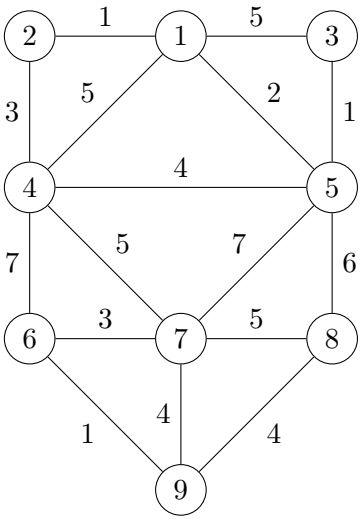
Beachte: Im Folgenden gilt die Konvention, dass jeder Algorithmus stets den Knoten bzw. die Kante mit der kleinsten beteiligten Knotennummer wählt, falls die Auswahl nicht eindeutig bestimmt ist.

Aufgabe 8.1 (4 Punkte)
Es sei der folgende Graph G gegeben:



Bestimme mit Hilfe des Algorithmus

a) Kruskal

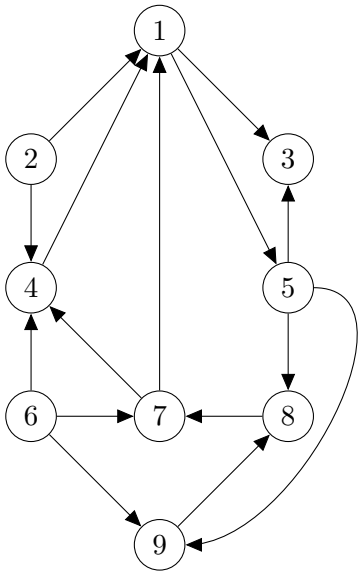
b) Delete-Reverse

aus der Vorlesung einen minimalen Spannbaum für G. Gib jeweils den resultierenden Spannbaum an. Gib zudem für a) die Reihenfolge, in der die Kanten für den Spannbaum ausgewählt werden, und die endgültige Form der Union-Find-Datenstruktur an und für b) die Reihenfolge, in der die Kanten aus G entfernt werden.

Hinweis: Nutze bei der Union-Find-Datenstruktur die beiden Tricks aus der Vorlesung! Starte mit den Bäumen $T_1, ..., T_9$. Wobei T_i zu Beginn nur das Element i enthält. Sollen Mengen i und j mit $i < j$ vereinigt werden, nutze $\text{UNION}(i, j, i)$. Sollten beide Mengen gleich groß sein, hänge j an i.

Aufgabe 8.2 (4 Punkte)
Es bezeichne C_k die Menge aller ungerichteten, gewichteten, zusammenhängenden Graphen, bei denen genau k Kanten dasselbe Kantengewicht q haben und die restlichen Kanten paarweise unterschiedliche Kantengewichte besitzen, die von q verschieden sind. Es bezeichne $\#MST(G)$ die Anzahl verschiedener minimaler Spannbäume eines ungerichteten, gewichteten, zusammenhängenden Graphen G. Sei nun $k \geq 1$. Zeige oder widerlege: Für jedes $j \in \{1, ..., k\}$ gibt es einen Graphen $G \in C_k$ mit $\#MST(G) = j$.

Aufgabe 8.3 (4 Punkte)
Sei folgender Graph G gegeben:



a) Durchlaufe den Graphen G mit dem DFS-Verfahren, wie in der Vorlesung beschrieben. Gib den Verlauf des Stacks und den resultierenden DFS-Wald an. Bestimme die Kantentypen (T,B,F,C) aller Kanten in G.

b) Durchlaufe den Graphen G mit dem BFS-Verfahren, wie in der Vorlesung beschrieben. Gib den Verlauf der Queue an. Gib zudem die Distanzen zum Startknoten an.

Aufgabe 8.4 (4 Punkte)
Für einen gerichteten Graphen sei d eine Funktion für die Größe des Stacks beim DFS-Verfahren. Sobald sich der Stack ändert, nennen wir dies einen Schritt. Zu Beginn, Schritt 0, ist der Stack leer und $d(0) = 0$. Die Größe des Stacks nach i Schritten beträgt $d(i)$. Analog sei b eine Funktion für die Größe der Queue beim BFS-Verfahren. $b(i)$ gibt also an, wie groß die Queue nach i Schritten ist. Beachte: Reichen x Schritte für das DFS- (bzw. BFS-)Verfahren aus, so gilt $d(y) = 0$ (bzw. $b(y) = 0$) für alle $y > x$.

a) Zeige, dass es für alle $n \in \mathbb{N}$ einen zusammenhängenden Graphen mit n Knoten gibt, sodass $d(i) = b(i)$ für alle $i \in \mathbb{N}$ gilt.

b) Zeige, dass kein Graph mit $n \geq 1$ Knoten existiert, sodass $B = 2D$ gilt, dabei ist $B := \min\{i \in \mathbb{N} \mid b(j) = 0 \, \forall j \geq i\}$ und $D := \min\{i \in \mathbb{N} \mid d(j) = 0 \, \forall j \geq i\}$.
D.h. zeige, dass die Anzahl der Schritte beim BFS-Durchlauf nicht doppelt so groß sein kann wie beim DFS-Durchlauf.

c) Zeige, dass es für jedes $k \geq 1$ einen Graphen mit mindestens einem Knoten gibt, sodass $\max_{i \in \mathbb{N}} b(i) = k \cdot \max_{i \in \mathbb{N}} d(i)$.