

# **PROGRAMACION EN C**

por JOYANES AGUILAR

Isbn 8448198441

## **Indice del Contenido**

Prólogo a la 2º edición

Prólogo a la 1º edición

### **PARTE I. METODOLOGÍA DE LA PROGRAMACIÓN**

#### **Capítulo 1. Introducción a las computadoras y a los lenguajes de programación**

- 1.1. ¿Qué es una computadora?
  - 1.1.1. Origen de las computadoras
  - 1.1.2. Clasificación de las computadoras
- 1.2. Organización física de una computadora
  - 1.2.1. Dispositivos de Entrada/Salida (E/S)
  - 1.2.2. La memoria principal
  - 1.2.3. El procesador
- 1.3. Dispositivos de almacenamiento de información
  - 1.3.1. Discos magnéticos
  - 1.3.2. Discos ópticos: CD-ROM y DVD
  - 1.3.3. Discos y Memorias Flash USB
  - 1.3.4. Otros dispositivos de Entrada y Salida (E/S)
- 1.4. Conectores de dispositivos de E/S
  - 1.4.1. Puertos serie y paralelo
  - 1.4.2. USB
  - 1.4.3. Bus IEEE 1394 - Firewire
- 1.5. Redes e Internet
  - 1.5.1. Redes P2P, igual-a-igual (peer-to-peer, P2P)
  - 1.5.2. Aplicaciones de las redes de comunicaciones
  - 1.5.3. Módem
  - 1.5.4. Internet y la World Wide Web
- 1.6. Software (los programas)
  - 1.6.1. Software del sistema
  - 1.6.2. Software de aplicación
- 1.7. Lenguajes de programación
  - 1.7.1. Evolución de los lenguajes de programación
  - 1.7.2. Paradigmas de programación
- 1.8. El lenguaje C: Historia y características
  - 1.8.1. Ventajas de C
  - 1.8.2. Características técnicas de C
- 1.9. Breve historia de los herederos de C: C++, Java y C#
  - 1.9.1. C++: el heredero natural
  - 1.9.2. Java: el paradigma de los lenguajes en Internet
  - 1.9.3. C# ya no tan joven
  - 1.9.4. ¿Por qué Java y C# son importantes para Internet?
- 1.10. Resumen

#### **Capítulo 2. Metodología de la programación y desarrollo de software**

- 2.1. Fases en la resolución de problemas
  - 2.1.1. Análisis del problema
  - 2.1.2. Diseño del algoritmo
  - 2.1.3. Herramientas de programación

- 2.1.4. Codificación de un programa
- 2.1.5. Compilación y ejecución de un programa
- 2.1.6. Verificación y depuración de un programa
- 2.1.7. Documentación y mantenimiento
- 2.2. Programación modular
- 2.3. Programación estructurada
  - 2.3.1. Recursos abstractos
  - 2.3.2. Diseño descendente (top-down)
  - 2.3.3. Estructuras de control
  - 2.3.4. Teorema de la programación estructurada: estructuras básicas
- 2.4. Concepto y características de algoritmos
  - 2.4.1. Características de los algoritmos
  - 2.4.2. Diseño del algoritmo
- 2.5. Escritura de algoritmos
- 2.6. Representación gráfica de los algoritmos
  - 2.6.1. Pseudocódigo
  - 2.6.2. Diagramas de flujo
  - 2.6.3. Diagramas de Nassi-Schneiderman (N-S)
- 2.7. El ciclo de vida del software
  - 2.7.1. Análisis
  - 2.7.2. Diseño
  - 2.7.3. Implementación (codificación)
  - 2.7.4. Pruebas e integración
  - 2.7.5. Verificación
  - 2.7.6. Mantenimiento
  - 2.7.7. La obsolescencia: programas obsoletos
  - 2.7.8. Iteración y evolución del software
  - 2.7.9. Factores en la calidad del software
- 2.8. Métodos formales de verificación de programas
  - 2.8.1. Aserciones
  - 2.8.2. Precondiciones y postcondiciones
  - 2.8.3. Reglas para prueba de programas
  - 2.8.4. Invariantes de bucles
  - 2.8.5. Etapas a establecer la exactitud (corrección) de un programa
- 2.9. Resumen
- 2.10. Ejercicios
- 2.11. Ejercicios resueltos

