

Contenido

Prólogo	XIII
CAPÍTULO 1	
Introducción al diseño de algoritmos y programación	1
1.1. Lección 1	1
1.1.1. ¿Qué es un algoritmo?	1
1.1.2. Lenguaje algorítmico	4
1.1.3. Recursos de programación	4
1.1.3.1. La computadora	4
1.1.3.2. Operaciones aritméticas y expresiones lógicas	4
1.1.3.3. Lenguaje de programación	5
1.1.3.4. Programa	5
1.1.3.5. Variables y tipos de dato	6
1.1.3.6. Convención de nombres de variables	7
1.1.3.7. Consola	8
1.1.3.8. Compilador	8
1.1.3.9. Entorno Integrado de desarrollo ..	9
1.1.4. Teorema de la programación estructurada	9
1.1.4.1. Acción simple	10
1.1.4.2. Acción condicional	11
1.1.4.3. Acción iterativa	15
1.1.4.4. Acciones de única entrada y única salida	19
1.1.5. Más recursos de programación	20
1.1.5.1. Contadores y acumuladores	20
1.1.5.2. Prueba de escritorio	20
1.1.5.3. Operadores aritméticos	21
1.1.5.4. Otras operaciones matemáticas ..	23
1.1.5.5. Operadores y expresiones lógicas ..	24
1.1.5.6. Operadores relacionales.....	25
1.1.5.7. Operadores relacionales y cadenas de caracteres	26
1.1.6. Análisis de ejercicios y problemas... ..	27
1.1.6.1. Datos de entrada, de contexto y de salida	27
1.1.6.2. Procesos para transformar la entrada en salida.....	28
1.1.7. Tipos de problema	28
1.1.7.1. Problemas de registro simple y problemas de múltiples registros	29
1.1.7.2. Multiplicidad	30
1.1.7.3. Registros y tablas	30
1.1.7.4. Problemas de procesamiento horizontal y procesamiento vertical.....	32
1.1.7.5. Problemas de corte de control.....	33
1.1.8. Autoevaluación y ejercicios	35
1.2. Lección 2	35
1.2.1. Tipo de dato	36
1.2.1.1. Tipos enteros	37
1.2.1.2. Tipos flotantes	38
1.2.1.3. Tipos alfanuméricos	39
1.2.1.4. Tipos lógicos	40
1.2.1.5. Tipo de dato nulo.....	40
1.2.1.6. Tipos de dato primitivos y tipos definidos por el programador.....	40
1.2.2. Alcance de una variable	41
1.2.3. Metodología Top-Down	43
1.2.3.1. Funciones	43
1.2.3.2. Prototipo de una función.....	46
1.2.3.3. Compilar un programa que invoca funciones	47
1.2.3.4. Reusabilidad del código	48
1.2.3.5. Legibilidad del código fuente	51
1.2.3.6. Bibliotecas de funciones.....	51
1.2.3.7. Convención de nombres de funciones	51
1.2.3.8. Parámetros y argumentos.....	52
1.2.3.9. Parámetros por valor y referencia ..	52
1.2.3.10. Variables locales	54
1.2.4. Autoevaluación y ejercicios	54
1.3. Lección 3	55
1.3.1. Estructuras.....	55
1.3.1.1. Inicialización de una estructura... ..	56

