Seminar 7 – Recapitulare

1. Codul următor este soluția pentru a șterge dintr-o listă toate elementele care se repetă. De ex. pentru lista [1,2,3,2,1] rezultatul este lista [3].

   Există mai multe variante de a rezolva problema, această soluție conține 3 predicate:

- Predicatul exista, care verifică dacă un element apare într-o listă sau nu.
- Predicatul stergeelem, care șterge toate aparițiile unui element dintr-o listă.
- Predicatul rezolva, care elimină elementele care se repetă.

```
% exista (L:lista, E: element)
% model de flux: (i,i), (i,o)
% L – lista in care verificam
% E – elementul cautat in lista L
exista([H|T],E) :- H=E.
exista([H|T],E) :- exista(T,E).

% stergeelem (L: lista, E: element, R: lista)
%model de flux: (i,i,o), (i,o,i)
% L – lista din care stergem
% E – elementul de sters
% R – lista rezultat
stergeelem([],X,[]).
stergeelem([H|T],X,R) :- H=X,stergeelem(T,X,R).
stergeelem([H|T],X,[H|R]) :- stergeelem(T,X,R).

%rezolva (L: lista, R: lista)
%model de flux (i,o)
% L – lista de transformat
% R – lista rezultat
rezolva([],[]).
rezolva([H|T],S) :- exista(T,H),
                    stergeelem(T,H,R),
                    !,
                    rezolva(R,S).
rezolva([H|T],[H|S]) :-  not(exista(T,H)),
                   rezolva(T,S).
```

- Este corect codul de mai sus?
  - Dacă ne uităm mai bine, vedem că avem o problemă în predicatul stergeelem. Predicatul este nedeterminist (are mai multe soluții), deși noi vrem un predicat determinist. De exemplu, pentru apelul stergeelem([1,2,4,2,1,6], 2, X), rezultatele vor fi:
    - [1,4,1,6]
    - [1,4,2,1,6]
    - [1,2,4,1,6]
    - [1,2,4,2,1,6]

- Deci, stergeelem e gresit, pentru ca e nedeterminist si ne dă și soluții greșite. Este atunci toată soluția greșită? În mod interesant, dacă am rula predicatul rezolva, am observa că merge bine. Ne dă soluția corectă și este deterministă. De ce?

im

2. Ați făcut la curs cod Prolog pentru a genera toate combinăriile luate a câte N dintr-o listă. Codul arată așa:

```
%comb(L: lista, N: integer, R:lista)
%model de flux (i,i,o)
%L - lista initiala
%N - cate elemente vrem in combinari
%R - rezultat
comb([E|_], 1, [E]).
comb([_|T], N, R):-
      comb(T, N, R).
comb([H|T],N, [H|R]):-
      N > 1,
      N1 is N - 1,
      comb(T, N1, R).
```

- Să presupunem că nu vrem toate combinările, ci doar cele în care elementele sunt în ordine crescătoare. Evident, se poate face un alt predicat, care să verifice dacă o listă este crescătoare și un al 3-lea predicat care să genereze o combinare și să verifice dacă este crescătoare. Dar să presupunem că noi nu vrem alt predicat, vrem să-l modificăm pe cel pe care îl avem.
    - Dacă vreau ca rezultatul să fie o listă crescătoare, modificare trebuie să fac acolo, unde adaug element în soluție, adică în a 3-a clauză. Înainte de a adăuga elementul H, trebuie să verificăm dacă respectă condiția, adică este mai mic, decât primul element din soluție. Este în regulă să modificăm a 3-a clauză așa?

      ```
      comb([H|T], N, [H,H1|R]):-
            H < H1,
            N > 1,
            N1 is N - 1,
            comb(T, N1, [H1|R]).
      ```
    - Dacă facem această modificare vom avea eroare la H < H1, pentru că variabila H1 nu are valoare. Parametrul 3 fiind de tip output, el nu are valoare in momentul apelului, ci trebuie să primească valoare în interiorul clauzei undeva. Dacă ne uităm pe varianta originală a predicatului și căutăm unde primește parametrul R valoare, vedem ca e în ultima instrucțiune, în apelul recursiv. De aceea, dacă vrem să accesăm elemente din rezultat, trebuie să facem după apelul recursiv.

      ```
      comb([H|T],N, [H,H1|R]):-
            N > 1,
            N1 is N - 1,
            comb(T, N1, [H1|R]),
            H < H1.
      ```

3. La teme de laborator apare problema următoare: să se șteargă 1-ul, al 3-lea, al 7-lea, etc. element dintr-o listă.
   Să considerăm codul următor pentru a rezolva problema:

```
%elimina(list, pozC, pozR, listR)
%model de flux: (i, i, i, o)
elimina([], _, _, []).
elimina([_|T],PC, PR, R):-
    PC =:= PR, !,
    elimina(T, PC+1, PR*2+1, R).
elimina([H|T], PC, PR, [H|R]):-
    elimina(T, PC+1, PR, R).
```

E în regulă codul anterior?