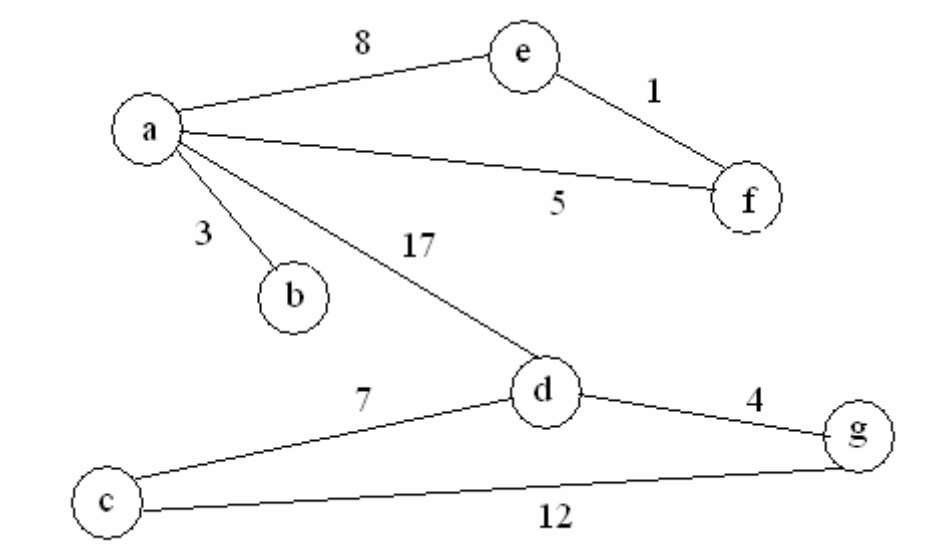
CS2223
HW#6

DUE: Friday, April 21

1. (5 points) Suppose you are given as input a weighted graph $G=(V,E)$, $w: E \to \Re^+$, with a nonempty set of sources $\Sigma \subseteq V$, and you need to compute $D: V \to \Re^+$ such that for each $v \in V$, $D[v]$ is the length (weight, cost) of a shortest path from **any** vertex $w \in \Sigma$ to v. For example, for graph



with $\Sigma = \{a,b,c\}$ the values of D should be

V	a	b	c	d	e	f	g
D	0	0	0	7	6	5	11

To solve this problem for any imput graph, you may modify the graph but you may only invoke DIJKSTRA's algorithm one time. Show how this may be done.

2. (10 points) In the classical single-source shortest path problem, the length of a path is the sum of the lengths of the edges on the path. In the BOTTLENECK SHORTEST PATH PROBLEM, the *bottleneck length* of a path is the weight (length) of a heaviest (longest) edge on the path. For example, the bottleneck length of the path (e,a,d,g,c) in the graph above is 17 because the heaviest (longest) edge on the path is (a,d). An instance of the BOTTLENECK SHORTEST PATH PROBLEM is a connected weighted graph $G=(V,E)$ with $w: E \to \Re^+$ and a pair of vertices $\{\sigma,\tau\} \subseteq V$. The output should be a path from σ to τ of minimal bottleneck length over all

paths from σ to τ in *G*. For the graph above with source $\sigma = e$ and terminus $\tau = b$ the path returned should be (e, f, a, b) of bottleneck length 5. Find an algorithm to solve this problem in polynomial time.

CS2223
HW#6 SOLUTIONS

1. Given $G=(V,E)$, add a new source σ not already in V. Add an edge of weight 0 from σ to each $v \in \Sigma$. Given the array D returned by one invocation of DIJKSTRA's algorithm, just remove the entry $D[\sigma]$ and return D.

2. The bottleneck shortest path from σ to τ in connected graph G is the unique shortest path from σ to τ in the minimum spanning tree of G. So a solution would be to first compute the minimum spanning tree of G using Kruskal's Algorithm, and then compute the shortest path from σ to τ in the minimum spanning tree of G using Dijkstra's Algorithm, although one could stop Kruskal's Algorithm as soon as σ and τ belong to the same component.