Funktionale Programmierung SS 2013 - Aufgabenblatt 1

Prof. Dr. U. Kastens
Institut für Informatik, Fakultät für Elektrotechnik, Informatik und Mathematik, Universität Paderborn
15.04.2013

Praktische Hinweise

In der heutigen Übung verwenden wir das Lisp-System *clisp*. Wenn Sie *clisp* ohne Parameter aufrufen, wird der interaktive Modus gestartet, den Sie durch die Tastenkombination `<STRG>-d` wieder verlassen können. Lisp arbeitet dann in der sog. *read-eval-print*-Schleife:

- lies die Eingabe
- werte sie aus
- gib den Ergebniswert aus

Um Ihre Lösungen aufzuheben, empfiehlt es sich, daneben einen Editor (z.B. kate oder gedit) geöffnet zu halten und korrekte Lisp-Programme durch Copy-and-Paste mit der Maus (Kopieren linke Taste, Einfügen mittlere Taste) in den Editor zu übertragen.

Eine so entstehende Lisp-Datei kann dem Lisp-System als Kommandozeilen-Parameter zur nicht-interaktiven Auswertung übergeben werden. In diesem Modus entfällt allerdings die dritte Phase der *read-eval-print*-Schleife, die automatische Ausgabe der Ergebniswerte. Stattdessen können Sie die Funktion "print" verwenden, um Ergebniswerte auszugeben.

Bsp.:

```
(print (* 1 2 3 4))
```

Aufgabe 1 (Erste Schritte mit Lisp)

Starten Sie den CLISP Interpreter in einem Terminal-Fenster und öffnen Sie einen Texteditor. Entwickeln Sie die Lösungen interaktiv und übertragen Sie sie dann mit Hilfe des Editors in eine Datei `a1.lisp`.

Eine Übersicht der Grundfunktionen in Lisp steht auf Folie 202.

a) Schreiben Sie einen Lisp Ausdruck, der die Zahlen 13 und -12 addiert.

b) Schreiben Sie einen Lisp Ausdruck, der 5*(1+2)-16 berechnet. Formen Sie den Ausdruck dabei nicht algebraisch um.

c) Binden Sie den Ausdruck 2*3*4 an die Variable `sinn`. Hinweis: Sie können den arithmetischen Operatoren auch mehr als zwei Parameter übergeben.

d) Erzeugen Sie eine Liste, die die Zahlen 1 und 2 enthält. Benutzen Sie dazu den Listenkonstruktor `cons` und die leere Liste `NIL`.

e) Binden Sie eine Liste, die die Zahlen von 1 bis 4 enthält an eine Variable `mylist`. Benutzen Sie dazu die `quote` Funktion oder die Kurznotation `'( )`.

f) Definieren Sie eine Funktion `avg(x,y)`, die den Durchschnitt von `a` und `b` berechnet. Rufen Sie die Funktion mit den Werten 2 und 4, sowie 10 und 15 auf.

g) Definieren Sie eine Funktion `sign(x)`, die -1 liefert, wenn `x` negativ ist, 1 wenn `x` positiv ist und 0 wenn `x` gleich 0 ist. Für die Fallunterscheidung benötigen Sie die Funktion `cond`.

Aufgabe 2 (Rekursive Funktionen in Lisp)

a) Schreiben Sie eine Funktion `sum(n)`, die die Summe der Zahlen von 0 bis n berechnet. Gehen Sie davon aus, dass n eine natürliche Zahl $\in \{0,1,2,...\}$ ist.

b) Schreiben Sie eine Funktion `pow(b,e)`, die b^e rekursiv berechnet. Gehen Sie davon aus, dass e eine natürliche Zahl $\in \{0,1,2,...\}$ ist.

Aufgabe 3 (Funktionen auf Listen)

a) Die eingebaute Funktion `nth` liefert das n-te Element einer Liste (Zählung beginnt bei 0). Hier sind Beispiel-Ausdrücke:

```
(nth 0 '(hallo welt !))
(nth 1 '(hallo welt !))
(nth 2 '(hallo welt !))
(nth 3 '(hallo welt !))
```

Schreiben Sie die Funktion `ntes`, die sich genauso verhält.

b) Die eingebaute Funktion `append` hängt zwei Listen aneinander. Hier ist ein Beispiel-Aufruf:

```
(append '(a b c) '(d e))
```

Schreiben Sie die Funktion `myappend`, die sich genauso verhält.

Aufgabe 4 (Funktionen höherer Ordnung)

a) Definieren Sie die Funktion `twice(f x)`, die die Funktion `f` "zweimal" auf das Argument `x` anwendet.

Bsp.:

```
(twice (lambda (x)(* x x)) 2)
```

ergibt 16.

Hinweis: Verwenden Sie `funcall`, um eine Funktion aufzurufen, die als Parameter übergeben wird, siehe Folie 204.

b) Neben der Funktion `cond` für allgemeine Fallunterscheidungen, gibt es in Lisp eine Funktion `if` für Unterscheidungen mit zwei Zweigen. Recherchieren Sie, wie `if` verwendet wird und schreiben Sie damit die `map`-Funktion von Folie 204. Nennen Sie sie `mymap`, da der Name `map` in *clisp* schon definiert ist.

Verwenden Sie `mymap`, um

1. alle Elemente einer Liste von Zahlen um 1 zu erhöhen.
2. alle negativen Elemente einer Liste durch 0 zu ersetzen.
3. jedes Element durch eine einelementige Liste zu ersetzen, die dieses Element enthält.

Diese 3 Aufgabenbeschreibungen für `mymap` sind recht salopp formuliert: gut verständlich aber im funktionalen Umfeld nicht korrekt. Versuchen Sie, die Aufgaben präzise zu formulieren.