TD numéro 2 : listes

À préparer avant la séance

• *Pour chaque expression Scheme écrite ci-dessous, réfléchissez au résultat qu'elle devrait donner. Vous pouvez vérifier avec Racket le résultat, et nous discuterons en TD des cas qui posent question.*

- (car (cdr '(a b c d)))
- (car (cdr '(abc d)))
- (cdr '(a (b c d)))
- (cdr '((a b c d)))
- (cdr (car (cdr '(a (b c) (d e)))))
- (cons '(a b) '(c d))
- (cons 'a (cons 'b '(c d)))
- (cons '(a b) 'c)
- (list? (+ 2 3))
- (list? '(+ 2 3))

• Définir en Scheme :

- une fonction qui retourne le second élément d'une liste d'au moins deux éléments ;

- une fonction qui calcule la longueur d'une liste (i.e. son nombre d'éléments).

À faire pendant la séance

• Définir en Scheme :

- une fonction qui retourne vrai si et seulement si une liste n'a qu'un seul élément ;

- une fonction qui prend une liste et un élément et retourne vrai si et seulement si l'élément appartient à la liste ;

- une fonction qui retourne le $n^{ième}$ élément d'une liste ;

- une fonction qui insère un élément dans une liste après le $i^{ème}$ élément ;

- une fonction qui retourne le dernier élément d'une liste non vide.

• Donner la spécification de la fonction mystère ci-dessous :

```
(define mystere
  (lambda (x l)
    (cond ((null? l) 0)
          ((equal? x (car l)) (+ 1 (mystere x (cdr l))))
          (else (mystere x (cdr l))))))
```