Dokumentation der Praktischen Arbeit zur Prüfung zum Mathematisch-technischen Softwareentwickler

23. Mai 2019

Jakob Rockenbauch

Prüfungs-Nummer: 30 6511 142 18688

Programmiersprache: golang

Inhaltsverzeichnis

1. Aufgabenanalyse

1.1. Analyse

In der Aufgabe geht es darum schemenhafte Karten auf Basis von vorgegebenen Kennwerten zu erstellen. Dabei werden Staaten als Kreise dargestellt und die Fläche der Kreise ist proportional zum Kennwert. Die Schwierigkeit liegt hierbei eine möglichst überschneidungsfreie Darstellung zu finden und gleichzeitig die urspünglichen Lage- und Nachbarschaftsbeziehungen möglichst beizubehalten.

1.2. Eingabeformat

Das Programm erwartet den Pfad zu einer Datei oder Verzeichniss enthält als Kommandozeilenparameter. Falls der übergebene Pfad ein Verzeichniss ist werden alle Dateien in diesem Verzeichniss die mit .txt enden als Eingaben verwendet und hintereinander abgearbeitet. Falls nur eine Datei übergeben wird muss sie mit .txt enden.

Die Eingabedateien werden nach folgenden Formatsvorgaben eingelesen. Bei Abweichungen von der Vorgabe ist das Verhalten undefiniert oder eine Fehlermeldung wird ausgegeben.

1.2.1. Formatsforgaben

Kommentare

Alle Zeilen die mit einem oder mehreren # Zeichen beginnen werden als Kommentare ignoriert.

Titel

Die erste Zeile wird als Titel des Kennwertes gepeichert.

Staaten

Die darauf folgenden Zeilen beschreiben die Staaten die in der Karte dargestellt werden. Sie müssen aus einer ID, einem Kennwert, einem Längengrad und einem Breitengrad bestehen. Die einzelnen Werte müssen durch eine beliebige Anzahl von Whitespace getrennt sein. Die ID muss ein string sein, der Kennwert eine ganze positive Zahl und Längengrad und Breitengrad eine Fließkommazahl. Es werden so lange Staaten eingelesen bis eine Zeile ein : Symbol enthält.

Nachbarschaften

Alle weiteren Zeilen müssen mit der ID eines Staates gefolgt von einem Doppelpunkt(:), gefolgt von einer Liste von IDs bestehen. Diese Zeilen bilden die Nachbarschaftsbeziehungen der Staaten ab. Nachbarschaften sind symetrisch das heißt wenn z.B. NL ein Nachbar von D ist auch D ein Nachbar von NL. Es reicht jedoch wenn in der Eingabe die Nachbarschaft nur in eine richtung enthalten ist. Die Rückrichtung gilt dann automatisch auch.

1.2.2. Beispiel

```
Bierkonsum
# Staat Bierkonsum Laengengrad Breitengrad
D   8692     10.0     51.3
NL  1156      5.3     52.2
B    781      4.8     50.7
L     80      6.1     49.8
F   2077      2.8     47.4
CH   440      8.2     46.9
A    945     14.2     47.6
CZ  1573     15.3     49.8
PL  3724     18.9     52.2
DK   360      9.6     56.0
# Nachbarschaften
D: NL B L F CH A CZ PL DK
NL: B
B: L F
F: CH
CH: A
A: CZ
CZ: PL
```

1.3. Ausgabeformat

Die Ausgabe erfolgt in eine Datei oder meherere Dateien. Die Ausgabe ist speziell auf das Programm gnuplot ab Version 5 zugeschnitten. Diese Dateien haben den gleichen Namen wie die Eingabedateien doch eine .gnu Endung anstatt der .txt Endung.

1.3.1. Beschreibung

Das Format der Ausgabedateien ist wie folgt:

```
reset
set xrange [<xmin>:<xmax>]
set yrange [<ymin>:<ymax>]
set size ratio 1.0
set title "<titel>, Iteration: <nr>"
unset xtics
unset ytics
$data << EOD
<Liste aus <xpos> <ypos> <radius> <ID> <numericID>>
EOD
plot \
'$data' using 1:2:3:5 with circles lc var notitle, \
'$data' using 1:2:4:5 with labels font "arial,9" tc var notitle
```

Dabei werden die <Tags> wie folgt ersetzt.

Tag	Inhalt
<xmin>	Kleinster X-Wert im Darstellungsbereich
<xmax>	Größter X-Wert im Darstellungsbereich
<ymin>	Kleinster Y-Wert im Darstellungsbereich
<ymax>	Größter Y-Wert im Darstellungsbereich
<titel>	Der eingelesene Titel
<nr>	Anzahl der Iterationen
<Liste aus ....>	Liste aus Staaten, je Zeile ein Staat.
<xpos>	X-Koordinate des Kreismittelpunktes
<ypos>	Y-Koordinate des Kreismittelpunktes
<ID>	ID des Staates
<numericID>	Forlaufende numerische ID (dient zur Färbung)

1.3.2. Beispiel

```
reset
set xrange [-33.86357381155014:85.0741647360979]
```

```
set yrange [-2.1735154975148223:116.76422305013321]
set size ratio 1.0
set title "Bierkonsum, Iteration: 1000"
unset xtics
unset ytics
$data << EOD
16.751229415522495 63.89562847551619 29.676315947556564 D 0
-23.041037681301255 71.42772638479775 10.822536130248883 NL 1
-20.220018532838186 51.91242303158999 8.895608153727649 B 2
-10.217085354402021 46.65785245565221 2.8470501736687086 L 3
-11.701867227882616 29.77460395147963 14.506691505134297 F 4
9.429475402706583 28.287342791648207 6.676924501791871 CH 5
25.645069779242753 25.449506312958032 9.78511719613052 A 6
46.450989068634634 33.77447230655958 12.624508777350401 CZ 7
65.64943596362681 59.437154636063894 19.42472877247108 PL 8
-3.546190833310066 93.28328965373461 6.039505452538444 DK 9
EOD
plot \
'$data' using 1:2:3:5 with circles lc var notitle, \
'$data' using 1:2:4:5 with labels font "arial,9" tc var notitle
```

1.4. Fehlerfälle

Im Falle eines Fehler durch falsche Eingaben wird eine Fehlermeldung ausgegeben und die Lösungsfindung für die entsprechende Eingabe abgebrochen.

1.5. Vereinfachungen

Gegenüber der Realität unserer Erde wurden hierbei eine Reihe von vereinfachten Ahnnahmen getroffen.

1. Die Karte ist unendlich groß.
2. Die Karte ist flach und nicht gewölbt.
3. Der Abstand zwischen zwei Längengraden ist immer gleich groß.
4. Der Abstand benachbarter Längengrade ist gleich dem Abstand benachbarter Breitengrade
5. Daraus folgend sind Abstände nicht auf einer Kugel berechnet.

2. Verfahrensbeschreibung

2.1. Übersicht

Zuerst werden die Staaten und Nachbarschaftsbeziehungen aus der Eingabedatei eingelesen. Danach wird mithilfe eines iterativen Algorithmus mit einer festen Anzahl von Iterationen eine Lösung gesucht. Diese wird dann in eine für gnuplot aufbereitete Form in eine Ausgabedatei geschrieben.

2.2. Einlesen

Das Einlesen geschieht mit einem speziell auf das Eingabeformat abgestimmten Parser der die Eingabedatei nur genau einmal durchläuft. Der Parser ist so geschrieben das er die Eingabe nicht nur aus Dateien lesen kann sondern ist über das io.Reader interface aus der golang Standardbibliothek erweiterbar.

2.3. Algorithmus

Der Algorithmus besteht aus einer festen Anzahl von Iterationen. In jeder Iterationen werden die Mittelpunkte der Staaten verschoben. Diese Verschiebungen geschehen in Abhängigkeit von den aktuellen Überschneidungen sowie den Abständen von Nachbarstaaten. Dafür werden zuerst für jeden Staat Abstoßungskräfte durch Überschneidungen und Anziehungskräfte durch Nachbarn berechnet, dann diese pro Staat addiert und schlussendlich alle Staaten proportional zur berechneten Kraft verschoben. Diese Verschiebung ist gedämpft um die echten Lagebeziehungen möglichst beizubehalten und um Schwingungseffekte zu verringern.

2.3.1. Berechnung der Kräfte

Zur Berechnung der Kräfte wird über jeden Staat(a) iteriert und der Abstand zu jedem anderen Staat(b) berechnet, hierbei ist der kürzeste Abstand von Kreisaußenkante zu Kreisaußenkante gemeint und nicht der Abstand der Mittelpunkte. Falls die Kreise sich über-

schneiden ist der Abstand negativ. Abhängig davon ob die Länder benachbart sind wird dann eine Kraft auf den Staat berechnet. Die Stärke der Kraft(f) abhängig vom Abstand(d) berechnet sich bei Nachbarstaaten wie folgt:

$$f(d) = d$$

Dies sorgt dafür das Nachbarn sich berühreren. Bei nicht Nachbarschaftsbeziehungen:

$$f(d) = \begin{cases} 1,2 \cdot d & \text{falls } d < 0 \\ 0 & \text{sonst} \end{cases}$$

Dies sorgt dafür das nicht benachbarte Länder sich nicht überschneiden. Die Richtung der Kraft ist hierbei immer vom Mittelpunkt des Staates a in Richtung des Mittelpunktes Staat b. Bei einer negativen Stärke wirkt die Kraft in die umgekehrte Richtung. Da die Kraft von b auf a auch gleichzeitig eine gleichstarke Kraft von a auf b bewirkt, werden alle Kräfte halbiert damit die Abstände durch sie nicht doppelt kompensiert werden.

2.3.2. Anwendung der Kräfte

Nachdem alle Kräfte berechnet wurden wird über alle Staaten iteriert und auf jeden die berechnete Kraft(f) angewendet. Dies geschieht durch eine Verschiebung(v):

$$v = c \cdot f, c\epsilon(0,1)$$

Wobei c ein Dämpfungsfaktor zwischen 0 und 1 ist. Die Dämpfung sorgt dafür das die Lagebeziehungen nicht sicht nicht zu stark ändern und soll Schwingungseffekte verhindern.

