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(L1 L2)
        (APPEND (F (CAR L1) L2)
                (COND
                    ((NULL L1) (CDR L2))
                    (T (LIST (F (CAR L1) L2) (CAR L2)))
                )
                )
        )
```

Rewrite the definition in order to avoid the double recursive call **(F (CAR L1) L2)**. Do NOT redefine the function. Do NOT use SET, SETQ, SETF. Justify your answer.0

B. Write a PROLOG program that generates the list of all combinations of k elements with numbers from 1 to N, with the property that difference between two consecutive numbers from a combination has an even value. Write the mathematical models and flow models for the predicates used. For example, for the **N**=4, **k**=2 ⇒ [[1,3],[2,4]] (not necessarily in this order).

C. Given a nonlinear list, write a Lisp function to return the list with all occurrences of the element **e** replaced by the value **e1**. **A MAP function shall be used.**

Example **a)** if the list is (1 (2 A (3 A)) (A)), **e** is A and **e1** is B => (1 (2 B (3 B)) (B))

b) if the list is (1 (2 (3))) and **e** is A => (1 (2 (3)))