INTRODUCTION : 1ere approche

Structure de base : la liste

- Une liste est une séquence ordonnée, éventuellement vide, d'atomes ou de listes, précédée par une parenthèse ouvrante et suivie d'une parenthèse fermante.
- Exemples : (* 2 3) (ceci est une liste)

INTRODUCTION : 1ere approche

Une expression est soit un atome soit une liste d'expression.

- La boucle d'interaction correspond à un cycle, elle est composée de 3 étapes bien distinctes :
- Read : lecture d'une expression
- Eval : évaluation de l'expression
- Print : impression du résultat du calcul dans la fenêtre d'interaction
- Dès qu'un cycle est terminé, l'invite (>) réapparaît, l'environnement lisp est prêt pour une autre interaction.

INTRODUCTION : 1ere approche

La valeur d'un nombre est ce nombre luimême

La valeur d'un symbole est :

- Prédéfinie et non modifiable : le symbole est alors une des 2 constantes Lisp : 't' ou 'nil'
- Définie et modifiée dynamiquement par l'utilisateur ou par programme : le symbole est une variable. La valeur d'un nombre est ce nombre lui-même

INTRODUCTION : 1ere approche

Soit la liste (s0 s1 s2 … sN)

- LISP considère que s0 est un symbole qui désigne un nom de fonction ; s0 n'est pas évalué.
- LISP évalue les autres éléments.
- LISP applique la fonction associée à s0 aux arguments évalués et retourne comme résultat la valeur de cette application.
- Exemple

```
> (+ 1 2)
3
```

INTRODUCTION : 1ere approche

Langage classique

- Ensemble fixe de variables
- Gamme de valeurs fixée par le type : nombres

entiers, chaîne de caractères, etc.

En Lisp

- Symboles crées et détruits dynamiquement : arguments de fonctions, symboles locaux,symboles globaux
- Les valeurs sont également des symboles

INTRODUCTION : 1ere approche

L'environnement Lisp contient un certain nombre de fonctions prédéfinies : les primitives.

scheme comporte environ un millier de primitives. En particulier, on trouve les fonctions (comme celles opérants sur Les nombres), les opérateurs spéciaux ou formes spéciales (define, quote, setf, …)

INTRODUCTION : quote

Exemple :

> a

⇒ Erreur Le symbole a n'est pas connu a priori de Lisp, il n'a pas de valeur

- La forme spéciale QUOTE permet de bloquer l'évaluation de son argument.
- Abréviation de QUOTE : apostrophe
- Exemple :

> (quote a)

