

1. Funcția G calculează produsul primelor 2 elemente ale unei liste.

(setă căr' G) \rightarrow căr' se va evalua ea G

Totuși, pentru a apela funcția G prin intermediul lui CAR, va fi nevoie de a evalua:

- una pt. a lega de la CAR ea G

- una pt. a evalua G ea funcția pe care o reprezintă

Dacă în procesul lui CAR ar fi fost ca denumirea ce nu coincide cu denumirea unei funcții predefinite, apoi el își rezolvă să îl dat eroare, megașimol și funcție cu numele respectiv (s-ar fi evaluat CAR ea G, dacă ar mai fi fost nevoie de încă o evaluare pentru apel). Totuși, există o funcție cu numele CAR și atunci se va propasi definitia acesteia.

\Rightarrow rezultatul lui: (CAR '(2 3 5 6)) este 2.

3. Aranjamente de pe elemente cu o sumă dată.

(PROLOG)

Modele matematice:

înserarea ($e_1 \dots e_m, x$) =

1. $x \oplus e_1 \dots e_m$

2. $e_1 \oplus \text{înserarea} (e_2 \dots e_m, x)$, $n > 1$

cremij ($e_1 \dots e_m, p_a, s$) =

1. (e_1), dacă $p_a = 1$ și $s = e_1$

2. $\text{cremij} (e_2 \dots e_m, p_a, s)$

3. $\text{înserarea} (\text{cremij} (e_2 \dots e_m, p_a - 1, s - e_1), e_1)$,
dacă $e_1 < s$ și $p_a > 1$

$\text{main}(e, \beta_0, S) = \bigcup \text{arang}(e, \beta_0, S)$

inserarea ($L, E, [E(L)]$).

model de flux: (c, i, ce)

inserarea ($[HITS], E, R$): -

determinist

inserarea (T, E, R_1),
 $R = [H|R]$.

ceranj ($[H-T], L, H, [H]$).

model de flux:

(c, i, l, ce)

ceranj ($L-HT], K, S, R$): -

determinist

aranj (T, K, S, R).

ceranj ($HHT], K, S, R$): -

$K > L$,

$S > H$,

$K_1 \leq K - 1$,

$S_1 \leq S - H$,

aranj (T, K_1, S_1, R_1),

inserarea (R_1, H, R).

main (L, K, S, R): -

functie ($R_1, \text{aranj}(L, K, S, R_1), R$).

model de flux: (c, i, l, ce) \rightarrow determinist

4. Nr. de moduri de pe nivelul R . (LISP)

(nivel superficial = L)

modele matematice:

$$\text{numara}(e, \beta_0, miv) = \begin{cases} 1, & \text{daca } miv = \beta_0 \\ 0, & \text{daca } miv \neq \beta_0 \\ \sum_{i=1}^n \text{numara}(e_i, \beta_0, miv+1), & \text{altele} \end{cases}$$

$$\text{mcum}(e_1, e_2) = \text{numera}(e_1, e_2, 0)$$

↪ pt. că numerația presupune că
lista întregă (min=0), că
pt. să ajunge că min. suprafața și
trebuie să treacă pe un min.
suprafață (min=1)

(definim numerația (e la min))

(cond)

$$((\text{atom } e) (\text{egual la min})) \perp$$

$$((\text{atom } e) \circ)$$

$$(+ (\text{aply } \#) + (\text{mapcar } \#) (\text{fameдалy}))$$

(numerația y la ($+ \text{min}$))

)

)

(definim mcum(e la))

$$(\text{numera } e \text{ la } \circ)$$

)

5. Nr. de subcărți pt. care primită atom numeric este impar.

Modeluri matematice:

$$\text{prum}(e_1 \dots e_n) = \begin{cases} \emptyset, & n=0 \\ e_1, & e_1 \text{ e nr.} \\ \text{prum}(e_1), & e_1 \text{ e cărți și } \text{prum}(e_1) \neq \emptyset \\ \text{prum}(e_2 \dots e_n), & \text{altele} \end{cases}$$

$$\text{subcărți}(e) = \begin{cases} \emptyset, & e \text{ e atom} \\ 1 + \sum_{i=1}^n \text{subcărți}(e_i), & \text{prum}(e) \text{ este nr. impar} \\ \sum_{i=1}^n \text{subcărți}(e_i), & \text{altele} \end{cases}$$

(defun psum (e))

```
(cond  
  ((null e) nil)  
  ((numberp (car e)) (car e))  
  ((and (eqlp (car e)) (psum (cdr e)))  
   (psum (cdr e)))  
  (+ (psum (cdr e))))  
 ) )
```

(defun seqiste (e))

```
(cond  
  ((atom e) 0)  
  ((and (numberp (psum e)) (oddp (psum e))))  
   (+ 1 (apply #'+ (mapcar #'seqiste (cdr e)))))  
  (+ (apply #'+ (mapcar #'seqiste (cdr e))))  
 ) )
```