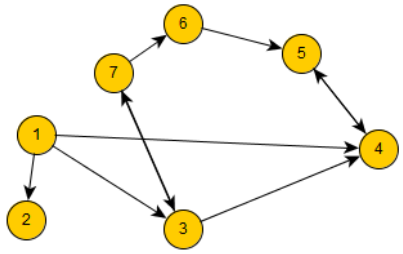
Lösungen zu denTrainingsaufgaben Graphen

Motivation

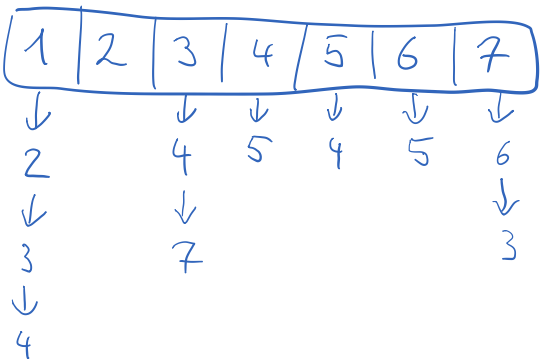
Diese Aufgaben ergänzen die Übungsaufgaben, um Ihnen die Möglichkeit zu zusätzlichem Training zu geben. Die Lösungen finden Sie in einem separaten Dokument.

1. Darstellung von Graphen

Stellen Sie den Graphen als Adjazenzmatrix, Kantentabelle und Adjazenzliste dar:



Adj-Liste:



Adj.-Matrix:

(T = True)

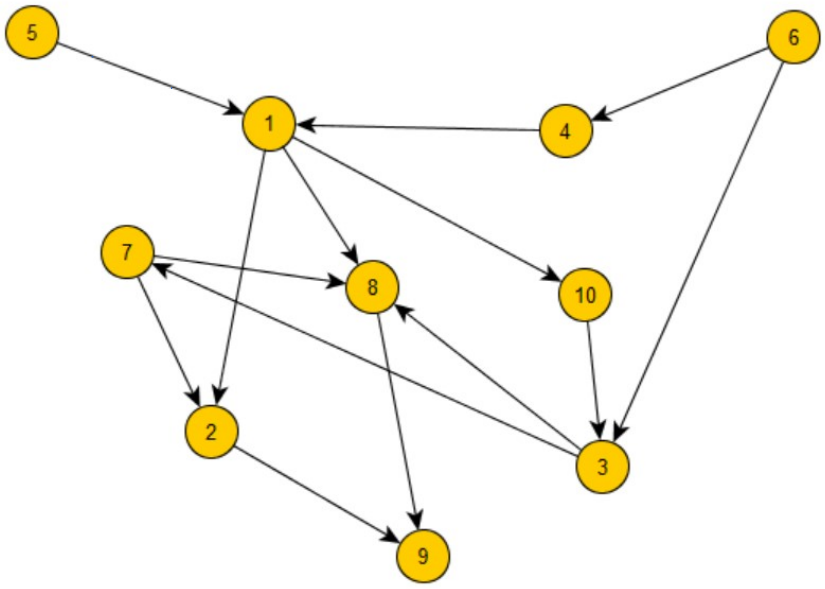
	1	2	3	4	5	6	7
1		T	T	T			
2							
3				T			T
4					T		
5				T			
6					T		
7			T			T	

Kantentabelle:

from	1	1	1	3	3	4	5	6	7	7
to	2	3	4	4	7	5	4	5	3	6

2. Topologisches Sortieren

Geben Sie mindestens eine mögliche Topologische Sortierung zum folgenden Graph an:



Geben Sie alle möglichen Topologischen Sortierungen an:

