Übung 8: Graphen

Abgabetermin: 29.05.2018

Name: **Matrikelnummer:**

Gruppe: ☐ **G1** Di 10:15-11:00 ~~☐ **G2** Di 11:00-11:45~~ ☐ **G3** Di 10:15-11:00

Aufgabe	Punkte	gelöst	abzugeben schriftlich	abzugeben elektronisch	Korr.	Punkte
Aufgabe 1	8	☐	Simulation	Simulation	☐	
Aufgabe 2	16	☐	Java-Programm	Projekt Archiv	☐	

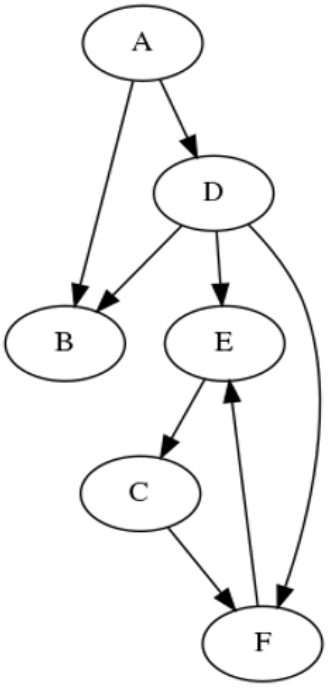
Aufgabe 1: Transitive Hülle mit dem Warshall-Algorithmus bestimmen (8 Punkte)

Berechnen Sie die transitive Hülle des Graphen mit dem Warshall-Algorithmus. Geben Sie alle Zwischenergebnisse als Adjazenzmatrix an. Markieren Sie neu hinzugekommene Kanten durch Einkreisen.

	A	B	C	D	E	F
A						
B						
C						
D						
E						
F						

Spalte A →

	A	B	C	D	E	F
A						
B						
C						
D						
E						
F						



Spalte B →

	A	B	C	D	E	F
A						
B						
C						
D						
E						
F						

Spalte C →

	A	B	C	D	E	F
A						
B						
C						
D						
E						
F						

Spalte D →

	A	B	C	D	E	F
A						
B						
C						
D						
E						
F						

Spalte E →

	A	B	C	D	E	F
A						
B						
C						
D						
E						
F						

Spalte F →

	A	B	C	D	E	F
A						
B						
C						
D						
E						
F						

Abzugeben ist: Simulation

Aufgabe 2: Implementierung des Warshall-Algorithmus (16 Punkte)

Implementieren Sie den Warshall-Algorithmus in Java. Erstellen Sie zuerst eine Adjazenzmatrix in der Methode *Warshall.graphToMatrix* mit den Kanten des Graphen. Verwenden Sie dafür ein *long[]*-Array. Die Einträge sind als Bit-Vektor zu verstehen. Zum Beispiel ist das 3. Bit im 5. Array-Eintrag gesetzt existiert eine Kante vom Knoten mit Index 5 zum Knoten mit Index 3. Verwenden Sie als Indizes in die Matrix die Position der Knoten in der *vertices*-Liste des Graph-Objekts. Bestimmen Sie dann mit dem Warshall-Algorithmus die Kanten der transitiven Hülle (*transitiveClosure*). Erstellen Sie danach als Ergebnis einen neuen Graphen der alle Kanten der transitiven Hülle enthält (*matrixToGraph*).

Abzugeben ist: Projekt Archiv

Implementierungshinweise:

- Verwenden Sie das Vorgabeprojekt **PI2_UE08.zip**.
- Fügen Sie Ihre Implementierung in den mit **TODO** markierten Teilen der Klassen *Warshall* ein.
- Berücksichtigen Sie, dass bei einer ungerichteten Kante zwischen A und B sowohl A direkt von B als auch B direkt von A erreichbar ist.
- Ignorieren Sie eventuell vorhandene Kantengewichte.
- Implementieren und benutzen Sie *isBitSet* und *setBit* in der Klasse *Warshall* um die Bit-Manipulationen auf der Matrix zu abstrahieren.
- Unabhängig davon ob der übergebene Graph gerichtet oder ungerichtet ist: Das Ergebnis von *transitiveClosure* soll ein gerichteter Graph sein (das macht Ihre Lösung einfacher).
- Die *indexOf*-Methode von *Graph.vertices* liefert den Index eines Vertex-Objekts.
    - Verwenden Sie *indexOf* nur wenn unbedingt notwendig. In vielen Fällen gibt es Alternativen mit besserem Laufzeitverhalten.
- Ändern sie **keine public Interfaces** vorgegebener Skeleton Klassen.
- Halten Sie sich an die **Codierungsrichtlinien** auf der Kurs Website.