1.3.1.2. Convención de nombres de estructuras 57

1.3.1.3. Función de inicialización de una estructura 57

1.3.1.4. Estructuras anidadas 58

1.3.2. Tipo Abstracto de Dato (TAD) 59

1.3.2.1. Usuario del TAD 62

1.3.2.2. Inicialización de un TAD 62

1.3.2.3. Sobrecarga de funciones 62

1.3.3. Autoevaluación y ejercicios 63

1.4. ¿Qué sigue? 63

CAPÍTULO 2

Cadenas de caracteres y estructura de datos 65

2.1. Lección 4 65

2.1.1. Carácter 65

2.1.2. Cadena de caracteres 67

2.1.2.1. Caracteres especiales y carácter de escape 67

2.1.2.2. Carácter nulo que indica el final de una cadena 68

2.1.2.3. Acceso directo a los caracteres de una cadena 68

2.1.2.4. Longitud de una cadena 69

2.1.2.5. Operadores aritméticos unarios... 69

2.1.2.6. Ciclo iterativo for 70

2.1.2.7. Concatenar cadenas de caracteres 71

2.1.2.8. Operadores relacionales aplicados a cadenas 71

2.1.2.9. Función de comparación 72

2.1.2.10. If-inline 74

2.1.3. Biblioteca de funciones y API 76

2.1.3.1. Tratamiento de cadenas de caracteres 77

2.1.3.2. Actividad práctica: API de tratamiento de cadenas de caracteres 77

2.1.4. Argumentos en línea de comandos. 79

2.2. Lección 5 80

2.2.1. Tratamiento de tokens 81

2.2.2. Actividad práctica: API de tratamiento de tokens 81

2.2.3 Autoevaluación y ejercicios 84

2.3. Lección 6 84

2.3.1. Funciones como argumentos de otras funciones 85

2.3.2. Tipo de dato genérico (template) 87

2.3.3. Autoevaluación y ejercicios 93

2.4. Lección 7 93

2.4.1. Colecciones 94

2.4.2. TAD Coll 95

2.4.2.1. Ejemplo de uso 96

2.4.2.2. Estructura del TAD Coll 98

2.4.2.3. Actividad práctica: API del TAD Coll 98

2.4.3. Autoevaluación y ejercicios 101

2.5. Lección 8 101

2.5.1. Ordenamiento 101

2.5.1.1. Ordenamiento por inserción simple 101

2.5.1.2. Ordenamiento por burbujeo 102

2.5.1.3. Ordenamiento por burbujeo mejorado 103

2.5.1.4. Ordenamiento por inserción avanzado 104

2.5.2. Búsqueda 105

2.5.2.1. Búsqueda lineal 105

2.5.2.2. Búsqueda binaria 106

2.5.3. Autoevaluación y ejercicios 108

2.6. Lección 9 108

2.6.1. Estructura de datos (parte 1) 109

2.6.1.1. Estructuras estáticas y dinámicas 109

2.6.1.2. Estructura de datos como cimiento del algoritmo 110

2.6.1.3. Colección de estructuras 110

2.6.1.4. Colección de colecciones 116

2.6.1.5. Colecciones de estructuras que tienen colecciones 119

2.6.2. Autoevaluación y ejercicios 122

2.7. ¿Qué sigue? 122

CAPÍTULO 3

Archivos 123

3.1. Lección 10 123

3.1.1. Introducción 123

3.1.1.1. Archivo 123

3.1.1.2. Tipos de archivo 124

3.1.1.3. Archivos binarios 124

3.1.1.4. Archivos de texto 124

3.1.2. Gestión de archivos 125

3.1.2.1. Funciones de biblioteca 125

3.1.2.2. Grabar y leer datos 125

3.1.2.3. Archivo secuencial 127

3.1.2.4. Archivos de registros de longitud fija 127

3.1.2.5. Big-endian y Little-endian 128

3.1.2.6. Archivos de registros de longitud variable.....	129
3.1.2.7. Archivos de estructuras.....	129
3.1.2.8. Posicionamiento directo.....	132
3.1.2.9. Posicionamiento directo en registros.....	133
3.1.2.10. Eliminar registros físicamente ...	134
3.1.2.11. Eliminar registros lógicamente ..	134
3.1.2.12. Modificar registros.....	136
3.1.2.13. Longitud de un archivo.....	137
3.1.2.14. Cantidad de registros.....	137
3.1.2.15. Restricciones.....	137
3.1.3. Actividad práctica: API para el tratamiento de archivos de registros	138
3.1.4. Ejemplos de uso de las funciones de la API.....	139
3.1.4.1. Escribir y leer caracteres.....	139
3.1.4.2. Escribir y grabar números enteros (short).....	139
3.1.4.3. Escribir y leer estructuras (Persona).....	139
3.1.4.4. Baja lógica.....	140
3.1.5. Autoevaluación y ejercicios.....	141
3.2. Lección 11.....	141
3.2.1. Operadores de bit.....	141
3.2.1.1. Operadores de desplazamiento ...	141
3.2.1.2. Bases numéricas 8 (octal) y 16 (hexadecimal).....	142
3.2.1.3. Operadores lógicos.....	143
3.2.1.4. Máscara de bit.....	144
3.2.2. Autoevaluación y ejercicios.....	145
3.3. ¿Qué sigue?.....	145

CAPÍTULO 4

Resolución de problemas	147
4.1. Cómo analizar un problema.....	147
4.1.1. Contexto, relevamiento y enunciado del problema.....	148
4.1.2. Archivos de novedades y consultas	148
4.1.3. Problemas de corte de control	149
4.1.4. Problemas de apareo de archivos...	150
4.1.5. Restricciones.....	152
4.1.6. Estrategia	152
4.2. Búsqueda sobre archivos de consulta.....	153
4.2.1. Subir el archivo a memoria, en una colección de objetos.....	153
4.2.1.1. Subir archivo.....	153
4.2.1.2. Buscar registro.....	154

4.2.1.3. Ejemplo de uso.....	155
4.2.1.4. Funciones de comparación, tToString y tFromString.....	155
4.2.2. Búsqueda binaria sobre un archivo	156
4.2.2.1. Buscar registro.....	156
4.2.2.2. Ejemplo de uso.....	157
4.2.2.3. Funciones de comparación, tToString y tFromString.....	157
4.2.3. Indexar un archivo.....	159
4.2.3.1. Estructura del índice.....	159
4.2.3.2. Indexar.....	159
4.2.3.3. Buscar un registro.....	160
4.2.3.4. Ejemplo de uso.....	161
4.2.3.5. Funciones de comparación, tToString y tFromString.....	161
4.3. Ordenar archivos de consulta.....	162
4.3.1. Ordenamiento en memoria.....	162
4.3.2. Ordenamiento por indexación.....	163
4.4 Resolución de problemas.....	164
4.5 ¿Qué sigue?.....	165

