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

Important:

1. Subjects are graded as follows: of - 1p; A – 1.5p; B - 2.5p; C - 2.5p; D - 2.5p.
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(L)
        (COND
                ((NULL L) 0)
                ((> (CAR L) 0)
                        (COND
                                ((> (CAR L) (F (CDR L))) (CAR L))
                                (T (F (CDR L)))
                        )
                (T (F (CDR L)))
        )
)
```

Rewrite the definition in order to avoid the repeated recursive call **(F (CDR L))**. Do NOT redefine the function. Do NOT use SET, SETQ, SETF. Justify your answer.

B. Given a heterogeneous list made of numbers and nonempty numeric lists, write a SWI-PROLOG program that verifies if all numbers (including those in sublists) form an increasing sequence of numbers. For example, for the list [2,4,6, [10, 12, 19], 30, 201, [1000, 1003, 1006, 2003], 2020] the result will be true, but for the list [2,4,6, [10, 12, 11], 30, 201, [1000, 1003, 1006, 2003], 2020] the result will be false.

C. Write a PROLOG program that generates the list of all subsets of even sum, using the elements of a list. Write the mathematical models and flow models for the predicates used.
For example, for the list L=[2, 3, 4] ⇒ [[],[2],[4],[2,4]] (not necessarily in this order).

D. An n-ary tree is represented in Lisp as (node subtree1 subtree2 ...). Write a Lisp program to return the ***height*** of a node of a tree. **A MAP function shall be used.**
Example for the tree (a (b (g)) (c (d (e)) (f)))
a) nod=e => the height is 0 **b)** nod=v => the height is -1 c**)** nod=c => the height is 2.