Expression Logique et Fonctionnelle . . . Évidemment

TP4 : Les listes

1 Listes en Prolog

On peut spécifier des listes en Prolog en mettant les éléments entre crochets, et séparant les éléments par des virgules. Les éléments d'une liste sont de types quelconques. Par exemple :

- [] est la liste vide,
- [mia, vincent, jules] est une liste de *longueur* 3,
- [mia, voleur(lapin), X, 2, [vincent ,3]] est une liste de longueur 5.

L'opérateur | permet d'ajouter un (ou plusieurs) éléments en tête de liste. Par exemple, [vincent | [mia]] est la liste [vincent, mia].

Question 1 Testez (et comprenez !) les requêtes d'unification suivantes :

- ?- [H|T] = [mia,vincent,jules]
- ?- [H|T] = []
- ?- [X,Y|T] = [mia,vincent,jules]
- ?- [a,b,c,d] = [a,[b,c,d]]
- ?- [a,b,c,d] = [a|[b,c,d]]
- ?- [a,b,c,d] = [a,b,[c,d]]
- ?- [a,b,c,d] = [a,b|[c,d]]
- ?- [a,b,c,d] = [a,b,c,d |[]]
- ?- [] = [_]
- ?- [] = _

Question 2 Définissez le prédicat longMinDeux/1 qui est vrai pour toute liste de longueur *au moins* 2.

2 Listes et récursion

Question 3 Programmez le prédicat récursif memeLong/2 permettant de tester si deux listes sont de même longueur. Pour le cas de base, partez de la liste vide. Pour la règle récursive, supposez dans le corps de la règle que le prédicat est satisfait pour deux listes.

Question 4 Définissez le prédicat member/2, qui est satisfait si le premier argument est un élément de la liste donnée en deuxième argument. Pour le cas de base, supposez que l'élément cherche se trouve en tête de liste.

Question 5 Écrivez un prédicat deuxFois/2 dont le premier argument est une liste, et dont le deuxième argument est une liste contenant chaque élément de la première liste écrit deux fois. Par exemple,

```
?- deuxfois([a,4,bidule],X).
X=[a,a,4,4,bidule,bidule].
?- deuxfois([1,2,3,4],X).
X=[1,1,2,2,3,3,4,4].
```

3 Combinaisons

Question 6 Écrivez le prédicat ternaire combine1/3 qui combine les éléments de deux listes dans une troisième :

```
?-combine1([a,b,c],[1,2,3],X).
X = [a,1,b,2,c,3]
?-combine1([foo,bar,yip,yup],[glub,glab,glib,glob],X).
X = [foo,glub,bar,glab,yip,glib,yup,glob]
```

Question 7 Écrivez un prédicat ternaire combine2/3 qui combine maintenant deux listes dans une troisième de la façon suivante :

```
?-combine2([a,b,c],[1,2,3],X).
X = [[a,1],[b,2],[c,3]]
?-combine2([foo,bar,yip,yup],[glub,glab,glib,glob],X).
X = [[foo,glub],[bar,glab],[yip,glib],[yup,glob]]
```

Question 8 Écrivez enfin un prédicat ternaire combine3/3 qui combine maintenant deux listes dans une troisième de la façon suivante :

```
 ?-combine3([a,b,c],[1,2,3],X).
X = [paire(a,1),paire(b,2),paire(c,3)]
?-combine3([foo,bar,yip,yup],[glub,glab,glib,glob],X).
X = [paire(foo,glub),paire(bar,glab),paire(yip,glib),paire(yup,glob)]
```

4 Concaténation de listes

Vous allez travailler avec le prédicat suivant, permettant d'accoler deux listes :

```
accoler([],L,L).
accoler([T|Q],L2,[T|L3]) :- accoler(Q,L2,L3).
```

Question 9 Testez ce prédicat.
Question 10 Comment peut-on utiliser ce prédicat pour diviser une liste en deux listes consécutives ?
Question 11 Expliquez le prédicat avec vos propres mots, en français.
Question 12 Dessiner l'arbre de résolution de la requête accoler([je,te,dis],[c,est,facile],X) .
Question 13 Examiner la trace de la requête precedente. Expliquer le processus de construction du résultat, notamment l'instanciation des variables.

5 Ensembles

Question 14 Écrivez un prédicat inclusion/2 satisfait lorsque la liste donnée en premier argument est un sous-ensemble de la liste donnée en deuxième argument.
Question 15 Écrivez un prédicat notmember/2 qui est satisfait lorsque l'élément donne en premier argument n'est pas contenu dans la liste la liste donnée en deuxième argument.
Question 16 Écrivez un prédicat union/3 satisfait lorsque la liste donnée en premier argument est l'union des listes données en deuxième et troisième arguments.
Question 17 Écrivez un prédicat intersection/3 satisfait lorsque la liste donnée en premier argument est l'intersection des listes données en deuxième et troisième arguments.