Talleres LISP

LISP exercises

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 las formas principales de las expresiones en el lenguaje LISP.**

***Palabras clave*— LISP, lista, expresión.**

***Abstract*— This article has been written with the purpose of illustrating the main structure of LISP language expressions.**

***Key Word*— LISP, list, expression.**

I. INTRODUCCIÓN

Se evidenciará en este paper la evaluación de múltiples expresiones escritas en lenguaje LISP, además, la solución a diferentes problemas propuestos al final del documento.

II. CONTENIDO

Para empezar, ejecutaremos en el entorno LISPWorks las expresiones propuestas, para finalizar, se hallará solución a una variedad de problemas planteados.

A. Presentación

Los ejercicios para realizar tendrán la siguiente estructura:

TALLER 1

1- Evalúe las siguientes expresiones:

```
a) ( * ( /  ( + 3 3 ) ( - 4 4 ) ) ( * 0 9 ) ).
b) (+ (* 3 (- 5 3)) (/ 8 4)).
c) (* 3 2 2 (- 8 5)).
d) ( ( * )  ( * 2 ( - 4 2 ) )  ( / 8 2 ) ).
```

2- Evalúe las siguientes formas:

```
a) ( SQRT ( ABS( - 71 ( EXPT ( + 2  4 )  4 ) ) ) ).
b) '(  ' +  '1  '( * 2  4 ) ).
c) ( EXPT ( MOD 5  3 ) ( ABS ( - 8  9 ) ) ).
d) ( QUOTE ( NIL  'NIL  T  'T ) ).
e) (LOG (EXPT 2 (- 15 5 4)))
f) (QUOTE (QUOTE (Hello ByeBye))).
```

Figura 1. Ejemplo de estructura de los ejercicios

B. Fundamento teórico

El lenguaje LISP utiliza las expresiones simbólicas para representar los datos y para representar los programas.

Las expresiones simbólicas, conocidas como S-expressions, sirven para representar una lista anidada de datos. En el lenguaje LISP se utiliza la notación en prefijo para escribir dichas expresiones.

Se resuelven algunos talleres que servirán como casos de estudio de la sintaxis y algunas funciones propias de LISP.

C. Ejemplos de aplicación

```
CL-USER 1 > (* (/ (+ 3 3) (- 4 4)) (* 0 9))

Error: Division-by-zero caused by / of (6 0).
  1 (continue) Return a value to use.
  2 Supply new arguments to use.
  3 (abort) Return to level 0.
  4 Return to top loop level 0.
```

Figura 2. Ejercicio 1.1.a

```
CL-USER 4 > (+ (* 3 (- 5 3)) (/ 8 4))
8
```

Figura 3. Ejercicio 1.1.b

```
CL-USER 5 > (* 3 2 2 (- 8 5))
36
```

Figura 4. Ejercicio 1.1.c

```
CL-USER 6 > ( (*) (* 2 (- 4 2)) (/ 8 2))

Error: Illegal argument in functor position: (*) in »
 ((*) (* 2 (- 4 2)) (/ 8 2)).
  1 (continue) Evaluate (*) and ignore the rest.
  2 (abort) Return to level 0.
  3 Return to top loop level 0.
```

Figura 5. Ejercicio 1.1.d

```
CL-USER 10 > (sqrt (abs (- 71 (expt (+ 2 4) 4))))
35.0
```

Figura 6. Ejercicio 1.2.a

```
CL-USER 11 > '(' + '1 '(* 2 4))
((QUOTE +) (QUOTE 1) (QUOTE (* 2 4)))
```

Figura 7. Ejercicio 1.2.b

```
CL-USER 12 > (expt (mod 5 3) (abs (- 8 9)))
2
```

Figura 8. Ejercicio 1.2.c

```
CL-USER 13 > (quote (nil 'nil T 'T))
(NIL (QUOTE NIL) T (QUOTE T))
```

Figura 9. Ejercicio 1.2.d

```
CL-USER 14 > (log (expt 2 (- 15 5 4)))
4.158883
```

