

# 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 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)**), fără a redefini logica clauzelor și fără a folosi o funcție auxiliară. Nu folosiți SET, SETQ, SETF. Justificați răspunsul.

**B.** Scrieți un program PROLOG care determină dintr-o listă formată din numere întregi lista subșirurilor cu cel puțin 2 elemente, formate din elemente în ordine strict crescătoare. Se vor scrie modelele matematice și modelele de flux pentru predicatele folosite. ***Exemplu***- pentru lista [1, 8, 6, 4]  $\Rightarrow$  [[1,8],[1,6],[1,4],[6,8],[4,8],[4,6],[1,4,6],[1,4,8],[1,6,8],[4,6,8],[1,4,6,8]] (nu neapărat în această ordine)

**C.** Un arbore n-ar se reprezintă în LISP astfel (nod subarbore1 subarbore2 .....). Se cere să se verifice dacă un nod **x** apare pe un nivel par în arbore. Nivelul rădăcinii se consideră a fi 0. **Se va folosi o funcție MAP.**

**Exemplu** pentru arborele (a (b (g)) (c (d (e)) (f)))

**a)**  $x=g \Rightarrow T$       **b)**  $x=h \Rightarrow NIL$