Universität Stuttgart

Institut für Visualisierung und Interaktive Systeme

Übungsblatt 7

Datenstrukturen und Algorithmen (SS 2018)

Abgabe: Mittwoch, 20.06.2018, 23:55 Uhr — Besprechung: ab Montag, 25.06.2018

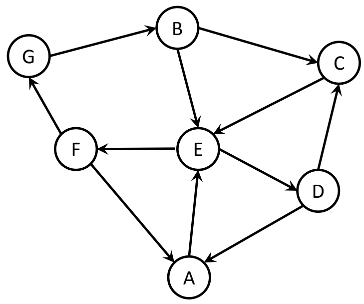
Bitte lösen Sie die Übungsaufgabe in **Gruppen von 3 Studenten** und wählen EINEN Studenten aus, welcher die Lösung im ILIAS als **PDF** als **Gruppenabgabe** (unter Angabe aller Gruppenmitglieder) einstellt. Bitte erstellen Sie dazu ein **Titelblatt**, welches die Namen der Studenten, die Matrikelnummern, und die E-Mail-Adressen enthält.

Die Aufgaben mit Implementierung sind mit Impl gekennzeichnet. Das entsprechende Eclipse-Projekt kann im ILIAS heruntergeladen werden. Bitte beachten Sie die Hinweise zu den Implementierungsaufgaben, die im ILIAS verfügbar sind.[1]

Dieses Übungsblatt beinhaltet 4 Aufgaben mit einer Gesamtzahl von 30 Punkten.

Aufgabe 1 Breitensuche *[Punkte: 5]*

Gegeben sei der Graph $\mathcal{G}_1$:



Wie bereits auf dem letzten Übungsblatt (Aufgabe 2) für die Tiefensuche, können Knoten eingefärbt werden, um ihren Status während der Breitensuche zu repräsentieren. In dieser Aufgabe sollen Nachfolger eines Knotens in alphabetischer Reihenfolge abgearbeitet werden; wenn z.B. der aktuelle Knoten D ist, wird Knoten A vor Knoten C abgearbeitet.

- Geben Sie die Reihenfolge an, in der die Knoten bei Ausführung der Breitensuche schwarz gefärbt werden, für den Fall, dass der **Startknoten B** ist.
- Geben Sie die Reihenfolge an, in der die Knoten bei Ausführung der Breitensuche schwarz gefärbt werden, für den Fall, dass der **Startknoten E** ist.

[1] https://ilias3.uni-stuttgart.de/goto_Uni_Stuttgart_crs_1432415.html

Aufgabe 2 Topologische Sortierung *[Punkte: 10]*

(a) *(2 Punkte)* Erläutern Sie was eine topologische Sortierung mit einem Graphen G macht.

(b) *(1 Punkt)* Spielen die Kantengewichte bei der topologischen Sortierung eine Rolle?

(c) *(1 Punkt)* Ist die topologische Sortierung eindeutig, d.h. gibt es für jeden beliebigen Graphen G maximal eine topologische Sortierung? Begründen Sie Ihre Antwort.

(d) *(4 Punkte)* Gegeben sei ein gerichteter Graph $\mathcal{G}_2 = (V, E)$ mit $V = \{A, B, C, D, E, F, G\}$ und $E = \{(A, C), (B, A), (B, D), (D, E), (D, F), (E, A), (F, A), (F, C), (G, B), (G, D)\}$.

- Zeichnen Sie diesen Graphen auf.
- Führen sie anschließend die topologische Sortierung, gemäß dem in der Vorlesung vorgestellten Schema/Algorithmus, darauf aus. Dokumentieren Sie Ihre einzelnen Schritte und geben Sie anschließend die topologische Sortierung Ihres Graphen an.

(e) *(2 Punkte)* Gibt es Graphen, auf denen eine topologische Sortierung nicht möglich ist? Bitte begründen Sie Ihre Antwort. Falls Ihre Antwort JA lautet, so geben Sie bitte, zusätzlich zur Begründung, ein graphisches Beispiel dafür an. Falls Ihre Antwort NEIN lautet, so geben Sie bitte eine Begründung ohne graphisches Beispiel an.

Aufgabe 3 Impl Bellman Ford *[Punkte: 10]*

Implementieren Sie in der Klasse `ShortestPath` den Bellman-Ford-Algorithmus zur Berechnung der kürzesten Pfade in einem gerichteten Graphen mit gewichteten Kanten und negativen Kantenkosten.

