(Séquence 6.8

Récursion sur arbres binaires

Récursion sur les arbres binaires

Un arbre binaire est :

- soit vide
- soit constitué d'une étiquette, d'un sous-arbre gauche et d'un sous-arbre droit

```
;;; fRec: ArbreBinaire[alpha] -> ...
(define (fRec B)
  (if (ab-noeud? B)
      (combinaison (ab-etiquette B)
                   (fRec (ab-gauche B))
                   (fRec (ab-droit B)))
      cas-arbre-vide ) )
```

Nombre de noeuds

```
;;; nombre-noeuds: ArbreBinaire[α] -> nat
;;; (nombre-noeuds B) rend le nombre de noeuds de B
(define (nombre-noeuds B)
  (if (ab-noeud? B)
      (+ 1
         (nombre-noeuds (ab-gauche B))
         (nombre-noeuds (ab-droit B)))
      0))
```

Somme des étiquettes

```
;;; somme-arbre:  ArbreBinaire[Nombre] -> Nombre
;;; (somme-arbre B) rend la somme des étiquettes de B
(define (somme-arbre B)
  (if (ab-noeud? B)
      (+ (ab-etiquette B)
         (somme-arbre (ab-gauche B))
         (somme-arbre (ab-droit B)))
      0))
```

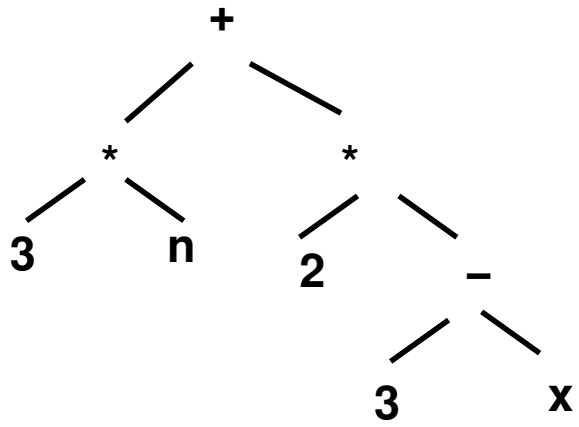
Profondeur

Définition récursive de la profondeur d'un arbre :

- la profondeur de l'arbre vide est 0,
- la profondeur d'un arbre non vide est égale au maximum des profondeurs de ses sous-arbres immédiats, augmenté de 1.

```
;;; profondeur:  ArbreBinaire[α] -> nat
;;; (profondeur B) rend la profondeur de B
(define (profondeur B)
  (if (ab-noeud? B)
      (+ 1
         (max (profondeur (ab-gauche B))
              (profondeur (ab-droit B))))
      0))
```

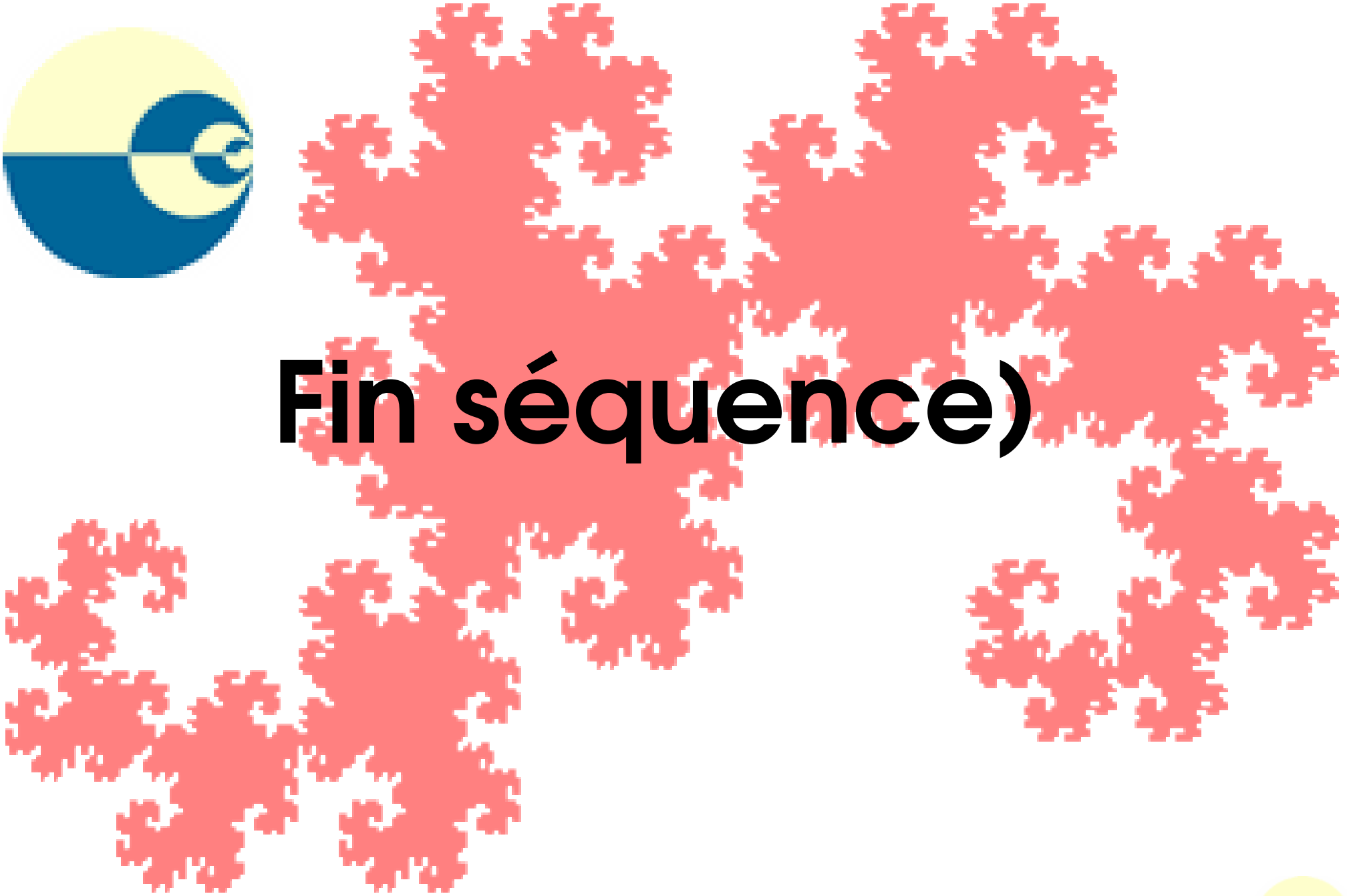
Différentes écritures d'une exp. arithm.



- infixe parenthésée : `((3 * n) + (2 * (3 - x)))`
- préfixe Scheme : `(+ (* 3 n) (* 2 (- 3 x)))`
- polonaise préfixe : `+ * 3 n * 2 - 3 x`
- liste préfixe : `(+ * 3 n * 2 - 3 x)`
- polonaise postfixe : `3 n * 2 3 x - * +`

Liste préfixe

```
;;; liste-pref:  ArbreBinaire[α] -> LISTE[α]
;;; (liste-pref B) rend la liste préfixée de B
(define (liste-pref B)
  (if (ab-noeud? B)
      (cons (ab-etiquette B)
            (append (liste-pref (ab-gauche B))
                    (liste-pref (ab-droit B))))
      (list)))
```



Fin séquence)