Seminar 7 – Recapitulare Prolog

P1. Se da o lista liniara, numerica.

Se cere sa se elimine toate elementele care se repeta.

Exemplu:

[1,2,3,2,1,1,4] => [3, 4]

Idei de rezolvare

1.

2.

3. 3 predicate:

- exista

- eliminaAparitii

- rezolva

Exista(e, l1l2...ln)= fals, n=0

adevarat, e = l1 si n!=0

exista(e, l2l3...ln), e!=l1 si n!=0

%Exista(E: INT, L: LIST)

%model de flux: (i, i) determinist.

Exista(E, [E|\_]):-!.

Exista(E, [\_|T]):- Exista(E, T).

EliminaAp(l1..ln, e) = [], n=0

eliminaAl(l2..ln, e), n !=0 si l1 = e

l1 (+) eliminaAp(l2..ln, e), n != 0 si l1 != e

% EliminaAp(L: List, E: Intreg, Rez:List)

%mode

l de flux (i, i, o) - determinist

EliminaAp([], \_, []).

EliminaAp([H|T], E, Rez):-

H = E,

EliminaAp(T, E, Rez).

EliminaAp([H|T], E, [H|Rez]):-

H \= E,

EliminaAp(T, E, Rez).

Rezolva(l1...ln)= [],n=0

= rezolva(eliminaAp(l2...ln),l1), exista(l1,l2...ln) si n>0

=l1(+)rezolva(l2...ln),altfel

Rezolva(L:Lista,Rez:Lista)

Model de flux: (i, o) - determinist

Rezolva([],[]).

Rezolva([H|T],Rez):-

Exista(H, T),!,

EliminaAp(T, H, TRez),

Rezolva(TRez, Rez).

Rezolva([H|T], Rez):- Rezolva(T, Rez).

Exista(E, [E|\_]).

Exista(E, [\_|T]):- Exista(E, T).

Ex: Exista(1, [1,2,1,1].

EliminaAp([], \_, []).

EliminaAp([H|T], E, [H|Rez]):-

EliminaAp(T, E, Rez).

EliminaAp([H|T], H, Rez):-

EliminaAp(T, E, Rez).

EliminaAp([1,2,1,3],1,R).

R = [2,3];

R=[2,1,3];

R=[1,2,3];

R=[1,2,1,3].

Rezolva([],[]).

Rezolva([H|T],Rez):-

Exista(H, T),

EliminaAp(T, H, TRez), !,

Rezolva(TRez, Rez).

Rezolva([H|T], [H|Rez]):- Rezolva(T, Rez).

P2. Se da o lista numerica liniara.

Sa se elimine elementele de pe pozitiile 1, 3, 7, 15 etc. din lista.

Ex: [1,2,3,4,5,6,7,8] => [2,4,5,6,8]

EliminPoz(l1...ln,contor,p) = [] , n =0

= l1 (+) eliminPoz(l2..ln,contor+1,p), contor !=p, n!=0

= eliminPoz(l2...ln,contor+1,2\*p+1), contor = p, n !=0

%EliminPoz(L: Lista, Contor: Intreg, P: Intreg, Rez: Lista)

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

EliminPoz([], \_, \_, []).

EliminPoz([H | T], Contor, P, [H | Rez]) :-

Contor \= P, !,

Contor1 is Contor + 1,

EliminPoz(T, Contor1, P, Rez).

EliminPoz([H | T], Contor, P, Rez) :-

Contor1 is Contor + 1,

P1 is P \* 2 + 1,

EliminPoz(T, Contor1, P1, Rez).

EliminPoz([], \_, \_, []).

EliminPoz([H | T], Contor, Contor, Rez) :- !

EliminPoz(T, Contor + 1, Contor \* 2 + 1, Rez).

EliminPoz([H | T], Contor, P, [H | Rez]) :-

EliminPoz(T, Contor + 1, P, Rez).

=\= negarea pentru =:=

=

Is

=:

P3.

Comb([E|\_], 1, [E]).

Comb([\_|T], N, R):-

Comb(T, N, R).

**Comb([H|T], N, [H,H1|R]):-**

**N > 1,**

**N1 is N-1,**

**Comb(T, N1, [H1,R]),**

**H < H1.**

[1,3,2,4,6,5]

N = 3

[1,2,6] sol

[1,3,2] nu sol

[1,2,3] sol