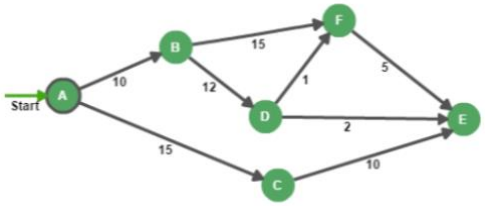
Abschlussprojekt im Modul 411



Zeitbedarf:	12 Lektionen
Hilfsmittel:	Eclipse mit Java, Wikipedia, youtube
Methode/Sozialform:	
Kompetenzen:	Modul 411
Abgaben:	- Dokumentation und Projekt
Quelle:	Stephan Dütsch
Links:	Für Gymniasasten und Lehrlinge geschriebene Einführung in die Algorithmik www.soi.ch/wiki/algorithms-intro/

Legende: Einzelarbeit, Partnerarbeit, Dokumentation, Code

Inhaltsverzeichnis

Ausgangslage:
Im Rahmen dieses Projekts sollen Sie möglichst viele der im Plan aufgeführten Kompetenzen demonstrieren und dokumentieren.
Der Übersicht halber sind diese Kompetenzen hier noch einmal aufgeführt:
Sie sollten in der Lage sein die dargestellten Kompetenzen zu belegen:

- Ein Java-Projekt in der von Ihnen gewählten Umgebung zu erstellen, dedizierte Java-Klassen mit Attributen zu definieren und zu erstellen, Werte von der Konsole einzulesen, zu verarbeiten und wieder auszugeben. Methoden zu definieren, welche aus den Attributen berechnete Werte zurück geben.

- in einem Programm Text von einem File zu lesen und zu verarbeiten
- Text auf ein File zu schreiben.
- Strukturierte Textzeilen zu zerlegen und in geeignete Datenobjekte abzufüllen.
- Ausgaben in Text Files durch Formatierung zu strukturieren.

- In einem Programm Zufallszahlen in verschiedenen Zahlenbereichen erzeugen zu lassen.
- Zu erklären, wozu das Seed bei einem Zufallszahlengenerator dient und Seeds sinnvoll zu setzen.
- Arrays von verschiedenen Datentypen erstellen und abfüllen zu können.
- *Den Zeitbedarf für einen Algorithmus für wachsende Datenmengen abzuschätzen und/oder zu messen.*

- Entweder
  - Rekursion in einem Programm zu erkennen
  - Rekursive Algorithmen gemäss einer Beschreibung umsetzen zu können.
  - Rekursion und allenfalls Backtracking für das Suchen von Lösungen von dazu geeigneten Problemen anwenden zu können.
- Oder
  - Algorithmen zu verwenden, welche (mehrfach) verschachtelte Schleifen verwenden (ohne Sortieren und binäre Suche)

- einen Algorithmus graphisch darzustellen (Struktogramm oder PAP)
- eine Applikation in sinnvolle Teilaufgaben zu zerlegen und so zu implementieren (Funktionen/Methoden)
- *den maximalen Speicherbedarf einer Applikation abzuschätzen.*

Vermerk: Das Projekt muss auf GitLab geführt, geplant, dokumentiert und versioniert werden!

Dokumentation

Die Projekte werden bezüglich der zu belegenden Kompetenzen dokumentiert. Das heisst, es wird für jede Kompetenz möglichst klar beschrieben, wo und wie sie im Projekt benötigt wurde.

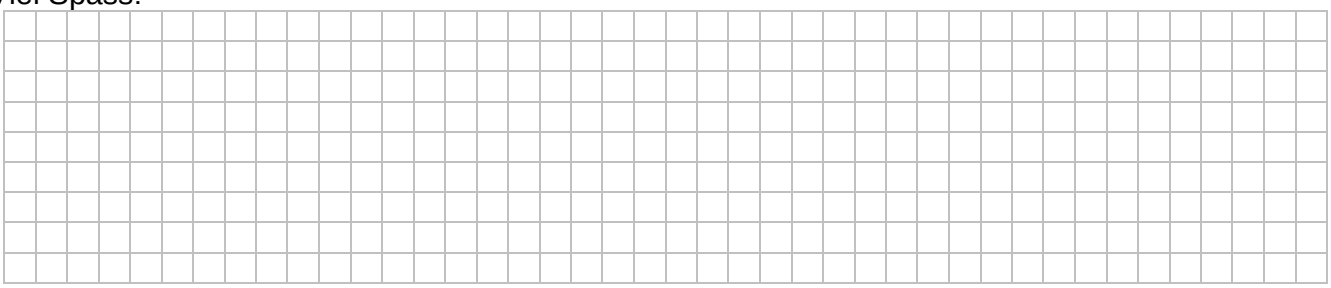
Bewertung
Die Bewertung ergibt sich aus :

1. den in der Dokumentation und dem Projekt belegten Kompetenzen
2. einer Gewichtung, welche sich am **Schwierigkeitsgrad** der Aufgabe orientiert
3. einer Gewichtung, welche sich aus dem **Fertigstellungsgrad** ergibt.

Projektideen:

Beschreibung der Aufgabe	**Schwierigkeitsgrad**
1. Eine einfache filebasierte Notenverwaltung mit Menusteuerung, Menüführung mit Java Spring Boot und Thymeleaf.	5.0
2. Game of Life (nach John Conway) mit Startzellen aus einem File und Konsolengrafik alternativ auch mit Java Spring Boot und Thymeleaf möglich.	5.0
- Erweiterung des Game of Life, so dass stabile Zustände erkannt werden.	5.5
3. Dominosteine aus einem File einlesen und alle möglichen passenden Reihenfolgen (nur lineare) erzeugen. Die Lösungen werden auf der Konsole (oder Java Thymeleaf und in einem File ausgegeben.	5.5
4. Das verflixte Flugzeugspiel mit einem Programm lösen.	6.0
5. Travelling Salesman.	6.0
6. Matt in zwei Zügen.	6.5

Viel Spass!



