```
% correct_colouring( Neighbours, Colouring )
%  Neighbours - a list of pairs, like   sweden:norway.
%  For each pair  Country1:Country2  from Neighbours,  Colouring has
%  members Country1-Colour1 and Country2-Colour2 with different colours.

%  If the lenght of Colouring equals the number of countries in Neighbours
%  then Colouring gives the correct colouring of the map, represented by
%  Neighbours.

correct_colouring( [], _Colouring ).
correct_colouring( [Country1:Country2|Neighbours], Colouring ) :-
        member( Country1-Colour1, Colouring ),
        member( Country2-Colour2, Colouring ),
        distinct_colours( Colour1, Colour2 ),
        correct_colouring( Neighbours, Colouring ).

distinct_colours( C1, C2 ) :- d_c( C1, C2 ).
distinct_colours( C1, C2 ) :- d_c( C2, C1 ).

d_c( red, green ).
d_c( red, blue ).
d_c( blue, green ).

/*
Example query:

    Neighbours = [belize     : guatemala,
                  guatemala  : el_salvador,
                  guatemala  : honduras,
                  el_salvador: honduras,
                  honduras   : nicaragua,
                  nicaragua  : costa_rica,
                  costa_rica : panama
                 ],
    Cols = [belize-_,   costa_rica-_, el_salvador-_, guatemala-_,
           honduras-_, nicaragua-_,  panama-_ ],
    correct_colouring( Neighbours, Cols ),
    member( guatemala-blue, Cols ),
    member( nicaragua-red, Cols ).
*/

% To avoid constructing the list Cols by hand (as in the query above).

% countries( Neighbours, Countries ) -
%   Neighbours is a list of pairs (of the form described previously).
%   Countries is a list of the elements of pairs of Neighbours
%   (one element for each occurrence in Neighbours).

countries( [], [] ).
countries( [C1:C2|Ns], [C1,C2|Countries] ) :- countries( Ns, Countries ).

% any_colouring( [t1,...,tn], [t1-_,...,tn-_] )

any_colouring( [], [] ).
any_colouring( [C|Countries], [C-_|Colouring] ) :-
        any_colouring( Countries, Colouring ).

%
```

```
% map_colouring( Neighbours, Colouring )
%  Neighbours - list of pairs, like   sweden:norway.
%  Colouring - list of pairs, like  sweden:green,
%  assignment of colours to countries from Neighbours, so that
%  neighbouring countries get different colours.

map_colouring( Neighbours, Colouring ) :-
        countries( Neighbours, Cs ),
                        %  Built-in sort/2 to remove repetitions.
        sort( Cs, Countries ),
        any_colouring( Countries, Colouring ),
        correct_colouring( Neighbours, Colouring ).
/*
Example query:

    Neighbours = [belize     : guatemala,
                  guatemala : el_salvador,
                  guatemala  : honduras,
                  el_salvador: honduras,
                  honduras   : nicaragua,
                  nicaragua  : costa_rica,
                  costa_rica : panama
                 ], map_colouring( Neighbours, Colouring ).
*/
```