5.
a. Write a predicate to compute the union of two sets.
b. Write a predicate to determine the set of all the pairs of elements in a list.
Eg.: L = [a b c d] => [[a b] [a c] [a d] [b c] [b d] [c d]].

```
% Student exercise profile
:- set_prolog_flag(occurs_check, error).        % disallow cyclic terms
:- set_prolog_stack(global, limit(8 000 000)).  % limit term space (8Mb)
:- set_prolog_stack(local,  limit(2 000 000)).  % limit environment space

% Your program goes here
% union of two sets - a)
% union_set([l1, ..., lk], [m1, ..., mj], U) = [m1, ..., mj], if k = 0
% union_set([l2, ..., lk], [m1, ..., mj], U), if l1 ∈ [m1, ..., mj]
% [l1 | U1], where union_set([l2, ..., lk], [m1, ..., mj], U1), if l1 ∉ [m1, ..., mj]

union_set([], S2, S2).

union_set([H|T], S2, U) :-
    member(H, S2),
    union_set(T, S2, U).

union_set([H|T], S2, [H|U]) :-
    \+ member(H, S2),
    union_set(T, S2, U).


% pairs(List, Pairs) -b)
% pairs([l1, ..., lk], P) =
%     [],                          if k = 0
%     [],                          if k = 1
%     find_pairs(l1, [l2, ..., lk]) ∪ pairs([l2, ..., lk])

pairs([], []).

pairs([_], []).

pairs([H|T], P) :-
    find_pairs(H, T, P1),
```

```
    pairs(T, P2),
    append(P1, P2, P).

% find_pairs(Element, List, PairsWithElement)
% find_pairs(E, [l1, ..., lk]) =
%     [],                                  if k = 0
%     [[E, l1]] ∪ find_pairs(E, [l2, ..., lk])

find_pairs(_, [], []).

find_pairs(E, [H|T], [[E,H]|P]) :-
    find_pairs(E, T, P).

/** <examples> Your example queries go here, e.g.
?- union_set([], [a,b,c], U1),
union_set([a,b,c],[d,e,a],U2),
pairs([a,b,c,d,e], P).


*/
```

Example

```
?- union_set([], [a,b,c], U1),
union_set([a,b,c],[d,e,a],U2),
pairs([a,b,c,d,e], P).
```

P = [[a, b], [a, c], [a, d], [a, e], [b, c], [b, d], [b, e], [c, d], [c, e], [d, e]],
U1 = [a,b,c],
U2 = [b,c,d,e,a]

union_set([], [a,b,c], U1), *union_set*([a,b,c],[d,e,a],U2), *pairs*([a,b,c,d,e], P).

P = [[a, b], [a, c], [a, d], [a, e], [b, c], [b, d], [b, e], [c, d], [c, e], [d, e]],
U1 = [a, b, c],
U2 = [b, c, d, e, a]

Next | 10 | 100 | 1,000 | Stop

```
?- union_set([], [a,b,c], U1),
union_set([a,b,c],[d,e,a],U2),
pairs([a,b,c,d,e], P).
```