

INSTITUTO POLITÉCNICO NACIONAL
Centro de Investigación en Computación

**Taller de programación
en Common Lisp**

01

**Fundamentos de
Common LISP**

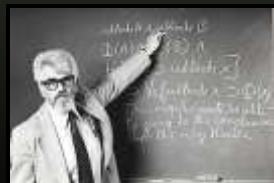
Dr. Salvador Godoy Calderón



Contexto del lenguaje...

El inicio...

John McCarthy (1927-2011) en el M.I.T. crea *LISP* en 1956, basado en el *Cálculo Lambda* para realizar Procesamiento Simbólico.

***LIS*P Processor**

Especialidad para expresar algoritmos recursivos y manejar tipos de datos dinámicos.



Laboratorio de Inteligencia Artificial

Dialectos y sabores...***Common LISP***

ANSI, 1996



Laboratorio de Inteligencia Artificial

Resumen de características...

- ◆ Manejo dinámico de la memoria.
- ◆ Soporte completo para recursividad.
- ◆ Aritmética de enteros con **precisión arbitraria**.
- ◆ Programación multiparadigma.
- ◆ Soporta **Threads**.
- ◆ Modo **Intérprete** y Modo **Compilador**.
- ◆ Un solo estándar (**ANSI INCITS 226-1994 (R2004)**)

Laboratorio de Inteligencia Artificial

***Steel Bank Common Lisp
(SBCL)***

Lo básico...

- ◆ Intérprete/Compilador sobre línea de comandos...
- ◆ Gratuito (licencia GNU)...
- ◆ Multiplataforma (Linux, Mac OS/X, Windows, etc)...

Para programar pueden usar el editor de su preferencia, pero se recomienda usar *EMACS*, *JUPYTER* o *ATOM*...

[http://functionalrants.wordpress.com/2008/09/06
/how-to-set-up-emacs-slime-sbcl-under-gnulinux/](http://functionalrants.wordpress.com/2008/09/06/how-to-set-up-emacs-slime-sbcl-under-gnulinux/)



Laboratorio de Inteligencia Artificial

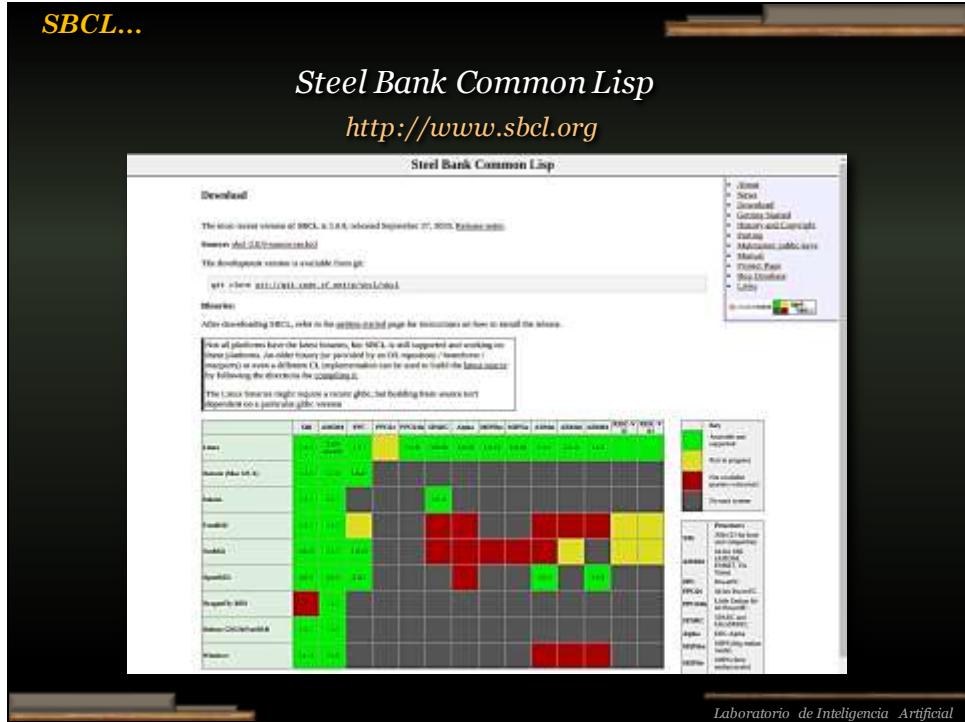
SBCL...

