ARTIFICIAL INTELLIGENCE

SUPINFO International University

Sign of Success

4AIT – TP

Logic Programming

A controlled world

Programming
State Space
Heuristic
Solver

Version 1.0
Last update: 05/29/2017
Use: Students/Staff
Author: Cyril Alexandre Pachon

Table of contents

1 PREAMBLE: GENERAL INSTRUCTIONS

Your response folder should conform to the following:

1. The folder should be named: **[4AIT]-IDopenCampus-SURNAME-FirstName**.
2. In folder, files in **.pdf** and **.pl** are acceptable.
3. Use an archive tool to zip your folder : **[4AIT]-IDopenCampus-SURNAME-FirstName.zip** (e.g. **[4AIT]-123456-Lutin-Marc.zip**)
4. Your archive must be placed in website: **sce.sad.supinfo.com**.

For this examination you can use the support slides (.ppt, LABs), pen and paper (these are not provided by the school) for your draft response to the questions. You can use swi-prolog tool.

You are **NOT ALLOWED** to use **Internet** for this examination.

If you are found cheating by your supervisor your examination session will be **CANCELLED**. Your transcript will be reviewed by the disciplinary committee.

2 RESOLUTION (4 POINTS)

Question 1.1 (2 points): Let the set: {a ∨ b, a ∨ ¬c, ¬b ∨ c, ¬a}
Prove that the clauses are satisfiable or unsatisfiable.

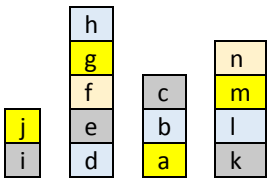
Question 1.2 (2 points): You have the following proposals:

- $(a \wedge b) \Rightarrow c$
- $d \Rightarrow a$
- $d \Rightarrow b \nvDash d \Rightarrow c$

The system it is valid? Answer to this question using the Clausale form.

3 PROLOG PROGRAM (6 POINTS)

You have 4 stacks of cubes:



Question 2.1 (1 point): For these 4 stacks, write the basic facts, with 3 Prolog predicates named, **SommetPile** with 1 arity, **BasPile** with 1 arity, **Sur** with 2 arities.

Question 2.2 (1.5 points): Write a Prolog predicate named **CompteCubeDessus** with 2 arities, returning the number of cubes deposed on a cube. For example **compteCubeDessus**(a,R) → R = 2, **compteCubeDessus**(m,R) → R = 1.

The predicates **SommetPile** and **BasPile** are now excluded from the facts base.

Question 2.3 (1 point): Write 2 new predicates named **NouveausommetPile** with 1 arity and **NouveaubasPile** with 1 arity depending only on the predicate **Sur** with 2 arities, of the facts base.

Question 2.4 (1.5 points): Write the Prolog predicates named **Sommet** with 0 arity and **Bas** with 0 arity, returning the list of all cubes of stack down and stack top, using the predicates of the **question 2.3**.

Question 2.5 (1 point): Write a Prolog predicate named **CompteCubeDessous** with 2 arities returning the cube number under a cube given using predicates of the **question 2.3**. For example **compteCubeDessous**(a,R) → R = 0, **compteCubeDessous**(m,R) → R = 2.

4 PROLOG PROBLEM (10 POINTS)

A drone can fly if its launcher gives it a start code. The code is given by a user and an automaton. The code is constrained. You must create a Prolog program to provide valid 5-digit codes. The codes are in the form of a sequence of 5 values.

1	2	3	4	5

Figure1: Example of code.

The characteristics imposed on the sequence:

1. 5 Digits.
2. The first 3 digits have a value between 1 and 5, or [1...5].
3. The 4th digit has a value between 1 and 6 or [1...6].
4. The 5th digit has a value between 1 and 7 or [1...7].
5. The code's digits sum is 16.

2	3	1	4	6

Figure2: Example of a code valid.

Question 2.1 (3 points): Write a Prolog predicate named **Vérifier** with 1 arity asking user to enter the two first digits between 1 and 5 and produces the 3 other digits respecting the constraints given.

Example:
1 ?- verifier(L).
The first digit
|: 2.
The second digit
|: 3.
L = [2, 3, 1, 4, 6]

Your program evolves:

1. The digits sum is always 16.
2. The two first digits given by the user are necessarily different (values are always between 1 and 5).
3. The 3 digits given by your program are different between them and also different from the user.
4. The 3 digits given by your program always follow the initial constraints:
    a. The first choice of the digit is between the values 1 and 5,
    b. The second choice of the digit is between the values 1 and 6,
    c. The third choice of the digit is between the values 1 and 7.

Question 2.2 (3 points): Write a Prolog predicate named **Combinaisons** with 1 arity, returning all combinations by one request. If the user values are not correct, your program re-asks to enter the digits.

Question 2.3 (1 point): With your Prolog predicate **Combinaisons** with 1 arity, give all combinations for the user digit couples: (1, 2), (2, 3), (3, 4) and (4, 5).

Your program evolves, we keep all constraints except now your program is not limited only to the value 16. The sum value is given by an argument.

Question 2.4 (3 points): Write a Prolog predicate named **Combinaisons2** with 2 arities, returning all combinations by one request and re-asks to the user to enter the digits, if there are not different between them and whish allow to give a sum value.

Example:
1 ?- combinaisons2(L, 23).
The first digit
|: 4.
The second digit
|: 5.
[4, 5, 1, 6, 7]