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 predicat PROLOG **f(integer, integer),** având modelul de flux (i, o):

```
f(50, 1):-!.
f(I,Y):-J is I+1, f(J,S), S<1, !, K is I-2, Y is K.
f(I,Y):-J is I+1, f(J,Y).
```

Rescrieți această definiție pentru a evita apelul recursiv **f(J,V)** în ambele clauze, fără a redefini logica clauzelor. Justificați răspunsul.

B. 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)

C. Se consideră o listă neliniară. Să se scrie o funcţie LISP care să aibă ca rezultat lista iniţială in care atomii de pe nivelul **k** au fost inlocuiti cu **0** (nivelul superficial se consideră 1). **Se va folosi o funcție MAP.**

Exemplu pentru lista (a (1 (2 b)) (c (d)))

a) k=2 => (a (0 (2 b)) (0 (d))) **b)** k=1 => (0 (1 (2 b)) (c (d))) **c)** k=4 =>lista nu se modifică