Steel Bank Common Lisp

<http://www.sbcl.org>



Laboratorio de Inteligencia Artificial

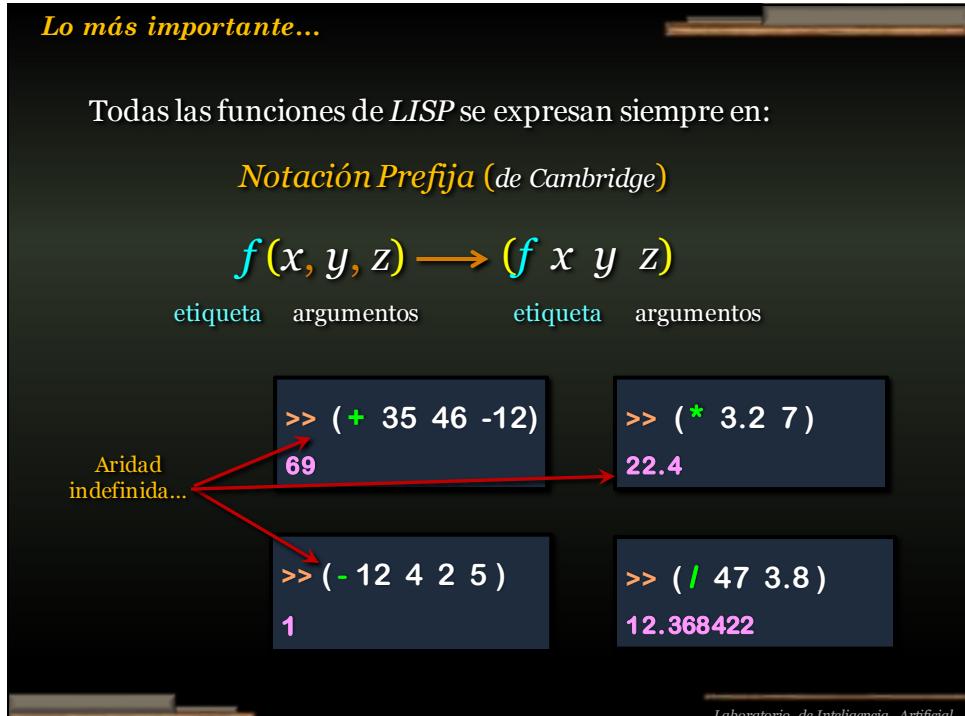
SBCL...***Steel Bank Common Lisp***<http://www.sbcl.org>

Laboratorio de Inteligencia Artificial

Lo más importante...Todas las funciones de *LISP* se expresan siempre en:*Notación Prefija (de Cambridge)*

$$\textcolor{blue}{f}(x, y, z) \longrightarrow (\textcolor{blue}{f} \ x \ y \ z)$$

etiqueta argumentos etiqueta argumentos



Laboratorio de Inteligencia Artificial

Asignación y asociación...

En *Common LISP* no se hace *asignación* de variables como en otros lenguajes...



En su lugar, los símbolos se ASOCIAN a valores de cualquier tipo...

Por ello, las variables (símbolos) no tienen *tipo de datos*, sólo los valores asociados lo tienen...

Laboratorio de Inteligencia Artificial

Tipos de Datos Básicos...

