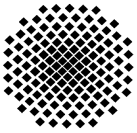
Universität Stuttgart
Fakultät 5, Institut FMI
EfidI 1, Professor Claus



WS 2002/2003
Bertele, Lewandowski
17. Januar 2003

Aufgabenblatt 12

Abgabe bis 24.1.2003, 20:00 Uhr – Besprechung in der Woche ab dem 27.1.2003

Aufgabe 1 Buchführung (schriftlich, mittel) 2+2+2+2 Punkte

Das ist jetzt eine Programmieraufgabe auf Klausurniveau. In der Klausur haben Sie dafür 60 Minuten Zeit. Für Teilaufgabe c) finden Sie im Anhang eine kleine Erklärung zur Speicherverwaltung in Ada95. Geben Sie Ihre Lösung als eine komplette Abgabe ab.

Bei der Doppelten Buchführung werden Geldbeträge auf Konten gebucht. Jede Verbuchung eines Buchungssatzes spricht zwei Konten an: Auf einem Konto wird der Betrag ins Soll gebucht, auf dem anderen ins Haben. Soll bedeutet eine positive Integerzahl, Haben bedeutet eine negative Integerzahl. Da der Betrag auf einem Konto vorzeichenverkehrt als auf dem anderen Konto verbucht wird, bleibt die Summe aller Kontostände stets gleich.

Die Konten bestehen aus einer Kontonummer (*Natural*) von 1 bis 99 und einem aktuellen Kontostand, der zur Vereinfachung nur ganze Euro-Beträge aufnehmen kann und daher mit *Integer* dargestellt wird.
Beim Start des Buchführungsprogrammes sollen alle 99 Konten mit dem Kontostand 0 initialisiert werden.

Jeder Buchungssatz besitzt drei Elemente: Einen (positiven oder negativen) Geldbetrag, eine Kontonummer, und eine Gegenkontonummer. Bei der Verbuchung wird der Betrag auf das Konto aufaddiert, während derselbe Betrag vom Gegenkonto subtrahiert wird. Das setzt natürlich voraus, dass Konto- und Gegenkontoangabe auch gültig (d.h. im Bereich von 1 bis 99) sind; sonst erfolgt eine Fehlermeldung. Falls ein Buchungssatz wegen eines Fehlers nicht verbucht werden kann, dürfen dadurch keinerlei Kontostände geändert werden.

Unser Buchhaltungsprogramm soll nach dem sogenannten Primanotaverfahren arbeiten: Eine Liste von Buchungssätzen wird ohne Prüfungen auf Korrektheit erfasst (d.h. eingegeben), ohne dass die Kontostände davon berührt werden.
Die Liste wird dann später auf einen Schlag verbucht, wobei nicht nur die Kontostände geändert, sondern gleichzeitig das Journal (= Liste der Kontoveränderungen) ausgegeben wird. Im Journal sind auch eventuelle Fehlermeldungen enthalten.

a) Die Konten sollen in einem Array gespeichert werden, während für die Buchungssätze eine (unsortierte) einfach verkettete Liste verwendet werden soll. Schreiben Sie die dazu notwendigen Ada95 Typdeklarationen. Unnötige oder umständliche Lösungen geben Punktabzug!
b) Schreiben Sie eine Prozedur zur Erfassung einer Buchungssatzliste, die keine globale Variablen verwendet (stattdessen Parameter verwenden!). Ein Buchungserfassungsdialog erfragt zuerst den Geldbetrag, dann die Kontonummer und zuletzt die Gegenkontonummer eines Buchungssatzes. Eingabe eines Geldbetrages 0 bedeutet das Ende der Liste (Abbruchkriterium für die Erfassungsprozedur).
c) Schreiben Sie eine Prozedur, die die Angaben in einer Buchungssatzliste verbucht und dabei jede Buchung in einer Zeile auf den Bildschirm ausgibt oder evtl. einen Fehler meldet (Journal). Der Speicherplatz für die Liste soll wieder freigegeben werden. Nach der Verbuchung wird eine Kontostandsliste (Kontonummer, Stand) auf den Bildschirm ausgegeben, wobei Konten mit Stand 0 weggelassen werden.
d) Schreiben Sie nun ein vollständiges Ada95-Package, in dem Sie durch Kommentare anzeigen, an welchen Stellen Sie die obigen Typ- und Prozedurdeklarationen einbauen wollen. Das Hauptprogramm soll die Prozeduren aus b) und c) ohne Benutzerdialog in fester Reihenfolge aufrufen.

