数据结构概念和原理复习

内容纯手打，如有出错还请谅解

第一章

1.数据结构：给定的数据对象及其上面定义的操作共同构成的整体叫做数据结构。

2.数据结构研究哪几个方面的内容？：①逻辑关系 ②存储关系 ③运算关系

3.常见的逻辑关系：①单对单（线性关系） ②单对多（树形关系） ③多对多（图关系）

4.常见的存储（物理）关系：①顺序 ②链接 ③散列 ④索引

5.7数据结构的优劣主要从哪几个方面进行评价？：①时间复杂度 ②空间复杂度

第二章

1.算法：算法是解决某个问题的指令的集合

2.算法有哪五个要求？：①有穷性 ②确定性 ③可行性 ④输入 ⑤输出

3.算法的描述方法：①具体的计算机程序设计语言 ②自然语言 ③PDL语言 ④流程图

4.算法评价的四个标准：①确定性 ②易读性 ③健壮性 ④高效性

5.有效算法：以多项式时间为界线的算法

第三章

1.线性表：一个线性表是n≥0个数据元素a1，a2…an的有限序列，序列中除第一个和最后一个外，每个元素有且仅有一个直接前驱和直接后继。

2.链表：通过指针联系起来的结点的整体（集合）。

3.静态链表：以整形变量的值作为存储指针值（即地址）联系起来的结点的整体，但空间已经不能再增加的链表是静态链表。

4.表头结点：增加一个附加结点，放置于链表的最前面，也称头结点。

5.存储密度：（数据本身所占存储量）/（整个结构所占存储量）

6.链表插入，删除：算法自己写。

第四章

1.栈：栈是一个下限为常数，上限可变化的向量（或反之）。

2.队列：队列是一个上限和下限都只能增加，不能减少的向量（或反之）。

3.栈，队列与线性表的异同：

同：①数据间的逻辑关系相同。②存储结构相同。

异：

栈：①增减只能在一端变化。②不能进行线性表的操作中除插入和删除以外的操作。

队列：①只能在一端插入，另一端删除。

②不能进行线性表的操作中除插入和删除以外的操作。

4.假溢出：系统给的空间有余，但由于管理或组织不当产生的无法压入的现象称为假溢出。

5.循环队列的组织：算法自己写。

第五章

1.串：由零个或多个字符组成的有穷序列称为串。

2.串的模式匹配：给定一个子串，求出主串中所有与该子串相同的子串的运算叫做串的模式匹配。

第六章

1.数组：一维数组是个向量，它的每个元素是该结构中不可分割的最小单位，n维数组是个向量，它的每个元素是n-1维数组，且具有相同的下限和上限。

2.稀疏数组：在一个数组中和某元素比较而言，不相同元素的个数很少时，我们称此数组为稀疏数组。

3.稀疏矩阵：在一个矩阵中和某元素比较而言，不相同元素的个数很少时，我们称此矩阵为稀疏矩阵。

4.广义表：是零个或多个原子或子表所组成的有限序列。一般简称为表。

5.稀疏矩阵的存储方法：

①三元组：将矩阵的全部非零元素抽象为一个线性表，每个元素含行，列，值三个数据项。

②十字链表：对于非零元，一句行，列特性，每一行和每一列建立一个循环链表。航标头的向下指针无效，列表头的向右指针无效。在行列为n\*n的时候可以将行表头和列表头共用。

