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

AG Betriebssysteme FB3
Kirsten Berkenkötter

Ausgabe: 21.1.2003
Besprechung: 23./30.1.2003
Abgabe: 30.1.2003

Übungszettel 6

Aufgabe 1: Bibliothek zum Addieren und Subtrahieren von Zahlen

Die Aufgabe ist es, eine Bibliothek zum Addieren und Subtrahieren von Zahlen zu schreiben (Datei *biblio.c*). Sie soll folgende Funktionen enthalten:

```
int addInt(int a, int b)
int subInt(int a, int b)
double addDouble(double a, double b)
double subDoubel(double a, double b)
```

Die ersten beiden Funktionen dienen zum Addieren und Subtrahieren von *int*-Zahlen, die beiden letzten zum Addieren und Subtrahieren von *double*-Zahlen.

Aufgabe 2: Headerdatei für die Bibliothek und Anwendung

Um die Bibliothek benutzen zu können, muss noch eine entsprechende Header-Datei (*biblio.h)* geschrieben werden. Diese deklariert die Funktionen und Variablen (in unserem Fall nur Funktionen), die von anderen Programmen verwendet werden können.

Ausserdem soll ein Programm *addieren.c* erstellt werden, dass die Bibliotheksfunktionen benutzt. Dazu muss die Bibliothek mit *#include "biblio.h"* eingebunden werden. Im Hauptprogramm sollen jeweils zwei *int*- bzw. *double*-Variablen angelegt und mit Werten initialisiert werden. Mit diesen Werten werden dann die vier Bibliotheksfunktionen aufgerufen und deren Ergebnis mit *printf* ausgegeben, also z.B. für *addInt:*

```
printf("%d + %d = %d\n", a, b, addInt(a,b));   //a, b Integer-Variablen
```

Aufgabe 3: Erweiterung: void-Pointer

Die Funktionen der Bibliothek *biblio.c* sollen so umgeschrieben werden, dass die Parameter als *void*-Pointer übergeben werden und auch der Rückgabewert ein *void*-Pointer ist, d.h. die Funktionen sollen typunabhängig werden. Um entscheiden zu können, welcher Datentyp vorliegt, wird eine zusätzliche char-Variable als Parameter verwendet:

```
void *add(char typ, void *a, void *b)
void *sub(char typ, void *a, void *b)
```

Wir für *typ* ein *'i'* übergeben, dann sollen *int*-Werte addiert werden, wird ein *'f'* übergeben, sind es *double*-Werte. Um die *void*-Pointer entsprechend verwenden zu können, müssen jeweils Type-Castings durchgeführt werden, also z.B.:

```
ergebnis = *(int *)a + *(int *b);   //ergebnis ist int-Variable
```

Achtung: Wenn Zeiger als Rückgabewerte zurückgegeben werden, muss es sich um globale Variablen handeln, da lokale Variablen nach Ablauf der Funktion wieder gelöscht werden.

Die geänderte Datei soll *biblio2.c* heissen, die entsprechende Header-Datei (die entsprechend der Änderungen ebenfalls geändert werden muss!) *biblio2.h.* Ändert auch das Programm *addieren.c* entsprechend der neuen Funktionen und benennt es *addieren2.c*.

Aufgabe 4: Erweiterung: 'Addieren' von Strings

Zusätzlich zum Addieren von Zahlen soll die Funktion *add* auch Zeichenketten 'addieren' können, d.h. sie soll zwei Zeichenketten aneinander hängen. Für den Typ soll in diesem Fall *'s'* übergeben werden. Ein Zeiger auf die zusammengesetzte Zeichenkette soll zurückgegeben werde.

Zum 'Addieren' sollen die Funktion *strlen*, *strcpy* und *strcat* der Headerdatei *string.h* verwendet werden. Zunächst soll mit *strlen* die Größe der beiden Teilstrings ermittelt werden und dann mit *malloc* ein entsprechend großer Raum im Speicher reserviert werden. In diesen Speicherbereich wird dann mit Hilfe von *strcpy* und *strcat* der zusammengesetzte String geschrieben.

Die geänderte Datei soll *biblio3.h* heissen, die entsprechende Header-Datei *biblio3.h* (die Header-Datei muss in diesem Fall nicht geändert werden und wird nur aus Konsistenzgründen umbenannt). Erweitert das Programm *addieren2.c* entsprechend und benennt es *addieren3.c.*

Aufgabe 5: Erweiterung: 'Subtrahieren' von Strings

Als letztes soll auch die Funktion *sub* so erweitert werden, dass Zeichenketten voneinander 'subtrahiert' werden können. 'Subtrahieren' soll dabei folgende Bedeutung haben: wenn die zweite Zeichenkette in der ersten enthalten ist, soll das Result die erste Zeichenkette ohne die zweite sein, also z.B.

1. Zeichenkette: "Hello"
2. Zeichenkette: "ll"
1. Zeichenkette - 2. Zeichenkette: "Heo"

Mit Hilfe der Funktion *char *strstr(char *kette1, char*kette2)* der Headerdatei *string.h* kann man feststellen, ob der String *kette2* im String *kette1* enthalten ist. Wenn ja, wird ein Zeiger auf den Teilstring innerhalt von *kette1* zurückgegeben, ansonsten NULL.

Damit ist es möglich, den 'Subtraktionsstring' zu bilden, indem man zunächst alle Zeichen bis zum Beginn des Teilstrings in das Ergebnis zu kopieren, dann die Länge von *kette2* bestimmt und entsprechend viele Zeichen in *kette1* ausläßt um zuletzt die restlichen Zeichen von *kette1* in das Ergebnis zu kopieren. Ist der Teilstring *kette2* nicht in *kette1* enthalten, soll ein Zeiger auf *kette1* zurückgegeben werden, ansonsten ein Zeiger auf das Ergebnis.

Die geänderte Datei soll *biblio4.c* heissen, die Header-Datei *biblio4.h* (die Header-Datei muss wiederum nicht geändert werden). Ausserdem soll *addieren3.c* entsprechend erweitert werden und *addieren4.c* genannt werden.