Projet info

Créer script qui marchent et ne marchent pas (pour tester si les erreurs sont prises en compte.

Scheme commands :

car = 1er élément de la liste ; cdr = reste de la liste

Accepter sur une ligne, multiligne une seule expression ; refuser deux expressions sur la même ligne

Il doit reconnaître les atomes du langage, symbole, listes, construire en mémoire, …

Pas d’évaluation à l’incrément 1

BOUCLE INFINIE (

Char\* chaine = Saisie s-expression

Expression en Mémoire = analyse chaine construction s-expression en mémoire

(arbre)

SI ERREUR

Warning (message) ou erreur (message et arrêt du prog)

SINON

Résultat= évaluer expression en mémoire

SI ERREUR

Warning (message) …

SINON

Afficher résultat

FIN SI

FIN SI

)

s-expression valide : un atome ou une liste

atome = un identifier/symbole, un int, un char, un booléen, (), une chaîne

liste = ( s-expr2 {s-expr}\* )

identifier : lettre ou caractère particulier, +, - et … (voir R5RS)

\p = tabulation

L’état INIT, tant qu’on a un espace, on reste à INIT

Intdigit

Isspace

Int read\_atome(char \*chaine, int \*pB, object \*objetlu)

Object read\_atome(char\* chaine, int p) ;

ATOMES = char int, string

Parser\_atome(char\* chaine, int \*p\_pos

ETATs = Int, int en cours, char, char en cours

Etat = INIT

I=0 ;

While (i< strlen(chaine)) {

Switch(etat) {

Cas INIT :

Uf (isspace(chaine[i])) { i++

Etat=INIT ;

Break ;

}

if (isdigit(chaine[i]) || chaine[i]= « -  » || « + »)

…

…

case INT\_encours :

if (isdigit(chaine[i])) { i+ etat=init-en-cours ; break ;

if isspace[chaine[i]) { i++ \*p\_pas=i ; return INT ;

}

return ERROR ;

break ;

On peut inférer le type de l’atome qu’on va avoir en fonction du premier caractère

Readatome(…)

Regarde le premier carzctère

Si c’est un entier : readentier(…)

Si c’est un symoble : readsymbole(…)

Object read(char\* char, int\*ppos ,(object \*po) {

Si '('

Read\_pair(…)

Sinon

Read\_atome(…)

}

readpair (…) {

pair=make\_pair

object o = read …

make\_pair(o, …)

si')'

return pair

}

object read\_entier(char \*, int \*ppos) {

int valeur = strtol(…);

s'il reste des trucs , erreur

object o = make\_integer(valeur);

reutrn o;

}

make pair construit un objet qui représente une "pair"

\* Object : ne pas oublier la calloc

unsigned int type : on alloue pas tout, on va choisir dans l'objet le type

\*object make\_char( charc c) {

object ob;

ob= calloc (1, sizeof(\*ob2));

if (ob=NULL)

ob→type = SFS character:

ob→this.cjaracter =c

16/09/16 :

Décompression

tar –xvf <archive.tar>

tar –zxvf <archive.tgz ou.tar.gz>

object → eval → object

(define x (+ 5 9 )

(quote (+ 4 9 ))

(if (+ 4 9 ) (quote a ) '(quote b))

define

set!

Quote

And

Or

If

Object car (object o) {

If (o→type = pair) {

ERROR\_MSG()

}

Return o→this.pair.car

}