

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(L)
  (COND
    ((NULL L) NIL)
    (> (F (CAR L)) 0) (CONS (F (CAR L)) (F (CDR L))))
    (T (F (CAR L)))
  )
)
```

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

- B.** Dându-se o listă liniară de numere, să se scrie un program SWI-PROLOG care construiește o listă de liste astfel: primul element este lista inițială, iar ulterior, fiecare element este reprezentat de lista precedentă, în care secvențele crescătoare de numere au fost inversate. Ultimul element al listei va fi lista în care toate elementele sunt ordonate descrescător. **De exemplu**, pentru lista [1,3,6,5,2] rezultatul va fi: [[1, 3, 6, 5, 2], [6, 3, 1, 5, 2], [6, 3, 5, 1, 2], [6, 5, 3, 2, 1]].

- 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 lista nodurilor de pe nivelurile pare din arbore (în ordinea nivelurilor 0, 2, ...). Nivelul rădăcinii se consideră 0. **Se va folosi o funcție MAP.**
- Exemplu** pentru arborele (a (b (g)) (c (d (e)) (f))) => (a g d f)