1.1 Filebasierte Notenverwaltung

1.1.1 Spezifikation

Das zu entwickelnde Programm soll es dem Benutzer erlauben, seine Noten in verschiedenen Fächern zu verwalten. Als Speichermedium dient ein Textfile.
Die Bedienung soll über ein Menu mit JavaFX erfolgen, welches die folgenden Punkte enthält:

- Auswahl und Laden des Files (Option: Wenn kein Filename angegeben wird, so wird das zuletzt verwendete geladen)
- Anzeigen aller Noten (ein Fach pro Zeile mit allen Noten und dem Durchschnitt)
- Erfassen eines neuen Fachs
- Erfassen einer neuen Note
- Ändern einer bestehenden Note
- Verlassen des Programms
- Das File soll automatisch bei jeder Änderung an den Daten gespeichert werden.

1.2 Game of Life (nach John Conway)

Auf Wikipedia finden Sie eine Beschreibung des Game of Life
http://de.wikipedia.org/wiki/Conways_Spiel_des_Lebens .

1.2.1 Spezifikation Grundaufgabe

Ihr Programm soll das Game of Life http://de.wikipedia.org/wiki/Conways_Spiel_des_Lebens implementieren.

- Die ganze Konsole oder alternativ mit Java Spring Boot, ist das Spielfeld.
- Der Zustand der Zellen beim Start soll von einem File eingelesen werden.
- Das Alter der Zellen soll optional (durch Tastendruck) als Farbe dargestellt werden.
- Anfänglich soll ein Lebenszyklus eine Sekunde dauern.
- Die Dauer eines Lebenszyklus soll über die Tastatur verkürzt oder verlängert werden können.

1.2.2 Spezifikation Erweiterung

Die Simulation soll stoppen, wenn ein stabiler Zustand erreicht wird. Stabil heisst:

- Keine Zelle ändert von einem Zyklus zum nächsten ihren Zustand
  oder
- Die “Welt” als Ganzes hat nach einer bestimmten Anzahl Zyklen immer wieder den gleichen Zustand, das heisst es gibt eine Oszillation. Beschränken Sie sich zunächst darauf, zu erkennen, wenn die Oszillationsfrequenz 2 ist. Das heisst nach jedem zweiten Zyklus wiederholt sich der Zustand.

1.3 Dominokette

Es soll ein Programm geschrieben werden, welches Dominosteine in einer linearen Kette anordnet, so dass immer zwei zusammenstossende Flächen die gleiche Augenzahl aufweisen.

1.3.1 Backtracking

Um diese Aufgabe zu lösen, sollten Sie Rekursion und Backtracking verwenden. Andere Lösungsvarianten gibt es zwar, sie sind aber eher schwerfällig und unhübsch.

1.4 Das verflixte Flugzeugspiel

Dieses Spiel besteht aus 9 Karten, bei denen auf den vier Rändern Heck oder Bug eines Flugzeug in verschiedenen Farben abgebildet sind.

Diese Karten sollen in einem 3 x 3 - Raster so angeordnet werden, dass auf nebeneinander liegenden Karten jeweils ein komplettes Flugzeug in einheitlicher Farbe entsteht.
Ihre Applikation soll die 9 Karten von einem dafür geeigneten Textfile einlesen, Lösungen suchen und diese auf der Konsole (oder Java) und auf einem File oder HTML ausgeben.

1.5 Travelling Salesman

Aus einem File werden die Koordinaten eines Startpunkts sowie von einer Serie von zu besuchenden Orten eingelesen. Die Applikation soll dann den kürzesten Weg finden, der vom Startpunkt zu **allen** Orten und wieder zurück zum Startpunkt führt.
Als Distanz jedes Streckenteils wird die Luftlinie zwischen zwei Orten verwendet.
Die drei besten Reiserouten sollen auf der Konsole (oder JavaFX) und in einem Textfile ausgegeben werden (Liste in der Reihenfolge der besuchten Orte mit Distanz).
Links:
https://www-m9.ma.tum.de/games/tsp-game/index_de.html
https://www.java-forum.org/thema/dijkstra-algorithmus.95643/
https://www.vogella.com/tutorials/JavaAlgorithmsDijkstra/article.html

https://developers.google.com/optimization/routing/tsp
http://mathworld.wolfram.com/TravelingSalesmanProblem.html

1.6 Matt in zwei

Auf den Rätselseiten von Zeitschriften findet man oft Schachrätsel, welche eine Position vorgeben und es ist dann ein Zug von Weiss so zu suchen, dass unabhängig von der Antwort von Schwarz, Weiss im zweiten Zug ein Matt erreichen kann.
Die Applikation soll eine solche Schachposition von einem File einlesen können und die Lösung finden.
Wenn die speziellen Regeln wie Rochade, Bauernverwandlung, Schlagen en passant nicht umgesetzt sind, kann trotzdem der Skalierungsfaktor 6.0 erreicht werden.
Bei Umsetzung aller Regeln kommt der Faktor 6.5 zur Anwendung.

http://jug-karlsruhe.de/assets/slides/szoerner_jugka2013_architekturentwurf_schach_deploy.pdf