这是我根据搜集到的一些信息做的复习提纲，主要是根据 09 级的一些回想和 10 猪脚 GG 给出的往年重点（虽然这两方面都说今年应该不会大变，但 ljq 老师没有说）。所以本提纲不是基于今年试卷的，希望各位能自己斟酌一个完整的复习计划。对于没有复习本提纲外的东西以及硬背本提纲（要理解哦）导致的任何严重后果，概不负责~\(≧▽≦)/~

提纲基本是围绕着 ljq 老师的 summaryPPT 的，下划线是注意要理解的内容，重点部分予以加粗。不加说明的 PPT 页数均指那一课的 PPT，而非 summary

0、 程序设计基础

程序的一般概念：程序=数据结构+算法

1、 绪论

① 数据的存储结构 – 逻辑结构在存储器中的映像
数据元素是用二进制位串表示的
关系的话则可以用顺序映象（以相对的存储位置表示后继关系）和链式映象（以附加信息，如指针，表示后继关系）的方式表示

② 抽象数据结构（ADT）- 一个数学模型及定义在这个数学模型上的一组操作
两个重要特征 – 数据抽象、数据封装
描述方法 - （D，S，P）

③ 算法 – 为了解决某类问题而规定的有限长操作序列
五个重要特性：有穷性、确定性、可行性、有输入、有输出
设计原则：正确性、可读性、健壮性、高效率和存储需求
效率衡量方法：事后统计 – 事后统计法 – 缺点
事前分析评估 – 问题的规模 – 时间复杂度的估计
空间复杂度 – 原地工作

2、 线性表（比较简单，注重理解就好）

线性表的基本特征，顺序映象
单链表的结构，插入操作
双向链表的结构，插入操作，删除操作，
链表的改进 LinkList 的引入（主要是加了头结点指针、尾节点指针、当前节点指针和表长）和使用指针 p 去访问链表（其实一般都是用 p 去访问的，顺序表才会用索引值 i）

3、 栈和队列

栈和队列的结构特征

表达式的三种标识方法：前缀表示法（波兰式）、中缀表示法、后缀标识法（逆波兰式）
由原表达式得到后缀表达式方法（忠实执行课件中算法即可，对于“(”可以认为从这个点开始不再使用原来栈，而是转移到一个新的临时栈继续执行，到“)”的时候将新栈中残余内容全部依次输出并返回原来的栈）

顺序栈表示、链栈表示

顺序队列、链式队列

队列应用 – 计算 N 行杨辉三角（请参照 chap003 的 PPT93 页的算法，不过其实这问题是个坑，我研究了 40 分钟结论如下——首先 PPT93 页的算法算“一轮”是没问题的，看它循环的退出条件也能发现它就是算“一轮”用的，对应 PPT 图中的“第一轮”，也就是算第二行也是对的。但图中算“第二轮”，也即第三行是明显不对的——错在最后一步。为什么会错，根本上说，PPT 中对循环队列的定义要求在队列满时必须“浪费”一个存储单元，已

保证“满”和“空”的判定条件不混淆，所以，其实要算出第三行，其实循环队列长度应该至少为 6，或者修改循环队列定义，让头尾指针在“满”时可以重叠，代价是有一个单元读不出。关于这个问题，纯属个人看法，欢迎有意见同学和我讨论。

划分无冲突子集问题：稍微注意一下 clash 数组的作用和理解 i<pre 这个判断（这是判定 i“有资格”创建新组的关键，i 是“本轮”第一个从队列中出来的元素，注意这里“一轮”开始的标志是从队列中取出了上一轮插入队列的元素）

4、 串

串的 ADT 定义 —— 串的逻辑结构和线性表极为相似，区别在于串的数据对象约束为字符集；串的基本操作和线性表有很大差别——线性表的基本操作多以“单个元素”，但串的基本操作多对“串的整体”

串的三种存储表示

串的模式匹配算法 —— KMP 算法

注意：①主串不回溯，子串部分回溯（子串的回溯免去了主串回溯的必要——因为其实当前有效的这一部分主串 == 子串中已经到达的位置前面的那一部分）

②next 值的计算方法，其实就是用子串自身去匹配子串（不过如果两者都从第一个字符开始，则会完全匹配，所以第一个默认是不匹配，体现在代码中就是设 j=0 但设 i = 1,而不是 0），next 值的含义参见 ppt69 页，说得比较清楚