Alle Prozeduren sollten nicht auf globale Variablen zugreifen. Gestalten Sie die Dialoge und Ausgaben aussagekräftig.

Aufgabe 2 Große Zahlen IX (Votieraufgabe, mittel) 4 Punkte

Auf dem vergangenen Blatt haben wir mit großen Zahlen gearbeitet und dabei ein bereits recht brauchbares Paket Gzahl programmiert. Diese Aufgabe stellt nun die Anwendung des Pakets in den Vordergrund.

Erstellen Sie ein Paket *Fzahl* (Flexible Zahlen), welches wieder mit beliebigen Zahlen arbeiten kann, dennoch auf *Gzahl* basiert. Überlegen Sie sich hierbei, wie man mit dem generischen *Gzahl* effizient arbeiten kann. Es soll immer nur minimal Speicherplatz belegt werden. Reicht der Speicherplatz nicht aus, dann muss eine neue Instanz von *Gzahl* herhalten.

Aufgabe 3 Große Zahlen X (schriftlich, mittel) 4 Punkte

Hier machen wir jetzt ernst mit der Anwendung. Die in dieser Aufgabe vorkommenden Zahlen können recht schnell sehr groß werden.

Lesen Sie eine Zeichenkette aus einer Datei ein und berechnen Sie die Anzahl der unterschiedlichen Untersequenzen. Ein Untersequenz ist eine Zeichenkette, die durch Streichen von einem oder mehreren Zeichen des Ausgangsstrings entsteht.

Aufgabe 4 Eine interessante Funktion (schriftlich, mittel) 4 Punkte

In dieser Aufgabe sehen Sie, dass selbst hinter einer unscheinbaren Funktion Unerwartetes stecken kann.

Die Funktion *A(m,n)* ist für $m, n \geq 0$ folgendermaßen definiert:

$$A(m,n) = \begin{cases} n+1 & \textit{falls } m = 0 \\ A(m-1,1) & \textit{falls } n = 0 \wedge m > 0 \\ A(m-1, A(m, n-1)) & \textit{sonst}. \end{cases}$$

Schreiben Sie ein Ada95 Programm zur Berechnung der Funktion *A*. Erweitern Sie Ihr Programm so, dass zusätzlich die Anzahl der rekursiven Aufrufe, sowie die Argumente mitausgegeben werden. Erstellen Sie eine Tabelle, in der Sie für möglichst viele Werte von *m* und *n* das Ergebnis der Funktion, sowie die Anzahl der dafür nötigen Aufrufe angeben.

Allgemeine Hinweise:

- Ada95 verfügt über eine leistungsfähige Speicherverwaltung. Nicht mehr benötigtet Speicher wird automatisch freigegeben. Dennoch gehört es zum guten Programmierstil für die Freigabe des Speichers selbst zu sorgen. In Ada steht dafür die generische Prozedur *Unchecked_Deallocation* zur Verfügung. Diese können Sie mit *with* einbinden und dann eine neue Instanz generieren. Als generischen Parameter besitzt die Prozedur den Typ und einen Zeiger auf selbigen Typ. Als nicht generischer Parameter fungiert dann ein Zeiger auf den freizugebenden Typ. Die neu generierte Routine gibt den Speicher frei und setzt den übergebenen Zeiger auf *null*.
- Die schriftlichen Aufgaben (16 Punkte) geben Sie bitte zum Abgabezeitpunkt im eClaus-System ab. Bitte votieren Sie bitte ebenfalls bis zum Abgabezeitpunkt im eClaus-System.
- Bei weiteren Fragen, wenden Sie sich bitte an das Forum (http://fachschaft.informatik.uni-stuttgart.de/forum/), Ihren Tutor, oder per Mail direkt an J. Bertele (inf@studbs.informatik.uni-stuttgart.de).

Weitere Hinweise finden Sie auf unserer Veranstaltungswebseite unter:
http://www.informatik.uni-stuttgart.de/ifi/fk/lehre/ws02-03/info_I_0203.html