CAPÍTULO 5

Estructuras indexadas, lineales y gestión de memoria	167
5.1. Lección 12.....	167
5.1.1. Array.....	167
5.1.1.1. Capacidad del array.....	168
5.1.1.2. Inicializar un array a partir un conjunto de valores.....	169
5.1.1.3. Inicialización programática.....	169
5.1.1.4. Longitud del array.....	170
5.1.1.5. Arrays de estructuras.....	170
5.1.1.6. Arrays multidimensionales.....	172
5.1.2. Operaciones sobre arrays.....	173
5.1.2.1. Agregar un elemento al final de un array.....	173
5.1.2.2. Determinar si el array contiene un elemento especificado.....	174
5.1.2.3. Insertar un elemento en una determinada posición.....	175
5.1.2.4. Eliminar el elemento ubicado en una determinada posición.....	176
5.1.3. Actividad práctica: API de operaciones sobre arrays.....	178
5.1.4. Autoevaluación y ejercicios.....	179
5.2. Lección 13.....	179
5.2.1. Gestión de memoria.....	179
5.2.1.1. Punteros.....	180

5.2.1.2. Operador de dirección (&)181

5.2.1.3. Operador de dirección o contenido (*)181

5.2.1.4. Funciones que reciben punteros como parámetros.....182

5.2.1.5. Punteros a punteros183

5.2.1.6. Punteros a estructuras y operador -> (flecha).....183

5.2.1.7. Punteros y arrays. Punteros a estructuras y operador -> (flecha).....184

5.2.1.8. Aritmética de direcciones185

5.2.1.9. Memoria estática186

5.2.1.10. Memoria dinámica.....186

5.2.1.11. Crear arrays dinámicamente.....189

5.2.1.12. Redimensionar un array.....190

5.2.1.13. Crear matrices dinámicamente .191

5.2.2. Actividad práctica: TAD Array192

5.2.3. Actividad práctica: TAD Map193

5.2.4. Autoevaluación y ejercicios.....194

5.3. Lección 14195

5.3.1. Nodo195

5.3.2. Lista enlazada (linked list).....196

5.3.2.1. Recorrer una lista enlazada196

5.3.2.2. Agregar un valor al final de una lista enlazada198

5.3.2.3. Liberar la memoria que ocupa la lista200

5.3.2.4. Determinar si la lista contiene un valor especificado201

5.3.2.5. Insertar un valor en una lista ordenada201

5.3.2.6. Eliminar un elemento de la lista...204

5.3.3. Actividad práctica: API de operaciones sobre listas enlazadas205

5.3.4. Pila (stack).....207

5.3.4.1. Apilar un elemento (push)207

5.3.4.2. Desapilar un elemento (pop).....208

5.3.4.3. Determinar si la pila tiene elementos209

5.3.4.4. Ejemplo de uso209

5.3.5. Actividad práctica: API de operaciones sobre pilas (extensión).....209

5.3.6. Cola (queue)209

5.3.6.1. Encolar un elemento (enqueue)...210

5.3.6.2. Desencolar un elemento (dequeue)211

5.3.6.3. Determinar si la cola tiene elementos212

5.3.6.4. Implementación sobre una lista circular213

5.3.7. Actividad práctica: API de operaciones sobre colas (extensión)215

5.3.8. Actividad práctica: TAD List, Stack y Queue.....216

5.3.9. Autoevaluación y ejercicios217

5.4. Lección 15.....217

5.4.1. Estructura de datos (parte 2).....218

5.4.1.1. Colección de estructuras.....218

5.4.1.2. Colección de colecciones219

5.4.1.3. Colección de estructuras que tienen colecciones220

5.4.2. Autoevaluación y ejercicios222

5.5. ¿Qué sigue?.....222

CAPÍTULO 6

Algoritmo de Huffman. Ejercicio integrador

.....223

6.1. Lección 16.....223

6.1.1. Introducción223

6.1.2. Alcance del ejercicio225

6.1.3. Algoritmo de Huffman.....225

6.1.3.1. Paso 1 – Contar cuántas veces aparece cada byte227

6.1.3.2. Paso 2 – Crear una lista enlazada228

6.1.3.3. Paso 3 – Convertir la lista en un árbol binario (Árbol Huffman)228

6.1.4. Árbol Huffman231

6.1.4.1. Características.....231

6.1.4.2. Códigos Huffman231

6.1.4.3. Longitud máxima de un código Huffman232

6.1.5. Implementación del ejercicio232

6.1.5.1. Setup.....233

6.1.5.2. TAD HuffmanTree233

6.1.5.3. Estructura HuffmanTreeInfo234

6.1.5.4. Recopilación de los códigos Huffman234

6.1.5.5. Compresión.....235

6.1.5.6. Estructura del archivo comprimido (.huf).....236

6.1.5.7. Descompresión.....236

6.1.6. Ejemplo237

6.2. ¿Qué sigue?.....227