2.4. Komplexität

Dadurch das für jeden Staat der Abstand zu jedem anderen Staat berechnet werden muss, liegt die Komplexität des Algorithmus bei $O(n)^2$ wobei n die Anzahl der Staaten ist.

2.5. Ausgabe

Die Ausgabe ist getrennt von dem Algorithmus und bereitet die Ergebnisse für gnuplot auf.

3. Programmbeschreibung

3.1. Entwicklerdokumentation

Eine Dokumentation kann mit dem go doc Befehl erstellt werden oder ist hier zu finden.

3.2. Packages

Das Gesamte Programm ist in 4 Packages aufgeteilt. Es gibt ein Package model in dem die Datenstrukturen definiert sind. Außerdem gibt es jeweils für die Eingabe, den Algorithmus und die Ausgabe ein eigenes Package. Diese Packages sind untereinander unabhänging und loose über die im model Package definierten Datenstrukturen gekoppelt. Außerdem gibt es natürlich das main Package das in der main() Methode alles verbindet.

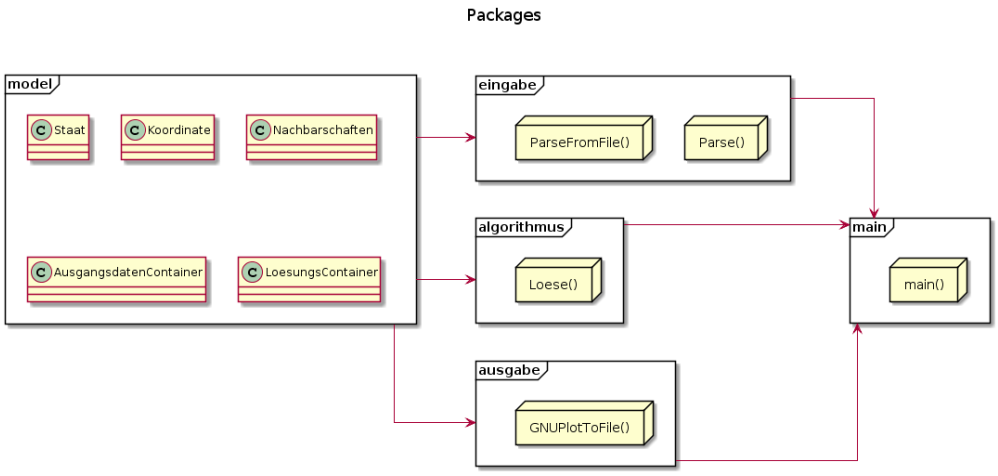


Abbildung 3.1.: Übersicht Packages

3.2.1. Model

Das Package Model enthält alle benötigten Klassen. Es wurde versucht die Klassenhirarchien möglichst simpel zu gestalten.

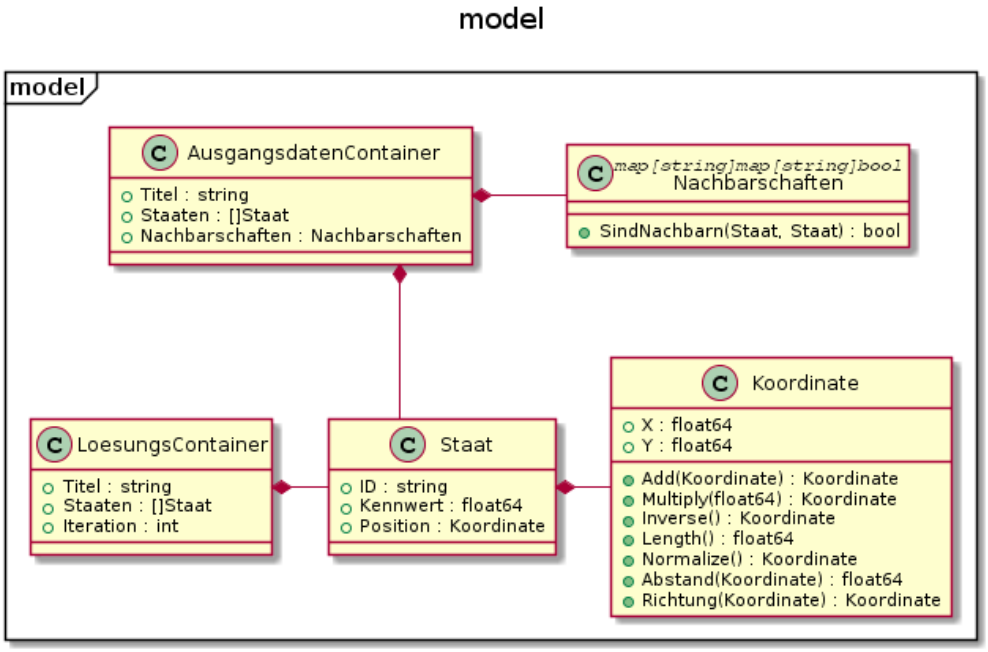


Abbildung 3.2.: Package model

Koordinate

Die Klasse Koordinate beschreibt einen Vektor im R^2. Sie besitzt eine Reihe von Helfermethoden, die einfache Operationen wie die Streckung oder Addition von Koordinaten abbilden.

Als Datentyp für die Felder wurde float64 gewählt um eine möglichst große Genauigkeit zu haben. Außerdem entahlten einige Methoden NaN Checks nach den Operationen und liefern im Zweifel einen Nullvektor zurück um numerische Instabilitäten zu vermeiden.

Staat

Die Klasse Staat ist die Datenstruktur für einen Staat und besteht aus einer string als ID, einem float64 als Kennwert und einer Koordinate als Position.

Nachbarschaften

Die Klasse Nachbarschaften beschreibt die Nachbarschaften zwischen den Staaten. Sie ist ein typedef auf den Type map[string]map[string]bool, und beschreibt somit eine 2-dimensionale boolsche Set mit string Keys. Durch das Ausnutzen von go's zero value erlaubt

uns dies sehr einfache Prüfungen von Nachbarschaftsbeziehungen. Mehr Informationen dazu lassen sich hier finden. Die Methode SindNachbarn(Staat, Staat) prüft ob zwei Staaten Nachbarn sind. Dies ist dank der Verwendung einer Map in $O(1)$ möglich.

AusgangsdatenContainer

Die Klasse AusgangsdatenContainer fasst alle zur Lösung des Problems notwendigen Informationen zusammen. Sie dient als Schnittstellendatentyp zwischen Eingabe und Algorithmus.

LoesungsContainer

Die Klasse LoesungsContainer fasst die Lösung zusammen. Sie dient als Schnittstellendatentyp zwischen Algorithmus und Ausgabe.

3.2.2. Eingabe

Das Package Eingabe enthält Funktionen um das spezifizierte Eingabeformat zu parsen. Dafür stellt es zwei Funktionen zu Verfügung. Die Funktionen liefern alle einen AusgangsdatenContainer und einen error zurück.

Parse

Die Methode Parse() erwartet ein Objekt des Interfaces io.Reader und liefert ein Objekt der Klasse AusgangsdatenContainer und ein Objekt des Interfaces error zurück. Falls beim Einlesen kein Fehler aufgetreten ist, wird nil als error zurück geliefert. Durch das io.Reader Interface ist die Parse Funktion nicht auf einen festen Eingabetypen festgelegt und somit leicht für verschiedene Datentypen verwendbar.

ParseFromFile

Die Methode ParseFromFile erwartet als Parameter den Pfad zu einer Datei. Sie versucht dann diese Datei zu öffnen und seinen Inhalt mithilfe der Parse() Funktion einzulesen. Sie liefert ein Objekt der Klasse AusgangsdatenContainer und ein Objekt des Interfaces error zurück. Der error ist heirbei entweder ein Fehler der beim öffnen der Datei passiert oder ein Fehler beim Einlesen der Datei mit der Parse() Funktion.

3.2.3. Algorithmus

Das Package Algorithmus enthält die Funktionen zum lösen des Problemes.

Loese

Die Funktion Loese ist die einzig öffentliche Funktion des Packages. Sie erhält einen AusgangsdatenContainer als Parameter und liefert einen LoesungsContainer zurück. Sie erstellt ein Objekt der Klasse algorithmus, initialisiert es und ruft dann in einer Schleife 1000 mal die iteration Methode auf ihm auf. Danach liefert sie die Loesung zurück.

