

TD n°7 - Types

Exercice 1

Prévoir le type et le résultat des expressions suivantes :

1. `let x = 2 in let x = 1 in x;;`
2. `let x = 2 in let y = x in let x = 1 in 10 * x + y;;`
3. `let y = 2 in let x = y + 1 in let y = x + 2 in x + y;;`
4. `let (x, y) = (1, 2) in y;;`
5. `let x = 1 in let y = 2 in let (y, x) = (x, y) in y;;`
6. `let f x = 3 * x in f 3 - 1;;`
7. `let f x = 3 * x in (f 3) - 1;;`

Exercice 2

Déterminer le but de la fonction `f` et son type.

```
let fct (a, b, c) =
  let m = a in
    if b > a then
      if c > b then m = c else m = b
    else
      if c > a then m = c else m = a
;;
```

Exercice 3 (fonctions numériques)

Prévoir le type des expressions suivantes :

```
let f x = float_of_int x +.2.;;
let g x y = x +. y;;
let h x y z = let p = float_of_int (x+y) in p**z;;
let i (x,y) = int_of_char y + 2*x;;
let j (x,y) = x*.2. +. (f y);;
```

Exercice 4 (types polymorphes)

Prévoir le type des fonctions suivantes :

```
let f x y = x<>y;;
let g x y = (x,y);;
let h x y z = if z then (x,y,0)
              else (x+.1, y,1);;
let i x y = (x, x+3);;
let j x y z = (x z, y);;
```

Exercice 5 (Autres types)

On définit un type énumération `type ex = A|B|C;;`.

Prévoir le type des expressions suivantes :

```
let x = B;;
let f1 x y = if x=A then y
             else if x=B then y
             else 0;;
let f2 () = A;;
let f3 x = if x=A then Printf.printf "A"
           else if x=B then Printf.printf "B"
           else Printf.printf "C";;
(*pour les deux suivants il faut connaitre les listes*)
let f4 x = [];;
let f5 x y z t = if x>y then z
                 else if t>0 then [3]
                 else z;;
```

Exercice 6 (Et dans l'autre sens?)

Dans cet exercice, on va s'intéresser à l'exercice inverse de ce qu'on faisait avant : à partir d'un type, écrire quelque chose de ce type.

1. Écrire une expression de type `int*float`
2. Écrire une fonction de type `int -> int -> int`
3. Même chose avec `float -> int -> float`
4. Même chose avec `'a -> 'b -> 'b`
5. Même chose avec `float -> int*float -> int*int*int`
6. Même chose avec `'a -> int -> 'a -> 'a*'a*float`
7. Même chose avec `'a list -> 'b -> int -> 'b list`
8. Même chose avec `'a list -> 'b -> int -> ('a*'b) list`
9. Même chose avec `'a*'b -> 'c*'d -> 'a*'d -> 'c*'b`
10. Est-il possible d'écrire une expression globale de type `int -> 'a`? De type `'a`?
11. Et une expression locale? Donner des exemples.