

# Programare logică și funcțională

## - examen scris -

### Notă

1. Subiectele se notează astfel: of - 1p; A - 1.5p; B - 2.5p; C - 2.5p; D - 2.5p.
2. Problemele Prolog vor fi rezolvate în SWI Prolog. Se cere: (1) explicarea codului și a raționamentului; (2) modelul recursiv de rezolvare, pentru fiecare predicat folosit; (3) specificarea fiecărui predicat (semnificația parametrilor, model de flux, tipul predicatului - determinist/nedeterminist).
3. Problemele Lisp vor fi rezolvate în Common Lisp. Se cere: (1) explicarea codului și a raționamentului; (2) modelul recursiv de rezolvare, pentru fiecare funcție folosită; (3) specificarea fiecărei funcții (semnificația parametrilor).

**A.** Fie următoarea definiție de predicat PROLOG **f(integer, integer)**, având modelul de flux (i, o):

f(50, 1):-!.  
f(I,Y):-J is I+1, **f(J,S)**, S<1, !, K is I-2, Y is K.

f(I,Y):-J is I+1, **f(J,Y)**.

Rescrieți această definiție pentru a evita apelul recursiv **f(J,V)** în ambele clauze. Nu redefiniți predicatul. Justificați răspunsul.

- B.** Dându-se o listă neliniară care conține atomi numerici și nenumeriți, se cere un program Lisp care înlocuiește fiecare atom numeric par de pe niveluri impare cu suma cifrelor. Nivelul superficial este impar. De exemplu, pentru lista (A 2 (B 31 F (D 102 5 T (66) E) (D 10 (E R 51)) 99)) rezultatul va fi (A 2 (B 31 F (D 3 5 T (66) E) (D 1 (E R 51)) 99)).

- C. Să se scrie un program PROLOG care generează lista submulțimilor formate cu elemente unei liste listă de numere întregi, având suma elementelor număr impar și număr impar de elemente impare. Se vor scrie modelele matematice și modelele de flux pentru predicatele folosite.

**Exemplu**- pentru lista  $[2,3,4] \Rightarrow [[2,3],[3,4],[2,3,4]]$  (nu neapărat în această ordine)

- D. Se consideră o listă neliniară. Să se scrie o funcție care să aibă ca rezultat lista inițială în care atomii de pe nivelul **k** au fost înlocuiți cu 0 (nivelul superficial se consideră 1). **Se va folosi o funcție MAP.**

**Exemplu** pentru lista (a (1 (2 b)) (c (d)))

**a)** k=2 => (a (0 (2 b)) (0 (d)))    **b)** k=1 => (0 (1 (2 b)) (c (d)))    **c)** k=4 => lista nu se modifică