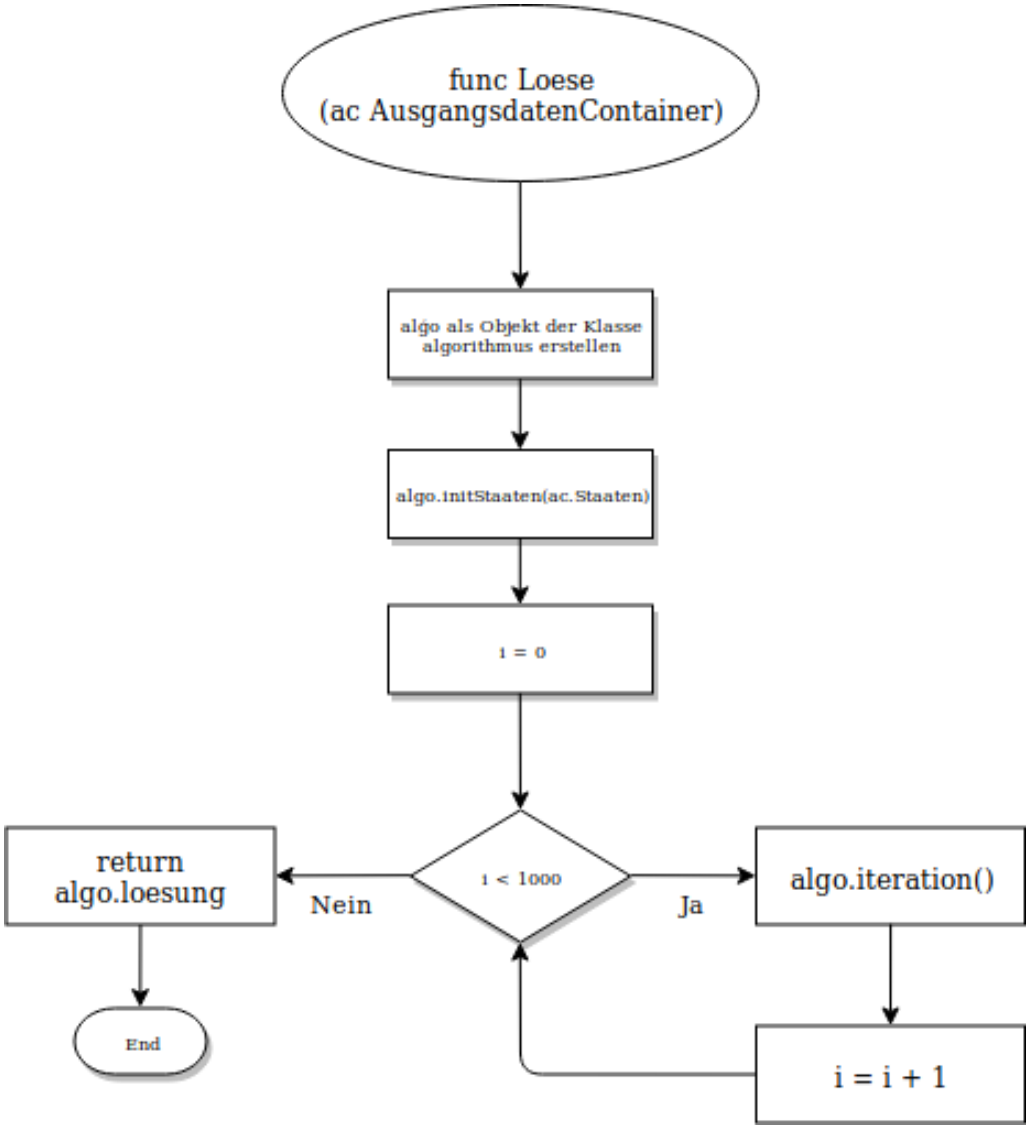


Abbildung 3.3.: Funktion Loese

algorithmus

Die Klasse algorithmus implementiert die Lösung des Problemes und speichert den aktuellen Stand der Lösung. Zusätzlich hat sie ein Feld für Nachbarschaftsbeziehungen und eins für die aktuell in dieser Iteration berechneten Kräfte. Die wichtigste Methode ist iteration(). Diese berechnet die nächste Iteration der Lösung.

algorithmus.iteration()

Die Methode Iteration ist der Kern des Algorithmus, hier werden die Kräfte berechnet und die Staaten verschoben.

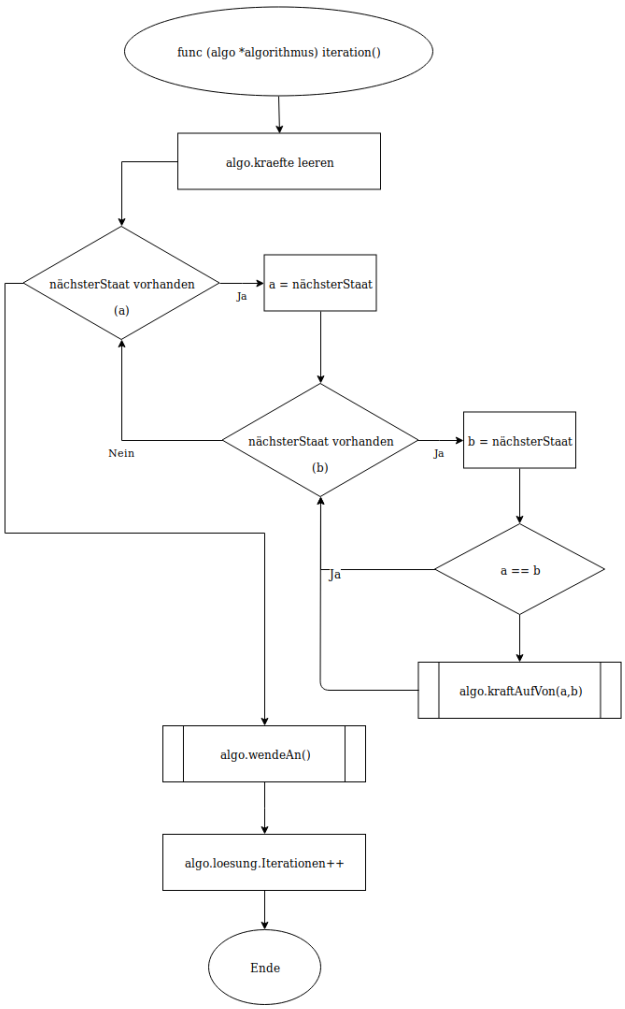


Abbildung 3.4.: Methode algorithmus.Iteration

algorithmus.kraftVonAuf()

In der Methode kraftVonAuf wird die Kraft von Staat b auf Staat a berechnet.

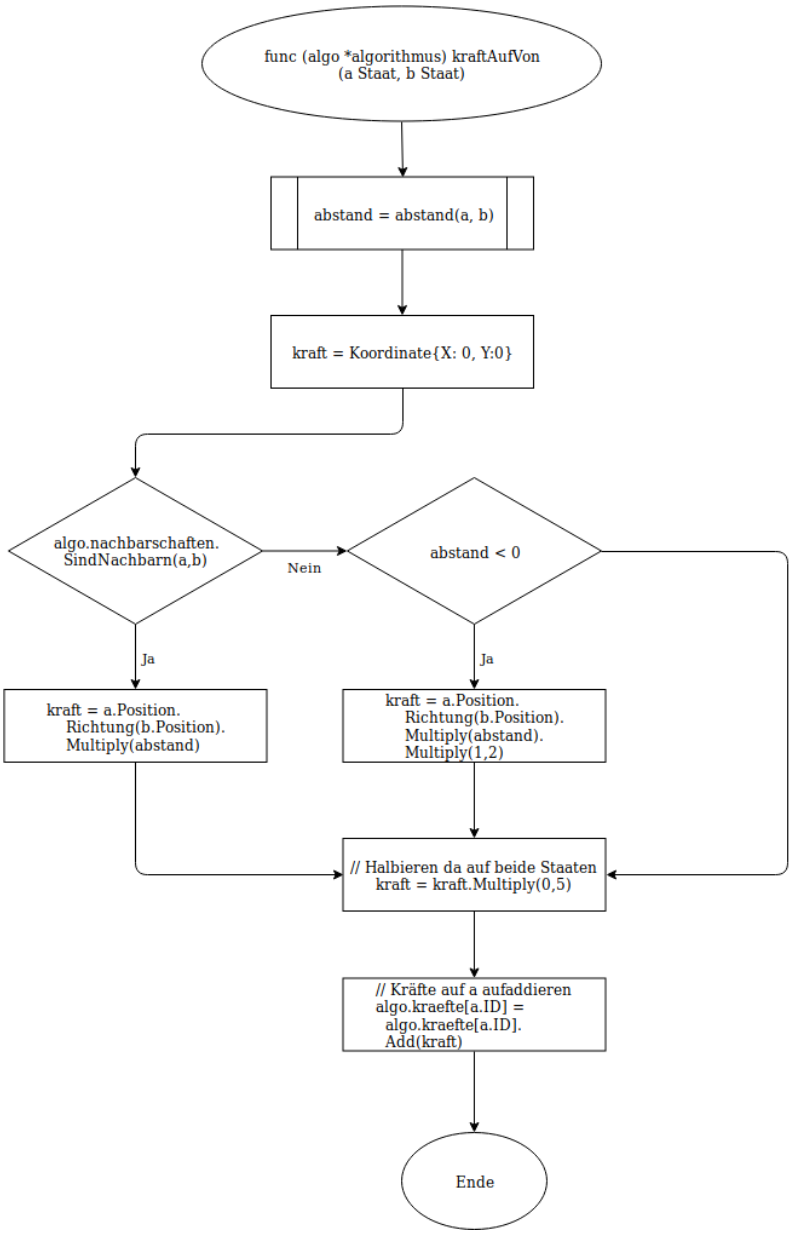


Abbildung 3.5.: Methode algorithmus.kraftVonAuf

algorithmus.wendeAn()

Die Methode wendeAn() wendet die berechneten Kräfte auf die Staaten an.

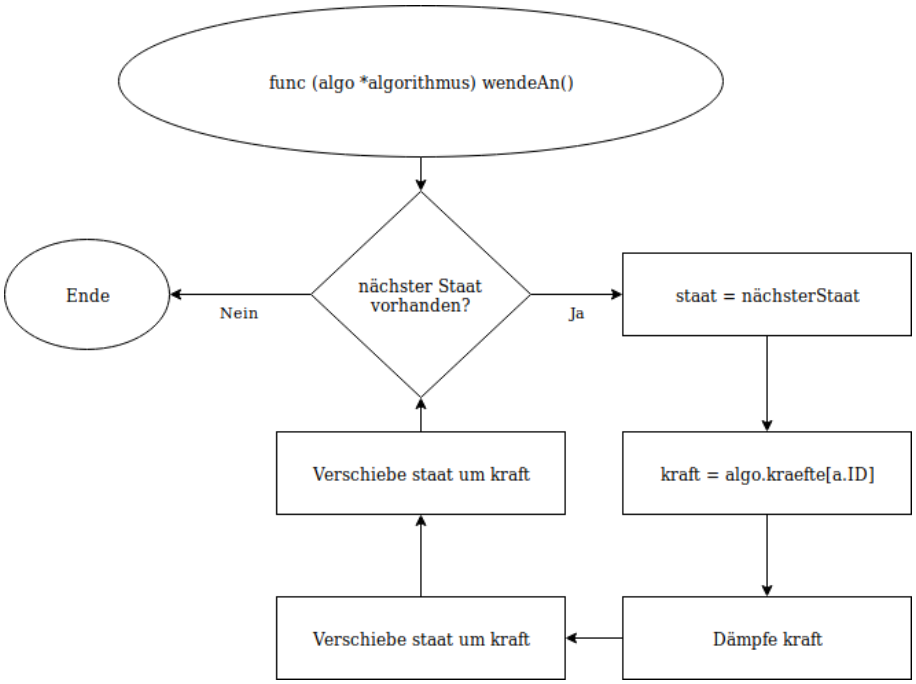


Abbildung 3.6.: Methode algorithmus.wendeAn

3.2.4. Ausgabe

Das Package ausgabe enthält die nötigen Funktionen um die Lösung für gnuplot aufzubereiten und in eine Ausgabedatei zu schreiben.

