

Rekursion in Prolog am Beispiel des Systemprädikats `append`

Datenbasis:

```
myAppend([],F,F).  
myAppend([H|T],L,[H|R]):-myAppend(T,L,R).
```

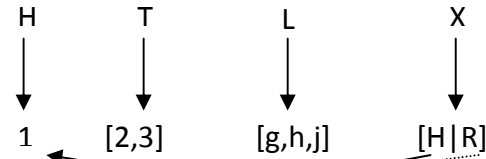
Aufruf mit `trace.` in Prolog:

```
[trace] 2 ?- myAppend([1,2,3],[g,h,j],X).  
  Call: (6) myAppend([1, 2, 3], [g, h, j], _G594) ? creep  
  Call: (7) myAppend([2, 3], [g, h, j], _G679) ? creep  
  Call: (8) myAppend([3], [g, h, j], _G682) ? creep  
  Call: (9) myAppend([], [g, h, j], _G685) ? creep  
  Exit: (9) myAppend([], [g, h, j], [g, h, j]) ? creep  
  Exit: (8) myAppend([3], [g, h, j], [3, g, h, j]) ? creep  
  Exit: (7) myAppend([2, 3], [g, h, j], [2, 3, g, h, j]) ? creep  
  Exit: (6) myAppend([1, 2, 3], [g, h, j], [1, 2, 3, g, h, j]) ? creep
```

```
X = [1, 2, 3, g, h, j].
```

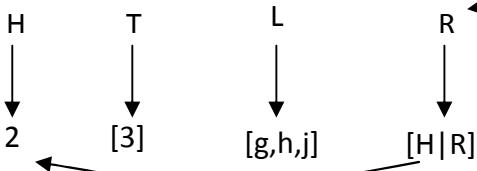
myAppend([],F,F).
 myAppend([H|T],L,[H|R]) :- myAppend(T,L,R).

myAppend([1,2,3],[g,h,j],X).



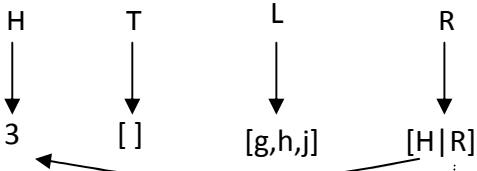
myAppend([H|T], L ,[H|R]) :- myAppend(T, L, R).
 myAppend([1|2,3], [g, h, j], [1|R]) :- myAppend([2,3], [g, h, j], R).

$_G594$
 $_G594 = [1 | _G679] = [1 | [2 | _G682]] = [1 | [2 | [3 | _G685]]] = [1 | [2 | [3 | F]]] =$
 $= [1 | [2 | [3 | g,h,j]]]$



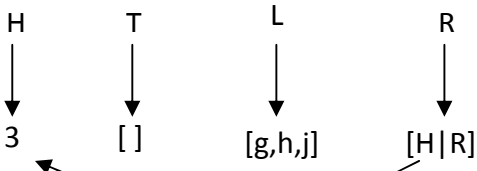
myAppend([H|T], L ,[H|R]) :- myAppend(T, L, R).
 myAppend([2|3], [g, h, j],[2|R]) :- myAppend([3], [g, h, j], R).

$_G679$
 $_G679 = [2 | _G682] = [2 | [3 | F]] = [2 | [3 | g,h,j]]$



myAppend([H|T], L ,[H|R]) :- myAppend(T, L, R).
 myAppend([3|], [g, h, j] ,[3|R]) :- myAppend([], [g, h, j], R).

$_G682$
 $_G682 = [3 | F] = [3 | g,h,j]$



myAppend([],F,F).
 myAppend([], [g, h, j], [g, h, j]).

$_G685$
 $_G685 = F = [g,h,j]$