Prof. Dr. Franz Schweiggert Abteilung Angewandte Informationsverarbeitung 4. Februar 2004
Dr. Andreas Borchert, Michael Wiedemann Blatt 14

Allgemeine Informatik I (WS 2003/2004)

Abgabetermin: 1. Aufgabe 11. Februar 2004 / 2. Aufgabe freiwillig

27 Telefonbuch – aber richtig – 10 Punkte

Ihr erinnert Euch sicherlich nach an das kleine Telefonverzeichnis, dass wir auf dem 12. Übungsblatt schreiben sollten. Dieses Prográmmchen werden wir nun ein wenig erweitern. Zum einen sollen zusätzliche Datentypen behandelt (und somit weitere Vergleichsprozeduren erstellt) werden, zum anderen wollen wir die Programmbenutzung ein wenig komfortabler machen.

Außer den bekannten Einträgen Vorwahl, Vorname, Nachname und Ort sollen nun die folgenden Daten aufgenommen werden:

- Alter – abzuspeichern als INTEGER
- Geschlecht (m/w) – abzuspeichern als CHAR
- Internetzugang vorhanden (j/n) – einzulesen als CHAR, abzuspeichern als BOOLEAN

Ihr könnt entweder mit Eure alten Version oder mit der Musterlösung arbeiten. Voraussetzung ist allerdings, dass Ihr Prozedurtypen verwendet, denn nun müssen drei weitere Vergleichsprozeduren zu obigen neuen Daten geschrieben werden. Euch wird (hoffentlich) auffallen, dass die Änderungen aufgrund der vorgegebenen Struktur des Programms sehr schnell zu erledigen sind.

Sobald dies getan ist, solltet Ihr Euch Gedanken um die Argumentverarbeitung machen. Euer Programm sollte nämlich folgendermaßen funktionieren (wir nehmen an, die ausführbare Datei heisse Telefonbuch):

- Telefonbuch –[*vVnNoOwWaAgGiI*] :
  bekommt das Programm eine oder mehrere *Flags*, gross oder klein geschrieben, beim Aufruf, sollte entsprechend sortiert werden: nach **V**orname, **N**achname, **O**rt, **V**orwahl, **A**lter, **G**eschlecht, **I**nternet-Anschluss
  zum Beispiel:
  Telefonbuch –n (sortierte Ausgabe nach Nachnamen)
  Telefonbuch –aGi (sortierte Ausgabe nach Alter, Geschlecht und Internetanschluss (nacheinander))
- Wird das Programm ohne zusätzliche Angaben aufgerufen, soll die ursprüngliche Liste ausgegeben werden. In *allen* anderen Fällen soll eine Fehlermeldung ausgegeben werden.

Anmerkung: speziell diejenigen unter Euch, die eine Klausur schreiben, sollten sich an dieser Aufgabe versuchen, denn Argumentverarbeitung wird Klausur-relevant sein.

28 Das Beste zum Schluss – Ferienaufgabe

Um nun das Gelernte des gesamten Semester abzurunden, ein etwas umfangreicheres Programm zum Schluss.
Es handelt sich hierbei um das Spiel ***"Reversi"***, in leicht abgeänderter Form. Für diejenigen unter Euch, die dieses Spiel nicht kennen, hier kurz die wesentlichen Regeln:

- Zunächst braucht man ein Spielfeld, zum Beispiel 8x8 (gerade Anzahl ist Voraussetzung). In der Mitte des Spielfeldes werden jeweils zwei Steine der beiden Spieler positioniert.

```
 |1|2|3|4|5|6|7|8|
1| | | | | | | | |
2| | | | | | | | |
3| | | | | | | | |
4| | | |X|X| | | |
5| | | |O|O| | | |
6| | | | | | | | |
7| | | | | | | | |
8| | | | | | | | |
```

