Prise en main de CaML Light - 2

Lycée Thiers 2015

Fonctions d'affichage

- Afficher une chaîne de caractères :

```
print_string;;
- : string -> unit = <fun>
```

- Afficher une valeur entière :

```
print_int;;
- : int -> unit = <fun>
```

- Afficher une valeur décimale :

```
print_float;;
- : float -> unit = <fun>
```

- Effectuer un saut de ligne :

```
print_newline;;
- : unit -> unit = <fun>
```

Fonctions d'affichage

- Afficher une chaîne de caractères :

```
print_string "Hello, world !\n";;
Hello, world !
- : unit = ()
```

- Afficher une valeur entière :

```
print_int (4+1);;
5- : unit = ()
```

- Afficher une valeur décimale :

```
print_float (2. ** 10.);;
1024.0- : unit = ()
```

Fonctions d'affichage

- Attention au parenthèses !

```
print_float 2. ** 10.;;
Entrée interactive:
>print_float 2. ** 10.;;
>^^^^^^^^^^^^^^
Cette expression est de type unit,
mais est utilisée avec le type float.
```

- Déclencher un saut de ligne :

```
print_newline ();;

- : unit = ()
```

Valeur de retour / Valeur affichée

- Une fonction qui renvoie le carré d'un entier :

```
let f x = x * x;;
f : int -> int = <fun>
f 5;;
- : int = 25
```

- Une fonction qui affiche le carré d'un entier :

```
let g x =
  print_int (x * x);
  print_newline ()
;;
g : int -> unit = <fun>
g 5;;
25
- : unit = ()
```

Valeur de retour / Valeur affichée

- Ce qu'on peut faire avec `f` :

```
1 + f 5;;
- : int = 26
```

- Ce qu'on ne peut pas faire avec `g` :

```
1 + g 5;;
Entrée interactive:
>1 + g 5;;
>    ^^^
Cette expression est de type unit,
mais est utilisée avec le type int.
```

Aspects “impératifs” de CaML

1. **Références**

```
let x = ref valeur;
x := expr;;
!x;;
```

2. **Boucles**

```
for compteur = debut to fin do expr done

while (condition) do expr done
```

3. **Tests**

```
if (condition) then expr_1 else expr_2
```

Calcul de n !

```
let fact n =
  let f = ref 1 in
    for k = 1 to n do
      f :=!f * k
    done;
   !f
;;
fact : int -> int = <fun>

fact 5;;
- : int = 120

fact 0;;
- : int = 1
```

Une suite vérifiant une RR_1

$$u_0 = 0; \qquad \forall n \in \mathbb{N},\ u_{n+1} = \frac{1}{1 + u_n}$$

```
let u n =
  let t = ref 0. in
    for k = 1 to n do
      t := 1. /. (1. +. !t)
    done;
   !t
;;
u : int -> float = <fun>

u 20;;
- : float = 0.618033985017
```

Une boucle conditionnelle

La suite $(u_n)_{n\in\mathbb{N}}$ converge vers $\ell = \dfrac{\sqrt{5}-1}{2}$.

Etant donné $e > 0$, calcul de $\min\{n \in \mathbb{N};\ |u_n - \ell| < e\}$

```
let rang e =
  let lim = (sqrt 5. -. 1.) /. 2. in
  let rg = ref 0 in
  let t = ref 0. in
    while abs_float (!t -. lim) >= e do
      t := 1. /. (1. +. !t);
      rg := 1 + !rg
    done;
   !rg
;;
rang : float -> int = <fun>
```

