Analysis of Algorithms: Exam 2

March 25, 1999

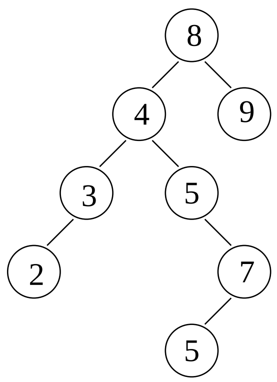
The exam includes nine regular problems, 10 points each, and a bonus problem. The length of the exam is 70 minutes (11:05 to 12:15).

Print your name (10 points):

____

Problem 1 (10 points)

Suppose that you call the TREE-INSERT procedure to add a new node, with value 6, to the binary-search tree shown below, and then you call TREE-DELETE to remove the node with value 4. Draw the resulting trees **(a)** after the insertion of 6 and **(b)** after the deletion of 4.

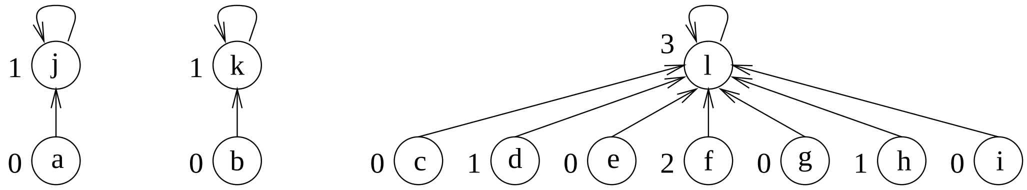


Problem 2 (10 points)

Give the time complexity (O-notation) of the operations on disjoint sets, for the **(a)** linked-list representation and **(b)** forest representation of disjoint sets. Explain the meaning of the variables (m and n) in your complexity expressions.

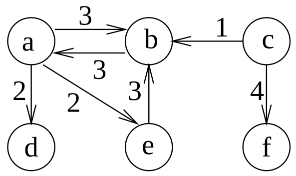
Problem 3 (10 points)

Consider the disjoint-set forest shown below, where numbers are the ranks of elements, and suppose that you apply three successive operations to this forest: UNION(a, b), UNION(b, c), and FIND-SET(a). Give a picture of the disjoint forest after each of these operations (thus, you need to draw three different pictures).



Problem 4 (10 points)

For the following weighted graph, give its **(a)** adjacency-lists representation and **(b)** adjacency-matrix representation. Assume that edge weights represent distances, and the absence of an edge between two vertices corresponds to an infinite distance.



Problem 5 (10 points)

Give the worst-case time complexity for each of the following graph algorithms:

(a) Topological sort

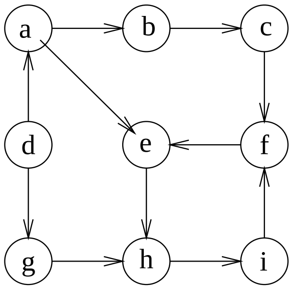
(b) Kruskal's minimum spanning tree

(c) Prim's minimum spanning tree (with a binary-heap queue)

(d) Dijkstra's single-source shortest paths (with a binary-heap queue)

(e) Single-source shortest paths in an acyclic graph

Problem 6 (10 points)

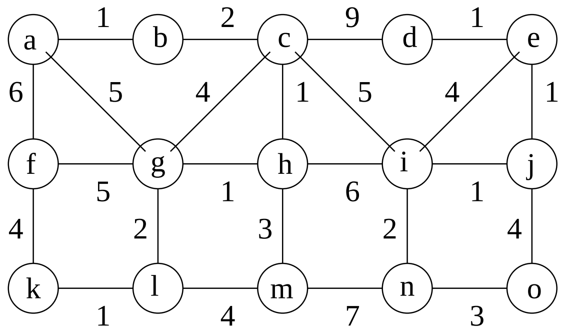


(a) Suppose that you apply the breadth-first search algorithm to the above graph, with vertex a as the source. List all vertices visited by the algorithm, *in the order of painting them gray.*

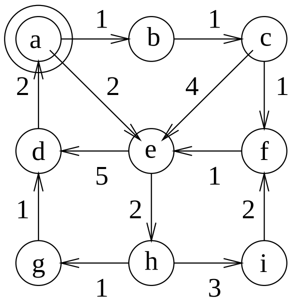
(b) Now suppose that you apply the depth-first search algorithm to the same graph, and the main loop of the algorithm processes the vertices in the alphabetical order, from a to i. List the vertices of the graph *in the order of painting them gray.*

Problem 7 (10 points)

(a) Construct a minimum spanning tree for the following graph. You may draw the edges of the tree directly in the graph.

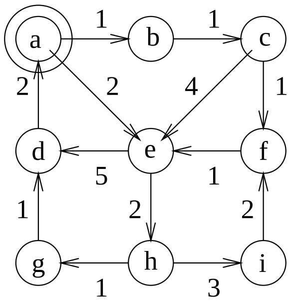


(b) Construct a shortest-paths tree for the following graph, with vertex a as the source.



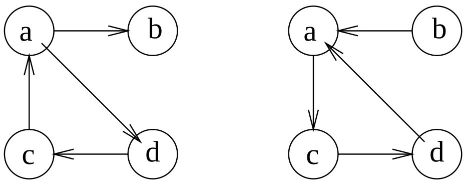
Problem 8 (10 points)

Suppose that you are running Dijkstra's shortest-paths algorithm on the graph shown below (which is the same as in the previous problem), with vertex a as the source, and the algorithm has just painted vertex f black. At this point, which other vertices are black, and which vertices are gray? Mark all black and gray vertices in the graph.



Problem 9 (10 points)

Write an algorithm that reverses all edges of a given graph, that is, replaces every edge (u, v) with the opposite edge (v, u), and returns the resulting graph of reversed edges. For example, given Graph A (see below), the algorithm must return Graph B. Both initial and returned graph must be represented by *adjacency lists*. Give the asymptotic (Θ-notation) time complexity of your algorithm.



Graph A   Graph B

Problem 10 (bonus)

This problem is optional and does not affect your grade for the exam; if you solve it, then you get 5 bonus points toward your final grade for the course.

Suppose that you are using a programming language that supports four operations on real numbers: addition, subtraction, multiplication, and division; the running time of each operation is constant, that is, $\Theta(1)$. Note that this language does *not* have operations for logarithms and exponentiation.

Write an efficient algorithm POLYNOMIAL(x, A, n) for computing the value of a polynomial. The arguments of the algorithm are a value of x and an array of coefficients $A[0..n]$, and the output is the value of the following polynomial:

$$A[n] \cdot x^n + A[n-1] \cdot x^{n-1} + A[n-2] \cdot x^{n-2} + ... + A[1] \cdot x + A[0].$$

Give the asymptotic time complexity (Θ-notation) of your algorithm. Note that computing the polynomial in $\Theta(n^2)$ time is *too slow,* and an algorithm with this running time will get you only 1 bonus point; try to design a faster algorithm.