5, 6, 4, 1, 10, 3, 7, 8, 2, 9

5, 6, 4, 1, 10, 3, 7, 2, 8, 9

6, 5, 4, 1, 10, 3, 7, 8, 2, 9

6, 5, 4, 1, 10, 3, 7, 2, 8, 9

6, 4, 5, 1, 10, 3, 7, 8, 2, 9

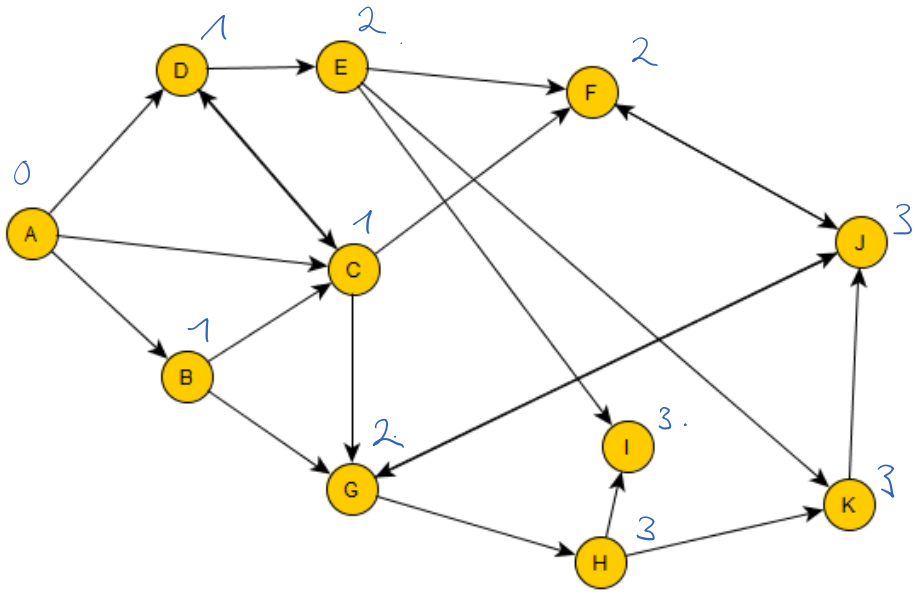
6, 4, 5, 1, 10, 3, 7, 2, 8, 9

Hinweis: Wenn sie noch weitere Trainingsmöglichkeiten wünschen, dann können Sie sich auch die Übung „7-4 – Implementierung Adjazenzlisten“ zur Hilfe nehmen. Sollte ihre Implementation noch nicht einwandfrei funktionieren, verwenden Sie die Lösung auf dem Netzlaufwerk. Wählen Sie zuerst „File -> New Graph -> directed, unweighted“. Zeichnen Sie dann im Grapheditor einen beliebigen Graphen, indem Sie Nodes und gerichtete Kanten einfügen. Berechnen dann von Hand die topologische Sortierung (oder besser alle, da es mehrere Lösungen geben kann) und vergleichen dann Ihre Lösung(en) mit der Ausgabe auf der Konsole.

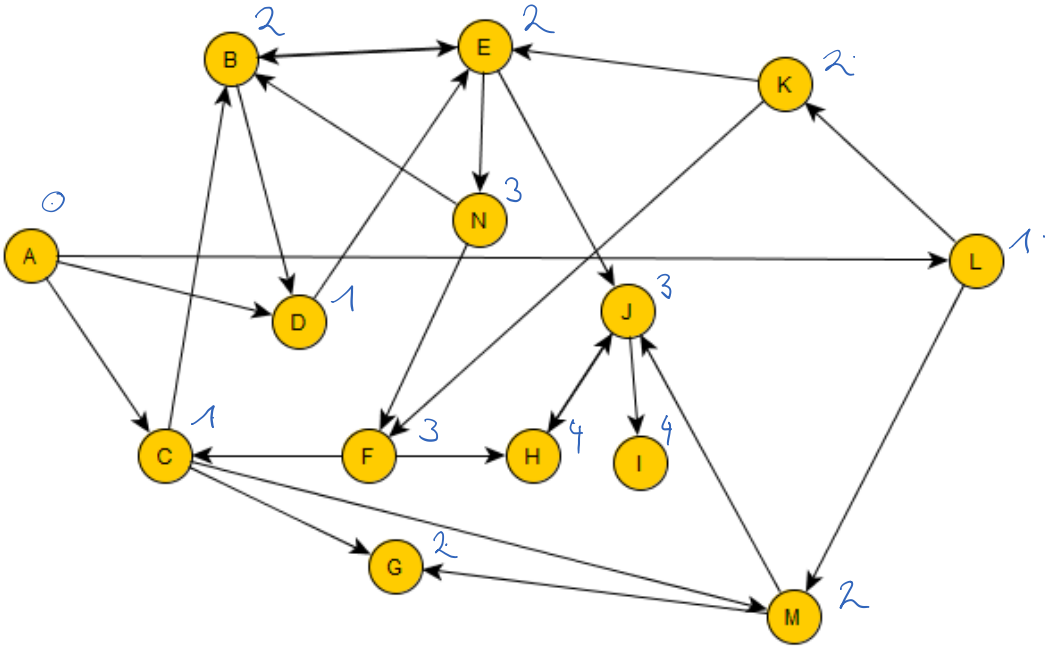
3. Kürzeste Wege (ungewichtete Graphen)

Berechnen Sie mit Hilfe der BFS für die Graphen unter a.) und b.) die kürzesten Wege vom Knoten A zu allen anderen Knoten. Tragen Sie die Distanzen direkt in die Graphen ein.

a.)

