

# 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(G L)
  (COND
    ((NULL L) NIL)
    (> (FUNCALL G L) 0) (CONS (FUNCALL G L) (F (CDR L))))
    (T (FUNCALL G L))
  )
)
```

Rescrieți această definiție pentru a evita apelul repetat **(FUNCALL G L)**. Nu redefiniți funcția. Nu folosiți SET, SETQ, SETF. Justificați răspunsul.

- B. Dându-se o listă liniară de numere, scrieți un program SWI-PROLOG care returnează (sub forma unei liste de perechi) toate posibilele partiții ale listei inițiale în două subliste, astfel încât media valorilor elementelor din prima sublistă este mai mică sau egală cu cel mai mare divizor comun al elementelor din a doua sublistă. **De exemplu**, pentru lista [8, 4, 6, 1], rezultatul va fi (nu neapărat în această ordine): [[1, 4, 8], [6]], [[1, 6, 4], [8]], [[1, 6], [4, 8]], [[1], [6, 4, 8]].

- C. Să se scrie un program PROLOG care generează lista combinărilor de **k** elemente dintr-o listă de numere întregi, având suma număr par. Se vor scrie modelele matematice și modelele de flux pentru predicatele folosite.

**Exemplu** pentru lista [6, 5, 3, 4], **k**=2  $\Rightarrow$  [[6,4],[5,3]] (nu neapărat în această ordine)

- D. Se consideră o listă neliniară. Să se scrie o funcție LISP care să aibă ca rezultat lista inițială din care au fost eliminate toate aparițiile unui element **e**. **Se va folosi o funcție MAP.**

**Exemplu**

- a)** dacă lista este (1 (2 A (3 A)) (A)) și **e** este A => (1 (2 (3)) NIL)  
**b)** dacă lista este (1 (2 (3))) și **e** este A => (1 (2 (3)))