Übungen zur Vorlesung
GRUNDLAGEN DER INFORMATIK I
Studiengang Elektrotechnik
WS 02/03

AG Betriebssysteme FB3
Kirsten Berkenkötter

Ausgabe: 7.1.2003
Besprechung: 9.1./16.1.2003
Abgabe: 16.1.2003

Übungszettel 5

Anmerkung: Die Punkte 1., 2. und 3. aus Aufgabe 2 gelten jeweils als eingenständige Aufgabe, es können also insgesamt vier Leute vorrechnen!

Aufgabe 1: Mehrdimensionale Arrays

Es soll ein Programm geschrieben werden, dass zwei Matrizen der Größe 5x5 addiert. Dies soll in einer Funktion mit folgendem Aussehen geschehen:

void matrixadd(int a[5][5], int b[5][5], int result[5][5])

Im Hauptprogramm sollen zwei Matrizen deklariert und mit Werten belegt werden. Diese werden dann zusammen mit einer Ergebnismatrix als Zeiger an die Funktion übergeben (Arrays können nicht von Funktionen zurückgegeben werden). Zu guter Letzt sollen alle drei Matrizen zur Ergebnisüberprüfung ausgegeben werden.

Aufgabe 2: Doppelt verkettete Listen

In der Vorlesung wurde die Datenstrukur Liste zunächst in der einfach verketteten Form vorgestellt (Programm *mylist.c*, erhältlich auf der Webseite unter Veranstaltungsinhalte -> Session 5 -> Hintergrundinformationen), d.h. jedes Element der Liste enthält einen Zeiger auf seinen Nachfolger. Dieses Programm soll nun zu einer doppelt verketteten Liste erweitert werden.

Doppelt verkettete Listen enthalten neben dem Zeiger auf das nachfolgende Element auch einen Zeiger auf das vorhergehende. Damit ist es möglich, die Liste sowohl vorwärts als auch rückwärts zu durchlaufen. Dies vereinfacht viele mögliche Operationen auf Listen wie Suchen von Elementen, Einfügen neuer Elemente an einer bestimmten Position und das Löschen bestimmter Elemente, da man die Liste in beide Richtungen durchlaufen kann, anstatt stets wieder von vorne zu suchen.

Damit ändert sich die Datenstruktur für ein Listenelement wie folgt:

```
struct list_element_t
{
userdata_t data;   //eigentliche Daten
struct list_element_t *next;   //Zeiger auf das nächste Element
struct list_element_t *previous;   //Zeiger auf das vorherige Element
};
```

Dementsprechend müssen auch einige der Funktionen für die Liste modifiziert, bzw. neu geschrieben werden.

1. Ändert die Funktion *appendList()*
2. Ändert die Funktion *insertList()*
3. Schreibt eine Funktion *traverseListBackward()*, die analog zu *traverseList()* die Liste rückwärts anstatt vorwärts traversiert

Die einfach verkettete Liste *mylist.c* aus der Vorlesung soll als Grundlage dienen, die ursprünglichen Funktionen müssen also lediglich für eine doppelt verkettete Liste abgeändert werden.