

# Inteligencia Artificial e Ingeniería del Conocimiento

## PRACTICAS LISP

(Allegro Common Lisp)

2º curso

Ingeniero Superior Informática

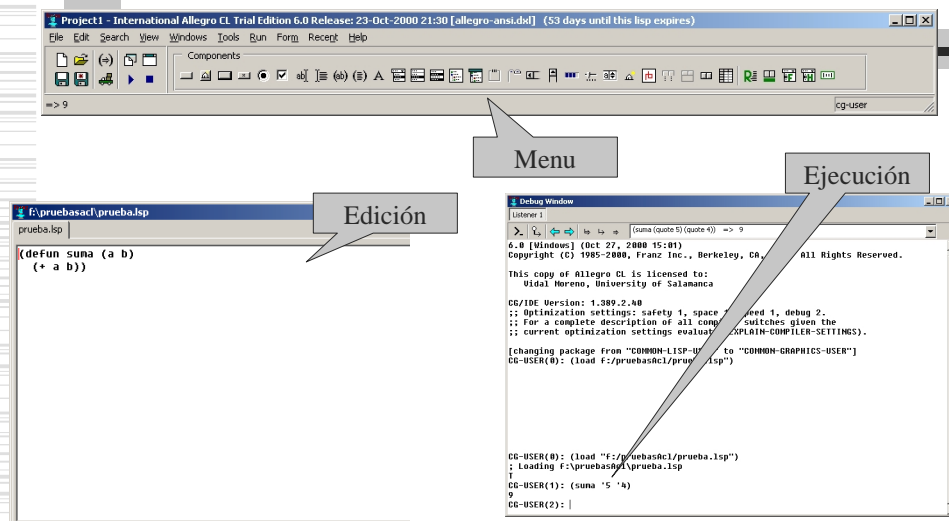
## El lenguaje LISP

- ◆ Historia
  - LISP (LISt Processing)
  - Nace a principios de los años 60
- ◆ Standard
  - Common Lisp
    - ◆ <http://www.cs.cmu.edu/afs/cs.cmu.edu/project/ai-repository/ai/html/cltl/cltl2.html>
  - Versiones
    - ◆ XLISP
      - Distribución RedHat
    - ◆ CLISP.
      - URL: [http://arrakis.linuxberg.com/conhtml/dev\\_languages\\_lisp.html](http://arrakis.linuxberg.com/conhtml/dev_languages_lisp.html)
    - ◆ ACL
      - URL:<http://www.franz.com>

# Allegro Common Lisp

- ◆ Versión 6.0
  - 60 días
- ◆ Versiones
  - Windows
  - Linux
    - Se utiliza el editor *emacs*
- ◆ Ejecución
  - Interpretada
  - Compilada (Proyectos)

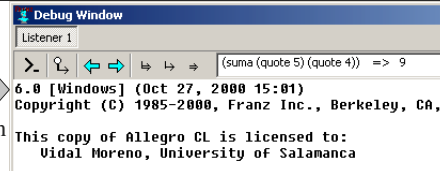
# Allegro Common Lisp



# Introducción al LISP

## ◆ Componentes de la aplicación

- "Listener" o línea de comandos
- "Evaluator". Encargado de la obtención de resultados



## ◆ Definición

- Símbolo: Unidad de almacenamiento de información. Existen dos tipos
  - Átomo: unidad sintáctica básica
  - Lista: Secuencia de átomos o de otras listas separadas por espacios y encerradas entre paréntesis

## ◆ Ejemplo

```
> 1
1
> "hola"
"hola"
> (+ 1 2)
3
> '(1 2)
(1 2)
> (+ 1 2)
(+ 1 2)
```

# Introducción al LISP

## ◆ Funciones más utilizadas

- *atom*. Define si átomo
- *eq*. Define si dos átomos son iguales
  - Para listas *equal*
- *first*. Obtiene el primer elemento de una lista
  - Antiguo *car*
- *rest*. Obtiene el resto de los elementos de una lista
  - Antiguo *cdr*
- *cons*. Añade un elemento a una lista

```
> (atom '1)
T
> (atom '(1))
NIL
> (eq '1 '1)
T
> (eq '(1) '(1))
NIL
> (first '(1 2 3))
1
> (rest '(1 2 3))
(2 3)
> (cons '1 '(2 3))
(1 2 3)
```

# Introducción al LISP

- ◆ Asignación de valores `> (setf q '(1 2 3))`  
`(1 2 3)`
- Definición de funciones

Ejemplo 1 →

```
> (defun prim_seg(p)
  (list (first p) (first (rest p))))
PRIM_SEG
> (prim_seg '(1 (1 2) 3))
(1 (1 2))
```

Ejemplo 2 →

```
> (defun lista(p)
  (if (atom p) NIL T))
LISTA
> (lista '1)
NIL
> (lista '())
NIL
> (lista '(1 2))
T
> (atom '())
T
```

# Introducción al LISP

- ◆ Calculo condicional

```
(defun fact1(n)
  (if (or (= n 0) (= n 1))
      1
      (* n (fact1 (- n 1)))))
```

```
(defun fact2(n)
  (cond((= n 0) 1)
        ((= n 1) 1)
        (T (* n (fact2 (- n 1)))))
```

---

## Prácticas

---

- ◆ Problema 1:
  - Calcular la serie de Fibonacci
    - $a(n)=a(n-1)+a(n-2)$  con  $a(0)=1$   $a(1)=1$
- ◆ Problema 2:
  - Dados dos predicados
    - $P(x,f(h,z),A) :-> (P (? x) (f (? h) (? z)) A)$
    - $P(a,b,c) :-> (P (? a) (? b) (? c))$
  - Obtener el unificador

---

## Prácticas

---

- ◆ Ayuda: Determinación de átomo

```
(defun atomo (var)
  (cond ((atom var) T)
        ((eq (first var) '?) T)
        (T NIL)))
```