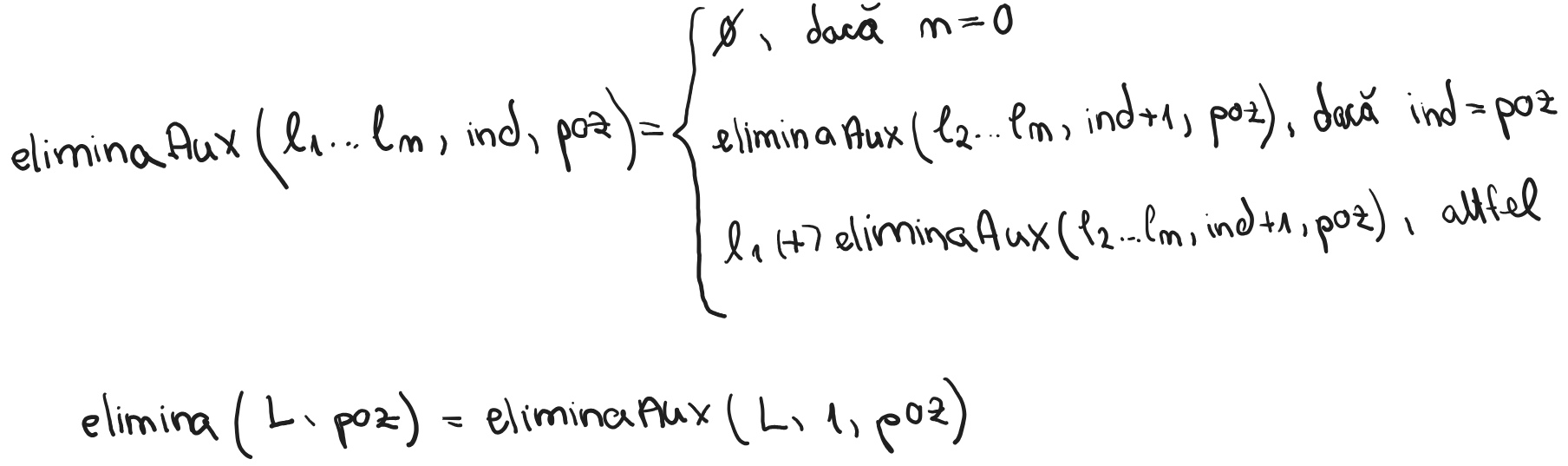
LABORATOR L1: Problema 8

# a) Sa se elimine elementul de pe pozitia a n-a a unei liste liniare.

## MODEL RECURSIV



## COD

;eliminaAux(L:list ind:integer poz:integer)

;L - lista din care se elimina elementul de pe pozitia poz

;ind - indicele ce va simula parcurgerea listei

;poz - pozitia de pe care se sterge

;model de flux: (i,i,i)

(defun eliminaAux(L ind poz)

(cond

((null L)nil)

((equal ind poz)(eliminaAux (cdr L) (+ ind 1) poz))

(T (cons (car L) (eliminaAux (cdr L) (+ ind 1) poz)))

)

)

;elimina(L:list poz:integer)

;L - lista din care se elimina elementul de pe pozitia poz

;poz - pozitia de pe care se sterge

;model de flux (i,i)

(defun elimina(L poz)

(eliminaAux L 1 poz)

# )b) Definiti o functie care determina succesorul unui numar reprezentat cifra cu cifra intr-o lista. De ex: (1 9 3 5 9 9) --> (1 9 3 6 0 0)

## MODEL RECURSIV

A close-up of a math problem

Description automatically generated

A person standing next to a white sheet of paper

Description automatically generated



A close up of a handwritten text

Description automatically generated

## COD

;inversAux(L:list, invers:list)

;L - lista care se inverseaza

;invers - inversul listei L

;model de flux (i,i)

(defun inversAux(L invers)

(cond

((null L)invers)

(T (inversAux (cdr L) (cons (car L) invers)))

)

)

;invers(L:list)

;L - lista care se inverseaza

;model de flux (i)

(defun invers(L)

(inversAux L nil)

)

;aduna(A:list B:list transport:integer)

