Lists in PROLOG

In this laboratory we will experiment with various predicates dealing with lists.

Type in the `member1/2`, `concat1/3`, and `delete1/3` predicates as given in the lectures, and experiment with the various ways in which they can be used (i.e. by starting with different choices of free variables).

Using the arithmetic operations defined in the last laboratory, write a predicate

\[ \text{len}(X,N) \] to be true when \( N \) is the length of the list \( X \).

(NOTE: do not call your predicate \textit{length} - this is the name of a built-in predicate and you will get an error message)

What does prolog return to the following query and why?

\[ \text{?-len([a, [a,b], c], X).} \]

Write a predicate called `rlen(X,N)` to be true when \( N \) counts the total number of occurrences of atoms in the list \( X \).

Experiment with the two versions of the `reverse` predicate defined in lectures (see also lecture notes from 1993). What does prolog return to the following queries:

\[ \text{?-reverse1([a,b, [c,a]], X).} \]
\[ \text{?-reverse2([a,b, [c,a]], [], X).} \]

Check what happens with `reverse2`, if accumulator is a non-empty list e.g. [a].

Write the following predicates:

\[ \text{sum}(X,N) \] to be true when \( N \) is the sum of a list \( X \) of integers.

\[ \text{avg}(X,N) \] to be true when \( N \) is the (integer) average of a list of integers.

\[ \text{count}(X,Y,N) \] to be true if the list \( Y \) contains \( N \) occurrence of the element \( X \).

\[ \text{double}(X,Y) \] to be true when \( Y \) is the list consisting of each element of \( X \) repeated twice (e.g. `double([a,b],[a,a,b,b])` is true).

\[ \text{repeat}(X,Y,N) \] to be true when \( Y \) is the list consisting of each element of \( X \) repeated \( N \) times (e.g. `repeat([a,b], [a,a,a,b,b,b],3)` is true).

Finally, try to write a predicate that will \textit{sort1} a list of integers. Do not worry about efficiency at this stage. First write a simple predicate \textit{sort1/1} which answers whether a given list is sorted or not. Then write a predicate \textit{naivesort/2} which finds also a sorted list (hint: see also lecture notes from 1993).

Assoc. Prof. R. Kozera