数据结构的章节结构及重点构成：

数据结构学科的章节划分基本上为：概论，线性表，栈和队列，串，多维数组和广义表，树和二叉树，图，查找，内排。

概论：内容很少，概念简单，分数大多只有几分。

线性表：基础章节，必考内容之一。考题多数为基本概念题，名校考题中，鲜有大型算法设计题。如果有，也是与其它章节内容相结合。

栈和队列：基础章节，容易出基本概念题，必考内容之一。而栈常与其它章节配合考查，也常与递归等概念相联系进行考查。

串 ：基础章节，概念较为简单。专门针对于此章的大型算法设计题很少，较常见的是根据 KMP 进行算法分析。

多维数组及广义表 ：基础章节，基于数组的算法题也是常见的，分数比例波动较大，是出题的“可选单元”或“侯补单元”。一般如果要出题，多数不会作为大题出。数组常与“查找，排序”等章节结合来作为大题考查。

树和二叉树 ：重点难点章节，各校必考章节。各校在此章出题的不同之处在于，是否在本章中出一到两道大的算法设计题。通过对多所学校的试卷分析，绝大多数学校在本章都曾有过出大型算法设计题的历史。

图 ：重点难点章节，名校尤爱考。如果作为重点来考，则多出现于分析与设计题型当中，可与树一章共同构成算法设计大题的题型设计。

查找 ：重点难点章节，概念较多，联系较为紧密，容易混淆。出题时可以作为分析型题目给出，在基本概念型题目中也较为常见。算法设计型题中可以数组结合来考查，也可以与树一章结合来考查。

排序 ：与查找一章类似，本章同属于重点难点章节，且概念更多，联系更为紧密，概念之间更容易混淆。在基本概念的考查中，尤爱考各种排序算法的优劣比较此类的题。算法设计大题中，如果作为出题，那么常与数组结合来考查。

第一章　绪论

一.　基本概念和术语

数据结构是一门研究非数值计算的程序设计问题中计算机的操作对象以及它们之间的关系和操作等的学科。

术语：数据、数据元素、数据对象、数据结构、抽象数据类型、算法。

数据结构的形式定义（二元组）

数据的逻辑结构：（四类基本结构）线性结构、树型结构、图状结构、集合

它们各自的特性？

数据的存储结构（物理结构）：主要有顺序存储结构（以存储地址相邻来表示逻辑上的相邻关系），链式存储结构（用指针来指示逻辑上的相邻关系）

抽象数据类型（三元组）

算法的定义，算法的 5 个重要特性。

二.　算法的时间复杂度和空间复杂度

算法的评价：正确性、可读性、健壮性、高效率、低存储量

求语句频度、时间复杂度

第二章　线性表

一.　线性表的定义

线性表的相关基本概念，如：前驱、后继、表长、空表、首元结点，头结点，头指针等概念。

线性表的结构特点，主要是指：除第一及最后一个元素外，每个结点都只有一个前

趋和只有一个后继。

二. 线性表的存储结构

1. 顺序存储结构（顺序表）：随机存取
   插入/删除元素时，需移动元素的次数。
   线性表的顺序存储方式及其在具体语言环境下的两种不同实现：表空间的静态分配和动态分配。静态链表与顺序表的相似及不同之处。
2. 链式存储结构（链表，分为单(向)链表、双(向)链表）：非随机存取（顺序存取）
   单链表：带头结点的链表和不带头结点的链表；
   循环链表；
   链表空与非空的情况（判断条件）。
   插入、删除操作
3. 两种存储结构的优缺点比较，各适合那些场合。

三. 线性表操作的实现（算法描述）

插入元素、删除元素、查找、判表是否满足某种特性、逆置等。

例：

判断题：1. 线性表的逻辑顺序与存储顺序总是一致的。

2. 线性结构的基本特征是：每个结点有且仅有一个直接前驱和一个直接后继。

3. 线性表的链式存储结构优于顺序存储结构。

选择题：线性表 L 在(　　)情况下适于使用链表结构实现。

A. 不需修改 L 的结构　　B. 需不断对 L 进行删除、插入

C. 需经常修改 L 中结点值　　D. L 中含有大量结点

填空题：1. 对于顺序表，在第 i 个元素前插入一个元素需移动_____个元素，要删除第 i 个元素，需移动_____个元素。

2. 在双向循环链表中某结点(由指针 p 指示)之后插入 s 指针所指结点的操作是：

__;_____;_____;_____;。

第三章　栈和队列

4.循环队列中判队空、队满条件，循环队列中入队与出队算法。

如果你已经对上面的几点了如指掌，栈与队列一章可以不看书了。注意，我说的是可以不看书，并不是可以不作题哦。

一. 栈

1. 栈的定义、特点（后进先出）
2. 栈的存储结构：顺序存储结构　链式存储结构
3. 栈的应用：数值表达式的求解，括号的配对等的原理，只作原理性了解。二叉树的先序、中序、后序遍历算法　图的深度优先遍历算法。将递归算法改写为非递归算法可借助栈来完成;。递归算法的执行需用栈来实现。

二. 队列

1. 队列的定义、特点（先进先出）
2. 队列的存储结构：顺序存储结构（**循环队列**），链式存储结构
3. 队列的应用：二叉树层序遍历算法　图的广度优先遍历算法
4. 循环队列：循环队列中判队空、队满条件，循环队列中入队与出队算法、求队

