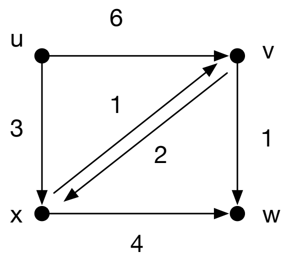
COMP 3711 – Spring 2019
Tutorial 9

1. Execute Dijkstra's algorithm on the following digraph, where u is the source vertex.



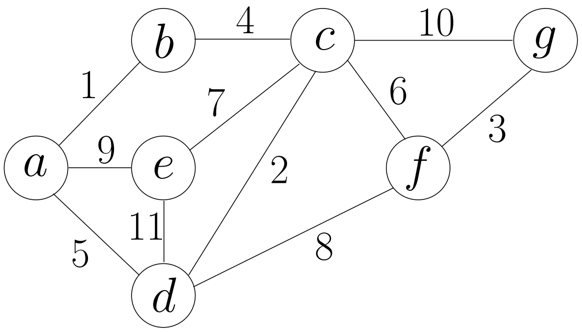
You need to indicate only the following:

(a) the order in which the vertices are removed from the priority queue.

(b) the final distance values $d[]$ for each vertex.

(c) the different distance values $d[]$ assigned to vertex w, as the algorithm executes.

2. Prim's minimum spanning tree algorithm and Dijkstra's shortest path algorithm are very similar, but with crucial differences. Run both algorithms on the following graph, and show the partial MST / shortest path tree after every new edge has been added. The starting vertex for both algorithms is "a".



3. Suppose that instead of using a heap to store the tentative vertex distances, Dijkstra's algorithm just kept an array in which it stored each vertex's tentative distance.

It then finds the next vertex by running through the entire array and choosing the vertex with lowest tentative distance.

What would the algorithm's running time be?

Is this better than our implementation for some graphs?

4. Extra Problem.

   Suppose that we are given a cable network of n sites connected by duplex communication channels. Unfortunately, the communication channels are not perfect.
   The channel between sites u annd v is known to fail with (given) probability $f(u, v)$

   The probabilities of failure for different channels are known to be mutually independent events.

   One of the n sites is the central station and your problem is to compute the most reliable paths from the central station to all other sites (i.e., the paths of lowest failure probabilities from the central station to all other sites).

   Design an algorithm for solving this problem, justify its correctness, and analyze its time and space complexities.