mbScheme

1°) Introduction

Lisp est une famille de langages de programmation inventée par John McCarthy en 1958 au Massachusetts Institute of Technology (MIT) dont Common Lisp est le descendant le plus connu. Le langage de programmation Scheme est un langage de programmation dérivé du langage Lisp, créé dans les années 1970 au MIT par Gerald Jay Sussman et Guy L. Steele. Parmi les langages dérivés de Scheme le langage Racket est l'un des plus importants en raison de ses outils spécifiques. Wikipedia fournit très facilement sur Internet des pages d'informations pour Lisp, Scheme et Racket aux adresses suivantes : https://fr.wikipedia.org/wiki/Lisp, https://fr.wikipedia.org/wiki/Scheme et https://fr.wikipedia.org/wiki/Racket_(langage). On trouve aussi sur Internet divers interpréteurs Scheme simples ou évolués et différents tutoriels. mbScheme est un interpréteur du langage Scheme dont le programme exécutable est programmé en langage Rust par M. Martin Billinger à https://github.com/mbillingr/interpreter. Ce programme sous Windows, interpreter.exe, fournit les fonctions natives utilisables. mbScheme comporte en plus une suite de fonctions complémentaires qui sont définies par l'utilisateur et qui fournissent d'autres fonctions du langage Scheme.

2°) Les fonctions natives

Pour utiliser seulement les fonctions natives le fichier interpreter.bat permet ce choix éventuel. La liste des fonctions natives est obtenue avec l'instruction : (print-env).

Les entités élémentaires, appelées atomes, qui sont utilisables sont les suivantes.

- les constantes booléennes : #t et #f

- les nombres entiers, exemples : 123, -456, 9876543210

- les nombres rationnels, exemples : 7/5, -41/101, 852369741/987456321

- les nombres réels, exemples : 3.14, -852.258

- les nombres complexes, exemples : 0+i, 1+2i, 1-2i, -3.4+1.2i

- les caractères, exemples : #\A, #\a, #\space, #\+, #\", #\!

- les chaines de caractères, exemples : "", "Bonjour le Monde !", "c'est exact"

Les nombres entiers et les nombres rationnels sont sans limite, ils ont la longueur aussi grande que nécessaire et ils permettent d'effectuer des calculs arithmétiques mathématiquement exacts. Les entiers sont gérés en tant qu'éléments des rationnels, les rationnels en tant qu'éléments des réels et les réels en tant qu'éléments des complexes. Les nombres réels ont une précision limitée, ils sont mémorisés en flottants sur 64 bits : la mantisse a environ l'équivalent de 16 chiffres décimaux au plus et l'exposant est compris entre -308 et +308, c'est pourquoi il y a aussi les quatre nombres réels particuliers : +inf.0, -inf.0, +nan.0 et -nan.0.

Il y a deux entités composées. La liste est l'entité composée qui est habituellement utilisée, exemples : (1 2 3 4), ("j'ai" 2 "tutoriels"). Le vecteur, exemple : [point 1 2], est utilisé à titre distinctif pour montrer un emploi particulier.

Les fonctions natives qui sont utilisables comprennent principalement les suivantes.

- les prédicats : boolean?, integer?, rational?, real?, complex?, char?, string?, string=?, vector?, number?, procedure?, null? (liste vide), pair? (liste non vide), exact? (précision illimitée), eq?, eqv?, equal?, symbol? (un nom valide pour une variable), file?.

- les opérateurs numériques : +, -, *, /, <, =, >.

- les opérateurs des listes : car, cdr, cons.

- les convertisseurs : number->string, list->string, symbol->string, string->number, string->list, string->symbol, object->class.

- les fonctions algébriques : min, max, numerator, denominator, quotient, remainder, round, floor, sin, cos, atan, log, make-rectangular, make-polar, magnitude, angle.

- les mots du langage : include, define, list, make-vector, quote, lambda, if, not, and, or, cond, case, begin, apply, display, read, file-open, fdisplay, newline, file-read, file-close!, let, set!, vector-set!, vector-length, print-env, help, exit.

3°) Les fonctions complémentaires

Le fichier mbScheme.bat permet d'utiliser directement à la fois les fonctions natives et les fonctions complémentaires qui sont disponibles dans le fichier mbScheme.scm. Ce qui ajoute les fonctions suivantes.

- les prédicats : atom?, belongs?, even? odd?, point?, prime?.

- les opérateurs numériques : %, <=, >=.

- les opérateurs des listes : add, append, cadr, cddr, enclose, head, last, length, map, nth, rank, remove, replace, reverse, tail, truncate.

- les convertisseurs : rational->integer, rational->real.

- les fonctions algébriques : abs, doble, exp, fact, fib, gcd, next-prime, pi, sign, sqrt, square.

- les mots du langage : new-line, open, opaque-vector, vector.

- des exemples de procédures spécifiques : check-point, make-point, point-x, point-x-set!, point-y, point-y-set!, translate.