列长度。

例：

判断题：因为栈是特殊的线性表，所以对线性表的所有操作都可以用于对栈操作。

填空题：循环队列队满的条件是_____。

第四章　串

基础章节，概念较为简单，掌握基本概念即可。串、空串、子串、子串位置、串的长度等基本概念。

第五章　数组和广义表

基础章节，概念较为简单，掌握基本概念即可。二者与线性表的关系、数组定义、数组的顺序表示与压缩表示（稀疏矩阵<三元组顺序表>、<十字链表>）；广义表、原子、子表、表头、表尾等定义、广义表头尾链表存储表示。

第六章　树和二叉树

树一章，处处是重点，道道是考题，大家务必个个过关。

一.　树、二叉树的定义

二.　二叉树

1. 二叉树的性质（5 条）考查方法可有：直接考查二叉树的定义，让你说明二叉树与普通双分支树的区别；考查满二叉树和完全二叉树的性质，普通二叉树的五个性质：第 i 层的最多结点数，深度为 k 的二叉树的最多结点数，n0=n2+1 的性质，n 个结点的完全二叉树的深度，顺序存储二叉树时孩子结点与父结点之间的换算关系（左为：2*i，右为：2*i+1）。
2. 二叉树的存储结构：二叉树的顺序存储和二叉链表存储的各自优缺点及适用场合，二叉树的三叉链表表示方法

   顺序存储结构（按完全二叉树层序存放）

   链式存储结构（二叉链表、三叉链表）（空链域的个数）
3. 遍历二叉树：这一知识点掌握的好坏，将直接关系到树一章的算法能否理解，进而关系到树一章的算法设计题能否顺利完成。二叉树的遍历算法有三种：先序，中序和后序。其划分的依据是视其每个算法中对根结点数据的访问顺序而定。不仅要熟练掌握三种遍历的递归算法，理解其执行的实际步骤，并且应该熟练掌握三种遍历的非递归算法。由于二叉树一章的很多算法，可以直接根据三种递归算法改造而来（比如：求叶子个数），所以，掌握了三种遍历的非递归算法后，对付诸如：“利用非递归算法求二叉树叶子个数”这样的题目就下笔如有神了。求叶子个数，求二叉树结点总数，求度为 1 或度为 2 的结点总数，复制二叉树，建立二叉树，交换左右子树，查找值为 n 的某个指定结点，删除值为 n 的某个指定结点，诸如此类等等等等。如果你可以熟练掌握二叉树的递归和非递归遍历算法，那么解决以上问题就是小菜一碟了。

   i. 递归遍历算法

   ii. 非递归遍历算法

   iii. 应用：给定二叉树的先序和中序（或后序和中序）序列，画出相应的二叉树
4. 线索二叉树：线索二叉树的概念和线索化算法以及线索化后的查找算法

线索二叉树的引出，是为避免如二叉树遍历时的递归求解。众所周知，递归虽然形式上比较好理解，但是消耗了大量的内存资源，如果递归层次一多，势必带来资源耗尽的危险，为了避免此类情况，线索二叉树便堂而皇之地出现了。对于线索二叉树，应该掌握：线索化的实质，三种线索化的算法，线索化后二叉树的遍历算法，基本线索二叉树的其它算法问题（如：查找某

一类线索二叉树中指定结点的前驱或后继结点就是一类常考题）

三. 树和森林

二叉树是一种特殊的树，这种特殊不仅仅在于其分支最多为 2 以及其它特征，一个最重要的特殊之处是在于：二叉树是有序的！即：二叉树的左右孩子是不可交换的，如果交换了就成了另外一棵二叉树，这样交换之后的二叉树与原二叉树我们认为是不相同的两棵二叉树。但是，对于普通的双分支树而言，不具有这种性质

1. 树的存储结构：双亲表示法　孩子表示法　孩子-兄弟（二叉树）表示法
2. 树（森林）与二叉树的相互转换
3. 树的遍历：先根、后根次序遍历
4. 森林的遍历：先序、中序遍历

四. 赫夫曼树及其应用

最优二叉树是为了解决特定问题引出的特殊二叉树结构，它的前提是给二叉树的每条边赋予了权值，这样形成的二叉树按权相加之和是最小的。最优二叉树一节，直接考查算法源码的很少，一般是给你一组数据，要求你建立基于这组数据的最优二叉树，并求出其最小权值之和，此类题目不难，属送分题。

1. 构造最优二叉树（赫夫曼树），概念、特点、求 WPL 等
2. 赫夫曼编码

五. 各种二叉树概念的区分

1. 满二叉树（若深度为 k，结点数？）
2. 完全二叉树（若深度为 k，结点数至多、至少？）
3. 最优二叉树
4. （折半查找的）判定树
5. 二叉排序（二叉查找树）树
6. 平衡二叉树
7. 堆

例：

判断题：1. 已知二叉树的先序序列和后序序列并不能唯一地确定这棵二叉树，因为不知道二叉树的根结点是哪一个。

2. 一般二叉树的第 i 层上有 2^{i-1} 个结点，深度为 k 的二叉树有 2^k-1 个结点。

选择题：1. 下列二叉树中，（　）可用于实现符号不等长高效编码。

A. 最优二叉树　　B. 二叉查找树　　C. 平衡二叉树　　D. 二叉排序树

2. 结点数为 n 的二叉树高度至多为（　）。

A. 不定　　B. n　　C. $\lfloor \log_2 n \rfloor +1$　　D. $\log_2 n$

填空题：1. 将满/完全二叉树（含 n 个结点）中各结点从上到下，从左到右进行编号（根结点编号为 1），则编号为 i 的结点，其双亲编号为_____，其左孩子编号为_____，其右孩子编号为_____。

2. 对树的遍历有先根次序遍历树和后根次序遍历树。若以二叉链表作树的存储结构，则树的先根遍历可借用二叉树的_____遍历算法来实现，而树的后根遍历可借用二叉树的_____遍历算法来实现。

