### 数据结构中缀表达式转换为前缀与后缀表达式：

这里我给出一个中缀表达式：a+b\*c-(d+e)

第一步：按照运算符的优先级对所有的运算单位加括号：式子变成了：((a+(b\*c))-(d+e))

第二步：转换前缀与后缀表达式

前缀：把运算符号移动到对应的括号前面

则变成了：-( +(a \*(bc)) +(de))

把括号去掉：-+a\*bc+de 前缀式子出现

后缀：把运算符号移动到对应的括号后面

则变成了：((a(bc)\* )+ (de)+ )-

把括号去掉：abc\*+de+- 后缀式子出现

**循环队列中，设front为队头指针，rear为队尾指针，存储空间大小为n; 则队列满的条件是（rear+1）%n==front; 队列空的条件是front==rear;**

**队头元素出队列，front=（front+1）%n ; 元素从队尾入队列，rear=(rear+1)%n;**

**队列长度：(rear-front+QueueSize)%QueueSize**

树中结点数等于所有结点度数的和加1

**关于哈夫曼树的构造：  
首先，对这些已知的数进行排序---从小到大  
然后，在这些数中找到两个最小的数字（哈夫曼树是从下往上排列的）  
然后，用一个类似于树杈的“树枝”连接上两个最小的数。在顶点处计算出这两个数字的和并写在上面。然后再比较剩下的数字和这个和的大小，再取出两个最小的数字进行排列  
然后，如果两个数的和正好是下一步的两个最小数的其中的一个那么这个树直接往上生长就可以了。如果这两个数的和比较大不是下一步的两个最小数的其中一个那么，就并列生长**

**对任何非空二叉树T，若n0 表示叶结点的个数、n2 表示度为2 的非叶结点的个数，那么两者满足关系n0 = n2 + 1。**

**证明：首先，假设该二叉树有N 个节点，那么它会有多少条边呢？答案是N - 1，这是因为除了根节点，其余的每个节点都有且只有一个父节点，那么这N 个节点恰好为树贡献了N - 1 条边。这是从下往上的思考，而从上往下(从树根到叶节点)的思考，容易得到每个节点的度数和 0\*n0 + 1\*n1 + 2\*n2 即为边的个数。**

**因此，我们有等式 N - 1 = n1 + 2\*n2，把N 用n0 + n1 + n2 替换，得到n0 + n1 + n2 - 1 = n1 + 2\*n2，于是有**

**n0 = n2 + 1**

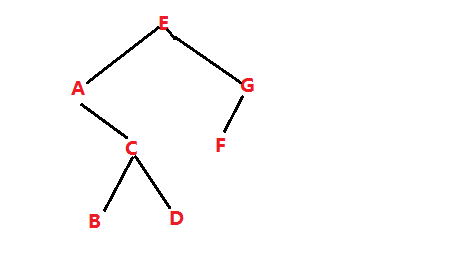
****可以依据二叉树的根节点有无右儿子结点，将一棵二叉树还原成树，具体方法如下：****

****（1） 若某结点是其双亲的左儿子，则把该结点的右儿子，右儿子的右儿子，...都与该结点的双亲结点用线连起来；****

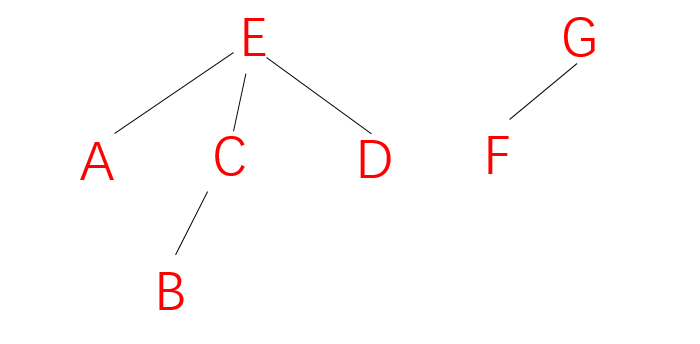
**（2）**删掉原二叉树中****所有的双亲结点与右儿子结点的连线****；

**（3）整理由（1）（2）两步所得到的树，使之结构层次分明。**

**得到二叉树图为**

****

**按上述方法得到还原后的树为**

****

因此该二叉树对应的森林结点的层次序列为EGACDFB