

Corrigé type du module Calcul Formel

Exercice 1.....(7 pts)

- a) 1. # (function x -> x 2);;
- :(int -> 'a) -> 'a = <fun> (0.5 pt)
- 2. # (function x -> x 2) (+);;
- :int -> int = <fun> (0.5 pt)
- 3. # (function x -> x 2) (+) 5;;
- :int = 7 (0.5 pt)
- 4. # (function x -> x 2) (+) 5 5;;
ERREUR (0.5 pt)
- b) 1. # let x=[0;1;2] in (let x=List.tl in x [3;4]) @ x;;
- :int list = [4;0;1;2] (2 pts)
- 2. # let f a b liste = a(liste) + b(liste);;
val f :('a -> int) -> ('a -> int) -> 'a -> int = <fun> (1 pt)
- 3. # let f (x,y) (z,w) = [x;z];;
val f :'a * 'b -> 'a * 'c -> 'a list = <fun> (2 pts)

Exercice 2.....(6 pts)

- a) (3 pts)
let rec remplacer x y l = match l with [] -> []
| a::b -> (if a=x then y else a) ::remplacer x y b;;
val remplacer :'a -> 'a -> 'a list -> 'a list = <fun>
- b) (3 pts)
let rec extraire_element l n = match (l,n) with
([],_) -> failwith "liste trop courte"
| (a::b,1) -> a
| (a::b,n) -> if n<1 then failwith "indice invalide"
else extraire_element b (n-1);;
val extraire_element :'a list -> int -> 'a = <fun>

Exercice 3.....(7 pts)

- a) (3 pts)
let rec f (x,n) =
if n<0 then failwith "error"
else match n with
0 -> x
| _ -> x +. float_of_int(n*n) +. f (x,n-1);;
val f : float * int -> float = <fun>
- b) 1. # f(1.00,2);;
- :float = 8. (2 pts)
- 2. # f 0;;
ERREUR (1 pt)
- 3. # (function a -> 2.55) f;;
- :float = 2.55 (1 pt)