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 L o listă numerică şi următoarea definiţie de predicat PROLOG **f(list, integer)**, având modelul de flux (i, o):

```
f([], 0).
f([H|T],S):-f(T,S1),S1<H,!,S is H.
f([_|T],S):-f(T,S1),S is S1.
```

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

B. Să se scrie un program PROLOG care generează lista submulţimilor de sumă pară, cu elementele unei liste. Se vor scrie modelele matematice și modelele de flux pentru predicatele folosite.

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

C. Un arbore n-ar se reprezintă în LISP astfel (nod subarbore1 subarbore2 .....)
Se cere să se înlocuiască nodurile de pe nivelurile impare din arbore cu o valoare **e** dată. Nivelul rădăcinii se consideră a fi 0. **Se va folosi o funcție MAP.**

Exemplu pentru arborele (a (b (g)) (c (d (e)) (f))) şi **e**=h => (a (h (g)) (h (d (h)) (h)))