Entwicklung eines Softwaresystems

als Teil der Prüfung zur

Mathematisch-technischen Softwareentwicklerin (IHK)

Entwicklung einer Software Simulation

vorgelegt von: Ketli Cenollari

Abgabedatum: 17. Mai 2019

Prüflingsnummer: 101 20602

Aachen, 15. Mai 2019

Eidesstattliche Erklärung

Cenollari, Ketli
IHK Nummer: 101
Prüfungsnummer: 20602

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.

Aachen, den 17.Mai 2019

Ort, Datum	Unterschrift

Inhaltsverzeichnis

1 Aufgabe

1.1 Aufgabenanalyse

Die Aufgabe ist es, schemenhafte Landkarten zu erstellen,auf einer vorgegeben Kennwert basieren. Die reale Lage der Staaten wird dabei als Ausgangspunkt betrachtet- Man kann z.B. verschiedene Länder Europas als Kreise darstellen. Für einen Staat, der als Kreis dargestellt ist, gibt es einen Mittelpunkt, der auf eine x-Koordinate und y-Koordinate bestimmt ist, es gibt ein Radius und es gibt eine Liste mit Nachbarstaaten. Man kann die Qualität von einer Karte messen, indem man die Nachbarschaft und Lagebeziehungen mit der Ausgangsdarstellung vergleicht. Bei der Erstellung von einer Landkarte ist dabei zu beachten, dass Kreise mit einander sich überschneiden können. Unsere Aufgabe ist es einen Algorithmus zu implementieren um den neuen Mittelpunkt von dem Kreis zu berechnen. Dabei müssen zuerst die Abstoß,- und Anziehungskräfte berechnet werden.

Wir bekommen eine Eingabedatei die sich aus folgenden Daten besteht:

- Erste Zeile ist der Name des Kennwerts
- Zweite Zeile ist eine Kommentarzeile, der mit den Zeichen # anfängt
- Dann sind zeilenweise für jeden Staat die folgende Werte gegeben: Autokennzeichen, Kennwort, geographische Länge und Breite
- Dann gibt es noch eine Kommentarzeile
- Dann folgen zeilenweise die Nachbarschaften von einem Staat

Die Daten sollen eingelesen werden, gespeichert und am Ende muss eine Ausgabe Datei erstellt werden.

Unser Algorithmus muss wie gesagt den Kreismittelpunkt berechnen. Außerdem gibt es noch weitere Daten, die berechnet werden müssen, die dann in die Ausgabedatei mit ausgegeben werden. Das sind zB. die x-max und x-min Werte im Darstellungsbereich, y-min und y-max Werte im Darstellungsbereich. Dabei muss man aufpassen, dass x-max – x-min = y-max – y-min passt. Man muss noch die Anzahl von der letzten Iteration im Algorithmus ausgeben. Ebenfalls soll es eine eindeutig Id zur Durchnummerierung der Staaten hinzugefügt werden.

1.2 Entwurf des Algorithmus

Zunächst wird die Datei eingelesen und Zeile für Zeile interpretiert. Der erste Zeile wird als Kennwertname gespeichert und muss bei der Ausgabe Datei mit ausgegeben werden. Der Kommentar Zeile wird ignoriert. Für die folgenden Zeile wird jeweils ein Objekt von Typ Staat erzeugt und eine Liste die alle Staaten enthält. Die Zeile ab dem 2. Kommentar Zeile werden in eine List mit Strings für jeweils jeden Staat gespeichert, da die die Nachbarstaaten sind. Anschließend muss überprüft werden, ob die eingelesene Zeile valide ist und alle Informationen beinhaltet. Im Fehlerfall soll das Programm eine genaue und aussagekräftige Fehlermeldung ausgeben. In der Algorithmus Klasse wird eine Objekt von Typ Einlesen eingelesen und darauf werden die programmierte Methoden angewendet. Für den Algorithmus muss man erstmal die Staaten finden, mit dem unseres Ausgangsstaat eine Überlappung hat und auf diese Staaten wird der Funktion aufgerufen um die Abstoß Kräfte zu berechnen. Dies müssen nicht unbedingt Nachbarstaaten sein. Bei der Anziehungskräfte muss dann der Kraft von unseren Staat mit alle Nachbarstaaten berechnet werden. Nachdem man die Kräfte bestimmt hat, kann man die auf der Staat anwenden und somit den neuen Mittelpunkt berechnen. Bei der Ausgabe wird dann ein Objekt von Typ Algorithmus erzeugt der auf eine Einlesen Objekt die Methoden anwendet. Die Daten die für die Ausgabe notwendig sind werden gespeichert

und dann wird der Ausgabe zusammengefügt mittels ein StringBuilder und die Datei geschrieben.

