AI Solution: Basic List Processing

Nahyeon Kim

This very modest programming exercise involves three Lisp sessions that feature aspects of basic list processing. Two of these are reproductions, and the is created from specification.

Task 1: Mimic “Lisp Session: CAR, CDR and CONS”

[1]> (car '(blue red yellow))

BLUE

[2]> (cdr '(blue red yellow))

(RED YELLOW)

[3]> (car '((1 2) buckle my shoe))

(1 2)

[4]> (cdr '((1 2) buckle my shoe))

(BUCKLE MY SHOE)

[5]> (car '("sunshine"))

"sunshine"

[6]> (cdr '("sunshine"))

NIL

[7]> (cons 'ESPRESSO '(LATTE CAPPUCCINO))

(ESPRESSO LATTE CAPPUCCINO)

[8]> (cons '(A B C) '(1 2 3))

((A B C) 1 2 3)

[9]> (cons 'SYMBOL '())

(SYMBOL)

[10]> (bye)

Bye.

Task 2: Mimic "Redacted Lisp Session: Three additional referencers and constructors"

[1]> (setf oo-languages '(simula smalltalk java clos))

(SIMULA SMALLTALK JAVA CLOS)

[2]> oo-languages

(SIMULA SMALLTALK JAVA CLOS)

[3]> 'oo-languages

OO-LANGUAGES

[4]> (quote oo-languages)

OO-LANGUAGES

[5]> (car oo-languages)

SIMULA

[6]> (cdr oo-languages)

(SMALLTALK JAVA CLOS)

[7]> (car (cdr oo-languages))

SMALLTALK

[8]> (cdr (cdr oo-languages))

(JAVA CLOS)

[9]> (cadr oo-languages)

SMALLTALK

[10]> (cddr oo-languages)

(JAVA CLOS)

[11]> (first oo-languages)

SIMULA

[12]> (second oo-languages)

SMALLTALK

[13]> (third oo-languages)

JAVA

[14]> (nth 2 oo-languages)

JAVA

```
[15]> (setf numbers '(1 2 3))
(1 2 3)
[16]> (setf letters '(a b c))
(A B C)
[17]> (cons numbers letters)
((1 2 3) A B C)
[18]> (list numbers letters)
((1 2 3) (A B C))
[19]> (append numbers letters)
(1 2 3 A B C)
[20]> (list numbers (cdr numbers) (cddr numbers))
((1 2 3) (2 3) (3))
[21]> (append numbers (cdr numbers) (cddr numbers))
(1 2 3 2 3 3)
[22]> (setf elle '(anr bat cat dog eel))
(ANR BAT CAT DOG EEL)
[23]> (car (cdr (cdr (cdr elle))))
DOG
[24]> (nth 3 elle)
DOG
[25]> (setf a 'apple b 'peach c 'cherry)
CHERRY
[26]> (cons a (cons b (cons c ())))
(APPLE PEACH CHERRY)
[27]> (list a b c)
(APPLE PEACH CHERRY)
[28]> (setf x '(red blue) y '(green yellow))
(GREEN YELLOW)
[29]> (cons (car x) (cons (car (cdr x)) y))
```

(RED BLUE GREEN YELLOW)

[30]> (append x y)

(RED BLUE GREEN YELLOW)

[31]> (bye)

Bye.

Task 3: Create a Lisp session according to specification

---

[1]> (setf ENGLISH '(ONE TWO THREE FOUR))

(ONE TWO THREE FOUR)

[2]> (setf FRENCH '(UN DEUX TROIS QUARTE))

(UN DEUX TROIS QUARTE)

[3]> (setf PAIR1 (list (car ENGLISH) (car FRENCH)))

(ONE UN)

[4]> (setf PAIR2 (list (car (cdr ENGLISH)) (car (cdr FRENCH))))

(TWO DEUX)

[5]> (setf PAIR3 (list (nth 2 ENGLISH) (nth 2 FRENCH)))

(THREE TROIS)

[6]> (setf PAIR4 (list (nth 3 ENGLISH) (nth 3 FRENCH)))

(FOUR QUARTE)

[7]> (setf DICTIONARY (list PAIR1 PAIR2 PAIR3 PAIR4))

((ONE UN) (TWO DEUX) (THREE TROIS) (FOUR QUARTE))

[8]> (setf EF-WORDS (append PAIR1 PAIR2 PAIR3 PAIR4))

(ONE UN TWO DEUX THREE TROIS FOUR QUARTE)

[9]> (setf ALT-WORDS (append ENGLISH FRENCH))

(ONE TWO THREE FOUR UN DEUX TROIS QUARTE)

[10]> (bye)

Bye.