# Sample Final Exam

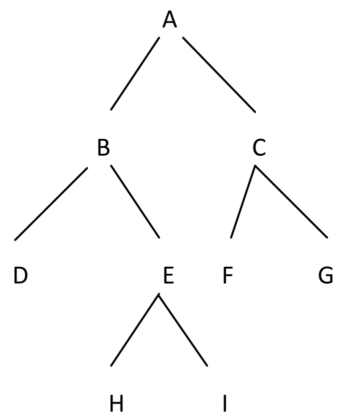
## CSE 680: Introduction to Algorithms and Data Structures

1. Solve the following recurrences by giving tight Θ-notation bounds using the Master Method. If I thought you would need the actual value of a logarithm, I have given it to you.
   (a) $T(n) = 9T(n/3) + n$

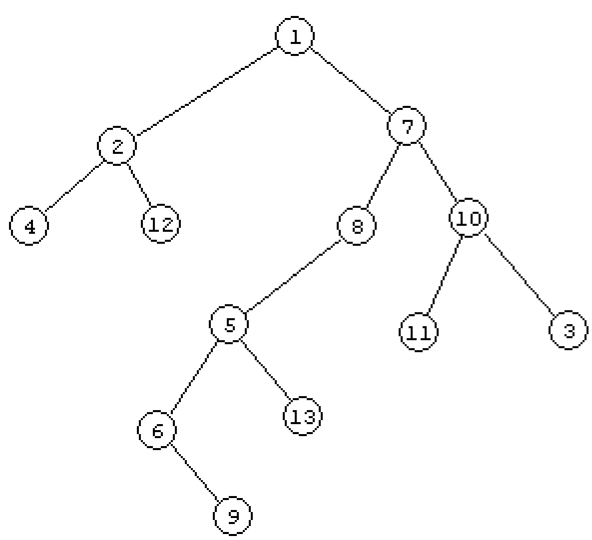
   (b) $T(n) = T(2n/3) + 1$

   (c) $T(n) = 3T(n/4) + n\lg n$

2. Give the Inorder, Postorder and Preorder traversals of the following binary tree.




3. The picture below represents a binary search tree. The numbers shown are arbitrary node labels, not numbers representing the contents of the nodes. **The contents are not shown**. If node 1 is deleted, using binary search tree deletion, what will be the new root node?



4. Answer True or False. Justify your answer. Each answer is for 3 points.

a. The topological sort of an arbitrary directed graph *G*(*V;E*) can be computed in linear time.

b. Kruskal's algorithm for minimum weight spanning trees is an example of a divide and conquer programming algorithm.

c. The shortest path between two vertices is unique if all edge weights are distinct.

d. An arbitrary graph with *G*(*V;E*), with $|E| = |V|$ edges is a tree.

e. A directed graph is strongly connected if and only if a DFS started from any vertex will visit every vertex in the graph without needing to be restarted.

5. Given an adjacency matrix representation of a graph how long does it take to compute the out-degree of all vertices? How long does it take to compute in-degree of all vertices? How long does it take to compute in-degree and out-degree of a single vertex?

6. Consider the graph in Figure 2. Unless otherwise indicated, always visit adjacent nodes in alphabetical order.

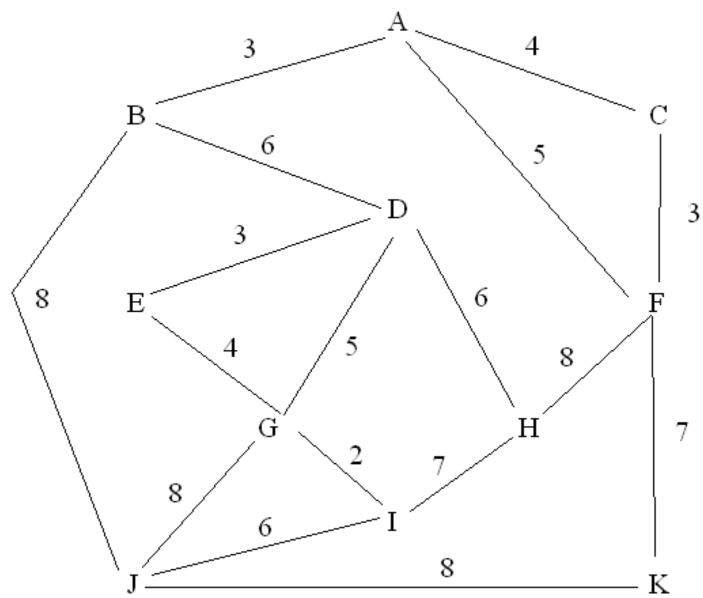



**Figure 1 Weighted Graph**

(a) Provide the DFS tree starting at node A.
(b) Provide the BFS tree starting at node A.
(c) Provide the DFS tree starting at node H.
(d) Provide the BFS tree starting at node H.
(e) Use Kruskal's algorithm to derive the MST.
(f) Use Prim's algorithm to derive the MST starting at node A.
(g) Using Dijkstra's algorithm, determine the shortest path from node A to I. Show the steps, your tables and the resulting path.

7. Given the structure of a heap as sketched below, where the second-smallest value in the set is marked with a **2**. Mark a **4** for each node that can possibly contain the fourth-smallest value in the set. Assume that there are no duplicate node values.