Die Klassen `WeightedGraph` für den Graphen, die Klasse `Edge` für eine Kante und die Klasse `ShortestPath` sind bereits vorgegeben. In der Klasse `ShortestPath` ist bereits ein Konstruktor vorhanden, der die Methode `bellmanFord(graph, startVertex)` aufruft. Diese Methode führt den Bellman-Ford-Algorithmus auf dem Graphen `graph` ausgehend vom Startknoten `startVertex` aus. Es sollen nur die Methoden aus dem Interface `IShortestPath` implementiert werden. Die Distanzen der einzelnen Kanten werden als `double` gespeichert. Wenn kein Pfad zu einem Knoten existiert, dann soll die Distanz den Wert `Double.POSITIVE_INFINITY` haben.

(a) *(5 Punkte)* Implementieren Sie die Methode `bellmanFord(graph, startVertex)`. Die Methode `distanceTo(destination)` soll die Distanz zum Zielknoten `destination` zurückgeben. Falls ein negativer Zyklus vorhanden ist, soll die Methode `distanceTo(destination)` eine `IllegalStateException` werfen.

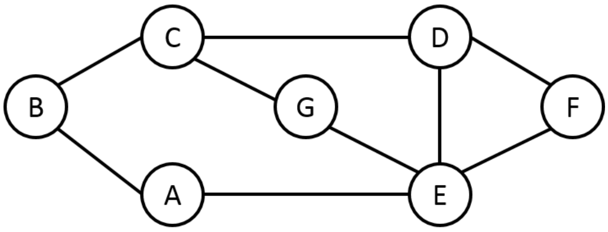
(b) *(1 Punkt)* Implementieren Sie die Methode `existsPathTo(destination)` zur Überprüfung, ob im Graph `graph` ein Pfad vom Startknoten `startVertex` zum Zielknoten `destination` existiert.

(c) *(1 Punkt)* Implementieren Sie die Methode `hasNegativeCycle()` zur Überprüfung, ob im Graph `graph` vom Startknoten `startVertex` aus ein negativer Zyklus im Graph vorhanden ist.

(d) *(3 Punkte)* Implementieren Sie die Methode `pathTo(destination)`, mit der es möglich ist, über den Pfad zum Zielknoten zu iterieren. Beachten Sie, dass der Iterator den Pfad rückwärts durchlaufen soll, also vom Zielknoten `destination` zum Startknoten `startVertex`. Der Startknoten `startVertex` soll vom Iterator nicht mehr erfasst werden. Falls ein negativer Zyklus vorhanden ist, soll die Methode `pathTo(destination)` eine `IllegalStateException` werfen.

Aufgabe 4 Euler, Hamilton und kürzeste Wege *[Punkte: 5]*

(a) Gegeben ist der folgende Graph $\mathcal{G}_3$.



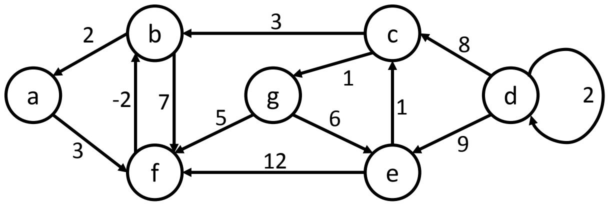
Beantworten Sie die folgenden Fragen. Lautet Ihre Antwort JA, so geben Sie bitte ein Beispiel an. Lautet Ihre Antwort NEIN, so begründen Sie Ihre Antwort. Eine Antwort ohne Beispiel oder Begründung wird mit Null Punkten bewertet.

i. *(1 Punkt)* Gibt es in dem gegebenen Graph $\mathcal{G}_3$ einen *Eulerschen Weg*?

ii. *(1 Punkt)* Gibt es in dem gegebenenen Graph $\mathcal{G}_3$ einen *Eulerschen Kreis*?

iii. *(1 Punkt)* Gibt es in dem gegebenen Graph $\mathcal{G}_3$ einen *Hamiltonschen Weg*?

(b) *(2 Punkte)* Gegeben ist der gewichtete gerichtete Graph $\mathcal{G}_4$.



Welchen der folgenden *Algorithmen* kann man verwenden, um die *kürzesten Wege* vom Knoten a zu allen anderen Knoten zu bestimmen? Begründen Sie Ihre Antwort. Eine Antwort ohne Begründung wird mit Null Punkten bewertet.

- A*-Algorithmus
- Bellman-Ford-Algorithmus
- Dijkstra-Algorithmus