GNUPlotToFile()

Die Funktion GNUPlotToFile schreibt die Ausgabedatei in dem an gnuplot ausgerichteten Format. Sie ruft dabei die Helferfunktion findExtrema auf um sicherzustellen das die xRange gleich groß wie die yRange ist. Falls es einen Fehler beim erstellen der Datei gibt wird dieser zurückgegeben.

findExtrema()

findExtrema ist eine Helferfunktion um sicherzustellen das die xRange gleich groß wie die yRange ist und alle Staaten vollständig im Bild enthalten sind. Dafür findet sie erst das kleinste Rechteck um alle Staaten herum und verlängert dann die kürzere Seite des Rechtecks gleichmäßig auf beiden Seiten um ein Quadrat zu erschaffen.

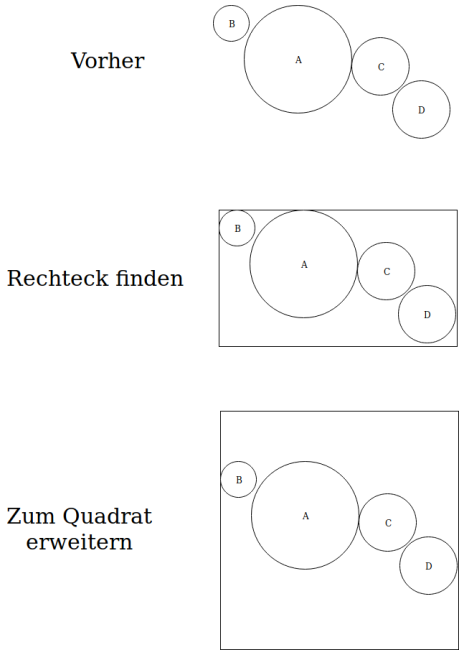


Abbildung 3.7.: Funktion findExtrema

4. Testdokumentation

Das Programm ist durch umfangreiche Unit Tests und durch die 3 Testbeispiele der Aufgabenstellung getestet.

5. Ausblick

5.1. Ausblick

Das Programm funktioniert, doch es sind noch viele weitere Verbesserungen denkbar und sinnvoll.

5.1.1. Abruchkriterium

Es wäre sinnvoll die Anzahl der Iterationen dynamisch früher abzubrechen falls ein gewisses Abruchkriterium erreicht ist. Denkbar wäre ein Abbruch z. B. wenn alle Kräfte sehr klein sind.

5.1.2. Voriteration

Auch eine Voriteration in der z.B. einfach alle Staaten weiter auseinander geschoben werden oder die durchschnittliche Größe der Kreise auf den durschnittlichen Abstand von Nachbarstaaten normiert werden könnte von Vorteil sein.

5.1.3. Direkte Anbindung an gnuplot

Es wäre denkbar gnuplot direkt aus dem Programm heraus aufzurufen und die Bilder zu erstellen.

5.1.4. Andere Ausgabeprogramme

Eine Anbindung an andere Grafikprogramme zum anzeigen der Bilder wäre möglich.

5.1.5. REST Service

Eine weitere Verbesserung der Bedienbarkeit wäre es das Programm in einen Service mit einer HTTP/REST Schnittstelle zu erweitern.

A. Abweichungen und Ergänzungen zum Vorentwurf

Die größten Abweichungen zur Vorlage liegen darin das die Abstoßungs- und Anziehungskräfte nicht getrennt berechnet werden sondern in einem Schritt.

B. Benutzeranleitung

B.1. Vorraussetzungen

B.1.1. Binary

Um das vorkompilierte Binary auszuführen ist ein 64bit Linux System notwendig.

B.1.2. Kompilation

Vorraussetzungen zur Kompilation des Programmes ist eine go in Version >= 1.11. Für die Ausführung der Unittests ebenfalls. Das Programm unterstütz go modules.

Die Kompilation erfolget mit dem

```
build.sh
```

Skript oder dem Befehl

```
go build .
```

B.2. Ausführung

Die Ausführung geschieht durch das aufrufen der Binary, durch das

```
run.sh
```

Skript oder dem Befehl

```
go run .
```

Alle Varianten erfordern als Parameter den Pfad zu einer Datei oder einen Ordner mit ".txt"Dateien.

B.3. Tests

Die Tests können mit dem

```
run:_tests.sh
```

Skript oder dem Befehl

```
go test --cover ./...
```

ausgeführt werden.

C. Entwicklungsumgebung

Programmiersprache	:	golang
Compiler	:	go1.12.4 linux/amd64
Rechner	:	amd64
Betriebssystem	:	Ubuntu 18.04

D. Verwendete Hilfsmittel

D.1. Programme

- Editor: Visual Studio Code
- Layout: LaTeX
- Diagramme: https://www.planttext.com/
- Programmablaufpläne: https://draw.io
- Versionsverwaltung: git, https://github.com

D.2. Quellen

- https://godoc.org
- https://stackoverflow.com

E. Erklärung

Erklärung des Prüfungsteilnehmers / der Prüfungsteilnehmerin:

Ich erkläre verbindlich, dass das vorliegende Prüfprodukt von mir selbstständig erstellt wurde. Die als Arbeitshilfe genutzten Unterlagen sind der Arbeit vollständig aufgeführt. Ich versichere, dass der vorgelegte Ausdruck mit dem Inhalt der von mir erstellten digitalen Version identisch ist. Weder ganz noch in Teilen wurde die Arbeit bereits als Prüfungsleistung vorgelegt. Mir ist bewusst, dass jedes Zuwiderhandeln als Täuschugnsversuch zu gelten hat, der die Anerkennung des Prüfproduktes als Prüfungsleistung ausschließt

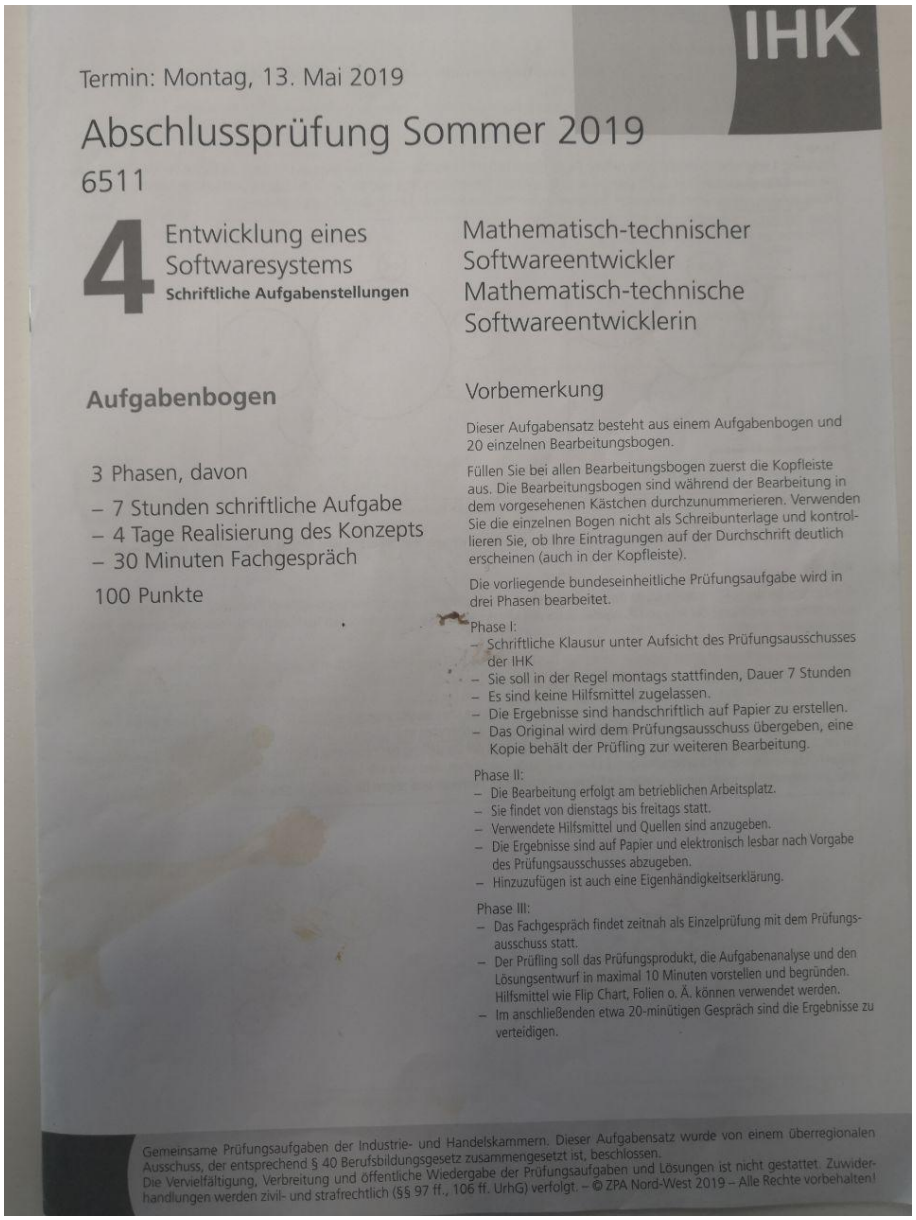
Bonn, den 23. Mai 2019

Ort und Datum

Unterschrift des Prüfungsteilnehmers

Anhang F

Aufgabenstellung



IHK

Termin: Montag, 13. Mai 2019

Abschlussprüfung Sommer 2019

6511

4 Entwicklung eines Softwaresystems
Schriftliche Aufgabenstellungen

Mathematisch-technischer Softwareentwickler
Mathematisch-technische Softwareentwicklerin

Aufgabenbogen

3 Phasen, davon
- 7 Stunden schriftliche Aufgabe
- 4 Tage Realisierung des Konzepts
- 30 Minuten Fachgespräch

100 Punkte

Vorbemerkung

Dieser Aufgabensatz besteht aus einem Aufgabenbogen und 20 einzelnen Bearbeitungsbogen.

