Talleres - LISP

Workshops – LISP

Autor: Daniel Patiño Rojas
Estudiante de Ingeniería en Sistemas y computación, Universidad Tecnológica de Pereira, Pereira, Colombia
Correo-e: danielrojas0198@utp.edu.co

***Resumen*— El presente artículo se ha escrito con el propósito realizar los 11 talleres propuestos en clase**

***Palabras clave*— LISP, funcional, función, recursividad, listas.**

***Abstract*— This article has been written with the purpose of carrying out the 11 workshops proposed in class**

***Key Word*— LISP, functional, function, recursion, lists.**

INTRODUCCIÓN

Este paper está desarrollado con el fin de entender y evaluar los siguientes talleres en el lenguaje LISP, el cual es un lenguaje multiparadigma con una gran historia. El lenguaje de programación LISP fue diseñado originalmente en 1958 por John McCarthy y sus colaboradores en el insti Tecnológico de Massachusetts; siendo este el según lenguaje de programación de alto nivel con mayor antigüe entre los que existen actualmente. LISP, originalmente fue creado en base al cálculo lambda Alonzo Churcha, convirtiéndose así en uno de los lengu favoritos para la investigación de la Inteligencia Artificial. LISP es pionero de muchas ideas en la ciencia de computación tales como las estructuras de datos de árbol manejo de almacenamiento dinámico entre otras. Las li encadenadas son una de las estructuras de datos importa de LISP, dado que su código está compuesto de listas da resultado que los programas escritos en este pue manipularse como si fuera una estructura de datos.

CONTENIDO

Los talleres son los siguientes:

A. TALLER 1

```
;1- Evalue las siguientes expresiones

(* (/ (+ 3 3) (- 4 4) ) (* 0 9) )
(+ (* 3 (- 5 3) ) (/ 8 4) )
(* 3 2 2 (- 8 5) )
( (*) (* 2 (- 4 2) ) (/ 8 2) )

;2- Evalue las siguientes formas
(SQRT (ABS (- 71 (EXPT (+ 2 4) 4) ) ) )
'(' + '1 '(* 2 4) )
(EXPT (MOD 5 3) (ABS (- 8 9) ) )
(QUOTE (NIL 'NIL T 'T) )
(LOG (EXPT 2 (- 15 5 4) ) )
(QUOTE (QUOTE (HElLo ByeBye) ) )
```

Los resultados son los siguientes:

1. Evaluando las expresiones, cabe recalcar que no muestran nada si no establecemos un print para poder mostrar en pantalla.
   - *** - /: division by zero
   - 8
   - 36
   - *** - EVAL: (*) is not a function name; try using a symbol instead
2. Evaluanto las siguientes formas, recalcando igualmente que sin un print no mostrara nada.
   - 35
   - ('+ '1 '(* 2 4))
   - 2
   - (NIL 'NIL T 'T)
   - 4.158883
   - '(HELLO BYEBYE)

B. TALLER 2

```
;TALLER 2

(SETQ A 4 B 6 C 5 X (+ A B) Y (- B C) Z (MAX A C))
(print (+ A B C))
(print (EVAL X))
(print (EVAL Y))
(print (EVAL Z))
```

Los resultados de las sentencias anteriores son los siguientes, recalcando que se les ha añadido un print para poder mostrar en pantalla:

- 15
- 10
- 1
- 5

C. TALLER 3

```
;TALLER 3
;Evaluar teniendo SETQ
(SETQ A 4 B 6 C 5 X (+ A B) Y (- B C) Z (MAX A C))
(print (+ (* C Z) B C))
(print (ABS (+ (* (- Z X A) -100 ) B )))
(print (* (+ (* C X) (- A Z C)) 2 Y))
;Evaluar estas otras expresiones con un nuevo SETQ
(SETQ M (+ Z A) N (- Y C) P (* 2 Z))
(print (+ M Z X Y P B N))
(print (- (- (* 3 Z) ( / 100 C)) A C N P))
```

Evaluando las expresiones nos da como resultado lo siguiente:

- 36
- 906
- 88
- Evaluando el nuevo SETQ (aca cabe aclarar, si se evalua todo en un mismo codigo como se muestra anteriormente dan los resultados siguientes, si no no encuentra la variable Z)
- 37
- -20

