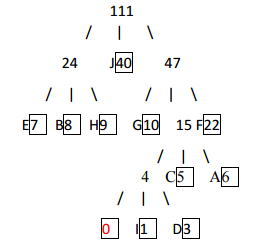
答：Huffman编码总长度至少201位，等长编码 333 （注意是三叉树）



2.

答：1）成立。

（1）如果先序遍历序列和中序遍历序列都是空或则只有一个结点，易知该二叉树是唯一的。

（2）其它的情况，设少于n (n ≥2)个结点的二叉树可由先序遍历序列和中序遍历序列唯一确定。则对于有n 个结点的二叉树，先序遍序序列中的第一个结点必然是该二叉树的根结点，然后在中序遍历序列中找到根结点，故根结点可以唯一确定。在中序遍历序列中根结点前面的结点序列就是根结点左子树的中序遍历序列，在先序遍历序列中根结点后面的属于中序遍历序列中根结点前面的那些结点组成序列就是根结点左子树的先序遍历序列。因为根结点的左子树至少比原二叉树少一个结点，所以根据归纳假设可知根结点的左子树是唯一的: 原中序遍历序列中根结点后面的结点序列就是根结点右子树的中序遍历序列，原先序遍历序列中去掉左子树先序遍历序列后剩下的结点序列就是根结点右子树的先序遍历序列，根结点的右子树至少比原二叉树少一个结点，根据归纳假设可知根结点的右子树也是唯一的。

所以对于有n个结点的二叉树也可以由先序遍历序列和中序遍历序列唯一确定。

由(1)、(2)可知对于任意个结点的二叉树都可由先序遍历序列和中序遍历序列唯一确定。

2）成立。

（1）如果后序遍历序列和中序遍历序列都是空或则只有一个结点，易知该二叉树是唯一的。

（2）其它情况，设少于n (n ≥0) 个结点的二叉树可由后序遍历序列和中序遍历序列唯一确定。则对于有n 个结点的二叉树，后序遍历序列中的最后一个结点必然是该二叉树的根结点。然后在中序遍历序列中找到根结点，故根结点可以唯一确定，在中序遍历序列中根结点后面的结点序列就是根结点右子树的中序遍历序列，在后序遍历序列中根结点前面的属于中序遍历序列中根结点后面的那些结点组成的序列就是根结点右子树的后序遍历序列。因为根结点的右子树至少比原二叉树少一个结点，所以根据归纳假设可知根结点的右子树是唯一的; 原中序遍历序列中根结点前面的结点序列就是根结点左子树的中序遍历序列，原后序遍历序列中去掉右子树后序序列后剩下的结点序列就是根结点左子树的后序遍历序列，根结点的左子树至少比原二叉树少一个结点，根据归纳假设可知根结点的左子树也是唯一的。

所以对于有n 个结点的二叉树也可由后序遍历序列和中序遍历序列唯一确定。

由(1) 、(2) 可知对于任意个结点的二叉树都可由后遍历序列和中序遍历序列唯一确定。

3）不成立。同样的先序遍历序列和后序遍历序列，可以对应不同的二叉树。

反例：已知一棵二叉树的先序遍历序列和后序遍历序列分别为ABC和CBA，则以下四棵二叉树均符合要求：

A A A A

\ \ / /

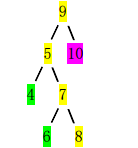
B B B B

\ / / \

C C C C

故已知先序遍历序列和后序遍历序列不能确定唯一一棵二叉树。

答：否。举例如下：5<6



该结论不成立。对于任一a∈A，可在B中找到最近祖先f。a在f左子树上。对于从f到根节点路径上所有b∈B，有可能f在b的右子树上，因而a也就在b的右子树上，这时a>b，因此a<b不成立。同理可以证明b<c不成立。而对于任何a∈A，c∈C均有a<c。

4.