- Nun dürfen die Spieler nacheinander einen Stein legen. Wichtig ist nur die Tatsache, dass sich horizontal oder vertikal benachbart bereits ein Stein befindet. Hier würde das bedeuten, dass der Zug (3,3) unzulässig, aber der Zug (3,4) zulässig ist. Ist der Zug unzulässig, kommt der nächste Spieler zum "Zug".
- Schließt ein Spieler mit seinen Zug die des Gegners vollständig ein, so werden alle Steine der Reihe in Steine des Spielers umgewandelt.
  Ein Beispiel:
  Spieler 1 (X) ist an der Reihe und setzt seinen Stein auf (5,6).Er schließt die Steine des zweiten Spieler vollständig ein, somit ergibt sich beim Zug von Spieler 1 folgendes Bild:

```
Vor dem Zug:               Zwischenzustand:           Nach dem Zug:
 |1|2|3|4|5|6|7|8|          |1|2|3|4|5|6|7|8|          |1|2|3|4|5|6|7|8|
1| | | | | | | | |         1| | | | | | | | |         1| | | | | | | | |
2| | | | | | | | |         2| | | | | | | | |         2| | | | | | | | |
3| | | | | | | | |         3| | | | | | | | |         3| | | | | | | | |
4| | | |X|X|O| | |   -->   4| | | |X|X|O| | |   -->   4| | | |X|X|O| | |
5| | |X|O|O| | | |         5| | |X|O|O|X| | |         5| | |X|X|X|X| | |
6| | | | | | | | |         6| | | | | | | | |         6| | | | | | | | |
7| | | | | | | | |         7| | | | | | | | |         7| | | | | | | | |
8| | | | | | | | |         8| | | | | | | | |         8| | | | | | | | |
```

- Das Spiel wird so lange fortgesetzt, bis alle Felder belegt sind. Gewonnen hat derjenige Spieler, der mehr Steine auf dem Brett liegen hat.

Anmerkungen:
Keine Sorge, das ganze Spiel zu programmieren, ist relativ umfangreich. Zu Beginn sollte Euer Programm erst einmal folgendes erledigen:

- *Aufbau des Spielfeldes mit einer festen konstanten Dimension (gerade Zahl + 1 (für die Beschriftung des Spielfeldes)). Die gerade Zahl ist maßgebend, damit man die "Mitte" des Spielfeldes bestimmen kann.*
- *Ausgabe des ursprünglichen Spielfeldes.*
- *Einlesen des Spielzug, abwechselnd von Spieler 1 und Spieler 2.*
- *Überprüfung, ob der Zug korrekt ist, das heißt, ob rechts, links, oben oder unten neben des zu setzenden Steines bereits ein Stein ist.*
- *Entweder Verwerfen des Spielzug oder Setzen des Steines und erneute Ausgabe des neuen Spielfeld.*
- *Fortsetzen des Spiel, bis das ganze Spielfeld mit Steinen gefüllt ist.*
- ***Achtung, nur für Fortgeschrittene:** Falls Euch das noch zu wenig ist, könnt Ihr mal probieren, das Spiel vollständig zu programmieren, das heisst, falls ein Spieler die Steine seines Gegners horizontal oder vertikal vollständig umschließt, werden sie umgewandelt. Zudem muss man am Ende die jeweiligen Steine des Spielers zählen. Zur Illustration gibt es das fertige Spiel zum Ausprobieren in den Beispielen zu diesem Blatt. Aber seid gewarnt, das ist nicht trivial! Ihr könnt bei diesem Programm fast alle Aspekte der vorherigen Übungsblätter benutzen. Vor allem das Prozedurenkonzept und etwa offene Feldparameter finden Anwendung. Zur Information: in meiner Lösung verwende ich sechs Prozeduren, das Hauptprogramm beschränkt sich auf 11 Zeilen.*

Viel Erfolg für die Aufagen, für die anstehenden Klausuren und schon einmal eine schöne vorlesungsfreie Zeit!