- ◆ Átomo: 
 - Numéricos *7, 3, 45.28, 79.12, ...*
 - No numéricos *HOLA, "CASA", VALOR23, NIL, T*

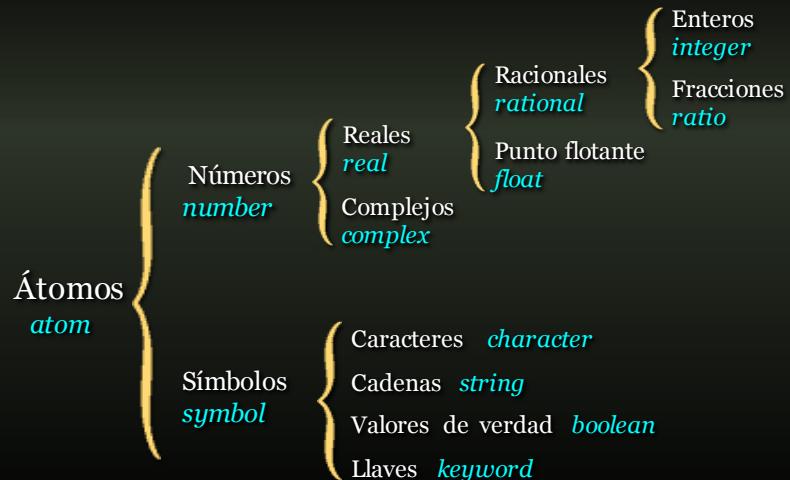


- ◆ Lista: Secuencia de elementos, separados por espacios y delimitada entre paréntesis.

(Esta es una LISTA)
(CASA 3 POLLO 45.2 (5 6 7) (3.1 3.2 3.3))
(Rojo (Verde Azul) () AMARILLO)
((uno dos) (TRES CUATRO CINCO))

Laboratorio de Inteligencia Artificial

Los átomos...



Laboratorio de Inteligencia Artificial

Procesamiento de átomos

Precisión arbitraria, no infinita...

```
>> (expt 2 (expt 2 10))
```

```
1797693134862315907729305190789024733617976978942306572734
3008115773267580550096313270847732240753602112011387987139
3357658789768814416622492847430639474124377767893424865485
2763022196012460941194530829520850057688381506823424628814
7391311054082723716335051068458629823994724593847971630483
5356329624224137216
```

```
>> (expt 2 (expt 2 (expt 2 100)))
```

Error: Attempt to create an integer which is too large to represent.
[condition type: SIMPLE-ERROR]

Laboratorio de Inteligencia Artificial

En Modo Intérprete...

La labor del intérprete es **EVALUAR** cada expresión y entregar el **VALOR** asociado a cada expresión...

Algunos átomos tienen valor asociado pre-definido:
números, caracteres, cadenas y valores de verdad

Estos átomos pueden ser directamente evaluados por el intérprete:

```
>> 23.47
23.47
```

```
>> "Hola"
"HOLA"
```

```
>> T
T
```

```
>> nil
NIL
```

Laboratorio de Inteligencia Artificial

Valor asociado...

Pero si se intenta evaluar un símbolo que no tiene algún valor asociado, el intérprete genera un error...

```
>> casa
```

<Error>

[c condition type: UNBOUND-VARIABLE]

Error: símbolo sin valor
asociado...

```
>> X
```

<Error>

[c condition type: UNBOUND-VARIABLE]

Laboratorio de Inteligencia Artificial

La solución...

Para evitar la evaluación de expresiones se usa la función QUOTE:

```
>> ( QUOTE Cañaveral )
```

CAÑAVERAL

```
>> ( QUOTE X)
```

X

Quote evita la evaluación de una expresión *LISP*.

Con listas, **Quote** hace la diferencia entre considerarlas como código o como datos...

Laboratorio de Inteligencia Artificial

Otra forma...

