In preparing your solutions to the exam, you are **allowed to use any sources** including your textbook, other students and professors, previous homeworks and solutions, or any sources on the Internet. You may ask me for feedback on potential solutions, but I will not give you any hints. Of course, I am happy to answer general questions, go over homework problems, or answer clarifying questions about exam problems.

The exam will take place in class on Friday, February 24 (MC Reynolds 317, 8:10-9:00am). You are **not** allowed to bring any notes, textbooks, calculators, or any other resources with you to the exam. Bring only something to write with; I will provide a fresh copy of the exam, paper for writing your solutions, and scratch paper.

Question 1 (20 points)**.** Characterize the asymptotic behavior of each of the following in terms of big-Theta, as a function of n and/or m. Prove/justify your answers. Full credit will only be given for the best (fastest) possible algorithms.

(a) $1 + 2 + 3 + \cdots + n/2$

(b) $4 + 8 + 16 + 32 + \cdots + 2^n$

(c) Time to find the shortest path between two given vertices in an unweighted, undirected graph with n vertices and m edges.

(d) Time to find the shortest path between two given vertices in a weighted, undirected graph (assuming all weights are nonnegative) with n vertices and m edges.

(e) Given an array A of length n, the time to fill in a matrix M such that $M[i, j] = A[i] + A[j]$.

(f) Given an array A containing n positive integers, the time to compute the smallest positive integer which is not contained in the array.

(g) Given an array A containing n positive integers which are all at most 100, the time to compute the smallest positive integer which is not contained in the array.

Question 2 (15 points)**.** Prove, or disprove with a counterexample: if a tree T has a vertex of degree $k \geq 1$, then T has at least k leaves.

Question 3 (15 points)**.** Consider the following algorithm to determine whether an undirected, unweighted graph G has any cycles: pick an arbitrary vertex v and run a breadth-first search (BFS), generating a sequence of layers L_0, L_1, L_2, ... If there are any edges between two vertices in the same layer, then G has a cycle; otherwise, G has no cycles.

Prove that this algorithm is correct, or give a counterexample.

Question 4 (20 points)**.** Recall *Dijkstra's Algorithm* for finding shortest paths in a directed, weighted graph.

(a) Why doesn't Dijkstra's algorithm work if edges in the graph can have negative weights? Give an example of a graph where Dijkstra's algorithm fails to find the minimum-weight path between a pair of vertices.

(b) What happens if we replace the word "smallest" in Dijkstra's algorithm with the word "biggest"—that is, we use a max priority queue so we pull out the *biggest* edge on each iteration, and when we add a vertex u to the set S, for each edge (u, v) we update $d[v]$ to be the *maximum* of $d[v]$ and the sum $d[u] + w_{uv}$. Can we use this modified Dijkstra's algorithm to find *longest* paths bewteen nodes in a graph? Prove that this works, or give a counterexample where it doesn't.

Question 5 (30 points)**.** Let $G = (V, E)$ be an undirected, connected graph with non-negative edge lengths. We say that T is a *min-max spanning tree* on G if T is a spanning tree whose longest edge is as short as possible. That is, if we imagine taking all the possible spanning trees of G and sorting them by the length of their *longest* edge, in increasing order, T would be first (or tied for first) on the list. Put yet another way, T is a min-max spanning tree if and only if every other spanning tree T' has some edge which is longer than (or equal to) every edge in T.

Prove or disprove the following claims:

(a) If T is a minimum spanning tree for G, then T is a min-max spanning tree for G.

(b) If T is a min-max spanning tree for G, then T is a minimum spanning tree for G.