应用题：1. 已知某二叉树的先序遍历序列是 ABCDEFGHI，中序遍历序列是 BDCEAGHFI，画出

该二叉树。

2. 以数据集(4,5,6,7,10,12,18)为叶结点权值，构造一棵赫夫曼树，给出每个叶子的赫夫曼编码，并求其带权路径长度。

第七章 图

如果说，从线性结构向树形结构研究的转变，是数据结构学科对数据组织形式研究的一次升华，那么从树形结构的研究转到图形结构的研究，则进一步让我们看到了数据结构对于解决实际问题的重大推动作用。图这一章的特点是：概念繁多，与离散数学中图的概念联系紧密，算法复杂，极易被考到，且容易出大题，尤其是名校，作为考研课程，如果不考查树与图两章的知识，几乎是不可想像的。

一. 图的定义和术语

无向图、有向图、（无向/有向）完全图、入度、出度、子图、（强）连通图、（强）连通分量、生成树

二. 图的存储结构

在考查时，有的学校是给出一种存储形式，要求考生用算法或手写出与给定的结构相对应的该图的另一种存储形式。

无向图（或网）：邻接矩阵、邻接表

有向图（或网）：邻接矩阵、邻接表、逆邻接表

三. 图的遍历（针对具体的存储结构进行）

深度优先搜索遍历图（利用栈） 广度优先搜索遍历图（利用队列）

四. 图的遍历的应用

深度遍历和广度遍历是图的两种基本的遍历算法，这两个算法对图一章的重要性等同于“先序、中序、后序遍历”对于二叉树一章的重要性。在考查时，图一章的算法设计题常常是基于这两种基本的遍历算法而设计的，比如：“求最长的最短路径问题”和“判断两顶点间是否存在长为 K 的简单路径问题”，就分别用到了广度遍历和深度遍历算法。

- 求无向图的连通分量
- 生成树（生成森林）

五. 图的应用

1.生成树、最小生成树的概念以及最小生成树的构造：PRIM 算法和 KRUSKAL 算法。

考查时，一般不要求写出算法源码，而是要求根据这两种最小生成树的算法思想写出其构造过程及最终生成的最小生成树。

2.拓扑排序问题：

拓扑排序有两种方法，一是无前趋的顶点优先算法，二是无后继的顶点优先算法。换句话说，一种是“从前向后”的排序，一种是“从后向前”排。当然，后一种排序出来的结果是“逆拓扑有序”的。

3.关键路径问题：

这个问题是图一章的难点问题。理解关键路径的关键有三个方面：一是何谓关键路径，二是最早时间是什么意思、如何求，三是最晚时间是什么意思、如何求。简单地说，最早时间是通过“从前向后”的方法求的，而最晚时间是通过“从后向前”的方法求解的，并且，要想求最晚时间必须是在所有的最早时间都已经求出来之后才能进行。这个问题拿来直接考算法源码的不多，一般是要求按照书上的算法描述求解的过程和步骤。

在实际设计关键路径的算法时，还应该注意以下这一点：采用邻接表的存储结构，求最早时间和最晚时间要采用不同的处理方法，即：在算法初始时，应该首先将所有顶点的最早时间全部置为 0。关键路径问题是工程进度控制的重要方法，具有很强的实用性。

4.最短路径问题：

与关键路径问题并称为图一章的两只拦路虎。概念理解是比较容易的，关键是算法的理解。最短路径问题分为两种：一是求从某一点出发到其余各点的最短路径；二是求图中每一对顶点之间的最短路径。这个问题也具有非常实用的背景特色，一个典型的应该就是旅游景点及旅游路线的选择问题。解决第一个问题用 DIJSKTRA 算法，解决第二个问题用 FLOYD 算法。注意区分。

重点掌握

1. 求最小生成树（无向网）：Prim 算法、Kruskal 算法
   分别用这两种方法依次得到最小生成树边的过程
2. 拓扑排序（AOV 网）
3. 关键路径（AOE 网）

例：

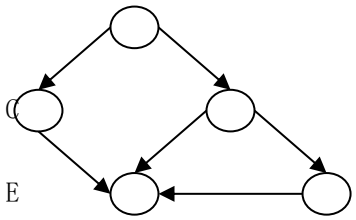
判断题：一个连通图的连通分量是包含该图中的所有（n 个）顶点和任意 n-1 条边的子图。

应用题：已知图 G 如下，画出该有向图的邻接矩阵和邻接表，并根据你的邻接表分别写出深度优先遍历和广度优先遍历的顶点序列。

A

B C

D E



第九章　查找

在不少数据结构的教材中，是把查找与排序放入高级数据结构中的。应该说，查找和排序两章是前面我们所学的知识的综合运用，用到了树、也用到了链表等知识，对这些数据结构某一方面的运用就构成了查找和排序。

在复习这一章的知识时，你需要先弄清楚以下几个概念：

关键字、主关键字、次关键字的含义；静态查找与动态查找的含义及区别；平均查找长度 ASL 的概念及在各种查找算法中的计算方法和计算结果，特别是一些典型结构的 ASL 值，应该记住。

一.　静态查找表

1. 顺序查找表（设置岗哨）
2. 有序查找表（折半/二分查找）　要求：元素值有序的顺序表
   查找算法，折半查找的判定树

二.　动态查找表

这是本章的重点和难点。由于这一节介绍的内容是使用树表进行的查找，所以很容易与树一间的某些概念相混淆。本节内容与树一章的内容有联系，但也有很多不同，应注意规纳。树表主要分为以下几种：二叉排序树，平衡二叉树，B 树，键树。其中，尤以前两种结构为重，也有部分名校偏爱考 B 树的。由于二叉排序树与平衡二叉树是一种特殊的二叉树，所以与二叉树的联系就更为紧密，二叉树一章学好了，这里也就不难了。

