

TD de *Programmation logique et par contraintes* n° 2

Listes et I/O de base.

Exercice 1 Si x_1, \dots, x_n sont des termes, $[x_1, \dots, x_n]$ désigne la liste de ces termes, et si x est un terme et l une liste, $[x|l]$ désigne la liste qui a x comme premier élément (“head”) et l comme reste (“tail”) (par exemple $[1|[2|[]]] = [1,2]$, $[]$ étant la liste vide)

Donner le résultat de chacune des requêtes suivantes (travailler sur papier, puis lancer la requête pour vérifier).

1. ?- $[a, [a]] = [H|T]$.
2. ?- $[[a,b], c] = [[H|T1]|T2]$.
3. ?- $[a,b, [c]] = [H1|[H2|[H3|T]]]$.

- Définir le predicat `concat(X,Y,Z)`, vrai si Z est la concatenation de X et Y .
- Définir le prédicat `dernier(X,L)` qui réussit si X est le dernier élément de L . Donner deux versions de ce prédicat, avec et sans utilisation de `concat`.
- Définir deux prédicat (mutuellement récursifs) `longueurpaire(L)` est `longueurimpaire(L)` qui réussissent si leur argument est une liste avec un nombre pair (impair) d’éléments (n’utilisez pas d’opérations arithmétiques).
- Définir le prédicat `miroir(L,L1)` pour inverser une liste. Par exemple `miroir([a,b,c],L)` donne $L = [c,b,a]$. (Pour ajouter un élément à la fin d’une liste on peut utiliser `concat`). Est-ce que votre programme marche aussi avec `miroir(L, [a,b,c])` ?

Exercice 2 1. Écrire un prédicat `insert(Entier,ListeArgument,ListeResultat)` pour insérer un entier donné dans une liste d’entiers donnée, que l’on suppose triée, de telle façon que la liste résultat reste triée.

Exemple d’utilisation:

```
?- insert(7, [3,5,13], R) .  
R = [3,5,7,13] ?  
yes
```

2. Écrire un prédicat `tri(ListeArgument,ListeResultat)` pour trier une liste d’entiers. L’algorithme à utiliser est celui du “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 utilisant `insert`.

Exemple d'utilisation:

```
?- tri([5,3,13,1],R).  
R = [1,3,5,13] ?  
yes
```

3. 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.

Écrire un prédicat `trie(ListeArgument)` qui réussit si l'argument est une liste triée, échoue sinon.

Exemples d'utilisation:

```
?- trie([6,13,90]).  
yes  
?- trie([4,2]).  
no
```

Exercice 3 En vous inspirant du prédicat `fonction/3` du cours, définir un prédicat `injection(domaine, codomaine, fonction_injective)`, tel que (par exemple):

```
[eclipse 2]: injection([a,b],[1,2,3],L).
```

```
L = [(a, 1), (b, 2)]  
Yes (0.00s cpu, solution 1, maybe more) ? ;
```

```
L = [(a, 1), (b, 3)]  
Yes (0.00s cpu, solution 2, maybe more) ? ;
```

```
L = [(a, 2), (b, 1)]  
Yes (0.00s cpu, solution 3, maybe more) ? ;
```

```
L = [(a, 2), (b, 3)]  
Yes (0.00s cpu, solution 4, maybe more) ? ;
```

```
L = [(a, 3), (b, 1)]  
Yes (0.00s cpu, solution 5, maybe more) ? ;
```

```
L = [(a, 3), (b, 2)]  
Yes (0.00s cpu, solution 6, maybe more) ? ;
```

```
No (0.00s cpu)
```

Exercice 4 Définir un prédicat `afficher/1` tel que, pour n entier naturel, `afficher(n)` affiche les entiers de 0 à n dans l'ordre (pour l'affichage, utiliser le prédicat prédéfini `write/1`).

Exercice 5 Définir un prédicat `boucle/0` qui affiche à l'écran "choisir un entier", lit un entier, si l'entier donné est 0 termine, sinon repose la question est ainsi de suite. (Pour lire un entier, utiliser le prédicat prédéfini `read/1`; l'exécution de la requête `read(X)` a pour effet de donner la main à l'utilisateur, qui rentre un terme suivi d'un point. La variable X est alors unifiée avec ce terme).

Exercice 6 (casse-tête) Indiquer ce que font les programmes suivants en général, et illustrer votre réponse sur le cas particulier proposé.

1. $a(0,0)$.

$a(X,Y):- V \text{ is } X-1, a(V,Z), Y \text{ is } Z + X.$

cas particulier: $X=5$.

2. $b(0,0)$.

$b(X,Y):-V \text{ is } X-1, b(V,Z), Y \text{ is } Z + 2*V + 1.$

cas particulier: $X=9$.

3. $c(0,1)$.

$c(X,Y):- V \text{ is } X-1, c(V,Z), Y \text{ is } Z+Z.$

cas particulier: $X=7$.

4. $d(X,0,1)$.

$d(X,Y,R):- Z \text{ is } Y-1, d(X,Z,T), R \text{ is } T*X.$

cas particulier: $X=5,Y=3$.