2 Änderungen zum Ursprünglichen Konzept

Im Vergleich zum Konzept am Montag wurden folgende Änderungen vorgenommen:

3 Klassen und Methoden

3.1 Klassendiagramm

TODO Hier kommt das Bild.

3.2 Klasse Main

Diese Klasse dient als Grundklasse des gesamten Programmes. In dieser ist nur die Main-Methode enthalten, welche immer als erste Methode des gesamten Programmes aufgerufen wird.

3.3 Klasse Einlesen

In dieser Klasse findet das Einlesen und Speicherung von Daten der Eingabedatei, statt. Um die Daten zu speichern, wird eine Liste mit Staaten erzeugt. Die Methode einlesen() bekommt den Pfad des Dateies übergeben und liest mit einem Scanner die Datei zeilenweise durch. Bei der Aufgabenstellung steht, dass wir nicht überprüfen müssen ob die Datei syntaktisch korrekt ist. Aus diesem Grund wird hier auch nicht überprüft ob die erste Zeile z.B. eine Kommentarzeile oder eine leere Zeile ist. Die erste Zeile wird in einem String gespeichert und bei der Ausgabedatei als der Name des Kennwerts ausgegeben. Der zweite Kommentar wird überprüft ob es eine Kommentar Zeile ist und ignoriert. Jede andere Zeile, bis auf der nächste Kommentar Zeile wird dann in ein neues Staat Objekt gespeichert. Wenn alle die Staate erstellt sind, werden die dann in eine Liste von Typ Staat gespeichert. Dann der Rest von der Datei wird in das Attribut Nachbarn für jeden Staat gespeichert.

3.4 Klasse Ausgabe

3.5 Algorithmus

- In diese Klasse ist die Logik implementiert wie der neue Mittelpunkt für einen Kreis zu bestimmen ist.

- Die bekommt als Attribute ein Objekt von Typ einlesen, wo die eingelesene Daten gespeichert sind und auf diese Daten wird der Algorithmus angewendet. Es gibt eine Liste wo alle die Staaten gespeichert sind und ebenfalls noch vier double Werte, für den minimalen und maximalen Wert von x und y, die bei der Visualisierung der Landkarte benötigt werden.

- Es eine Methode berechneKraefte() die einen Staat als Übergabe bekommt und für diesen Staat liefert es eine Array von Typ Punkt zurück. Punkt ist eine von mir programmierte Klasse die eigentlich einen Vektor darstellt, mit eine x und y Koordinate. Ganz am Anfang bei der Methode werden zwei double Faktoren initilisiert, einmal für Abstoß und einmal für Anziehungskraft. Dann werden mittels eine for each Schleife alle die staaten durchgelaufen, und für jeden von diese Elemente wird dann der Kraft zwischen diese Elementen und den Staat berechnet. Es wird überprüft ob die Kreise sich mit ein ander überlappen, wenn das der Fall ist wird der Kraft berechnet und gespeichert. Sollten die Kreisen nicht überlappend und Nachbarn sein, dann wird der Anziehungskraft berechnet und ebenfalls gespeichert. Die genaue Ablauf der Methode kann in den Struktogramm gesehen werden.

- TODO insert bild

- Nachdem die Kräfte von einem Staat mit alle anderen berechnet und in eine Array gespeichert sind, müssen dann diese Kräfte auf der Staat angewendet werden um den Mittelpunkt zu verschieben. Dafür habe ich eine void methode wendeKraft() geschrieben. Hier werden alle die Staaten durchgelaufen, für jeden Staat die Kräfte berechnet, und alles ist in eine Liste mit Arrays von Typ Punkt gespeichert. Dann wird diese Liste durchgelaufen, und für jeden Element dann die dazugehörigen Werte von den Kraft vektor an die Mittelpunkt addiert. Dann wird für jeden Staat der neue mittelpunkt gesetzt. Die genaue Ablauf

kann in den Struktogramm gesehen werden.