1. 二叉排序树（二叉查找树）

二叉排序树，简言之，就是“左小右大”，它的中序遍历结果是一个递增的有序序列。平衡二叉树是二叉排序树的优化，其本质也是一种二叉排序树，只不过，平衡二叉树对左右子树的深度有了限定：深度之差的绝对值不得大于 1。对于二叉排序树，“判断某棵二叉树是否二叉排序树”这一算法经常被考到，可用递归，也可以用非递归。

定义、查找、插入、删除

特征：中序遍历有序

从空树开始，依次插入元素，创建一棵二叉排序树

2. 平衡二叉树

平衡二叉树的建立也是一个常考点，但该知识点归根结底还是关注的平衡二叉树的四种调整算法，所以应该掌握平衡二叉树的四种调整算法，调整的一个参照是：调整前后的中序遍历结果相同。

定义、查找、插入

从空树开始，依次插入元素，创建一棵平衡的二叉排序树

3. n 个元素的二叉排序树、平衡二叉树的（最大、最小）深度

三. 哈希表

哈希一词，是外来词，译自“hash”一词，意为：散列或杂凑的意思。哈希表查找的基本思想是：根据当前待查找数据的特征，以记录关键字为自变量，设计一个 function，该函数对关键字进行转换后，其解释结果为待查的地址。基于哈希表的考查点有：哈希函数的设计，冲突解决方法的选择及冲突处理过程的描述。

1. 哈希表的定义
2. 哈希函数的构造方法
3. 处理冲突的方法
4. 哈希表的造表、查找
5. 装填因子影响哈希表的查找效率

四. 平均查找长度的计算

1. 等概率情况下，查找成功时的平均查找长度 ASL
2. 查找不成功时的查找长度

例：

判断题：1. 任一个二叉排序树的平均查找长度都小于用顺序查找法查找同样结点的线性表的平均查找长度。

2. 当二叉排序树是一棵平衡树时，其平均查找长度为 $O(\log_2 n)$。

选择题：1. 折半查找算法要求被查找的表是(　　)。

A. 键值有序的链表　　B. 键值不一定有序的链表

C. 键值有序的顺序表　　D. 键值不一定有序的顺序表

2. 若查找表是一个有序的单链表，则应采用(　　)方法。

A. 顺序查找　　B. 折半查找　　C. 分块查找　　D. 哈希查找

3. 设线性表中元素的关键字序列为（8,16,24,29,35,40,46,58,66,73,87,98），用折半查找法查关键字等于 87 的元素时，需依次比较关键字（　　）。

A. 40,58,87　　B. 46,87　　C. 40,66,87　　D. 46,66,87

应用题：1. 已知如下所示长度为 8 的表：(12，71，36，45，47，50，2，9)，按表中元素顺序构造一棵平衡的二叉排序树，请画出该树。（二叉排序树）

2. 设哈希表长为 16，哈希函数为 H(key)=key mod 13，用开放定址法的二次探测再散列处理冲突。依次存入 10 个元素：17,24,36,21,83,96,13,34,57,46，请画出它们在散列表中的分布情形，并计算等概率时的 ASL。

3. 对于上题，如果采用链地址法处理冲突，请画出相应的散列表，并计算等概率时的 ASL。

第十章　内部排序

内排是 DS 课程中最后一个重要的章节，建立在此章之上的考题可以有多种类型：填空，选择，判断乃至大型算法题。但是，归结到一点，就是考查你对书本上的各种排序算法及其思想以及其优缺点和性能指标

（时间复杂度）能否了如指掌。本章各种排序算法的思想以及伪代码实现，及其时间复杂度都是必须掌握的，学习时要多注意规纳、总结、对比。此外，对于教材中的 10.7 节，要求必须熟记，在理解的基础上记忆，这一节几乎成为很多学校每年的必考点。

从排序算法的种类来分，本章主要阐述了以下几种排序方法：插入、选择、交换、归并、基数等五种排序方法。

A. **插入排序**中又可分为：直接插入、折半插入、2 路插入、希尔排序。这几种插入排序算法的最根本的不同点，说到底就是根据什么规则寻找新元素的插入点。直接插入是依次寻找，折半插入是折半寻找。希尔排序，是通过控制每次参与排序的数的总范围“由小到大”的增量来实现排序效率提高的目的。

B. 交换排序，又称冒泡排序，在交换排序的基础上改进又可以得到快速排序。快速排序的思想，一语以敝之：用中间数将待排数据组一分为二。快速排序，在处理的“问题规模”这个概念上，与希尔有点相反，快速排序，是先处理一个较大规模，然后逐渐把处理的规模降低，最终达到排序的目的。

C. 选择排序，相对于前面几种排序算法来说，难度大一点。具体来说，它可以分为：简单选择、树选择、堆排。这三种方法的不同点是，根据什么规则选取最小的数。简单选择，是通过简单的数组遍历方案确定最小数；树选择，是通过“锦标赛”类似的思想，让两数相比，不断淘汰较大（小）者，最终选出最小（大）数；而堆排序，是利用堆这种数据结构的性质，通过堆元素的删除、调整等一系列操作将最小数选出放在堆顶。堆排序中的堆建立、堆调整是重要考点。树选择排序，也曾经在一些学校中的大型算法题中出现，请大家注意。

