Reading. CLRS Sections 17.1, 17.2, 17.3, 17.4, 22.1, 22.2, 22.3, 22.4, pages 451 -- 471, 589 -- 615.

Exercise 9-1. Below is an implementation of a first-in-first-out (FIFO) queue that represents the queue as a pair of two lists, front and back:

Empty-Queue()
  return new record {front = EmptyList(), back = EmptyList()}

Empty-Queue?(Q)
  return EmptyList?(front[Q]) and EmptyList?(back[Q])

Enq(x, Q)
  back[Q] = Prepend(x, back[Q])

Deq(Q)
  if EmptyList?(front[Q])
    front([Q]) = Reverse(back[Q]) {linear-time Reverse}
    back([Q]) = EmptyList()
  if EmptyList?(front[Q])
    "empty queue"
  else
    h = Head(front[Q])
    front[Q] = Tail(front[Q])
  return h

a. Show the state of the queue (i.e., the front and back lists) after each of the operations in the following sequence:

q = Empty-Queue()
Enq(1,q)
Enq(2,q)
Enq(3,q)
Deq(q)
Enq(4,q)
Enq(5,q)
Deq(q)
Deq(q)
Enq(6,q)
Deq(q)
Deq(q)
b. Using the accounting method, argue that the amortized cost of \texttt{Enq} and \texttt{Deq} are each \(O(1)\).

c. Using the potential method, argue that the amortized cost \texttt{Enq} and \texttt{Deq} are each \(O(1)\).

**Exercise 9-2.** Draw the 2-3-4 tree (that is the B-tree with \(t = 2\)) that results from inserting the following letters one-by-one (from left-to-right) into an empty tree:

\[
\text{ALGORITHM}
\]

You need only show the final tree that results from inserting all of the letters; do not show the intermediate tree after inserting each letter.

b. Draw the sequence of 2-3-4 trees that are obtained by deleting the letters of \texttt{ALG} one by one from the tree given in part \(a\).

**Exercise 9-3.** Consider the following directed graph \(G\):

In the following problems, you should assume that \(G\) is represented as a collection of adjacency lists, and that vertices are ordered alphabetically within each adjacency list.

a. Draw the tree that is induced by performing breadth-first search starting at node \(d\). Indicate clearly the order of actions taken. Show the queue's contents at all times.

b. Draw the tree that is induced by performing a depth-first-search starting at node \(d\). Label each vertex by its discovery and finishing times and classify each edge as tree, back, forward or cross as in lecture.

c. Find a topological sort of the graph \(G''\), where \(G''\) is the result of removing the two edges \((i, d)\) and \((g, f)\) from \(G\). (Note: A topological sort of \(G\) is not well-defined because \(G\) contains cycles.)

d. Draw a copy of \(G\) and circle the strongly-connected components of \(G\).
Exercise 22.4-3. Give an algorithm that determines whether or not a given undirected graph $G = (V, E)$ contains a cycle. Your algorithm should run in $O(V)$ time, independent of $|E|$.

a. Provide pseudocode for your algorithm. For full credit carefully justify your algorithm's running time.

b. Does your algorithm from part (a) work for directed graphs? Explain. If so, compare your algorithm's running time for the two different types of graphs (directed and undirected).