La función **Quote** también se representa mediante un Apóstrofo ' antes del argumento:

```
>> 'Sandía
SANDÍA
```

```
>> '(Uno Dos Tres)
(UNO DOS TRES)
```

OJO: **Quote** sólo devuelve el PRIMER elemento (átomo o lista) de sus argumentos:

```
>> '(1 2) (3 4) (5 6)
(1 2)
<Error>
[condition type: TYPE-ERROR]
```

El error lo causa la segunda lista pues al intentar evaluarla no encuentra la función "3"...

Laboratorio de Inteligencia Artificial

Expresiones anidadadas...

$$\begin{aligned} 2x^2 + 7x + 5 &= 0 \\ ax^2 + bx + c &= 0 \end{aligned}$$

$$x = \frac{-7 \pm \sqrt{7^2 - 4(2)(5)}}{2(2)}$$

```
>> (/ (+ -7.0 (sqrt (- (expt 7 2) (* 4 2 5)))) (* 2 2))
```

-1.0

```
>> (/ (- -7.0 (sqrt (- (expt 7 2) (* 4 2 5)))) (* 2 2))
```

-2.5

División ...

Raíz cuadrada ...

Exponente ...

Multiplicación ...

Multiplicación ...

Laboratorio de Inteligencia Artificial

Las cadenas...

Tipo especial de átomo alfanumérico, que se expresa entre comillas dobles y que también es un objeto auto-evaluado.

```
>> "Star Trek"
"Star Trek"
```

```
>> Star Wars
<Error>
[condition type: UNBOUND-VARIABLE]
```

```
>> ( length "Inteligencia Artificial" )
```

```
23
>> (string= "Inteligencia" "Sabiduría" )
NIL
```

```
>> (string= "Yo soy Investigador" "Yo soy Investigador" )
```

```
T
```

Laboratorio de Inteligencia Artificial

Caracteres...

Las cadenas también son vistas como arreglos de caracteres (con índice inicial = 0)

Encontrar el carácter, dentro de una cadena, que ocupa la posición indicada ...

```
>> (char "Star Trek" 0)
```

```
#\S
```

El prefijo `\#` es la forma estándar de identificar a un carácter

```
>> (char "Star Trek" 7)
```

```
#\e
```

```
>> (char "Star Trek" 4)
```

```
#\Space
```

Laboratorio de Inteligencia Artificial

Más sobre caracteres...

Los caracteres también son objetos auto-evaluados

```
>> #\K
```

```
#\K
```

```
>> (quote #\$)
```

```
#\$
```

```
>> '#\Q
```

```
#\Q
```

```
>> (quote #\ )
```

```
#\Space
```

```
>> '#\
```

```
#\NewLine
```

OJO: cuidado con la diferencia entre escribir y no escribir un espacio después del prefijo `#\`

Laboratorio de Inteligencia Artificial

Otros ejemplos...

```
>> (char "Ahuehuéte" 6 )
```

```
#\'é
```

```
>> (char= #\u (char "fUnCiOnAl" 1))
```

```
NIL
```

Determinar si dos caracteres son iguales...

```
>> (char= (char "Dos abanicos" 2 )
```

```
(char "simpático" 0 )
```

```
#\s )
```

```
T
```

Laboratorio de Inteligencia Artificial



Celdas de Construcción...

En *LISP* existe una estructura de datos fundamental llamada **Celda de Construcción (*Cons Cell*)**...

Se trata de una estructura con sólo dos campos y cada uno de ellos contiene un apuntador...



En particular, una celda puede apuntar a otra, con lo que se pueden construir cadenas ligadas de celdas

Representación interna...

Las listas siempre se representan internamente como listas ligadas de *Celdas de Construcción (Cons Cells)*.

