Prof. Dr. Franz Schweiggert Abteilung Angewandte Informationsverarbeitung 22. Januar 2004
Dr. Andreas Borchert, Michael Wiedemann Blatt 12

Allgemeine Informatik I (WS 2003/2004)

Abgabetermin: 28. Januar 2004

25 Mein kleines Telefonverzeichnis – 15 Punkte

Wir wollen nun ein kleines Telefonregister schreiben. Um die Sache zu vereinfachen bzw. um dem Datenschutz zu genügen, sind folgende Informationen eines Eintrages bekannt:

- die Vorwahl
- der Vorname
- der Nachname
- der Ort

Diese sollen diese Mal der Einfachheit halber nacheinander von der Standardeingabe eingelesen werden (siehe Beispiele). Danach sollen die Einträge nach den einzelnen Kriterien absteigend sortiert werden, also entweder nach Vorwahl, Vorname, Nachname oder Ort. Anschliessend erfolgt die Ausgabe der jeweils sortierten Liste. Zum Sortieren verwenden wir dieses Mal den Bubblesort. Dieser bedient sich folgendem Algortihmus:

Man beginnt mit den ersten Eintrag. Dieser wird nacheinander mit allen folgenden Einträgen verglichen. Ist ein nachfolgender Eintrag im zu untersuchenden Attribut kleiner (numerisch oder lexikalisch), so wird dieser mit dem ersten Eintrag getauscht. Dies wird fortgesetzt, bis man alle folgenden Einträge mit dem ersten verglichen hat und somit der Eintrag mit dem kleinsten Attribut ganz vorne steht. Dann setzt man das ganze mit dem zweiten Eintrag und allen folgenden fort, usw.

Da das Sortieren quasi unabhängig von dem Sortierkriterium durchgeführt wird (der Algorithmus des Sortierens ist immer der gleiche, nur der Vergleich der Einträge ändert sich), verwenden wir das Konzept der Prozedurtypen. Wir definieren also einen Prozedurtyp, der zum Beispiel diese Gestalt haben könnte:

TYPE Vergleich = PROCEDURE(entry1, entry2: Eintrag) : INTEGER;

Ausserdem sollte folgendes beachtet werden:

Liest man die Vorwahl als Integer ein, so wird die eingegebene Vorwahl *0731* als *731* gespeichert. Dies ist insofern ungenügend, falls eine vierstellige Vorwahl eingegeben wird, die keine führende Null besitzt. Deswegen sollten die Vorwahlen nicht als Integer, sondern als *Array von Integer* gespeichert werden, wobei von einer vierstelligen Vorwahl auszugehen ist.
Der Vergleich von zwei Strings kann unter Umständen Probleme bereiten. Deswegen verwenden wir das Oberon-Modul *ConstStrings*. Dieses Modul beinhaltet die Funktion *Compare*, mit der man bequem zwei Strings vergleichen kann.

Tipps:

- Man kann ausnahmsweise davon ausgehen, dass die Daten korrekt eingegeben wurden. Eine Fehlerbehandlung ist somit nicht notwendig. Wir gehen davon aus, dass die Vorwahl genau 4 Stellen besitzt. Ausserdem beinhalten Vorname, Name und Ort keine Blanks, es sind also zusammenhängende Zeichenketten.
- Die Anzahl der einzugebenen Eintraegen sollte im Programm als Konstante mitgegeben werden. Ein guter Testwert ist 5.
- StringCompare liefert $< 0, 0$ oder > 0 zurück (im Falle der erste String ist lexikalisch kleiner, gleich oder grösser als der zweite String). Man sollte den Vergleich der Vorwahlen dementsprechend anpassen, so dass die Sortier-Prozedur immer passende Werte erhält.
- Es sollte wieder das Prozedurkonzept beachtet werden. Das bedeutet, ausser dem Telefonbuch an sich sollte so wenig wie möglich global deklariert werden, sondern je nach Bedarf lokal innerhalb der Prozeduren.
- In den Beispielen findet Ihr Programme zu Prozedurtypen und StringCompare, einen exemplarischen Programmaufruf und einen Programmrumpf. Damit sollte der Start um einiges einfacher sein.
- *Für die folgenden Blätter werden nicht nur die Programme an sich, sondern auch der Stil (das heisst, Kommentare, sprechende Variablen und Einrückung) bewertet. Achtet darauf.*

Viel Erfolg!