A

> 'a

A

>

INTRODUCTION : quote

La fonction *set*

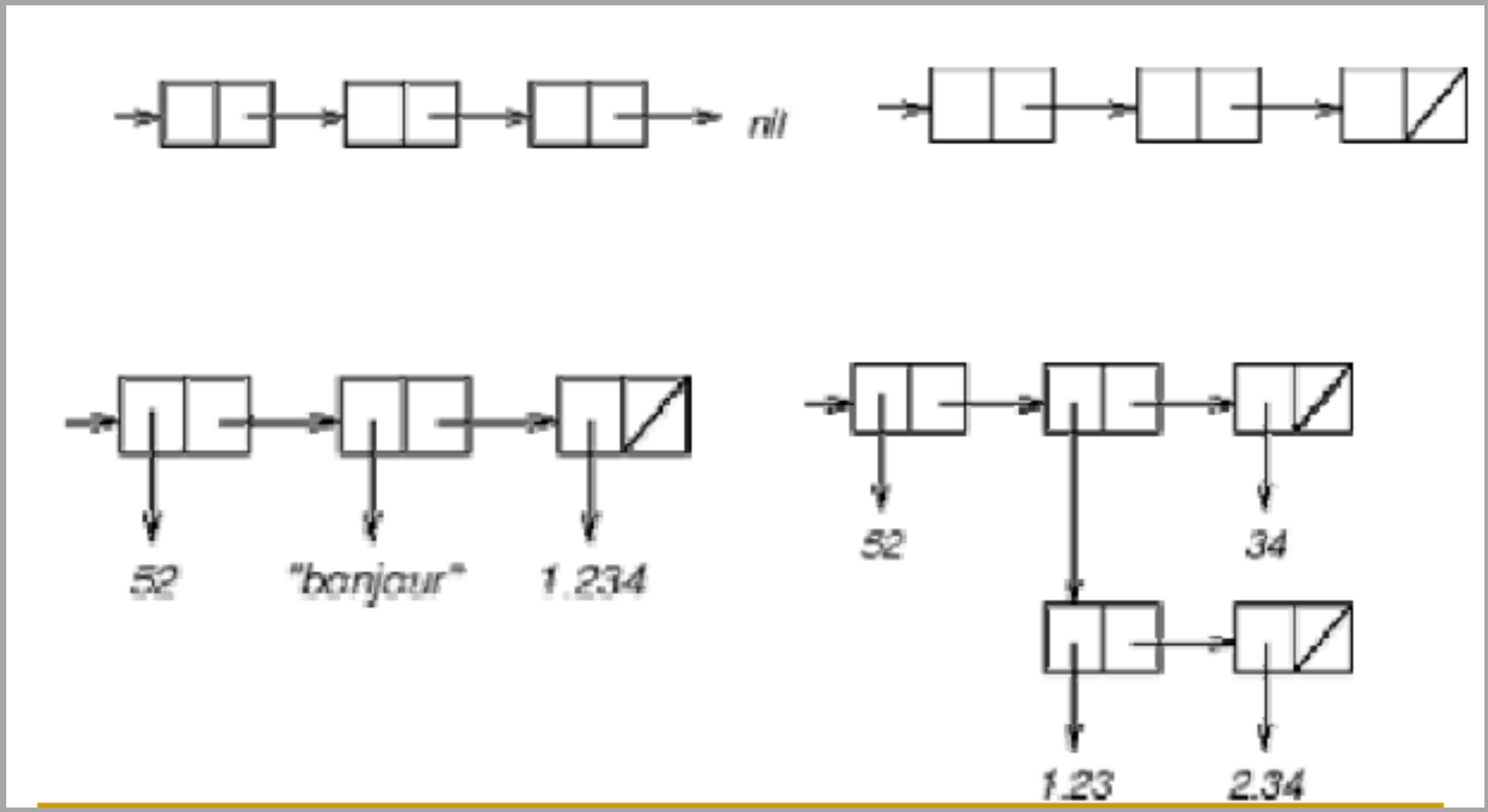
- (set a 12)
- Processus d'évaluation d'une liste : le 1er argument est
- Évalué
- => Même erreur que précédemment
- Solution : (set 'a 12)
- La fonction *setq*
- Combinaison de la fonction *set et de la forme* spéciale *quote*
- (setq a 2) ≡ (set 'a 2) ≡ (set (quote a) 2)

INTRODUCTION : Liste

Une liste en Lisp est représentée par une liste simplement chaînée.

- Une liste est construite à partir de paires constituées de deux cellules (pointeurs) :
  - La première cellule (*car) contient le premier* Élément
  - La deuxième cellule (*cdr) contient une autre liste* potentiellement vide
- Pour indiquer la fin de la liste, le *cdr de la dernière* paire est le symbole *nil*

INTRODUCTION : exemples



INTRODUCTION : Liste

La fonction *cons à deux arguments x et y :*

retourne la paire dont le car est x et le cdr y

- (cons 'a '(b c)) -> (a b c)
- (cons 'a nil) -> (a)
- La fonction *list admet un nombre arbitraire* d'arguments :

retourne une liste constituée de ces éléments

- (list 1 2 3 4) -> (1 2 3 4)

INTRODUCTION : Liste

La fonction *first (ou car) retourne le premier élément* de la liste :

- (first '(a b c)) -> a
- (first '((a) (b c))) -> (a)
- La fonction *rest (ou cdr) retourne le reste de la liste :*
- (rest '(a b c)) -> (b c)
- (rest '((a) (b c))) -> ((b c))
- La fonction *last retourne le dernier élément de la l*iste :
- (last '(a b e)) -> (e)

INTRODUCTION : Liste

Les fonctions *car et cdr peuvent être* combinées et peuvent faire l'objet d'abréviations jusqu'à 4 niveaux.

- Dans le schéma C*R, * représente une chaîne d'au plus quatre éléments de *a ou d.*
- (cadr '(a b c)) ≡ (car (cdr '(a b c)))
- (cdddr '(a b c d)) ≡ (cdr (cdr (cdr '(a b c d))))
- (cadar '((a b c) d)) ≡ (car (cdr (car '((a b c) d))))

INTRODUCTION : Liste

(append l1 l2 l3) retourne une liste constituée des éléments de l1, l2 et l3 dans l'ordre donné.

- (member x l) si x est un des éléments de l, retourne la liste l à partir de la première occurrence de x.
- (length l) retourne le nombre d'éléments de la liste l.