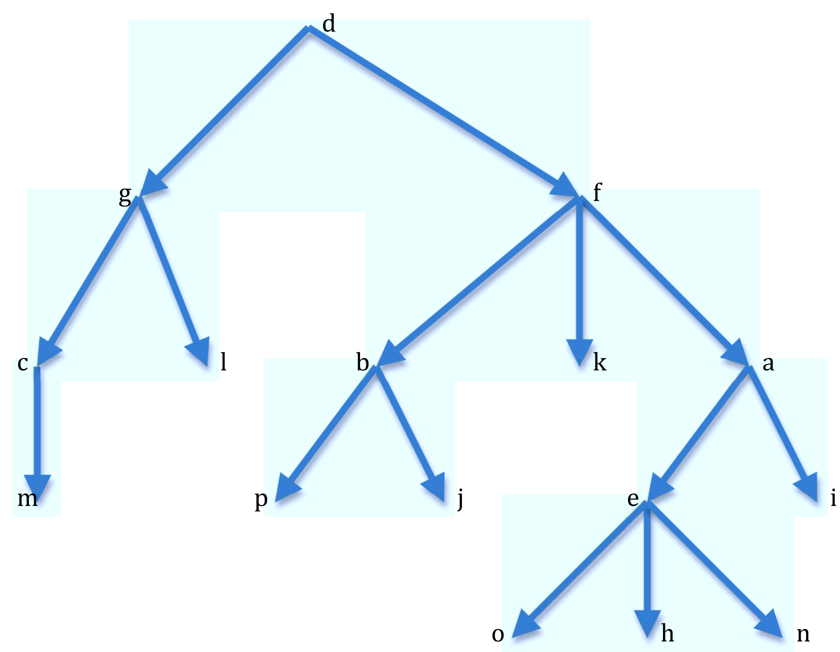
CS182 **Homework 9 Sketch of Solution** **Spring 2011**

Question 1.



The inorder listing of the nodes is:

$$m\ c\ g\ l\ d\ p\ b\ j\ f\ k\ o\ e\ h\ n\ a\ i$$

Question 2.

- v is ancestor of w if and only if

$$Preorder(v) < Preorder(w) \leq Preorder(v) + Desc(v) - 1$$

- v is to the left of w if and only if it is not an ancestor of w and $Preorder(v) < Preorder(w)$.

Question 3. It is correct, for the following reasons. The first observation we make is that we never visit a node that has fewer than $n/3$ descendants. This is certainly true when we start at the root,

and if it is true at any v then it remains true at its child w which is visited next: because v is not the answer (otherwise we would have stopped at v and not gone to w), v must have more than $2n/3$ descendants and therefore its "heavier" child w must have at least $n/3$ descendants.
The number of descendants of the nodes visited is strictly decreasing as we go from one node to the next, so we must end up reaching some node that has no more than $2n/3$ descendants. Such a node has at least $n/3$ descendants (by the above observation) and therefore it is returned as answer.

Question 4. The algorithm is correct. First, note that a node x with the property we seek is characterized as follows in terms of the $Desc$ array: every one of its children y has $Desc(y) \leq n/2$, and, if we let $T' = T$ minus the subtree of x, then T' has no more than $n/2$ nodes (which is same as saying that $Desc(x) \geq n/2$). Next, note that we visit only nodes that have more than $n/2$ descendants. This is certainly true when we start at the root, and if it is true at any v then it remains true (by definition) at its child w which is visited next. Because the number of descendants of the nodes visited is strictly decreasing as we go from one node to the next, we must end up reaching a v for which each child w has $Desc(w) \leq n/2$. To show that it is correct to return such a v as answer, we must also show that, if we let $T' = T$ minus the subtree of v, then T' has no more than $n/2$ nodes. But in fact T' has fewer than $n/2$ nodes, because $Desc(v) > n/2$ (otherwise the search would not have gone to v in the first place).