(Rojo Verde Amarillo Azul)

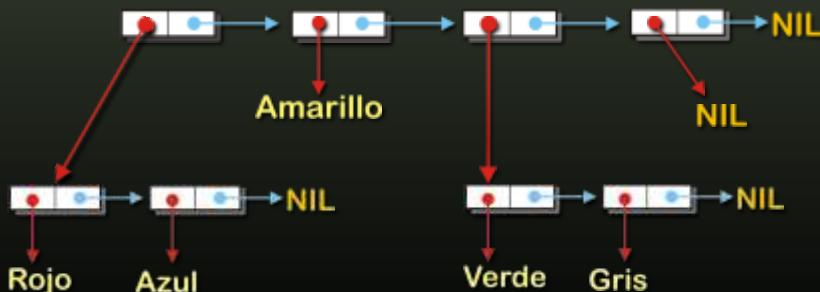


Una lista puede representar un conjunto de datos, o bien, la invocación a una función previamente definida...

Laboratorio de Inteligencia Artificial

Inclusive con sublistas...

((Rojo Azul) Amarillo (Verde Gris)())



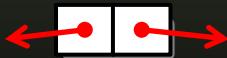
Laboratorio de Inteligencia Artificial

Creación...

La función **CONS** crea una celda de construcción y apunta cada una de sus secciones a los argumentos...

```
>> (cons 'A '(B C) )
(A B C)
```

```
>> (cons '(1 2) '(A B C) )
((1 2) A B C)
```



```
>> (cons 2 (cons 4 (cons 6 '(8))) )
```

(2 4 6 8)

```
>> (cons 'a (cons 'b (cons 'c '()))) )
```

(A B C)

Laboratorio de Inteligencia Artificial

Construcción alternativa...

Otra forma de construir listas (alternativa a **CONS**) es la función **LIST**:

```
>> (list 'A 'B '(C D) 'E )
(A B (C D) E )
```

```
>> (list nil)
(NIL)
```

```
>> (list '(nil))
((NIL))
```

A diferencia de **CONS**, la función **LIST** acepta cualquier número de argumentos y construye la lista a partir de ellos como sus elementos integrantes...

Laboratorio de Inteligencia Artificial

CONS vs LIST...

- ◆ **CONS** agrega un elemento a una lista (una celda cons)
- ◆ **LIST** construye a partir de puros elementos
- ◆ Las listas construidas por **LIST** siempre terminan en **NIL** (listas propias)
- ◆ Es posible usar **CONS** para construir una lista no-propia (no terminada en **NIL**)

¿? ¿?¿? !!!

Laboratorio de Inteligencia Artificial

Notación de componentes...

¿Cómo se denota o visualiza una lista con la siguiente estructura interna?



Se usa una notación alternativa a las listas llamada *Notación de Parejas Punteadas (separadas por un punto)* (*Dotted-pair*).

(Rojo Verde Amarillo Azul . Gris)

Laboratorio de Inteligencia Artificial

Otro uso de CONS...

Si los dos parámetros de **CONS** resultan ser átomos (en lugar de un elemento y una lista), entonces construye una *pareja punteada*:

```
>> ( cons 'Buenos 'Días )
( BUENOS . DÍAS )
```

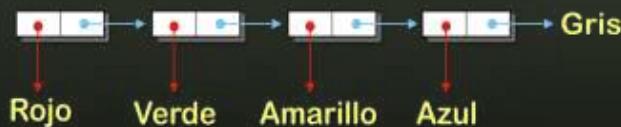
```
>> ( cons 'A (cons 'B (cons 'C 'D) ) )
( A B C . D )
```

```
>> ( list (cons 'A 'B) (cons 'C 'D) )
( (A . B) (C . D))
```

Laboratorio de Inteligencia Artificial

Cuidado...

La longitud de una lista se define como el **número de celdas, de 1er nivel**, que la componen



Por lo tanto, la longitud de:

```
( Rojo Verde Amarillo Azul . Gris )
```

es **4** y no **5**.

Laboratorio de Inteligencia Artificial

Inicio y resto...