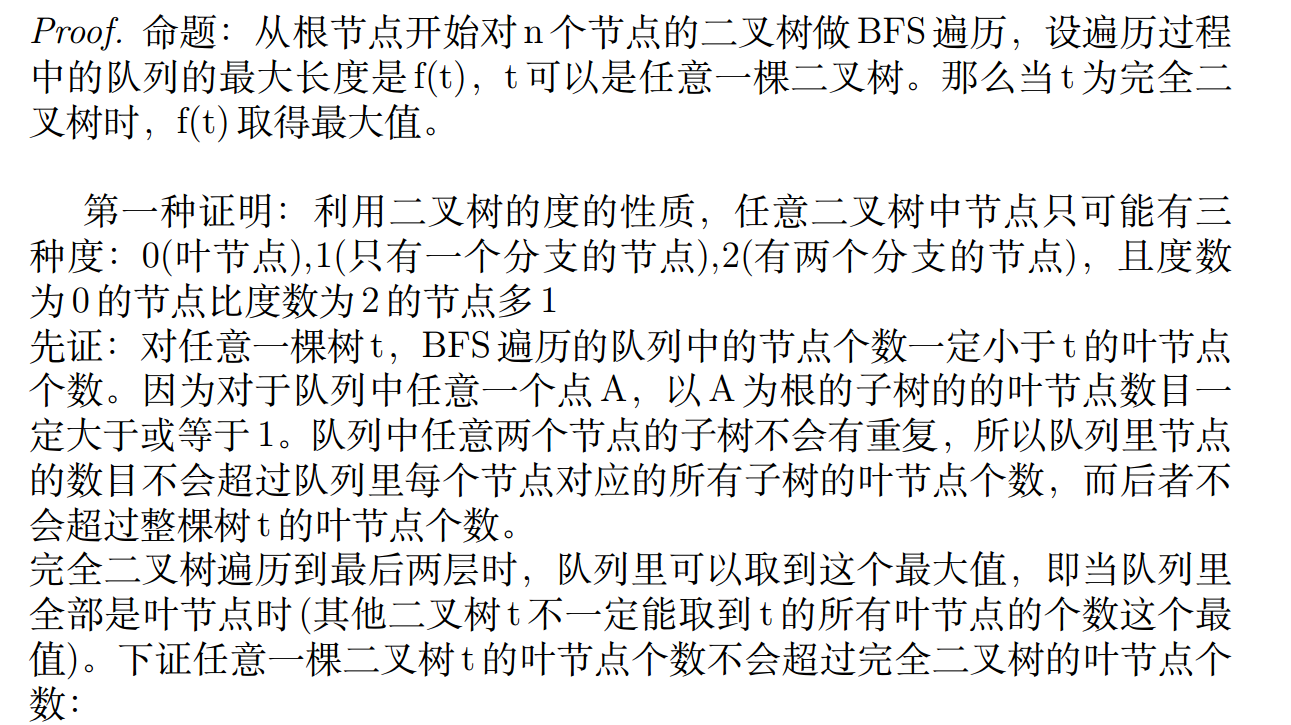
答：①element.tag = Left;

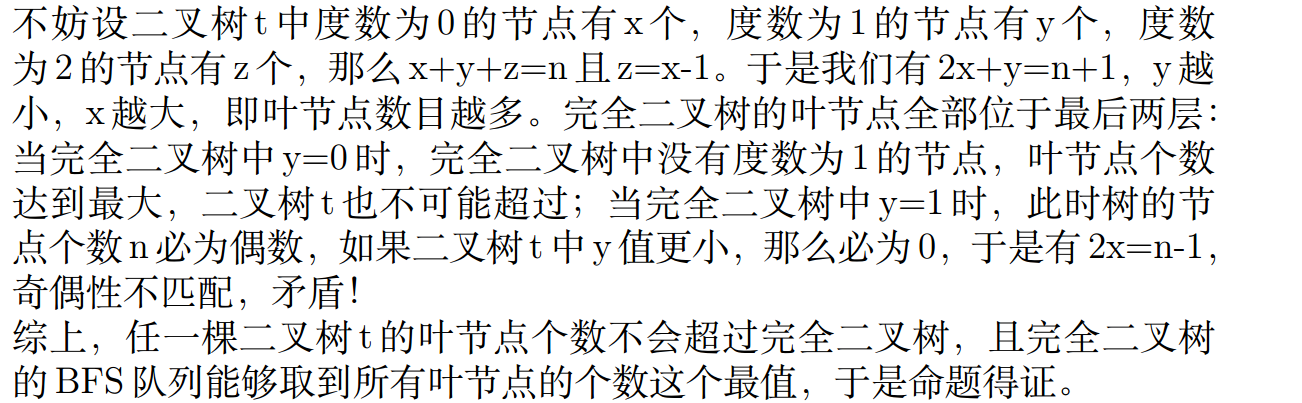
② !aStack.empty() && aStack.top().tag == Right