- TODO insert bild

- Ebenfalls gibt es hier eine methode wendeKraft die aber einen int anzahl als Parameter bekommt, und das soll der Anzahl der Iterationen sein und hier wird nur die vorherige methode so oft aufgerufen wie der übergabe parameter sagt.

- TODO insert Bild

- Für die Berechnung von die ranges habe ich hier zwei Methoden geschrieben, wo erstmal die x Werte berechnet werden und danach die y Werte. Es werden alle die Staaten durchgelaufen werden, für jeden Staat wird der differenz von Mittelpunkt mit radius berechnet und der kleinste Wert davon wird als minimale Wert von x gespeichert. Das gleiche folgt auch für die Bestimmung von maximale x, nur werden die mittelpunkte mit dem Radius addiert und davon der gerößte Wert. Bei der Methode setyrange wird nochmal das gleiche gemacht, nur für y Werte.

- Bei der Aufgabe stand dass wir aufpassen müssen dass der x range gleich y range ist, sonst wird der Plot nicht richtig angezeigt. Dafür gibt es eine Methode setRange(), die überprüft die werten die vorher berechnet wurden und berechnet die nur dann neu wenn die Bedingung xmax - xmin = ymax-ymin nicht true ist. Sollte die differenz von x kleiner als die von y sein, werden die neue ranges von x mit die hälfte von der different y-x addiert, bzw. substrahiert. Das gleiche wird dann auch für y.

- TODO insert Bild

- In der Konstruktor von Klasse Algorithmus wird dann die setrange und wendeKraft aufgerufen.

4 Benutzeranleitung

4.1 ZIP Datei

In der zip-Datei befindet sich neben dieser Dokumentation (”Dokumentation.pdf”) eine Ordner Programm, welches zwei Ordner hat, einmal wo der Entwicklerdokumentation zu finden ist, und der andere wo das Programm zu finden ist. Der Ordner ”GrosseProg”beinhaltet eine ausführbare .jar Datei. Neben dieser ist noch ein Skript für Windows run.bat. Mit diesem Skript können alle Testfälle ausgeführt. Außerdem gibt es einen ïnOrdner. In diesem sind die IHK Beispiele als auch andere Testfälle zu finden. Die Ergebnisse können dann in den Ordner öut”gefunden werden. Im Ordner ßrc”befindet sich der geschriebene Quellcode.

4.2 Programmaufruf

Für das automatische Ausführen von Testfällen muss die beiliegende Batchdatei run.bat ausgeführt werden. Ansonsten kann das Programm auch über die Kommandozeile ausgeführt werden. Folgende Form muss vorliegen:

<Ordner der jar-Datei> groprog.jar input <datei>
Dabei dienen alle Felder mit <> als Platzhalter und müssen ersetzt werden. Dabei muss die Eingabedatei im input Ordner liegen. Die Entwicklerdokumentation kann mit Hilfe der index.html in dem Ordner Javadoc geöffnet werden.

5 Zusammenfassung und Ausblick

6 Entwicklungsumgebung

6.1 Hardware

Entwickelt und getestet wurde das Programm auf einem Dell Inc. Latitude E5470 Rechner mit Intel(R) Core(TM) i5-6300U Prozessor (2.40GHz). Arbeitsspeicher 8 GB.

6.2 Software

Das Programm wurde unter dem Betriebssystem Windows 7 Enterprise mit Eclipse IDE for Java Developers Neon Version 4.6.3 erstellt. Die verwendete Programmiersprache ist Java und kompiliert wurde es mit der JDK Version "JavaSE-1.8". Diese Dokumentation wurde in LaTeX mittels bibTex geschrieben. Zur Erzeugung des Diagramme wurde das Tool umlet benutzt und das Ergebnis als Bilddatei in LaTeX eingebunden.

7 Testfälle

7.1 IHK Beispiele

7.1.1 IHK Beispiel 1

7.1.2 IHK Beispiel 2

7.1.3 IHK Beispiel 3

7.2 Normalfälle

7.3 Sonderfälle

7.4 Fehlerfälle

8 Quellcode