02-713 Homework #4: Shortest Paths
Due: Feb. 12 by 11:59pm — Note unusual time

You may discuss these problems with your current classmates, but you must write up your solutions independently, without using common notes or worksheets. You must indicate at the top of your homework who you worked with. Your write up should be clear and concise. You are trying to convince a skeptical reader that your answers are correct. Your homework should be submitted via Autolab (`https://autolab.cs.cmu.edu/02713-s14/`) as a typeset PDF. A LaTeX tutorial and template are available on the class website if you choose to use that system to typeset. For problems asking for an algorithm: describe the algorithm, an argument why it is correct, and an estimation of how fast it will run. Use O notation for running times.

1. Let $G = (V, E)$ be a connected, undirected graph with positive edge weights $d(u, v)$, and suppose $|E| = \Omega(|V|^2)$. That is, $|E|$ is perhaps $\binom{|V|}{2}/100$. Give an implementation of Dijkstra's shortest path algorithm that is asymptotically faster on such dense graphs than the heap-based version we saw in class.

2. Suppose $G = (V, E)$ is a directed, acyclic graph (a DAG) with positive weights $d(u, v)$ on each edge. Let s be a vertex of G with no incoming edges and such that every other node is reachable from s through some path.

   (a) Give an $O(|V| + |E|)$-time algorithm to compute the shortest paths from s to all other vertices in G. Note that this is faster than Dijkstra's algorithm in general.

   (b) Give an efficient algorithm to compute the *longest* paths from s to all other vertices. (Interestingly, this is a hard problem in general, non-DAG graphs.)

3. Give an $O(|E|)$-time algorithm to check whether a given tree T is a shortest-path tree of an undirected, connected graph $G = (V, E)$ with positive weights $d(u, v)$ on each edge.

   Hint: check whether the edges that are not in T are correctly excluded.