

## INDICE

<b>Prologo</b>	xxi
<b>Parte I. Instalación y utilización de Turbo C++</b>	
<b>1. El entorno integrado de desarrollo Turbo C++</b>	<b>3</b>
1.1 Instalación de turbo C++	
1.2 Arranque de Turbo C++	4
1.3 El entorno integrado de desarrollo (EID)	5
1.3.1. Los menús	6
1.3.2. Las ventanas	9
1.3.3. Operaciones en ventanas	11
1.3.4. La línea de estado	
1.3.5 Cuadros de dialogo	12
1.4. Funcionamiento del copilador Turbo C++	
1.4.1. Edición, grabación y compilación de un programa	15
1.4.2 Elección de un modelo de memoria	16
1.5. Secuencias de teclas	17
1.6. Uso del editor	18
1.6.1. Seleccionar texto y operaciones co0n bloques de texto	
1.6.2. Uso del portapapeles	19
1.6.3. Abrir un archivo existente	20
1.7. El sistema de ayuda EID	
1.7.1. Help contents	21
1.7.2. Help about Help	
1.7.3. Help Index	22
1.7.4. Ayuda sensible al contexto	
1.7.5. About	
1.7.6. Copiar texto el sistema de ayuda	23
1.8. Manipulación de ventanas	
¿Cómo hacer una ventana activa?	
¿Cómo cerrar una ventana?	
¿Cómo desplazarse por la ventana activa?	
¿Cómo ocupar toda la pantalla de una ventana?	24
¿Cómo desplazarse de un cuadro de dialogo?	
1.9. Consejos prácticos de edición	
¿Como crear un nuevo archivo de texto?	
¿Cómo abrir un archivo de texto existente?	
¿Cómo almacenar (guardar) un archivo de texto?	
¿Cómo desplazarse en el archivo de texto?	
¿Cómo seleccionar y una parte de texto?	25
¿Cómo suprimir?	
¿Cómo insertar una línea?	
¿Cómo anular una operación?	
¿Como grabar texto seleccionado?	
¿Cómo insertar un archivo en la posición del cursor?	
¿Cómo imprimir texto seleccionado?	
¿Cómo utilizar el portapapeles?	26
1.10. Salir de turbo C++	
1.11. Depurar programas	27

1.11.1. Paso a paso (Trace into)	28
1.11.2. Utilizar una ventana de observación "Watch"	
1.11.3. Saltar subrutinas depuradas (Step over)	
1.11.4. Volver a arrancar el programa (Program reset)	
1.11.5. Establecer puntos de ruptura	29
1.11.6. Como evaluar y modificar el contenido de una variable	
1.11.7. Volver a cargar un programa	30
1.12. Resumen	
<b>Parte II. El lenguaje C++</b>	
<b>2. El lenguaje C++: elementos básicos</b>	33
2.1. Estructura general de un programa en C++	35
2.1.1. Directivas del preprocesador	
2.1.2. Declaraciones globales	37
2.1.3. Función main ()	
2.1.4. Funciones definidas por el usuario	38
2.1.5. Comentarios	40
2.2. Creación de un programa	43
2.2.1. Edición del código fuente	44
2.3. Los elementos de un programa en C++	45
2.3.1. Tokens (Elementos léxicos de los programas)	
2.3.2. Identificadores	46
2.3.3. Palabras reservadas	
2.3.4. Comentarios	47
2.3.5. Signos de puntuación	
2.3.6. Separadores	49
2.3.7. Archivos de cabecera	
2.4. Tipos de datos en C++	50
2.4.1. Enteros	
2.4.2. Tipos de coma flotante	49
2.4.3. Caracteres	50
2.4.4 Tipos constante de coma flotante	
2.5. Constantes	51
2.5.1. Constantes literales	
2.5.2 Constantes definidas (simbólicas)	55
2.5.3 Constantes enumeradas	
2.5.4. Constantes declaradas: const y volatile	46
2.6. Variables	57
2.6.1. Declaración	58
2.6.2. Inicialización de variables	59
2.7. Duración de una variable	60
2.7.1. Variables locales	
2.7.2. Variables globales	61
2.7.3 Variables dinámicas y de objetos	
2.8. Entradas y salidas	62
2.8.1 Salida (cout)	63
2.8.2. Entrada (cin)	65
2.9. Resumen	66
<b>3. Operadores y expresiones</b>	67
3.1. Operadores y expresiones	

