





LISP - Funciones Recursivas

Uploaded by foxfive_15g

Full description

-  Save
-  Embed
-  Share
-  Print

RELATED TITLES



Inteligencia Artificial e Ingeniería del Conocimiento I
BÚSQUEDA
PDF

Inteligencia Artificial e



SERVICIOS DE RECURSION
PDF

Recursividad.pdf



Ejercicios de Listas lisp
PDF

Ejercicio Listas lisp

07200078 Ccaipani Sánchez Marco
07200021 Garaundo Rodríguez Carlos
07200042 Rojas Alvarado Giovana
07200115 Sulca Paucar Eder Kim

Inteligencia Artificial [06/10/10]

LISP - Funciones Recursivas

- 1) Factorial (n!)

```
(defun factorial (n)
  (if (= n 0) 1
      (* n (factorial (- n 1)))
  )
)
```
- 2) Potencia (n^m)

```
(defun potencia (x m)
  (if (= m 0) 1
      (* x (potencia x (- m 1)))
  )
)
```
- 3) Longitud de una Lista

```
(defun longlist (lista)
  (cond ((null lista) 0)
        (T (+ 1 (longlist (CDR lista)))))
)
```
- 4) Suma de los N primeros números enteros

```
(defun sumatoria (n)
  (if (= n 0) 0
      (+ n (sumatoria (- n 1)))
  )
)
```
- 5) Producto de los N primeros números enteros


```
(defun producto (n)
  (if (= n 1) 1
      (* n (producto (- n 1)))
  )
)
```
- 6) Aplana una lista


```
(defun aplana (lista)
  (cond((null lista) NIL)
        ((atom (CAR lista))
```


LISP - Funciones Recursivas


Uploaded by foxfive_15g

Full description


 Save

 Embed

 Share

 Print

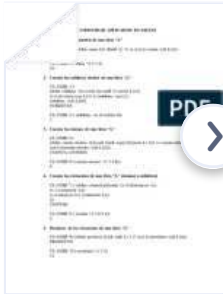
RELATED TITLES




Inteligencia Artificial e Ingeniería del Conocimiento I: BÚSQUEDA PDF



PROCESOS DE RECURSION PDF



Ejercicios de Listas en LISP PDF



Ejercicios de Listas en LISP PDF

```
(defun prodescalar (lista1 lista2)
  (if (or (null lista1) (null lista2)) 0
      (+ (* (CAR lista1) (CAR lista2))
         (prodescalar (CDR lista1) (CDR lista2)))
      )
  )
)
```

8) Sumar los elementos de una lista (no anidada)

```
(defun sumalista (lista)
  (cond ((null lista) 0)
        (T (+ (CAR lista) (sumalista (CDR lista)))
          )
  )
)
```

9) Verificar si un elemento pertenece a una lista

```
(defun pertenece (x lista)
  (if (endp lista)
      nil
      (if (= x (car lista))
          lista
          (pertenece x (cdr lista)))
      )
)
```

10) Elimina elemento de una lista

```
(defun eliminar (x lista)
  (cond ((endp lista) lista)
        ((equal x (CAR lista)) (CDR lista))
        (T (cons (CAR lista) (eliminar x (CDR lista))))
  )
)
```