

Übungsblatt #07

Bearbeitungszeitraum: bis Donnerstag, 10.12.2009

Was will uns dieses Übungsblatt sagen?

1. Abgesehen von der im letzten Übungsblatt betrachteten Stärke von Perl im Umgang mit Textdateien, eignet sich Perl zudem auch hervorragend zur Auswertung von Daten(massen).
2. Im Unterschied zu manchen Office-Programmen ist Perl bei der Anzahl der einlesbaren Zeilen nicht auf maximal 65.535 Zeilen beschränkt.
3. Über diese „Nicht-Beschränkung“ hinaus bietet Perl aber auch eine hocheffiziente interne Datenhaltung, so dass die Bearbeitungsgeschwindigkeit deutlich schneller ist, als bei Office-Programmen. Manches würde ohne erheblichen Aufwand mit Office-Tools gar nicht gehen.
4. Listen und Hashes können in Perl ausschließlich Skalare aufnehmen.
5. Um komplexere Strukturen, wie beispielsweise einen Hash von Hashes, eine Liste von Listen usw., aufzubauen, kann mit Referenzen (Zeigern) auf Listen bzw. Hashes gearbeitet werden, denn
6. eine Referenz ist aus Sicht von Perl wieder „nur“ ein Skalar.
7. Wenn Ihr via Copy/Paste das Einlesen der Daten wiederverwendet, lassen sich die einzelnen Aufgaben mit nur sehr wenigen Zeilen lösen, und gerade das macht ja den Reiz von Perl aus.

Perl als Datenknecht

Die Sprache Perl ist nicht nur im Umgang mit Text(datei)en sehr mächtig, sondern vor allem auch bei der Verarbeitung sehr großer Datenmengen.

Auf unserer Homepage findet Ihr die Datei „buchungen.csv“, die aus 500.000 Zeilen besteht. Wir interpretieren dabei jede Zeile als Buchung eines Hotelaufenthalts auf einer einsamen Insel im Jahr 2009. Dann haben die (durch ein Semikolon voneinander getrennten) Spalten einer Zeile die folgende Bedeutung:

- Buchungscode (eindeutig, dient als Schlüssel),
- Name (zusammengesetzt aus Nach- und Vorname)
- Herkunftsland des Touristen,
- Dauer des Aufenthalts in Tagen

Schreibt zu den nachfolgenden Aufgaben jeweils ein kleines Perl-Skript, das Euch aus den Daten die gewünschte(n) Information(en) gewinnt und aufbereitet.

a) Doppelte Buchungscodes

Die erste Spalte stellt den Buchungscode dar, der eigentlich (aufgrund seiner Schlüsseleigenschaft) eindeutig sein sollte. Leider wurden aber aufgrund eines Fehlers einige Codes mehrfach vergeben. Lasst Euch durch ein Perl-Skript anzeigen,

- welche Codes dies sind,
- in welchen Zeilen sie zum wiederholten Male auftreten und
- wie oft sie vorkommen.

(zur Kontrolle: mehr als 3 mal kommt kein Code vor!)

b) Wer ist Reiseweltmeister?

Nun möchten wir eine kleine Statistik haben: wieviele Touristen kamen insgesamt aus den einzelnen Ländern?

Schreibt also ein Skript, das zunächst für jedes Land ausgibt, wieviele Tage insgesamt die Touristen aus diesem Land vor Ort waren. Dabei soll die Ausgabe *alphabetisch sortiert nach Ländernamen* erfolgen.

Wenn das soweit funktioniert, ist es möglich, Euer Programm mit minimalem Aufwand derart umzubauen, dass es Euch dieselbe Ausgabe liefert, allerdings *absteigend sortiert nach der Anzahl der Tage* (dann hat man eine „HighScore-Liste“). Also: ran an's Werk :-)

c) Varianz-Analyse

Keine Panik - so „stochastisch“, wie es die Überschrift suggeriert, ist diese Aufgabe nicht. Aber wir wollen nun dennoch ein kleines Perl-Skript für eine typische Auswertung unserer Daten basteln.

Schreibt ein Perl-Skript, das zwei Parameter d und r übergeben bekommt. Dabei steht r für diejenige Anzahl an Aufenthaltstagen, die wir für unsere momentane Auswertung als Referenzwert verwenden wollen, und der zweite Parameter d ist die Mindestabweichung davon in Tagen. Schreibt ein Skript, das aufgrund dieser beiden Parameter alle diejenigen Zeilen (=Aufenthalte) ausgibt, bei denen die Dauer des Aufenthalts um mindestens d Tage von unserem Referenzwert r abweicht (egal ob nach oben oder unten).

Alles klar? Nein? Gut, dann erklären wir die Aufgabe mit folgendem

Beispiel: Wenn Ihr Euer Skript mit $d=4$ und $r=14$ aufruft, dann sollen alle Zeilen angezeigt werden, die für Aufenthalte stehen, die weniger als 10 ($=r-d$) oder mehr als 18 ($=r+d$) Tage gedauert haben.

d) Komplexe(re) Datenstrukturen in Perl

Nun schreiben wir ein Perl-Skript, das als Abfrage-Tool für die Buchungsdaten verwendet werden kann.

Lasst zunächst einmal die gesamte Datei einlesen und erstellt dabei eine Liste von Listen: jede eingelesene Zeile soll am Trennzeichen „;“ aufgeteilt werden, und die daraus resultierende Liste wird (als Referenz) in eine andere Liste gesteckt.

Lasst den Benutzer dann eine Zahl n mit $1 \leq n \leq 500.000$ eingeben. Euer Skript sollte daraufhin den n . Eintrag Eurer Liste (d.h. die dort referenzierte Liste mit den Zeileneinträgen) ausgeben.

Wenn das funktioniert, ist es zwar schön, aber ein solches Tool ist nicht sehr realistisch. Deshalb kommt nun die eigentliche Aufgabe:

Andert Euer Programm so um, dass die Datenstruktur etwas anders ist: anstelle der Liste von Listen soll es nun einen Hash von Listen geben, dessen Schlüssel die Buchungscodes sind und dessen Werte der Rest der zu den Codes gehörenden Daten (in Form einer Listenreferenz darauf) sind. (Dass - wie in Aufgabe 1 gesehen - einige Codes mehrfach vorkommen, sei an dieser Stelle mal egal.)

Der Benutzer des Skripts soll einen Buchungscodes eingeben können, und das Perl-Skript gibt dann die zu diesem Code gespeicherten Informationen aus.

e) Und jetzt dasselbe (in Gedanken) mit Office Software...

Wenn Ihr diese Aufgaben bis hierhin gelöst habt, dann solltet Ihr spätestens jetzt von den Möglichkeiten, die Euch mit Perl zur Datenauswertung zur Verfügung stehen, überzeugt sein.

Wenn nicht, dann überlegt Euch mal, wie Ihr diese Aufgaben in Office (z.B. Excel) gelöst hättet - insbesondere dann, wenn Ihr mit Excel 2003 und abwärts oder OpenOffice arbeitet, denn diese Versionen sind noch auf 65.535 Zeilen beschränkt!

Und abgesehen davon: wie hättet Ihr Aufgabe a) in Office-Tools lösen können?



Palmen auf der in diesem Übungsblatt betrachteten einsamen Insel!