Füllen Sie bei allen Bearbeitungsbogen zuerst die Kopfleiste aus. Die Bearbeitungsbogen sind während der Bearbeitung in dem vorgesehenen Kästchen durchzunummerieren. Verwenden Sie die einzelnen Bogen nicht als Schreibunterlage und kontrollieren Sie, ob Ihre Eintragungen auf der Durchschrift deutlich erscheinen (auch in der Kopfleiste).

Die vorliegende bundeseinheitliche Prüfungsaufgabe wird in drei Phasen bearbeitet.

Phase I:
- Schriftliche Klausur unter Aufsicht des Prüfungsausschusses der IHK
- Sie soll in der Regel montags stattfinden, Dauer 7 Stunden
- Es sind keine Hilfsmittel zugelassen.
- Die Ergebnisse sind handschriftlich auf Papier zu erstellen.
- Das Original wird dem Prüfungsausschuss übergeben, eine Kopie behält der Prüfling zur weiteren Bearbeitung.

Phase II:
- Die Bearbeitung erfolgt am betrieblichen Arbeitsplatz.
- Sie findet von dienstags bis freitags statt.
- Verwendete Hilfsmittel und Quellen sind anzugeben.
- Die Ergebnisse sind auf Papier und elektronisch lesbar nach Vorgabe des Prüfungsausschusses abzugeben.
- Hinzuzufügen ist auch eine Eigenhändigkeitserklärung.

Phase III:
- Das Fachgespräch findet zeitnah als Einzelprüfung mit dem Prüfungsausschuss statt.
- Der Prüfling soll das Prüfungsprodukt, die Aufgabenanalyse und den Lösungsentwurf in maximal 10 Minuten vorstellen und begründen. Hilfsmittel wie Flip Chart, Folien o. Ä. können verwendet werden.
- Im anschließenden etwa 20-minütigen Gespräch sind die Ergebnisse zu verteidigen.

Gemeinsame Prüfungsaufgaben der Industrie- und Handelskammern. Dieser Aufgabensatz wurde von einem überregionalen Ausschuss, der entsprechend § 40 Berufsbildungsgesetz zusammengesetzt ist, beschlossen.
Die Vervielfältigung, Verbreitung und öffentliche Wiedergabe der Prüfungsaufgaben und Lösungen ist nicht gestattet. Zuwiderhandlungen werden zivil- und strafrechtlich (§§ 97 ff., 106 ff. UrhG) verfolgt. – © ZPA Nord-West 2019 – Alle Rechte vorbehalten!

Landkarten

Für die Darstellung staatenspezifischer Kennwerte können Landkarten erstellt werden, in denen die einzelnen Staaten
– einerseits nur schemenhaft, z. B. in Form von Kreisen,
– andererseits die Größe der Staaten nicht proportional zur Fläche der Staaten, sondern zur Größe des jeweiligen Kennwertes
dargestellt werden.

Beispiel:
Abbildung 1 zeigt eine schematische Darstellung Deutschlands und seiner Nachbarstaaten. Der Kennwert ist dabei die Fläche der jeweiligen Staaten, wie es auf Karten meist der Fall ist. Abbildung 2 zeigt die gleichen Staaten, allerdings wird die Größe der Staaten jetzt nicht im Verhältnis zu ihrer Fläche, sondern zum gesamten Bierkonsum des Staates dargestellt. Der Kennwert ist hier also der Gesamtbierkonsum:

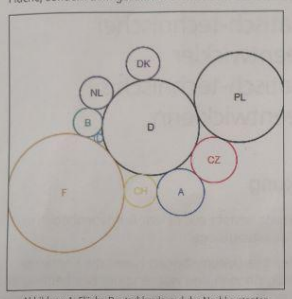


Abbildung 1: Fläche Deutschlands und der Nachbarstaaten

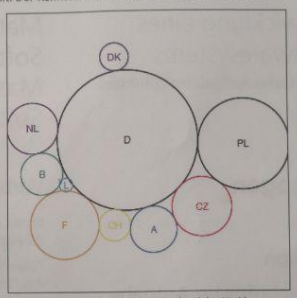


Abbildung 2: Bierkonsum in Deutschland und den Nachbarstaaten

Man erkennt am Vergleich der beiden Karten, dass in Frankreich deutlich weniger Bier getrunken wird als z. B. in Deutschland oder Tschechien.

Aufgabe

Ihre Aufgabe bei der MATSEgraphie AG ist die Erstellung solcher schemenhaften Karten auf Basis vorgegebener Kennwerte. Ausgangspunkt ist dabei immer die reale Lage der Staaten (angegeben durch den Mittelpunkt/Schwerpunkt), die realen Nachbarschaftsbeziehungen sowie der jeweils darzustellende Kennwert.

Die Darstellung der Karte ist umso besser, je mehr reale Nachbarschafts- und Lagebeziehungen erhalten bleiben. In den obigen Beispielen werden alle Nachbarschafts- und Lagebeziehungen (im Wesentlichen) korrekt dargestellt, so befindet sich Frankreich beispielsweise weiter im Südwesten von Deutschland und grenzt an Belgien, Luxemburg, die Schweiz und eben Deutschland. Die Koordinaten der einzelnen Staaten werden durch die Darstellung der Kennwerte in der Regel verfälscht, wie man beispielsweise oben an den unterschiedlichen Mittelpunkten der Kreise für die Niederlande gut sehen kann. Deshalb ist die numerische Größenordnung der Koordinaten im Ergebnis beliebig änderbar und die x- und y-Achse werden bei der Darstellung ausgeblendet. Die Qualität der Karte misst sich bei der Darstellung der Kennwerte allein an der möglichst guten Wiedergabe der realen Nachbarschafts- und Lagebeziehungen. Es kann allerdings vorkommen, dass nicht alle Lage- und Nachbarschaftsbeziehungen darstellbar sind. Die folgenden Beispiele nutzen wieder die Fläche der Staaten als Kennwert und zeigen die Staaten Mitteleuropas:

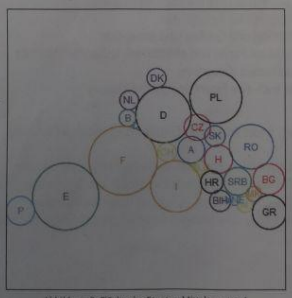


Abbildung 3: Fläche der Staaten Mitteleuropas 1

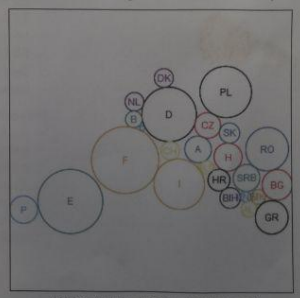
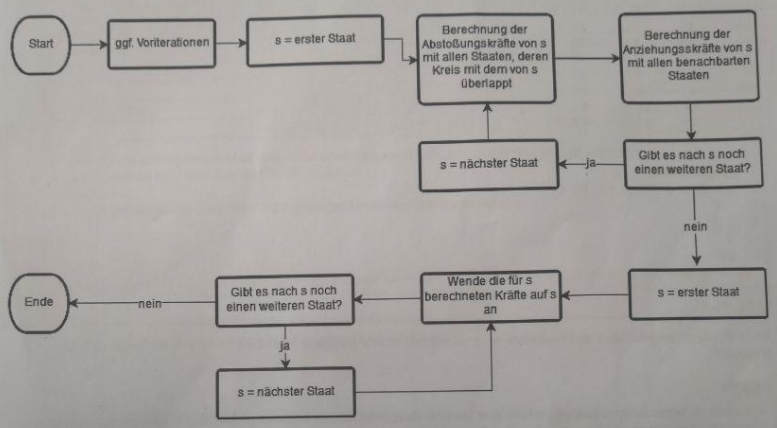


