Übungsblatt 1

Problem 0
Auf `https://fmi.uni-stuttgart.de/alg/research/stuff/` finden Sie einige Beispielgraphen zum herunterladen. Diese haben folgendes Text-Format:

1. Anzahl Knoten
2. Anzahl Kanten
3. Knoten
4. Kanten

Ein Knoten ist wie folgt kodiert:

```
<Knoten ID> <OSM ID> <Längengrad> <Breitengrad> <Höhe über NN>
```

Eine Kante hingegen:

```
<Startknoten ID> <Endknoten ID> <Gewicht> <Typ> <Maximal erlaubte Geschwindigkeit>
```

Der Knoten

```
0 1212985 53.82233040000000557 10.72339740000000141 0
```

ist ein Knoten mit

- der Id 0
- Osm Id 1212985
- Längengrad 53.82233040000000557
- Breitengrad 10.72339740000000141
- einer Höhe über NN von 0

Die Kante

```
626656 626655 67 15 30
```

ist eine Kante

- von Knoten 626656 zu Knoten 626655
- mit dem Gewicht (hier Länge) von 67
- dem Typ 15 = highway:service
- mit einer maximalen Geschwindigkeit von 30 kmh

Die Dateien können auch mit Hilfe des OsmGraphCreator (`https://github.com/fmi-alg/OsmGraphCreator`) erstellt werden. Hierzu muss das Programm mit den Parametern:

```
./creator -g fmimaxspeedtext -t distance -c ../../data/configs/all.cfg \
 -o output.graph -s source.osm.pbf
```

gestartet werden. Die Quelldateien können Sie sich von `http://download.geofabrik.de/` herunterladen.

Problem 1
Implementieren Sie ein Programm, welches diese Dateien einliest und in eine Graphdatenstruktur im Arbeitsspeicher überführt. Den Graph können Sie mithilfe von Offset-Arrays ablegen.

Wichtig: Ihre Graphdarstellung sollte es erlauben, sowohl über alle ausgehenden, als auch alle eingehenden Kanten eines Knotens zu iterieren.

Problem 2
Bestimmen Sie die Anzahl der schwachen Zusammenhangskomponenten in den jeweiligen Graphen und messen Sie die Laufzeit ihres Programms. Auf dem Deutschlandgraph sollte die Lfz. (exklusive Einlesen) unter 10s betragen.

Problem 3
Untersuchen Sie den Effekt der Reihenfolge in der die Knoten abgespeichert sind: permutieren Sie die Knoten des Graphs zufällig und wiederholen Sie den Versuch der vorhergehenden Frage.

Problem 4
Implementieren Sie den Algorithmus von Dijkstra zur Berechnung kürzester Wege. Wählen Sie zufällig Punkte im Graphen aus und starten Sie eine kürzeste Wege Suche. Messen Sie die Laufzeit für 100 Knotenpaare. Auf dem Deutschlandgraph sollte die Lfz. im Schnitt unter 10s betragen

Problem 5
Erweitern Sie Ihre Implementierung auch dahingehend, dass eine Datei mit Source-Target-Paaren eingelesen werden kann, für die dann die Distanzen berechnet werden (und samt Queryzeiten in eine Datei rausgeschrieben werden können). Verifizieren Sie die Distanzen mithilfe der Distanzwerte auf `https://fmi.uni-stuttgart.de/alg/research/stuff/`. Die Input Datei sollte pro Zeile folgendes Format haben:

```
<Start ID> <Target ID>
```

Die Output Datei sollte folgendes Format mit einer Query pro Zeile haben:

```
<Start ID> <Target ID> <Distanz> <Laufzeit>
```

Problem 6
Reichen Sie Ihren Programmcode samt Anleitung, wie man es zu bauen/auszuführen hat, ein. Bitte beachten Sie, dass wir Ihr Programm automatisiert testen werden. Achten Sie daher bitte darauf, dass das Programm lauffähig ist und einen Output im geforderten Format produziert!

Bei diesem konkreten Blatt: sollten wir Ihren Code wie folgt ausführen können:

- auspacken Ihres Programmcodes
- *wir* kopieren Dateien `graph.fmi` und `queries.txt` in das oberste Verzeichnis Ihres Programmcodes
- wir tippen `make` ein; folgendes sollte passieren:
  - eines Ihrer Programme liest `graph.fmi` ein, berechnet die Anzahl der schwachen Zusammenhangskomponenten und gibt die benötigte Laufzeit hierfür aus (Frage 2)
  - eines Ihrer Programm liest `graph.fmi, queries.txt` ein und schreibt die Ergebnisse in eine Datei `result.txt` (Frage 5)