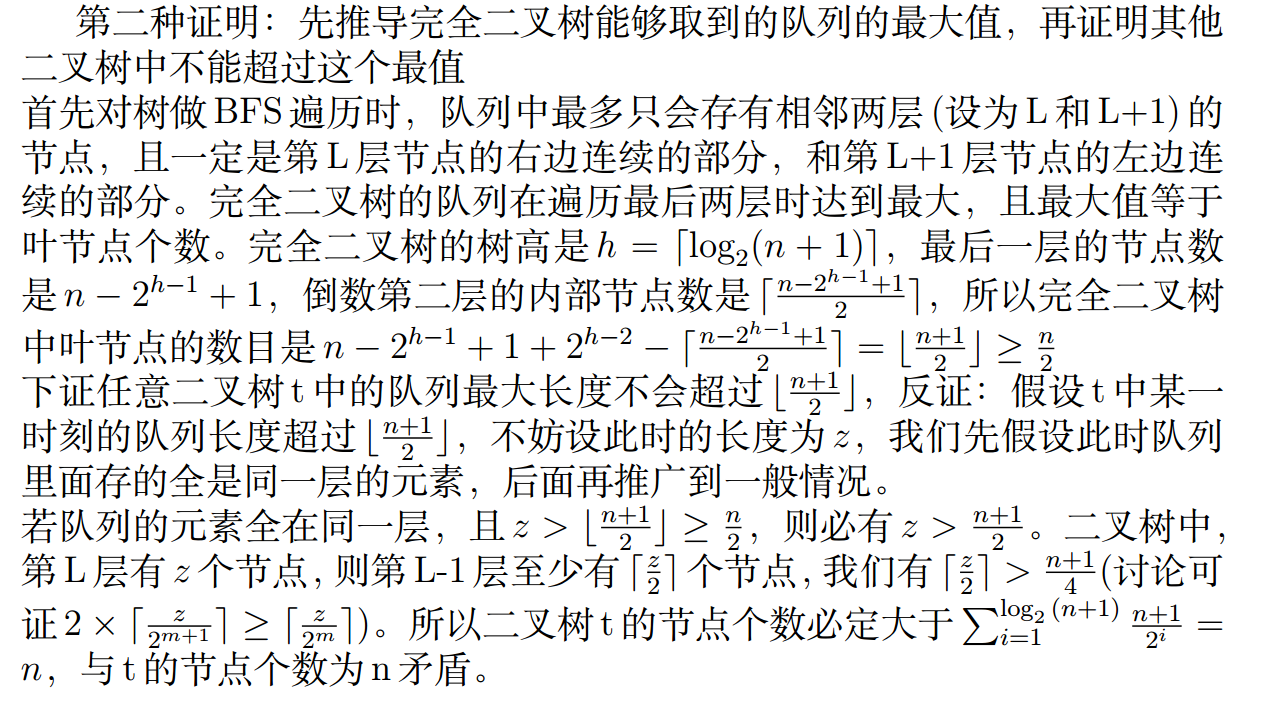
③ pointer = aStack.top().pointer->rightchild();

