This homework is optional, and is due Wednesday, December 5, at the start of class. The questions are drawn from the material in the lectures and Chapter 24 of the text on shortest paths.

The homework is worth a total of 100 points. The points you earn on this homework may be used to replace your lowest prior homework score (if this benefits your grade). When questions with several parts do not specify the points for each part, each part has equal weight.

Remember to (a) put your name at the top of every page, (b) start each problem on a new page, (c) write on just one side of a page and do not use scrap paper, (d) put your answers in the correct order, and (e) staple your pages together. If you can’t solve a problem, state this, and write only what you know to be correct. Neatness and conciseness count.

(1) **(Shortest paths with vertex weights)** (20 points) Suppose you are given a directed graph where vertices are weighted, instead of edges. The length of a path $P$ is now the sum of the weights of the vertices on $P$.

Given a source vertex $s$, design an efficient algorithm that finds the shortest path from $s$ to every other vertex in the graph, under this new definition of path length.

(Hint: Reduce this to the standard single-source shortest paths problem by constructing an equivalent graph that has weights on edges rather than vertices.)

(2) **(Most reliable path)** (20 points) Suppose you are given a directed graph whose vertices correspond to computers and whose edges correspond to communication links. Each edge is weighted by the probability of the link failing, and links fail independently of each other. The reliability of a path is the probability that all the links on it do not fail.

Given two vertices $s$ and $t$, design an efficient algorithm that finds the most reliable path from $s$ to $t$.

(Hint: Reduce this to the single-source shortest paths problem by transforming the edge weights.)

(3) **(Bottleneck shortest paths)** (30 points) Suppose you are given a directed graph $G = (V, E)$ with edge-weight function $\omega$, where the weights of edges can be positive or negative, together with a fixed source vertex $s$ and sink vertex $t$. A bottleneck shortest path from $s$ to $t$ is an $(s, t)$-path such that the length of the longest edge in the path is minimum. In other words, in a bottleneck $(s, t)$-path we only care about the weight of the heaviest edge in the path, and we want its heaviest edge weight to be as small as possible.

Design an algorithm that finds a bottleneck shortest $(s, t)$-path in $O((n+m) \log m)$ time, where $n = |V|$ and $m = |E|$ are respectively the total number of vertices and edges in $G$.

(Hint: First solve the bottleneck shortest path problem where you are given a fixed upper bound $\ell$ on the allowed length of any edge in an $(s, t)$-path. Then figure out how to quickly find the optimal value of $\ell$ for a bottleneck shortest $(s, t)$-path.)

(4) **(Shortest paths with integer weights)** (30 points) Suppose all edges in the graph have integer weights in the range $[0, k]$. Show how to modify Dijkstra’s algorithm for single-source shortest paths so that it runs in $O(kn + m)$ time on a graph with $n$ vertices and $m$ edges.

(Hint: Change the implementation of the heap used in Dijkstra’s algorithm. Note that the values of the keys returned by the calls to $\text{DeleteMin}$ in Dijkstra’s algorithm form a monotonic increasing sequence.)
(5) (bonus) (Shortest paths revisited) (20 points) Suppose again that all edges have integer weights in the range $[0, k]$. Show how to modify Dijkstra’s algorithm for single-source shortest paths so that it now runs in $O((n+m) \log k)$ time on a graph with $n$ vertices and $m$ edges.

(Hint: Bound the number of distinct values of keys that can appear in the heap. The heap holds distance estimates for vertices on the fringe, which are just outside the set for which shortest path lengths are known.)

Note that Problem (5) is a bonus question, and is not required.