D. TALLER 4

```
;TALLER 4
;Evalue las siguientes expresiones:

(print (CONS(CAR ' (AXL WIL RICH)) (CDR '(ESTE ANTO ALLA))))
(print (CONS (CAR '((CONS Go Up))) (THIRD '(We Find(All Fine)))))
(print (CAR (CDR (CAR (CDR '((a b) (c d) (e f)))))))
(print (CAR (CAR (CDR (CDR '((a b) (c d) (e f)))))))
(print (CAR (CAR (CDR '(CDR ((a b) (c d) (e f)))))))
(print '(CAR (CAR (CDR (CDR ((a b) (c d) (e f)))))))
(print (CONS (CAR NIL) (CDR NIL)))

;Evalue las siguientes formas

(print (CDR (CAR (CDR (CAR '(( D (E F) G (H I)))))))
(SETQ A '(+ 3 6))
(print (CDR A))
(print (CAR (CDR A)))
(print (CAR (CDR (CDR A))))
```

En el taller número 5 nos piden evaluar expresiones y formas, dando como resultado lo siguiente:

- Evaluando las expresiones
  - (AXL ANTO ALLA)
  - ((CONS GO UP) ALL FINE)
  - D
  - E
  - (A B)
  - (CAR (CAR (CDR (CDR ((A B) (C D) (E F))))))
  - (NIL)
- Evaluando las formas tenemos un pequeño problema, la segunda forma tiene un parentesis adicional al final, dando asi error por parentesis, corrigiendo este error nos da lo siguiente:
  - (F)
  - (3 6)
  - 3
  - 6

E. TALLER 5

```
;TALLER 5
;Escribir las secuencias CAR y CDR para extraer el simbolo MAPACHE
(print '(oso gato mapache ardilla))
(print '((oso gato) (mapache ardilla)))
(print '(((oso) (gato) (mapache) (ardilla))))
(print '(oso (gato) ((mapache)) (((ardilla)))))
```

Nos piden establecer las secuencias CAR y CDR para obtener MAPACHE en todas las listas, la respuesta es la siguiente:

```
;TALLER 5
;Escribir las secuencias CAR y CDR para extraer el simbolo MAPACHE
(print (CAR (CDR (CDR'(oso gato mapache ardilla)))))
(print (CAR (CAR (CDR '((oso gato) (mapache ardilla))))))
(print (CAR (CDR (CDR (CAR '(((oso) (gato) (mapache) (ardilla)))))))
(print (CAR (CDR (CDR '(oso (gato) ((mapache)) (((ardilla)))))))
```

F. TALLER 6

```
;TALLER 6
;Utilizando las funciones para acceder a las listas, extraer leon y futbol
(print '(Managua Chinandega Rivas Leon Boaco))
(print '((Managua) (Chinandega Rivas Leon) Boaco))
(print '(Managua (Chinandega (Rivas Leon Boaco))))
(print '(deportes (Beisbol tenis) ((futbol) billar)))
```

La solución a este taller es la siguiente:

```
;TALLER 6
;Obteniendo LEON
(print (SECOND (CDR (CDR '(Managua Chinandega Rivas Leon Boaco)))))
(print (NTH 3 '(Managua Chinandega Rivas Leon Boaco)))
(print (NTH 1 (NTHCDR 1 (CAR (CDR '((Managua) (Chinandega Rivas Leon) Boaco))))))
(print (THIRD (NTH 0 (NTHCDR 1 '((Managua) (Chinandega Rivas Leon) Boaco)))))
(print (SECOND (SECOND (SECOND '(Managua (Chinandega (Rivas Leon Boaco)))))))
(print (CAR (CDR (CDR (CDR '(Managua (Chinandega (Rivas Leon Boaco)))))))
;Obteniendo FUTBOL
(print (NTH 0 (CDR (NTHCDR 1 '(deportes (Beisbol tenis) ((futbol) billar))))))
(print (CAR (THIRD '(deportes (Beisbol tenis) ((futbol) billar)))))
```

