(Séquence 3.3

puissance

Définitions de la puissance

Définition **récursive** de x puissance n :

$$\begin{aligned} x^0 &= 1 \\ x^n &= x * x^{n-1} \qquad \text{pour} \qquad n \geq 1 \end{aligned}$$

Définitions de la puissance

Définition **récursive** de x puissance n :

$$\begin{aligned} x^0 &= 1 \\ x^n &= x * x^{n-1} \qquad \text{pour} \qquad n \geq 1 \end{aligned}$$

Une autre définition récursive de la puissance :

$$\begin{aligned} x^0 &= 1 \qquad \text{pour} \qquad n = 0 \\ x^{2n} &= (x^n)^2 \qquad \text{pour} \qquad n \geq 1 \\ x^{2n+1} &= x * x^{2n} \qquad \text{pour} \qquad n \geq 1 \end{aligned}$$

mais on peut aussi écrire :

$$x^{2n} = (x^2)^n \qquad \text{pour} \qquad n \geq 1$$

La fonction `puissance`

Définition **récursive** de x puissance n :

$$\begin{aligned} x^0 &= 1 \\ x^n &= x * x^{n-1} \qquad \text{pour} \qquad n \geq 1 \end{aligned}$$

```
;;; puissance : Nombre * nat -> Nombre
;;; (puissance x n) rend x a la puissance n
(define (puissance x n)
  (if (> n 0)
      (* x (puissance x (- n 1)))
      1 ) )
```

Lenteur ?

Une autre définition de la puissance calquée sur la définition :

$$\begin{aligned} x^0 &= 1 \quad \text{pour} \quad n = 0 \\ x^{2n} &= (x^n)^2 \quad \text{pour} \quad n \geq 1 \\ x^{2n+1} &= x * x^{2n} \quad \text{pour} \quad n \geq 1 \end{aligned}$$

```
;;; puissanceLent : Nombre * nat -> Nombre
;;; HYPOTHESE n est un entier positif ou nul
;;; (puissanceLent x n) rend x a la puissance n
(define (puissanceLent x n)
  (if (= n 0)
      1
      (if (even? n)
          (* (puissanceLent x (quotient n 2))
             (puissanceLent x (quotient n 2)))
          (* x
             (puissanceLent x (quotient n 2))
             (puissanceLent x (quotient n 2))))))
```

Puissance

Le calcul de $x^{n/2}$ est fait 2 fois à chaque appel récursif

```
|(puissanceLent 2 6)
| (puissanceLent 2 3)
| |(puissanceLent 2 1)
| | (puissanceLent 2 0)
| | 1
| | (puissanceLent 2 0)
| | 1
| |2
| |(puissanceLent 2 1)
| | (puissanceLent 2 0)
| | 1
| | (puissanceLent 2 0)
| | 1
| |2
| 8
```

suite↗

```
| (puissanceLent 2 3)
| |(puissanceLent 2 1)
| | (puissanceLent 2 0)
| | 1
| | (puissanceLent 2 0)
| | 1
| |2
| |(puissanceLent 2 1)
| | (puissanceLent 2 0)
| | 1
| | (puissanceLent 2 0)
| | 1
| |2
| 8
|64
```

Un autre essai de définition de la puissance en deux fonctions :

```
;;; carre : Nombre -> Nombre
;;; (carre y) rend le carre de y
(define (carre y)
  (* y y) )
```

```
;; puissanceBis a la meme specification que puissance
(define (puissanceBis x n)
  (if (= n 0)
      1
      (if (even? n)
          (carre (puissanceBis x (quotient n  2)))
          (* (carre (puissanceBis x (quotient n 2)))
             x ) ) ) )
```

et trace de `puissanceBis`

```
|(puissanceBis 2 6)
| (puissanceBis 2 3)
| |(puissanceBis 2 1)
| | (puissanceBis 2 0)
| | 1
| | (carre 1)
| | 1
| |2
| |(carre 2)
| |4
| 8
| (carre 8)
| 64
|64
```

Et, stylistiquement, avec une fonction interne

```
;;; puissanceBisBis : Nombre * nat ->  Nombre
;;; (puissanceBisBis x n) rend x a la puissance n
(define (puissanceBisBis x n)
  ;; carre : Nombre -> Nombre
  ;; (carre y) rend le carre de y
  (define (carre y)
    (* y y) )
  (if (= n 0)
      1
      (if (even? n)
          (carre (puissanceBisBis
                  x (quotient n  2)))
          (* x (carre (puissanceBisBis
                       x (quotient n 2)))))))
```

Puissance encore

L'imbrication d'alternative consomme de la marge gauche et des parenthèses. On peut aussi utiliser une nouvelle forme spéciale `cond` et écrire :

```
;;; puissance : Nombre * nat ->  Nombre
;;; (puissance x n) rend x a la puissance n
(define (puissance x n)
  ;; carre : Nombre -> Nombre
  ;; (carre y) rend le carre de y
  (define (carre y)
    (* y y) )
  (cond
   ((= n 0) 1)
   ((even? n)
    (carre (puissance x (quotient n  2))) )
   (else
    (* x (carre (puissance x (quotient n 2)))))) ) )
```

