Functional and logic programming
- **written exam -**

Important:

1. Subjects are graded as follows: By default - 1p; A – 2p; B - 4p; C - 3p.
2. Prolog problems will be resolved using SWI Prolog. The following are required: (1) explanation of the code and of the reasoning behind it; (2) recursive model that solves the problem, for all the predicates used; (3) specification of every predicate (parameters and their meaning, flow model, type of the predicate - deterministic/non-deterministic).
3. Lisp problems will be resolved using Common Lisp. The following are required: (1) explanation of the code and of the reasoning behind it; (2) recursive model that solves the problem, for each function used; (3) specification of every function (parameters and their meaning).

A. The following function definition in LISP is given

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

Rewrite the definition in order to avoid the double recursive call **(F (- N 1))**. Do NOT redefine the function. Do NOT use SET, SETQ, SETF. Justify your answer.

B. Given a list composed of integer numbers, generate in PROLOG the list of arrangements of N elements ending with an odd value and have the sum S given. Write the mathematical models and flow models for the predicates used. For example, for the list L=[2,7,4,5,3], N=2 and S=7 ⇒ [[2,5], [4,3]] (not necessarily in this order).

C. Given a nonlinear list, write a Lisp function to return the list with all atoms on level **k** replaced by **0**. The superficial level is assumed 1. **A MAP function shall be used.**

Example for the list (a (1 (2 b)) (c (d)))

(a) k=2 => (a (0 (2 b)) (0 (d)))

(b) k=1 => (0 (1 (2 b)) (c (d)))

(c) k=4 => the list does not change