UNIVERSITÉ D'ORLÉANS — L3 Informatique
Programmation logique — Année 2018-2019

Contrôle terminal

Documents autorisés : 1 feuille A4. Durée : 2h.
Chaque question peut être traitée indépendamment en supposant faites les questions précédentes.

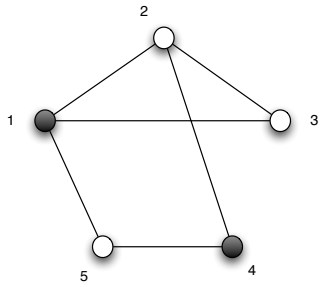
Exercice 1 (4 points).

```
elfe(dobby).    elfe(kreature).
sorciere(hermione).    sorciere(mcGonagall).    sorciere(rita_skeeter).
possede(hermione, baguette_bois_vigne).
possede(mcGonagall, baguette_bois_sapin).
possede(rita_skeeter, plume_a_papote).
magique(X) :- elfe(X).
magique(X) :- sorciere(X).
predicat1(X,Y) :- magique(X), !, possede(X,Y).
predicat2(X) :- \+ sorciere(X), elfe(X).
predicat3(X) :-  sorciere(X), \+ elfe(X).
```

Pour chacune des requêtes suivantes, donnez l'arbre de recherche et les réponses de l'interpréteur Prolog.

```
?- predicat1(X,Y).
?- predicat2(X).
?- predicat3(Hermione).
```

Exercice 2 (6 points). La percolation modélise certains phénomènes de dissémination de l'information dans un réseau. Nous disposons d'un graphe G non orienté dont chaque sommet a une couleur noire ou blanche. Le processus de percolation consiste à répéter, tant qu'il existe un sommet u blanc qui a strictement plus de voisins noirs que de voisins blancs, alors le sommet u devient noir. Par exemple sur le graphe suivant la percolation va changer en noir les sommets 2, 3 et 5.



Les sommets sont numérotés de 1 à n. Le graphe est codé comme suit, avec le prédicat `couleur` dynamique, pour pouvoir changer la couleur des sommets.

```
:- dynamic(couleur/2).
sommet(1). sommet(2). sommet(3). sommet(4). sommet(5).
arete(1,2).  arete(1,3).  arete(1,5).  arete(2,3).  arete(2,4).  arete(4,5).
couleur(1,noir).  couleur(2,blanc).  couleur(3,blanc).  couleur(4,noir).  couleur(5,blanc).
```

1. Un sommet X est voisin à un sommet Y s'il existe une arête entre X et Y ou entre Y et X. Ecrire un prédicat `voisin(+X,+Y)` qui réussit si X est voisin à Y.
2. Écrire un prédicat `compter(+X,+C,?N)` qui calcule le nombre N de voisins de X ayant la couleur C. Indication : utiliser un prédicat du second ordre.

   ```
   ?- compter(2,noir,N).
   N = 2
   ```

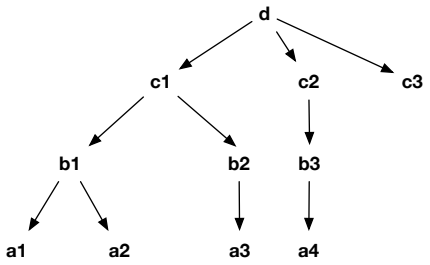
3. Écrire un prédicat `trouver(-X)`, qui trouve un sommet X qui a plus de voisins noirs que de voisins blancs.

```
?- trouver(X).
X = 2 ? ;
X = 5
```

4. Ecrire un prédicat `percolation`, qui réalise le processus de percolation. Pour changer la couleur d'un sommet X de C_1 en C_2, utilisez `retract(couleur(X,C1))` et `asserta(couleur(X,C2))`.

```
:- percolation.
:- listing(couleur).
couleur(5, noir).
couleur(3, noir).
couleur(2, noir).
couleur(1, noir).
couleur(4, noir).
```

Exercice 3 (10 points). Considérons la hiérarchie suivante dans une entreprise :



Ici d est le grand patron, les a_i, b_j, c_l sont des employés, d est supérieur hiérarchique direct de c_1, c_2 et c_3, c_1 est supérieur hiérarchique direct de b_1 et b_2, etc. Pour coder cette hiérarchie, on utilisera les prédicats suivants :

— `employé(X)` : X est un(e) employé(e) de l'entreprise.

— `grandpatron(X)` : X est le (la) grand(e) patron(ne) unique de l'entreprise.

— `superieurdirect(X,Y)` : X est un supérieur hiérarchique direct de Y.

On suppose en outre que, dans cette entreprise, chaque employé a un "rang hiérarchique" unique : le grand patron est de rang 1, et si X est supérieur hiérarchique direct de Y, cela implique que le rang de Y vaut le rang de X plus 1.

1. Donner les faits utilisant `employé`, `grandpatron` et `superieurdirect` pour coder cette hiérarchie.
2. Ecrire un prédicat `rang(+X,?N)` qui est vrai si N est le rang de l'employé X.
3. Ecrire un prédicat `unsuperieur(+S,+E)` qui est vrai si S est un supérieur (pas forcément direct) de E.
4. Ecrire un prédicat `unehierarchie(+L)` qui est vrai si la liste L est une suite d'employés en ordre de hiérarchie directe. Par exemple `unehierarchie([d,c1,b1,a2])` succès mais `unehierarchie([d,c1,b1,a3])` échoue.
5. Ecrire un prédicat `superieurs(+E,?L)` qui est vrai si la liste L est la liste de tous les supérieurs, directs ou indirects, de E. Par exemple `superieurs(a1,L)` réussit avec `L = [b1,c1,d]` et `superieurs(d,[])` réussit.
6. Ecrire un prédicat `meilleurrang(+L,?R)` qui est vrai si R est le meilleur rang hiérarchique des employés de la liste L. Par exemple `meilleurrang([a1,b2,c3],R)` réussit avec `R=2`.
7. Ecrire un prédicat `personnerang(+L,+R,?P)` qui est vrai si P est une personne du rang R dans la liste de personnes L. Par exemple `personnerang([a1,b1,c1,c2,d],2,X)` réussit avec `X=c1` puis `X=c2`.