COMP 3711 Design and Analysis of Algorithms
Spring 2015 Final Exam

1. **Data structures** (12 pts)

   For each of the following tasks, decide what should be the most efficient data structure, and say what the cost is for each operation to be supported (you don't need to describe how to do it). The data structure should occupy linear space.

   1.1 Data: A collection of n (IP address, URL) pairs. Each IP address is an integer, and URL is a string.

   Operations to be supported:

   (a) Given an IP address, find its corresponding URL, or report that it doesn't exist.

   (b) Given a new (IP address, URL) pair, insert it into the data structure.

   1.2 Data: A collection of n (IP address, URL) pairs. Each IP address is an integer, and URL is a string.

   Operations to be supported:

   (a) Given an IP address, find its corresponding URL; if it doesn't exist, find the closest IP address and its corresponding URL.

   (b) Given a new (IP address, URL) pair, insert it into the data structure.

   1.3 Data: A collection of n IP addresses. Each IP address is an integer.

   Operations to be supported:

   (a) Find the smallest IP address.

   (b) Delete the smallest IP address from the data structure.

   (c) Given a new IP address, insert it into the data structure.

   1.4 Data: A collection of n IP addresses. Each IP address is an integer.

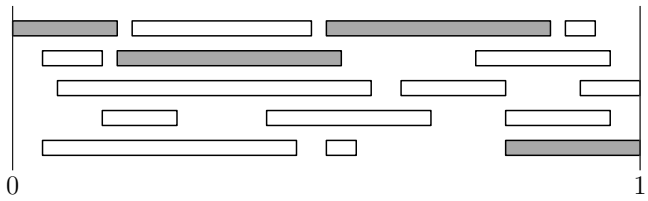
   Operations to be supported:

   (a) Given two IP addresses x and y, check if they are connected.

   (b) A network link is built between two given IP addresses x and y. Reflect this change in your data structure.

2. **Interval covering** (15 pts)

   This problem is similar to the homework question but it is different. Suppose your company wants to build cell phone base stations to cover a highway. However, not every location is allowed to build, and the government has designated a certain number of locations. Depending on how far the location is from the highway, its coverage on the highway also varies. Thus, this problem can be generally modeled as follows. The highway can be thought of as the real line from 0 to 1, and each location covers a certain interval $[x, y] \subseteq [0, 1]$ if a base station is built there. You are given a total of n such intervals and are guaranteed that all of them together cover $[0, 1]$ (otherwise there is no solution to the problem). Your job is to design an algorithm that finds the minimum number of intervals that together cover $[0, 1]$. The figure belows shows an example where an optimal solution consists of the 4 shaded intervals (the optimal solution may not be unique). The intervals are given as two arrays $(x_1, \ldots, x_n)$ and $(y_1, \ldots, y_n)$ where the i-th interval is $[x_i, y_i]$. If you use a greedy algorithm, you must prove that it is correct.



3. **Shortest symmetric supersequence** (13 pts)

   Recall that one of the homework questions is to find the longest symmetric subsequence of a given string. Here your job is to design an algorithm to find the shortest symmetric *supersequence*. For example, for the string `ACBAC`, the shortest symmetric supersequence is `CACBCAC` or `ACBABCA`, both of which have length 7. Your algorithm only needs to output the optimal length, not the actual symmetric supersequence.

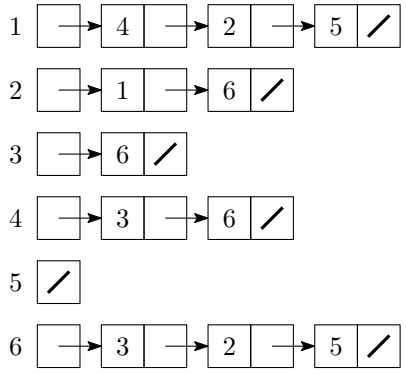
4. **BFS and DFS** (15 pts)

   The figure below shows the adjacency list representation of a directed graph.

   (a) (5 pts) Draw the graph represented.

   (b) (5 pts) Draw the BFS-tree of the graph, using vertex 1 as the source vertex.

   (c) (5 pts) Draw the DFS-tree of the graph, using vertex 1 as the source vertex.



5. **Minimum spanning tree and the cut lemma** (15 pts)

   Let G be a connected undirected graph with distinct weights on the edges, and let T be the MST of G. Recall that the cut lemma states: For *any* cut $(S, V - S)$ of the graph, the minimum-weight edge e crossing the cut must belong to T. Prove or disprove (i.e., give a counter example) the following "reverse cut lemmas".

   (a) Every edge e of T must be the minimum-weight edge crossing *any* cut of G.

   (b) Every edge e of T must be the minimum-weight edge crossing *some* cut of G.

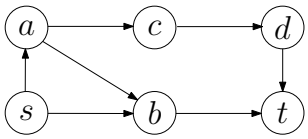
6. **Longest path in a DAG** (15 pts)

   You have solved the longest path problem on a directed acyclic graph when either the vertices or the edges have weights. Now please design an algorithm that solves the most general case where both edges and vertices have weights. More precisely, given the weight $w(v)$ for every $v \in V$ and the weight $w(u, v)$ for every edge $(u, v) \in E$, a source vertex s and a destination t, your algorithm should find the path from s to t whose total weight (sum of all edge weights and vertex weights along the path) is maximized. Your algorithm should output the actual longest path.

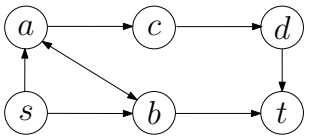
7. **Random routing** (15 pts)

We can model a computer network as a directed graph, where each node is a computer and a directed edge (u, v) means that u can send messages to v. Ideally, messages should be routed along the shortest path, but this information may not be available since no node has a global view of the entire network. Thus, decisions have to be made by each node locally. There are sophisticated routing protocols (which you can learn in COMP 4621); here we just consider one of the simplest routing protocols: RANDOM. In RANDOM, each node simply randomly chooses an outgoing link, and forwards the message along that link, hoping the message can find its way eventually.

Consider the graph G_1 below. Suppose we have a message at the source node s intending to reach destination t. Using RANDOM, s will forward the message to a or b, each with probability $1/2$. If b gets it, it will forward the message to t in the next step, as t is its only outgoing neighbor. If a gets the message, it will choose b or c in the next step, each with probability $1/2$, and so on so forth.



G_1



G_2

(a) (2 pts) On graph G_1, what's the probability that RANDOM indeed chooses the shortest path from s to t to route the message?

(b) (4 pts) On graph G_1, what's the expected length (in terms of the number of edges) of the path chosen by RANDOM?

(c) (9 pts) Answer question (a) and (b) on graph G_2. Note that the only difference is that G_2 has one more directed edge from b to a. [Hint: For question (b), compute the expected contribution of each edge to the path and use linearity of expectation. The key is: how long will the message stay in the loop $a \leftrightarrow b$ in expectation?]