Prof. Dr. Ir. Joost-Pieter Katoen

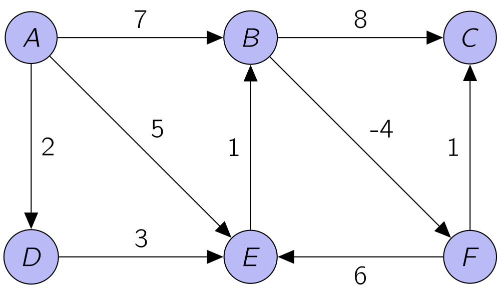
Christian Dehnert, Jonathan Heinen, Thomas Ströder, Sabrina von Styp

Hinweise:

- Die Übungsblätter sind in Gruppen von je 3 Studierenden aus der gleichen Kleingruppenübung zu bearbeiten.
- Die Lösungen müssen bis Montag, den 2. Juli um 11:00 Uhr in den entsprechenden Übungskasten eingeworfen werden. Sie finden die Kästen am Eingang Halifaxstr. des Informatikzentrums (Ahornstr. 55).
- Namen und Matrikelnummern der Studenten sowie die Nummer der Übungsgruppe sind auf jedes Blatt der Abgabe zu schreiben. Heften bzw. tackern Sie die Blätter!

Aufgabe 1 (Bellman-Ford Algorithmus): (5 + 6 + 3 = 14 Punkte)

a) Bestimmen Sie für den Startknoten A die Längen der kürzesten Wege zu jedem Knoten im unten gegebenen Graphen G_1 mit Hilfe des Bellman-Ford Algorithmus. Dabei sei die Reihenfolge der Knoten gegeben durch die alphabetische Sortierung $(A, B, C, \ldots)$. Geben Sie für jeden Knoten K die Kosten an, die dem Knoten während der Berechnung zugewiesen werden. Z.B. im Format $K : \infty, 10, 7, 6, 3$. Geben Sie außerdem an, ob der Algorithmus einen Zyklus mit negativem Gewicht erkennt.



b) Passen Sie die Implementierung von Bellman-Ford, die Sie in der Vorlesung kennengelernt haben (Vorlesung 17, Folie 13), so an, dass der kürzeste Pfad zwischen zwei Knoten i und j ausgegeben (nicht zurückgegeben!) wird. Sie dürfen davon ausgehen, dass der übergebene Graph keine negativen Zykel besitzt.

Ihre Funktion soll die folgende Signatur haben:

```
void bellFord(List adjLst[n], int n, int i, int j)
```

Zur Ausgabe können Sie annehmen, dass eine Funktion `ausgabe` mit folgender Signatur existiert:

```
void ausgabe(String text)
```

Außerdem können Sie annehmen, dass `int`-Werte automatisch nach `String` konvertiert werden können.

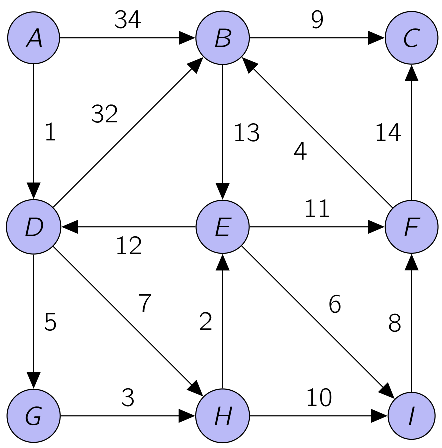
c) Einem Studenten gefällt die Laufzeit des Bellman-Ford Algorithmus nicht und er schlägt das folgende alternative Verfahren vor, um Zyklen mit negativem Gesamtgewicht zu erkennen:

Wir benutzen Sharirs Algorithmus, um alle starken Zusammenhangskomponenten zu finden. Dies kostet $O(|V| + |E|)$. Da $|E| \in O(|V|^2)$ sind die Kosten damit in $O(|V| + |V|^2) = O(|V|^2)$. Für jede starke Zusammenhangskomponente berechnen wir dann die Summe der Gewichte ihrer Kanten. Dies kostet insgesamt $O(|E|)$ und ist also wiederum in $O(|V|^2)$. Ist eine dieser Summen negativ, geben wir TRUE aus, sonst FALSE.

Wo liegt der Fehler, den der Student begangen hat?

Aufgabe 2 (Dijkstra Algorithmus): (5 + 4 + 4 + 3 = 16 Punkte)

a) Gegeben sei der folgende Graph G_2:



Ermitteln Sie mit Hilfe des Dijkstra-Algorithmus die kürzesten Wege von Startknoten A zu allen anderen Knoten des Graphen. Verwenden Sie dazu eine Tabelle der folgenden Form. Notieren Sie für jeden Rechenschritt den aktuell gewählten Knoten zur Verbesserung der Wege und die Länge der bis zu diesem Zeitpunkt möglichen kürzesten Wege für jeden noch nicht abgeschlossenen Knoten ($D[\ldots]$). Streichen Sie Felder der Tabelle, die nicht mehr benötigt werden, durch (dies geschieht, sobald ein Knoten als Baum-Knoten klassifiziert wurde).

Schritt	0	1	2	3	4	5	6	7	8
Knoten									
$D[A]$									
$D[B]$									
$D[C]$									
$D[D]$									
$D[E]$									
$D[F]$									
$D[G]$									
$D[H]$									
$D[I]$									

b) Beweisen Sie die Korrektheit des Dijkstra Algorithmus, d.h. zeigen Sie, dass dieser Algorithmus für einen Graphen G den kürzesten Abstand vom Startknoten s zu jedem anderen von s erreichbaren Knoten in G berechnet. Sie dürfen dazu das Theorem aus Vorlesung 17, Folie 19 nutzen.

c) Beweisen oder widerlegen Sie die folgenden Aussagen für einen gewichteten, zusammenhängenden, ungerichteten Graphen G:

i) Der vom Dijkstra Algorithmus berechnete SSSP-Baum ist ein Spannbaum.

ii) Der vom Dijkstra Algorithmus berechnete SSSP-Baum ist ein minimaler Spannbaum.

d) Arbeitet der Dijkstra Algorithmus immer korrekt, wenn man ihn auf einen Graphen mit negativen Gewichten anwendet, der keine Zyklen mit negativem Gesamtgewicht enthält? Begründen Sie Ihre Antwort!