Ausgangslage

Ein eher künstlerisch orientierter Kollege möchte Texte (Bücher, Filmskripte, etc.) in Wortwolken darstellen. Leider hat er keine Ahnung, wie er für die Wortwolken brauchbare Informationen aus den Texten entnehmen kann, worauf er dich um Hilfe bittet.

Theoretische Grundlagen

Im Rahmen dieser Aufgaben soll eine Applikation entwickelt werden, womit einfache Textdateien (`.txt`) ausgewertet werden können. Unter «Auswertung» ist zu verstehen, dass Informationen anhand einer Textdatei generiert werden, die nicht direkt in der Datei selbst enthalten sind. Einige Beispiele von solchen Infos wären:

- Anzahl an einzigartigen Wörtern
- Gesamtanzahl an Wörtern
- Meistvorkommendes Wort
- …

Zusammengefasst muss dieses Textanalysetool die folgenden Anforderungen erfüllen:

- Lesen von Textdateien von beliebiger (bzw. unbekannter) Grösse.
- Für jedes Wort in der Textdatei die Anzahl an Vorkommnisse ermitteln.
- Erstellen einer neuen Textdatei am gleichen Ort wie die gelesene Textdatei mit dem Namen «`<name der gelesenen Textdatei>_evaluation.txt`».
- Der Inhalt der generierten Textdatei mit den Ergebnissen entspricht dem Muster:

```
[Datum / Uhrzeit] [Name der gelesenen Datei]
-------------------------------------------
Number of unique words:    [TAB]    [Anzahl]
Total number of words:     [TAB]    [Anzahl]
Most common word:          [TAB]      [Wort]
-------------------------------------------
[Wort]    [TAB]    [Anzahl]
[Wort]    [TAB]    [Anzahl]
...
```

- Je nachdem, wie die Anforderungen implementiert werden, variiert die Anzahl an ermittelten Wörtern in der gleichen Textdatei von Person zu Person. Aus diesem Grund wird eine Textdatei zur Verfügung gestellt, um allfällige Differenzen untereinander feststellen zu können. Für Testzwecke sollten zusätzlich auch eigene Textdateien verwendet werden.
- Am Ende werden zufällige Kandidaten ihre Arbeit präsentieren. Dazu gehört eine Demonstration, eine Erklärung der wichtigsten Komponenten im Quellcode und eine Reflexion.

Nach der Implementierung der Grundanforderungen besteht die Möglichkeit für selbstdefinierte Erweiterungen. Einige Ideen dafür wären:

- Weitere Auswertungen (längstes Wort, Anzahl an Zeichen, sich reimende Wörter, etc.).
- Sortieren der Einträge der Ausgabedatei nach Anzahl oder lexikographisch.
- Bedienung des Analysetools über eine Konsolenapplikation mit Ein- und Ausgaben.
- Suche nach bestimmten Wörtern und wo sich diese in der gelesenen Textdatei befinden.
- Herausfiltern von sogenannten *Stop Word*s anhand einer *Stop Word* Liste.

Praktische Umsetzung

Neben den bereits erwähnten funktionalen Anforderungen muss die Implementierung auch gewisse formelle Aspekte erfüllen. Dazu gehört, ist aber nicht limitiert auf:

- Eingaben und Ausgaben über die Konsole sind sparsam und gezielt einzusetzen. Dies mit der Absicht, dass nicht im gesamten Projekt in jeder Klasse Scanner-Objekte und `System.out.println()` Aufrufe verstreut sind.
- Alles, was direkt mit dem Umgang von Dateien zu tun hat, ist in einer eigenen Klasse zu implementieren.
- Generell gilt zu beachten, dass Codefragmente, die nicht direkt miteinander zu tun haben, auch keinen Grund haben, sich in der gleichen Methode oder gar in der gleichen Klasse aufzuhalten.
- Die Starter-Klasse ist wie üblich kurz zu halten.
- Eine saubere und sinnvolle Klassenstruktur wird erwartet.

Oder kurzgefasst, würde diese Aufgabe benotet werden, dann würde nur das Erfüllen der funktionalen Anforderungen in der Note 3.0 resultieren.

Bei Interesse und mit vorheriger Absprache darf diese Aufgabe mittels einer anderen Programmiersprache gelöst werden. Dabei gelten aber dieselben Richtlinien was die Qualität des Quellcodes angeht.

Hilfestellungen

Wie es auch schon der Fall war, Tipps zum Lösen dieser Aufgabe bzw. darin enthaltene Teilaufgaben sind unten mit weisser Schriftfarbe vorhanden.

Beschreibung	Tipp
Klassen	
Datum und Uhrzeit	
Lesen / Aufteilung der Datei	
Erkennen von Wörtern	
Zählen von Wörtern	