TD3 – Prolog

Chloé Conrad, Nathalie Guin, Marie Lefevre & Maëva Somny

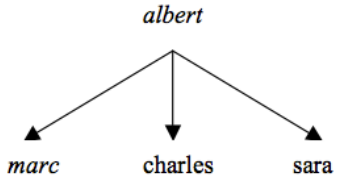
PARTIE 1 – TRADUCTION D 'ENONCES

Question 1 : Traduire en Prolog l'énoncé suivant :

Marie aime le vin
Pierre est un voleur
Pierre aime tous ceux qui aiment le vin
Si quelqu'un est un voleur et aime quelque chose alors il le vole
Qui vole quoi ?

Question 2 : Soit le programme Prolog suivant :

```
homme(albert).
homme(marc).
homme(charles).
femme(sara).
pere(albert, marc).
pere(albert, charles).
pere(albert, sara).
```



qui décrit l'arbre généalogique suivant :

À partir de ces assertions (faits), définir les prédicats généraux suivants :

- `enfant(X, Y)` qui exprime que X est un enfant de Y ;
- `fils(X,Y)` qui exprime que X est un fils de Y ;
- `fille(X, Y)` qui exprime que X est une fille de Y ;
- `frere-ou-sœur(X, Y)` qui exprime que X est frère ou sœur de Y. Il est à noter qu'un individu n'est pas son propre frère ou sa propre sœur.

PARTIE 2 – RECURSIVITE

Question 3 : Définir deux prédicats : l'un affiche les nombres de 1 à N par ordre croissant, l'autre par ordre décroissant.

Question 4 : Définir un prédicat qui calcule récursivement la somme des N premiers entiers.

PARTIE 3 – MANIPULATION DE LISTES

Question 5 : Écrire un prédicat `compresse` permettant de supprimer des doublons consécutifs dans une liste L pour obtenir une liste L1. L'ordre des éléments doit être respecté.

```
Exemple:
?- compresse([a,a,a,a,b,c,c,a,a,d,e,e,e,e],L1).
L1 = [a,b,c,a,d,e]
```

Question 6 : Définir le prédicat `sous-ensemble(L1,L2)` qui est vrai si tous les éléments de la liste L1 font partie de la liste L2.

Question 7 : Écrire un prédicat `inverse` permettant de renverser la liste L.

Question 8 : Définissez le prédicat `sous-liste(L1,L2)` qui est vrai si la liste L1 est une sous-liste de la liste L2.

BONUS…

Question 9 : Définir le prédicat `longueur(L,N)`, qui étant donnée la liste L, calcule sa longueur N.

Question 10 : Définir le prédicat `concat(L1,L2,L3)` où L3 est le résultat de la concaténation de L1 et L2 (sans utiliser append).

Question 11 : Définir le prédicat `palindrome(L)` qui retourne vrai si la liste L est sa propre image renversée.

Question 12 : Définir un prédicat `rang_pair(X,Y)` qui extrait les éléments de la liste X qui ont des indices de rang pair afin de construire la liste Y.

Ex. `rang_pair([a,b,c,d,e],L). -> L=[b,d]`

Question 13 : Définir le prédicat `indice(X,L,N)`, qui étant donnés un élément X et une liste L, X appartenant à L, calcule N l'indice de la première occurrence de X dans L. Peut-on utiliser ce prédicat pour formuler une requête permettant de calculer le ième élément d'une liste ?

Question 14 : Écrire le prédicat `remplace(X1,X2,L1,L2)` qui construit la liste L2 qui est la liste L1 dans laquelle X1 est remplacé par X2.

Question 15 : Définir le prédicat `partage(L,X,L1,L2)`, qui étant donnés une liste de nombre L et un nombre X, calcule la liste L1 des nombres de L inférieurs à X, et la liste L2 des nombres de L supérieurs ou égaux à X.

Question 16 : Définir le prédicat `somme(L,R)`, qui étant donnée L une liste de nombres Xi, calcule la somme des (i*Xi).