Abbildung 4: Fläche der Staaten Mitteleuropas 2

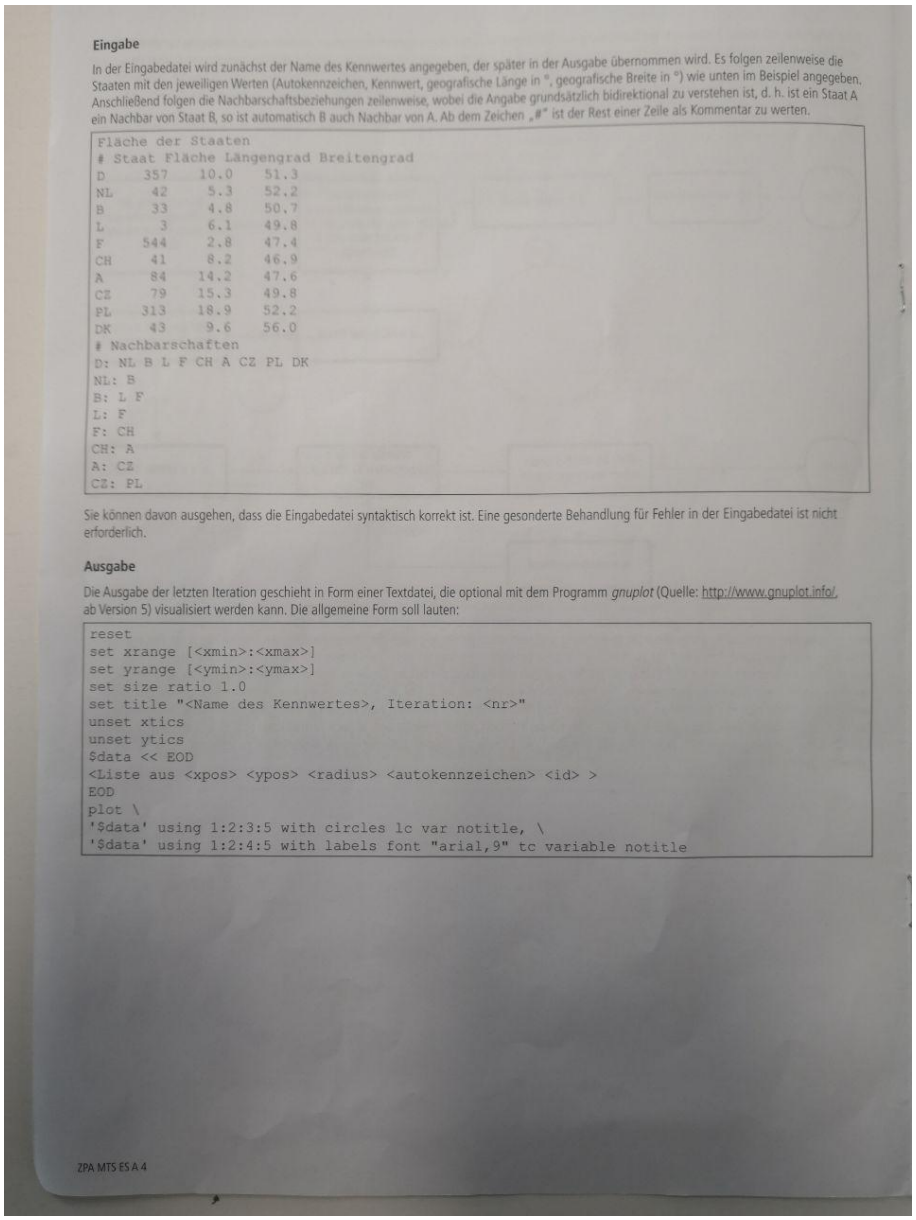
ZPA MTS ES A 2

Man erkennt in Abbildung 3, dass sich die Darstellung von Tschechien (CZ) mit den umgebenden Kreisen überschneidet. Anderseits berühren sich die umgebenen Staaten nicht, obwohl sie Nachbarn sind. Abbildung 4 ist eine alternative Darstellung, bei der bei der Bestimmung der Kreismittelpunkte einerseits mehr Wert auf das Vermeiden der Überschneidungen gelegt wurde, andererseits aber auch größere Abstände benachbarter Staaten (z. B. Polen und Deutschland) in Kauf genommen wurden. Die beiden Abbildungen zeigen, dass die geforderte einfache Darstellung mit Kreisen nicht immer exakt alle Nachbarschaftsbeziehungen darstellen kann.

Die wesentliche Schwierigkeit des Algorithmus ist die Bestimmung der Mittelpunkte der Kreise. Als Ansatz kann dazu ein Iterationsverfahren verwendet werden, dessen einzelne Iterationen folgender Idee entsprechen:



Die erwähnten Abstoßungskräfte ergeben sich dabei, wenn die zu zwei Staaten gehörigen Kreise sich überschneiden. Die Kräfte wirken immer in die vom anderen Kreismittelpunkt entgegengesetzte Richtung und sind umso größer je größer die Überschneidung der beiden Kreise ist. Die erwähnten Anziehungskräfte ergeben sich, wenn die zu zwei laut Vorgabe benachbarten Staaten gehörigen Kreise sich nicht berühren. Die Kräfte wirken immer zum anderen Kreismittelpunkt hin und sind umso größer je größer der Abstand der beiden Kreise ist. Wie im Programmablaufplan zu sehen ist, müssen diese Kräfte zuerst alle berechnet werden. Erst anschließend wendet man die Kräfte an, indem man die Kreis(mittelpunkt)e gemäß der Kräfte verschiebt. Es bietet sich zum Beispiel an, die Anziehungs- und Abstoßungskräfte zwischen zwei Kreisen je hälftig auf beide Kreise anzuwenden. Voriterationen – wie im Ablaufdiagramm erwähnt – können das Ergebnis positiv beeinflussen. Je nach Testfall kann es z. B. im Falle sehr starker Überschneidungen der Staaten in direkter Folge der Eingabedaten lohnen, die Staaten zunächst einmal auseinander zu schieben, um die Überschneidungen zu reduzieren.



Eingabe

In der Eingabedatei wird zunächst der Name des Kennwertes angegeben, der später in der Ausgabe übernommen wird. Es folgen zeilenweise die Staaten mit den jeweiligen Werten (Autokennzeichen, Kennwert, geografische Länge in °, geografische Breite in °) wie unten im Beispiel angegeben. Anschließend folgen die Nachbarschaftsbeziehungen zeilenweise, wobei die Angabe grundsätzlich bidirektional zu verstehen ist, d. h. ist ein Staat A ein Nachbar von Staat B, so ist automatisch B auch Nachbar von A. Ab dem Zeichen „#" ist der Rest einer Zeile als Kommentar zu werten.

```
Fläche der Staaten
# Staat Fläche Längengrad Breitengrad
D     357    10.0    51.3
NL     42     5.3    52.2
B      33     4.8    50.7
L       3     6.1    49.8
F     544     2.8    47.4
CH     41     8.2    46.9
A      84    14.2    47.6
CZ     79    15.3    49.8
PL    313    18.9    52.2
DK     43     9.6    56.0
# Nachbarschaften
D: NL B L F CH A CZ PL DK
NL: B
B: L F
L: F
F: CH
CH: A
A: CZ
CZ: PL
```

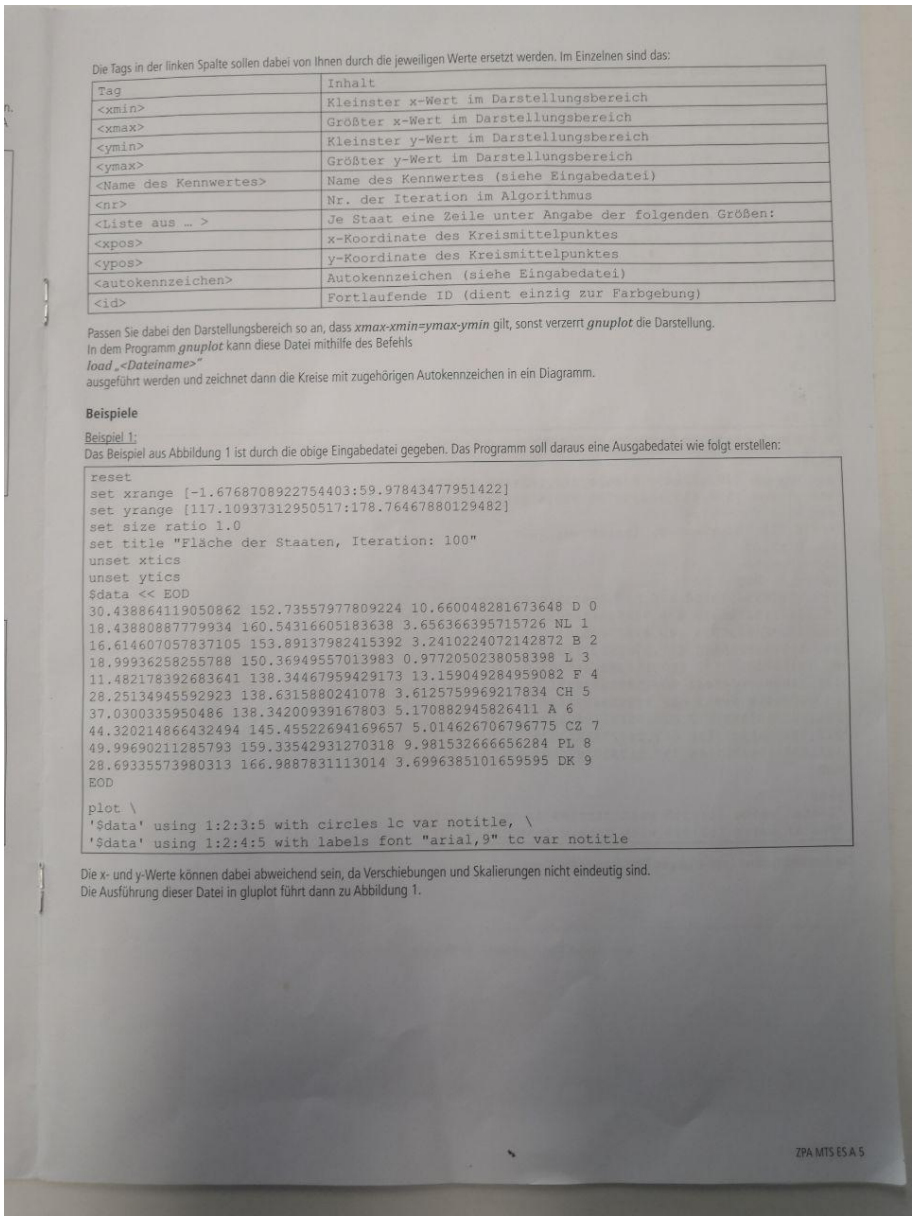
Sie können davon ausgehen, dass die Eingabedatei syntaktisch korrekt ist. Eine gesonderte Behandlung für Fehler in der Eingabedatei ist nicht erforderlich.

Ausgabe

Die Ausgabe der letzten Iteration geschieht in Form einer Textdatei, die optional mit dem Programm *gnuplot* (Quelle: http://www.gnuplot.info/, ab Version 5) visualisiert werden kann. Die allgemeine Form soll lauten:

```
reset
set xrange [<xmin>:<xmax>]
set yrange [<ymin>:<ymax>]
set size ratio 1.0
set title "<Name des Kennwertes>, Iteration: <nr>"
unset xtics
unset ytics
$data << EOD
<Liste aus <xpos> <ypos> <radius> <autokennzeichen> <id> >
EOD
plot \
'$data' using 1:2:3:5 with circles lc var notitle, \
'$data' using 1:2:4:5 with labels font "arial,9" tc variable notitle
```

ZPA MTS ES A 4



Die Tags in der linken Spalte sollen dabei von Ihnen durch die jeweiligen Werte ersetzt werden. Im Einzelnen sind das:

Tag	Inhalt
<xmin>	Kleinster x-Wert im Darstellungsbereich
<xmax>	Größter x-Wert im Darstellungsbereich
<ymin>	Kleinster y-Wert im Darstellungsbereich
<ymax>	Größter y-Wert im Darstellungsbereich
<Name des Kennwertes>	Name des Kennwertes (siehe Eingabedatei)
<nr>	Nr. der Iteration im Algorithmus
<Liste aus … >	Je Staat eine Zeile unter Angabe der folgenden Größen:
<xpos>	x-Koordinate des Kreismittelpunktes
<ypos>	y-Koordinate des Kreismittelpunktes
<autokennzeichen>	Autokennzeichen (siehe Eingabedatei)
<id>	Fortlaufende ID (dient einzig zur Farbgebung)

Passen Sie dabei den Darstellungsbereich so an, dass *xmax-xmin=ymax-ymin* gilt, sonst verzerrt *gnuplot* die Darstellung.
In dem Programm *gnuplot* kann diese Datei mithilfe des Befehls
load „<Dateiname>"
ausgeführt werden und zeichnet dann die Kreise mit zugehörigen Autokennzeichen in ein Diagramm.

