3 Listes

Les tableaux en Prolog sont des listes. Contrairement à la plupart des langages de programmation, les indices des éléments ne sont pas disponibles. En revanche, une liste L peut toujours être décomposée en L=[E|R] où E est le premier élément de la liste et où R est le reste de la liste L (R est une liste : c'est en fait la tranche de L qui démarre après E). Les éléments d'une liste peuvent être séparés par des virgule : [1,2,3]. La liste vide est [ ].
La liste [1,2,3] peut s'ecrire également [1|[2,3]] ou [1,2,3|[ ]] ou encore [1,2|[3|[ ]]] .

Exercice 1
Pour chacun des cas suivants, donner la réponse de l'interpréteur Prolog, puis vérifier.

1. `[a,[a]]=[H|T].`
2. `[[a,b],c]=[[H|T1]|T2].`
3. `[a,b,[c]]=[H1|[H2|[H3|T]]].`
4. `[a,b,[c,d]]=[H1,H2|[H3|T]].`
5. `p([X|Y], [Y|Z], V) = p([a,b,c], W, [X,W]).`
6. `p(X,[a,b]) = p([X1|P1],[X1|L1]).`
7. `s(X,[a,b]) = s(L,L).`
8. `e(1,R,[2,3]) = e(X,[Y|Q],[Y|L]).`
9. `c([X|A],B,[X|C]) = c([a,b],[c],L).`

Exercice 2
Ecrire les prédicats suivants.

1. affiche(+L) est vrai si tous les éléments de la liste L sont affichés à l'écran.
2. affiche2(+L) est vrai si tous les éléments de la liste L sont écrits en ordre inverse.
3. premier1(+L, ?X) est vrai si X est le premier élément de L.
4. premier2(+L) est vrai si le premier élément de la liste L est affiché (et aucun autre).
5. dernier1(+L, ?X) est vrai si X est le dernier élément de L. Donner l'arbre de dérivation pour la requête `dernier1([a,b,c],X)`.
6. dernier2(+L) est vrai si le dernier élément de la liste L est affiché (et aucun autre). Donner l'arbre de dérivation pour la requête `dernier2([a,b,c])`.
7. element(?X,+L) est vrai si X est élément de la liste L. Donner l'arbre de dérivation pour chacune des requêtes `element(2,[1,2,3])` et `element(X,[1,2,3])`.
8. compte(+L, ?N) est vrai si N est le nombre d'éléments dans la liste L. Donner l'arbre de dérivation pour la requête `compte([a,b,c],X)`.
9. somme(+L, ?N) est vrai si N est la somme des éléments de la liste d'entiers L. Donner l'arbre de dérivation pour la requête `somme([3,4,6],X)`.
   Remarque : Attention, un prédicat somme(...) a déjà été défini dans le TD 2. Si vous rangez toutes vos définitions dans le même fichier cela va entraîner des conflits. Dans ce cas il faut en renommer un des deux.
10. nieme1(+N,+L, ?X) est vrai si X est le N-ème élément de la liste L.
11. nieme2(?N,+L,+X) est vrai si X est le N-ème élément de la liste L. (Attention à la spécification du prédicat.)

12. occurrence(+L,+X, ?N) est vrai si N est le nombre de fois où X est présent dans la liste L.
13. sous-ensemble(+L1,+L2) est vrai si tous les éléments de la liste L1 font partie de la liste L2.
14. substitue(+X,+Y,+L1,-L2) est vrai si L2 est le résultat du remplacement de X par Y dans L1.
15. ajoute_fin(+X,+L1,-L2) est vrai si L2 est le résultat de l'ajout de X à la fin de la liste L1.
16. retourne(+L,-L1) est vrai si la liste L1 est la liste L dans l'ordre inverse.
17. selection(+L,+LI,-R) est vrai si R est la liste composée des éléments de L ayant l'indice dans LI. Par exemple selection([a,b,c,d,e],[1,3,5],R) réussit avec R=[a,c,e].
18. aplatir(+LL, -L) est vrai si L est la liste LL aplatie. La liste LL est composée de constantes ou de nombres. Par exemple aplatir([a,[[2],[3]]],L) réussit avec L=[a,2,3].

    Vous pouvez utiliser les prédicats prédéfinis `atomic(T)` qui réussit si T est une constante ou un nombre, `list(T)` qui réussit si T est une liste.

Exercice 3

1. Rappel : le terme `[P,D|R]` désigne une liste dont le premier élément est P, le deuxième est D et le reste est la liste R.

   Ecrire un prédicat triee(+L) qui réussit si L est une liste triée dans l'ordre croissant, échoue sinon. Par exemple triee([4,6,8]) réussit et triee([4,2,6,8]) échoue.
2. Ecrire un prédicat insert(+X,+L,-R) pour insérer un entier X dans une liste L, que l'on suppose triée, de telle façon que la liste résultat R soit triée. Par exemple insert(7,[3,6,10],R) réussit avec R=[3,6,7,10].
3. Ecrire un prédicat tri_insert(+L,-R) pour trier une liste d'entiers L en une liste triée R, par un tri par insertion :
   — La liste vide est triée.
   — Pour trier la liste [N—L] on commence par trier L, puis on insère N dans le résultat en respectant l'ordre croissant des éléments.