Figura 10. Ejercicio 1.2.e

```
CL-USER 15 > (quote (QUOTE (Hello ByeBye)))
(QUOTE (HELLO BYEBYE))
```

Figura 11. Ejercicio 1.2.f

```
CL-USER 16 > (setq A 4 B 6 C 5 X (+ A B) Y (- B C) »
z (max A C))
5

CL-USER 17 > (+ a b c)
15

CL-USER 18 > (eval x)
10

CL-USER 19 > (eval y)
1

CL-USER 20 > (eval z)
5
```

Figura 12. Ejercicio 2

```
CL-USER 22 > (+ (* c z) b c)
36
```

Figura 13. Ejercicio 3.3.a

```
CL-USER 23 > (abs (+ (* (- z x a) -100) b))
906
```

Figura 14. Ejercicio 3.3.b

```
CL-USER 24 > (* (+ (* c x) (- a z c)) 2 y)
88
```

Figura 15. Ejercicio 3.3.c

```
CL-USER 25 > (setq m (+ z a) n (- y c) p (* 2 z))
10

CL-USER 26 > (+ m z x y p b n)
37
```

Figura 16. Ejercicio 3.3.d

```
CL-USER 27 > (- (- (* 3 z) (/ 100 c)) a c n p)
-20
```

Figura 17. Ejercicio 3.3.e

```
CL-USER 28 > (cons (car '(axl wil rich)) (cdr '(est »
e anto alla)))
(AXL ANTO ALLA)
```

Figura 18. Ejercicio 4.4.a

```
CL-USER 29 > (cons (cdar '((cons go up))) (third '( »
we find(all fine))))
((GO UP) ALL FINE)
```

Figura 19. Ejercicio 4.4.b

```
CL-USER 30 > (car (cdr (car (cdr '((a b) (c d) (e f »
))))))
D
```

Figura 20. Ejercicio 4.4.c

```
CL-USER 32 : 1 > (car (car (cdr (cdr '((a b) (c d) »
(e f))))))
E
```

Figura 21. Ejercicio 4.4.d

```
CL-USER 34 > (car (car (cdr '(cdr ((a b) (c d) (e f »
))))))
(A B)
```

Figura 22. Ejercicio 4.4.e

```
CL-USER 43 > '(car (car (cdr (cdr ((a b) (c d) (e f »
))))))
(CAR (CAR (CDR (CDR ((A B) (C D) (E F))))))
```

Figura 23. Ejercicio 4.4.f

```
CL-USER 44 > (cons (car nil) (cdr nil))
(NIL)
```

Figura 24. Ejercicio 4.4.g

```
CL-USER 48 > (cdr (car (cdr (car '((d (e f)) g (h i »
))))))
(F)
```

Figura 25. Ejercicio 4.5.a

```
CL-USER 49 > (setq a '(+ 3 6))
(+ 3 6)
Error while reading: Unmatched right parenthesis.
```

Figura 26. Ejercicio 4.5.b

Se corrige el ejercicio anterior para continuar con los ejercicios planteados para esta asignación

```
CL-USER 51 > (setq a '(+ 3 6))
(+ 3 6)

CL-USER 52 > (cdr a)
(3 6)
```

Figura 27. Ejercicio 4.5.b.1

```
CL-USER 53 > (car (cdr a))
3
```

Figura 28. Ejercicio 4.5.b.2

```
CL-USER 54 > (car (cdr (cdr a)))
6
```

Figura 29. Ejercicio 4.5.b.3

TALLER 5

6- Escriba secuencias de CAR y CDR para extraer el símbolo MAPACHE de cada una de las siguientes expresiones:

a) (oso gato mapache ardilla)
b) ((oso gato) (mapache ardilla))
c) (((oso) (gato) (mapache) (ardilla)))
d) (oso (gato) ((mapache)) (((ardilla))))

Figura 30. Taller 5

```
CL-USER 57 > (car(cdr(cdr '(oso gato mapache ardill »
a))))
MAPACHE
```

Figura 31. Ejercicio 5.6.a

```
CL-USER 58 > (car (car(cdr '((oso gato) (mapache ar »
dilla)))))
MAPACHE
```

