# Homework Assignment #8
# Due: March 16, 2020 at 2:30 p.m.

**1.** In class, we described how to implement the Prim-Jarník minimum spanning tree algorithm using a priority queue. Suppose we run the algorithm on an input graph with $n$ nodes and $m$ edges. Assume the input graph is represented using adjacency lists and that the nodes are numbered $1..n$. If the priority queue is implemented by a min-heap, we saw that the worst-case running time of the algorithm is $\Theta(m \log n)$. Suppose we instead implement the priority queue as an array $A[1..n]$ where $A[i]$ stores the priority of node $i$. For each of the following,

**(a)** What is the worst-case running time of a DecreasePriority operation? State your answer using $\Theta$ notation in terms of $n$ and provide a brief justification.

**(b)** What is the worst-case running time of a ExtractMax operation? State your answer using $\Theta$ notation in terms of $n$ and provide a brief justification.

**(c)** What is the worst-case running time of the Prim-Jarník algorithm using the array implementation of the priority queue. State your answer using $\Theta$ notation in terms of $n$ and $m$ and provide a brief justificaiton.

**(d)** Are there situations where the array implementation of a priority queue is better to use in the Prim-Jarník algorithm than a min-heap?

**2.** Suppose you have an undirected graph that models a computer network. The $n$ nodes in the graph represent computers and the edges connect pairs of computers that share a communication channel. Each edge $(u, v)$ has an associated probability of failure $p(u, v)$. Whenever a message is sent across the edge $(u, v)$, $p(u, v)$ is the probability that it is *not* received by the receiver. We wish to broadcast a message from one node to all other nodes by sending the message (once) across each of $n - 1$ edges. Describe an algorithm that finds the set of $n - 1$ edges to use for this broadcast that maximizes the probability that the message reaches all $n$ computers in the network. You do not have to provide a formal proof of correctness, but you should give a brief justification of why your algorithm finds the optimal solution.