**1、最小生成树**

加权连通图里搜索最小生成树，连通所有的顶点，并且权值和最小。

有可能树种某条边的值大于未选中的边的值。

权值最小的是否出现：不一定

不同顶点构造的最小生成树是否唯一：不一定

同一顶点构造的最小生成树是否唯一：有权相等则不唯一。

不同方法构造出来的最小生成树是否唯一：不一定。

**2、n个结点的完全二叉树的深度k**

9ece43d8634217478ecf03dd42b335db.png

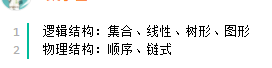
完全二叉树的叶子节点有一个或没有。

**3、n个结点的线索二叉树上含有的线索数为**

n+1：一棵n结点树包含n-1条边,而每个结点有两个指针域即总共2n个指针,减去表示边的指向关系（即左右子树）的n-1条边,剩下n+1条边即为线索.

**4、二叉树 叶子结点为n，则度为2的结点的个数为n-1。二叉树的度为分支的个数。**

**5、**



**7、森林F转换为对应的二叉树T，F中叶结点的个数等于T中左孩子指针为空的结点个数**

**8、宽度优先搜索BFS，深度优先搜索DFS**

深度优先搜索一边无法访问到图中的所有结点

**9、堆**

大顶堆：父节点大于子节点

小顶堆：父节点小于子节点

建堆：从第一个非叶子节点开始调整，时间复杂度为o(n)

2.堆排序：有序区与无序区，将根节点与无序区的最后一个元素进行交换，然后调整堆。

堆排序适用于找出最大的最小的前几个。

**10、排序时间复杂度空间复杂度稳定性辅助空间**



**11、一个整数减去1再和原整数做与运算，就会把原整数最右边的1变为0；**

**12、二叉搜索树、二叉排序树：**左节点小于根节点，右子树大于根节点。

**13、树的路径是从根节点到叶子节点；**

**14、哈弗曼树**

带权路径长度WPL最小的二叉树称为哈弗曼数，哈弗曼编码。

不一定是平衡二叉树

**15、最小生成树**

构造连通网的最小代价生成树

Prim算法：P250

Kruskal算法：P251

**16、散列函数的构造方法**

直接定址法

数字分析法：适合知道关键字分布，位数较大

平方取中法：适合不知道关键字分布，位数较小

折叠法：适合不知道关键字分布，位数较大

除留余数法

**解决冲突的方法**

开方地址法：寻找下一个地址

链地址法

公共溢出区法

**装填因子**

填入表中记录个数/散列表长度

**17、线索二叉树**

左孩子为空，指向后序遍历的前驱

右海子为空，指向后序遍历的后继

**18、AVL平衡二叉树**

查找、插入和删除在平均和最坏情况下都是O（log n）

**19、排序的各个特点**

对原始数据排列的敏感度：堆排序没有。如果整体有序，冒泡和直接插入排序更好O(n)

最坏的情况：堆排序和归并排序要好O(nlogn)

稳定性：归并排序，冒泡，选择，插入

空间复杂度：归并最差

**20、各种二叉树**

二叉排序树（二插查找树）：左子树（全部值）小于根，右子树大于根。中序遍历为递增序列。

线索二叉树：加上线索的二叉树

完全二叉树：满二叉树的特例

满二叉树：所有节点都存在左右子树

平衡二叉树：0、1