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 LISP

```
(DEFUN F(L)
        (COND
                ((NULL L) 0)
                ((> (F (CAR L)) 2) (+ (CAR L) (F (CDR L))))
                (T (F (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. Dându-se o listă formată din numere întregi, să se genereze în PROLOG lista aranjamentelor cu număr par de elemente, având suma număr impar. Se vor scrie modelele matematice și modelele de flux pentru predicatele folosite.

Exemplu- pentru lista L=[2,3,4] ⇒ [[2,3],[3,2],[3,4],[4,3]] (nu neapărat în această ordine)

C. Se consideră o listă neliniară. Să se scrie o funcţie LISP care să aibă ca rezultat lista iniţială din care au fost eliminați toți atomii numerici pari situați pe un nivel impar. Nivelul superficial se consideră a fi 1. **Se va folosi o funcție MAP.**

Exemplu **a)** dacă lista este (1 (2 A (4 A)) (6)) => (1 (2 A (A)) (6))

b) dacă lista este (1 (2 (C))) => (1 (2 (C)))