ADL

第一题

算法 leafNode(binarytree,node)

IF node= NULL

THEN RETURN 0.

ELSE IF node.left=NULL AND node.right = NULL

THEN RETURN 1.

ELSE RETURN(leafNode(node.left)+leafNode(node.right)).

第二题

算法 DFS(graph,G)

D1.[初始化]

count←0.

visited[i]←1.

G.matrix←NULL.

G←AVAIL.

G.numVertexes←numVertexes.

D2.[递归]

count←count-1.

FOR j←0 TO G.numVertexes DO

IF G.matrix[i][j]= 1 AND visited[j] != 1 THEN DFS(G,j).

D3.[判断]

IF count = G.numVertexes THEN RETURN 1.

ELSE RETURN 0.

证明

第一题

证明:给出n个权值{w1，w2，...，wn}构造一棵具有n个叶子结点的哈夫曼树的方法如下：第一步，构造n个只有根结点的二叉树集合F={ T1，T2，...，Tn}，其中每棵二叉树Ti的根结点带权为 Wi (1≤k≤n)；第二步，在集合F中选取两棵根结点的权值最小的二叉树作为左右子树，构造一棵新的二叉树，令新二叉树根结点的权值为其左、右子树上根结点的权值之和；第三步，在F中删除这两棵二叉树，同时将新得到的二叉树加入到F中；第四步，重复第二步和第三步，直到F只含有一棵二叉树为止，这棵二叉树便是哈夫曼树。综上所述，我们可以知道哈夫曼树中权值最小的两个结点互为兄弟结点，且至少与其他叶结点同一层。

第二题

证明:由于深度优先遍历先输出当前结点，在根据一定的次序去递归查找孩子。图中每个顶点仅访问一次，且从一个顶点到另一个顶点时必须经过这两个顶点所连接的边。当深度优先遍历将连通图全部顶点访问一次后，共通过了其中n-1条边，，而这n-1条边恰好把图的n个顶点全部连通，图的n个顶点和n-1条边构成了图的一个连通子图。具有n个顶点和n-1条边的连通图一定是一棵树。