Informatik III WS2004/2005

Übungsblatt 14

Institut Intelligente Systeme
Leiter: Prof. Dr. E. Lehmann
Dietmar Lippold

Besprechungstermin: 14.02.05 - 18.02.05

Aufgabe 1

Die Pattersprache aus Aufgabe 4 von Übungsblatt 13 soll verändert werden. Anstatt der Wildcards sollen nun Variablen verwendet werden können (die Wildcars sollen entfallen). An eine Variable kann ein Ausdruck (Atom oder Liste) gebunden werden. Wenn eine Variable an mehr als einer Stelle im Pattern vorkommt, muß sie an allen Stellen den gleiche Ausdruck binden.
Eine Variable soll als Symbol dargestellt werden, dessen erstes Zeichen ein Fragezeichen ist. Die gebundenen Variablen sollen in einer Assoc-Liste (s. R^{5}RS, S. 27) gespeichert werden.

1. Definieren Sie eine Prozedur `(variable? symbol)`, die prüft, ob das Symbol `symbol` eine Variable ist (lesen Sie gegebenenfalls nach über die Verarbeitung von Symbolen, Strings und Zeichen).

2. Erstellen Sie eine Prozedur `(erw-bind bindungen varname ausdr)`, die die Assoc-Liste `bindungen` um die Bindung der Variablen `varname` an den Ausdruck `ausdr` erweitert. Wenn die Variable schon an den Ausdruck gebunden war, soll die bisherige Bindungsliste geliefert werden. Wenn die Variable schon an einen anderen Ausdruck gebunden war, soll `#f` geliefert werden.

3. Erstellen Sie eine Prozedur `(matchv ausdr pat bind)`, die versuchen soll, das Pattern `pat` mit dem Ausdruck `ausdr` zu matchen und im Erfolgsfall die Liste der Bindungen liefern soll, wobei `bind` eine initiale Bindungsliste ist. Ist kein Match möglich, soll die Prozedur `#f` liefern.

   Beispiele:

```
> (matchv '(f b c) '(f b c) '())
()                    ;; Match, keine Variablenbindung erfolgt

> (matchv '(g b c (h a) b) '(g b ?x ?y b) '())
((?x c) (?y (h a)))  ;; Match, ?x <- c, ?y <- (h a)

> (matchv '(f b c) '(f ?x ?x) '())
#f                    ;; kein Match
```

Aufgabe 2 (schriftlich)

In dieser Aufgabe brauchen nur die 4 Grundrechenarten mit 2 Argumenten, Integerzahlen und Symbole als Variablen berücksichtigt werden.
Die Prozedur `(berechne ausdr bind)` soll den Wert des Ausdrucks `ausd` für gegebene Variablenwerte (Bindungen) `bind` berechnen. Die Bindungen sind wie in Aufgabe 1 als Assoc-Liste gespeichert (wobei die Variablen hier aber nicht mit einem Fragezeichen beginnen).

```
> (berechne '(+ (* x x)(* 3 y)) '((x 5)(y 7)))
46
```

1. Definieren Sie die Prozedur `(berechne ausdr bind)` ohne die Verwendung von Prozeduren höherer Ordnung.

   Sie soll den Ausdruck rekursiv abarbeiten.

2. Definieren Sie eine Fassung von `berechne`, die die Prozeduren zum Operator-Symbol aus einer Tabelle heraussucht und definieren Sie (unter Verwendung der Prozedur `list`) eine entsprechende Tabelle für die Grundrechenarten.

3. Testen Sie `eval`:

```
(set! x 6)
(eval 'x)                        (eval '(* 5 x))
(let ((x 7)) (eval 'x))          (eval '(let ((x 7)) x))
```

4. Die Ähnlichkeit der in Teil 1 und 2 verwendeten Ausdrücke mit Scheme-Code ist offensichtlich. Erstellen Sie eine Fassung von `berechne`, die `eval` benutzt, statt den Ausdruck selbst abzuarbeiten (rekursiv zu durchwandern).

   Hinweise: Erzeugen Sie Code, der dann mittels `eval` ausgewertet wird. Der Code darf **keine** globalen Variablen erzeugen, verändern oder benutzen und keine benannten Prozeduren definieren. Im erzeugten Code sollten die verwendeten Variablen also durch `let` lokal gebunden sein.

Aufgabe 3

Die logische Programmierung in Prolog unterscheidet sich wesentlich von der prozeduralen und funktionalen Programmierung.

1. Nennen Sie die wesentlichen charakteristischen Eigenschaften dieses Programmier-Paradigmas.

2. Wie ist ein Prolog-Programm aufgebaut? Was sind Hornklauseln? Was versteht man in Prolog unter Fakten, Regeln und Zielausdrücken?

3. Bitte erklären Sie genau die Bedeutung (Aufbau und Wirkungsweise) der folgenden zwei Zeilen Prolog-Code (einschließlich der darin enthaltenen Notationselemente):

```
member(X, [X|Y]).
member(X, [Y|Z]) :- member (X, Z).
```

4. Schreiben Sie eine Prozedur bzw. Funktion `myappend` in Scheme, Lisp, ML und Prolog, die (wie append) zwei Listen aneinander hängt. Geben Sie jeweils einen Beispiel–Aufruf an.

Weiteres

Übungsscheine gibt es bei Frau Castro, Raum 1.453. Die Fertigstellung der Scheine wird auf der WWW-Seite zur Veranstaltung [1] bekannt gegeben.

Prüfungstermin: Mo. 14.03.2005 (vorläufig, ohne Gewähr, s.a. WWW-Seite zur Veranstaltung, bitte nochmal selbst überprüfen).

Für die Prüfung viel Erfolg !

[1] `http://www.informatik.uni-stuttgart.de/ifi/is/Lehre/Vorlesung/info3.html`