```
%%%  Two reverse procedures  %%%    Version 1.1

% Naive reverse (inefficient)

%  reverse0( L, LR )  - LR is the list L reversed

reverse0( [], [] ).
reverse0( [X|Xs], Zs ) :- reverse( Xs, Ys ), append( Ys, [X], Zs ).

% ............................................................

%  reverse( L, LR )  - LR is the list L reversed

reverse(Xs,Ys):- reverse(Xs,[],Ys).

%  reverse( [t1,...,tn], t, [tn,...,t1|t] )

reverse([],Ys,Ys).
reverse([X|Xs],Acc,Ys) :- reverse(Xs,[X|Acc],Ys).

% Justification of the last rule:
% If the body arguments are  ( [t1,...,tn], [x|t], [tn,...,t1,x|t] )
% The the head arguments are  ( [x,t1,...,tn], t, [tn,...,t1,x|t] )

% ,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
% Other descriptions of reverse/3

%  reverse( Xs, Acc, Ys) -
%    Xs = [t1,...,tn],  Ys= [tn,...,t1|Acc]

%  reverse( Xs, Acc, Ys ) -
%   Ys is the concatenation of the reversed list Xs and Acc

%  reverse( Xs, Acc, Ys ) -
%   Xs is a list;
%   if Acc is a list then Ys is a list;
%   Ys is the concatenation of the reversed Xs and Acc
```