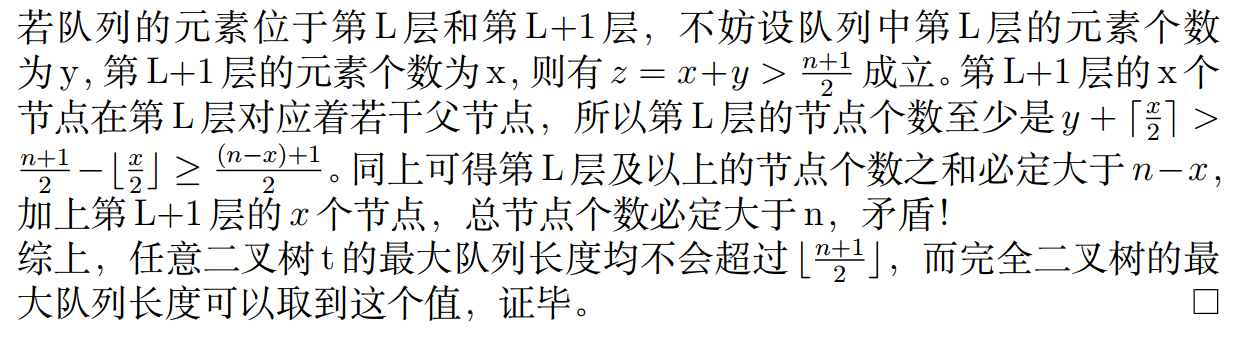
5.

答：说法正确，以下是证明：









注：

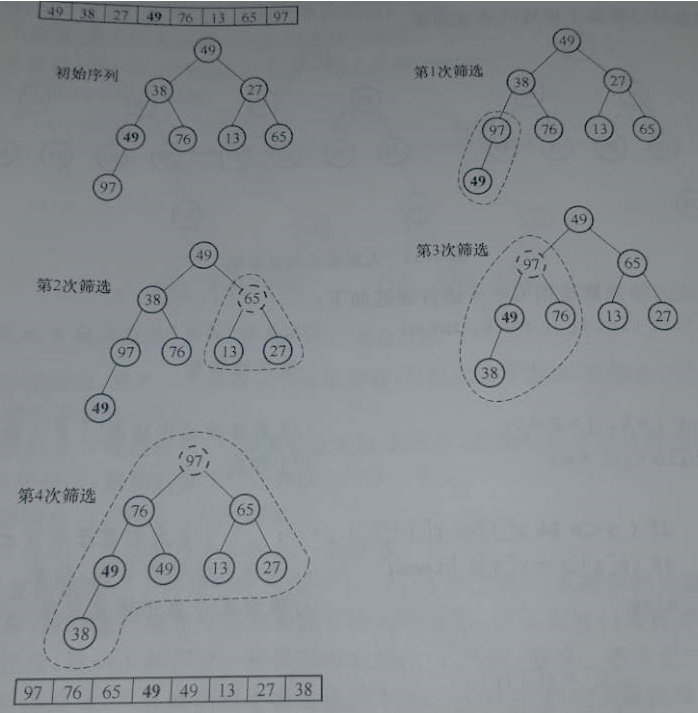
（1）二叉树中每个度数为2的节点都会增加一条分支，也就是新增一条路径。最终路径数目是1+度数为2的节点数目，而每个叶节点又对应一条路径，所以度数为0的节点数目比度数为2的节点数目多1。(多叉树或2-3树也可以类似分析)

（2）和另一种度数联系：在图中，总的度数之和是边数的两倍。若把二叉树看成图，总度数是2(n-1)，原本度数为0、1、2的节点实际上度数分别为1、2、3（注意根节点的度数仍然为2），计算可知这和二叉树的路径数目是能对应上的

（3）完全二叉树一定取到最值，但取到最值的不一定是完全二叉树，很容易举出其他的例子

6.

答：调整过程如下：



最终序列为{97, 76, 65, 49, 49, 13, 27, 38}，移位次数为6。