Puissance

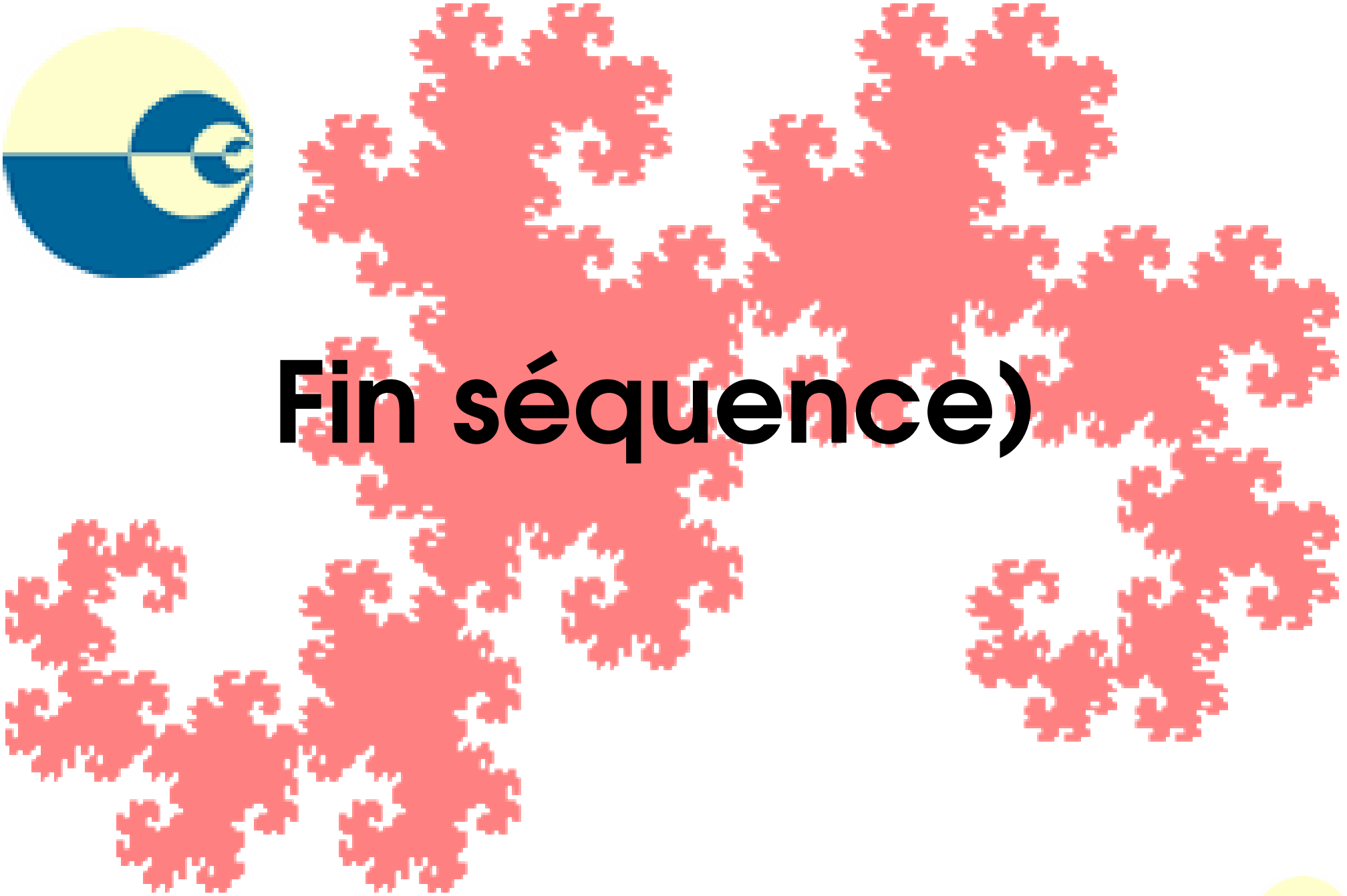
Et encore une autre définition où l'on met en facteur le calcul de $x^{n/2}$

```
(define (puissanceTer x n)
  (if (= n 0)
      1
      (let ((P (puissanceTer x (quotient n  2))))
        (if (even? n)
            (* P P)
            (* x P P) ) ) ) )
```

```
|(puissanceTer 2 6)
| (puissanceTer 2 3)
| |(puissanceTer 2 1)
| | (puissanceTer 2 0)
| | 1
| |2
| 8
|64
```

Complexité de la puissance

- La ***complexité*** s'intéresse au nombre d'opérations qu'il faut effectuer pour un calcul.
- Pour la puissance, la mesure sera le nombre de multiplications à réaliser.
- Avec `puissanceTer`, on divise par deux, à chaque fois, l'argument. Si l'on part d'un entier n, il faut environ $log_2(n)$ divisions par deux pour l'amener à 1 car $n = 2^{log_2(n)}$.
- Il faut donc environ 20 multiplications pour élever un nombre à la puissance 1000 (en fait 17). Il en faut 19 pour élever un nombre à la puissance 959 mais seulement 14 pour 960.
- Comparer avec les 1000 multiplications que nécessite le premier schéma récursif naïf.



Fin séquence)