Beispiele

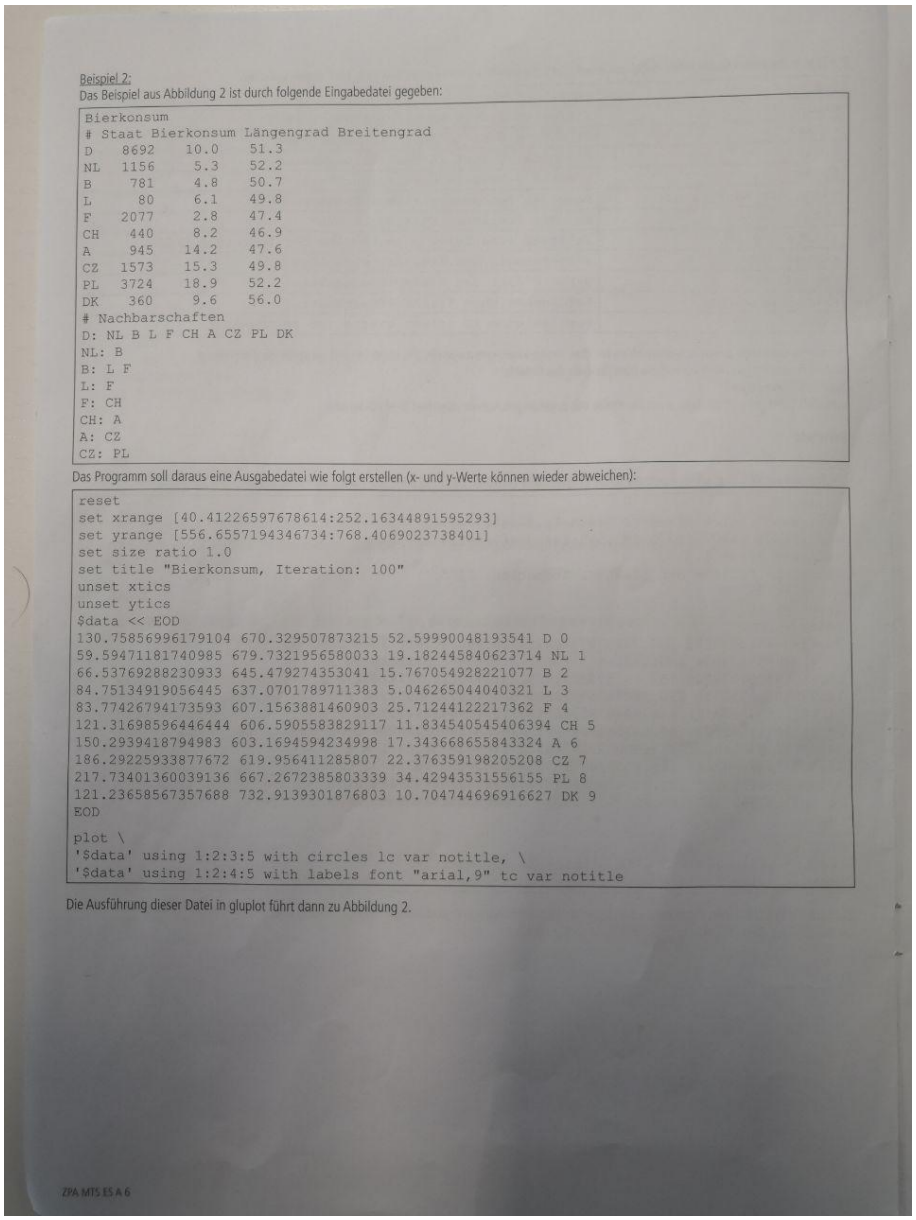
Beispiel 1:
Das Beispiel aus Abbildung 1 ist durch die obige Eingabedatei gegeben. Das Programm soll daraus eine Ausgabedatei wie folgt erstellen:

```
reset
set xrange [-1.676870892275440 3:59.97843477951422]
set yrange [117.10937312950517:178.76467880129482]
set size ratio 1.0
set title "Fläche der Staaten, Iteration: 100"
unset xtics
unset ytics
$data << EOD
30.438864119050862 152.73557977809224 10.660048281673648 D 0
18.43880887779934 160.54316605183638 3.656366395715726 NL 1
16.614607057837105 153.89137982415392 3.2410224072142872 B 2
18.99936258255788 150.36949557013983 0.9772050238058398 L 3
11.482178392683641 138.34467959429173 13.159049284959082 F 4
28.25134945592923 138.6315880241078 3.6125759969217834 CH 5
37.0300335950486 138.34200939167803 5.170882945826411 A 6
44.320214866432494 145.45522694169657 5.014626706796775 CZ 7
49.99690211285793 159.33542931270318 9.981532666656284 PL 8
28.69335573980313 166.9887831113014 3.6996385101659595 DK 9
EOD

plot \
'$data' using 1:2:3:5 with circles lc var notitle, \
'$data' using 1:2:4:5 with labels font "arial,9" tc var notitle
```

Die x- und y-Werte können dabei abweichend sein, da Verschiebungen und Skalierungen nicht eindeutig sind.
Die Ausführung dieser Datei in gluplot führt dann zu Abbildung 1.

ZPA MTS ES A 5



Beispiel 2:
Das Beispiel aus Abbildung 2 ist durch folgende Eingabedatei gegeben:

```
Bierkonsum
# Staat Bierkonsum Längengrad Breitengrad
D    8692    10.0    51.3
NL   1156     5.3    52.2
B     781     4.8    50.7
L      80     6.1    49.8
F    2077     2.8    47.4
CH    440     8.2    46.9
A     945    14.2    47.6
CZ   1573    15.3    49.8
PL   3724    18.9    52.2
DK    360     9.6    56.0
# Nachbarschaften
D: NL B L F CH A CZ PL DK
NL: B
B: L F
L: F
F: CH
CH: A
A: CZ
CZ: PL
```

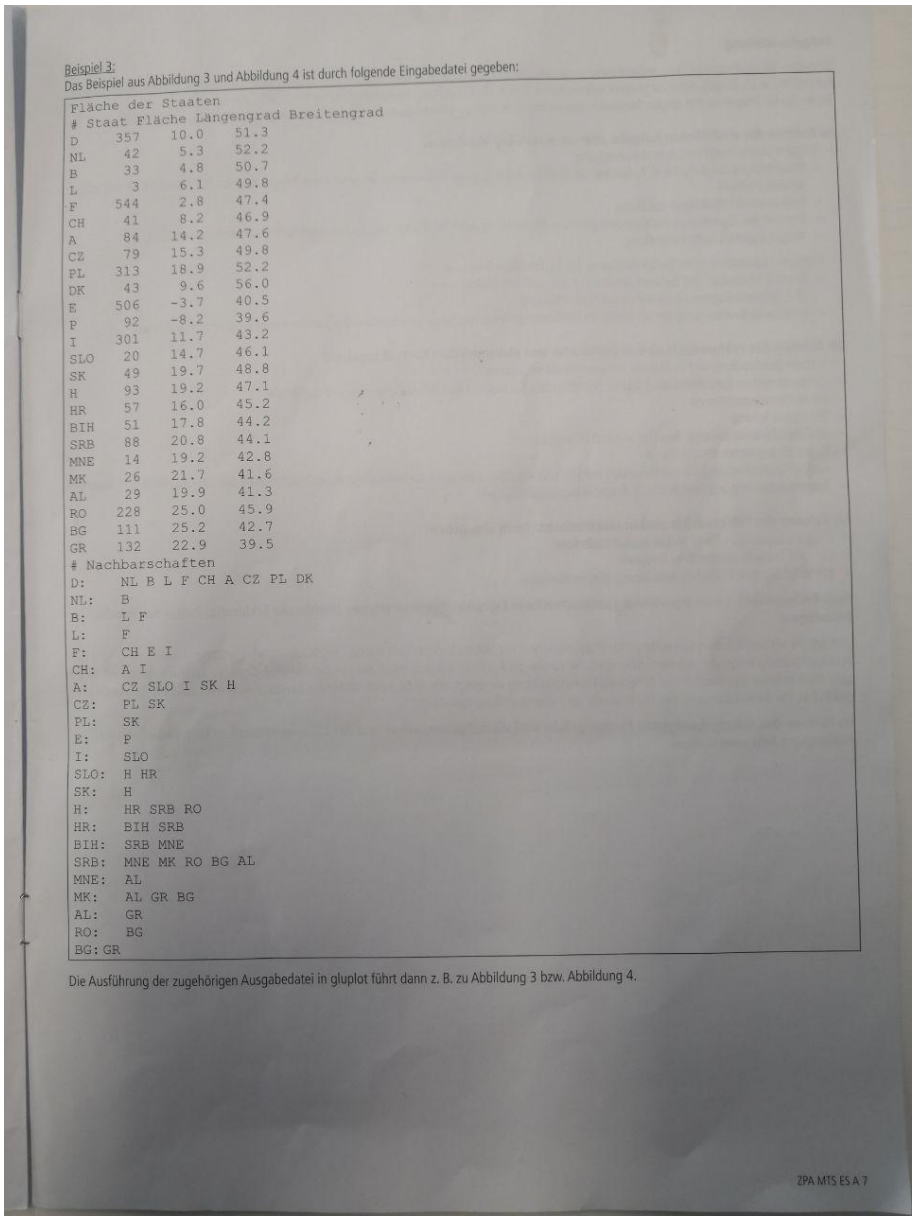
Das Programm soll daraus eine Ausgabedatei wie folgt erstellen (x- und y-Werte können wieder abweichen):

```
reset
set xrange [40.4122659767861 4:252.16344891595293]
set yrange [556.6557194346734:768.4069023738401]
set size ratio 1.0
set title "Bierkonsum, Iteration: 100"
unset xtics
unset ytics
$data << EOD
130.75856996179104 670.3295078732 15 52.59990048193541 D 0
59.59471181740985 679.7321956580033 19.182445840623714 NL 1
66.53769288230933 645.479274353041 15.767054928221077 B 2
84.75134919056445 637.0701789711383 5.046265044040321 L 3
83.77426794173593 607.1563881460903 25.71244122217362 F 4
121.31698596446444 606.5905583829117 11.834540545406394 CH 5
150.2939418794983 603.1694594234998 17.343668655843324 A 6
186.29225933877672 619.956411285807 22.376359198205208 CZ 7
217.73401360039136 667.2672385803339 34.42943531556155 PL 8
121.23658567357688 732.9139301876803 10.704744696916627 DK 9
EOD

plot \
'$data' using 1:2:3:5 with circles lc var notitle, \
'$data' using 1:2:4:5 with labels font "arial,9" tc var notitle
```

Die Ausführung dieser Datei in gluplot führt dann zu Abbildung 2.

