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. Let L be a list of numbers and given the following PROLOG predicate definition with flow model (i, o):

```
f([],0).
f([H|T],S):-f(T,S1),S1>=2,!,S is S1+H.
f([_|T],S):-f(T,S1),S is S1+1.
```

Rewrite the definition in order to avoid the recursive call **f(T,S)** in both clauses. Do NOT redefine the predicate. Justify your answer.

B. Write a PROLOG program that generates the list of all arrangements of k elements with the value of sum of all elements from each arrangement equal with a given S, from a list of integers. Write the mathematical models and flow models for the predicates used. For example, for the list [6, 5, 3, 4], **k**=2 and **S**=9⇒ [[6,3],[3,6],[5,4],[4,5]] (not necessarily in this order).

C. An n-ary tree is represented in Lisp as (node subtree1 subtree2 ...). Write a Lisp function to determine the number of nodes on level **k**. The root level is assumed zero. **A MAP function shall be used.** ***Example*** for the tree (a (b (g)) (c (d (e)) (f)))
a) k=2 => nr=3 (g d f) **b)** k=4 => nr=0 ()