Seminar 5 – Programare recursiva in Lisp

Enunt pb 1.

Se dau doua liste numerice, ordonate crescator, formate din elemente distincte.

Se cere sa se scrie o functie care interclaseaza, fara pastrarea dublurilor, cele doua liste.

Exemplu:

L1 = [-2 3 7 9 11]

L2 = [-4 0 3 5 8 9 10]

=> L = [-4 –2 0 3 5 7 8 9 10 11]

Interclasare(l1l2..ln, s1s2..sm) = { s1s2..sm , daca n = 0

={ l1l2..ln , daca m = 0 si n!=0

={ l1 (+) interclasare(l2l3..ln, s1s2..sm), daca n > 0 si m>0 si l1 < s1

={ s1 (+) interclasare(l1l2..ln, s2s3..sm), daca n > 0 si m>0 si s1 < l1

={ l1 (+) interclasare( l2l3..ln,s2s3..sm) , daca n > 0 si m>0 si l1 = s1

(defun interclasare (L S)

(cond

((null L) S)

((null S) L)

((< (car L) (car S)) (cons (car L) (interclasare (cdr L) S)))

((< (car S) (car L)) (cons (car S) (interclasare L (cdr S))))

((= (car S) (car L)) (cons (car L) (interclasare (cdr L) (cdr S))))

)

)

|  |  |  |  |
| --- | --- | --- | --- |
| Arg | cons | list | append |
| ‘A ‘B | (A.B) | (A B) | Eroare |
| ‘A ‘(B) | (A B) | (A (B)) | Eroare |
| ‘(A) ‘(B) | ((A) B) | ((A) (B)) | (A B) |
| ‘A ‘B ‘C | Eroare | (A B C) | Eroare |
| ‘A ‘(B) ‘C | Eroare | (A (B) C) | Eroare |
| ‘(A) ‘(B) ‘C | Eroare | ((A) (B) C) | (A B.C) |

Enunt pb 2.

Se da o lista eterogena.

Se cere sa se elimine toate aparitiile unui atom dat din lista data.

Exemplu:

L = [9 1 [] 4 a 5 [1] 2 [4 5 1] 1]

E = 1

=> [9 [] 4 a 5 [] 2 [4 5]]

Elimina(l1 l2 .. ln, e) = { lista vida, daca n = 0

{ l1 (+) elimina (l2 .. ln, e), daca l1 = atom, l1 != e

{ elimina (l2 .. ln, e), daca l1 = atom, l1 = e

{ elimina(l1, e) (+) elimina (l2 .. ln, e), daca l1 = lista

(defun elimina ( L E)

(cond

(( null L) () )

((and (atom (car L) (not (equal (car L) E ) )) (cons ( car L) (elimina (cdr L) E )) )

((equal (car L) E) (elimina (cdr L) E) )

(t (cons (elimina (car L) E) (elimina (cdr L) E) ) )

)

)

()

nil

‘()

‘nil

Pb3.

Se da lista liniara.

Se cere sa se determine lista pozitiilor pe care apare elementul numeric minim in lista data.

Exemplu:

L = (a –3 –10 5 b d –10 1 0)

=> ( 3 7)

Numara\_poz\_min(l1,..ln, col, Min, poz) = { col, n == 0

{ numara\_poz\_min( l2...ln, (poz), l1, poz+1 ) , numar(l1) si l1 < Min

{ numara\_poz\_min(l2...ln, poz (+) Col , Min, poz+ 1), numar(l1) si l1 = min

{ numar\_poz\_min(l2... ln, Col, Min, poz+1), altfel

[**most-positive-double-float**](http://clhs.lisp.se/Body/v_most_1.htm#most-positive-double-float)

Numara(l1,..ln) = Numara\_poz\_min(l1,..ln, (), +inf, 1)

(defun Numara\_poz\_min (L C M P)

(cond

((null L) C)

((and ( numberp (car L)) (< (car L) M)) (numara\_poz\_min (cdr L) (list P) (car L) (+ P 1) ))

( (equal (car L) M) ( numara\_poz\_min (cdr L) (cons P C) M (+ P 1)))

(t (numara\_poz\_min ( cdr L) C M (+ P 1)))

)

)

(defun Numara (L)

(Numara\_poz\_min L () [**most-positive-double-float**](http://clhs.lisp.se/Body/v_most_1.htm#most-positive-double-float) **1)**

)