事情先做哪个再做哪个：原理像编译库
有向没有环路



算法实现：

1. 先找到入度为 0 的节点
2. 删掉步骤 1 的节点，再寻找入度为 0 的节点

1.5.7 02:41:46 最小生成树 K 算法 class_05

K 算法：依次选小权重的边，形成连通最后一个边丢弃
无向图

最小连通集合（权值最小）



1.5.8 02:53:05 最小生成树 P 算法 class_06

P 算法：先解锁点，再解锁边