第七章

1.树：树是满足如下性质的有限个结点组成的非空集合：

①T中有且仅有一个称为根的结点。

②除根结点以外，其余结点分成m（m>0）个不相交的集合T1，T2,…Tm，其中每个Ti都是树，而且都称为T的子树。

2.二叉树：T是满足一下性质的结点的有限集合：

①T为空集。

②或者T包含一个根结点，且其余结点分成两个不相交的集合，并分别被称为根结点的左子树和右子树。

3.遍历：对于给定的数据结构，系统地访问该结构中的每个结点，且每个结点仅被访问一次的操作过程叫做遍历。

4.二叉树的遍历规则：①先序遍历②中序遍历③后序遍历。

5.二叉排序树：二叉排序树或是空二叉树，或者满足如下性质：

①若它的左子树非空，则左子树上所有结点的值均小于根结点的值。

②若它的右子树非空，则右子树上所有结点的值均大于根结点的值。

6.线索：将二叉树的空指针利用起来，用于表示某线性关系下的直接前驱或直接后继时，这种指针叫做线索。

7.线索化：给二叉树加线索的过程叫线索化。

8.线索树：带线索的二叉树叫做线索二叉树，简称线索树。

9.霍夫曼树（哈夫曼树）：按Huffman算法构造的具有最小加权路径长度的二叉树称Huffman树（或最优二叉树）。

Huffman算法的基本思想：

1. 给定一组权值集合{w1,w2…wn},据此构成n棵二叉树组成的森林F。注：F中的每棵二叉树只有一个带权值为Wi（1≤i≤n）的根结点。
2. 将F={T1，T2…Tn}按根结点的值由小到大进行排序。
3. 取出T1和T2组成一棵二叉树T，再将T插入到F中，并使F依据根结点的值有序。
4. 反复执行③直到F={T}为止。

10.树，森铃与二叉树的相互转换：

①树转为二叉树：

a.加线，将所有兄弟连接起来。

b.抹线，将所有双亲结点与除第一个左孩子以外的所有孩子的关系抹掉。

c.调整，将原来左孩子的兄弟作为它的右孩子。

②二叉树转为树：

1. 加线，若某结点x是双亲结点的左孩子，则将该结点的右孩子以及当且仅当连续地沿着此右孩子的右链不断搜索到的所有右孩子都与x的双亲结点连接起来。
2. 抹线，抹掉所有结点与所有右孩子的关系。
3. 调整，原来沿着x的右链搜索到的所有结点变成x的兄弟。

③森林转为二叉树：依据森林中树的次序，依次将转换得来的二叉树，后一棵作为前一棵二叉树的根结点的右子树。

1. 二叉树还原为森林：
2. 抹线，沿着二叉树的根结点的右孩子的右链抹掉所有右孩子关系。
3. 还原，将分出来的二叉树还原为树。

11.二叉树的前中后遍历方法：

①先序：a.访问根结点。b.遍历左子树。c.遍历右子树

②中序：a.遍历左子树。b.访问根结点。c.遍历右子树

③后序：a.遍历左子树。b.遍历右子树。c.访问根结点

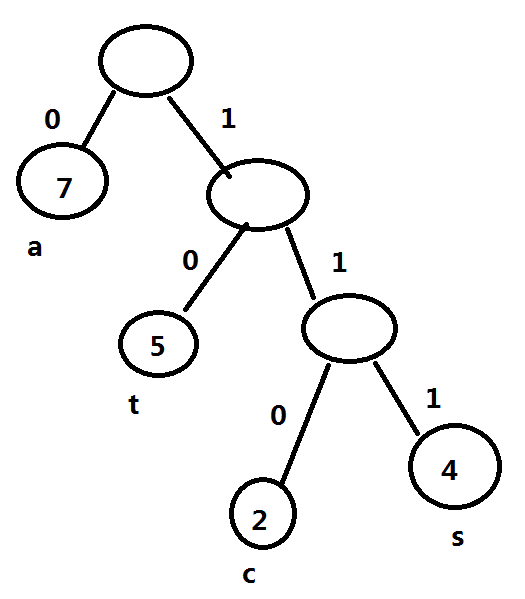
12.加线索：通过加标志位区分线索和孩子。先输出遍历结果再根据结果加线索。

特别介绍中序线索树中寻找前驱后继的方法：

直接后继：沿着其右子树的左链找到的第一个没有左孩子的结点为其直接后继。

直接前驱：沿着其左子树的右链找到的第一个没有右孩子的结点为其直接前驱。

13.霍夫曼书的构造及编码：构造上面已经讲了。

编码：比如有以下正文cast，cats，sat，at，a，tasa符号集D=（c,a,s,t），对应的每个字符出现次数为WD=（2,7,4,5），依据WD构造Huffman树，如图

