## Programare logică și funcțională - examen scris -

## <u>Notă</u>

- 1. Subjectele 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 definiție de funcție LISP

(DEFUN Fct(F L)

(COND

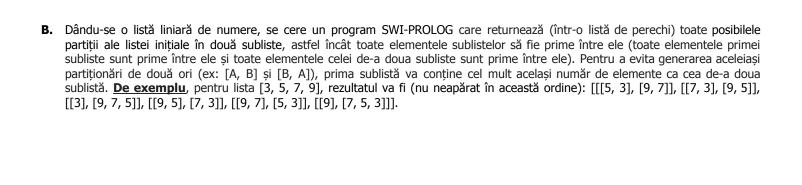
((NULL L) NIL)

((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.



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],  $\mathbf{k}=2$  și  $\mathbf{P}=20 \Rightarrow [[2,10],[10,2],[5,4],[4,5]]$  (nu neapărat în această ordine)

D. Un arbore n-ar se reprezintă în LISP astfel ( nod subarbore1 subarbore2 .....). Se cere să se determine înălţimea unui nod în arbore. Se va folosi o funcţie MAP.

 <u>Exemplu</u> pentru arborele (a (b (g)) (c (d (e)) (f)))
 a) nod=e => înălţimea e 0
 b) nod=v => înălţimea e -1
 c) nod=c => înălţimea e 2