Theoretische Aufgaben

Aufgabe 1

Der Algorithmus funktioniert nicht. Eine Zusammenhangskomponente beim ursprünglichen Algorithmus, welche den Knoten mit der grössten Endzeit enthält, hat keine Kante, die aus der Zusammenhangskomponente hinausgeht. Wird der Graph nicht transponiert und nimmt man die kleinste Endzeit, so kann gilt die obige Tatsache nicht.

Aufgabe 2

Mit Kruskal: Die Kante wird als erstes ausgewählt und zur Lösungsmenge hinzugefügt.
Mit Prim: Sobald der erste Knoten an der Kante ausgewählt wird (was früher oder später geschieht, da alle Knoten im minimalen Spannbaum vorkommen müssen), wird danach automatisch über die Kante der andere Knoten ausgewählt, wobei die Kante also im minimalen Spannbaum vorkommt.

Aufgabe 3

Wenn `(u,v)` die leichteste Kante des Graphs ist, wird sie beim Kruskal-Algorithmus immer zuerst genommen (z.B. wie in Aufgabe 2). Dies heisst, dass eine Partitionierung der Knoten V in disjunkte Mengen S und V-S auf jeden Fall bei der Kante (u,v) geschnitten wird (da sie die leichteste ist und 'geschnitten' wird).

Aufgabe 4

Sequenz der Kanten (Kruskal): **AF, FI, GH, BC, GK, BG, GJ, IE, BF, GD, GL**.

Aufgabe 5

Sequenz der Kanten (Prim), Startknoten J (zufällig): **JG, GH, GK, GB, BC, BF, FA, FI, IE, GL, GD**.

Aufgabe 6

$$d(a), d(b), d(c), d(d), d(e) \Leftarrow \infty \quad (1)$$
$$a \Rightarrow S \quad (2)$$
$$d(a) \Leftarrow 0 \quad (3)$$
$$d(d) \Leftarrow 1 \quad (4)$$
$$d(c) \Leftarrow 3 \quad (5)$$
$$d(b) \Leftarrow 4 \quad (6)$$
$$d \Rightarrow S \quad (7)$$
$$d(c) \Leftarrow 2 \quad (8)$$
$$c \Rightarrow S \quad (9)$$
$$d(b) \Leftarrow 3 \quad (10)$$
$$d(e) \Leftarrow 8 \quad (11)$$
$$b \Rightarrow S \quad (12)$$
$$d(e) \Leftarrow 5 \quad (13)$$
$$e \Rightarrow S \quad (14)$$

Dies ergibt folgende Minimaldistanzen: `a:0, b:3, c:2, d:1, e:5`.

Aufgabe 7

Wenn im Vorraus bekannt ist, dass es einen MST mit Länge m gibt, brauchen wir nicht zu überprüfen, ob es einen gültigen Weg gibt. Das heisst, die Überprüfung am Ende des Algorithmus' fällt weg.

Listing 1: Aufgabe 7

```
1 bellman-ford(G, w, s)
2 for i := 1 to V - 1
3     for each edge (u,v) in E
4         relax(u,v,w)
```

Aufgabe 8

Es ist zwar eleganter, den verbleibenden Knoten auch aus der Warteschlange zu entfernen, aber dies hat keine Auswirkungen auf die korrekte Arbeitsweise des Algorithmus.

Gegenbeweis: Sei der letzte Knoten der Knoten t. Angenommen, es gibt für irgendeinen anderen Knoten u einen kürzeren Weg, der über t führt. Wenn jedoch die Anweisung `EXTRACT-MIN`, so sollte dieser Weg vor den anderen gegangen werden sein, und der Knoten t wäre nicht der letzte übrigbleibende Knoten. Dies widerspricht der Annahme, t sei der letzte Knoten. Dies kann verallgemeinert werden; Es gibt für einen beliebigen Knoten, der nicht der letzte Knoten ist, keinen kürzeren Weg über t. somit wurde für alle Knoten der kürzeste Pfad gefunden.