FUNCIONES DE CONSTRUCCIÓN DE LISTAS	OBSERVACIONES
(append lista ...) *Toma cualquier número de listas y las ejecuta como una sola*	* Devuelve la lista compuesta por las listas argumento.
(cons primer_elemento_nuevo lista) *El constructor de lista básico: incorpora un elemento en cabeza de una lista*	• La función cons toma un elemento (primer_elemento_nuevo) y una lista y devuelve la suma de dicho elemento al principio de la lista. El primer elemento puede ser un átomo o una lista. • La función cons también acepta un átomo en lugar del argumento lista, en cuyo caso construye una estructura denominada par punteado. En los pares punteados, AutoLISP incluye un punto entre el primer y el segundo elemento. Para obtener el segundo átomo de un par punteado, puede utilizar la función cdr.
(list expr ...) *Recupera cualquier número de expresiones y las combina en una lista*	• En AutoLISP, esta función suele utilizarse para definir una variable de punto 2D o 3D (una lista de dos o tres números reales). • Como alternativa a utilizar list, puede indicar una lista de forma explícita con la función quote si la lista no contiene variables ni opciones sin definir. El carácter de comilla (') se define como la función quote.

FUNCIONES DE ACCESO A ELEMENTOS DE LISTAS	OBSERVACIONES
(assoc elemento lista_asoc) *Busca una lista de asociaciones de un elemento y devuelve la entrada asociada de la lista*	• Busca en la lista de asociaciones lista_asoc el elemento como elemento clave y devuelve la entrada lista_asoc. Si assoc no encuentra el elemento como clave de lista_asoc, devuelve nil. • Las listas de asociaciones se suelen utilizar para almacenar los datos a los que puede acceder mediante una clave. La función subst proporciona un medio apropiado para sustituir el valor asociado con una clave en una lista de asociaciones.
(acad_strlsort lista) *Ordena alfabéticamente una lista de cadenas*	• El argumento lista corresponde a la lista de las cadenas que se van a ordenar. La función acad_strlsort devuelve una lista con las mismas cadenas en orden alfabético. Si el argumento lista no es válido o no se dispone de memoria suficiente para realizar la ordenación, acad_strlsort devuelve nil.
(car lista) y (cdr lista) *CAR devuelve el primer elemento de una lista* *CDR devuelve una lista que contenga todos los elementos excepto el primero*	• Si lista está vacía, car devuelve nil. • Si lista está vacía, cdr devuelve nil. • Cuando el argumento lista es un par de puntos (véase "cons"), cdr devuelve el segundo elemento sin incluirlo en una lista. • AutoLISP permite concatenar las funciones car y cdr hasta en cuatro niveles. Por ejemplo: (CADR lista) equivale a (CAR(CDR lista))
(last lista) *Devuelve el último elemento de una lista*	
(nth n lista) *Devuelve el elemento número n de una lista*	• El argumento n es el número del elemento que debe devolverse (cero es el primer elemento). Si n es mayor que el número de elemento más de la lista, nth devuelve nil.

EJEMPLOS DE FUNCIONES DE CONSTRUCCIÓN DE LISTAS

(SETQ L (LIST 10 20 (LIST 1 2) "ABC"))
(10 20 (1 2) "ABC")
(CAR L)
10
(CADR L)
20
(CADDR L)
(1 2)
(CAADDR L)
1
(LAST L)
"ABC"
(NTH 1 L)
20
(NTH 0 (NTH 2 L))
1

(SETQ LASSOC (LIST (CONS 0 "DATO1" (CONS 1 "DATO2" (LIST 2 "DATO3" "DATO4")))
((0 . "DATO1") (1 . "DATO2") (2 "DATO3" "DATO4"))
(SETQ D1 (ASSOC 1 LASSOC))
(1 . "DATO2")
(SETQ D2 (CDR (ASSOC 0 LASSOC)))
"DATO1"

Mecanismo básico:

LIST integra los argumentos en una lista
CONS inserta el primer argumento como primer elemento de la lista que se refiere como segundo argumento de la función
APPEND ;funde listas.

(SETQ L (LIST 10 20.0 30 "ABC"))
(10 20.0 30 "ABC")
(SETQ L2 (CONS 23 L))
(23 10 20.0 30 "ABC")
(SETQ L3 (APPEND L L2))
(10 20.0 30 "ABC" 23 10 20.0 30 "ABC")

Observación: las estructuras de lista pueden ser tomadas como argumento simple, de modo que LIST puede fundir listas y CONS puede incorporar una lista como primer elemento de una lista dada. Es tarea del programador la selección de la función adecuada para conseguir la estructura pretendida. Como ejemplo, las tres instrucciones que se proponen a continuación operan con los argumentos propuestos devolviendo soluciones distintas, pero correctas desde el punto de vista de la ejecución de la orden:

(setq s1 (list l l2))
((10 20.0 30 "ABC") (23 10 20.0 30 "ABC"))
(setq s2 (cons l l2))
((10 20.0 30 "ABC") 23 10 20.0 30 "ABC")
(setq s3 (append l l2))
(10 20.0 30 "ABC" 23 10 20.0 30 "ABC")