MANUAL LIPS

Comando adición:

```
CL-USER 1 > + 8 7
15

CL-USER 2 > + 5 3 4
12
```

Tipo de Datos:
- Átomos

Átomo: 9, T, nil, p

- Evaluar un átomo significa que se muestre el valor asignado del átomo.

9 → 9
p → Error (porque no está definido como variable).

```
CL-USER 3 > 9
9

CL-USER 4 > p

Error: The variable P is unbound.
  1 (continue) Try evaluating P again.
  2 Return the value of :P instead.
  3 Specify a value to use this time instead of evaluating P.
  4 Specify a value to set P to.
  5 Return the result of calling (P) instead.
  6 (abort) Return to level 0.
  7 Return to top loop level 0.

Type :b for backtrace, :c <option number> to proceed,  or :? for other options
```

- Asignar un valor un átomo usando setq y setp.

(setq p 9) → 9

```
CL-USER 6 : 2 > ( setq p 9 )
9
```

Reservados nil y T
nil → nil
T → T
Casos insensitivos
cDe y CDE son el mismo átomo

- Listas

Listas: (d), (- 7 8), (b (n l m) k), u , nil
nil puede ser átomo o lista.
nil = () = lista vacía
Evaluar una lista invoca una función
(+ 5 8)=13

```
CL-USER 11 : 3 > ( + 5 8 )
13
```

Funciones primitiva

Incluye operaciones: +, -, *, /, max, min, sqrt, …
También podemos aplicar una lista entera.
(cons 'a '(x y z)) → (a x y z)

```
CL-USER 11 : 3 > ( + 5 8 )
13
Error while reading: Unmatched right parenthesis.

CL-USER 12 : 3 >( cons 5 nil )
(5)

CL-USER 13 : 3 > ( cons 8  (cons 1 nil ) )
(8 1)

CL-USER 14 : 3 > ( cons 'a nil )
(A)
```

Se puede hacer una lista 'a, 'b, 'c, 'd sin usar una lista.

```
CL-USER 28 : 7 > list 'juan 'manuel ' maria
(JUAN MANUEL MARIA)

CL-USER 29 : 7 > list 'juan '(maunel maria )
(JUAN (MAUNEL MARIA))

CL-USER 30 : 7 > append '(juan manuel maria) ' (jose ricardo carlos)
(JUAN MANUEL MARIA JOSE RICARDO CARLOS)
```

Backquote y coma

A veces se desea invocar a una lista pero no a todos sus miembros:
(list ‘c ‘2 (/ my_weight 2) ‘f ‘9) → (c 2 2 6 f 9)

- El backquote y la coma nos ayuda a que evalué los valores que se encuentran después de la coma:

(a 1,(/ my_weight 2) b 4) Solo evaluará los valores que están después de
La coma

- Cuando la lista está formada como ingresamos a sus miembros:

- First and car me muestra el primer elemento de la lista

```
CL-USER 7 : 3 > first '(manuel jose)
MANUEL

CL-USER 8 : 3 > first '((manuel jose) ricardo maria)
(MANUEL JOSE)
```

```
CL-USER 9 : 3 > car '(manuel jose maria)
MANUEL
```

- Last and cdr me muestra la lista menos el primer elemento.

```
CL-USER 10 : 3 > last '(lapicero borrador cuaderno)
(CUADERNO)

CL-USER 11 : 3 > cdr '( lapicero borrador cuaderno)
(BORRADOR CUADERNO)
```

```
CL-USER 12 : 3 > rest '( mesa lapto computadora)
(LAPTO COMPUTADORA)

CL-USER 13 : 3 > rest '((lapto computadora) mesa television radio)
(MESA TELEVISION RADIO)
```

- Car and cdr pueden combinarse.

```
CL-USER 14 : 3 > caar '( (lapiz borrador) tajador cuaderno)
LAPIZ

CL-USER 15 : 3 > cdar  '( (lapiz borrador) tajador cuaderno)
(BORRADOR)
```