可以得出编码：a：0，t：10，c：110，s：111

14.二叉排序树的构造：将S插入二叉排序树T中，若T为空则S直接作为T的根；若不为空则与T进行比较，如果S小于T则进入T的左子树，否则进入右子树。

15.二叉树遍历方法的运用：交换左右子树，求二叉树高度，统计二叉树叶子结点的个数等等，就不详细写了。

第八章

1.图：由n（n≥1）个结点v1，v2…vn构成的数据G称为图，若结点集V={v1，v2…vn}上定义的称为后继的关系E是非自反的。可以表示为G={V，E}V称为顶点集，E称为边集。

2.图的遍历：给出图G和其中的任意一个顶点v0，从v0出发系统地访问G中所有的顶点，且每个顶点仅被访问一次，这一过程称为图的遍历，或遍历图。需要辅助数组记录该结点是否被访问过。

3.图的遍历策略：

①深度优先策略（DFS）：搜索中，结点扩展的次序向某一个分支纵深推进，到底后回溯。

②广度优先策略（BFS）：搜索中，对所在层次的所有结点逐个依次进行扩展后，再推进到下一个层次进行扩展。

4.最小生成树：设V={1,2…n}是图G的顶点集合，由一个初值{v0}的集合开始，它每次生成一条边，逐渐长成一个具有最小代价的数，叫做最小生成树。

5.AOV-网：若在有向图G中，顶点表示活动或任务，有向边表示活动或任务之间的有限关系，则此有向图称为顶点表示活动的网络（AOV-网）。

6.AOE-网：若在带权有向图中顶点表示事件，有向边表示活动，权表示活动持续的时间，则次有向图称为边表示活动的网络（AOE-网）。

7.关键路径：任务计划作业图上的需要时间最长的路径（可有多条）。它决定完成总任务的时间。

8.拓扑排序：

①从图中选择一个入度为零的顶点。

②输出该结点，从图中删除此顶点及其所有的出边。

③反复执行以上两步，知道所有顶点都输出，此时拓扑排序完成。

④或者直到剩下的图中再无入度为零的顶点，此时说明图G是有环的图，拓扑排序无法完成。

9.图的存储方法：

①邻接矩阵：矩阵中的1表示该行代表的结点能到达列代表的结点

②领接表：领接表有三个属性 入度域indegree，顶点vertex，指针link。Indegree用于表示入度，在输入边建立邻接表的过程中产生。vertex记录当前顶点的下标，link是一组链表，表示与顶点有邻接关系的结点。

10.深度广度的遍历方法：

深度策略：

①访问触发的顶点v0。

②选择一个v0邻接到的未被访问过的结点u。

③再从u开始进行深度搜索

如果从任一顶点触发再也无法到达一个未曾访问过的顶点，则本次搜索就算结束。

回退策略：每当到达一个其所有相邻结点均已被访问过的顶点时，就回退到最后访问过的那个顶点—该顶点有一个与其邻接的未曾访问过的顶点y，并从y再开始进行深度遍历。

广度策略：

1. 问出发顶点v0。

②访问v0邻接到的所有未被访问的结点。

③如此进行下去，直到无法找到未被访问的结点时，则本次搜索就算结束。

11.Prim算法：设V={1,2…n}是图G的顶点集合，Prim算法由一个初值为{v0}的集合开始，它每次生成一条边，逐渐长成一个最小代价的生成树。

12.Kruskal算法：对于图G={V,E}该算法初始化从T={V,**∅**}开始此时生成树是由n个连通分量（即n个孤立顶点）组成的一个子图—目标使得n个连通分量逐渐成为一个连通分量，具体做法：

按照权值递增的顺序逐个考虑E中的每条边：

1. 该变联通了两个不同连通分量中的顶点，则将改变添加到T中。
2. 复①，一旦T中包含了n-1条边，则终止运算。

第九章

1.排序：设含有n个记录的集合为R={r1,r2…rn}，其对应的关键字集合为K={k1,k2…kn}，给定关系α，按照关系α针对关键字集合K对R进行运算，使得R有如下序列：{**rα1, rα2 ,…, rαn**},我们将这个操作过程称为排序。

