

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 funcție LISP

```
(DEFUN F(N)
  (COND
    ((= N 0) 0)
    (> (F (- N 1)) 1) (- N 2))
    (T (+ (F (- N 1)) 1))
  )
)
```

Rescrieți această definiție pentru a evita dublul apel recursiv (**F (- N 1)**). Nu redefiniți funcția. Nu folosiți SET, SETQ, SETF. Justificați răspunsul.

- B.** Dându-se o listă formată din numere întregi și subliste de numere întregi, se cere un program SWI-Prolog care verifică dacă toate elementele listei (inclusiv și cele din subliste) formează o secvență simetrică. De exemplu, pentru lista [1, 5, [2,4], 7, 11, 25, [11, 7, 4], 2, 5, 1] rezultatul va fi **true**.

- C. Să se scrie un program PROLOG care generează lista aranjamentelor de **k** elemente dintr-o listă de numere întregi, pentru care produsul elementelor e mai mic decât o valoare **V** dată. Se vor scrie modelele matematice și modelele de flux pentru predicatele folosite.

Exemplu- pentru lista [1, 2, 3], **k**=2 și **V**=7 \Rightarrow [[1,2],[2,1],[1,3],[3,1],[2,3],[3,2]] (nu neapărat în această ordine)

- D. Un arbore n-ar se reprezintă în LISP astfel (nod subarbore1 subarbore2). Se cere să se determine calea de la rădăcină către un nod dat. **Se va folosi o funcție MAP.**

Exemplu pentru arborele (a (b (g)) (c (d (e)) (f)))

a) nod=e => (a c d e) **b)** nod=v => ()