## PARTE II. FUNDAMENTOS DE ALGORITMOS Y PROGRAMACIÓN EN C

### Capítulo 3. El lenguaje C: elementos básicos

- 3.1. Estructura general de un programa en C
  - 3.1.1. Directivas del preprocesador
  - 3.1.2. Declaraciones globales
  - 3.1.3. Función main ()
  - 3.1.4. Funciones definidas por el usuario
  - 3.1.5. Comentarios
- 3.2. Creación de un programa
- 3.3. El proceso de ejecución de un programa en C
- 3.4. Depuración de un programa en C
  - 3.4.1. Errores de sintaxis
  - 3.4.2. Errores lógicos
  - 3.4.3. Errores de regresión
  - 3.4.4. Mensajes de error
  - 3.4.5. Errores en tiempo de ejecución
- 3.5. Pruebas

- 3.6. Los elementos de un programa en C
  - 3.6.1. Tokens (elementos léxicos de los programas)
  - 3.6.2. Identificadores
  - 3.6.3. Palabras reservadas
  - 3.6.4. Comentarios
  - 3.6.5. Signos de puntuación y separadores
  - 3.6.6. Archivos de cabecera
- 3.7. Tipos de datos en C
  - 3.7.1. Enteros (int)
  - 3.7.2. Tipos de coma flotante (float/double)
  - 3.7.3. Caracteres (char)
- 3.8. El tipo de dato lógico
  - 3.8.1. Escritura de valores lógicos
- 3.9. Constantes
  - 3.9.1. Constantes literales
  - 3.9.2. Constantes definidas (simbólicas)
  - 3.9.3. Constantes enumeradas
  - 3.9.4. Constantes declaradas const y volatile
- 3.10. Variables
  - 3.10.1. Declaración
  - 3.10.2. Inicialización de variables
  - 3.10.3. Declaración o definición
- 3.11. Duración de una variable
  - 3.11.1. Variables locales
  - 3.11.2. Variables globales
  - 3.11.3. Variables dinámicas
- 3.12. Entradas y salidas
  - 3.12.1. Salida
  - 3.12.2. Entrada
  - 3.12.3. Salida de cadenas de caracteres
  - 3.12.4. Entrada de cadenas de caracteres
- 3.13. Compilación de Programas C en UNIX y LINUX
  - 3.13.1. Orden de compilación cc
- 3.14. Resumen
- 3.15. Ejercicios

#### Capítulo 4. Operadores y expresiones

- 4.1. Operadores y expresiones
- 4.2. Operador de asignación
- 4.3. Operadores aritméticos
  - 4.3.1. Asociatividad
  - 4.3.2. Uso de paréntesis
- 4.4. Operadores de incrementación y decrementación
- 4.5. Operadores relacionales
- 4.6. Operadores lógicos
  - 4.6.1. Evaluación en cortocircuito
  - 4.6.2. Asignaciones booleanas (lógicas)
- 4.7. Operadores de manipulación de bits
  - 4.7.1. Operadores de asignación adicionales
  - 4.7.2. Operadores de desplazamiento de bits > >, <<
    - \*\*\*\*\* con o sin espacio >>
  - 4.7.3. Operadores de direcciones
- 4.8. Operador condicional
- 4.9. Operador coma
- 4.10. Operadores especiales: () , []
  - \*\*\*\*\* es asi la coma?