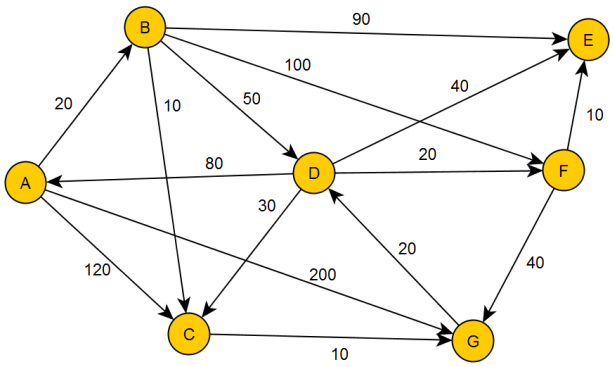


b.)

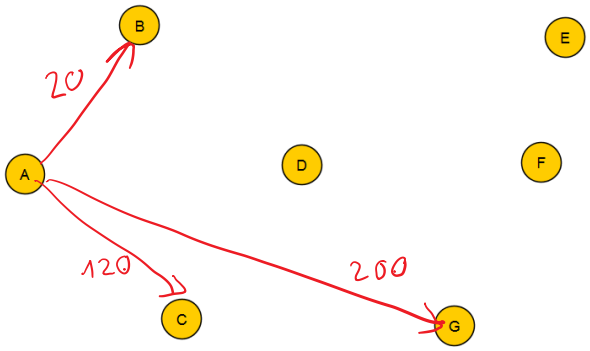


4. Kürzeste Wege mit Dijkstra (ungewichtete Graphen)

Berechnen Sie mit Hilfe des Dijkstra-Algorithmus die kürzesten Pfade vom Knoten A zu allen anderen Knoten. Verwenden Sie zur Hilfe untenstehende und bereits aus dem Unterricht bekannte Tabelle. Zur Erinnerung: Die Spalte „bekannt“ bedeutet, dass der kürzeste Pfad zum entsprechenden Knoten bereits bekannt ist.

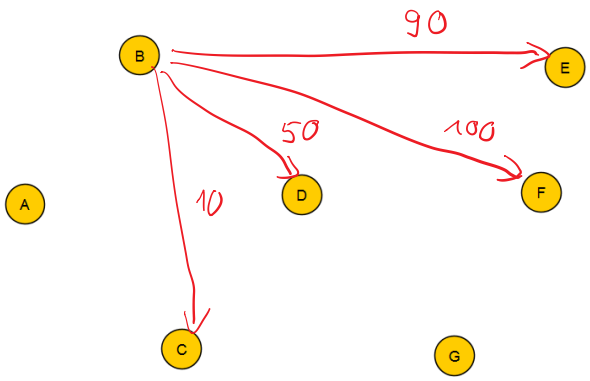


1. Schritt



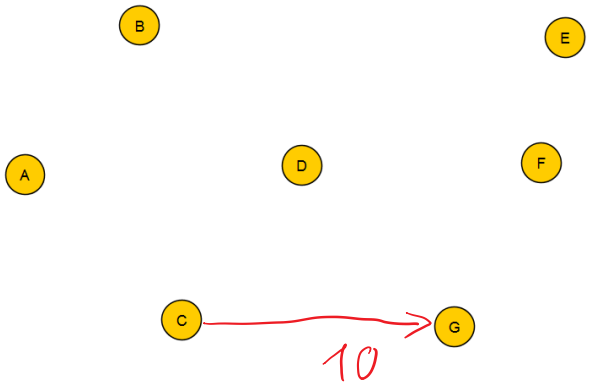
Knoten	bekannt	Distanz	Via
A	✓	0	-
B		20	A
C		120	A
D		∞	–
E		∞	–
F		∞	–
G		200	A

2. Schritt



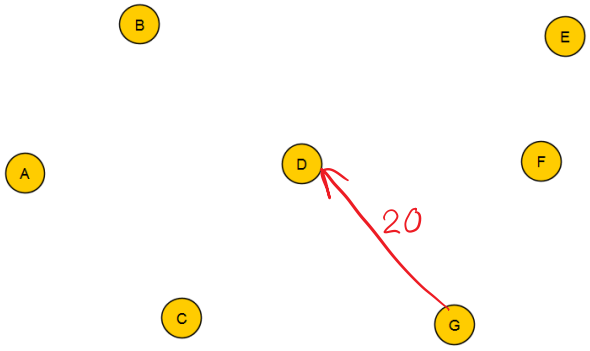
Knoten	bekannt	Distanz	Via
A	✓	0	-
B	✓	20	A
C		30	B
D		90	B
E		110	B
F		120	B
G		200	A

