Récursion et programmation fonctionnelle

INFO1 - Semaine 36

Guillaume CONNAN

IUT de Nantes - Dpt d'informatique

Dernière mise à jour : 2 septembre 2012

Sommaire

Sommaire

```
let mvt depart arrivee=
print_string
("Déplace un disque de la tige "^depart^" vers la tige "^
    arrivee);
print_newline();;
```

mouvement élémentaire

```
let rec hanoi a b c= function
   | 0 -> ()  (*0 disque : on ne fait rien*)
   | n -> hanoi a c b (n-1); (*n-1 disques sont déplacés de a
       vers b*)
          mvt a c; (*on déplace le disque restant en a vers c*)
          hanoi b a c (n-1) (*n-1 disques sont déplacés de b
              vers c*);;
```

résolution récursive du problème des tours de Hanoï

```
let mvt depart arrivee=
print_string
("Déplace un disque de la tige "^depart^" vers la tige "^
    arrivee);
print_newline();;
```

mouvement élémentaire

```
let rec hanoi a b c= function
   | 0 -> ()  (*0 disque : on ne fait rien*)
   | n -> hanoi a c b (n-1); (*n-1 disques sont déplacés de a
       vers b*)
          mvt a c; (*on déplace le disque restant en a vers c*)
          hanoi b a c (n-1) (*n-1 disques sont déplacés de b
              vers c*);;
```

résolution récursive du problème des tours de Hanoï

```
# hanoi "A" "B" "C" 3;;
Déplace un disque de la tige A vers la tige C
Déplace un disque de la tige A vers la tige B
Déplace un disque de la tige C vers la tige B
Déplace un disque de la tige A vers la tige C
Déplace un disque de la tige B vers la tige A
Déplace un disque de la tige B vers la tige C
Déplace un disque de la tige A vers la tige C
- : unit = ()
```

```
# hanoi "A" "B" "C" 4;;
Déplace un disque de la tige A vers la tige B
Déplace un disque de la tige A vers la tige C
Déplace un disque de la tige B vers la tige C
Déplace un disque de la tige A vers la tige B
Déplace un disque de la tige C vers la tige A
Déplace un disque de la tige C vers la tige B
Déplace un disque de la tige A vers la tige B
Déplace un disque de la tige A vers la tige C
Déplace un disque de la tige B vers la tige C
Déplace un disque de la tige B vers la tige A
Déplace un disque de la tige C vers la tige A
Déplace un disque de la tige B vers la tige C
Déplace un disque de la tige A vers la tige B
Déplace un disque de la tige A vers la tige C
Déplace un disque de la tige B vers la tige C
- : unit = ()
```

Sommaire

Sommaire

« *Soit f la fonction qui à un entier k associe l'entier naturel k^2. Calculer $f(2)$ et $f(2-3)$* ».

```
function F (X: in INTEGER) return INTEGER is
begin
   return X*X;
end F;
```

```
let f = fun x -> x*x ;;
```

```
function F (X: in INTEGER) return INTEGER is
begin
   return X*X;
end F;
```

```
let f = fun x -> x*x ;;
```

$$\text{puissance}(x, n) = \underbrace{x \times x \times \cdots \times x}_{n \text{ fois}}$$

$$\text{puissance} = [f \mid \forall x \ (x \in \mathbb{R} \wedge f(x,0) = 1) \wedge \forall x \forall n \ ((x,n) \in \mathbb{R} \times \mathbb{N} \wedge f(x,n+1) = x \times f(x,n)$$

