Blatt 11

Abgabe bis: 26. Januar 2012

14. dups.c (10P)

Schreibt ein Programm `dups`, das doppelte Dateien in einem Verzeichnisbaum findet und auflistet. Es soll ab einem definierten Startverzeichnis **den Verzeichnisbaum rekursiv durchlaufen**, dabei aber ‘‘soft links’’ ignorieren. Ohne Argumente aufgerufen, soll im aktuellen Verzeichnis begonnen werden, andernfalls nacheinander in jedem der uebergebenen Verzeichnisse.

Euer Programm soll (potenziell) gleiche Dateien anhand ihrer Dateigroesse finden. Packt dazu die Pfadnamen in eine **Hashtabelle**, deren Hashkey von der Dateigroesse abhaengt. Loest Kollisionen im einfachsten Fall durch Verkettung.

Am Ende soll euer Programm eine Liste aller Pfadnamen ausgeben, die potentiell Duplikate sind.

Tatsaechliche Duplikate koennen dann gefunden werden indem man z.B. die Dateiinhalte dieser Kandidaten vergleicht. Dieser Schritt ist nicht mehr Teil eurer Aufgabe, darf aber natuerlich trotzdem freiwillig gemacht werden.

Die Funktionsweise und Verwendung von Hashtabellen ist in der englischen und deutschen Wikipedia ausfuehrlich erklaert. Relevante Manpages fuer die Aufgabe sind: *opendir(3)*, *readdir(3)*, *closedir(3)*, und *lstat(2)*. Eine einfache Hashfunktion ist ‘`size % HASHSIZE`’.

Zum Einreichen eurer Loesung:

```
submit ss1 14 team [notes] dups.c
```

Fragen zur Selbstkontrolle

- Wie ist das Dateisystem bei Unix organisiert?
- Welche Informationen enthaelt die Inode? Welche nicht?
- Was ist eine regulaere Datei bei Unix?
- Wie unterscheidet sich das Lesen aus einer Datei vom Lesen von der Standardeingabe?
- Was sind *hard links* und was *soft links*?
- Welche Probleme koennen sich bei rekursiven Baumdurchlaeufen ergeben?

- Wie gehen rekursiv arbeitende Unix-Tools damit um? Was sagt POSIX dazu?
- Welche Rechtegruppen werden in Unix unterschieden?
- Welche Rechte benoetigt man fuer welche Operationen bei Verzeichnissen?
- Was ist der Vorteil von Hashtabellen?
- Wie findet man eine gute Groesse fuer die Hashtabelle?