

TP numéro 1

Connectez-vous à l'aide de votre numéro d'étudiant. Vous pouvez lancer DrRacket via un raccourci sur le bureau (ou via le menu *Démarrer>Programmes>PLT Scheme*).

Configuration de DrRacket :

Choisissez *Interagir avec DrRacket en Français*, puis *Accepter et Quitter* et relancez DrRacket. Ne prenez pas la peine de chercher une version plus récente (*Suivant*). Vous devez alors configurer le niveau de langage : choisissez *Niveau débutant avec abréviations pour les listes*, puis dans *Montrer les détails*, choisissez *write pour syntaxe de sortie* (si vous ne voyez pas le bouton Fin, cachez les détails).

La partie inférieure de la fenêtre vous permet d'évaluer des expressions. Vous pouvez l'utiliser pour effectuer la première partie du TP.

La partie supérieure de la fenêtre vous permet de définir des fonctions que vous devez enregistrer (bouton *Sauvegarder*) sur disque (utilisez votre compte sur W). L'extension ajoutée est *.scm*. Prenez l'habitude de créer un fichier par TP.

Le bouton *Vérifier* vous permet de vérifier la syntaxe de votre fonction. Le bouton *Exécuter* compile la fonction, pour que vous puissiez ensuite la tester dans la partie inférieure de la fenêtre. Le bouton *Stopper* vous permet d'interrompre l'exécution d'une fonction. Le bouton *Step* vous permet de dérouler pas à pas l'exécution d'une fonction.

1 Évaluation d'expressions

Avant de faire évaluer ces expressions par Scheme, essayez de prédire la réponse, et de l'expliquer.

(+ 3 5)	(define alpha 10)
(+ 2 3 4 5 6)	(symbol? alpha)
(+ (* 2 3) (- 1 2) (+ 3 (* 2 3)))	(symbol? 'alpha)
(if (< 1 2) 1 2)	(symbol? "alpha")

Imaginez d'autres expressions simples et testez-les!

2 Premières fonctions

- Écrire une fonction qui calcule le carré d'un nombre.
- Écrire une fonction qui retourne la racine carrée d'un nombre s'il est positif et retourne 0 sinon.
- Écrire une fonction réalisant le ou-exclusif de deux booléens.

X	Y	X OU-ex Y
V	V	F
V	F	V
F	V	V
F	F	F

- Écrire une fonction *somme* qui calcule récursivement la somme des n premiers entiers. Dérouler (somme 3) pas à pas.

3 Évaluation d'expressions sur les listes

(car (1 2 3 4))	(cddddr '(1 2 3 4))	(cons 1 '(2 3 4))	(cons alpha '(20 30))
(car '(1 2 3 4))	(cdadr '(1 2 3 4))	(cons (1) '(2 3 4))	(cons 'alpha '(20 30))
(cdr '(1 2 3 4))	(cdadr '(1 (2 3) 4))	(cons '(1) '(2 3 4))	
(cadr '(1 2 3 4))		(cons '1 '(2 3 4))	

4 Exercices sur les listes

- D'après le modèle donné, construisez les expressions suivantes en utilisant exclusivement la fonction *cons*.

Par exemple :

(cons 'paon (cons (cons 'poule '()) '())) =>
(paon (poule))

... => (() () poule)

... => (oie poule (poule) paon)

... => (poule oie poule ((poulet)))

... => (((poule) oie) paon)

- Quelle expression faut-il évaluer pour extraire *oie* des expressions suivantes ?

Par exemple, pour l'extraire de :

((poule oie pie) paon aigle), c'est cadar.

'(aigle (paon (pie oie)))

'(((oie) poule))

'(poule paon (oie) aigle)

'(paon (poule (poulet (oie))))

5 Fonctions sur les listes

- Écrivez une fonction qui prend une liste d'au moins deux éléments en argument et rend la liste équivalente où les deux premiers éléments ont été échangés.

(echange '(a b c d)) -> (b a c d)

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

(dernier '(a b c d)) -> d

- Écrivez une fonction qui renvoie le calcul de la factorielle d'un nombre positif.
- Écrivez une fonction qui prend une liste de nombres en argument et qui renvoie son maximum.
- Écrivez une fonction qui teste si un nombre est impair.
- Écrivez une fonction qui prend une liste en argument et qui renvoie cette liste privée de son dernier élément.
- Écrivez une fonction qui prend une liste en argument et rend la liste privée d'un élément sur deux.

(elimine '(a b c d e f)) -> (a c e)

- Écrivez une fonction qui prend une liste en argument et qui répète chacun de ses éléments.

(repete '(a b c d e)) -> (a a b b c c d d e e)