ESSI 2ème Année **Année 2002-2003**

Examen de logique et Prolog

A) LOGIQUE

1 Formalisation (3 points)

Soit le langage du premier ordre formé de l'ensemble de variables $V = \{x,y\}$, des symboles fonctionnels 0-aire H et L, des symboles de prédicat 1-aire *homme*,*vtt*, *yeuxVerts* et des symboles de prédicats 2-aire *égal* et *estMariéAvec*.
Dans ce langage, exprimer les énoncés suivants:

1. Les hommes qui font du VTT ont les yeux verts.
2. Les hommes aux yeux verts sont mariés.
3. Hélène ne peut épouser qu'un homme qui fait du VTT.
4. Il existe un unique homme aux yeux verts qui est marié avec Hélène.

2 Validité (2 points)

1. Soit la formule $\Phi = \exists\ x\ \forall\ y\ p(x,y,y)$. Donner une interprétation $I1$ qui est un modèle de Φ et une interprétation $I2$ qui n'est pas un modèle de Φ.
2. Montrer par 0-résolution que $A \Rightarrow (B \Rightarrow A)$ est universellement valide.

3 Preuve dans tous les modèles (5 points)

Soit les axiomes :

- A1 : $\forall x_1\,\forall y_1\,\forall z_1\,\forall x_2\,\forall y_2\,\forall z_2\,(p(x_1,y_1,x_2)\ \wedge\ p(y_1,z_1,y_2)) \Rightarrow (p(x_1,y_2,z_2) \Leftrightarrow p(x_2,z_1,z_2))$
- A2: $\exists x\,[\forall y\ \ p(x,y,y)\ \ \wedge\ \ \forall z\,\exists u\ \ p(u,z,x)]$

Montrer que :

$$\Phi = \exists x[\forall y\, p(x,y,y)\ \wedge\ \forall z\,\exists u\, p(z,u,x)]$$

est valide dans tous les modèles de $\{$A1, A2$\}$. Prenez soin de détailler les étapes de mise sous forme de clauses ainsi que de noter les substitutions lors de la résolution.

B) PROLOG

4 La classique "famille" (3 points)

Soit le programme :

```
descendant(Descendant,Ascendant) :-
  parent(Ascendant,Descendant).
descendant(Descendant,Ascendant) :-
  descendant(Descendant,Enfant),
  parent(Ascendant,Enfant).
```

```
parent(mathilde,francois).  parent(pierre,francois).    parent(pierre,jean).
parent(mathilde,rose).      parent(jean,bastien).       parent(jean,rudolph).
parent(rose,bastien).       parent(rose,rudolph).       parent(rudolph,gwendoline.
```

1. Donner l'ensemble de solutions de la requête `descendant(D,A)`
2. Dessiner l'arbre de recherche pour la requête `descendant(D,A)`

5 La classique "send+more" (2 points)

Donner la taille de l'espace de recherche pour la requête t `mm2([S,E,N,D,M,O,R,Y])`

```
mm2([S,E,N,D,M,O,R,Y]) :-
     generate([S,E,N,D,M,O,R,Y],[0,1,2,3,4,5,6,7,8,9]),
     S>0, M >0,
     test_sum(S,E,N,D,M,O,R,Y).

generate([],_).
generate([X|Xs],Vs):-
     select(X,Vs,Vr), generate(Xs,Vr).
select(J,[J|L],L).
select(J,[K|L],[K|M]):- select(J,L,M).
test_sum(S, E, N, D, M, O, R, Y) :-
     V1 is            1000*S + 100*E + 10*N + D
          +           1000*M + 100*O + 10*R + E,
     V2 is 10000*M + 1000*O + 100*N + 10*E + Y,  V1 = V2.
```

6 Un programme de coloriage (3 points)

Ecrire un prédicat `color(Lr,Lc)` qui permet de colorier une carte planaire avec 4 couleurs tel que les couleurs de deux régions adjacentes soient différentes. `Lr` est la liste des régions à colorier et `Lc` la liste des couleurs associées. Les relations d'adjacence sont définies à l'aide du prédicat `adj(Région1,Région2)`.
Exemple:
Soient

```
adj(france,allemagne).    adj(france,italie).  adj(france,suisse).    adj(allemagne,suisse).
adj(allemagne,autriche).  adj(suisse,italie).  adj(suisse,autriche).  adj(autriche,italie).
```

alors

```
?- color([france,allemagne,autriche,suisse,italie],Lc).
Lc=[rouge,vert,rouge,jaune,vert] ; Lc=[rouge,vert,rouge,jaune,bleu] ; ...
```

7 Un petit parser (3 points)

Ecrire la DCG qui permet de parser des expressions booléennes. Une expression booléenne est constituée de variables (notées x_1 à x_{99}) et de connectives (notées `and`, `or`, `not`) avec les arités usuelles. Il n'y pas de parenthèses et vous pouvez définir les règles de priorités que vous souhaitez.