D. 归并排序，故名思义，是通过“归并”这种操作完成排序的目的，既然是归并就必须是两者以上的数据集合才可能实现归并。所以，在归并排序中，关注最多的就是 2 路归并。算法思想比较简单，有一点，要铭记在心：归并排序是稳定排序。

E. 基数排序，是一种很特别的排序方法，也正是由于它的特殊，所以，基数排序就比较适合于一些特别的场合，比如扑克牌排序问题等。基数排序，又分为两种：多关键字的排序（扑克牌排序），链式排序（整数排序）。基数排序的核心思想也是利用“基数空间”这个概念将问题规模规范、变小，并且，在排序的过程中，只要按照基排的思想，是不用进行关键字比较的，这样得出的最终序列就是一个有序序列。

一. 内部排序的方法（排序过程）

1. 直接插入排序
2. 希尔排序
3. 起泡排序
4. 快速排序
5. 简单选择排序
6. 堆排序
7. 归并排序
8. 基数排序

二. 各种内部排序方法的比较

（最好、最坏、平均）时间复杂度　平均空间复杂度

三 . 排序方法的稳定性

哪些是稳定的排序方法，哪些是不稳定的排序方法？

例：

判断题：归并排序和堆排序的平均时间和最坏情况下的时间性能都是 O(nlogn)，因此，它们在最坏情况下的时间性能比快速排序好。

应用题：若要按升序对关键字序列(36, 45, 85, 16, 23, 16, 58, 22, 59, 74, 12, 46)进行排序，请

写出：(1). 堆排序初始建堆(大顶堆)的结果；
(2). 以第一个元素为枢轴的快速排序一趟扫描的结果；
(3). 起泡排序第一趟的结果；
(4). 希尔排序第一趟（增量 5）的结果

一. 判断题

（正确的打“√”，错误的打“×”。）

1. 数据结构可用三元式表示（D，S，P）。其中：D 是数据对象，S 是 D 上的关系集，P 是对 D 的基本操作集。×
2. 线性表的链式存储结构具有可直接存取表中任一元素的优点。 ×
3. 队列是数据对象特定的线性表。×
4. 二叉树是一棵结点的度最大为二的树。 ×
5. 邻接表可以用以表示无向图，也可用以表示有向图。√
6. 可从任意有向图中得到关于所有顶点的拓扑次序。×
7. 一棵无向连通图的生成树是其极大的连通子图。×
8. 二叉排序树的查找长度至多为 $\log_2 n$ 。×
9. 对于一棵 m 阶的 B-树，树中每个结点至多有 m-1 个关键字。除根之外的所有非终端结点至少有$\lceil m/2 \rceil$ -1 个关键字。 √
10. 对于目前所知的排序方法，快速排序具有最好的平均性能。 √
11. 顺序存储方式的优点是存储密度大，且插入、删除运算效率高。×
12. 二维数组是其数据元素为线性表的线性表。√
13. 连通图 G 的生成树是一个包含 G 的所有 n 个顶点和 n-1 条边的子图。×
14. 折半查找不适用于有序链表的查找。√
15. 完全二叉树必定是平衡二叉树。√
16. 中序线索二叉树的优点是便于在中序下查找直接前驱结点和直接后继结点。√
17. 队列是与线性表完全不同的一种数据结构。×
18. 平均查找长度与记录的查找概率有关。√
19. 广义表的表头和表尾都有可能是原子或广义表。×
20. 算法的时间复杂性越好，可读性就越差；反之，算法的可读性越好，则时间复杂性就越差。×

二. 选择题

1. 若广义表 LS 满足 GetHead(LS)==GetTail(LS)，则 LS 为(b)。
   A. ( )　　B. (( ))　　C. (( ),( ))　　D. (( ),( ),( ))

2. 递归程序可借助于(b)转化为非递归程序。
   a:线性表　　b: 栈　　c:队列　　d:数组

3. 在下列数据结构中(c)具有先进先出（FIFO）特性，(b)具有先进后出(FILO)特性。
   a:线性表　　b:栈　　c:队列　　d:广义表

4. 假设用于通讯的电文仅由 6 个字符组成，字母在电文中出现的频率分别为 7，19，22，6，32，14。若为这 6 个字母设计哈夫曼编码（设生成新的二叉树的规则是按给出的次序从左至右的结合，新生成的二叉树总是插入在最右），则频率为 7 的字符编码是（ g ），频率为 32 的字符编码是（ c ）。

a：00　　b：01　　c：10　　d：11

e：011　　f：110　　g：1110　　h:1111

5. 对二叉排序树按（ c ）可得到有序序列。

a:层次遍历　　b:前序遍历　　c:中序遍历　　d:后序遍历

6. 已知某树的先根遍历次序为 abcdefg，后根遍历次序为 cdebgfa。若将该树转换为二叉树，其后序遍历次序为（ d ）。

a：abcdefg　　b：cdebgfa　　c：cdegbfa　　d：edcgfba

7. 对一棵完全二叉树进行层序编号。则编号为 n 的结点若存在右孩子，其编号是(d)。编号为 n 的结点若存在双亲，其编号是(a)。

a：n/2　　b：2n　　c：2n-1　　d：2n+1　　e：n　　f：2(n+1)