Las funciones FIRST y REST regresan el elemento inicial de una lista y su resto, respectivamente...

```
>> (first '(A B C D))  
A
```

```
>> (rest '(A B C D))  
(B C D)
```

```
>> (first (cons 'a '(b c)))  
A
```

```
>> (rest (cons 'a '(b c)))  
(B C)
```

```
>> (first '())  
NIL
```

```
>> (rest '())  
NIL
```

Laboratorio de Inteligencia Artificial

Último...

La función LAST regresa la última celda (cons) que compone una lista. La respuesta, al ser una celda cons, siempre tiene forma de lista...

```
>> (last '(A B C D))  
(D)
```

```
>> (first (last '(a b c d)))  
D
```

```
>> (rest (last '(a b c d)))  
NIL
```

```
>> (last '())  
NIL
```

Laboratorio de Inteligencia Artificial

Curiosamente...

Además de las funciones **FIRST**, **REST** y **LAST**, están definidas también:

```
>> (second '(A B) C D (E) F)
C
```

```
>> (third '(A B C D E F))
C
```

fourth, fifth, sixth, seventh, eighth,
ninth, tenth

Laboratorio de Inteligencia Artificial

Posición de un elemento ...

Cuando no resultan útiles las funciones **FIRST**, **SECOND**, ... **TENTH** ó para realizar búsqueda aleatoria, se puede usar la función **NTH**:

```
>> (nth 12 '(a b c d e f g h i j k l m n o p))
M
```

OJO: La función **NTH** considera los índices de posición comenzando en **cero**.

```
>> (third '(a b c d))
C
>> (nth 3 '(a b c d))
D
```

Laboratorio de Inteligencia Artificial

Nota: Length y Equal...

La función **LENGTH** de cadenas también se puede usar sobre listas:

```
>> (length '( a (b c) d ))  
3
```

```
>> (length '(()()()) )  
3
```

equal compara listas:

```
>> (equal '(a) (first '((a)) ) )  
T
```

```
>> (equal '( a (b c) d ) '( a b c d ) )  
NIL
```

Laboratorio de Inteligencia Artificial

*** Nota histórica ...**

Antes del estándar ANSI de *Common LISP*, las funciones **FIRST** y **REST** no existían y sus equivalentes se llamaban **CAR** y **CDR** (*/cou-der/*)

Esas siglas son reliquias del pasado y correspondían al nombre de secciones de instrucción en la *IBM 704* (1958)

Contents of Address portion of Register
Contents of Decrement portion of Register

Aún hoy es fácil encontrar código escrito usando CAR y CDR, sin embargo no es lo recomendado...

Laboratorio de Inteligencia Artificial

Antiguamente...

En “*LISP antiguo*” las funciones **CAR** y **CDR** se podían combinar para examinar listas anidadas:

Las letras **A** (en **CAR**) y **D** (en **CDR**) se encadenaban en orden inverso para generar nuevas funciones...

```
>> ( CADR '( (A B) C D (E) F )  
C
```

/ kae-der/

/ cou-dar /

```
>> ( CDAR '( (A B) C D (E) F )  
(B)
```

Laboratorio de Inteligencia Artificial

Más...

```
>> ( CAAR '( (A B) C D (E) F )  
A
```

```
>> ( CDDR '( (A B) C D (E) F )  
(D (E) F)
```

Pero, evidentemente, la combinación dependía de la estructura de la lista de mayor nivel:

```
>> ( CDADR '( (A B) C D (E) F )
```

Error: Attempt to take the CDR of C which is not listp.
[condition type: TYPE-ERROR]

Laboratorio de Inteligencia Artificial

Equivalencia parcial...

<i>f</i>	pronunciación	equivalencia
CAR	/ kar /	FIRST
CDR	/ cou-der /	REST
CAAR	/ka-ar/	
CADR	/kae-der/	
CDAR	/cou-dar/	SECOND
DDR	/cou-dih-der/	
CAAAR	/ka-a-ar/	
CAADR	/ka-ae-der/	
CADAR	/ka-dar/	
CADDR	/ka-dih-der /	
CDAAR	/cou-da-ar/	THIRD
CDADR	/cou-dae-der/	
CDDAR	/cou-dih-dar/	
CDDDR	/cou-did-dih-der/	
CADDAR	/ka-dih-dih-der/	FOURTH
...

