

S08 - Récursivité et Listes

Instructions

- Date butoir : Jeudi prochain à 05h00
- Travail à rendre : un fichier `.zip` contenant tout votre travail pour la série
- N'oubliez pas de mettre votre nom au début du fichier

Tests

Pour tester vos prédicats, utilisez la commande suivante :

```
swipl -g "<filename>" -g run_tests -t halt s08_tests.plt
```

Exercice 1 : Instantiation patterns

Expliquez les modes de notation suivants. Pour chaque mode, proposez un exemple et expliquez-le.

1. `+`
2. `++`
3. `-`
4. `?`

Exercice 2 : Prédicats sur des listes

Développez les prédicats suivants **sans utiliser** les prédicats prédéfinis équivalents.

- `my_length(+L,?N)` : vrai si `N` est le nombre d'éléments de la liste `L`

Exemple : `?- my_length([a,b,c],N).` retourne `N = 3.`

- `my_append(?L1,?L2,?L3)` : vrai si `L3` est la concaténation des listes `L1` et `L2`

Exemple : `?- my_append([a,b,c],[d,e],L).` retourne `L = [a,b,c,d,e].`

- `my_last(+L,?E)` : vrai si E est le dernier élément de L

Exemple : `?- my_last([a,b,c],X)` . retourne `X = c` .

- `my_select(?E,+L1,?L2)` : vrai si L2 est la liste L1 sans une occurrence de E

Exemple : `?- my_select(b,[a,b,c],L)` . retourne `L = [a,c]` .

- `my_sum_list(+L,?S)` : vrai si S est la somme des éléments de L

Exemple : `?- my_sum_list([1,2,3],X)` . retourne `X = 6` .

- `my_reverse(+L1,?L2)` : vrai si L2 est la liste L1 renversée à tous les niveaux (y compris les listes internes)

Exemple : `?- my_reverse([a,[b,c],d,e,[f,g,h]],L)` . retourne
`L = [[h,g,f],e,d,[c,b],a]` .

- `my_min_member(?M,+L)` : vrai si M est le plus petit élément de la liste L

Exemple : `?- my_min_member(X,[a,b,c])` . retourne `X = a` .

- `atoms(+L)` : vrai si L est une liste dont les éléments sont des atoms. Utilisez le prédicat prédéfini `atom/1`.

Exemple : `?- atoms([a,[1,2],b,c])` . retourne `false` .

- `insert(+E,?L1,?L2)` : vrai si L2 est la liste triée L1 dans laquelle l'élément E est ajouté à sa place. On part du principe que L1 est triée.

Exemple : `?- insert(5,[2,4,7,12],L)` . retourne `L = [2,4,5,7,12]` .

- `merge_lists(+L1,+L2,?L3)` : vrai si L3 est triée et elle est la fusion des listes triées L1 et L2. On part du principe que L1 et L2 sont triées.

Exemple : `?- merge_lists([2,6,9],[1,8,13,23],L)` . retourne
`L = [1,2,6,8,9,12,23]` .