



TD5 - PROLOG (PROgrammation LOGique)

1. Traitement de listes

- On utilise les listes sans répétition pour représenter les ensembles.

Définir le prédicat `intersection(L1, L2, L3)`, où la liste `L3` est l'intersection de deux ensembles, représentés par les listes `L1, L2`.

- Dérouler à la main le fonctionnement du moteur de Prolog sur l'exemple suivant :

?- `intersection([a, b, c, d], [d, b, e], L)`.

Exemple

?- `intersection([a, b, c, d], [d, b, e], L)`.
`L = [b, d]`

?- `intersection([1, 2, 3], [4, 2, 5, 6], L)`.
`L = [2]`

2. send + more = money (cryptarithmétique)

Ecrire un programme (optimisé) en Prolog qui résout ce problème sachant que nous avons un ensemble de huit variables, `s, e, n, d, m, o, r, y`, toutes différentes, qui satisfont aux contraintes du problème posé, à savoir les règles de l'addition.

Comme une addition se fait colonne par colonne (FIG. 1), nous allons définir le prédicat `somme`, qui fait la somme d'une colonne.

Ainsi, `somme(r, x, y, z, r')` exprime que la somme de la retenue `r` avec `x` et `y` est égale à `z + 10*r'`, `r'` étant la prochaine retenue.

Le prédicat d'appel devra s'appeler : `solution([S,E,N,D,M,O,R,Y])`.

0	R1	R2	R3	R4	0
+	0	S	E	N	D
+	0	M	O	R	E
=	M	O	N	E	Y

FIG. 1