Gruppe 1: 18.Nov., 8.30 Uhr; Gruppe 2: 19.Nov., Gruppe 3: 26.Nov., je 13.15 Uhr.

Abgabe der Übungsblätter vor der Vorlesung oder im Kasten, Foyer, 1. Etage.
Die Übungsblätter und Termine sind auch im WWW zugänglich:

http://www.informatik.uni-stuttgart.de/ifi/fk/uebungen/ws9899/ti2st.html

Hinweis: Die Teile des Skripts, die sich von dem durch die Fachschaft gedruckten Skript unterscheiden, liegen im Semesterapparat und in der Fachschaft zum Kopieren bereit. Dies betrifft z.B. den in Aufgabe 2 verwendeten Prim-Algorithmus, der in dem von der Fachschaft gedruckten Skript nicht enthalten ist.

Aufgabe 1: (2+2 Punkte)

Sei $G = (V, E)$ ein ungerichteter schlingenfreier Graph. Das Komplement $\overline{G}$ von G ist definiert als:

$$\overline{G} = (V, \{\{u, v\} \mid \{u, v\} \notin E \wedge u \neq v\})$$

Ein Graph G heißt selbstkomplementär, wenn G und $\overline{G}$ isomorph sind.

a) Geben Sie alle selbstkomplementären Graphen mit 4, 5 und 6 Knoten an (Hinweis: Machen Sie zunächst eine Fallunterscheidung, welchen Grad die einzelnen Knoten haben können; bedenken Sie, daß $(V, E \cup \overline{E})$ ein vollständiger Graph ist).

b) Beweisen Sie: Für alle endlichen Graphen $G = (V, E)$ (ohne Multikanten) gilt $\alpha(G) = \omega(\overline{G})$, wobei $\alpha(G)$ die Unabhängigkeitszahl und $\omega(G)$ die Cliquenzahl ist.

Aufgabe 2: (3+1 Punkte)

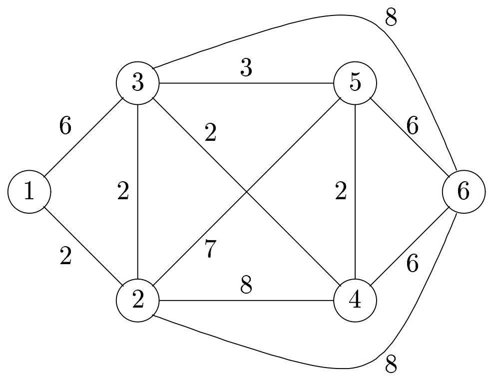
Der Algorithmus von Prim bestimmt einen minimal aufspannenden Baum eines ungerichteten ,zusammenhängenden, kantenmarkierten Graphen $G = (V, E, \delta)$ mit $\delta : E \to \mathbb{R}^+$, d.h. ein Teilgraph $G' = (V, E')$ von G mit G' ist Baum und $\sum_{e \in E'} \delta(e) \leq \sum_{\tilde{e} \in \tilde{E}} \delta(\tilde{e})$ für alle Bäume $\tilde{G} = (V, \tilde{E})$, die Teilgraph von G sind.

a) Seien nun in dem Graphen auch negative Kantengewichte erlaubt, also $\delta : E \to \mathbb{R}$. Schreiben Sie einen Algorithmus, der zu einem ungerichteten, zusammenhängenden, kantenmarkierten Graphen $G = (V, E, \delta)$ einen **maximal** aufspannenden Wald konstruiert, d.h. einer Menge von (Knoten-) disjunkten Bäumen $T_i = (V_i, E_i)$ mit $V = \bigcup_i V_i$ und $\sum_i \sum_{e \in T_i} \delta(e)$ maximal.

b) Schätzen Sie den Aufwand Ihres Algorithmus ab.

Aufgabe 3: (2+1 Punkte)

Gegeben Sei folgender ungerichteter Graph G



a) Bestimmen Sie die kürzesten Wege in G von Startknoten 1 zu allen übrigen Knoten mit Hilfe des Dijkstra-Algorithmus. Geben Sie für jeden Schritt an, welcher Knoten zur Baummenge aufgenommen wird und geben Sie den entstehenden Kürzeste-Wege-Baum an.

b) Arbeitet der Dijkstra-Algorithmus auch noch korrekt, wenn man negative Kantengewichte erlaubt? Begründen Sie Ihre Antwort und geben Sie ggf. ein Gegenbeispiel, bei dem der Dijkstra-Algorithmus ein falsches Ergebnis liefern würde.

Aufgabe 4: (3+1+1 Punkte)

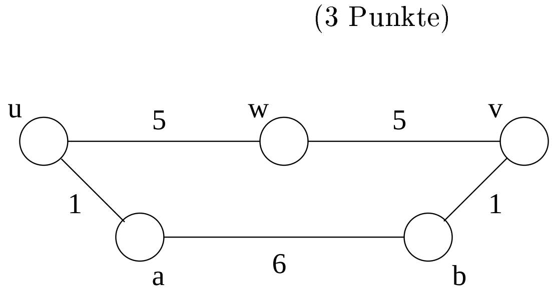
a) Entwickeln Sie einen Algorithmus, der zu einem gegebenen Graphen den längsten doppelpunktfreien Weg zwischen zwei einzugebenen Knoten u und v bestimmt.

b) Schätzen Sie ab, wieviele Schritte Ihr Algorithmus im schlimmsten Fall benötigen wird.

c) Versuchen Sie plausibel zu machen, warum das Problem, den längsten Weg zu finden, wesentlich schwerer ist als das Problem des kürzesten Weges.

Zusatzaufgabe: (3 Punkte)

In der Vorlesung wurde ein alternativer Algorithmus zur Bestimmung kürzester Wege von u nach v in ungerichteten Graphen vorgestellt: Starte den Dijkstra-Algorithmus gleichzeitig vom Startknoten u und Endknoten v. Sowie ein Knoten in beiden Baummengen aufgenommen ist,



wird daraus der kürzeste Weg zwischen u und v bestimmt. In dem Beispiel führt das zu einem falschen Ergebnis. Klären Sie den Fehler und formulieren Sie das zweiseitige Aufeinanderzusuchen neu.