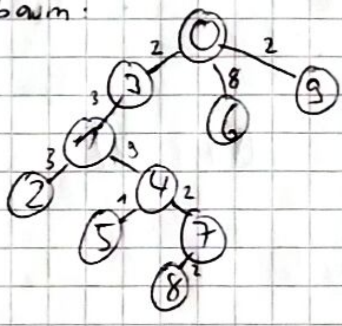
9/10

4.1a) Kruskal-Algorithmus grafisch — Union-Find-DS

	0 1 2 3 4 5 6 7 8 9	[0,1,2,3,4,5,6,7,8,9]
union(4,5)	0 1 2 3 4–5 6 7 8 9	[0,1,2,3,4,4,6,7,8,9]
union(3,0)	0 1 2 3 4–5 6 7 8 9	[0,1,2,0,4,4,6,7,8,9]
union(0,9)	0 1 2 3 4–5 6 7 8 9	[0,1,2,0,4,4,6,7,8,0]
union(7,8)	0 1 2 3 4–5 6 7–8 9	[0,1,2,0,4,4,6,7,7,0]
union(2,1)	0 1–2 3 4–5 6 7–8 9	[0,1,1,0,4,4,6,7,7,0]
union(1,3)	0 1–2 3 4–5 6 7–8 9	[0,3,1,0,4,4,6,7,7 0]
union(7,4)	0 1–2 3 4–5 6 7–8 9	[0,3,1,0,7,4,6,7,7,0]
union(0,6)	0 1–2 3 4–5 6 7–8 9	[0,3,1,0,7,4,0,7,7,0]
union(1,4)	0 1–2 3 4–5 6 7–8 9	[0,3,1,0,1,4,0,4,7,0]

⇒ Minimale Spannbaum:



Leider nein. Union (1,3) hangt nicht 1 an 3, sondern 1 an Wurzel von 3.

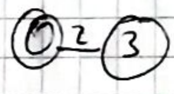
b) Starte in Knoten V_0 0 — Kanten zur Auswahl:

Schritt 1

$\underline{(\{0,3\},2)}$ ← Diese wird gewählt!

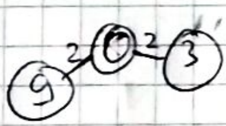
$(\{0,9\},\underbrace{2}_{}$) Gewicht d. Kante

2



$\underline{(\{0,9\},2)}$

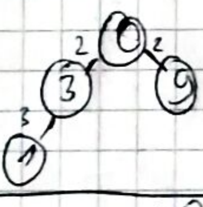
3



$\underline{(\{3,1\},3)}$

$(\{9,2\},3)$

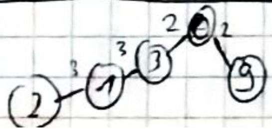
4



$(\{9,2\},3)$

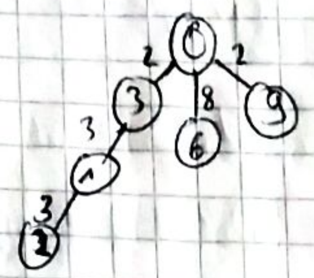
$\underline{(\{1,2\},3)}$

5



$\underline{(\{0,6\},8)}$

6



({3,7}, 9)

({2, 5}, 9)

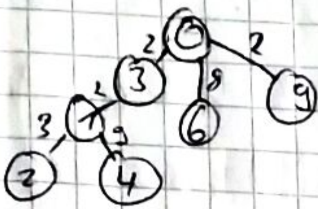
({1, 4}, 9)

({1, 8}, 9)

({6, 7}, 9)

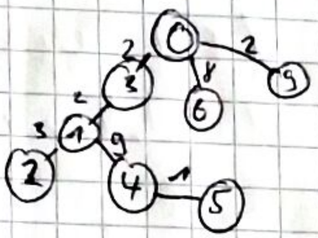
({5, 9,} 9)

7



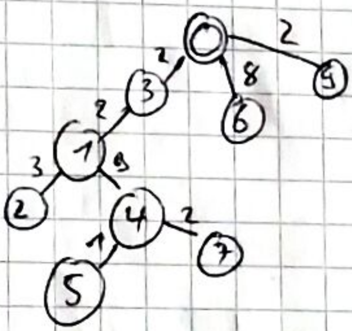
({4, 5}, 1)