3.2. Operador de asignación	68
3.3. Operadores aritméticos	
3.3.1. Asociatividad	70
3.4. Operadores de incrementación	71
3.5. Operadores relacionales	73
3.6. Operadores lógicos	74
3.6.1. Evaluación en cortocircuito	76
3.7. Operadores de manipulación de bits	77
3.7.1. Operadores de asignaciones adicionales	
3.7.2. Operadores de desplazamiento de bits (>>,<<)	79
3.7.3. Operadores de direcciones	
3.8. Operador condicional	80
3.9. Operador coma	81
3.10. Operadores especiales: (),{} y ::	
3.10.1. El operador ()	82
3.10.2. El operador {}	
3.10.3. El operador ::	
3.11. El operador sizeof	83
3.12. Conversiones de tipos	
3.12.1. Conversión implícita	84
3.12.2. Reglas	
3.12.3. Conversiones explícitas	
3.13. Prioridad y asociatividad	85
3.13. Resumen	86
<b>4. Tomar decisiones</b>	
4.1. Sentencias y bloques	89
4.2. Sentencias condicionales	
4.2.1 La sentencia if	90
4.2.2 Sentencias verdaderas y falsas	
4.2.3. Expresiones condicionales	92
4.2.4. La sentencia if-else	93
4.2.5. Problemas con la sentencia if	
4.2.6. La sentencias múltiples alternativas if-else	94
4.2.7. Operador condicional	96
4.3. La sentencia de múltiples alternativas: switch	97
4.3.1. Casos múltiples	
4.4. Resumen	99
<b>5. Bucles</b>	101
5.1. La sentencia for	102
5.1.1 Definición de la variable de control dentro del bucle for	104
5.1.2. Uso del tipo de dato char como variable de control de bucle	
5.1.3. Uso de expresiones complejas para la expresión de incremento	
5.14. Omisión de secciones en un bucle for	105
5.2. La sentencia break	
5.3. La sentencia continue	
5.3.1. Diferencias entre continue y break	107
5.4. La sentencia goto	109
5.5. La sentencia while	110
5.6. La sentencia do-while	112

5.7. Elementos múltiples en bucles for	
5.8. Resumen	114
<b>6. Funciones</b>	
6.1. Concepto de función	117
6.2. Estructura de una función	119
6.2.1. Nombre de una función	119
6.2.2. Tipo de dato de retorno	120
6.2.3. Resultados de una función	121
6.2.3. Resultados de una función	122
6.2.4. Llamada a una función	123
6.3. Prototipos de la s funciones	125
6.3.1. Prototipos con un numero no especificado de parámetros	
6.3.2. Ejemplos de prototipos	126
6.4. Parámetros de una función	
6.4.1. Paso de parámetros por valor	127
6.4.2. Paso de parámetros por referencia	128
6.4.3. Diferencias entre los parámetros por valor y por referencia	130
6.4.4 Parámetros const. de una función	131
6.5. Argumentos por omisión	132
6.6. Funciones en línea (inline)	135
6.6.1. Creación de funciones en línea	136
6.7. Ámbito (alcance)	137
6.7.1 Ámbito del programa	138
6.7.2 Ámbito del archivo fuente	
6.7.3 Ámbito de una función	139
6.7.4 Ámbito del bloque	
6.7.5. Variables locales	
6.8. Clases de almacenamiento	140
6.8.1 Variables automáticas	
6.8.2. Variables externas	
6.8.3. Variables registro	141
6.8.4. Variables estáticas	
6.9. Resumen	142
<b>7. Arrays (listas y tablas)</b>	
7.1. Arrays	145
7.1.2. Subíndices de un Array	
7.1.3. Ejemplos de Arrays	147
7.1.4. Almacenamiento en memoria de los Arrays	
7.1.5. El tamaño de los Arrays	
7.2. Inicialización de un array	149
7.3. Arrays de caracteres y cadenas de texto	152
7.4. Arrays multidimensionales	154
7.4.1. Inicialización de Arrays multidimensionales	156
7.4.2. Acceso a los elementos de los Arrays bidimensionales	
7.4.3. Lectura y escritura de elementos de Arrays bidimensionales	158
7.4.4 Acceso a elementos mediante bucles	
7.4.5. Array de mas de dos dimensiones	159
7.4.6. Una aplicación practica	
7.5. Utilización de Arrays como parámetros	160

7.5.1. Precauciones	162
7.5.2 Paso de cadenas como parámetros	164
7.6. Ordenación de listas	
7.6.1. Algoritmo de la burbuja	165
7.7. Búsqueda en listas	167
7.7.1. Búsqueda secuencial	168
7.8. Resumen	170
<b>8. Estructuras y uniones</b>	
8.1. Estructuras	171
8.1.1. Declaración de una estructura	
8.1.2. Definición de variables de estructuras	172
8.1.3. Uso de estructuras en asignaciones	
8.1.4. inicialización de un a declaración de estructuras	174
8.2. Acceso a estructuras	
8.2.1. Almacenamiento de información en estructuras	175
8.2.2. Lectura de información de una estructura	
8.2.3. Recuperación de información de una estructura	177
8.3. Estructuras anidadas	178
8.3.1. Un ejemplo de estructuras	
8.4. Arrays de estructuras	180
8.5. Utilización de estructuras como parámetros	182
8.6. Uniones	183
8.6.1. Definición de tipos unión	184
8.6.2. Una aplicación de unión	185
8.6.3. Otra aplicación	186
8.6.4. Uniones anónimas	187
8.7. Resumen	188
<b>9. Biblioteca de funciones</b>	189
9.1. Concepto y uso de funciones de biblioteca	190
9.2. Funciones de carácter	
9.2.1. Comprobación alfabética y de dígitos	191
9.2.2. Funciones de prueba de caracteres especiales	
9.2.3. Funciones de conversión de caracteres	193
9.3. Funciones numéricas	
9.3.1 Funciones matemáticas	194
9.3.2 Funciones trigonométrica	195
9.3.3 Funciones Logarítmicas y exponenciales	
9.3.4 Funciones aleatorias	196
9.4. Funciones de fecha y hora	198
9.5. Funciones de utilidad	200
9.6. Resumen	201
<b>10. Punteros (apuntadores)</b>	
10.1. Concepto de puntero (apuntador)	203
10.1.1. Declaración de punteros	
10.1.2. Inicialización (iniciación) de punteros	205
10.1.3. Indirección de punteros	206
10.1.4. Punteros y verificación de tipos	208
10.2. Punteros null y void	209
10.3. Punteros a punteros	210

