Seminar 7 – Recapitulare

Pb1. Sa se sterarga toate elementele care se repta dintr-o lista liniara data.

Ex: [1,2,3,3,1] => [2]

3 predicate:

* Verificarea apartenentei unui element la o lista
* Stergerea tuturor aparitiilor unui element dintr-o lista
* Predicatul principal care sterge toate elementele care se repeta, folosind predicatele mentionate anterior

Apare\_in\_lista ( l1 l2 .. ln, e) = { fals, daca n = 0

{ adevărat, dacă l1 = e, n > 0

{ Apare\_în\_listă( l2 l3 .. ln, e), altfel (n > 0 si l1 != e)

Apare\_in\_lista( L- Lista, E- Element)

Model de flux (I,I) determinist

E - elementul pe care il cautam

L – lista in care cautam elementul

Apare\_in\_lista(H ,[H|T] ).

Apare\_in\_lista(E, [H|T]) :- not(E=H),

Apare\_in\_lista(E,T).

not()

= \=

Is

=:= =\=

Apare\_in\_lista(H ,[H|T]) :- !.

Apare\_in\_lista(E, [H|T]) :- Apare\_in\_lista(E,T).

Sterge\_din\_lista(l1 l2 .. ln, e) = { (), daca n = 0

{ l1 (+) sterge\_din\_lista(l2 l3 .. ln, e), daca l1 != e, n > 0

{ sterge\_din\_lista(l2 l3 .. ln, e), altfel

% Sterge\_din\_lista(L,E,Rez)

% L : lista

% E: elemente

% Rez: lista rezultat

% model de flux (i,I,o) determinist

Sterge\_din\_lista([],\_,[]).

Sterge\_din\_lista([H|T],E,Rez1):-

H \= E,

!,

Sterge\_din\_lista(T,E,Rez),

Rez1 = [H|Rez].

Sterge\_din\_lista([H|T],H,Rez1):-

Sterge\_din\_lista(T,H,Rez1).

Sterge\_din\_lista([],\_,[]).

Sterge\_din\_lista([H|T],E,[H|Rez]):-

H \= E,

Sterge\_din\_lista(T,E,Rez).

Sterge\_din\_lista([H|T],H,Rez1):-

Sterge\_din\_lista(T,H,Rez1).

Principal(l1...ln) = {[], n = 0

{ principal(sterge\_din\_lista(l2...ln, l1)) , n > 0 si apare\_in\_lista(l1, l2...ln)

{ l1 (+) principal(l2...ln), altfel (n > 0 si !apare\_in\_lista(l1, l2...ln)

Principal( L – lista, Rez - lista)

L – lista din care stergem elementele care se repeta

Rez – lista rezultata prin stergerea elemetelor care se repeta

Model de flux (I,o) detereminism

(I,I) determinism

Principal( [], [] ).

Principal([ H|T], R) :- apare\_in\_lista(H, T), !,

Sterge\_din\_lista(T,H,R1),

Principal( R1, R).

Principal([H|T|, [H|R]): - Principal(T, R).

Apare\_in\_lista(H ,[H|T] ).

Apare\_in\_lista(E, [H|T]) :- Apare\_in\_lista(E,T).

Exemplu: Apare\_in\_lista(1 ,[2,3]).

False.

Apare\_in\_lista(1 ,[1,3, 2]).

True;

False.

Apare\_in\_lista(1 ,[1,3,1]).

True;

True;

False.

Sterge\_din\_lista([],\_,[]).

Sterge\_din\_lista([H|T],H,Rez1):-

Sterge\_din\_lista(T,H,Rez1).

Sterge\_din\_lista([H|T],E,[H|Rez]):-

Sterge\_din\_lista(T,E,Rez).

Sterge([1,2,4,2,1,6], 2, X).

X = [1, 4, 1, 6]

X = [1, 4, 2, 1, 6]

X = [1, 2, 4, 1, 6]

X = [1, 2, 4, 2, 1, 6]

Principal( [], [] ).

Principal([ H|T], R) :- apare\_in\_lista(H, T),

Sterge\_din\_lista(T,H,R1),

!,

Principal( R1, R).

Principal([H|T|, [H|R]): - Principal(T, R).

B. findalll

U [numele\_predicatului\_nedeterminist(argumentele\_predicatului\_nedeterminist)]

2. Să se ștearga 1-ul, al 3-lea, al 7-lea, etc. element dintr-o listă.

%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).

Principal(Argi,LS):-

Findall(X, predNeter(Argi, X),LS).