- 4.10.1. El operador ()
- 4.10.2. El operador []
- 4.11. El operador sizeof
- 4.12. Conversiones de tipos
- 4.12.1. Conversión implícita
- 4.12.2. Reglas
- 4.13. Prioridad y asociatividad
- 4.14. Resumen
- 4.15. Ejercicios
- 4.16. Problemas

## Capítulo 5. Estructuras de selección: sentencias IF y SWITCH

- 5.1. Estructuras de control
- 5.2. La sentencia if
- 5.3. Sentencia if de dos alternativas: if-else
- 5.4. Sentencias if-else anidadas
- 5.4.1. Sangría en las sentencias if anidadas
- 5.4.2. Comparación de sentencias if anidadas y secuencias de sentencias if
- 5.5. Sentencia de control switch
- 5.5.1. Caso particular de case
- 5.5.2. Uso de sentencias switch en menús
- 5.6. Expresiones condicionales: el operador ?:
- 5.7. Evaluación en cortocircuito de expresiones lógicas
- 5.8. Puesta a punto de programas
- 5.9. Errores frecuentes de programación
- 5.10. Resumen
- 5.11. Ejercicios
- 5.12. Problemas

## Capítulo 6. Estructuras de control: bucles

- 6.1. Sentencia while
- 6.1.1. Operadores de incremento y decremento (++ , --)
- 6.1.2. Terminaciones anormales de un bucle (ciclo)
- 6.1.3. Diseño eficiente de bucles
- 6.1.4. Bucles while con cero iteraciones
- 6.1.5. Bucles controlados por centinelas
- 6.1.6. Bucles controlados por indicadores (banderas)
- 6.1.7. La sentencia break en los bucles
- 6.1.8. Bucles while (true)
- 6.2. Repetición: el bucle for
- 6.2.1. Diferentes usos de bucles for
- 6.3. Precauciones en el uso de for
- 6.3.1. Bucles infinitos
- 6.3.2. Los bucles for vacíos
- 6.3.3. Sentencias nulas en bucles for
- 6.3.4. Sentencias break y continue
- 6.4. Repetición: el bucle do-while
- 6.4.1. Diferencias entre while y do-while
- 6.5. Comparación de bucles while, for y do-while
- 6.6. Diseño de bucles
- 6.6.1. Bucles para diseño de sumas y productos
- 6.6.2. Final de un bucle
- 6.6.3. Otras técnicas de terminación de bucle
- 6.6.4. Bucles for vacíos
- 6.7. Bucles anidados
- 6.8. Enumeraciones

- 6.9. Resumen
- 6.10. Ejercicios
- 6.11. Problemas

## Capítulo 7. Funciones

- 7.1. Concepto de función
- 7.2. Estructura de una función
  - 7.2.1. Nombre de una función
  - 7.2.2. Tipo de dato de retorno
  - 7.2.3. Resultados de una función
  - 7.2.4. Llamada a una función
- 7.3. Prototipos de las funciones
  - 7.3.1. Prototipos con un número no especificado de parámetros
- 7.4. Parámetros de una función
  - 7.4.1. Paso de parámetros por valor
  - 7.4.2. Paso de parámetros por referencia
  - 7.4.3. Diferencias entre paso de variables por valor y por referencia
  - 7.4.4. Parámetros const de una función
- 7.5. Funciones en línea: macros
  - 7.5.1. Creación de macros con argumentos
- 7.6. Ámbito (alcance) de una variable
  - 7.6.1. Ámbito del programa
  - 7.6.2. Ámbito del archivo fuente
  - 7.6.3. Ámbito de una función
  - 7.6.4. Ámbito de bloque
  - 7.6.5. Variables locales
- 7.7. Clases de almacenamiento
  - 7.7.1. Variables automáticas
  - 7.7.2. Variables externas
  - 7.7.3. Variables registro
  - 7.7.4. Variables estáticas
- 7.8. Concepto y uso de funciones de biblioteca
- 7.9. Funciones de carácter
  - 7.9.1. Comprobación alfabética y de dígitos
  - 7.9.2. Funciones de prueba de caracteres especiales
  - 7.9.3. Funciones de conversión de caracteres
- 7.10. Funciones numéricas
  - 7.10.1. Funciones matemáticas de carácter general
  - 7.10.2. Funciones trigonométricas
  - 7.10.3. Funciones logarítmicas y exponenciales
  - 7.10.4. Funciones aleatorias
- 7.11. Funciones de fecha y hora
- 7.12. Funciones de utilidad
- 7.13. Visibilidad de una función
  - 7.13.1. Variables locales frente a variables globales
  - 7.13.2. Variables estáticas y automáticas
  - 7.13.3. Variables globales estáticas
- 7.14. Compilación separada
- 7.15. Resumen
- 7.16. Ejercicios
- 7.17. Problemas