G. TALLER 7

```
;TALLER 7
;Evaluar las siguientes expresiones
(print (CONS (SECOND '(leo mana china)) (CONS (LIST (CAR '(1 2 3 4)) (CDR '(A B C D))) '2)))
(print (LIST (LIST '(ca ce) (LAST '(0 a 1 b ci) (THIRD '(5 4 3 2 1))) (NTH 3 '(pa pe pi po pu)))))
(print (LIST (APPEND '(H O L A) '(M U N D O)) (CDDR '(sal pan arroz pollo)) (CAR '(buen mal)) (CADDR '(desayuno recreo almuerzo cena))
(print (LIST '¿Como estas?) (NTH 0 '(bien mal rematado)) (APPEND (CAR '((estas) estoy)) (CDR '(entiendo entendiendo lisp)))))
(print (LENGTH (CONS (CAR '(verdad mentira falso)) (LIST* 'es 'muy '(facil)))))
(print (CDR (LIST (SUBSEQ '(z q y x w v) 0 2) (CONS '(a e i) '(o u)))))
```

La solución a este taller es la siguiente:

- (MANA (1 (B C D)) . 2)
- (((CA CE) (1 B CI) PO))
- ((H O L A M U N D O) (ARROZ POLLO) BUEN ALMUERZO)
- ((¿COMO ESTAS?) BIEN (ESTAS ENTENDIENDO LISP))
- 4
- (((A E I) O U))
- La siguiente respuesta sucede por que en algunas sentencias hacen falta parentesis, hay que organizarlas para que el compilador funcione bien.
- *** - READ: input stream #<INPUT BUFFERED FILE-STREAM CHARACTER #P"source_file.lsp" @7> ends within an object. Last opening parenthesis probably in line 6

H. TALLER 8

```
;TALLER 8
;Sea SETQ evaluar las siguientes expresiones
(SETQ paises '((Nicaragua.Managua) (Italia.Roma)(España.Madrid)))
(print (LIST 'Nicaragua 'Italia 'España (SUBLIS paises '(Nicaragua Italia España)) (NTH 1 '(eran son seran)) (CAR (NTHCDR 1
        (CDR '(pueblos estados capitales barrios)))) (CONS 'de (CONS 'estos (CONS 'paises NIL)))))
```

La solucion sera la siguiente:

- (NICARAGUA ITALIA ESPAÑA (NICARAGUA ITALIA ESPAÑA) SON CAPITALES (DE ESTOS PAISES))

I. TALLER 9, 10 y 11

```
;TALLER 9, 10 y 11
(SETQ Palabras '(grande bonito feliz húmedo)
      Sinónimos '(alto bello contento mojado)
      Antónimos '(pequeño feo triste seco)
      Ingles '(big beautiful happy humid))

(SETQ Palabras-Sinónimos (PAIRLIS Palabras Sinónimos))
(SETQ Palabras-Antónimos (PAIRLIS Palabras Antónimos))
(SETQ Palabras-Ingles (PAIRLIS Palabras Ingles))

(print (CDR (ASSOC 'Grande Palabras-Ingles)))
(print (CDR (ASSOC 'Feliz Palabras-Sinónimos)))
(print (FIRST (ASSOC 'Humedo (ACONS 'pal 'anto Palabras-Antónimos))))
(print (LAST (CONS (LENGTH (ACONS 'Lenguaje 'Ingles Palabras-Ingles)) (CONS '(Agregando) (CONS '(Palabras) (LIST '(ala)
       '(Lista-Asoc)))))))
```

El taller 9 tenemos que establecer las listas Palabras, Sinónimos, Antónimos y Ingles, lo cual se ha realizado con un SETQ y se definen las listas asociativas Palabras-Sinónimos, Palabras-Antónimos y Palabras-Ingles, las cuales se han definido con la función PAIRLIS la cual asocia las palabras de las listas.

Una vez realizado esto se procede a evaluar las expresiones las cuales dan como resultado lo siguiente:

- BIG
- CONTENTO
- NIL
- ((LISTA-ASOC))

Como observamos en comparación con las respuestas de los talleres 10 y 11, tenemos una diferente la cual es la expresión número tres, en el taller debe de dar HUMEDO pero nos da como resultado NIL.