Figura 32. Ejercicio 5.6.b

```
CL-USER 59 > (car(car(cdr(cdr(car'(((oso) (gato) (m »
apache) (ardilla)))))))
MAPACHE
```

Figura 33. Ejercicio 5.6.c

```
CL-USER 60 > (car(car(car(cdr(cdr '(oso (gato) ((ma »
pache)) (((ardilla)))))))))
MAPACHE
```

Figura 34. Ejercicio 5.6.d

```
CL-USER 61 > (second (cdr (cdr '(managua chinandega »
 rivas león doaco))))
LEÓN
```

Figura 35. Ejercicio 6.1.a

```
CL-USER 63 > (nth 1(nthcdr 1(car(cdr '((managua) (c »
hinandega rivas león) boaco)))))
LEÓN
```

Figura 36. Ejercicio 6.1.b

```
CL-USER 66 > (second(second(second '(managua(chinan »
dega(rivas león boaco))))))
LEÓN

CL-USER 67 >
```

Figura 37. Ejercicio 6.1.c

```
CL-USER 69 > (car(car(third '(deportes (béisbol ten »
is)((fútbol) billar)))))
FÚTBOL
```

Figura 38. Ejercicio 6.1.d

```
CL-USER 70 > (cons (second '(leo mana china)) (cons »
 (list(car '(1 2 3 4)) (cdr '(a b c d))) '2))
(MANA (1 (B C D)) . 2)
```

Figura 39. Ejercicio 7.2.a

```
CL-USER 83 > (list (list '(ca ce) (last '(0 a 1 b c »
i) (third '(5 4 3 2 1))) (nth 3 '(pa pe pi po pu)))
```

Figura 40. Ejercicio 7.2.b

Se agrega un paréntesis para corregir el ejercicio

```
CL-USER 83 > (list (list '(ca ce) (last '(0 a 1 b c »
i) (third '(5 4 3 2 1))) (nth 3 '(pa pe pi po pu)))

)
(((CA CE) (1 B CI) PO))

CL-USER 84 >
```

Figura 41. Eejercicio 7.2.b corregido

```
CL-USER 84 > (list (append '(h o l a) '(m u n d o)) »
 (cdr '(sal pan arroz pollo)) (car '(buen mal)) (ca »
ddr '(desayuno recreo almuerzo cena)))
((H O L A M U N D O) (PAN ARROZ POLLO) BUEN ALMUERZ »
O)
```

Figura 42. Ejercicio 7.2.c

```
CL-USER 85 > (list '(¿como estas?) (nth 0 '(bien ma »
l rematado)) (append (car '((estas) estoy)) (cdr '( »
entiendo entendiendo lisp))))
((¿COMO ESTAS?) BIEN (ESTAS ENTENDIENDO LISP))
```

Figura 43. Ejercicio 7.2.d

```
CL-USER 86 > (length (cons (car '(verdad mentira fa »
lso)) (list* 'es 'muy '(facil))))
4
```

Figura 44. Ejercicio 7.2.e

```
CL-USER 87 > (cdr (list (subseq '(z y x w v) 0 2) ( »
cons '(a e i) '(o u)))

```

Figura 45. Ejercicio 7.2.f

Se agrega un paréntesis para corregir el ejercicio

```
CL-USER 87 > (cdr (list (subseq '(z y x w v) 0 2) ( »
cons '(a e i) '(o u)))

)
(((A E I) O U))
```

Figura 46. Ejercicio 7.2.f corregido

```
CL-USER 89 > (list 'nicaragua 'italia `España (subl »
is paises '(nicaragua italia españa)) (nth 1 '(eran »
 son serán)) (car (nthcdr 1(cdr '(pueblos estados c »
apitales barrios)))) (cons 'de (cons `estos (cons ' »
paises nil))))
(NICARAGUA ITALIA ESPAÑA (MANAGUA ROMA MADRID) SON »
CAPITALES (DE ESTOS PAISES))

CL-USER 90 >
```