## Capítulo 8. Recursividad

- 8.1. La naturaleza de la recursividad
- 8.2. Funciones recursivas
  - 8.2.1. Recursividad indirecta: funciones mutuamente recursivas

- 8.2.2. Condición de terminación de la recursión
- 8.3. Recursión versus iteración
- 8.3.1. Directrices en la toma de decisión iteración/recursión
- 8.4. Recursión infinita
- 8.5. Algoritmos divide y vence
- 8.5.1. Búsqueda binaria recursiva
- 8.5.2. Torres de Hanoi
- 8.5.3. Ordenación por mezclas: Mergesort
- 8.6. Resumen
- 8.7. Ejercicios
- 8.8. Problemas

## Capítulo 9. Arrays (listas y tablas)

- 9.1. Arrays
- 9.1.1. Declaración de un array
- 9.1.2. Subíndices de un array
- 9.1.3. Almacenamiento en memoria de los arrays
- 9.1.4. El tamaño de los arrays
- 9.1.5. Verificación del rango del índice de un array
- 9.2. Inicialización de un array
- 9.3. Arrays de caracteres y cadenas de texto
- 9.4. Arrays multidimensionales
- 9.4.1. Inicialización de arrays multidimensionales
- 9.4.2. Acceso a los elementos de los arrays bidimensionales
- 9.4.3. Lectura y escritura de elementos de arrays bidimensionales
- 9.4.4. Acceso a elementos mediante bucles
- 9.4.5. Arrays de más de dos dimensiones
- 9.4.6. Proceso de un array de tres dimensiones
- 9.5. Utilización de arrays como parámetros
- 9.5.1. Precauciones
- 9.5.2. Paso de cadenas como parámetros
- 9.6. Operaciones de organización de la información en listas de datos
- 9.6.1. Búsqueda en listas
- 9.7. Resumen
- 9.8. Ejercicios
- 9.9. Problemas

## Capítulo 10. Algoritmos de ordenación y búsqueda

- 10.1. Ordenación
- 10.2. Ordenación por burbuja
- 10.2.1. Algoritmo de la burbuja
- 10.2.2. Codificación del algoritmo de la burbuja
- 10.2.3. Análisis del algoritmo de la burbuja
- 10.3. Ordenación por selección
- 10.3.1. Algoritmo de selección
- 10.3.2. Codificación del algoritmo de selección
- 10.4. Ordenación por inserción
- 10.4.1. Algoritmo de ordenación por inserción
- 10.4.2. Codificación del algoritmo de inserción
- 10.5. Ordenación Shell
- 10.5.1. Algoritmo de ordenación Shell
- 10.5.2. Codificación del método Shell
- 10.6. Ordenación rápida (quicksort)
- 10.6.1. Algoritmo quicksort
- 10.6.2. Codificación del algoritmo quicksort
- 10.6.3. Análisis del algoritmo quicksort

- 10.7. Búsqueda en listas
- 10.7.1. Búsqueda binaria
- 10.7.2. Algoritmo y codificación de la búsqueda binaria
- 10.7.3. Análisis de los algoritmos de búsqueda
- 10.8. Resumen
- 10.9. Ejercicios
- 10.10. Problemas