Modernamente se usan sólo las nuevas funciones, aunque las antiguas aún sirven...

Laboratorio de Inteligencia Artificial

Predicados de Identificación...

¿Qué son los predicados?

Se trata de funciones cuyo resultado es siempre un valor de verdad (**T**, **NIL**) y, sobre los cuales, es posible siempre aplicar los conectores lógicos

```
( AND <arg1> <arg2> ... )
( OR <arg1> <arg2> ... )
( NOT <arg> )
```

- ◆ **NOT** acepta estrictamente sólo un argumento
- ◆ **AND** y **OR** aceptan cualquier número de argumentos y evalúan secuencialmente...

Laboratorio de Inteligencia Artificial

En listas...

El predicado **NULL** verifica si su único argumento, es o no, una lista vacía:

```
>> ( null '(1 2 3) )
NIL
```

```
>> ( null '() )
T
```

```
>> ( null NIL )
T
```

```
>> ( null () )
T
```

```
>> ( null '(NIL) )
NIL
```

Laboratorio de Inteligencia Artificial

Algunos más...

Otros predicados en *LISP* son los siguientes:

```
( numberp <arg>)
  ( oddp <arg>)
  ( evenp <arg>)

  ( > <arg1> <arg2> )
  ( >= <arg1> <arg2> )
  ( < <arg1> <arg2> )
  ( <= <arg1> <arg2> )
```

Además de los ya vistos:

```
( EQUAL <arg1> <arg2> ... )
( STRING= <arg1> <arg2> ... )
( CHAR= <arg1> <arg2> ... )
```

Laboratorio de Inteligencia Artificial

Los predicados de identificación...

Los predicados de identificación permiten saber si un objeto es de algún tipo de datos específico...

```
>> (atom '(a b c))
```

```
NIL
```

```
>> (atom 312.26)
```

```
T
```

Predicado	Tipo de datos que identifica
atom	átomos
numberp	átomos numéricos
symbolp	átomos simbólicos
listp	listas
realp	números reales
complexp	números complejos
rationalp	números racionales
floatp	números de punto flotante
integerp	números enteros
ratiofp	fracciones
characterp	caracteres
alpha-char-p	caracteres alfabéticos
alphanumericp	caracteres alfanuméricos
keywordp	llaves
stringp	cadenas

Laboratorio de Inteligencia Artificial

La tradición...

Por tradición, la mayoría de los predicados de identificación terminan con el carácter ‘p’

Yo prefiero usar el carácter ‘?’

Predicado	Tipo de datos que identifica
atom	átomos
numberp	átomos numéricos
symbolp	átomos simbólicos
listp	listas
realp	números reales
complexp	números complejos
rationalp	números racionales
floatp	números de punto flotante
integerp	números enteros
ratiofp	fracciones
characterp	caracteres
alpha-char-p	caracteres alfabéticos
alphanumericp	caracteres alfanuméricos
keywordp	llaves
stringp	cadenas

Laboratorio de Inteligencia Artificial

TAREA...

- 1) Bajar e instalar *SBCL* y configurarlo para su uso con *EMACS*...
- 2) Averiguar las funciones *LISP* para entrada/salida (consola y archivo), así como la forma de compilar un programa y generar código ejecutable...
- 3) Estudiar:



Chapter 1:
Introduction:
Why *LISP*?

Chapter 2:
Lather, Rinse, Repeat:
A Tour of the *REPL*



Preface:
Why *LISP*?
Why *Common LISP*?

Chapter 1:
Introduction to *LISP*
1.1 - 1.4

Laboratorio de Inteligencia Artificial

