Université d'Orléans - M1 Informatique - Module PLC - TD $n^o 4$
2012-2013

– Première partie –

Utilisation de quelques prédicats prédéfinis "spéciaux"

1. Prédicat : **univ**

   Ecrire un prédicat qui réalise, au choix, la somme ou la multiplication (ou autre) des éléments deux à deux d'une liste :
   Le but `appliquer(plus, [1,2,3,4,5,6],L)` donnerait la réponse `L = [3,7,11]`

   Comment faire pour appeler ce prédicat avec : `appliquer('+', [1,2,3,4,5,6],L)`

2. Pédicats : **bagof**, **setof** et **findall**

   Soit le programme :

   ```
   p(1,3,5). p(2,4,1). p(3,5,2). p(4,3,1). p(5,2,4).
   ```

   Donner le résultat du but : `bagof(Z,p(X,Y,Z), Sac)`. Comment faire en n'utilsant que bagof pour avoir le résultat escompté (sans affichage des variabe X et Y et avoir la bonne liste dans Sac)
   Donner le résultat du but : `bagof(Z,(p(X,Y,Z), Z>6), Sac)`. Comment faire pour avoir la liste vide comme réponse.
   Si le prédicat **findall** n'existait pas, l'inventer en définissant un prédicat **mon_findall/3**.

3. Prédicat : négation par l'échec \+

   Soit le programme :

   ```
   p(X) :- q(X), \+ r(X).
   r(X) :- w(X), \+ s(X).
   q(a). q(b). q(c).
   s(a). s(c).
   w(a). w(b).
   ```

   Ecrire l'arbre de recherche résultant du but `p(a)`. Que rend le but `p(b)` ?

   Même question (arbre de recherche résultant du but `p(a)`) avec le programme :

   ```
   p(X) :- \+ s(X).
   s(X) :- s(f(X)).
   ```