Prolog : TP3

jeudi 25 septembre 2014

Maud Leray
Corentin Nicole
classe 1.2

Question 1

```
/* QUESTION 1.1 */

membre(A,[A|_]).
membre(A,[_|Reste]):- membre(A,Reste).

/********** TESTS **********

[eclipse 2]: membre(1,[1,2,3]).

Yes (0.00s cpu, solution 1, maybe more) ? ;

[eclipse 3]: membre(4,[1,2,3]).

No (0.00s cpu)

*****************************/

compte(_,[],0).
compte(A,[A|Reste],Cpt):- compte(A, Reste,Cptbis) ,
                     Cpt is Cptbis + 1.
compte(A,[Tete|Reste],Cpt):- \==(A,Tete),
                       compte(A,Reste,Cpt).


/********** TESTS **********

[eclipse 34]: compte(1,[1,1,1,3,1],N).

N = 4
Yes (0.00s cpu, solution 1, maybe more) ?
[eclipse 35]: compte(145,[1,1,1,3,1],N).

N = 0
Yes (0.00s cpu)

*****************************/

renverser([],A,A).
renverser([Tete1|Reste1],A,R) :-renverser( Reste1, [Tete1|A],R).

/********** TESTS **********

[eclipse 48]: renverser([1,2,3],[],N).
  (1) 1 CALL renverser([1, 2, 3], [], N) %> creep
  (2) 2 CALL renverser([2, 3], [1], N) %> creep
  (3) 3 CALL renverser([3], [2, 1], N) %> creep
  (4) 4 CALL renverser([], [3, 2, 1], N) %> creep
  (4) 4 EXIT renverser([], [3, 2, 1], [3, 2, 1]) %> creep
  (3) 3 EXIT renverser([3], [2, 1], [3, 2, 1]) %> creep
  (2) 2 EXIT renverser([2, 3], [1], [3, 2, 1]) %> creep
  (1) 1 EXIT renverser([1, 2, 3], [], [3, 2, 1]) %> creep
```

```
N = [3, 2, 1]
Yes (0.00s cpu)

*****************************/

comparer([],[]).
comparer([Tete1|Reste1],[Tete1|Reste2]):- comparer(Reste1,Reste2).

palind(L1):- renverser(L1,[],L2),
          comparer(L1,L2).

/*********** TESTS ***********

[eclipse 2]: palind([1,2,3,2,1]).

Yes (0.00s cpu)
[eclipse 3]: palind([1,2,3,2,1,65]).

No (0.00s cpu)

*****************************/

nieme(1,[Tete|_],Tete).
nieme(N,[_|Reste],A) :-\==(N,1),
                  nieme(N2,Reste,A),
                  N is N2+1.

/*********** TESTS ***********

[eclipse 3]: nieme(5,[1,2,3,4,5,6,7,8,9],A).
A = 5

*****************************/

hors_de(_,[]).
hors_de(A,[Tete|Reste]):- \==(A,Tete),
                    hors_de(A,Reste).

/*********** TESTS ***********

[eclipse 6]: hors_de(5,[1,2,3]).
Yes (0.00s cpu)

[eclipse 7]: hors_de(3,[1,2,3]).
No (0.00s cpu)

*****************************/

tous_diff([]).
tous_diff([Tete|Reste]) :-hors_de(Tete,Reste),
                     tous_diff(Reste).

/*********** TESTS ***********
```

```
[eclipse 9]: tous_diff([1,2,3,4,5]).
Yes (0.00s cpu)

[eclipse 10]: tous_diff([1,4,3,4,5]).
No (0.00s cpu)

*****************************/

conc2([],L,L).
conc2([Tete|Reste],L2,[Tete|R]):-conc2(Reste,L2,R).

conc3([],[],L,L).
conc3(L1,L2,L3,R):-conc2(L1,L2,L12),
                   conc2(L12,L3,R).

/*********** TESTS ***********

[eclipse 24]: conc3([1,2,3],[4,5,6],[7,8,9],R).

R = [1, 2, 3, 4, 5, 6, 7, 8, 9]
Yes (0.00s cpu)

*****************************/

debute_par(_,[]).
debute_par([Tete1|Reste1],[Tete1|Reste2]) :-debute_par(Reste1,Reste2).

/*********** TESTS ***********

[eclipse 26]: debute_par([1,2,3,4,5,6],[1,2,3]).

Yes (0.00s cpu)
[eclipse 27]: debute_par([1,2,4,4,5,6],[1,2,3]).

No (0.00s cpu)

*****************************/

sous_liste(L1, L2) :-debute_par(L1,L2).
sous_liste([_|Reste1], L2) :-sous_liste(Reste1,L2).

/*********** TESTS ***********

[eclipse 39]: sous_liste([1,2,4,1,2,3],[1,2,3]).

Yes (0.00s cpu, solution 1, maybe more) ? ;

No (0.00s cpu)
[eclipse 40]: sous_liste([1,2,4,1,2,3],[1,2,3]).

Yes (0.00s cpu, solution 1, maybe more) ?
[eclipse 41]: sous_liste([1,2,4,1,2,8],[1,2,3]).

No (0.00s cpu)
```

```
*****************************/

elim([],L,L).
elim([Tete|Reste],L2,R):-(membre(Tete,L2),
                    elim(Reste,L2,R));
                    elim(Reste,[Tete|L2],R).

/*********** TESTS ***********

[eclipse 43]: elim([1,2,2,2,3,7,4,5,5,9],[],R).

R = [9, 5, 4, 7, 3, 2, 1]

*****************************/


inserer(E,[Tete|Reste],[Tete|Reste2]):-E>Tete,
                        inserer(E,Reste,Reste2).
inserer(E,L1,[E|L1]).


/*********** TESTS ***********

| ?- inserer(4,[1,2,3,5],L).

L = [1,2,3,4,5] ?

| ?- inserer(4,[1,2],L).

L = [1,2,4] ?

*****************************/

tri([E],[E]).
tri([Tete|Reste],R) :-tri(Reste,Rtmp), inserer(Tete,Rtmp,R).

/*********** TESTS ***********

| ?- tri([9,8,7,6,5,4,3,2,1],L).

L = [1,2,3,4,5,6,7,8,9] ?

*/
```

Listing 1: listeQ1.pl

Question 2

```
/* QUESTION 2.1 */

inclus([],_).
inclus([Tete|Reste],Y):-membre(Tete,Y),
```

```
                    inclus(Reste,Y).

/*********** TESTS ***********

| ?- inclus([2,3,6],[1,2,3,4,5]).

no
| ?- inclus([2,3,4],[1,2,3,4,5]).

true ?

*****************************/

non_inclus([Tete|Reste],Y):-(membre(Tete,Y),
                        non_inclus(Reste,Y));
                        hors_de(Tete,Y).

/*********** TESTS ***********

| ?- non_inclus([2,3,4],[1,2,3,4,5]).

no
| ?- non_inclus([2,3,42],[1,2,3,4,5]).

true ?

*****************************/

union_ens([],L,L).
union_ens([Tete|Reste],Y,Z):-(non_inclus([Tete],Y),
                         union_ens(Reste,[Tete|Y],Z));
                         union_ens(Reste,Y,Z).

/*********** TESTS ***********

| ?- union_ens([1,2,3],[4,5,6],Z).

Z = [3,2,1,4,5,6] ?

yes
| ?- union_ens([1,2,2],[3,5,6],Z).

Z = [2,1,3,5,6] ?

yes
| ?- union_ens([1,2,3],[3,5,6],Z).

Z = [2,1,3,5,6] ?

yes

*****************************/
```

Listing 2: listeQ2.pl