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 F(N)
        (COND
                ((= N 0) 0)
                (> (F (- N 1)) 1) (- N 2))
                (T (+ (F (- N 1)) 1))
        )
)
```

Rescrieți această definiție pentru a evita dublul apel recursiv **(F (- N 1))**. Nu redefiniți funcția. Nu folosiți SET, SETQ, SETF. Justificați răspunsul.

B. Dându-se o listă liniară de numere pozitive, scrieți un program SWI-PROLOG care returnează (sub forma unei liste de perechi) toate posibilele partiții ale listei inițiale în două subliste, astfel încât suma cifrelor elementelor din cele două subliste este egală (se presupune că există cel puțin o astfel de partiționare a listei inițiale). **De exemplu**, pentru lista [28, 21, 52, 34, 7], rezultatul ar trebui să fie (nu neapărat în această ordine): [[[52, 28], [7, 34, 21]], [[34, 28], [7, 52, 21]], [[7, 28], [34, 52, 21]], [[34, 52, 21], [7, 28]], [[7, 52, 21], [34, 28]], [[7, 34, 21], [52, 28]]].

C. Să se scrie un program PROLOG care generează lista permutărilor mulţimii 1..N, cu proprietatea că valoarea absolută a diferenţei între 2 valori consecutive din permutare este >=2. Se vor scrie modelele matematice și modelele de flux pentru predicatele folosite.

Exemplu pentru N=4 ⇒ [[3,1,4,2], [2,4,1,3]] (nu neapărat în această ordine)

D. Se consideră o listă neliniară. Să se scrie o funcţie LISP care să aibă ca rezultat lista iniţială din care au fost eliminate toate apariţiile unui element **e**. **Se va folosi o funcție MAP.**

Exemplu **a)** dacă lista este (1 (2 A (3 A)) (A)) şi **e** este A => (1 (2 (3)) NIL)
b) dacă lista este (1 (2 (3))) şi **e** este A => (1 (2 (3)))