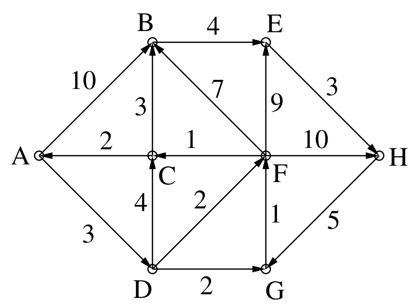
Algorithms

Judit Csima
csima@cs.bme.hu

Dijkstra's algorithm

1. Use Dijkstra's algorithm to determine the length of the shortest paths from A to any other vertices in the graph below. In each step indicate the edge with which we add the new vertex to set X and indicate the correct distances as well.
   How can we find the shortest path from A to E?



2. We run Dijkstra's algorithm for a directed graph G. We add vertices to set X in the following order: A, B, C, F, D, E, the computed distances are: $d(A) = 0, d(B) = 2, d(C) = 5, d(F) = 6, d(D) = 6, d(E) = 7$, and the edges with which we increase the set X are: (A, B), (B, C), (C, F), (C, D), and (D, E).
   Determine all the edges (and edge-weights) of G which can be reconstructed from the given data.

3. A directed, edge-weighted graph G is given by its adjacency list, and a vertex s is marked as source. All edge-weights are non-negative except one and there are no negative cycle in the graph. Design an algorithm to find the length of the shortest path from s to all other vertices, the running time should be $O(m \log n)$.

4. Let G be a directed, edge-weighted graph, given by an adjacency list. Some vertices of the graph are *important*, the distance of an important vertex v from another important vertex u is the smallest possible length of such a path from u to v which doesn't contain important vertices (except u and v).
   Design an algorithm to compute the distance between any two important vertices, the running time should be $O(f \cdot m \cdot \log n)$, where f is the number of important vertices.