5、 数组和广义表

数组的顺序表示和实现

稀疏矩阵的定义和压缩存储方法

广义表的定义

递归函数的设计方法 ——分治法解决 hanoi 问题（这个在 summaryPPT 的第 62 页讲得挺清楚了）

6、 树和二叉树

基础概念：结点、结点的度、树的度、叶子结点、分支结点、路径、孩子结点、双亲结点、兄弟结点、祖先结点、子孙结点、结点的层次、树的深度

森林、树、二叉树（注意虽然树和二叉树都采用了递归定义，但二叉树并不是树的一个子概念）、满二叉树、完全二叉树

三序遍历（这个分递归和非递归两种方法，递归简单到爆，非递归要借助栈，并不复杂，请自行练习），**由表达式建树（分先缀表达式和原表达式，首先明确要建立的树应该是由叶子到根应该是按照计算的顺序来的，先缀采用递归算法可以很容易实现，参见 chap006PPT 的 72 页，原表达式麻烦一些，因为可能还会有括号，必须用非递归的方法实现。采用双栈——一个栈 S 用来维护符号之间的关系，一个栈 PTR 用来保证双亲结点能链接到正确的孩子结点，对于括号，采取的是”借用 S 中一段“的处理方式，借用在读到右括号后结束。结点建立按照叶子到根的顺序。特别说明一下，PPT77 页中 precede 这个函数没有说清楚，个人观察应该是比较运算符的优先级，c 高于 ch 时输出为有效，从中也可以看出，”(“的优先级是很低的，因为”(”后面一个运算符必定会被暂存，而结束符“#”则很高，保证在算法最后所有的符号都能建立结点。）**由先序和中序序列建树，仔细看 PPT 第 80 页的图就能明白。

树的三种存储结构，树、森林、二叉树三者之间对应关系，遍历方式的对应关系（这个理解的话建议围绕着树-二叉树和森林-二叉树这两个联系进行，先理解结构上关系，再

理解三者遍历各自的定义，然后从递归式+结构关系就很好理解为什么遍历是这样的对应关系了）

哈夫曼树与哈夫曼编码

最优树的定义，最优树的构造方法（在 summaryPPT 中几页也说得比较清楚了，照着这个操作就好，也挺容易理解，不懂可以看 chap006），前缀编码（应简单理解一下前缀编码的概念，而一个哈夫曼编码序列就是从树根到当前节点上所有节点的 0/1 值的组合，在这里把每个结点的权值理解为当前编码值出现的概率，所以哈夫曼树能保证统计上看所有编码序列的期望长度最短。）

7、图

图的结构定义，名词术语（这里就不列了）

图的存储表示（都看一看，主要是邻接表和邻接矩阵）

图的遍历

深度优先遍历（允许递归，所以还是不难的，注意访问标识很重要）

广度优先遍历（队列） —— 最小生成树算法（本质上讲，广度优先遍历就是所有边权值=1 时的最小生成树 ——个人理解）分为普利姆算法和克鲁斯卡尔算法，这两个算法都是贪心算法，所以本质上是比较简单，所谓贪心，就是每次选择是，都选“目前看上去最好”的选择，然后选着选着就选出了全局的最好结果（从这段描述大家也可以看出，贪心不一定是对的，贪心的正确性需要专门的证明，这里的普利姆和克鲁斯卡尔都是，这两个证明涉及诸多引理和铺垫，所以这里就不讨论了，有兴趣参见《算法导论》，大家只管“贪心就好）

具体，普利姆是每次全局取权值最小边（这就是贪心），然后慢慢用这些边将整个图连接起来（一个原则是已经连接起来的部分就不再连接），直观看就像把很多拼图片拼在一起；克鲁斯卡尔则是选了一个“根据点”，每次选根据点能接触到的权值最小边（贪心，注意这条边必须连到还没连接到根据点的结点，也就是不能形成回路），发展它加入根据点（入党= =），满满的，就祖国大地一片红了。。。。。这两个算法思想不算难，但实现起来还是有点麻烦的，建议有兴趣可以自己写写看（据说这一次不允许用伪代码写？），关于两个方法的比较也留意一下

拓扑排序（简单算法见 summary）

8、排序（**MS 这一章中比较重要的几个要认真看一看，你们懂的，**本章内容都是理解内容，所以就不加下划线了。关于排序的算法如果不是要了解详细的性能特点的话，一般也都不复杂，很多就不啰嗦了）

插入系：

直接插入排序（简单的排序，但实现并不一定能很顺畅的写出来，建议可以写一写）

希尔排序（是上面那个的简单升级）

交换系：

冒泡排序（最好看性能真是个悲剧，所以大致了解一下吧）

快速排序（很厉害的排序，结构上，是一个划分算法+递归调用）

选择系：

简单选择

堆排序，堆定义，堆的“筛选”操作（直观是很好理解的，淘汰赛么），推排序的算法（因为堆是完全二叉树，可以用数组表示，所以这个算法实现是非常漂亮的。可以写了试一试）

归并排序：

归并类：

归并排序（同样是 nlogn 家族一员，算法也比较简单，就是分治的一个递归，然后递归返回时进行合并就好了）

Nlogn 是比较类（就是全部上面四类的总和）的极限，要突破这个，就不能依靠比较进行排序，就有了下面的

基数排序

链式基数排序

这两个都了解一下吧，会操作应该就可以，特别的，对于链式基数排序，注意是从低位到高位的，而不是直观上的从高位到低位；也应能理解为什么链式基数排序会要求稳定性

各排序性能综合比较，前面有个基本理解的话，这里水到渠成了