CMPS 101
Homework Assignment 8

1. B.5-4 page 1180
   Use induction to show that a nonempty binary tree with n nodes and height h satisfies $h \geq \lfloor \lg n \rfloor$.
   Hint: use the following recursive definition of height discussed in class:

$$h(T) = \begin{cases} -\infty & n(T) = 0 \\ 0 & n(T) = 1 \\ 1 + \max(\,h(L), h(R)\,) & n(T) > 1 \end{cases}$$

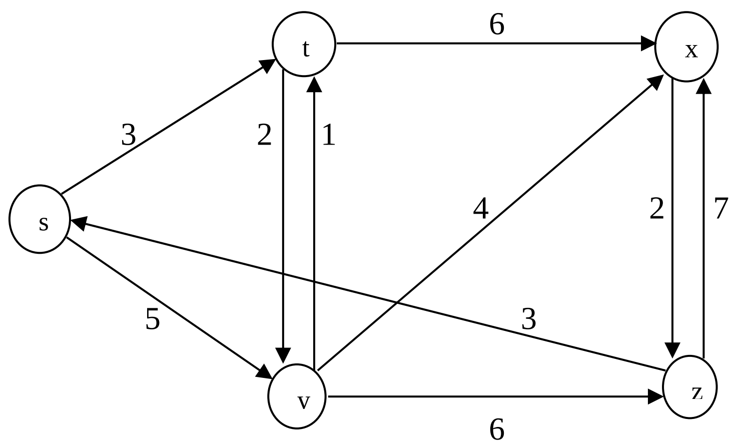
   Here $n(T)$ denotes the number of nodes in a binary tree T, $h(T)$ denotes its height, L denotes its left subtree, and R its right subtree. Note that this proof can be phrased equally well as an induction on $n(T)$ or on $h(T)$.
   Additional hint: use (and prove) the following fact: $\lfloor \lg(2k+1) \rfloor = \lfloor \lg(2k) \rfloor$ for any positive integer k.

2. 6.5-3 page 165
   Write pseudocode for the procedures HeapMinimum, HeapExtractMin, HeapDecreaseKey, and HeapInsert that implement a min-priority queue with a min-heap.

3. Let $G = (V, E)$ be a weighted directed graph and let $x \in V$. Suppose that after Initialize(G, s) is executed, some sequence of calls to Relax( ) causes $d[x]$ to be set to a finite value. Prove that G contains an s-x path of weight $d[x]$. (Hint: use induction on the number of calls to Relax( ).

4. 24.1-3 p. 654
   Given a weighted directed graph $G = (V, E)$ with no negative-weight cycles, let m be the maximum over all vertices $x \in V$ of the minimum number of edges in a shortest path from the source $s \in V$ to x. (Here, the shortest path is by weight, not by the number of edges.) Suggest a simple change to the Bellman-Ford algorithm that allows it to terminate in $m+1$ passes, even if m is not known in advance.

5. 24.3-1 p. 662
   Run Dijkstra's algorithm on the directed graph of Figure 24.2 p. 648 (pictured below), first using vertex s as the source and then using vertex z as the source. Show the d and π values and the vertices in set S after each iteration of the **while** loop.



6. 24.3-6 p. 663

   We are given a directed graph $G = (V, E)$ on which each edge $(u, v) \in E$ has an associated value $r(u, v)$, which is a real number in the range $0 \le r(u, v) \le 1$ that represents the reliability of a communication channel from vertex u to vertex v. We interpret $r(u, v)$ as the probability that the channel from u to v will not fail, and we assume that these probabilities are independent. Give an efficient algorithm to find the most reliable path between two given vertices.