Aufgabe 4: (7 Punkte)

(a) Zeichnen Sie die gerichteten Graphen, die durch folgende Daten definiert sind:

(i) Adjazenzmatrix

$$A = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

(ii) Adjazenzlisten

$A(1) = \{2\}$
$A(2) = \{3, 4\}$
$A(3) = \{4\}$
$A(4) = \emptyset$

(iii) Inzidenzmatrix

$$B = \begin{pmatrix} 1 & -1 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

Geben Sie für jeden Graphen an, ob er zyklisch, zusammenhängend bzw. stark zusammenhängend ist.

(b) Es sei A die binäre Adjazenzmatrix eines ungerichteten Graphen $G = (V, E)$ und $B := A^2$. Zeigen Sie, dass für jeden Knoten $i \in V$ das Diagonalelement b_{ii} gleich dem Grad von i ist.

(c) Sei $\tilde{G} = (V, \tilde{E})$ der transitive Abschluss eines gerichteten Graphen $G = (V, E)$ und $\tilde{A}$ die Adjazenzmatrix von $\tilde{G}$. Sei weiterhin $\tilde{B} := \tilde{A}^2$. Zeigen Sie, dass für jeden Knoten $i \in V$ das Diagonalelement $\tilde{b}_{ii}$ gleich der Anzahl der Knoten in der starken Zusammenhangskomponente von i ist (i nicht mitgezählt).

Aufgabe 5: (8 Punkte)

Gegeben sei ein ungerichteter Graph $G = (V, E)$ mit $V = \{1, \ldots, n\}$, der auf zwei Arten repräsentiert wird:

- durch seine (symmetrische) Adjazenzmatrix, gespeichert in einem Feld der Länge $n(n-1)/2$,
- durch Adjazenzlisten, wobei zu jedem Knoten $i \in V$ nur diejenigen Nachbarknoten j mit $j > i$ gespeichert sind.

Geben Sie Prozeduren (ihre Funktionsweise) und ihre Komplexität an, die für diese beiden Darstellungen

(a) prüfen, ob zwei Knoten benachbart sind,

(b) alle Nachbarn eines Knotens bestimmen.

Programmieraufgabe P1: (15 Punkte)

(a) Implementieren Sie Java-Klassen für gerichtete und ungerichtete Graphen. Die Graphen sollen zum Einen durch ihre Adjazenzmatrix, zum Anderen durch Adjazenzlisten repräsentiert werden. Das Hinzufügen, Löschen und Ausgeben von Kanten und Knoten soll möglich sein.

Die Klassen sollen die Interfaces `Graph` und `RenderableGraph` implemententieren. Letzteres Interface finden Sie im jar-File `RenderGraph.jar`. Beide Dateien (`RenderGraph.jar` und `Graph.java`) sind bei `studIP` hochgeladen.

Schreiben Sie Methoden, die beide Darstellungen ineinander transformieren. Testen Sie Ihre Methoden an einigen zufällig generierten Graphen.

(b) Bei `studIP` stehen in der Archivdatei `P1.zip` sowohl gerichtete (in den Dateien `ga_dir_01.gra` bis `ga_dir_05.gra`) als auch ungerichtete Graphen (Dateien `ga_undir_01.gra` bis `ga_undir_05.gra`) zur Verfügung. Das verwendete Dateiformat ist in der Text-Datei `dateiformat.txt` erläutert, die ebenfalls in `P1.zip` enthalten ist.

Lesen Sie die Graphen aus den angegebenen Dateien ein und visualisieren Sie die Graphen mit dem im jar-File `RenderGraph.jar` bereitgestellten Renderer. Senden Sie die entstandenen png-Files mit Ihrem Quellcode mit.

Eine javadoc-konforme Dokumentation wird bei allen Programmieraufgaben vorausgesetzt.

Diese Aufgabe muss am **8.5.2015** zusammen mit Blatt 3 abgegeben werden. Senden Sie den Quellcode an Ihren Tutor (mbultmann@uos.de) und geben Sie einen Ausdruck ab.