

TD de *Logique et Circuits* n° 1**Introduction à Caml**

Exercice 1 Quel est le résultat (type et valeur) des commandes suivantes ? (Certaines peuvent comporter des erreurs.)

```
# 3 + 7;;
# 3.5 + 4.3;;
# 4 / 3;;
# 3 ** 2;;
# 5 = 6;;
# ceil((if 3 < 4 then 3.4 else 2.3) -. 4.5);;
```

Exercice 2 *i)* On effectue les deux commandes suivantes :

```
let a = let a = 3 and b = 2 in let a = a + b and b = a - b in a * a - b * b;;
let b = 2 in b + a * b;;
```

Quel est le résultat de chaque commande ?

ii) *Albert a 15 ans, Béatrice en a 19. L'âge de Charles est la somme de ceux d'Albert et Béatrice, mais l'âge de la maison qu'ils habitent est 3 fois la somme des âges de Béatrice et de Charles. Écrire une expression Caml qui traduise cet énoncé et dont la valeur est l'âge de la maison.*

iii) On tape successivement les commandes suivantes en Caml. Quelles sont les valeurs successives des variables ? (Certaines lignes peuvent provoquer des erreurs.)

```
let y = let x = 4 in x + 1 and z = 25;;
let x = let t = y + z in t / y;;
let y = let z = t - y in x * z;;
let t = let x = 3 in z * x;;
let t = t / x;;
```

Exercice 3 Soit le polynôme $P(x) = x^2 + 5x + 4$. Écrire une fonction qui calcule la valeur de ce polynôme pour tout flottant x .

Exercice 4 Écrire une fonction qui prend deux arguments de même type et retourne un couple contenant le plus petit comme première composante et le plus grand comme seconde.

Exercice 5 On définit la fonction `xor` (« ou » exclusif) dont l'argument est un couple de deux booléens et qui retourne la valeur *vrai* si et seulement si un seul des deux booléens est vrai. Écrire la table de la fonction `xor`. Comment définir la fonction `xor` à l'aide des opérateurs `or`, `&` et `not` ? Écrire la fonction Caml correspondante.

Écrire une fonction équivalente qui s'applique sur un couple de deux booléens. Quel opérateur agit en Caml sur les booléens comme la fonction `xor` ?

Exercice 6 Que fait la fonction suivante ? Quelle est son type ?

```
let f(x) = char_of_int ((int_of_char x) + 1);;
```

Exercice 7 Écrire une fonction qui prend un argument n de type `int` et renvoie 0 si n vaut 0 ou 1 et $n - 2$ sinon. On écrira d'abord cette fonction avec `if...then...else` puis avec `match`.

Exercice 8 Considérons la fonction suivante :

```
let f x y =  
  if y = 3 then 5 else if x then y + 1 else 0;;
```

Quel est le type des paramètres ? Traduire cette fonction en utilisant `match`.

Traduire la fonction suivante en utilisant `if...then...else` :

```
let g(x, y) =  
  match (x - y, y) with  
  | (0, _) -> x  
  | (_, 0) -> 0  
  | _      -> x + y;;
```

Exercice 9 On considère l'expression suivante, extraite d'un programme qui peut comporter des définitions antérieures que l'on ne connaît pas ici.

```
let x = 2.5 and y = true in  
if f x (g(y, "true")) > 3  
then let z = g(y, t y) in (z/.3.14)*.i(x)  
else let w = j("true", 'a') in k w;;
```

Énumérer les différents identifiants qui apparaissent dans cette expression. Déterminer leur type ainsi que celui de l'expression.

Exercice 10 Écrire une fonction `car` qui, à tout couple (x, y) d'éléments de type quelconque formé d'un élément de type quelconque T et d'un entier, associe une fonction qui prend un élément de type T et retourne y si l'argument est égal à x et 0 sinon.

Quel est le type de la fonction `car` ?

Exercice 11 Préciser les types des fonctions `f`, `g` et `h` définies ci-dessous :

```
let f x (a, b) = (x a, x b);;  
let g x y = (f x (2, 1), f y (true, false));;  
let h x y (a, b) = f x (f y (a, b));;
```

Exercice 12 Que fait la fonction `f2` définie ci-dessous ?

```
let f1 a b =  
  if a < b then (a, b) else (b, a);;
```

```
let f2 x y =  
  function z ->  
    let (x, y) = (f1 x y) in  
    let (y, z) = (f1 y z) in  
    let (x, y) = (f1 x y) in (x, y, z);;
```

Quel est le type de `f2` ?