Übung KogInf – Problemlösen, Suche, Lisp

Michael Siebers

November 9, 2010

Problemraum

Zwei Spieler (A und B) spielen das Spiel *Nimm* gegeneinander. Bei diesem Spiel liegen zu Beginn 9 Streichhölzer auf dem Tisch. Die Spieler müssen abwechselnd Streichhölzer wegnehmen. Sie können dabei frei wählen, ob sie eins, zwei oder drei Hölzer wegnehmen. Der Spieler, der das letzte Streichholz wegnimmt, verliert. Spieler A beginnt.

- Wie könnte man die einzelnen Spielzustände aufschreiben?
  Antwort: *Anzahl_der_Streichhölzer Spieler_der_am_Zug_ist*, z.B. 6B
- Wie lautet der Anfangszustand dieses Problems?
  Antwort: 9A
- Was ist das Problemlöseziel / welche Zielzustände gibt es?
  Antwort: 1B bzw. 0A
- Welche Problemlöseoperatoren gibt es?
  Antwort: nimm1, nimm2, nimm3
- Wie sieht der Problemlösegraph für dieses Problem aus?
  Antwort siehe take9.xml

Suche allgemein

Wegsuche zwischen Städten
Siehe distances.xml

Rekursion

Ergänzen Sie die Folgen. Wie lautet die rekursive Definition dieser Folgen?

4 7 10 13 16 19 22 25 28 ?=31
$F(0) = 4 \quad F(n) = F(n-1) + 3$

2 4 8 16 32 64 128 ?=256
$F(0) = 2 \quad F(n) = F(n-1) * 2$

2 3 5 9 17 33 65 ?=129
$F(0) = 2 \quad F(n) = F(n-1) * 2 - 1$

7 3 6 4 5 5 4 6 3 7 ?=2
$F(0) = 7 \quad F(1) = 3 \quad F(n) = F(n-2) + (-1)^{n-1}$
(In der Übung falsch angegeben!)

Fibonacci

0 1 1 2 3 5 8 13 21 34 55 89 ?=144
$F(0) = 0 \quad F(1) = 1 \quad F(n) = F(n-1) + F(n-2)$

LISP Einführung

1. Alle Ausdrücke in LISP sind Listen. LISP versucht Listen auszuwerten, indem das erste Element als Funktionsname und die verbleibenden Elemente als Parameter aufgefasst werden.
2. Die Auswertung einer Liste kann durch Voranstellen eines Apostrophs (’) verhindert werden. Diese Listen werden als Listen (=Daten) beibehalten.
3. Dezimalzahlen werden in amerikanischer Schreibweise, d.h. mit einem Punkt (.) als Dezimaltrenner geschrieben, z.B. 3.14, 5.2.

Mathematische Operatoren in LISP

Grundsätzlich kennt LISP alle Arithmetischen Operatoren:

+ Addiert zwei bis beliebig viele Zahlen.
$(+\ 2\ 3) \equiv 5$
$(+\ 1\ 2\ 3\ 4\ 5) \equiv 15$

- Mit einem Parameter wird die Zahl negiert. Bei mehreren Parametern werden alle von dem ersten abgezogen.
$(-\ 3) \equiv$ -3
$(-\ 3\ 2) \equiv 1$

***** Multipliziert alle Parameter auf.
$(*\ 1\ 2\ 3\ 4\ 5) \equiv 120$

/ Dividiert eine Zahl durch eine andere.
$(/\ 2\ 3) \equiv 0.66666666$

truncate Führt eine Ganzzahl-Teilung durch.
(**truncate** 9 2) $\equiv 4$

rem Berechnet den Rest einer Ganzzahl-Teilung.
(**rem** 9 2) $\equiv 1$

Operatoren auf Listen

car Extrahiert das erste Element einer Liste.
(car '(A B C)) ≡ A

cdr Eliminiert das erste Element aus einer Liste.
(cdr '(A B C)) ≡ (B C)

append Fügt mehrere Listen zusammen, so dass eine neue Liste entsteht, die als Elemente die Elemente der Einzellisten habt.
(**append** '(A B C) '(D E)) ≡ (A B C D E)

list Erstellt eine Liste, deren Elemente die Parameter in der gegebenen Reihenfolge sind.
(**list** '(A B C) '(D E)) ≡ ((A B C) (D E))

reverse Dreht eine Liste um.
(**reverse** '(A B C)) ≡ (C B A)

cons Fügt ein neues Element an den Anfang einer Liste ein.
(**cons** 'A '(B C D)) ≡ (A B C D)

length Gibt die Anzahl der Elemente in einer Liste wieder.
(**length** '(A B (C D E)) ≡ 3

Bedingte Ausführung

In LISP wird das Symbol T für den Wahrheitswert *wahr* und NIL für *falsch* verwendet.

= Vergleicht zwei Werte und gibt T zurück, falls sie gleich sind, andernfalls NIL.
(= 2 3) ≡ NIL
(= 2 2) ≡ T

cond Führt Anweisungen bedingt aus. Jeder Parameter muss eine Liste sein. Die Parameter werden nacheinander abgearbeitet. Falls das erste Element eines Parameters nicht zu NIL ausgewertet wird, werden die anderen Elemente dieses Parameters ausgewertet. Sobald ein Parameter vollständig ausgewertet worden ist, werden folgende Parameter nicht mehr beachtet. Sollte kein Parameter ausgewertet werden (alle Bedingungen werden zu NIL), gibt cond NIL zurück.
(**cond** ((= 0 0) 1) (t 0)) ≡ 1
(**cond** ((= 0 1) 1) (t 0)) ≡ 0

Funktionsdefinitionen

Mit Hilfe des Befehls **cond** können neue Funktionen definiert werden. Der Befehl erwartet mindestens drei Parameter: als erstes den Namen der zu definierenden Funktion, dann eine Liste mit Parameternamen für die Funktion. Weitere Parameter sind die Funktionen (inkl. Parameter) die Aufgerufen werden sollen, um die Aufgabe der Methode zu erfüllen. Die zuletzt aufgerufene Methode gibt den Rückgabewert der neu-definierten Funktion an.

```
(defun minus (a b) (- a b))
```

(minus 2 6) $\equiv$ -4

```
(defun seq (n)
  (cond
    ((= n 0) 2)
      (t (* 2 (seq (- n 1))))
))
```

(seq 2) $\equiv$ 8

Weitere Informationen zu LISP

Weitere Informationen zu LISP findet ihr unter `http://psg.com/~dlamkins/sl/contents.html` oder in Winston, Patrick H., Horn, Berthold. K. (1986). *LISP*. Addison-Wesley.