

R<sub>1</sub>

I. Funcția FC reunește toate rezultatele unei funcții și apărăte pe fiecare element al listei x, începând cu al 3-lea.

În apelul său, funcția φ este CAR  $\Rightarrow$  se ia primul element din fiecare elem. al listei x, începând cu al 3-lea.

Așadar, rezultatul este: (3 4)

mapor returnarea de la listă

2. Predicatorele adaugă elementele struct pozitive în rezultat, în același ordine în care se aplică în lista inițială.

$\Rightarrow$  Rezultatul evaluării date este: [2, 3, 7].

Cu -ve nu va avea decât efectul de a opri backtracking-ul astunci cănd elementul curent este struct pozitiv (pentru a nu avea un predicat nedeterminat).

3. Funcția G returnează lista rezultată din primul și al 3-lea element al unei liste date.

(set g Q 'G)  $\rightarrow$  Q se va evalua la G

(set g P 'Q)  $\rightarrow$  P se va evalua la Q

(EVAL P)  $\rightarrow$  Recurziv: evaluarea la G

se evaluă ca (LIST ((A B C)))  $\rightarrow$  ((A B C))

și se apelează funcția G cu argumentul (A B C)

Rezultatul este: (A C)

} se trimit parametrii  
într-o listă pt. ca să  
poată fi aplicate

4. Predicatorele dat nu va modifica rezultatul care este inițializat cu 0, dar va cere val. sa „cărui” eler este param. de iesire

$\Rightarrow$  eroare

Problema e ce s-ar fi dorit sa se rezolze, ar fi raspunsul sumă tuturor elementelor din lista. Pentru a face acest lucru însai, ar trebui să avem:  $S \in S_1 - H$ .

II Se generaza lista aranjamentele de la elemente, cu permutările.

înserare( $e_1 \dots e_n, e$ ) =

1.  $e \oplus e_1 \dots e_n$
2.  $e_1 \oplus \text{înserare}(e_2 \dots e_n, e)$

mai mult ( $e, R, p$ ) =

=  $\bigcup$  aranjamente( $e, R, p$ )

aranjamente ( $e_1 \dots e_n, R, p$ ) =

1.  $(e_1), R=1 \text{ și } p=e_1$
2. aranjamente( $e_2 \dots e_n, R, p$ )

3. înserare(aranjamente( $e_2 \dots e_n, R-1, [p/e_1]$ )), e) !!!

% inserare ( $L$  - lista,  $E$  - element,  $R$  - lista rezultată) Inchise verificat și  
cu p = dimensiunea lui  $E$

% funcția inserare elementul  $E$  în lista  $L$  pe toate pozitii

% model de flux:  $(i, i, \alpha)$  - ne determinist

înserare ( $L, E, [E|L]$ ).

înserare ( $[H|T], E, R|T$ ): -

înserare ( $T, E, R$ ),

$R|T = [H|R]$ .

% aranjamente ( $L$  - lista de elemente,  $K$  - Nr. mat  $\rightarrow$  dimensiunea

% aranjamentele,  $P$  - Nr.  $\rightarrow$  dimensiunea

% model de flux:  $(i, i, i, \alpha)$  - ne determinist

aranjamente ( $[H|L], K, P, [H]$ ): -  $K=1, P=H$ .

aranjamente ( $L-ITJ, K, P, Q$ ): - aranjamente ( $T, K, P, R$ ).

aranjamente ( $L, K, P, R_2$ ) :-

$K_1$  is  $K-1$ ,  $P_1$  is  $P$  div  $H$ ,

aranjamente ( $T, K_1, R$ ),

inserirea ( $R$ ,  $H$ ,  $R_2$ ).

- % maxim ( $L$  - lista de elemente,  $K$  - Nr. dat,  $P$  - Nr.,  $R$  - lista rezultat)
- % procedura va prezenta o lista cu toate aranjamentele
- % pozitie
- % modele de flux: ( $i, i, i, \alpha$ ) - deterministic

maxim ( $L, K, P, R$ ) :-

fundatia ( $R_1$ , aranjamente ( $L, K, P, R_1$ ),  $R$ ).

III Se da un eveniment  $\rightarrow$  să se împăcească modurile de pe niveluri impari ce că valoarea este  
nivelul maxim = 0

împăcuire ( $e, \text{min}, e$ ) =  $\begin{cases} e, & e \text{ este } \text{min} \text{ și } \text{min} \text{ e impar} \\ \emptyset, & e \text{ este } \text{min} \text{ și } \text{min} \text{ e par} \\ \bigcup_{i=1}^{\text{min}} \text{împăcuire } (e, \text{min}+1, e), & \text{altele} \end{cases}$

maxim ( $e, e$ ) = împăcuire ( $e, -1, e$ )

- ; împăcuire ( $e$  - lista) Atunci  $\rightarrow$  inițiala evenimentelor sunt, min - nivelul curent
- ; al modului,  $e$  - elementul împăcuitelor)

(debut împăcuire ( $e$  min  $e$ ))

(cond

- ; dacă  $e$  e mod și nivelul este impar  $\Rightarrow$  împăcuirea
- ((cond (atunci  $e$ ) (adăp min)))  $e$ )
- ((atunci  $e$ )  $e$ )

- ; dacă  $e$  există, vom apela funcția de inserare
- ; pe fiecare element și vor fi rezultate, cu list,
- ; rezultatelor

(+ (mapcar #'(lambda (x)

(inserare x (+ mil' 1) e))

)

e

- ; s-a utilizat funcția lambda, deoarece
- ; nu am fi putut adăuga param. mult;
- ; și la apelul altfel

)

,

)

; main (l -> rezultatul, e - elementul inserator)  
(de lung maim (l e))

(inserare l -1 e)

- ; nivelul este initializat cu -1, deoarece prima oară cănd
- ; se intră în apel, pentru a accesa trădăcina și să se genereze,
- ; se trebuie să aibă acces la elem. Elster, crescând valoarea
- ; nivelului curent (în parț. este ca nivelul rezultatului)
- ; să fie 0)

)