;A - lista care reprezinta primul numar

;B - lista care reprezinta al doilea numar

;transport - transportul care se acumuleaza de la adunarea cifrelor

;model de flux (i,i,i)

(defun aduna(A B transport)

(cond

((and (= (length A) 0) (= (length B) 0) (= transport 0)) nil)

((and (= (length A) 0) (= (length B) 0) (not(= transport 0))) (list transport))

((and (not(= (length A) 0)) (= (length B) 0)) (cons (mod (+ (car A) transport) 10) (aduna (cdr A) nil (truncate (+ (car A) transport) 10))))

((and (= (length A) 0) (not (= (length B) 0))) (cons (mod (+ (car B) transport) 10) (aduna nil (cdr B) (truncate (+ (car B) transport) 10))))

((and (not(= (length A) 0)) (not (= (length B) 0))) (cons (mod (+ (car A) (car B) transport) 10) (aduna (cdr A) (cdr B) (truncate (+ (car A) (car B) transport) 10))))

)

)

;succesor(L:list)

;L - numarul sub forma de lista pentru care calculam succesorul

;model de flux (i)

(defun succesor(L)

(invers (aduna (invers L) (list 1) 0))

)

# c) Sa se construiasca multimea atomilor unei liste.Exemplu: (1 (2 (1 3 (2 4) 3) 1) (1 4)) ==> (1 2 3 4)

## MODEL RECURSIV

A whiteboard with writing on it

Description automatically generated

A white paper with black text

Description automatically generated

## COD

;member1(L:list E:integer)

;L - lista in care se cauta

;E - elementul cautat

;model de flux (i, i)

(defun member1(L E)

(cond

((null L) nil)

((equal (car L) E) T)

(T (member1 (cdr L) E))

)

)

;multime(L:list M:list)

;L - lista din care se obtine multimea

;M - multimea rezultata

;model de flux (i,i)

(defun multime(L M)

(cond

((null L) M)

((and (atom (car L)) (not (member1 M (car L)))) (multime (cdr L) (cons (car L) M)) )

((and (atom (car L)) (member1 M (car L))) (multime (cdr L) M))

(T (multime (cdr L) (multime (car L) M)))

)

)

;multimeMain(L:list)

;L - lista din care se obtine multimea

;model de flux (i)

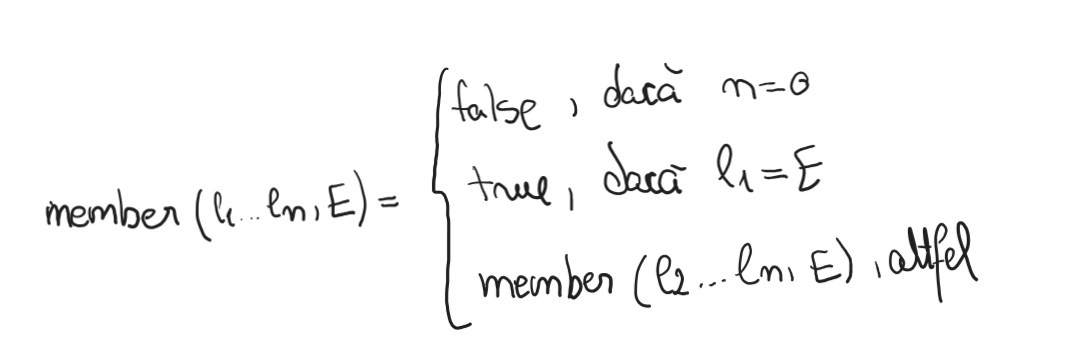
(defun multimeMain(L)

(multime L nil)

)

# d) Sa se scrie o functie care testeaza daca o lista liniara este o multime.

## MODEL RECURSIV



A white board with black text

Description automatically generated

## COD

;member1(L:list E:integer)

;L - lista in care se cauta

;E - elementul cautat

;model de flux (i, i)

(defun member1(L E)

(cond

((null L) nil)

((equal (car L) E) T)

(T (member1 (cdr L) E))

)

)

;isMultime(L: list)

;L - lista pe care o verificam

;model de flux (i)

(defun isMultime(L)

(cond

((null L) T)

((member1 (cdr L) (car L)) nil)

(T (isMultime (cdr L)))

)

)