Lehrstuhl für Algorith.
und Datenstrukturen
Prof. Dr. Hannah Bast
Axel Lehmann

Algorithmen und Datenstrukturen
(Informatik II) SS 2013

http://ad-wiki.informatik.uni-freiburg.de/teaching

UNI FREIBURG

Übungsblatt 4

Abgabe bis Dienstag, den 14. Mai um 16:00 Uhr

Für dieses Übungsblatt sollen Sie wieder etwas implementieren. Auf dem Wiki finden Sie für beide Aufgaben einen Vorschlag für das Klassendesign. Sie müssen diesem Vorschlag nicht unbedingt folgen. Was Sie aber auf jeden Fall machen sollen: (1) Die wesentliche Funktionalität sollte in einer separaten Klasse stehen, nicht in der *main* Funktion. (2) In der Klasse eine sinnvolle Aufteilung in (jeweils nicht zu große) Methoden. (3) Für jede nicht-triviale Methode einen Unit Test.

Aufgabe 1 (10 Punkte)

Laden Sie sich die drei auf dem Wiki verlinkten Dateien *acted-in.tsv*, *has-won.tsv* und *married-to.tsv* herunter. (Committen Sie die Dateien *nicht* in unser SVN, das gibt 100 Punkte Abzug.)

Schreiben Sie ein Programm *FreeBaseAnalyzerMain*, das diese drei Dateien einliest und Folgendes berechnet: alle verheirateten Paare (gemäß der Datei *married-to.tsv*), bei denen beide schauspielern (gemäß der Datei *acted-in*) und beide schon einmal den selben Preis gewonnen haben (gemäß der Datei *has-won.tsv*).

Geben Sie für jedes solche Paar eine Zeile aus, mit den beiden Namen, sowie den Namen des Preises, den beide schon ein mal gewonnen haben (ein solcher Preis reicht, falls es mehrere gibt). Beispiel: Ellen DeGeneres und Portia de Rossi, beide Satellite Award for Best Actress.

Benutzen Sie dabei assoziative Arrays aka Maps, wie in der Vorlesung erklärt. Sie werden sie für diese Aufgabe an mehreren Stellen brauchen. Die Gesamtanzahl der Operationen auf diesen Maps sollte linear in der Gesamtanzahl Zeilen der drei Eingabedateien sein.

Aufgabe 2 (10 Punkte)

Schreiben Sie eine Klasse, die n gegebene Zahlen aus einem *beliebigen* Bereich, dabei aber höchstens m verschiedene, mit der in der Vorlesung erklärten *MapCountingSort* Methode sortieren kann.

Schreiben Sie ein Programm, das für gebenene n und m eine zufällige Eingabe der beschriebenen Art erzeugt. Sortieren Sie dieses Feld dann ein mal mit Ihrem *MapCountingSort* und ein mal mit dem eingebauten *sort* von Java bzw. C++ und messen Sie jeweils die Laufzeit. Für wie kleine m ist *MapCountingSort* besser?

[bitte wenden]

Committen Sie Ihre Lösung in unser SVN, in einen neuen Unterordner *uebungsblatt_04*. Schreiben Sie für jede nicht-triviale Methode einen Unit Test, und stellen Sie sicher, dass auf Jenkins alles (insbesondere *checkstyle*) fehlerfrei durchläuft. Für jeden Checkstyle Fehler sind 10 Klimmzüge fällig oder 2 Stunden RTL2 schauen.

Committen Sie in diesem neuen Unterordner außerdem wie beim letzten Mal eine Textdatei *erfahrungen.txt*. Beschreiben Sie dort in ein paar Sätzen Ihre Erfahrungen mit diesem Übungsblatt und der Vorlesung dazu. Insbesondere: Wie lange haben Sie ungefähr gebraucht? An welchen Stellen gab es Probleme und wieviel Zeit hat Sie das gekostet?