8



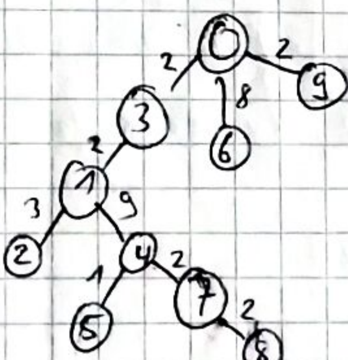
({7, 8}, 2)

9

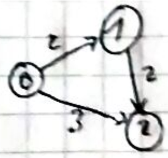


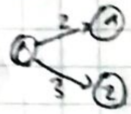
({7, 8}, 2)

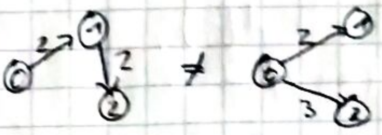
10



ENDE :o

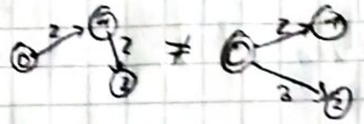
4.1 c) Sei G =  mit Wurzelbaum von G mit Wurzel = 0 :



Kruskal erzeugt MST(G): 

✓

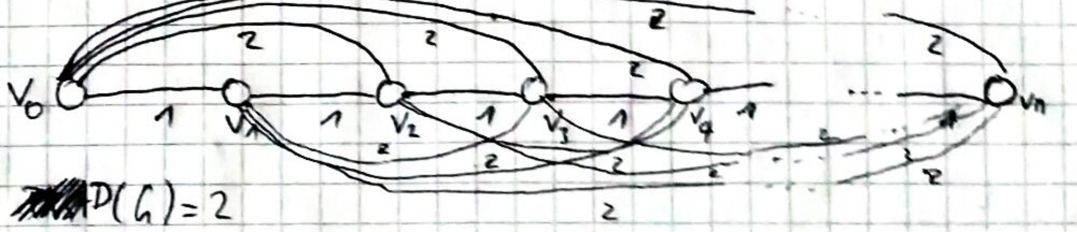
Ebenso Prim erzeugt MST (G):
- mit Startknoten = 0



6/6

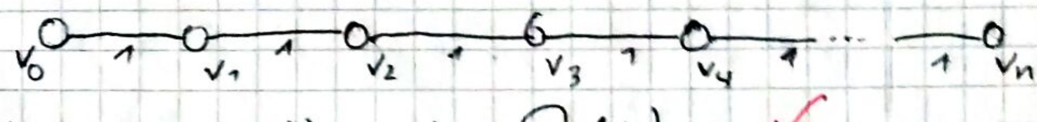
4.2 a) Es sei $e \in E \setminus F$ eine Kante, die in F keinen Kreis schließt und minimales Kantengewicht hat. Wenn wir als nächstes, oder generell NICHT e in F einfügen, ist offensichtlicher Weise die Summe der Kantengewichte im finalen Spannbaum nicht minimal, daher handelt es sich nicht um einen minimalen Spannbaum.
Wenn alle anderen Kanten in $E \setminus F$ beim Einfügen in F einen Kreis schließen, wäre die Struktur des Baumes zerstört.

b) Es sei G ein Graph mit n Knoten, bei dem jeder Knoten mit allen anderen Knoten v_n verbunden ist, Gewicht = 2. Nebenbei existiert ein direkter Weg zwischen jeden der n Knoten, Gewicht = 1 :



$D(G) = 2$

Dann ist MST (G):



Und $D(MST(G)) = n-1 = \Omega(n)$ ✓

6/6

4.3a) Initalisiere $A[i]$ mit $i \quad \forall i \in \{0 .. |V|-1\}$

b)
```
int find(int i){
    while (A[i] != i){
        i = A[i];
    }
    return A[i];
}
```

c)
```
void union(int k1, int k2){
    if (find(k1) != find(k2)){
        A[find(k1)] = A[find(k2)];
    }
}
```

4.4) Es seien folgende Intervalle gegeben

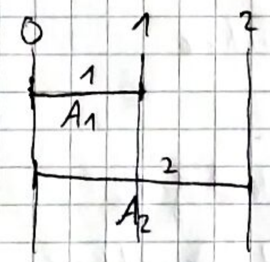
4/4

Startzeit, Endzeit, Gewicht

$A_1 = (0, 1, 1)$

$A_2 = (0, 2, 2)$

Das sieht so also aus:



Wobei der Greedy-Algorithmus aus der Vorlesung sich für A_1 entscheidet. Die Summe der gewählten Gewichte ist 1. Wobei die ~~...~~ schwerstmögliche Menge an Kollisionsfreien Intervallen eine Summe von $2 \neq 1$ hat.

Bestes Minimalbeispiel :D