Nota: Trabaja de la derecha a la izquierda

Cambiando los valores de los átomos

Usamos el setq para cambiar los valores de los átomos
Ejemplos:

```
CL-USER 11 : 3 > setq L1 '(ana maria jose)
(ANA MARIA JOSE)

CL-USER 12 : 3 > setq L2 '((ana maria ) ricardo daniel jose)
((ANA MARIA) RICARDO DANIEL JOSE)

CL-USER 14 : 4 > cons 'pablo L1
(PABLO ANA MARIA JOSE)

CL-USER 15 : 4 > cons 'fiorella L2
(FIORELLA (ANA MARIA) RICARDO DANIEL JOSE)
```

```
CL-USER 16 : 4 > L1
(ANA MARIA JOSE)

CL-USER 17 : 4 >
L2
((ANA MARIA) RICARDO DANIEL JOSE)
```

Setf es una versión del setq que toma la función como primer argumento.
(setf (cadr L1) karla) → karla
L1 → (ANA KARLA JOSE)

Equivalencia

Existen dos tipos de equivalencias, por ejemplos:

```
CL-USER 21 : 6 > (eq 'manuel 'manuel)
T

CL-USER 22 : 6 > (eq 'fiorella 'cristian)
NIL

CL-USER 23 : 6 > (eq '(fiorella) '(cristian))
NIL

CL-USER 24 : 6 >  (eq '(fiorella) '(fiorella))
NIL
```

```
CL-USER 25 : 6 > (equal 'karla 'karla)
T

CL-USER 26 : 6 > (equal '(maria)'(maria))
T

CL-USER 27 : 6 > (equal '(or m n) ' (or h t))
NIL
```

Sets

Se pueden tratar a las listas como conjuntos (sin preservar el orden)

```
CL-USER 30 : 7 > setq L1 '(ana fiorella cristian pablo )
(ANA FIORELLA CRISTIAN PABLO)

CL-USER 31 : 7 > setq L2 '(fiorella pablo jose maria)
(FIORELLA PABLO JOSE MARIA)

CL-USER 32 : 7 > L1
(ANA FIORELLA CRISTIAN PABLO)

CL-USER 33 : 7 > L2
(FIORELLA PABLO JOSE MARIA)

CL-USER 34 : 7 > (union L1 L2)
(MARIA JOSE ANA FIORELLA CRISTIAN PABLO)
```

```
CL-USER 35 : 7 >( union '((ana) (maria))'((maria)))
((MARIA) (ANA) (MARIA))
```

(union '((ana) (maria)) '((ana)) : test #' equal) → ((ana) (maria))

La condición de prueba para determinar si dos elementos del conjunto son los mismos es la función de la igualdad.

```
CL-USER 13 : 4 > L1
(ANA FIORELLA CRISTIAN PABLO)
```

```
CL-USER 12 : 4 > L2
(FIORELLA PABLO JOSE MARIA)
```

```
CL-USER 10 : 4 > adjoin 'fiorella L1
(ANA FIORELLA CRISTIAN PABLO)
```

```
CL-USER 11 : 4 > set-difference L1 L2
(CRISTIAN ANA)
```

Más Funciones

```
CL-USER 15 : 4 > L1
(ANA FIORELLA CRISTIAN PABLO)

CL-USER 16 : 4 > L2
(FIORELLA PABLO JOSE MARIA)
```

```
CL-USER 14 : 4 > length L1
4
```

```
CL-USER 18 : 5 > atom 'ana
T

CL-USER 19 : 5 > atom L1
NIL

CL-USER 20 : 5 > listp 'maria
NIL

CL-USER 21 : 5 > listp L1
T

CL-USER 22 : 5 >
```

Representando sentencias KIF

- Sentencias KIF

(<= a (and c b))
(not (not c))
(or b (not d) e)
- Como se representa en LISP?

Usando listas:

```
CL-USER 22 : 5 >( list '<='a (list 'and'c'b))
(<= A (AND C B))

CL-USER 23 : 5 > (list 'not(list'not'c ))
(NOT (NOT C))
```

Nota: los operadores KIF =>, <=, <=>, or, and, not, siempre serán los primeros de la lista: