

## Correction examen partiel (Programmation fonctionnelle avec Caml)

A. ED-DBALI

5 novembre 2001 – durée : 2h

**Exercice 1** (6 points) : Typier et évaluer les expressions suivantes :

1. `let f = function u -> function x -> u x x;;`  
`f : ('a -> 'a -> 'b) -> 'a -> 'b`
2. `let x = (function x -> function y -> function u -> u x + y) 4.0;;`  
`x : int -> (float -> int) -> int`
3. `let y = function x -> function y -> function z -> y x z;;`  
`y : 'a -> ('a -> 'b -> 'c) -> 'b -> 'c`
4. `let z = function u -> function x -> fst(u (fst x) x);;`  
`z : ('a -> 'a * 'b -> 'c * 'd) -> 'a * 'b -> 'c`
5. `let t = (function x -> function y -> snd (fst y (snd x))) (1, 'a');;`  
`t : (char -> 'a * 'b) * 'c -> 'b`

**Exercice 2** (5 points) : Soit *somme* la fonction (vue en cours) définie par :

```
let rec somme (op,e) f a b =
  if a > b then
    e
  else
    op (f a) (somme (op,e) f (a+1) b);;
```

Utilisez cette fonction pour écrire une fonction pour calculer l'expression  $E(n)$  définie par :

$$E(n) = \sum_{i=1}^{i=n} \left( \prod_{j=1}^{j=n} j^i \right)$$

```
let rec somme (op,e) f a b =
  if a>b then
    e
  else
    op (f a) (somme (op,e) f (a+1) b);;
```

```
let rec puissance (x, n) =
  if n = 0 then
    1
  else
    x * (puissance (x, n-1));;
```

```
let e n =
  let produit i =
    somme (( * ), 1) (function x -> puissance (x, i)) 1 n
  in somme ((+), 0) produit 1 n;;
```

### Exercice 3 (9 points) : Bavarder sur les chaînes :

1. Ecrire une fonction (chiffre  $C$ ) qui prend un chiffre sous sa forme caractère et renvoie le même chiffre sous sa représentation entière. Exemple: `chiffre '3' = 3`.
2. Ecrire une fonction (caractère  $C$ ) qui prend un chiffre sous sa forme entière et renvoie le même chiffre sous sa représentation caractère. Exemple: `caractère 3 = '3'`.
3. Ecrire une fonction (`nbr_chiffre C S`) qui donne le nombre d'occurrences du chiffre  $C$  dans la chaîne  $S$ . Attention  $C$  est de type `int` tandis que les éléments de  $S$  sont des caractères. Exemple: `nbr_chiffre 2 "5244221" = 3`.
4. Ecrire une fonction (`décrire S`) qui part de la chaîne  $S$  et construit une chaîne qui décrit son contenu : chaque chiffre de 0 à 9 présent dans  $S$  donne lieu, dans la chaîne résultat, à son nombre d'occurrences dans  $S$ . Exemples :
  - `décrire "27746448" = "1234162718"` (il y a 1 fois le chiffre 2, 3 fois le chiffre 4, 1 fois le chiffre 6, 2 fois le chiffre 7 et 1 fois le chiffre 8).
  - `décrire "3" = "13"` (il y a 1 fois le chiffre 3).

Les chiffres entre 0 et 9 qui n'appartiennent pas à la chaîne  $S$  ne sont pas décrits dans la chaîne résultat.

(\* Question 1 \*)

```
let chiffre c = int_of_char(c) - int_of_char('0');;
(* chiffre : char -> int *)
```

(\* Question 2 \*)

```
let caractere c = char_of_int((int_of_char '0') + c);;
(* caractere : int -> char *)
```

(\* Question 3 \*)

```
let rec nbr_chiffre c s =
  if s = "" then
    0
  else
    nbr_chiffre c (String.sub s 1 (String.length s - 1)) +
    if (chiffre s.[0]) = c then 1 else 0;;
(* nbr_chiffre : int -> string -> int *)
```

(\* Question 4 \*)

```
let decrire s =
  let rec decrire_chiffre s n =
    match n with
    | 10 -> "" (* j'ai fini de compter les chiffre de 0 a 9 *)
    | c -> if nbr_chiffre c s = 0 then (* C chiffre absent de S *)
            decrire_chiffre s (c+1)
          else
            string_of_int(nbr_chiffre c s) (* mettre le nbr d'occ de C dans S *)
            ^ (string_of_int c) (* le faire suivre de C *)
            ^ (decrire_chiffre s (c+1)) (* puis refaire pour le chiffre suivant *)
  in decrire_chiffre s 0;; (* commencer le comptage au chiffre 0 *)
(* decrire : string -> string *)
```