CSCI 305 Concepts of Programming Languages

Participation Test 6

Instructions: Write your name above. Relax and attempt the problems above. This is NOT a quiz and participation credit will be given for any sincere attempt. (Later, solutions will be posted on the course webpage.) Turn in the sheet at the end of the class to receive your participation credit.

Example: Define a prolog function \texttt{get\_last/2} that finds the last element of a list.

\begin{verbatim}
%get\_last(X,L) :- X is the last element of the list L
get\_last(X,[X]).
get\_last(X,[..L]) :- get\_last(X,L).
\end{verbatim}

Exercise A: Define a prolog function \texttt{last\_but\_one/2} that finds the 2nd-to-last element of a list.

\begin{verbatim}
%last\_but\_one(X,L) :- X is the last but one element of the list L
last\_but\_one(X,[X,-]).
last\_but\_one(X,[,-,Y|Ys]) :- last\_but\_one(X,[Y|Ys]).
\end{verbatim}

Exercise B: Define a prolog function \texttt{get\_length/2} that finds the number of elements of a list.

\begin{verbatim}
%get\_length(L,N) :- the length of list L is N
get\_length([],0).
get\_length([-|L],N) :- get\_length(L,N1), N is N1+1.
\end{verbatim}
Exercise C: Define a prolog function reverse_list/2 that finds the reverse of a list.

%reverse_list(L1,L2) :- L2 is the reverse of L1
reverse_list(L1, L2) :- my_rev(L1, L2, []).
my_rev([], L2, L2).  
my_rev([X|Xs], L2, Acc) :- my_rev(Xs, L2, [X|Acc]).

Exercise D: Define a prolog function is_palindrome/1 that determines if a list is a palindrome (i.e., it is equal to its reversal).

%is_palindrome(L) :- L is a palindrome
is_palindrome(L) :- reverse_list(L, L).

Exercise E: Define a prolog function compress/2 that eliminates consecutive duplicates of list elements.

% compress(L1,L2) :- L2 is obtained from list L1 by compressing repeated % occurrence of elements into a single copy of the element
compress([], []).
compress([X|Xs], Xs).  
compress([X,X|Xs], Zs) :- compress([X|Xs], Zs).
compress([X,Y|Ys], [X,Zs]):- X \= Y, compress([Y|Ys], Zs).

Exercise F: Define a prolog function dupli/3 that duplicates the elements of a list a given number of times.

% dupli(L1,N,L2) :- L2 is obtained from list L1 by duplicating all elements N times
dupli(L1, N, L2) :- dupli-helper(CL1,N,L2,N).
dupli-helper([], _, []).
dupli-helper([X|Xs], N, Ys, 0) :- dupli-helper(Xs, N, Ys, N).
dupli-helper([X|Xs], N, [X|Ys], K) :- K > 0, K1 is K-1,
   dupli-helper([X|Xs], N, Ys, K1).