Corrigé série TP n°3
Les fonctions récursives en Caml

Note : Chaque exercice est suivi par une séquence du symbole "*" indiquant son niveau de difficulté et d'un temps estimatif pour sa résolution.

Exercice 1 (*/5m)
let rec fact(n) = if (n=0) then 1 else n*fact(n-1);;

Réponse de Caml :

fact : int → int = <fun>

Exercice 2 (*/5m)
let rec exp(x,y) = if (y=0) then 1 else x*exp(x,y-1);;

Réponse de Caml :

exp : int * int → int = <fun>

Exercice 3 (*/5m)
let rec sigma(x) = if (x=0) then 0 else x+sigma(x-1);;

Réponse de Caml :

sigma : int → int = <fun>

Exercice 4 (*/10m)
Hypothèse : On suppose que : a<=b.

Solution 1 : Utiliser la fonction sigma.

let rec sigma21(a,b) = sigma(b) - sigma(a) + a;; //On a ici une récursivité indirecte.

Réponse de Caml :

sigma21 : int * int → int = <fun>

Solution 2 : Utiliser la récursivité directe.

let rec sigma22(a,b) = if (a=b) then b else a + sigma22(a+1,b);;

Réponse de Caml :

sigma22 : int * int → int = <fun>

Solution 3 : Utiliser la récursivité directe.

let rec sigma23(a,b) = if (b=a) then a else b+sigma23(a,b-1);;

Réponse de Caml :

sigma23 : int * int → int = <fun>

Exercice 5 (*/5m)
//A traiter par les étudiants

Exercice 6 (*/5m)
//A traiter par les étudiants

Exercice 7 (*/5m)
//A traiter par les étudiants

Exercice 8 (*/10m)

//A traiter par les étudiants

Exercice 9 (**/20m)

(* Rappel : Un nombre entier naturel est premier si et seulement si il possède 2 diviseurs : 1 et lui-même. *)
(* Exemple : 5 est premier. Ses 2 diviseurs sont : 1 et 5 *)

(* On doit définir une fonction, notée nbdiv(x, y), qui compte le nombre des diviseurs de x. Le paramètre y joue ici le rôle de diviseur éventuel (potentiel, possible) de x.
Dans la solution 1, on fait varier y de x à 1 (en sens décroissant).
Dans la solution 2, on fait varier y de 1 à x (en sens croissant). *)

Solution 1 : appel récursif décroissant nbdiv(x,y-1)

//On teste les diviseurs éventuels de x en faisant varier le paramètre y de x à 1 (en sens décroissant).

(* nbdiv(x,y) : compte les diviseurs de x en testant tous les diviseurs possibles de x à 1).*)

```
let rec nbdiv(x,y) = if (y = 0) then 0
                     else if (x mod y = 0) then 1 + nbdiv(x,y-1)
                                           else 0 + nbdiv(x,y-1);;
```

(* prem(n) : vérifie si n est premier. *)

```
let prem(n) = if nbdiv(n,n) = 2 then true else false;;
                  //L'appel nbdiv(n,n) démarre avec y=n.
```

Réponse de Caml :

nbdiv : int * int → int = <fun>
prem : int → bool = <fun>

Exemple : Déroulement de la fonction prem pour n=4

n=4 → prem(4) = if (nvdiv(4,4)=2) then true else false

L'appel de nbdiv(4,4) s'effectue comme suit :

nbdiv(4,4)=if (4=0) → else if (4 mod 4 = 0) → else 1 + nbdiv(4,3)
nbdiv(4,3)=if (3=0) → else if (4 mod 3 = 0) → then 0 + nbdiv(4,2)
nbdiv(4,2)=if (2=0) → else if (4 mod 2 = 0) → else 1 + nbdiv(4,1)
nbdiv(4,1)=if (1=0) → else if (4 mod 1 = 0) → then 1 + nbdiv(4,0)
nbdiv(4,0)=if (0=0) → then 0

Donc, résultat de la fonction nbdiv(4,4) est : nbdiv(4,4) = 1+0+1+1+0 = 3.

n=4 → prem(4) →= (nvdiv(4,4) = 3 ≠ 2) →false
//n=4 n'est pas un nombre premier (bien entendu !).

Solution 2 : appel récursif croissant nbdiv(x,y+1)

//On teste les diviseurs éventuels de x en faisant varier le paramètre y de 1 à x (en sens croissant).

(* nbdiv(x,y) : compte les diviseurs de x en testant tous les diviseurs possibles de 1 à x).*)

```
let rec nbdiv(x,y) = if (y = x+1) then 0
                     else if (x mod y = 0) then 1 + nbdiv(x,y+1)
                                           else  0 + nbdiv(x,y+1) ;;
```

```
(* prem(n) : vérifie si n est premier. *)
let prem(n) = if nbdiv(n, 1) = 2 then true else false ;;
                    //L'appel nbdiv(n,1) démarre avec y=1.
```

Réponse de Caml :

nbdiv : int * int → int = <fun>
prem : int → bool = <fun>

Exemple : Déroulement de la fonction prem pour n=4

n=4 → prem(4) = if (nvdiv(4,1)=2) then true else false

L'appel de nbdiv(4,1) s'effectue comme suit :

nbdiv(4,1)=if (1=5) → else if (4 mod 1 = 0) → then 1 + nbdiv(4,2)
nbdiv(4,2)=if (2=5) → else if (4 mod 2 = 0) → then 1 + nbdiv(4,3)
nbdiv(4,3)=if (3=5) → else if (4 mod 3 = 0) → else 0 + nbdiv(4,4)
nbdiv(4,4)=if (4=5) → else if (4 mod 4 = 0) → then 1 + nbdiv(4,4)
nbdiv(4,5)=if (5=5) → then 0

Donc, résultat de la fonction nbdiv(4,1) est : nbdiv(4,1) = 1+1+0+1+0 = 3.

n=4 → prem(4) →= (nvdiv(4,1) = 3 ≠ 2) →false
//n=4 n'est pas un nombre premier (bien entendu !).