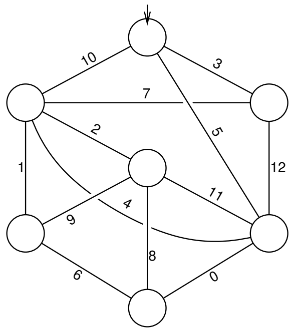
1. Using any method you like, compute the following subgraphs for the weighted graph below. Each subproblem is worth 3 points. Each incorrect edge costs you 1 point, but you cannot get a negative score for any subproblem.

   (a) a depth-first search tree, starting at the top vertex;

   (b) a breadth-first search tree, starting at the top vertex;

   (c) a shortest path tree, starting at the top vertex;

   (d) the **maximum** spanning tree.

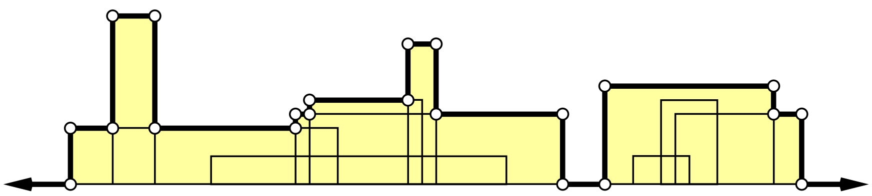


2. (a) *[4 pts]* Prove that a connected acyclic undirected graph with V vertices has exactly $V-1$ edges. ("It's a tree!" is not a proof.)

   (b) *[4 pts]* Describe and analyze an algorithm that determines whether a given undirected graph is a tree, where the graph is represented by an adjacency list.

   (c) *[2 pts]* What is the running time of your algorithm from part (b) if the graph is represented by an adjacency matrix?

3. Suppose we want to sketch the Manhattan skyline (minus the interesting bits like the Empire State and Chrysler builings). You are given a set of n rectangles, each rectangle represented by its left and right x-coordinates and its height. The bottom of each rectangle is on the x-axis. Describe and analyze an efficient algorithm to compute the vertices of the skyline.



A set of rectangles and its skyline. Compute the sequence of white points.

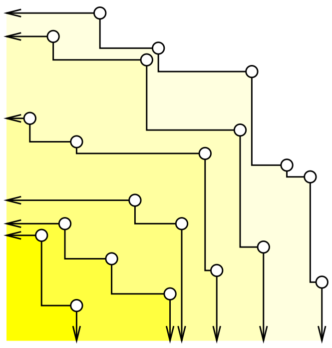
4. Suppose we model a computer network as a weighted undirected graph, where each vertex represents a computer and each edge represents a *direct* network connection between two computers. The weight of each edge represents the *bandwidth* of that connection—the number of bytes that can flow from one computer to the other in one second.[1] We want to implement a point-to-point network protocol that uses a single dedicated path to communicate between any pair of computers. Naturally, when two computers need to communciate, we should use the path with the highest bandwidth. The bandwidth of a *path* is the *minimum* bandwidth of its edges.

   Describe an algorithm to compute the maximum bandwidth path between *every* pair of computers in the network. Assume that the graph is represented as an adjacency list.

5. *[1-unit grad students must answer this question.]*

   Let P be a set of points in the plane. Recall that the *staircase* of P contains all the points in P that have no other point in P both above and to the right. We can define the *staircase layers* of P recursively as follows. The first staircase layer is just the staircase; for all $i > 1$, the ith staircase layer is the staircase of P after the first $i - 1$ staircase layers have been deleted.

   Describe and analyze an algorithm to compute the staircase layers of P in $O(n^2)$ time.[2] Your algorithm should label each point with an integer describing which staircase layer it belongs to. You can assume that no two points have the same x- or y-coordinates.



A set of points and its six staircase layers.

[1] Notice the bandwidth is symmetric; there are no cable modems or wireless phones. Don't worry about systems-level stuff like network load and latency. After all, this is a theory class!

[2] This is *not* the fastest possible running time for this problem.