(1) [10] For each of the following five statements argue convincingly why it is **correct** or **incorrect.**

(a) A minimum spanning tree is not necessarily unique.

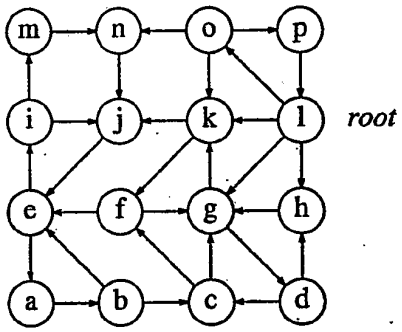
(b) A directed acyclic graph cannot be totally ordered.

(c) Depth first search can be used to determine the *strongly connected* components of a graph.

(d) A partially ordered tree can be transformed into a totally ordered tree in $O(n \lg n)$ time.

(e) Hash search is more efficient than binary search, but it can be as bad as linear search.

(2) [5] Consider the following graph.

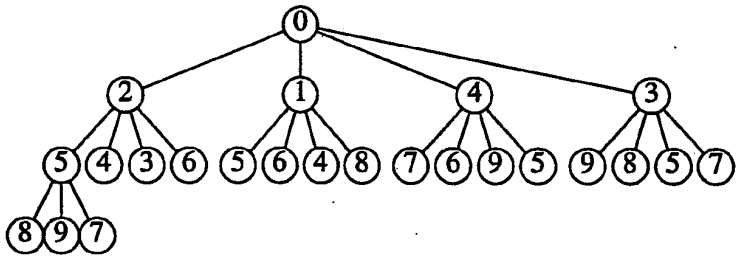


(a) Compute the *depth first spanning forest* of the graph depicted above and label the nodes with the *preorder* and *postorder* numberings (e.g., 7/*q*/11 where 7 is the preorder number and 11 is the postorder number of node *q*). The arcs incident upon a node are to be visited in clockwise order starting at nine o'clock. The node labeled *l* is the root node.

(b) Re-draw the entire graph and partition the entire set of arcs into *tree*, *back*, *forward*, and *cross* arcs using the depth first spanning forest computed under (a) and the following legend.

Tree arc
Cross arc
Back arc
Forward arc

(3) [8] Consider the following quaternary tree data structure.



(a) Traverse the above quaternary tree data structure in *quintuple order* where quintuple order is defined recursively as follows:

(1) Visit the root (i.e., output the node label).
(2) Traverse the first subtree in quintuple order.
(3) Visit the root again (i.e., output the node label).
(5) Traverse the second subtree in quintuple order.
(6) Visit the root again (i.e., output the node label).
(7) Traverse the third subtree in quintuple order.
(8) Visit the root again (i.e., output the node label).
(9) Traverse the fourth subtree in quintuple order.
(10) Visit the root again (i.e., output the node label).

(b) Show how the above tree data structure can be represented using a *quaternary heap* (i.e., an array data structure). Outline how parent and children relate to each other with respect to array indices. Note that all interior nodes with the possible exception of the "last" node have exactly four children).

(c) Turn the above quaternary partially ordered tree data structure into a *total order* using a minimal number of *Siftdown* operations. Show the exact tree configuration after each of the first three Siftdown operations.

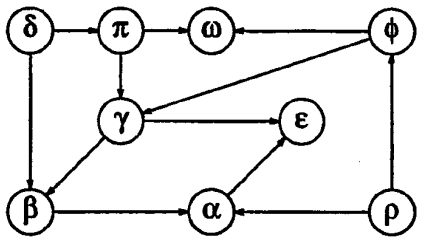
(d) Give the worst-case asymptotic time complexity of a single *DeleteMin* operation applied to a partially ordered *quaternary* tree.

(4) [3] Consider an *undirected* graph $G = (V, E)$.

(a) Describe an algorithm that determines whether G is a binary tree.

(b) What is the time complexity of your algorithm?

(5) [3] The activities of a project and the precedence among the tasks can be represented by a directed, acyclic graph (dag). Certain activities cannot be started until other have been completed. The figure below shows such a dag representing the activities of a project.

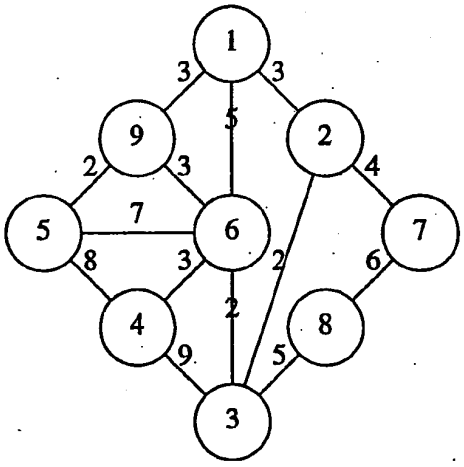


The problem of topological sorting is to embed the partial order in a linear order. Graphically, this implies the arrangement of the vertices of the graph in a row, such that all arrows point in one direction. Topological sort can easily be accomplished using a depth-first search algorithm.

(a) Outline a depth-first search algorithm which outputs the labels of the nodes in topological order. You need to show the Pseudo-code of the actual depth-first search routine and some code that outlines how this routine is called.

(b) Show how your algorithm traverses the dag and determine its topological sort.

(6) [4] Consider the following graph.



(a) Compute a minimum spanning tree of this graph.

(b) Is there another minimum spanning tree with equal weight? Explain.

(7) [7]

(a) Consider a perfectly balanced binary tree T with n nodes. Determine the number of edges of T, the number of interior nodes of T, and the number of leaf nodes of T. Justify your answer.

(b) How many edges can be added to this graph T to turn it into a complete graph G? Justify your answer.

(c) Given three nodes labeled 1 through 3. How many undirected self-loop-less graphs can be constructed with these nodes? Justify your answer.

(d) Given a complete graph consisting of 4 nodes labeled a, b, c, and d. How many distinct spanning trees does this graph have?

— END —