

Ce sujet comporte 2 pages.

Tous documents autorisés.

Le barème est indicatif.

Toutes les réponses doivent être justifiées (de façon concise).

Partie I - arbres de recherche (4 pts)

Soit Prog le programme Prolog ci-dessous :

$p(X) :- a(X), b(X) .$	$d(0,1) .$
$p(X) :- c(Y), !, d(Y,X) .$	$d(0,2) .$
$p(X) :- e(X) .$	$d(0,3) .$
$a(7) .$	$d(1,4) .$
$a(0) .$	$d(1,5) .$
$a(18) .$	$d(2,0) .$
$b(X) :- X < 4, ! .$	$e(X) :- a(Y), X \text{ is } Y-1 .$
$b(18) .$	
$c(0) .$	$c(1) .$

Question 1

Construire l'arbre de recherche du but $p(X)$.

Quels sont les témoins de la preuve de $\text{Prog} \vdash p(X)$?

Soit toto et totobis les prédicats suivants :

$\text{toto}(X) :- \text{lulu}(Y), X \text{ is } Y-1 .$
 $\text{totobis}(X) :- X \text{ is } Y-1, \text{lulu}(Y) .$

Question 2

Faites une étude comparative des prédicats toto et totobis (Sont-ils équivalents? Réflexion par rapport aux modes).

Partie II - Logique (6 pts)

Question 3

Construire une preuve en calcul des séquents et une preuve en déduction naturelle pour chacun des énoncés suivants :

$\neg(A \vee B) \Rightarrow \neg A \wedge \neg B$ et $((A \Rightarrow B) \Rightarrow A) \Rightarrow A$

Question 4

Montrer que l'énoncé suivant n'a pas de preuve en calcul des séquents.

$\exists x.A(x) \Rightarrow \forall x.A(x)$

Partie III - Programmation logique (10 pts)

Question 5 (3 pts)

Considérer le prédicat suivant :

```
filtrer_niveaux_profondeur_aux( [], _Precedents, [] ) .  
filtrer_niveaux_profondeur_aux( [Niveau|ListeNiveaux] , Precedents, [Filtrat|ListeFiltrats] ) :-  
    filtre_liste_liste( Precedents, Niveau, Filtrat ) ,  
    conc( Precedents, Filtrat, NouvPrecedents ) ,  
    filtrer_niveaux_profondeur_aux( ListeNiveaux, NouvPrecedents, ListeFiltrats ) .
```

Pour quelle raison est-il inefficace pour de grandes données ?

Par quelle méthode y remédier ? Proposer une variante de ce prédicat qui utilise cette méthode.

Faut-il modifier d'autres prédicats que `filtrer_niveaux_profondeur_aux` ?

Question 6 (4 pts)

Considérer un graphe orienté et colorié dont les sommets appartiennent à l'ensemble V , les couleurs appartiennent à l'ensemble $\{\text{rouge, noir}\}$ et dont les arcs appartiennent à l'ensemble de paires $V \times V$.

On décide de représenter les arcs du graphe par le prédicat `arc` : `arc(a,b)` représentant l'arc (a,b) . On décide aussi de représenter les sommets du graphe par le prédicat `sommet` : `sommet(a,rouge)` indiquant que le sommet a est de couleur rouge.

On appelle *chemin alterné* un chemin du graphe qui passe alternativement par des sommets de couleur noir et rouge.

On représentera un chemin par la liste de ses sommets.

Écrire en Prolog le prédicat `chemin` de mode `chemin(+A, ?B, -Chemin)` tel que le but `chemin(A, B, Chemin)` réussit si et seulement si `Chemin` est un chemin alterné qui part de A et arrive en B .

Question 7 (3 pts)

Comment se comporte votre prédicat `chemin` en cas de cycles dans le graphe ?

Donner successivement 3 variantes de votre prédicat `chemin` pour ne rechercher d'abord que des chemins alternés de longueur données, puis tous les chemins alternés par ordre de longueurs croissantes, puis tous les chemins alternés qui ne passent jamais plus d'une fois par un sommet.