## Capítulo 11. Estructuras y uniones

- 11.1. Estructuras
- 11.1.1. Declaración de una estructura
- 11.1.2. Definición de variables de estructuras
- 11.1.3. Uso de estructuras en asignaciones
- 11.1.4. Inicialización de una declaración de estructuras
- 11.1.5. El tamaño de una estructura
- 11.2. Acceso a estructuras
- 11.2.1. Almacenamiento de información en estructuras
- 11.2.2. Lectura de información de una estructura
- 11.2.3. Recuperación de información de una estructura
- 11.3. Sinónimo de un tipo de dato: typedef
- 11.4. Estructuras anidadas
- 11.4.1. Ejemplo de estructuras anidadas
- 11.5. Arrays de estructuras
- 11.6. Arrays como miembros
- 11.7. Utilización de estructuras como parámetros
- 11.8. Uniones
- 11.9. Tamaño de estructuras y uniones
- 11.10. Campos de bit
- 11.11. Resumen
- 11.12. Ejercicios
- 11.13. Problemas

## Capítulo 12. Punteros (Apuntadores)

- 12.1. Direcciones en memoria
- 12.2. Concepto de puntero (apuntador)
- 12.2.1. Declaración de punteros
- 12.2.2. Inicialización (iniciación) de punteros
- 12.2.3. Indirección de punteros
- 12.2.4. Punteros y verificación de tipos
- 12.3. Punteros NULL y void
- 12.4. Punteros a punteros
- 12.5. Punteros a arrays
- 12.5.1. Nombres de arrays como punteros
- 12.5.2. Ventajas de los punteros
- 12.6. Arrays de punteros
- 12.6.1. Inicialización de un array de punteros a cadenas
- 12.7. Punteros a cadenas
- 12.7.1. Punteros versus arrays
- 12.8. Aritmética de punteros
- 12.8.1. Una aplicación de punteros: conversión de caracteres
- 12.9. Punteros constantes frente a punteros a constantes
- 12.9.1. Punteros constantes
- 12.9.2. Punteros a constantes
- 12.9.3. Punteros constantes a constantes
- 12.10. Punteros como argumento de funciones
- 12.11. Punteros a funciones

- 12.11.1. Inicialización de un puntero a una función
- 12.11.2. Aplicación de punteros a función para ordenación
- 12.11.3. Arrays de punteros de funciones
- 12.11.4. Una aplicación de punteros de funciones
- 12.12. Punteros a estructuras
- 12.13. Resumen
- 12.14. Ejercicios
- 12.15. Problemas

## Capítulo 13. Asignación dinámica de memoria

- 13.1. Gestión dinámica de la memoria
  - 13.1.1. Almacén libre (free store)
  - 13.2. Función de asignación de memoria malloc ()
  - 13.2.1. Asignación de memoria de un tamaño desconocido
  - 13.2.2. Uso de malloc () para arrays multidimensionales
  - 13.3. La función free ()
  - 13.4. Funciones de asignación calloc () y realloc ()
    - 13.4.1. Función calloc ()
    - 13.4.2. Función realloc ()
  - 13.5. Asignación dinámica para arrays
    - 13.5.1. Asignación de memoria interactivamente
    - 13.5.2. Asignación de memoria para un array de estructuras
  - 13.6. Arrays dinámicos
    - 13.6.1. Comparación de los dos métodos para definir un array
  - 13.7. Reglas de funcionamiento de funciones de asignación dinámica
  - 13.8. Resumen
  - 13.9. Ejercicios
  - 13.10. Problemas

## Capítulo 14. Cadenas

- 14.1. Concepto de cadena
  - 14.1.1. Declaración de variables de cadena
  - 14.1.2. Inicialización de variables de cadena
- 14.2. Lectura de cadenas
  - 14.2.1. Función gets ()
  - 14.2.2. Función getchar ()
  - 14.2.3. Función putchar ()
  - 14.2.4. Función puts ()
- 14.3. La biblioteca string.h
  - 14.3.1. Utilización del modificador const con cadenas
- 14.4. Arrays y cadenas como parámetros de funciones
- 14.5. Asignación de cadenas
  - 14.5.1. Función strncpy ()
- 14.6. Longitud y concatenación de cadenas
  - 14.6.1. Función strlen ()
  - 14.6.2. Funciones strcat () y strncat ()
- 14.7. Comparación de cadenas
  - 14.7.1. Función strcmp ()
  - 14.7.2. Función strncmp ()
- 14.8. Conversión de cadenas a números
  - 14.8.1. Función atoi ()
  - 14.8.2. Función atof ()
  - 14.8.3. Función atol ()
  - 14.8.4. Función strtol () y strtoul ()
  - 14.8.5. Función strtod ()
  - 14.8.6. Entrada de números y cadenas



