Université A.Mira Béjaia **Département Maths MI 1ière année**

Examen de Calcul formel - S2 / 12 Juin 2008

Nom :	Emargement :
Prénom :	
Groupe :	Matricule : .....

Exercice1 : (2.5 pts) Donner un exemple d'expression pour chacun des types suivants :

1. 'a->int
2. int ->int ->string
3. int*int -> int*int
4. ('a ->int) ->'a ->int
5. int list list -> int list

```
# fun x ->2;;
# fun x y -> string_of_int (x*y);;
# fun (x,y) -> (x*x,y*y);;
# fun f x -> 2*(f x);;
# let rec f l = match l with []->[0] |t::r->t @ f r;;
```

Exercice2 : (2.5 pts) Pour chacune des expressions suivantes, si elle est correcte dire son type et si elle est incorrecte, indiquer pourquoi ?

1. # let f x y = y ;;
2. # let f x y = if true then y else x ;;
3. # let f x y = if x then y else x ;;
4. # let rec f x y z = if x > 2.3 then y else z ;;
5. # let rec f x y z = match x with
   t::r -> y t (f r y z)
   | _ -> z ;;

```
val f : 'a -> 'b -> 'b = <fun>
val f : 'a -> 'a -> 'a = <fun>
val f : bool -> bool -> bool = <fun>
val f : float -> 'a -> 'a -> 'a = <fun>
val f:'a list ->('a->'b->'b)->'b->'b=<fun>
```

Exercice 3 : (2 pts) Evaluer les expressions suivantes :

1. let x=3 in let y = x+1 in let x=x+y
   in let y=x-y in (x,y);;
2. let x=3 and y = x+1
   in let x=x+y and y=x-y in (x,y) ;;
3. let f x = x-1 in let f x = f (x-1) in f 2 ;;
4. let rec f n = if n=0 then 1
   else n*f(n-1) in (f 3)+(f 4) ;;

```
- : int * int = (7, 3)
```

```
let x=3 and y=x+1 in let x=x+y and y=x-y in (x,y);;
                     ^  Unbound value x
```

```
- : int = 0
- : int = 30
```

Exercice 4 : (2 pts) Soit la fonction récursive suivante :

```
# let rec etrange l = match l with
    [ ]->0
   |x::xs->1+etrange xs;;
```

– Donner le typage de la fonction « etrange »

```
val etrange : 'a list -> int = <fun>
```

– Que retourne l'exécution de # etrange [1;0;4;2];;

```
- : int = 4
```

Exercice 5 : (3 pts) On définit les monômes par le type suivant :

type monôme = { coefficient : int ; degré : int} ;;

Un polynôme sera alors constitué d'une liste de monômes, cette liste n'étant pas supposée être triée.
Question : Ecrivez une fonction de multiplication de deux monômes.

```
# type monome = {coefficient : int ; degré : int};;
type monome = { coefficient : int; degré : int; }

# let mult_monôme m n = {coefficient = m.coefficient*n.coefficient ; degré = m.degré+n.degré};;
val mult_monôme : monome -> monome -> monome = <fun>
```

Exercice 6 (4 pts) :Ecrire une fonctionnelle `forall` qui, étant donné un prédicat `p` et une liste `l`, retourne la valeur `true` si tous les éléments de `l` satisfont `p` et retourne `false` sinon.

– Donner une version avec filtrage

```
# let rec forall p l = match l with
  [ ]->true
  |t::r-> if p t then forall p r else false;;
val forall : ('a -> bool) -> 'a list -> bool = <fun>

Exemples :
# forall (p n -> n<>0) [-3;-2;0;1;2;3];;
- : bool = false
# forall (p n-> n<>0) [-3;-2;-1;1;2;3];;
- : bool = true
```

– Donner une version sans filtrage

```
# let rec forall p l = if l = [ ] then true
    else if  p ( List.hd l ) then forall p ( List.tl  l )
         else false;;
val forall : ('a -> bool ) -> 'a list  -> bool = <fun>
```

Exercice 7 (4 pts) : Ecrire une fonction `somme_tetes` qui calcule la somme des têtes d'une liste de listes, en utilisant la fonction `tete` qui donne la tête d'une liste et déclenche une exception si une liste est vide.

```
# let tete = function
  t::r->t
  |_->failwith "liste vide";;
val tete : 'a list -> 'a = <fun>

# let rec somme_tetes = function
  []->0
  |t::r->(tete t)+(somme_tetes r);;
val somme_tetes : int list list -> int = <fun>
```

Que retourne l'exécution de :

```
# somme_tetes [[1;5;2];[-3;4];[7]];;
- : int = 5
```

```
# somme_tetes [[1;5;2];[];[7]];;
Exception: Failure "liste vide".
```