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 (CDR L)) 2) (+ (F (CDR L)) (CAR L)))
                (T (+ (F (CDR L)) 1))
        )
)
```

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

B. Dându-se o listă liniară de numere, se cere un program SWI-Prolog care înlocuiește secvențele de numere egale cu suma secvenței. Acest proces trebuie repetat până când nu mai sunt elemente consecutive egale în listă. De exemplu, pentru lista [1, 2 , 1, 1, 4, 5, 6, 7, 7, 7, 3, 3, 3, 3, 3, 3, 3, 10], rezultatul va fi [1, 8, 5, 6, 42, 10].

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 ⇒ [[6,4],[5,3]] (nu neapărat în această ordine)

D. 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 multipli de 3. **Se va folosi o funcție MAP.**

Exemplu **a)** dacă lista este (1 (2 A (3 A)) (6)) => (1 (2 A (A)) NIL)
b) dacă lista este (1 (2 (C))) => (1 (2 (C)))