

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. Given the following PROLOG predicate definition **f(integer, integer)**, with the flow model (i, o):

f(0, 0):-!.
f(I,Y):-J is I-1, **f(J,V)**, V>1, !, K is I-2, Y is K.

f(I,Y):-J is I-1, **f(J,V)**, Y is V+1.

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

B. 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 $k=3 \Rightarrow [[1,5,9],[1,3,5]]$ (not necessarily in this order).

C. Given a nonlinear list, write a Lisp function to return the list with all even numerical atoms from an odd level removed. The superficial level is assumed 1. **A MAP function shall be used.**

Example **a)** if the list is (1 (2 A (4 A)) (6)) => (1 (2 A (A)) (6))

b) if the list is (1 (2 (C))) => (1 (2 (C)))