Terminplanung Termin 5 [07.03.24]

vorher

Marc zeigt Rekursionsdemo von der tu Dresden:
https://dditools.inf.tu-dresden.de/ovk/Informatik/Algorithmen/Algorithmen/Rekursion.html

nachholen

- Einordnung von Dijkstra: Präsentation ***P03-g Dijkstra Typisierung.pdf***
- Hinweis auf Material zu Dijkstra-Kanten-kleiner-Graph.zip
  → Präsentation ***P03-e3 Dijkstra-Schritte kleiner-Graph***
- Das Demoprojekt *Graphen Demoversion mit Schrittfunktion Dijkstra* erneut zeigen und erläutern mit dem Hinweis darauf, dass der nachfolgend betrachtete Algorithmus ein ähnliches Vorgehen zeigt.

Prim und Kruskal

- Präsentation ***P04 Minimal.pdf***
- Zwei Algorithmen (**Prim** und **Kruskal**), im Material zu Anforderungen nur Kruskal

Prim

- Einsetzen Demoprojekt *Graphen interaktive Version* zum Erarbeiten des Vorgehens am Beispiel *„kleiner_Graph.graph“*
- Zeigen: Animation-Prim-Algorithmus.zip
- Zeigen: Projekt *Graphen Demoversion mit Schrittfunktion Prim*
- **Nachweis** der Erfüllung von minimaler Länge
- Gibt es eine Abhängigkeit vom Startknoten bei Prim?
- Projekt Algorithmus-von-Prim.py
  verwendet die Abschnitte aus gallenbacher_dijkstra_iterativ.py
  ersetzt Wege durch Kanten
  prioSchlange auch hier objektorientiert umgesetzt
- dazu Präsentation ***P04-2 Von Dijkstra zu Prim.pdf***

Kruskal

- Einsetzen Demoprojekt *Graphen interaktive Version* zum Erarbeiten des Vorgehens am Beispiel *„Beispielgraph_bewertet.graph“*
- Zeigen: Animation-Kruskal-Algorithmus kommentiert.zip
- Zeigen: Projekt *Graphen Demoversion mit Schrittfunktion Kruskal*
- **Nachweis** der Erfüllung von minimaler Länge
- Präsentation ***P04-1 Algorithmus von Kruskal.pdf***
- Projekt *Algorithmus-von-Kruskal.py*
  - Variante ohne OO Prioritätswarteschlange, also nur Sortieren durch Einfügen eingesetzt
  - und mit Kantenliste
- ~~Wenn Bedarf ggf~~ Erläuterung der Datei *Algorithmus_von_Kruskal_Schritte.py* aus dem Demoprojekt