INDICE

1. El Manejo de Símbolos	1
Manejar símbolos es como trabajar con palabras y oraciones * LISP contribuye a hacer que los computadores sean inteligentes * LISP promueve la productividad y facilita la educación * LISP es el lenguaje para el manejo de símbolos que se aprender *COMMON LISP que se debe aprender *Cuidado con los mitos *Resumen * Referencias	
2. Primitivas básicas de LISP	9
LISP significa manejo de símbolos *Los procedimientos y datos en LISP son expresiones simbólicas *Problemas *Las listas son semejantes a recipientes *FIRST y REST descomponen listas *Los apóstrofes detienen la evaluación * Problemas * Algunos programadores veteranos continúan usando CAR y CDR * SETF asigna valor a los símbolos *SETF acepta múltiples parejas de símbolo y valor *Algunos átomos se autoevalúan *CONS, APPEND y LIST construye listas *Problemas *CONS; APPEND y LISP no alteran el valor de los símbolos *NTHCDR, BUTLAST y LAST acortan listas * LENGTH y REVERSE actúan sobre los elementos del nivel superior *Problemas *ASSOC accede a sublistas * LISP opera con números enteros, racionales y de punto flotante, entre otros *Otras primitivas que operan con números completan un repertorio básico *Problemas * Resumen	
3. Definición de procedimientos y ligaduras	39
DEFUN es la primitiva de LISP para definir procedimientos * Los valores de la parámetros se aíslan con cercas virtuales * Los valores de variables especiales no se aíslan con cercas virtuales * Los procedimientos hacen corresponder parámetros con valores iniciales *Las formas LET evalúan en paralelo las formas para valores iniciales *La envoltura progresiva y la traducción de comentarios ayudan a definir nuevos procedimientos * Resumen*	
4. Predicados y condicionales	53
Un predicado es un procedimiento que devuelve un valor verdadero o falso *EQUAL, EQ, EQL e= son predicados de igualdad * MEMBER verifica qué elementos son miembros de una lista *El uso de argumentos clave modifica el comportamiento *LISTP, ATOM, NUMBERP y SYMBOLP son predicados para tipos de datos *NIL es equivalente a lista vacía *NULL y ENDP son predicados para la lista vacía *Existen varios predicados para números *Problemas AND, OR y NOT permiten pruebas más elaboradas * Problemas * Los predicados ayudan a IF, WHEN y UNLESS a elegir entre varias alternativas *Problemas * CASE es otra forma condicional * Las condiciones de problemas ayuda a definir nuevos procedimientos * Resumen	
5. Abstracción de procedimientos y recursión	75
la abstracción de procedimientos oculta detalles detrás de barreras de abstracción *La recursión permite a los procedimientos usarse a sí mismo *La recursión puede ser eficiente *Problemas *La recursión puede usarse para analizar expresiones anidadas *Problemas * Los parámetros opcionales eliminan la necesidad de muchos auxiliares *Problemas *Los programadores experimentados usan parámetros restantes, calve auxiliares *Sólo necesarias unas cuantas primitivas de LISP *Problemas	

*Resumen * Referencias *	
6. Abstracción de datos y transformaciones	101
Los detalles de os datos entorpecen el avance * La abstracción de datos facilita el avance * se deben usar los procedimientos de acceso con liberalidad * Es útil transformar y filtrar *Los procedimientos recursivos permiten transformar y filtrar * Los procedimientos recursivos permiten contra y localizar *Problemas *Los clichés incorporan importantes conocimiento de programación *MAPCAR simplifica las operaciones de transformación *REMOVE – IF y REMOVE – IF – NOT simplifican las operaciones de filtración – CONT – IF y FIND – IF simplificar las operaciones de conteo y localización *FUNCALL y APPY también toman un procedimiento como argumento *LAMBDA define procedimientos anónimos * Resumen *Referencias	
7. Iteración sobre números y sobre listas	121
DOTIMES permiten efectuar iteración sobre números *Problemas * DOLIST permite efectuar iteración sobre lista * Problemas *DO es más general que DOLIST y DOTIMES *Problemas *LOOP casi nunca se detiene *PROG1 y PROGN manejan secuencias de manera explícita *Los problemas y su representación determinan la estrategia de control apropiada *Resumen	
8. Edición, compilación y carga de archivos	131
Los programas y los datos residen en archivos *Las especificaciones de archivos tienden a tener formas barrocas *ED nos lleva a un editor *EMACS es un editor especialmente eficaz para LISP *Compile – FILE compila archivos *LOAD ocasiona que LISP lea de archivos *Resumen *Referencias	
9. Lectura y escritura	139
PRINT y READ facilitan la impresión y la lectura de tipo sencillo * FORMAT permite realizar impresiones elegantes * Problemas *WITH – OPEN – FILE permite leer de archivos *Los argumentos opcionales en las formas READ especifican el tratamiento para el final de archivo * WITH – OPEN –FILE permite escribir en archivos *Problemas *READ no evalúan expresiones, pero EVAL y caracteres *READ LINE y READ – CHAR leen cadenas y caracteres *Resumen *	
10. Reglas para una buena práctica de programación y herramientas de depuración	155
Cumplir con las reglas para una buena práctica de programación ayuda a evitar errores *Los programas grandes requieren abstracción y modularidad *La mayoría de los programadores usan TRACE, STEP y BREAK con variada frecuencia * TRACE ocasiona que los procedimientos impriman sus argumentos y valores * STEP propia que la ejecución de los procedimiento proceda paso a paso *BREAK detiene la evaluación a fin de que se puedan evaluar formas *TIME, describe y DRIBBLE también son útiles *LA depuración de la instalación *Resumen *Referencias	
11. Propiedades y arreglos	169
Cada manera de almacenar datos cuenta con procedimientos constructores, de lectura y escritura *Las propiedades permiten efectuar el almacenamiento en lugares identificados con índices simbólicos *GET y SETF son los guardianes de las propiedades *Problemas * Los	

