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. 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. Given a nonlinear list containing both numerical and non-numerical atoms, write a Lisp program that builds a list with the elements from the initial list, from positions k to k (counting from left to right, considering all elements regardless of level), in reverse order. For example, for the list (A B 12 (5 D (A F (10 B) D (5 F) 1)) C 9) and k = 3 the result is (9 F B A 12).

C. Write a PROLOG program that generates the list of all subsets of k elements (all elements being odd numbers) in arithmetic progression. Write the mathematical models and flow models for the predicates used. For example, for L=[1,5,2,9,3] and k=3 ⇒ [[1,5,9],[1,3,5]] (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.