8. 关键路径是指在只有一个源点和一个汇点的有向无环网中源点至汇点（ c ）的路径。

a:弧的数目最多　　b:弧的数目最少　　c:权值之和最大　　d:权值之和最小

9. 哈希表的查找效率取决于（ d ）。

a：哈希函数　　b:处理冲突的方法。　　c:哈希表的装填因子。　　d:以上都是

10. 从逻辑上可以把数据结构分成(c)。

a：动态结构和静态结构　　b：顺序组织和链接组织

c：线性结构和非线性结构　　d：基本类型和组合类型

11. 在计算递归函数时，如不用递归过程，应借助于(b)这种数据结构。

a：线性表　　b：栈　　c：队列　　d：双向队列

12. 若已知某二叉树的中序和后序遍历序列分别 BCAEFD 和 CBFEDA，则该二叉树的先序序列为(a)。

a：ABCDEF　　b：ABDCEF　　c：ABDCFE　　d：ACBDFE

13. 当待排序序列的关键字次序为倒序时，若需为之进行正序排序，下列方案中(d)为佳。

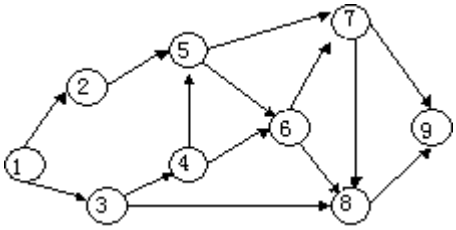
a：起泡排序　　b：快速排序

c：直接插入排序　　d：简单选择排序

14. 若从二叉树的根结点到其它任一结点的路径上所经过的结点序列按其关键字递增有序，则该二叉树是(c)。

a：二叉排序树　　b：赫夫曼树　　c：堆　　d：平衡二叉树

15．下图所有可能的拓扑序列有（ b ）种。

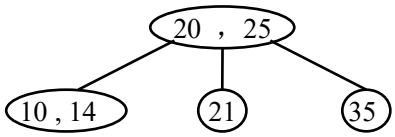


a：2　　b：3　　c：4　　d：5

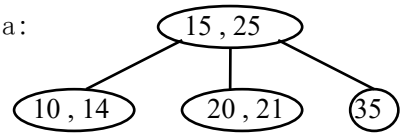
16．下列排序算法中，（ d ）算法可能会出现：初始数据为正序时，花费的时间反而最多。

a：堆排序　　b：起泡排序　　c：归并排序　　d：快速排序

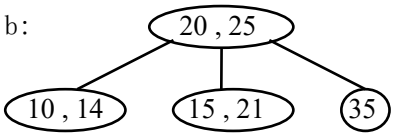
17．右图为一棵 3 阶 B-树。在该树上插入元素 15 后的 B-树是（ c ）。



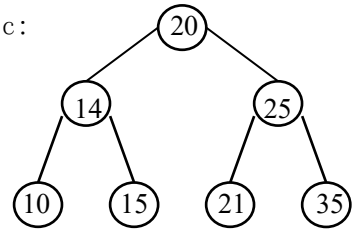
a：



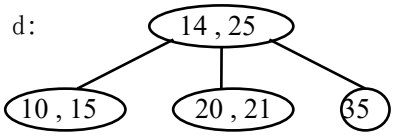
b：



c：



d：

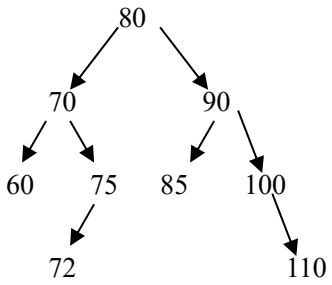


18．设森林 F 中有三棵树，第一、第二和第三棵树的结点个数分别为 m1、m2 和 m3，则与森林 F 对应的二叉树根结点的右子树上的结点个数是（ d ）。

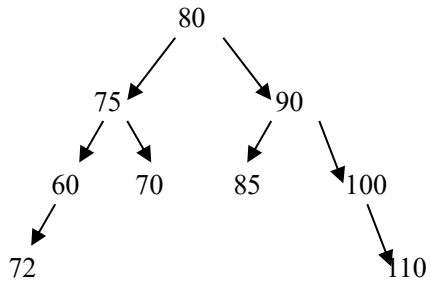
a：m1　　b：m1+m2　　c：m3　　d：m2+m3

19．根据插入次序（80，90，100，110，85，70，75，60，72）建立二叉排序树。图（ a ）是最终变化的结果。若仍以该插入次序建立平衡二叉树。图（ c ）是最终变化的结果。

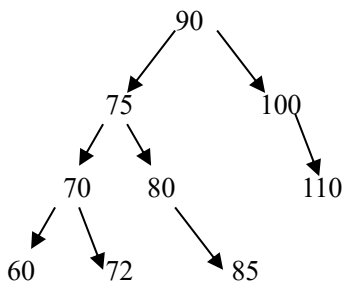
a：



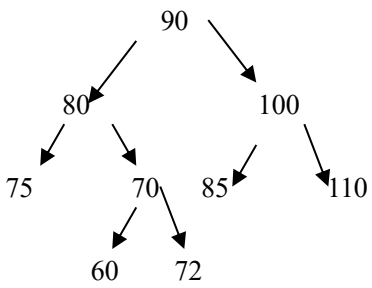
b：



c:

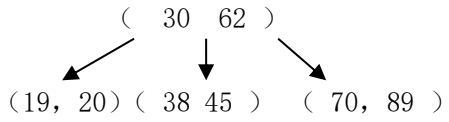


d:

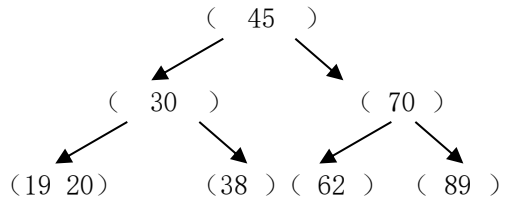


20. 设输入序列为 20, 45, 30, 89, 70, 38, 62，19，依次插入到一棵 2-3 树中(初始状态为空)，该 B-树为（ b ）。再删除 38，该 B-树为（ f ）。