- 14.9. Búsqueda de caracteres y cadenas
- 14.9.1. Función strchr ()
- 14.9.2. La función strrchr ()
- 14.9.3. Función strspn ()
- 14.9.4. Función strcspn ()
- 14.9.5. Función strpbrk()
- 14.9.6. Función strstr ()
- 14.9.7. Función strtok ()
- 14.10. Resumen
- 14.11. Ejercicios
- 14.12. Problemas

### PARTE III. ESTRUCTURA DE DATOS

#### Capítulo 15. Entradas y salidas por archivos

- 15.1. Flujos
- 15.2. Puntero FILE
- 15.3. Apertura de un archivo
- 15.3.1. Modos de apertura de un archivo
- 15.3.2. NULL y EOF
- 15.3.3. Cierre de archivos
- 15.3.4. Volcado del buffer: fflush ()
- 15.4. Funciones de entrada/salida para archivos
- 15.4.1. Funciones putc () y fputc ()
- 15.4.2. Funciones getc () y fgetc ()
- 15.4.3. Funciones fputs () y fgets ()
- 15.4.4. Funciones fprintf () y fscanf ()
- 15.4.5. Función feof ()
- 15.4.6. Función rewind ()
- 15.5. Archivos binarios en C
- 15.5.1. Función de salida fwrite ()
- 15.5.2. Función de lectura fread ()
- 15.6. Funciones para acceso aleatorio
- 15.6.1. Función fseek ()
- 15.6.2. Función ftell ()
- 15.6.3. Cambio de posición: fgetpos () y fsetpos ()
- 15.7. Datos externos al programa con argumentos de main ()
- 15.8. Resumen
- 15.9. Ejercicios
- 15.10. Problemas

#### Capítulo 16. Organización de datos en un archivo

- 16.1. Registros
- 16.1.1. Clave
- 16.1.2. Registro físico (bloque)
- 16.2. Organización de archivos
- 16.2.1. Organización secuencial
- 16.2.2. Organización directa
- 16.3. Archivos con función de direccionamiento Hash
- 16.3.1. Funciones Hash
- 16.3.2. Características de un archivo con direccionamiento hash
- 16.4. Archivos secuenciales indexados
- 16.4.1. Partes de un archivo secuencial indexado
- 16.4.2. Proceso de un archivo secuencial indexado
- 16.5. Ordenación de archivos: ordenación externa
- 16.5.1. Fusión de archivos

- 16.6. Clasificación por mezcla directa
- 16.6.1. Codificación del algoritmo mezcla directa
- 16.7. Resumen
- 16.8. Ejercicios
- 16.9. Problemas

## Capítulo 17. Tipos abstractos de datos (tad/objetos)

- 17.1. El papel (el rol) de la abstracción
  - 17.1.1. La abstracción como un proceso natural mental
  - 17.1.2. Historia de la abstracción del software
  - 17.1.3. Procedimientos
  - 17.1.4. Módulos
  - 17.1.5. Tipos abstractos de datos
  - 17.1.6. Objetos
- 17.2. Tipos de datos
- 17.3. Abstracción en lenguajes de programación
  - 17.3.1. Abstracciones de control
  - 17.3.2. Abstracciones de datos
- 17.4. Tipos abstractos de datos
  - 17.4.1. Ventajas de los tipos abstractos de datos
  - 17.4.2. Implementación de los TAD
- 17.5. Especificación de TAD
  - 17.5.1. Especificación informal de un TAD
  - 17.5.2. Especificación formal de un TAD
- 17.6. Tipos abstractos de datos en C
  - 17.6.1. Implementación del TAD Conjunto
  - 17.6.2. Aplicación del tipo abstracto de dato Conjunto
- 17.7. Orientación a objetos
  - 17.7.1. Abstracción
  - 17.7.2. Encapsulación
  - 17.7.3. Modularidad
  - 17.7.4. Jerarquía
  - 17.7.5. Polimorfismo
  - 17.7.6. Otras propiedades
  - 17.7.7. Lenguajes de programación orientados a objetos
- 17.8. Reutilización de software
- 17.9. Desarrollo tradicional frente a desarrollo orientado a objetos
- 17.10. Resumen
- 17.11. Ejercicios
- 17.12. Problemas

