# Übungen zum Programmierkurs I

Abgabe bis zum Freitag, 24.01.2003, 20:00 Uhr

Bitte die Hinweise am Ende des Aufgabenblattes beachten!

### Aufgabe 11.1 Schlösser knacken (leicht)

7 Punkte

Die Büros der Firma Recursi EDV-Systeme sind mit hochkomplizierten Schlössern gesichert. Jedes Schloß besteht aus einer Anzahl von Riegeln, deren Anzahl je nach Bedeutung des jeweiligen Büroinhabers variiert. Jeder Riegel kann sich in zwei Stellungen (oben oder unten) befinden und kann nach folgenden Regeln in seiner Stellung verändert werden:

- (a) Der erste Riegel von links kann jederzeit bewegt werden.
- (b) Der *i*-te Riegel von links  $(i \ge 2)$  kann dann bewegt werden, wenn der Riegel unmittelbar links von ihm unten steht und alle noch weiter links liegenden Riegel oben stehen.

Eine Tür läßt sich nur dann öffnen, wenn alle Riegel oben stehen.

Die Schlösser haben sich als sicher, aber auch recht umständlich erwiesen; die Angestellten haben es mitunter schwer, morgens in ihr Büro zu gelangen. Schreiben Sie ein Programm, welches ihnen dabei hilft. Dieses soll zunächst die Anzahl der vorhandenen Riegel und ihre derzeitigen Zustände abfragen. Sodann soll eine Folge der zu bewegenden Riegel (und die dadurch entstehenden Zwischenzustände) ausgegeben werden, die dazu führt, dass die Tür geöffnet werden kann.

Beispiel: Um bei drei Riegeln alle Riegel von unten nach oben zu bewegen, bedarf es folgender Schritte:

## Aufgabe 11.2 Stack (mittel)

4 Punkte

Schreiben Sie ein Paket namens lifo, welches einen generischen Datentyp namens stack implementiert. (Diese Aufgabe dient im Wesentlichen der Vorbereitung von Aufgabe 11.3.) Das Paket soll die folgenden Funktionen implementieren:

```
function isempty(s:stack) return boolean;
function empty return stack;
function top(s:stack) return element;
function push(s:stack; e:element) return stack;
function pop(s:stack) return stack;
```

Dabei soll der Datentyp element parametrisierbar sein. Die Funktionen sollen die in der Vorlesung eingeführte übliche Bedeutung haben. Werden die Operationen top oder pop auf einen leeren Stack angewandt, so soll eine Ausnahme mit dem Namen stack\_error entstehen. Implementieren Sie Ihren Stack mit Hilfe einer verketteten Liste, nicht etwa mit einem Array!

Auf der Webseite zum Programmierkurs finden Sie ein Programm namens stacktest.adb, welches die oben erwähnten Funktionen benutzt. Sie sollten Ihr Paket so gestalten, dass sich stacktest.adb ohne Änderungen übersetzen läßt und die Meldung ausgibt, dass es keine Probleme gab.

### Aufgabe 11.3 Große Zahlen XXV (mittel)

9 Punkte

(Nein, der Titel war nur ein Scherz. Aber ums Rechnen geht's trotzdem.)

Mathematische Terme werden gewöhnlich in der *Infix-Notation* aufgeschrieben, d.h. der Operator wird zwischen die Operanden geschrieben, wie z.B. in 2+4. Bei der *Postfix-Notation* wird der Operator hinter die Operanden geschrieben, das Beispiel von eben wird zu 24+. Ein Vorteil dieser Notation ist, dass sie ohne Klammern auskommt, die bei der Infix-Notation wegen der Punkt-vor-Strich-Regel bisweilen notwendig sind. So ist der Infix-Term 2\*(3+4) bedeutungsgleich mit dem Postfix-Term 234+\*.

Schreiben Sie ein Programm, das den Benutzer zur Eingabe eines Postfix-Terms auffordert, und deren Wert berechnet; in den Beispielstermen wäre die Ausgabe 6 bzw. 14. Die Terme können aus natürlichen Zahlen, + und \* bestehen. Zusätzlich können Leerzeichen vorkommen, und Operanden voneinander zu trennen. Benutzen Sie zur Lösung dieser Aufgabe Ihr Stack-Paket aus Aufgabe 11.2.

## Aufgabe 11.4 Taschenrechner II (Zusatzaufgabe, mittelschwer) 5 Punkte

Schreiben Sie ein Programm. welches statt Postfix-Termen Infix-Terme berechnet. Bei diesen kommen zusätzlich Klammern vor, mit denen Teilterme gruppiert werden. Die Punkt-vor-Strich-Regel soll beachtet werden.

#### Allgemeine Hinweise

- Die Webseite zum Programmierkurs ist wie folgt: http://www.informatik.uni-stuttgart.de/ifi/fk/lehre/ws02-03/ada95/ Dort finden Sie jeweils freitags das neue Übungsblatt vor, außerdem Musterlösungen zu vergangenen Aufgabenblättern.
- Der Besuch der Übungen ist grundsätzlich Pflicht. Sind Sie aus gutem Grunde verhindert, sagen Sie bitte zuvor Ihrem Tutor Bescheid. Sie riskieren andernfalls, dass Ihnen die Punkte für das jeweils besprochene Aufgabenblatt aberkannt werden!
- Pro Aufgabenblatt werden maximal 20 Punkte auf den Übungsschein angerechnet.
- Falls Sie Fragen irgendwelcher Art haben, wenden Sie sich bitte an Ihren Tutor oder an die Übungsleitung:

Stefan.Schwoon@informatik.uni-stuttgart.de, Raum 0.019, oder Tel. 7816-427