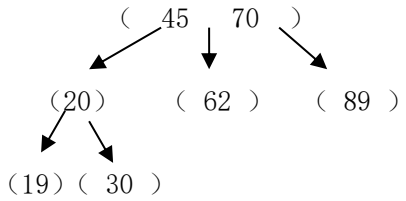
a:



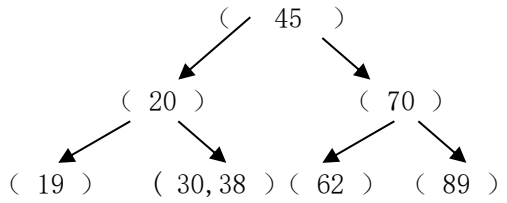
b:



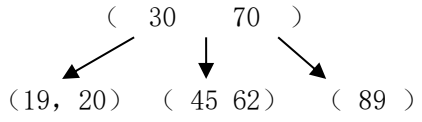
c:



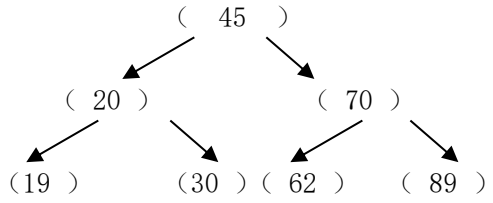
d:



e:



f:



21. 已知一组待排序的记录关键字初始排列如下：45,34,87,25,67,43,11,66,27,78 。

(g)是快速排序法一趟排序的结果；

(a)是希尔排序法（初始步长为 4）一趟排序的结果；

(b)是初始堆（大堆顶）；

(d)是基数排序法一趟排序的结果。

a：27, 34, 11, 25, 45, 43, 87, 66, 67, 78　　b：87, 78, 45, 66, 67, 43, 11, 25, 27, 34

c：11, 43, 34, 25, 45, 66, 27, 67, 87, 78　　d:11, 43, 34, 45, 25, 66, 87, 67, 27, 78

e：34, 45, 25, 67, 43, 11, 66, 27, 78, 87　　f：87, 45, 11, 25, 34, 78, 27, 66, 67, 43

g：27, 34, 11, 25, 43, 45, 67, 66, 87, 78　　h：34, 11, 27, 25, 43, 78, 45, 67, 66, 87

22. 若有序表中关键字序列为：14，20，25，32，34，45，57，69，77，83，92。对其进行

折半查找，则在等概率情况下，查找成功时的平均查找长度是(c)。查找 32 时需进行(c)次比较。

a：1　　b：2　　c：3　　d：4

23. 设一棵二叉树 BT 的存储结构如下：

	1	2	3	4	5	6	7	8
lchild	2	3	0	0	6	0	0	0
data	A	B	C	D	E	F	G	H
rchild	0	5	4	0	8	7	0	0

其中 lchild，rchild 分别为结点的左、右孩子指针域，data 为结点的数据域。则该二叉树的高度为(d)；

第 3 层有(a)个结点（根结点为第 1 层）。

a：2　　b：3　　c：4　　d：5

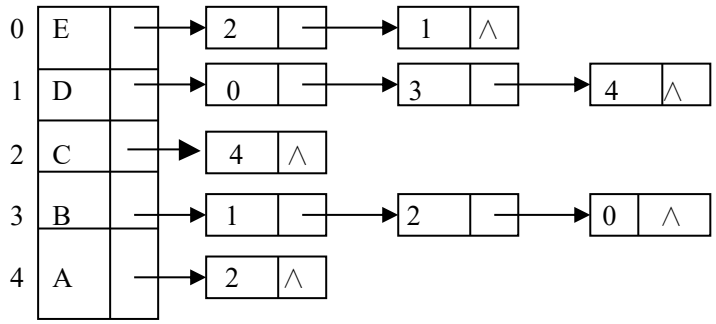
24. 一个连通图的最小生成树(b)。

a：只有一棵　　b：有一棵或多棵　　c：一定有多棵　　d：可能不存在

25.若某二叉树有 20 个叶子结点，有 20 个结点仅有一个孩子，则该二叉树的总结点数是(c)。

a：40　　b：55　　c：59　　d：61

27. 已知某有向图的邻接表存储结构如图所示。



根据存储结构，依教材中的算法其深度优先遍历次序为（ d ）。

广度优先遍历次序为（ c ）。各强连通分量的顶点集为（ f ）。

a: abcde.　　b: edcba.　　c: ecdab.　　d: ecadb.

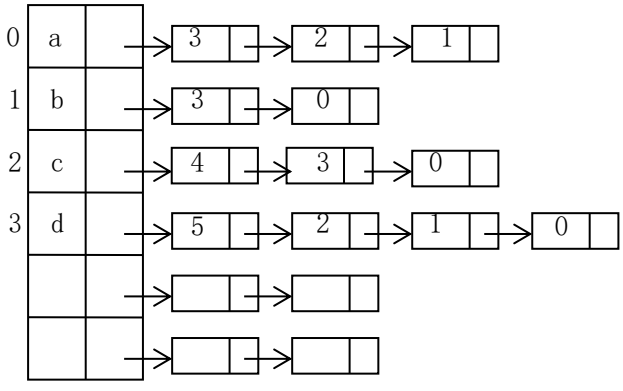
e: abc 及 ed　　f: ac 及 bed　　g: ab 及 ced　　h: bc 及 aed

