```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CS_125 Logic Programming
% Spring 2002
% Solutions to Exercises 1
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Question 1.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Two possible solutions (last/2 is predefined
% in most Prolog versions):

last1([X],X).
last1([_|L],X) :- last1(L,X).

last2(L,X) :- append(_,[X],L).


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Question 2.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

different(X,Y,L) :-
  select(X,L,L1),
  member(Y,L1).

% select/3 as in the course notes.


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Question 3.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

familymembers([anne,beatrice,ben,edward,
               elisabeth,george,harry,homer,
               jill,mark,paul,patrick,peter,
               phillip,robert]).

sibling(C1,C2) :-
  familymembers(L), different(C1,C2,L),
  parent(P,C1), parent(P,C2).

% Remarks:

% 1. Using the meta predicate setof/3,
%    the predicate familymembers/1 could
%    also be defined by

familymembers1(L) :-
  family(F),
  setof(X,
        P^G^CC^(member(person(P,G,CC),F),
                (X=P;member(X,CC))),
        L).

% 2. Using negation as failure, sibling/2
%    can also be defined by:

sibling1(C1,C2) :-
  parent(P,C1),
  parent(P,C2),
  not(C1=C2).
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Question 4.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

list_of_lists([]).
list_of_lists([L|LL]) :-
  list(L),
  list_of_lists(LL).

list([]).
list([_|L]) :- list(L).


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Question 5.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

appendall([],[]).
appendall([L|LL],K) :-
  appendall(LL,K1),
  append(L,K1,K).


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Question 6.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

equal_sets(L1,L2) :-
  subset(L1,L2),
  subset(L2,L1).

subset([],_).
subset([H|T],L) :- member(H,L), subset(T,L).


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Question 7.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

equal_edges(edge(V,W),edge(V,W)).
equal_edges(edge(V,W),edge(W,V)).


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Question 8.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

graph([edge(a,b),
       edge(a,d),
       edge(a,e),
       edge(b,c),
       edge(b,d),
       edge(b,e),
       edge(c,d),
       edge(d,e)]).

draw([],[_]).
draw(G,[V1,V2|D]) :-
equal_edges(edge(V1,V2),E),
select(E,G,G1),
draw(G1,[V2|D]).

drawbe(V,W,[V|D]) :-
graph(G),
draw(G,[V|D]),
my_last(D,W).
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Question 9.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

dict([entry(one,un),
      entry(two,deux),
      entry(three,trois),
      entry(house,maison)]).

translate_word(E,F) :-
  dict(D),
  member(entry(E,F),D).



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Question 10.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

translate_sentence([],[]).

translate_sentence([E|ES],[F|FS]) :-
  translate_word(E,F),
  translate_sentence(ES,FS).

% ?- translate_sentence([one, house],FS).

% FS = [un, maison];
% No



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Question 11.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

lh([],0).
lh([_|T],s(N)) :- lh(T,N).



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Question 12.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

head_tails([],[],[]).

head_tails([[H|T]|LL],[H|Heads],[T|Tails]) :-
  head_tails(LL,Heads,Tails).

% ?- head_tails([[a],[a,b],[b,c,d]], Heads, Tails).

% Heads = [a, a, b]
% Tails = [[], [b], [c, d]] ;
% No
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Question 13.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Using predicate lh/2:

rectangle([]).
rectangle([L]) :- list(L).

rectangle([L1,L2|LL]) :-
  lh(L1,N),
  lh(L2,N),
  rectangle([L2|LL]).

% ?- rectangle([[a,b,c],[d,e,f]]).
% Yes
% ?- rectangle([[a,b,c],[d,e]]).
% No


% Using predicate head_tails/3:

rectangle1(R) :- all_empty(R).

rectangle1(R) :-
  head_tails(R,_,R1),
  rectangle1(R1).

all_empty([]).
all_empty([[]|R]) :- all_empty(R).



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Question 14.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

matrix(M) :- M = [[_|_]|_], rectangle(M).



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Question 15.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

transpose([[]|_],[]).
         % a ‘matrix’ with zero columns
         % becomes a ‘matrix’ with zero lines

transpose(M,[C|TM1]) :-
  head_tails(M,C,M1),
  transpose(M1,TM1).
```