1、树

（1）定义

1°有一个根结点

2°其余结点可分为m个互不相交的集合，这些集合本身也是树

（2）父亲称为儿子的直接前驱，儿子称为父亲的直接后继

（3）基本术语

1°根结点、叶结点、内部结点

2°度：一个结点直接后继的数目

所有结点的度的最大值称为这棵树的度

3°子结点、父结点、祖先结点、子孙结点

4°兄弟结点

5°深度：从根结点到这个结点所经过的边数

高度：一棵树中结点的最大层次称为树的高度

结点的高度指的是以该结点为根的子树的高度

6°有序树、无序树：树中每个结点的子树从左向右是否有序

7°森林：M棵互不相交的树的集合

2、树的基本运算

（1）create

（2）clear

（3）isEmpty

（4）root：找根结点

（5）parent(x)：找出结点x的父结点

（6）child(x, i)：找出结点x的第i个子结点

（7）delete(x,i)：删除结点x的第i棵子树

（8）MakeTree(x, T1, ……, Tn)：构建一棵以x为根结点，以T1，……Tn为第i棵子树的树

（9）traverse：遍历

3、二叉树的基本概念

（1）定义：或者为空，或者由一个结点及两棵互不相交的左、右子树构成，而其左右子树又都是二叉树（子树可以是空树）

（2）二叉树是有序树

（3）二叉树的5种基本形态：

（《数据结构：思想与实现》108页）

（4）满二叉树（丰满树）：任意一层的结点个数都达到了最大值

（5）完全二叉树：满二叉树的最底层自右向左依次（不能跳过任何一个结点！！）去掉若干个结点（满二叉树是完全二叉树，完全二叉树不一定是满二叉树）

4、二叉树的主要性质

（1）一棵非空二叉树的第i层最多有2^(i-1)个结点（i≥1）

（2）一棵高度为k的二叉树，最多有2k – 1个结点

（3）对于一棵非空二叉树，如果叶结点数为n0，度为2的结点数为n2，则：n0 = n2 + 1

（4）具有n个结点的完全二叉树的高度k = [log2n] + 1

（5）如果对一棵有n个结点的完全二叉树的结点按层自上而下、每一层自左至右一次编号。若设根结点的编号为1，则对任意编号为i的结点

1°若i = 1，则该结点是二叉树的根结点；如果i > 1，则其父结点的编号为[i/2]；

2°如果2i > n，则编号为i的结点为叶结点；否则，其左儿子的编号为2i；

3°如果2i+1>n，则编号为i的结点无右儿子；否则，其右儿子的编号为2i+1。

5、二叉树的遍历

（1）前序遍历（先根遍历）

1°递归定义：

若二叉树为空，则操作为空；

否则，访问根结点 —> 前序遍历左子树 —> 前序遍历右子树

（2）中序遍历（中根遍历）

1°递归定义：

若二叉树为空，则操作为空；

否则，中序遍历左子树 —> 访问根结点 —> 中序遍历右子树

2°中序遍历中，左子树的结点先于根结点被访问，右子树的结点后于根结点被访问。所以，二叉树中序遍历相当于对一棵二叉树从上到下做一个投影。

（《数据结构：思想与实现》113页）

（3）后序遍历（后根遍历）

1°递归定义：

若二叉树为空，则操作为空；

否则，后序遍历左子树 —> 后序遍历右子树 —>访问根结点

（4）

1°已知二叉树的前序序列和中序序列，可以唯一确定一棵二叉树

2°已知二叉树的后序序列和中序序列，可以唯一确定一棵二叉树

3°已知二叉树的前序序列和后序序列，不可以唯一确定一棵二叉树

（5）层次遍历

按层访问结点

6、二叉树的顺序实现

（1）对于完全二叉树，可以通过编号作为存储二叉树的数组的下标。

根据性质5，根结点放在[1]中；存储在下标变量k的结点的两个儿子分别存储在下标变量2k和2k+1中；它的父结点存储在下标变量[k/2]中。

（2）对于非完全二叉树，性质5不可用。因此可以添加“虚结点”使之成为一棵完全二叉树。存储时，用一个特殊值表示这些虚结点。

注：这种方法会造成存储空间的浪费，因此只适用于静态且结点个数已知的情况或接近完全二叉树的情况。

7、二叉树的链接实现（主要存储方式）

（1）标准存储结构（二叉链表）

1°二叉链表中，每个存储结点由3个字段组成：存储数据元素值的数据字段、指向左儿子的指针、指向右儿子的指针

2°存储一棵二叉树只需要一个指向根结点的指针root，一般称为根指针

（2）广义的标准存储结构（三叉链表）：为了解决标准存储结构找父亲操作复杂的问题

1°三叉链表中，每个存储结点由4个字段组成：存储数据元素值的数据字段、指向左儿子的指针、指向父结点的指针、指向右儿子的指针

注：大多数情况下用二叉链表，需要查找父结点时用三叉链表

8、二叉树运算的实现

（《数据结构：思想与实现》119页）

9、二叉树遍历的非递归实现

（1）前序遍历：栈（《数据结构：思想与实现》127页）

（2）中序遍历：栈（根结点要进栈两次）

（3）后序遍历：栈（根结点要进栈三次）

10、二叉树的应用：计算表达式

11、哈夫曼编码：

（1）字符编码可以有不同的长度，只要每个字符的编码和其他字符编码的前缀不同即可。

（2）让出现频率高的字符尽量靠近树根，这样编码就会更短

12、哈夫曼算法

例：

|  |  |
| --- | --- |
| 字符 | 出现次数 |
| a | 10 |
| e | 15 |
| i | 12 |
| s | 3 |
| t | 4 |
| 空格 | 13 |
| 换行 | 1 |

1°找到出现次数最小的两个字符合并为1棵树：

换行—(合体字符1：1+3 = 4)—s

2、合体后的两个字符出现次数为4，参与下次的筛选，继续上一次的操作，变成了：

t—(合体字符2：4+4=8)—合体字符1

3、重复以上操作，直到归并完全结束

13、树的存储实现

（1）儿子链表表示法

1°树的每个结点包括：数据值+指向儿子链的指针

2°每个父亲的所有儿子组成一个链表

3°树的所有结点存放在一个数组中 —> 静态的儿子链表

或 树的所有结点存放在一个链表中 —> 动态的儿子链表

（2）儿子兄弟链表表示法

和二叉树表示形式一样，但是左指针指向第一个儿子，右指针指向兄弟

（3）双亲表示法

1°树的每个结点包括：数据值+存储父结点位置的父指针字段

2°树的所有结点存放在一个数组中 —> 静态的双亲表示法

或 树的所有结点存放在一个链表中 —> 动态的双亲表示法

14、树的遍历

（1）前序遍历

（2）后序遍历

（3）层次遍历