arreglos permiten efectuarse el almacenamiento en lugares identificados con índices numéricos *MAKE – ARRAY, AREF y SETF son los guardianes de los arreglos * Problemas *Resumen	
12. Marcos y apóstrofo invertido	177
Los macros traducen y luego evalúan *El mecanismo de apóstrofo invertido simplifica el llenado de planillas * el mecanismo de apóstrofo invertido simplifica la escritura de macros *Los parámetros opcionales, restantes y clave, permiten incrementar la capacidad de los macros *Problemas macros merecen su propio archivo *Resumen	
13. Estructura	185
La estructura facilitan la abstracción de datos *Las estructuras permiten el almacenamiento en lugares indizados por procedimientos * las estructuras son nuevos tipos de datos *Problemas * Una estructura puede incluir los campos de otra *Problemas *La estructuras son componentes importantes de grandes sistemas *DESCRIBE imprime descripciones *DEFSTRUCT merece su propio archivo * Resumen	
14. Clases y funciones genéricas	193
Qué hacer depende de los que se haga *Se pueden diseñar procedimientos comunes guiados por datos, aunque no resulten muy eficientes *Los métodos son procedimientos seleccionados a partir de funciones genéricas por parte del tipo de los argumentos *Las clases se parecen a las estructuras pero resuenan mejor como funciones genéricas * Cualquier clase de argumentos no opcional puede ayudar a seleccionar un método *Las clases permiten heredad métodos *Problemas *Los métodos más específicos tienen prioridad sobre los otros *El orden de los parámetros ayuda a determinar la prioridad del método *Las reglas simples se aproximan a algoritmo complejo de prioridad de clases *Problemas *Los métodos pueden especializarse a ocurrencias individuales * Problemas *La elección de métodos implica tres pasos	
14. Clases y funciones genéticas	193
Qué hacer depende de lo que se haga *Se pueden diseñar procedimientos comunes guiados por datos, aunque no resulten de funciones genéticas por parte del tipo de los argumentos *Las clases se parecen a las estructuras pero resuenan mejor con funciones genéticas * Cualquier clase de argumento no opcional puede ayudar a seleccionar un método * las clases permiten heredar métodos *Problemas *Los métodos más específicos tienen prioridad sobre los otros * El orden de los parámetros ayuda a determinar la prioridad del método *Las reglas simples se aproximan al algoritmo complejo de prioridad de clase *Problemas *Los métodos pueden especializarse a ocurrencias individuales *Problemas *La elección de métodos no cualidades mágicas *Resumen *Referencia	
15. Variables léxicas, generadores y encapsulamiento	217
LET produce cercas anidadas * Las cercas anidadas proporcionan valores a variables *Generalmente, las llamadas a los procedimientos no producen cercas anidadas *Problemas *Las definiciones anidadas producen cercas anidadas * Los generadores producen secuencias *Los procedimientos sin nombre producen cercas anidadas *Los procedimientos sin nombre sin nombre pueden encapsularse *El	

encapsulamiento permite la creación de generadores complejos *Los generadores pueden definirse mediante otros procedimientos *Los procedimientos sin nombre se vuelven cerraduras léxicas *Resumen *Referencia	
16. Variables especiales	237
La ligaduras pueden conservarse en un registro de llamadas *Algunas variables son declaradas indefinidamente como especiales *Las ligaduras de variables especiales en realidad se conservan en una pila *DEFVAR puede tanto declarar como asignar *Algunas ocurrencias de variables pueden ser especiales y otras léxicas *Tanto las variables léxicas como las especiales pueden ser libres *Resumen	
17. Almacenamiento, manipulación interna y recuperación de lista	
Las listas pueden representarse con cajas y apuntadores *Las cajas y los apuntadores pueden representarse por bytes *CONS construye nuevas listas depositando apuntadores en las cajas libres * APPEND construye nuevas listas mediante copiado * NCONC y DELETE pueden alterar peligrosamente el contenido de las cajas * Problemas *EQ sólo verifica apuntadores * La recolección de basura permite recuperar la memoria abandonada *LISP permite escribir procedimientos ineficientes *Problemas *Los recolectores de basura sencillos usan el método de marcar y barrer *Los procedimientos de simulación muestra detalles de la recolección de basura *MARCA identifica los cuartetos útiles *BARRE recoge los cuartetos no marcados *La colocación de marcas puede hacerse sin recursión *Nuestro procedimiento no recursivo de colocación de marcas deja un rastro de apuntadores *Algunos recolectores de basura se activan constantemente *Resumen	
18. LISP en LISP	273
Es fácil construir intérprete sencillo para un lenguaje similar a LISP *MICRO –EVAL y MICRO – APPL trabajan conjuntamente para evaluar formas *Las trazas muestran cómo trabaja conjuntamente MICRO – EVAL y MICRO – APPLY *Problemas *Las cerraduras encapsulan ambientes *Problemas *Se pueden manejar ligaduras de variables especiales *LISP hace llamadas por valor, no por referencia *LISP puede definirse en LISP *Las estructuras complejas de control generalmente empiezan como intérprete básicos de LISP *Resumen	
19. Ejemplos que involucran búsqueda	289
La búsqueda de primero en amplitud y primero en profundidad son estrategias básicas *Problemas *Las búsquedas de primero el mejor y escalamiento de colinas requiere ordenamiento *Problemas *Problemas acerca de la medición de un volumen de agua *Problemas de colocación de reinas en un tablero de ajedrez * Resumen *Referencias	