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 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))**. Nu redefiniți funcția. Nu folosiți SET, SETQ, SETF. Justificați răspunsul.

B. Dându-se o listă liniară formată din numere, se cere un program SWI-PROLOG care să furnizeze lista în care fiecare număr care este mai mic decât succesorul său din listă este înmulțit cu 2. Repetați această operație până când nu mai sunt posibile modificări în listă. **De exemplu**, pentru lista [1, 2, 3] rezultatul va fi [8, 16, 3].

C. Să se scrie un program PROLOG care generează lista submulţimilor cu **N** elemente, cu elementele unei liste, astfel încât suma elementelor dintr-o submulțime să fie număr par. Se vor scrie modelele matematice și modelele de flux pentru predicatele folosite.

Exemplu- pentru lista L=[1, 3, 4, 2] şi N=2 $\Rightarrow$ [[1,3], [2,4]]

D. 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