Prof. Dr. V. Linnemann
Stefan Werner
Universität zu Lübeck
Institut für Informationssysteme

Lübeck, den 13. Juni 2013

Algorithmen und Datenstrukturen

Sommersemester 2013

11. Übungsblatt

Aufgabe 1: Java - Stadtplan

Im Folgenden soll ein Stadtplan durch einen Graphen repräsentiert werden, um im Anschluss verschiedene Funktionen darauf ausführen zu können.

Ein Stadtplan besteht aus Orten (Knoten), die durch Straßen (Kanten) miteinander verbunden sein können. Orte und Straßen besitzen einen Namen. Straßen können in beide (ungerichtet) oder nur eine Richtung (gerichtet) befahrbar sein. Zusätzlich wird für eine Straße die Länge als gewichtender Faktor berücksichtigt. Somit wird der Stadtplan durch einen gerichteten und gewichteten Graphen beschrieben.

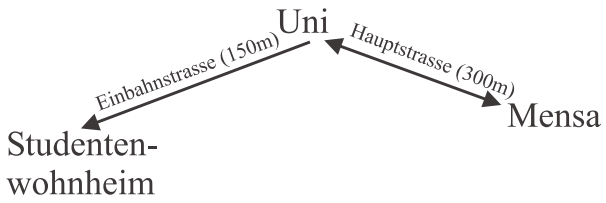
Die vollständigen Klassen `Ort` und `Strasse` sowie die unvollständige Klasse `Stadtplan` stehen auf der Webseite zur Übung zum Download bereit. Letzteres stellt bereits die Methode *einlesen()* zum Einlesen eines serialisierten Stadtplans bereit. Die Definition des Stadtplans unterliegt dabei folgendem Aufbau:

```
n
m
1;Knotenname_1
2;Knotenname_2
..
n;Knotenname_n
1;Kantenname_1;StartNr_1;ZielNr_1;Richtung_1;Gewicht_1
2;Kantenname_2;StartNr_2;ZielNr_2;Richtung_2;Gewicht_2
..
m;Kantenname_m;StartNr_m;ZielNr_m;Richtung_m;Gewicht_m
```

wobei n die Anzahl der Orte (Knoten) und m die Anzahl der Strassen (Kanten) ist. Die Richtung wird durch die Schlüsselwörter `gerichtet` und `ungerichtet` definiert. Das Gewicht ist eine ganze Zahl. Die StartNr und ZielNr bezieht sich auf die vorher definierten Knoten-

nummern. Die einzelnen Datenfelder sind immer durch Semikola (;) getrennt.

Als Beispiel wird folgender Stadtplan durch eine Datei beschrieben:




```
3
2
1;Uni
2;Studentenwohnheim
3;Mensa
1;Hauptstraße;1;3;ungerichtet;300
2;Einbahnstraße;1;2;gerichtet;150
```

Die Klasse `Stadtplan` soll nun erweitert werden. Nutzen Sie zum Testen den Stadtplan, der in der Datei *luebeck_city.txt* definiert ist.

a) Implementieren Sie eine Methode, die alle direkten Nachfolger eines Startortes zurückgibt. Ein direkter Nachfolger ist ein Zielort (Knoten), der über genau eine Straße (Kante) vom Startort erreichbar ist. Was sind die direkten Nachfolger des Ortes mit der Nummer 9? (2 Punkte)

b) Implementieren Sie eine Methode, die alle direkten Vorgänger eines Zielortes zurückgibt. Ein direkter Vorgänger ist analog zu a) ein Startort (Knoten), von dem aus der Zielort über genau eine Straße (Kante) erreichbar ist. Was sind die direkten Vorgänger des Ortes mit der Nummer 9? (2 Punkte)

c) Implementieren Sie eine Methode, die alle erreichbaren Orte eines Startortes zurückgibt (entspricht der reflexiven, transitiven Hülle eines Knotens). Welche Orte sind vom Ort mit der Nummer 9 erreichbar? (4 Punkte)

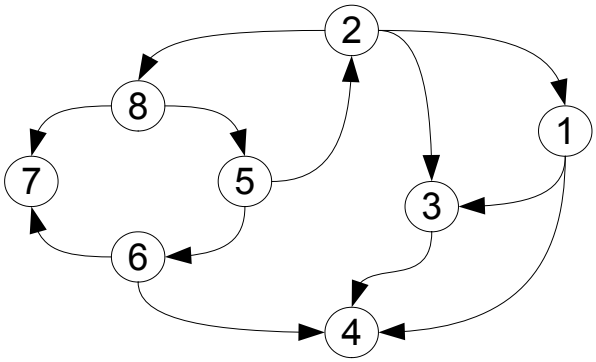
d) Implementieren Sie eine Methode, die zurückgibt, ob es einen Weg von einem Ort a zu einem Ort b gibt. Existiert ein Weg von Ort 9 zu Ort 49 bzw. von 10 zu 36? (2 Punkte)

e) Implementieren Sie eine Methode, die von einem Ort a aus eine **Tiefensuche** durch den Stadtplan durchführt. Dabei soll der Name eines Ortes beim erstmaligen Besuchen ausgegeben werden. Es ist **nicht** notwendig die Straßen (Kanten) in Baumpfeile, Vorwärtspfeile, etc. einzuordnen. Welche Ausgabe erzeugt Ihr Algorithmus, wenn die Tiefensuche beim Ort mit der Nummer 10 beginnt? (4 Punkte)

Testen Sie Ihre implementierten Methoden unbedingt mittels der `main`-Methode! Kommentieren Sie Ihren Quellcode ausführlich und senden Sie Ihre Lösung an Ihren Betreuer!

Aufgabe 2: Tiefensuche

Gegeben sei der folgende Graph:



Wenden Sie den Tiefensuche-Algorithmus (DFS) aus der Vorlesung auf Knoten 8 an und identifizieren Sie Baumpfeile (BP), Vorwärtspfeile(VP), Rückwärtspfeile (RP) und Seitwärtspfeile (SP). Wenn von einem Knoten mehrere andere Knoten erreichbar sind, soll derjenige mit der kleinsten Nummer gewählt werden.
Bestimmen Sie den *Depth-First-Begin-Index (DFBI)* und den *Depth-First-End-Index (DFEI)* für jeden Knoten.

Zeichnen Sie den DFS-Aufrufbaum. (11 Punkte)

Beachten Sie die Hinweise auf dem ersten Übungsblatt.

Abgabetermin: Donnerstag, den 20. Juni bis 10 Uhr im Institut für Informationssysteme, 2. OG im Informatik-Neubau (Raum 2.071, Küche neben Sekretariat)