Figura 47. Ejercicio 8.3

```
CL-USER 92 > (setq palabras '(grande bonito feliz húmedo)
sinónimos '(alto bello contento mojado)
antónimos '(pequeño feo triste seco)
ingles '(big beautiful happy humid))
(BIG BEAUTIFUL HAPPY HUMID)

CL-USER 93 > (setq palabras-sinonimos (pairlis palabras s»
inónimos))
((HÚMEDO . MOJADO) (FELIZ . CONTENTO) (BONITO . BELLO) (G»
RANDE . ALTO))

CL-USER 94 > (setq palabras-antónimos (pairlis palabras a»
ntónimos))
((HÚMEDO . SECO) (FELIZ . TRISTE) (BONITO . FEO) (GRANDE »
. PEQUEÑO))

CL-USER 95 > (setq palabras-ingles (pairlis palabras ingl»
es))
((HÚMEDO . HUMID) (FELIZ . HAPPY) (BONITO . BEAUTIFUL) (G»
RANDE . BIG))
```

Figura 48. Ejercicio 9.4.a

```
CL-USER 96 > (cdr (assoc grande palabras-ingles))

Error: The variable GRANDE is unbound.
  1 (continue) Try evaluating GRANDE again.
  2 Specify a value to use this time instead of evaluatin»
g GRANDE.
  3 Specify a value to set GRANDE to.
  4 (abort) Return to level 0.
  5 Return to top loop level 0.
```

Figura 49. Ejercicio 9.4.b.1

Se corrige el ejercicio

```
CL-USER 98 > (cdr (assoc 'grande palabras-ingles))
BIG
```

Figura 50. Ejercicio 9.4.b.1 corregido

```
CL-USER 99 > (cdr (assoc feliz palabras-sinónimos))

Error: The variable FELIZ is unbound.
  1 (continue) Try evaluating FELIZ again.
  2 Specify a value to use this time instead of evaluatin»
g FELIZ.
  3 Specify a value to set FELIZ to.
  4 (abort) Return to level 0.
  5 Return to top loop level 0.
```

Figura 51. Ejercicio 9.4.b.2

Se corrige el ejercicio

```
CL-USER 103 > (cdr (assoc 'feliz palabras-sinonimos))
CONTENTO
```

Figura 52. Ejercicio 9.4.b.2 corregido

```
CL-USER 104 > (first (assoc humedo(acons pal anto palabra»
s-antonimos)))

Error: The variable HUMEDO is unbound.
  1 (continue) Try evaluating HUMEDO again.
  2 Specify a value to use this time instead of evaluatin»
g HUMEDO.
  3 Specify a value to set HUMEDO to.
  4 (abort) Return to level 0.
  5 Return to top loop level 0.
```

Figura 53. Ejercicio 9.4.b.3

Se corrige el ejercicio

```
CL-USER 108 > (first (assoc 'humedo(acons 'pal 'anto pala»
bras-antónimos)))
NIL
```

Figura 54. Ejercicio 9.4.b.3

```
CL-USER 108 > (first (assoc 'humedo(acons 'pal 'anto pala»
bras-antónimos)))
NIL

CL-USER 109 > (last (cons (length (acons lenguaje ingles »
palabras ingles)) (cons `(agregando) (cons `(palabras) (l»
ist `(ala) `(lista-asoc))))
```

Figura 55. Ejercicio 9.4.b.4

Se corrige el ejercicio

```
CL-USER 111 > (last (cons (length (acons 'lenguaje 'ingle»
s palabras-ingles)) (cons '(agregando) (cons '(palabras) »
(list '(ala) '(lista-asoc))))))
((LISTA-ASOC))
```

Figura 56. Ejercicio 9.4.b.4 corregido

D. Conclusión

Se evidenciaron múltiples estructuras de expresiones simbólicas utilizadas en el lenguaje LISP, junto con diferentes funciones propias del lenguaje.

REFERENCIAS

Referencias en la Web:

http://www.lispworks.com/documentation/lw50/CLHS/Front/Contents.htm

http://clhs.lisp.se/