## Capítulo 18. Listas enlazadas

- 18.1. Fundamentos teóricos
- 18.2. Clasificación de las listas enlazadas
- 18.3. Operaciones en listas enlazadas
  - 18.3.1. Declaración de un nodo
  - 18.3.2. Puntero de cabecera y cola
  - 18.3.3. El puntero nulo
  - 18.3.4. El operador -> de selección de un miembro
  - 18.3.5. Construcción de una lista
  - 18.3.6. Insertar un elemento en una lista
  - 18.3.7. Búsqueda de un elemento
  - 18.3.8. Eliminación de un nodo en una lista
- 18.4. Listas doblemente enlazadas
  - 18.4.1. Declaración de una lista doblemente enlazada
  - 18.4.2. Insertar un elemento en una lista doblemente enlazada

- 18.4.3. Eliminación de un elemento en una lista doblemente enlazada
- 18.5. Listas circulares
  - 18.5.1. Insertar un elemento en una lista circular
  - 18.5.2. Eliminación de un elemento en una lista circular
- 18.6. Resumen
- 18.7. Ejercicios
- 18.8. Problemas

## Capítulo 19. Pilas y colas

- 19.1. Concepto de pila
  - 19.1.1. Especificaciones de una pila
- 19.2. El tipo pila implementado con arrays
  - 19.2.1. Especificación del tipo pila
  - 19.2.2. Implementación de las operaciones sobre pilas
  - 19.2.3. Operaciones de verificación del estado de la pila
- 19.3. Concepto de cola
- 19.4. Colas implementadas con arrays
  - 19.4.1. Definición de la especificación de una cola
  - 19.4.2. Especificación del tipo cola
  - 19.4.3. Implementación del tipo cola
  - 19.4.4. Operaciones de la cola
- 19.5. Realización de una cola con una lista enlazada
  - 19.5.1. Declaración del tipo cola con listas
  - 19.5.2. Codificación de las operaciones del tipo cola con listas
- 19.6. Resumen
- 19.7. Ejercicios
- 19.8. Problemas

## Capítulo 20. Árboles

- 20.1. Árboles generales
  - 20.1.1. Terminología básica
- 20.2. Árboles binarios
  - 20.2.1. Equilibrio
  - 20.2.2. Árboles binarios completos
- 20.3. Estructura de un árbol binario
  - 20.3.1. Diferentes tipos de representaciones en C
  - 20.3.2. Creación de un árbol binario
- 20.4. Operaciones en árboles binarios
  - 20.4.1. Profundidad y altura de un árbol binario
  - 20.4.2. Número de hojas de un árbol binario
  - 20.4.3. Eliminar los nodos de un árbol binario
- 20.5. Árbol de expresiones
  - 20.5.1. Reglas para la construcción de árboles de expresión
- 20.6. Recorrido de un árbol
  - 20.6.1. Recorrido preorden
  - 20.6.2. Recorrido enorden
  - 20.6.3. Recorrido postorden
  - 20.6.4. Evaluación de un árbol de expresión
- 20.7. Árbol binario de búsqueda
  - 20.7.1. Creación de un árbol binario de búsqueda
- 20.8. Operaciones con árbol binario de búsqueda
  - 20.8.1. Búsqueda
  - 20.8.2. Insertar una clave en un árbol ordenado
  - 20.8.3. Eliminación de una clave en un árbol ordenado
- 20.9. Resumen
- 20.10. Ejercicios

20.11. Problemas

Apéndice A. Compilación de programas C en Windows

Apéndice B. Bibliografía y recursos de programación

Índice