Programare logică și funcțională
- examen scris -

Notă

1. Subiectele se notează astfel: of - 1p; A – 2p; B - 4p; C - 3p.
2. Problema Prolog (B) vor fi rezolvată 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. Problema Lisp (C) va fi rezolvată 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 definiţie de funcţie în LISP

```
(DEFUN F(L)
        (COND
                ((NULL L) NIL)
                ((LISTP (CAR L)) (APPEND (F (CAR L)) (F (CDR L)) (CAR (F (CAR L)))))
                (T (LIST(CAR L)))
        )
 )
```

Rescrieți această definiție pentru a evita dublul apel recursiv **(F (CAR L))**, fără a redefini logica clauzelor și fără a folosi o funcție auxiliară. Nu folosiți SET, SETQ, SETF. Justificați răspunsul.

B. Să se scrie un program PROLOG care generează lista submulţimilor formate cu elemente unei liste listă de numere întregi, având număr suma elementelor număr impar și număr par nenul de elemente pare. Se vor scrie modelele matematice și modelele de flux pentru predicatele folosite.

Exemplu- pentru lista [2,3,4] ⇒ [[2,3,4]]

C. Un arbore n-ar se reprezintă în LISP astfel (nod subarbore1 subarbore2 .....). Se cere să se determine înălțimea unui nod în arbore. **Se va folosi o funcție MAP.**

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

a) nod=e => înălțimea e 0 **b)** nod=v => înălțimea e -1 c**)** nod=c => înălțimea e 2