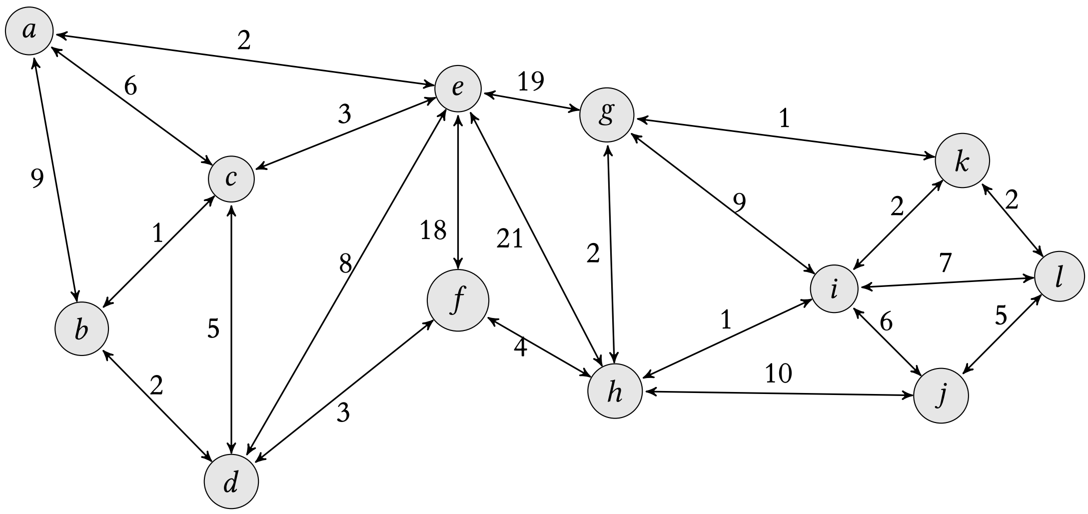
Übungsblatt 6.

Bearbeiten bis: **Montag, 14.04.2025**

Aufgabe 1 (Dijkstra). Gegeben sei der folgende Digraph $G = (V, E)$, welcher nur aus gewichteten Doppelkanten besteht:



(a) Berechnen Sie mittels des Dijkstra–Algorithmus schrittweise die Abstände von a zu allen anderen Knoten.

(b) Geben Sie einen kürzesten Weg von a nach l an.

Aufgabe 2 (Ein Spielbrett | 4 Punkte). Gegeben sei das folgende Spielbrett, das aus mit Zahlen markierten Feldern besteht.

4	6	2	12
9	3	7	4
6	7	12	5

In einem Spielzug dürfen Sie eine Figur um genau *ein* Feld entweder in senkrechter oder in waagerechter Richtung verschieben. Sind x und y die Zahlen der Felder eines Spielzuges, so ist der Aufwand für diesen Spielzug gleich $(x - y)^2$.

Wie müssen Sie Ihre Figur bewegen, so dass Sie mit möglichst geringem Gesamtaufwand (Aufwandssumme) von der linken oberen Ecke in die rechte untere Ecke gelangen? Berechnen Sie mithilfe der Graphentheorie eine Lösung.

Aufgabe 3 (Implementierung von Dijkstra | 4 Punkte). Implementieren Sie den Dijkstra-Algorithmus zur Berechnung der Einzelquelle-kürzeste-Wege für Digraphen mit nicht-negative Gewichten. Hierfür hat die Matrix der Gewichte $\boldsymbol{\omega}$ dieselbe Struktur wie die Matrix der Nachbarschaftsknoten $\mathbf{A}$ aus den letzten Implementierungsaufgaben, wobei sie im Eintrag $\boldsymbol{\omega}[i][j]$ das Gewicht der Kante $(j, \mathbf{A}[i][j])$ enthält.

Algorithmus dijkstra (A, ω, s)

Input: Matrix der Nachbarschaftsknoten A, Gewichte ω, Anfangsknoten s
Output: $\boldsymbol{l}(v)$ Kosten des $s-v$ Weges, $\mathbf{p}(v)$ Vorgänger von v

$\boldsymbol{l}(v) = \infty, v = 1, \dots, n$
$\boldsymbol{l}(s) = 0$
$q = [1, 2, \dots, n]$
while $q \neq [\ ]$ **do**
  choose $u \in q$ mit $\boldsymbol{l}(u)$ minimal
  $q = q \setminus \{u\}$
  for $v \in q$ **do**
    if $\exists w$ mit $v = A[w][u]$ **then**
      if $\boldsymbol{l}(v) > \boldsymbol{l}(u) + \omega[w][u]$ **then**
        $\boldsymbol{l}(v) = \boldsymbol{l}(u) + \omega[w][u]$
        $\mathrm{p}(v) = u$
      end if
    end if
  end for
end while

Testen Sie die Implementierung auf dem Digraphen aus Aufgabe 1.