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

AG Betriebssysteme FB3
Kirsten Berkenkötter

Ausgabe: 19.11.2002
Besprechung: 21./28.11.2002
Abgabe: 28.11.2002

Übungszettel 3

Der Übungszettel enthält sechs Aufgaben, um mehr Leuten die Gelegenheit zum Vorrechnen zu geben. Wenn ihr nicht alles schafft, dann löst auf jeden Fall die Aufgaben 4, 5 und 6.

Aufgabe 1: Verzweigung mit *switch*

Schreibt ein Programm, dass zu einer Integer-Zahl entsprechenden Text auf dem Bildschirm ausgibt. Für 1 soll *eins* ausgegeben werden, für 2 *zwei* und für drei *drei.* Bei 4 und 5 soll der Text *viele* lauten, bei allen anderen Zahlen *weiss nicht*. Diese Ausgabe soll mit Hilfe eines *switch*-Konstrukts realisiert werden.

In einer *while*-Schleife soll die entsprechende Zahl vom Benutzer mit *scanf* erfragt werden. Gibt er eine *0* ein, soll die Schleife abgebrochen und das Programm beendet werden.

Aufgabe 2: Tranformation von *switch* in *if - else if - else*

Das Problem aus Aufgabe 1 soll nun mit einer *if - else if - else*-Verzweigung anstatt des *switch* gelöst werden.

Aufgabe 3: Formale Spezifikation von Aufgabe 3, Übungsblatt 2

Gebt die formale Spezifikation der Vor- und Nachbedingungen des Algorithmus von Aufgabe 3, Übungsblatt 2 an. Damit die Aufgabe einheitlich gelöst werden kann, soll die Lösung von Übungsblatt 2 aus dem Internet zu Grunde gelegt werden.

Aufgabe 4: Definition eigener Datentypen

Definiert mit Hilfe von *typedef*, *enum* und *struct* eigene Datentypen und initialisiert entsprechende Variablen:

1. Monate (Januar, Februar, März, ..., Dezember);
2. Datum (Tag, Monat, Jahr) unter zur Hilfenahme von 1.
3. Komplexe Zahlen

Aufgabe 5: Binäre Suche in einem int-Array

In der Vorlesung wurde der Algorithmus *Binäre Suche* vorgestellt. Schreibt ein Programm, dass die *binäre Suche* anwendet, um einen Wert in einem vorsortiertem Integer-Array zu finden.

Legt dazu ein Integer-Array der Größe 10000 an und belegt es fortlaufend mit den Werten 0 bis 9999, da der Algorithmus ja nur für vorsortierte Arrays funktioniert. Die binäre Suche soll in einer Funktion folgenden Aussehens realisiert werden:

*int binaereSuche(int suchzahl, int *array, int laenge, int *zaehler);*

Dabei soll *suchzahl* die zu suchende Zahl sein, **array* ein Zeiger auf das zu durchsuchende Array, *laenge* die Länge dieses Arrays und **zaehler* ein Zeiger auf eine Integer-Variable, mit der die Anzahl der benötigten Vergleiche in der binären Suche gezählt werden soll. Der Rückgabewert der Funktion soll der Index der Zahl im Array sein, falls die Zahl gefunden wurde und -1, falls die Zahl nicht gefunden wurde.

Motivation: Prüft man die Zahlen in einem Array der Länge n nacheinander, benötigt man durchschnittlich $n/2$ Vergleiche, um die Zahl zu finden. Bei der binären Suche benötigt man durchschnittlich $log_2 n + 1$ Schritte. In unserem Beispiel mit $n=10000$ würde das bedeuten, dass im ersten Fall durchschnittlich *5000* Vergleiche notwendig sind um im zweiten Fall ca. *14!*

Aufgabe 6: Formale Spezifikation von Aufgabe 5

Gebt die formale Spezifikation der Voraussetzungen, Vor- und Nachbedingungen der Funktion *binaereSuche* aus Aufgabe 5 - Binäre Suche - an.