Sommaire

```
2*(3+4)
a*(b+c)
```

```
2*(3+4)
a*(b+c)
```

```
let f = fun k -> k*k;;
```

```
val f : int -> int = <fun>
```

```
# let g = fun x -> x *. x;;
val g : float -> float = <fun>
```

```
# f(3);;
- : int = 9
```

```
# f(1.2);;
Characters 1-6:
  f(1.2);;
    ^^^^^
Error: This expression has type float but an expression was
       expected of type int
```

```
let f = fun k -> k*k;;
```

```
val f : int -> int = <fun>
```

```
# let g = fun x -> x *. x;;
val g : float -> float = <fun>
```

```
# f(3);;
- : int = 9
```

```
# f(1.2);;
Characters 1-6:
  f(1.2);;
    ^^^^^
Error: This expression has type float but an expression was
    expected of type int
```

```
let f = fun k -> k*k;;
```

```
val f : int -> int = <fun>
```

```
# let g = fun x -> x *. x;;
val g : float -> float = <fun>
```

```
# f(3);;
- : int = 9
```

```
# f(1.2);;
Characters 1-6:
  f(1.2);;
    ^^^^^
Error: This expression has type float but an expression was
    expected of type int
```

```
let f = fun k -> k*k;;
```

```
val f : int -> int = <fun>
```

```
# let g = fun x -> x *. x;;
val g : float -> float = <fun>
```

```
# f(3);;
- : int = 9
```

```
# f(1.2);;
Characters 1-6:
  f(1.2);;
  ^^^^^
Error: This expression has type float but an expression was
    expected of type int
```

```
let f = fun k -> k*k;;
```

```
val f : int -> int = <fun>
```

```
# let g = fun x -> x *. x;;
val g : float -> float = <fun>
```

```
# f(3);;
- : int = 9
```

```
# f(1.2);;
Characters 1-6:
  f(1.2);;
   ^^^^^
Error: This expression has type float but an expression was
    expected of type int
```

```
# let h = fun a b -> f (b-a);;
val h : int -> int -> int = <fun>
# h 7 3;;
- : int = 16
```

Comment a été calculé *h* 7 3 ?

```
# let h = fun a b -> f (b-a);;
val h : int -> int -> int = <fun>
# h 7 3;;
- : int = 16
```

Comment a été calculé *h* 7 3 ?

```
let hh a b =
  let ff k =
    fun k -> k*k
  in
  ff (b-a);;
```

Sommaire

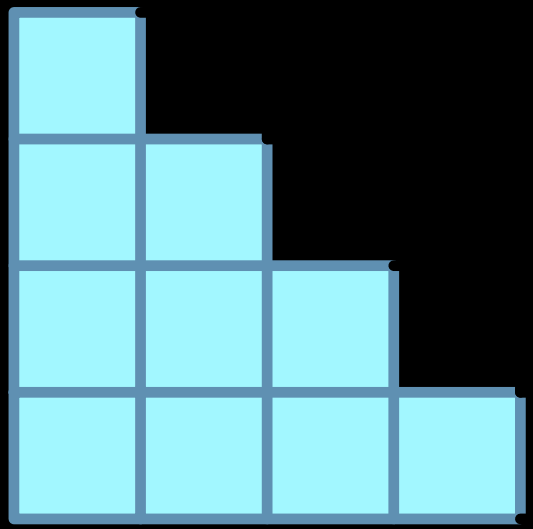
Sommaire

- induction
- induction mathématique (raisonnement par récurrence)
- fonction récursive, type récursif

- induction
- induction mathématique (raisonnement par récurrence)
- fonction récursive, type récursif

- induction
- induction mathématique (raisonnement par récurrence)
- fonction récursive, type récursif

Sommaire



$$n! = 1 \times 2 \times 3 \times \cdots \times (n-1) \times n$$

```
fac(n): si n=1 alors 1 sinon n*(fac (n-1))
```

$$n! = 1 \times 2 \times 3 \times \cdots \times (n-1) \times n$$

```
fac(n): si n=1 alors 1 sinon n*(fac (n-1))
```

Exercice 1

Pour la factorielle, on prouve par récurrence que `fac(n)` calcule bien $n!$ et que le nombre d'appels de la fonction `fac` engendrés par le calcul de `fac(n)` est égal à $n-1$: faites-le !

```
fac(n): si n=1 alors 1 sinon (fac (n+1))/(n+1)
```

```
let rec fac = function
  |1 -> 1
  |n -> n*fac(n-1);;

let rec fac n =
  if n = 1 then 1 else n*fac(n-1);;

let rec fac n =
  match n with
    |1 -> 1
    |_ -> n*fac (n-1);;

let fac n =
  let rec temp = function
    |1,acc -> acc
    |n,acc -> temp(n-1 , n*acc)
  in
  temp(n,1);;
```