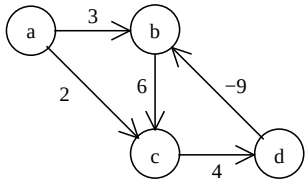
Data Structures, exercises week 12, 26-30.4

1. (a) Dijkstra's algorithm presumes, that in the graph there are no edges, that would have a negative weight. Give an example of a situation, where Dijkstra's algorithm gives a false solution, because a negatively weighted edge exists.

   (b) You can avoid the problem by adding an appropriate constant value to each weight of the edge, so that every weight is positive. Thus we have a graph, which is suitable for the Dijkstra's algorithm. Give an example, which demonstrates that this strategy, however, does not work.

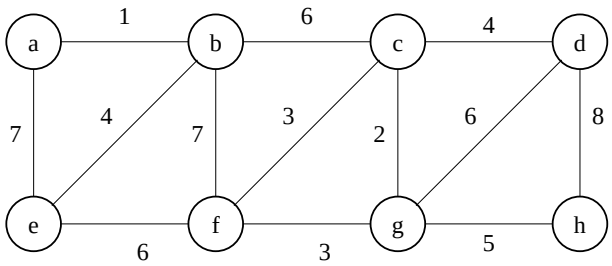
2. Simulate the Floyd-Warshall algorithm in the case of the following graph:



   Additional challenge: Extend the Floyd-Warshall algorithm so, that besides finding out the shortest distances between nodes, the algorithm also finds the shortest paths between nodes. Demonstrate how you find the shortest paths in the graph above.

3. Form the smallest spanning tree to the following graph by simulating Kruskal algorithm. Assume that edges with identical weight are handled in alphabetical order.

   What other results would be possible, if edges with identical weight would be handled in some other order?



4. You are given a graph whose vertices present all airports of the world. The edges present all direct flights between two airports. The weight of each edge is the probability that the luggage of a passenger gets lost during the corresponding flight (this is calculated based on the statistics gathered by IATA). Your task is to plan a route from Helsinki to Addis Abeba where the probability of your luggage getting lost is as low as possible. How can you use Dijkstra's algorithm to solve this problem?

   Hint: If the probability to loose luggage between Helsinki and London is p_1, between London-Mumbai p_2 and between Mumbai-Chennai p_3, then the probability that the luggage does not get lost between Helsinki-London-Mumbai-Chennai is $(1-p_1)(1-p_2)(1-p_3)$ and probability to luggage getting lost is $1-(1-p_1)(1-p_2)(1-p_3)$.

5. A railroad network is given as a graph $G = (V, E)$, where the vertices represent stations and edges represent direct rail connections. For each rail connection $(u, v) \in E$, there is a limit $w(u, v)$ for how heavy shipments the connection can safely handle. Give an algorithm to find the largest weight W such that from any starting station u, a shipment of weight W can be routed safely to any destination v.

6. Please help us further develop the course by filling in the feedback form found at

   `https://ilmo.cs.helsinki.fi/kurssit/servlet/Valinta?kieli=en`

7. **Bonus question, which replaces one other question from this or earlier exercises.**

   Program is given as input a table telling motorway distances. Here is part of the table:

```
            Pori        Tampere     Helsinki    Jyvaskyla   Lahti       //
Pori        0           114         242         ---         ---         //
Tampere     114         0           174         151         127         //
Helsinki    242         174         0           ---         106         //
Jyvaskyla   ---         151         ---         0           ---         //
Lahti       ---         127         106         ---         0           //
Heinola     ---         ---         ---         136         30          //
Mikkeli     ---         ---         ---         113         ---         //
Turku       138         ---         166         ---         ---         //
```

   The whole table is in here:
   `www.cs.helsinki.fi/u/mluukkai/tirak2010/eta.txt`

   The program prints out the distance table between every town, which is partly like this:

```
           Pori        Tampere     Helsinki    Jyvaskyla   Lahti       //
Pori        0           114         242         265         241         //
Tampere     114         0           174         151         127         //
Helsinki    242         174         0           272         106         //
Jyvaskyla   265         151         272         0           166         //
Lahti       241         127         106         166         0           //
Heinola     271         157         136         136         30          //
Mikkeli     362         248         227         113         121         //
Turku       138         252         166         403         272         //
```

   Implement the program with any programming language.

8. **Bonus question, which replaces one other question from this or earlier exercises.**

   Extend the previous program so that it also prints out the shortest route between certain towns:

```
Give departure town: Helsinki
Give arrival town:   Mikkeli

Helsinki  --> 106 km
Lahti     --> 30 km
Heinola   --> 91 km
Mikkeli

total length of the route is 227 km
```