3. Schritt



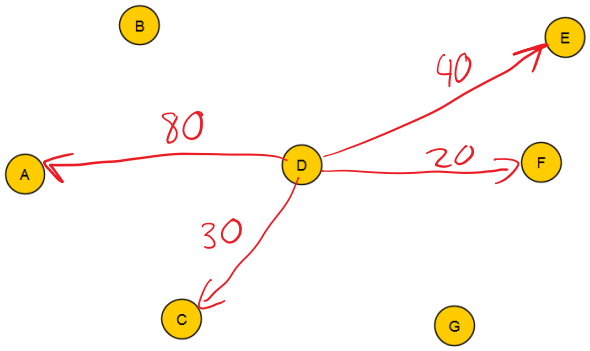
noten	bekannt	Distanz	Via
A	✓	0	-
B	✓	20	A
C	✓	30	B
D		40	B
E		110	B
F		120	B
G		40	C

4. Schritt



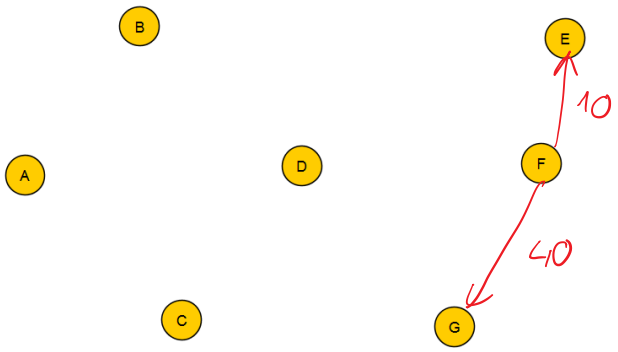
Knoten	bekannt	Distanz	Via
A	✓	0	-
B	✓	20	A
C	✓	30	B
D		60	G
E		110	B
F		120	B
G	✓	40	C

5. Schritt



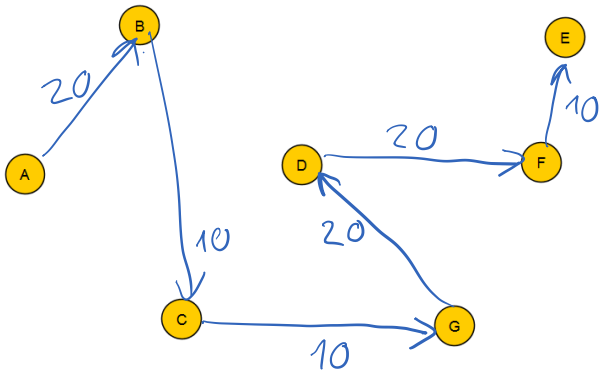
Knoten	bekannt	Distanz	Via
A	✓	0	-
B	✓	20	A
C	✓	30	B
D	✓	60	G
E		100	D
F		80	D
G	✓	40	C

6. Schritt



noten	bekannt	Distanz	Via
A	✓	0	-
B	✓	20	A
C	✓	30	B
D	✓	60	G
E	✓	90	F
F	✓	80	D
G	✓	40	C

7. Schritt



Knoten	bekannt	Distanz	Via
A	✓	0	-
B	✓	20	A
C	✓	30	B
D	✓	60	G
E	✓	90	F
F	✓	80	D
G	✓	40	C

Hinweis: Wenn sie noch weitere Trainingsmöglichkeiten wünschen, dann können Sie sich auch die Übung „7-7 Selbststudium Implementierung DFS und Dijkstra-Algorithmus“ zur Hilfe nehmen. Sollte ihre Implementation noch nicht einwandfrei funktionieren, verwenden Sie die Lösung auf dem Netzlaufwerk. Zeichnen Sie dann einen beliebigen Graphen, berechnen dann von Hand den kürzesten Pfad zwischen zwei Knoten. Im Anschluss wählen Sie im Graph-Editor „File -> New Graph -> directed, weighted“. Fügen Sie die Knoten und gewichteten Kanten ein. Wählen Sie im Anschluss den Selection-Mode „Selection“. Klicken Sie im Anschluss auf den Start- und Zielknoten und dann auf „Shortest Path“. Achtung: Alle Knoten und Kanten, die nicht zum kürzesten Pfad gehören, werden dann gelöscht.