ESTRUCTURAS EN COMMON LISP

LAURA DE MIGUEL
JAVIER FERNANDEZ
Dpto. de Estadística, Informática y Matemáticas
laura.demiguel@unavarra.es

ESTRUCTURAS

ESTRUCTURAS

Las estructuras son un tipo de dato definidos por el usuario. Permiten combinar campos de distinto tipo.

Para guardar información de libros en una biblioteca.

Se quieren guardar los siguientes atributos por cada libro: título, autor, año, id-libro.

```
(defstruct nombreEstruct
        componente-1        titulo
        componente-2        autor
        componente-3        año
        compontente-4       id-libro )
```

```
(defstruct  libro
        titulo
        autor
        año
        id-libro )
```

CREAR UNA ESTRUCTURA

```
(make-nombreStruct
        :componente-1 expresión
        :componente-2 expresión
        …
        :componente-n expresión))
```

ASIGNACIÓN DE UNA ESTRUCTURA

```
(setf libro1 (make-libro
        :titulo "Common Lisp"
        :autor "Graham"
        :año 1995
        :id-libro 1))

(setf libro2 (make-libro
        :titulo "Lisp"
        :autor "Winston"
        :año 1991
        :id-libro 2))
```

Los campos no inicializados explícitamente se inicializan con NIL.

MOSTRAR POR PANTALLA UNA ESTRUCTURA

```
> (write libro1)
  (terpri)
  (write libro2)

#S (LIBRO :TITULO “Common Lisp”  :AUTOR “Graham
      :AÑO 1995 :ID-LIBRO 1)
#S (LIBRO :TITULO “Lisp” :AUTOR “Winston
      :AÑO 1991 :ID-LIBRO 2)
```

ACCESO A LAS COMPONENTES

(nombreEstruct-componentej) – devuelve la información de la compontente j)

> (libro-autor libro1)
"Graham"
> (libro-año libro2)
1991

```
(defstruct (nombre  (:constructor nombreFuncCons)
                    (:conc-name prefijo-)
                    (:print-function nombreFuncPrint))
  componente-1
  …
  componente-n)

> (defstruct (punto (:constructor crea-punto)
                    (:conc-name coordenada-))
  x
  y)
  PUNTO
```

```
(nameConsFunct
      :componente-1 valor
      …
      :componente-n valor)

> (setf  *punto-1*  (crea-punto :x 2 :y 3))
   # S (PUNTO :X 2 :Y 3)


> (coordenada-y *punto-1*)
   3
```

COPIAR UNA ESTRUCTURA

(copy-nombreEstruct struct1)

```
> (setf  *punto-2* (copy-punto *punto-1*))
  # S (PUNTO :X 2 :Y 3)
```

MODIFICA UNA ESTRUCTURA

```
> (setf (coordenada-y *punto-2*) 45)
  45
> *punto-2*
  # S (PUNTO :X 2 :Y 45)
```

IGUALDAD ENTRE ESTRUCTURAS : EQUALP

```
> (setf (coordenada-y *punto-2*) 3)
   3
> *punto-1*
  #S(PUNTO :X 2 :Y 3)
> *punto-2*
  #S(PUNTO :X 2 :Y 3)
> (equal *punto-2* *punto-1*)
  NIL
> (equalp *punto-2* *punto-1*)
  T
```