Universität Ulm - Abteilung Angewandte Informationsverarbeitung

Dr. Johannes Mayer und Hans Braxmeier, Wintersemester 2003

13. Übungsblatt zur Vorlesung Allgemeine Informatik III

Abgabetermin: Dienstag, 10.02.2004

Temporäre Dateien *(10 Punkte)*

Implementieren Sie ein Programm, mit dem Datensätze fester Länge (Rekords) einer Datei hinzugefügt, gesucht bzw. gelesen, verändert und gelöscht werden können. Jeder Datensatz soll dabei folgende Informationen eines Studenten enthalten: Matrikelnummer, Name und Studienfach.

Beachten Sie beim Implementieren folgende Punkte:

- Existiert die Datei noch nicht bzw. enthält die Datei keinen Datensatz, so soll bei Lese-, Änderungs- oder Löschoperationen eine Warnung ausgegeben werden.
- Existiert die Datei noch nicht und wird ein Datensatz hinzugefügt, so soll die Datei (falls möglich) neu angelegt und der Datensatz hinzugefügt werden.
- Jeder Datensatz soll mit einer einzigen Schreiboperation hinzugefügt werden, d.h. es soll jeweils der komplette Datensatz geschrieben werden und nicht nur Teile davon.
- Alle weiteren Datensätze sollen nach Name sortiert in die Datei eingefügt werden.
- Existiert bereits ein Datensatz mit diesem Namen, so soll der alte Datensatz durch den neuen Datensatz ersetzt werden.
- Bei allen Operationen darf sich höchstens ein Datensatz im Arbeitsspeicher des Rechners befinden. Arbeiten Sie deshalb bei allen Operationen, die mehrere Datensätze benötigen, mit temporären Dateien.
- Verwenden Sie zum Lesen und Schreiben von Datensätzen nur die Funktionen `read()`, `write()` und `lseek()`
- Beim Suchen, das allen Operationen zugrunde liegt, soll stets der Studentenname als Suchkriterium dienen.
- Verwenden Sie Binäres Suchen, um einen Datensatz zu finden (Dateigröße und die Größe eines Datensatzes können bestimmt werden...).

Anmerkungen: Beachten Sie zu temporären Dateien die Hinweise in den Übungen und den Teil im Skript (unter *unlink()*). Temporäre Dateien gehören in das Verzeichnis */tmp*. Einen *eindeutigen* Namen für eine temporäre Datei kann man mittels der Prozess-ID (*getpid()*) und *sprintf()* bzw. via *mktemp()* (siehe Beispiel im Skript) erzeugen. Die Funktion *mkstemp()* darf nicht verwendet werden!

Beispiel:

```
thales$a.out

(a)dd, (f)ind, (c)hange, (d)elete, (p)rint, (q)uit? a

Matrikelnummer? 12345
Name? Markus
Fach? Löffler

(a)dd, (f)ind, (c)hange, (d)elete, (p)rint, (q)uit? a

Matrikelnummer? 5643
Name? Katrin
Fach? Mühlhaus

(a)dd, (f)ind, (c)hange, (d)elete, (p)rint, (q)uit? p
5643, Katrin, Mühlhaus
12345, Markus, Löffler
(a)dd, (f)ind, (c)hange, (d)elete, (p)rint, (q)uit? d

Name? Markus

(a)dd, (f)ind, (c)hange, (d)elete, (p)rint, (q)uit? p
5643, Katrin, Mühlhaus
(a)dd, (f)ind, (c)hange, (d)elete, (p)rint, (q)uit? f

Name? Markus

RECORD NOT FOUND!
(a)dd, (f)ind, (c)hange, (d)elete, (p)rint, (q)uit? q

thales$
```

Viel Erfolg!