*LS4 – Programmation Fonctionnelle*
*TD N°3*
Récursivité

**1) Fonction d'Ackerman**
Cette fonction de deux entiers (qui possède la propriété de croître très rapidement) se définit par :
```
Si m = 0, alors A(0, n) = n+1,
sinon si n=0, A(m, 0) = A(m-1, 1)
sinon A(m, n) = A(m-1, A(m, n-1))
```
**2)Ecrire une fonction qui renvoie la somme de ses deux paramètres si elle est positive, et 0 sinon (un seul calcul de la somme !).**
**3) Ecrire une fonction faisant la somme de deux nombres passés en paramètre en utilisant que les fonctions add1 et sub1.**
**4) Fonctions mystère**
On définit les fonctions `f1` et `f2` de cette façon :
```
(define (f1 n)
(or (= n 0) (f2 (- n 1))))
(define (f2 n)
(and (not (= n 0)) (f1 (- n 1))))
```
**a – donner le résultat des appels :**
```
(f1 6)
(f1 3)
```
**b- plus généralement, dire ce que calculent les fonctions f1 et f2 (quand on les applique à un entier positif)**
**5) Ecrire une fonction qui supprime les doublons d'une liste plate**
```
(dedoublonne '(a b c d e f))
(a b c d e f)
(dedoublonne '(a f b d c d a e f))
(b c d a e f)
```
**6) même fonction mais qui conserve l'ordre initial**
```
(dedoublonne_2 '(a b c d e f))
(a b c d e f)
(dedoublonne_2 '(a f b f d c d a e f))
(a f b d c e)
```
**7) fonction qui supprime le dernier élément d'une liste**
```
(tsd '(a b c d e f))
(a b c d e)
```
**8) fonction qui prend en paramètre une liste plate et qui renvoie #t si tous les éléments sont égaux (au sens de eq?)**
```
(tous-egaux? '(a s d f g))
#f
(tous-egaux? '(r r r r a))
#f
(tous-egaux? '(r r r r r))
#t
```
**9) fonction (pyra n) qui affiche :**
```
0
11
222
…
nnn … n
```
**10) même chose mais :**
```
        n
        …
  2 … 22
 11 … 11
000 … 00
```