ZPA MTS ES A 6

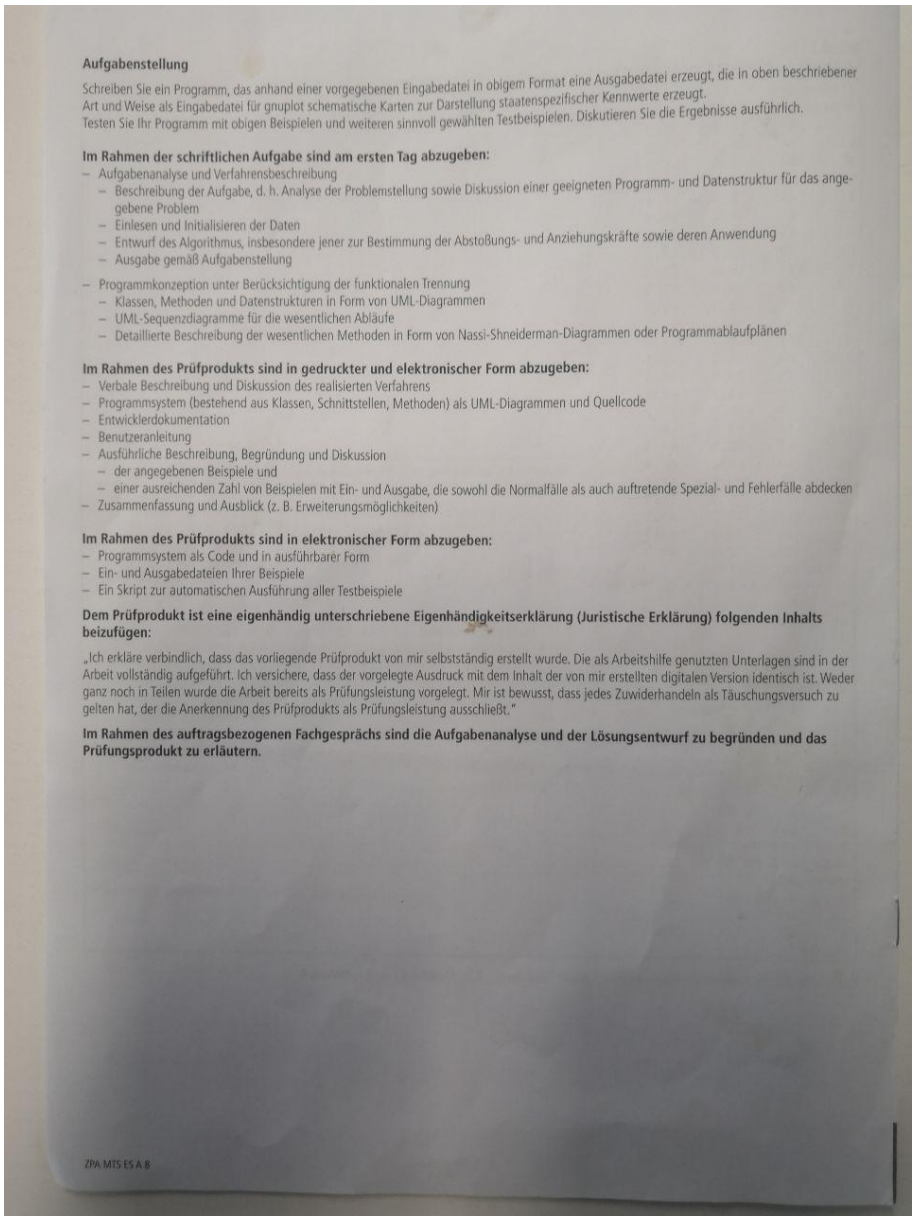


Beispiel 3:
Das Beispiel aus Abbildung 3 und Abbildung 4 ist durch folgende Eingabedatei gegeben:

```
Fläche der Staaten
# Staat Fläche Längengrad Breitengrad
D     357    10.0    51.3
NL     42     5.3    52.2
B      33     4.8    50.7
L       3     6.1    49.8
F     544     2.8    47.4
CH     41     8.2    46.9
A      84    14.2    47.6
CZ     79    15.3    49.8
PL    313    18.9    52.2
DK     43     9.6    56.0
E     506    -3.7    40.5
P      92    -8.2    39.6
I     301    11.7    43.2
SLO    20    14.7    46.1
SK     49    19.7    48.8
H      93    19.2    47.1
HR     57    16.0    45.2
BIH    51    17.8    44.2
SRB    88    20.8    44.1
MNE    14    19.2    42.8
MK     26    21.7    41.6
AL     29    19.9    41.3
RO    228    25.0    45.9
BG    111    25.2    42.7
GR    132    22.9    39.5
# Nachbarschaften
D:    NL B L F CH A CZ PL DK
NL:   B
B:    L F
L:    F
F:    CH E I
CH:   A I
A:    CZ SLO I SK H
CZ:   PL SK
PL:   SK
E:    P
I:    SLO
SLO:  H HR
SK:   H
H:    HR SRB RO
HR:   BIH SRB
BIH:  SRB MNE
SRB:  MNE MK RO BG AL
MNE:  AL
MK:   AL GR BG
AL:   GR
RO:   BG
BG: GR
```

Die Ausführung der zugehörigen Ausgabedatei in gluplot führt dann z. B. zu Abbildung 3 bzw. Abbildung 4.

ZPA MTS ES A 7



Aufgabenstellung

Schreiben Sie ein Programm, das anhand einer vorgegebenen Eingabedatei in obigem Format eine Ausgabedatei erzeugt, die in oben beschriebener Art und Weise als Eingabedatei für gnuplot schematische Karten zur Darstellung staatenspezifischer Kennwerte erzeugt.
Testen Sie Ihr Programm mit obigen Beispielen und weiteren sinnvoll gewählten Testbeispielen. Diskutieren Sie die Ergebnisse ausführlich.

Im Rahmen der schriftlichen Aufgabe sind am ersten Tag abzugeben:
- Aufgabenanalyse und Verfahrensbeschreibung
  - Beschreibung der Aufgabe, d. h. Analyse der Problemstellung sowie Diskussion einer geeigneten Programm- und Datenstruktur für das angegebene Problem
  - Einlesen und Initialisieren der Daten
  - Entwurf des Algorithmus, insbesondere jener zur Bestimmung der Abstoßungs- und Anziehungskräfte sowie deren Anwendung
  - Ausgabe gemäß Aufgabenstellung
- Programmkonzeption unter Berücksichtigung der funktionalen Trennung
  - Klassen, Methoden und Datenstrukturen in Form von UML-Diagrammen
  - UML-Sequenzdiagramme für die wesentlichen Abläufe
  - Detaillierte Beschreibung der wesentlichen Methoden in Form von Nassi-Shneiderman-Diagrammen oder Programmablaufplänen

Im Rahmen des Prüfprodukts sind in gedruckter und elektronischer Form abzugeben:
- Verbale Beschreibung und Diskussion des realisierten Verfahrens
- Programmsystem (bestehend aus Klassen, Schnittstellen, Methoden) als UML-Diagrammen und Quellcode
- Entwicklerdokumentation
- Benutzeranleitung
- Ausführliche Beschreibung, Begründung und Diskussion
  - der angegebenen Beispiele und
  - einer ausreichenden Zahl von Beispielen mit Ein- und Ausgabe, die sowohl die Normalfälle als auch auftretende Spezial- und Fehlerfälle abdecken
- Zusammenfassung und Ausblick (z. B. Erweiterungsmöglichkeiten)

Im Rahmen des Prüfprodukts sind in elektronischer Form abzugeben:
- Programmsystem als Code und in ausführbarer Form
- Ein- und Ausgabedateien Ihrer Beispiele
- Ein Skript zur automatischen Ausführung aller Testbeispiele

Dem Prüfprodukt ist eine eigenhändig unterschriebene Eigenhändigkeitserklärung (Juristische Erklärung) folgenden Inhalts beizufügen:

„Ich erkläre verbindlich, dass das vorliegende Prüfprodukt von mir selbstständig erstellt wurde. Die als Arbeitshilfe genutzten Unterlagen sind in der Arbeit vollständig aufgeführt. Ich versichere, dass der vorgelegte Ausdruck mit dem Inhalt der von mir erstellten digitalen Version identisch ist. Weder ganz noch in Teilen wurde die Arbeit bereits als Prüfungsleistung vorgelegt. Mir ist bewusst, dass jedes Zuwiderhandeln als Täuschungsversuch zu gelten hat, der die Anerkennung des Prüfprodukts als Prüfungsleistung ausschließt."

Im Rahmen des auftragsbezogenen Fachgesprächs sind die Aufgabenanalyse und der Lösungsentwurf zu begründen und das Prüfungsprodukt zu erläutern.

ZPA MITS ES A B