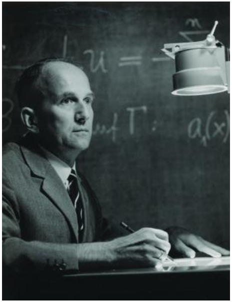
Une curieuse suite d'entiers

$$u_0 = s; \qquad \forall n \in \mathbb{N},\ u_{n+1} = \begin{cases} \dfrac{u_n}{2} & \text{si } u_n \text{ est pair} \\ 3u_n + 1 & \text{sinon} \end{cases}$$

$$17 \to 52 \to 26 \to 13 \to 40 \to 20 \to$$

$$10 \to 5 \to 16 \to 8 \to 4 \to 2 \to 1$$

Pour la petite histoire



Lothar Collatz (1910 - 1990)
mathématicien allemand

Conjecture (formulée vers 1937) : pour tout $s \in \mathbb{N}^\star$, la suite $(u_n)_{n\in\mathbb{N}}$ atteint inéluctablement le cycle $[4, 2, 1]$.

Vérification jusqu'à $5,764 \times 10^{18}$

Tomas OLIVEIRA E SILVA, 2008.

La suite de Syracuse en Caml

```
let syracuse s =
  let u = ref s in
    while !u <> 1 do
      print_int !u;
      print_string " ";
      u := if !u mod 2 = 0
           then !u / 2
           else 3 * !u + 1
    done;
    print_string “1\n”
;;

syracuse 13;;
13 40 20 10 5 16 8 4 2 1
 - : unit = ()
```

Test élémentaire de primalité

Soit $n \in \mathbb{N} - \{0, 1\}$.

Définition

n est dit premier lorsque :

$$\forall (p, q) \in \mathbb{N}^{\star 2},\ n = pq \Rightarrow (p = 1 \text{ ou } q = 1)$$

$$2, 3, 5, 7, 11, 13, 17, \cdots, 10007, \cdots, 2^{57\,885\,161} - 1, \cdots$$

Test élémentaire de primalité

Théorème

Si n ne possède aucun diviseur d tel que $2 \leqslant d \leqslant \sqrt{n}$, alors n est premier.

Démonstration.

Si n n'est pas premier, alors :

$$\exists\,(p, q) \in \mathbb{N}^{\star 2};\ n = pq, \quad \text{et} \quad p \geqslant 2, \quad \text{et} \quad q \geqslant 2$$

Mais alors $p > \sqrt{n}$ et $q > \sqrt{n}$ donc $pq > n$ □

Remarque :

si $n \geqslant 2$, alors le plus petit diviseur > 1 de n est son PPFP.

Test élémentaire de primalité

Algorithme

- Entrée : un entier $n \geqslant 2$
- Boucle de recherche d'un éventuel diviseur d parmi les entiers $2, \cdots, \left\lfloor \sqrt{n} \right\rfloor$
- Sortie : le plus petit facteur premier de n

Remarque :

$$(n \text{ est premier}) \Leftrightarrow (n = ppfp\,(n))$$

Un exemple pas à pas ...

$n = 391$ est-il premier ?

$\lfloor \sqrt{n} \rfloor = 19$

d	2	3	5	7	11	13	17	19
$d \mid n$?	N	N	N	N	N	N	Y	

391 est composé.

Un autre exemple ...

$n = 419$ est-il premier ?

$\lfloor \sqrt{n} \rfloor = 20$

d	2	3	5	7	11	13	17	19
$d \mid n$?	N	N	N	N	N	N	N	N

419 est premier.

Test élémentaire de primalité

```
let ppfp n =
  if n <= 1 then failwith "argument incorrect"
  else if n mod 2 = 0 then 2
  else
    let d = ref 3 in
    let encore = ref true in
      while (!d * !d <= n & !encore) do
        if n mod !d = 0 then (
          encore := false
        )
        else d := !d + 2
      done;
      if !encore then n else !d
;;
```

Ce test est-il performant ?

Si $n \simeq 10^{100}$, alors $\left\lfloor \sqrt{n} \right\rfloor \simeq 10^{50}$,

d'où environ 5×10^{49} tests de divisibilité.

A raison de $10^{-6} s$ par test, il faut environ $5 \times 10^{43} s$,

c'est-à-dire :

$$1,6 \times 10^{36} \text{ années ...}$$

The End !