Taller final - LISP

Final work – LISP

Autor: Luis Fernando Martínez Muñoz
IS&C, Universidad Tecnológica de Pereira, Pereira, Colombia
Correo-e: f.martinez@utp.edu.co

***Resumen*— El presente artículo se ha escrito con el propósito de ilustrar la implementación de los últimos ejercicios propuestos para el reconocimiento del lenguaje LISP.**

***Palabras clave*— LISP, lista, función, lambda.**

***Abstract*— This article has been written with the purpose of illustrating the implementation of the last exercises proposed for an easier LISP language understanding.**

***Key Word*— LISP, list, function, lambda function.**

I. INTRODUCCIÓN

Vamos a desarrollar en este paper, unos pocos ejercicios para finalizar el afianzamiento con las funciones y sintaxis del lenguaje LISP.

II. CONTENIDO

Se trabajarán las funciones lambda, FUNCALL y APPLY.

A. Presentación

Para empezar, se estudiarán las funciones lambda, su sintaxis y usos, a continuación, se introducirá la función FUNCALL y para finalizar se observará el comportamiento de la función APPLY.

B. Fundamento teórico

La función Lambda tiene como objetivo hacer el llamado a una función que no está declarada, "función anónima", esto justificado por el poco uso que podría tener esa función en el resto del programa, necesitándose solo en algún caso particular.

La función FUNCALL permite definir procedimientos que tengan procedimientos como argumentos.

La función APPLY permite aplicar una función, que recibe como primer argumento, a una lista de argumentos.

```
CL-USER 1 > (print ((lambda (x) (+ x 3)) 5))

8
8
```

Figura 1. Ejercicio 1.a

```
CL-USER 2 > (print ((lambda (a b) (+ a (* b 3))) 4 5))

19
19
```

Figura 2. Ejercicio 1.b

```
CL-USER 3 > (print ((lambda (m n) (+ m n)) 2 3))

5
5
```

Figura 3. Ejercicio 1.c

```
CL-USER 5 > (print ((lambda (x y) (+ (* x x) (* y y))) 3
 4))

25
25
```

Figura 4. Ejercicio 1.d

C. Ejemplos de aplicación

Figura 5. Enunciado ejercicio 1.e

Para transformar la función en una función lambda se debe borrar la palabra reservada defun y el nombre de la función, esto será reemplazado por “lambda”, los parámetros y funcionamiento son exactamente los mismos. Para entregarle valores a evaluar, se ponen inmediatamente después de la función lambda.

```
CL-USER 6 > (print (funcall '+ 1 2 3))

6
6
```

Figura 6. Ejercicio 2.a

```
CL-USER 7 > (print (funcall #'(lambda (x) (* 2 x)) 8))

16
16
```

Figura 7. Ejercicio 2.b

```
CL-USER 8 > (print (apply '+ '(2 4 6)))

12
12
```

Figura 8. Ejercicio 3.a

```
CL-USER 9 > (print (apply #'(lambda (x y z) (+ x y z)) '(
2 4 6)))

12
12
```

Figura 9. Ejercicio 3.b

D. Conclusión

Las funciones lambda, FUNCALL y APPLY son necesarias para el paso de funciones como parámetros, facilitando el trabajo del programador.

REFERENCIAS

Referencias en la Web:

http://www.redesep.com/materias/blanda/4_3_recursividad.php

http://www.redesep.com/materias/blanda/5_2_estructuras_recursivas.php