2.排序的分类：内部排序和外部排序。

3.排序的稳定性在排序关系下，假设排序前ri在rj之前，排序之后领先关系不变，则称此排序过程和排序方法是稳定的，否则是不稳定的。

4.堆：设L是长度为m的表，当1≤i≤ ⎣n/2⎦时，其数据元素满足：L（i）≤L（2

i）且L（i）≤L（2i+1），则称L是一个堆。其中将L（1）称为堆顶，L（n）称为堆底。

5.一趟排序：

①直接插入：将要排序的源文件F=R1,R2...Rn视为两部分 F’=R1,F”=R2...Rn。对F’和F”重复如下工作：

a.从F”中取出一个记录Ri,并将Ri从F”中删除。

b.将Ri插入到F’并使F’线性有序。

②快速排序：在待排序的n个记录中抽取一个记录r作为轴心元素，以r为标准将所有记录分为两组，第一组中各记录的关键字都小于r的关键字，第二组中个记录的关键字都大于r的关键字，并把r排在这两组中间。

1. 二路归并：将两个子文件进行合并，得到n/2个部分有序的较大子文件，每个子文件含有两个记录，再将这些子文件合并，如此反复，直到最后合并成一个文件时，排序就完成了。
2. 基数排序：从个位开始放到0至9的盒子中，直到排完最高位结束。

6.堆的构造（插入、删除）：从叶节点开始，管不住就往上跑那个，算法自己写。

7.二分插入排序：小的往前丢，大的往后丢那个，算法自己写。

第十章

1.检索：在给定数据结构中查找满足条件的数据元素的过程。

2.检索的分类：

①按性质分：关键字，属性。

②按数据组织方式分：线性，树形，散列。

3.平均检索长度：在检索过程中对关键字或属性要执行的平均运算次数。

4.AVL-树（平衡二叉树）：①一棵空树是AVL-树。②若T是一棵非空二叉树，其任何结点的左右子树的高度差不超过1，则T是AVL-树。

5.散列表：把用散列法组织存储的线性表成为散列表。

散列法基本思想：将关键字看成一个变量，通过一定的函数关系，将函数值解释为存储地址，将结点存入这样计算得到的地址单元中，检索过程是存储过程的逆过程。

6.碰撞：依据散列函数H计算出地址，若发现此地址已经被别的结点占用，即是说有两个不同的关键字映射到了同一个地址空间，我们把这种现象称为碰撞（或冲突）。

7.同义词：产生碰撞的两个（或多个）关键字称为同义词（相对于H而言）。

8.堆积：吧两个同义词子表结合在一起的现象称为“堆积现象”或“群集现象”。

9.AVL-树的构造：选择离插入或删除结点最近的不平衡点（平衡因子为±2）开始调整.

根据中序遍历结果调整即可，LL型，LR型，RL型，RR型，自己看，不好写。

10.二分检索：（类似折半查找）先取表的中间位置的记录关键字值与所给关键字值进行比较，如果给定值比该记录的关键字值大，则给定值必在表的后半部分；再继续折半，直到找到或找不到为止。

第十一章

无

第十二章

1. 文件：为了进行存取控制、检索和修改而组织在一起的数据记录的组合。
2. 文件的逻辑组织分类：字符流文件和记录文件。
3. 文件的物理组织（存储）分类：顺序文件，散列文件，索引文件。
4. 静态索引：静态索引结构是指索引结构在文件创建，初始装入记录时生成，一单生成就固定下来，在系统运行（如插入、删除）中索引结构并不发生变化，只有当文件重组时才允许改变索引结构。
5. 动态索引：动态索引结构是指文件创建，初始装入记录时生成的索引结构，在系统运行过程中索引结构本身也能发生改变。
6. B-,B+树的作用及本质：

①B-,B+树的本质是一种平衡的多分树。

②作用：由于对B树的查找、插入、删除均能够保持B树的动态平衡，因此B-树及其他一些改进形式（例如B+树）已经成为组织索引文件的一种标准的有效结构，已得到了广泛的应用。

1. ISAM的本质：ISAM（索引顺序存取法）是现今磁盘存取文件的常用组织方法。
2. VSAM文件的本质：VSAM（虚拟存储存取法）是B+树应用的一个典型例子。其本质是动态索引结构。
3. 外排序的基本方法：

合并排序：

1. 生成初始顺串，先用某种有效的内排序方法对文件的各段进行排序，这种经过排序的段通常称为顺串（或归并串），当它们生成以后就写到外存上，这样外存上就形成了许多初始归并段。
2. 合并初始顺串，k-路归并，于2-路归并类似。