

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

$f(0, -1):-!$.

$f(I,Y):-J \text{ is } I-1, \underline{f(J,Y)}, V>0, !, K \text{ is } J, Y \text{ is } K+V.$

$f(I,Y):-J \text{ is } I-1, \underline{f(J,Y)}, Y \text{ is } V+I.$

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

B. Să se scrie un program PROLOG care generează lista aranjamentelor de **k** elemente dintr-o listă de numere întregi, pentru care produsul elementelor e mai mic decât o valoare **V** dată. Se vor scrie modelele matematice și modelele de flux pentru predicatele folosite. **Exemplu** pentru lista [1, 2, 3], **k**=2 și **V**=7 \Rightarrow [[1,2],[2,1],[1,3],[3,1],[2,3],[3,2]] (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))) și **e=h** => (a (h (g)) (h (d (h)) (h)))