**Michael Arndt**
**Fritz Hamm**
**Peter Schroeder-Heister**

**Sommersemester 2003**
**Übungsblatt 6**
**11.6.2003**

# Übungen zur Vorlesung *Logikprogrammierung*

**Aufgabe 1** (2 Punkte)

Definieren sie ein Prädikat `permutation(L1,L2)`, das durch Backtracking alle Permutationen der Liste `L1` in `L2` berechnet:

```
?- permutation([1,2,3],L).
L = [1,2,3] ;
L = [1,3,2] ;
L = [2,1,3] ;
L = [2,3,1] ;
L = [3,1,2] ;
L = [3,2,1] ;
No
```

**Aufgabe 2** (2 Punkte)

Definieren Sie Prädikate `sublist/2`, `subset/2`, `prefix/2` und `suffix/2`, die folgendes leisten:

(1) `sublist(L1,L2)` testet, ob `L1` als zusammenhängende Teilliste in `L2` vorkommt.

(2) `subset(L1,L2)` testet, ob alle Elemente der Liste `L1` in `L2` enthalten sind.

(3) `prefix(L1,L2)` testet, ob `L1` das Präfix von `L2` ist.

(4) `suffix(L1,L2)` testet, ob `L1` das Suffix von `L2` ist.

**Aufgabe 3** (2 Punkte)

Definieren Sie Prädikate `palindrom1/1` und `palindrom2/1`, die auf unterschiedliche Weise ermitteln, ob eine Liste ein Palindrom ist.

**Aufgabe 4** (1 Punkt)

Wählen Sie Konstanten- und Funktionssymbole, um in Prolog Binärbäume als Terme zu repräsentieren. Definieren Sie ein Prädikat `binTree/1`, das von einem gegebenen Term entscheidet, ob er einen Binärbaum darstellt oder nicht.