Name: _____ Vorname: _____

Matrikelnummer:_____ Testat: _____

Paradigmen der Programmierung
Praktikum 1

Aufgabe 1) Machen Sie sich zunächst mit dem Prolog-System vertraut indem Sie ein paar einfache Dinge ausprobieren (Familienbeziehungen, einfache Listenprädikate, Fakultät, usw.).

Bei allen folgenden Aufgaben ist verlangt, dass Sie für jedes Prädikat einen deklarativen Kommentar schreiben. Vergleichen Sie dazu die vorgefundenen Kommentare.

Aufgabe 2) Ein Binärbaum sei wie folgt definiert:

```
tree(-).                    % leerer Baum
tree(n(X,L,R)):-            % Baumknoten mit Inhalt X
    tree(L), tree(R)-

wir haben weiter:

% enter(Tree0, X, Tree1)
%   Tree1 entsteht aus Tree0 durch sortiertes Hinzufügen
%   von V.
enter(-, X, n(X, -, -):- !.         % leerer Baum
enter(n(X,L,R), X, n(X,L,R):- !.  % X ist bereits vorhanden
enter(n(X0,L0,R), X, n(X0,L1,R)):-
    X @< X0, !,                     % X ist kleiner als X0
    enter(L0, X, L1).
enter(n(X0,L,R0), X, n(X0,L,R1)):-
    X @> X0, !,                     % X ist größer als X0
    enter(R0, X, R1).
```

2a) Schreiben Sie ein Prädikat, `l2t(Liste, Baum)`, das aus einer Liste rekursiv einen Baum erzeugt. Schreiben Sie das Prädikat zunächst normalrekursiv, dann aber auch endrekursiv (natürlich sollten Sie unterschiedliche Namen verwenden).

2b) Schreiben Sie ein Prädikat `t2l(Baum, Liste)`, das einen Baum in In-Order-Reihenfolge in eine (sortierte) Liste verwandelt.

Hinweis: Verwenden Sie `append(A, B, C)` zum Aneinanderhängen zweier Listen!

2c) Fassen Sie `l2t` und `t2l` zu einem Sortieralgorithmus zusammen. Was ist dabei nicht ganz korrekt? Wie kann man das ändern?

Aufgabe 3) Tippen Sie das folgende Prädikat ein:

```
% foldL(Liste, Praedikat, Init, Resultat)
%   Liste: eine Liste von Objekten
%   Praedikat(X, Y, Z):  weist Z einen aus X und Y
%     ermittelten Funktionswert zu
%   Init: der Anfangswert der Berechnung
%   Resultat: das Endergebnis.
%
%   Beginnend mit Init, wird Praedikat(Init, X1, Z1) berechnet,
%     dann Praedikat(Z1, X2, Z2) uws.
foldL([], _P, Init, Init).
foldL([X|Xs], P, Init, R):-
    call(P, Init, X, Y),      % ruft P(Init, X, Y) auf
    foldL(Xs, P, Y, R).
```

Als Beispiel können Sie wie folgt die Summe der Zahlen einer Liste berechnen:

```
?- foldL([1,2,3,4,5], plus, 0, S).
S = 15
```

(plus ist in SWI-Prolog schon vordefiniert)

Definieren Sie nun `l2t` aus Aufgabe 2a) mittels `foldL` neu. Welches ist die Funktion, die bei jedem Listenelement aufzurufen ist?

Weitere Hinweise erhalten Sie (bei Bedarf) im Praktikum !