28. 已知某无向图的邻接表如下所示；

(e)是其原图。

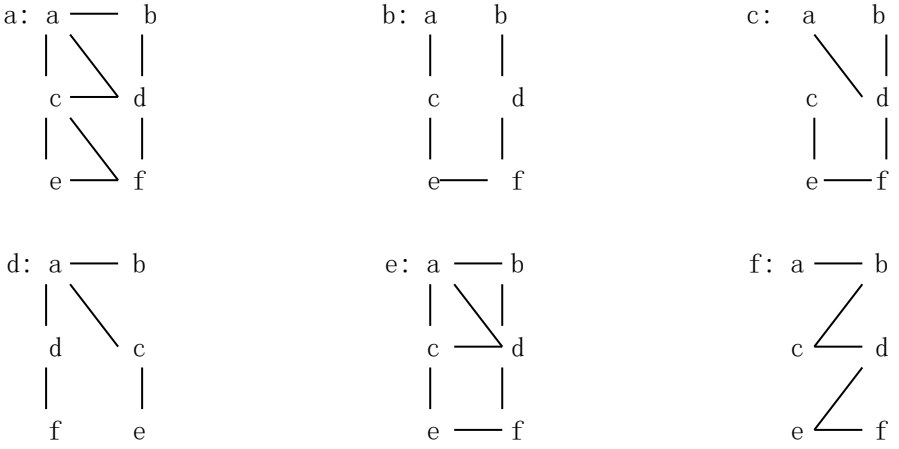
(c)是按该邻接表遍历所得深度优先生成树。

(d)是按该邻接表遍历所得广度优先生成树。



4 e 5 2

5 f 4 3



29. 设有二维数组 $A_{5\times7}$，每一元素用相邻的 4 个字节存储，存储器按字节编址。已知 A 的起始地址为 100。则按行存储时，元素 A_{06} 的存储地址是（ d ）；按列存储时，元素 A_{06} 的存储地址是（ a ）。

a. 220 b. 200 c. 140 d. 124

三．填空题

1. 顺序查找 n 个元素的顺序表，若查找成功，则比较关键字的次数最多为_____次。
2. 一个深度为 4 的满二叉树具有_____个结点。
3. _____遍历二叉排序树可得到一个按关键字的有序序列。
4. 在顺序表（8, 11, 15, 19, 25, 26, 30, 33, 42, 48, 50）中，用二分（折半）法查找关键码值 19，需做的关键码比较次数为_____。
5. 广义表 C=（a,（b,c,d））的深度为_____ 。

四．简答题

1. 将图 1 所示的无向图用链式存储结构“邻接表”存储，画出其邻接表（4 分），分别写出该图的深度优先遍历和广度优先遍历的顶点访问序列（6 分）。

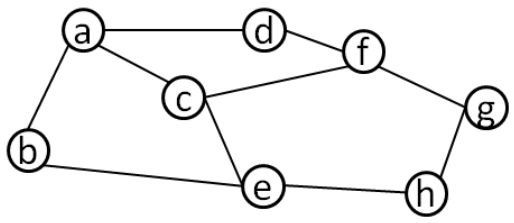


图 1

五．算法设计题

1．单链表结点的类型定义如下：

```
typedef struct LNode {
   int data;
   struct LNode *next;
} LNode, *Linklist;
```

写一算法，Contrary(linklist &L)，对一带头结点且仅设尾指针 L 的循环单链表就地逆置。（即首元变尾元，尾元变首元。）

```
void Contrary(linklist &L)
{ p=L->next; q=L;
 while(p!=L)
   { r=p->next; p->next=q; q=p; p=r; }
p->next=q;
}
```

2．二叉树用二叉链表存储表示。

```
typedef struct BiTNode {
   TelemType    data;
   Struct BiTNode *lchild, *rchild;
} BiTNode, *BiTree;
```

试编写销毁二叉树 T 的算法 DestroyBiTree (BiTree &T)。

```
void DestroyBiTree ( BiTree  &T)
{ if(!T) return;
 if(T-lchild) DestroyBiTree(T->lchild);
 if(T-rchild) DestroyBiTree(T->rchild);
 free(T); T=NULL;
}
```

3．设带头结点的单链表中的元素以值非递减有序排列，试编写算法，删除表中所有值相同的多余元素。

单链表结点的类型定义如下：

```
typedef struct LNode{
   int data;
   Struct LNode   *next;
}LNode, *LinkList;
```

```
void Delete_same(LinkList L)
{ if(!L->next) return;
 p=L->next; q=p->next;
 while(q)
  if(p->data==q->data)
    { p->next=q->next; free(q); q=p->next;}
  else
    { p=q; q=q->next;}
```

}

1. 已知两个无序单链表，均为有链表头结点的链表，现需将这两个无序单链表进行排序变为有序链表，然后再将这两个链表合并为一个链表。请按照以下提示和要求给出算法。

已知链表存储结构为：

```
typedef struct Node{
        int data;
        struct Node *next;
       }Linknode,*Link;
```

（1）对单链表中元素按插入方法排序的 C 语言描述算法如下，其中 L 为链表头结点指针。请填充算法中标出的空白处，完成其功能。

```
void Insertsort(Link &L)
    { Link p,q,r,u;
      p=L->next;
      L->next=NULL;
      while(p!=NULL)
        { r=L;   q=L->next;
          while(①_____&& q->data<=p->data) {r=q;   q=q->next;}
          u=p->next;
         ②_____;
         ③_____;
          p=u;
         }
}
```

①

②

③

（2）请给出有序单链表的合并算法及其算法的时间复杂度

```
void   MergeList_L (Link &La, Link &Lb, Link &Lc)
```