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 Fct(F L)
        (COND
                ((NULL L) NIL)
                ((FUNCALL F (CAR L)) (CONS (FUNCALL F (CAR L)) (Fct F (CDR L))))
                (T NIL)
        )
)
```

Rescrieți această definiție pentru a evita dublul apel recursiv **(FUNCALL F (CAR 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 returnează (într-o listă de perechi) toate posibilele partiții ale listei inițiale în două subliste, astfel încât toate elementele sublistelor să fie prime între ele (toate elementele primei subliste sunt prime între ele și toate elementele celei de-a doua subliste sunt prime între ele). Pentru a evita generarea aceleiași partiționări de două ori (ex: [A, B] și [B, A]), prima sublistă va conține cel mult același număr de elemente ca cea de-a doua sublistă. **De exemplu**, pentru lista [3, 5, 7, 9], rezultatul va fi (nu neapărat în această ordine): [[[5, 3], [9, 7]], [[7, 3], [9, 5]], [[3], [9, 7, 5]], [[9, 5], [7, 3]], [[9, 7], [5, 3]], [[9], [7, 5, 3]]].

C. Să se scrie un program PROLOG care generează lista aranjamentelor de **k** elemente dintr-o listă de numere întregi, având produs **P** dat. Se vor scrie modelele matematice și modelele de flux pentru predicatele folosite.

Exemplu- pentru lista [2, 5, 3, 4, 10], **k**=2 şi **P**=20 ⇒ [[2,10],[10,2],[5,4],[4,5]] (nu neapărat în această ordine)

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