10.4. Arrays de punteros	211
10.4.1 Inicialización de un array de punteros a cadenas	
10.5. Punteros y arrays	212
10.5.1. Nombres de arrays como punteros	
10.5.2 Ventajas de los punteros	213
10.6. Punteros de cadenas	214
10.6.1 Punteros frente a arrays	215
10.7. Aritmética de punteros	216
10.7.1 Una aplicación de punteros	218
10.8 Punteros constantes frente a punteros a constates	
10.8.1. Punteros constante	219
10.8.2. Punteros constantes	220
10.8.3. punteros constantes a constantes	221
10.9. Punteros como argumentos de funciones	
10.9.1. Paso por referencia frente a paso por dirección	223
10.10. Punteros a funciones	
10.10.1. Inicialización de un puntero a una función	224
10.10.2. Otra aplicación	227
10.10.3. arrays de punteros de funciones	228
10.10 4. Una aplicación práctica	229
10.11. punteros a estructuras	230
10.12. Resumen	232
<b>11. Cadenas</b>	
11.1. Concepto de cadena	233
11.1.1. Declaración de variables de cadena	234
11.1.2. Inicialización de variables de cadena	
11.2. Lectura de cadenas	235
11.3. La biblioteca STRING.H	237
11.3.1. La palabra reservada conts	238
11.4. Arrays y cadenas como parámetros de funciones	239
11.4.1 Uso del operador de referencia para tipos array	
11.4.2. uso de punteros para pasar una cadena	240
11.5. asignación de cadenas	241
11.5.1. La función strncpy	242
11.6. Longitud y concatenación de cadenas	
11.6.1. La función strlen	243
11.6.2. Las funciones stract y strncat	
11.7. Comparación de cadenas	
11.7.1. La función strcmp	244
11.7.2. La función strcmp	
11.7.3. La función strncmp	245
11.7.4. La función strnicmp	
11.8. Inversión de cadenas	246
11.9 Conversión de cadenas	
11.9.1. Funciónstrupr	247
11.9.2. Función strlwr	
11.10. Conversión de cadenas a números	248
11.10.1. Función atoi	
11.10.2. Función atof	249

11.10.3. Función atol	
11.11. Búsqueda de caracteres y cadenas	250
11.11.1. La función strchr	
11.11.2. La función strchr	
11.11.3. La función strspn	251
11.11.4. La función strcspn	
11.11.5. La función strpbrk	252
11.12. Búsqueda de cadenas	
11.12.1. La función strstr	
11.12.2. La función strtok	253
11.13. Resumen	254
<b>12. Tratamiento avanzado de funciones</b>	
12.1. Visibilidad de una función	255
12.1.1. Variables locales frente a variables globales	256
12.1.2. Variables estáticas y automáticas	259
12.2. Compilación separada	261
12.3. Variables registro (register)	263
12.4. Sobrecarga de funciones (polimorfismo)	264
12.4.1. ¿Cómo determina C++ la función sobrecarga correcta?	265
12.5. Recursividad	268
12.5.1. Un algoritmo recursivo	269
12.5.2. Funciones mutuamente recursivas	
12.5.3. Condición de determinación de la recursión	270
12.5.4. el algoritmo recursivo máximo común divisor	
12.6. Plantillas de funciones	271
12.6.1. Utilización de las plantillas de funciones	273
12.6.2. plantillas de función min y max	274
12.7. Resumen	275
<b>13. Asignación dinámica de memoria</b>	
13.1. Gestión dinámica de la memoria	277
13.1.2. Ventajas de la asignación dinámica de memoria en C++	
13.2. El operador new	280
13.2.1. Asignación de memoria de un tamaño desconocido	
13.2.2. Inicialización de memoria con un valor	285
13.2.3. Uso de new para arrays multidimensionales	
13.3. El operador delete	286
13.4. Ejemplos que utilizan new y delete	287
13.5. Asignación de memoria para arrays	288
13.5.1. Asignación de memoria interactivamente	
13.5.2. Asignación de memoria para un array de estructuras	289
13.6. Gestión del desbordamiento de memoria: set-new-handler	291
13.7. Reglas de funciones de new y delete	292
13.8. Resumen	294
<b>14. Entradas y salidas: flujos (streams)</b>	
14.1. Flujos	295
14.2. La biblioteca de clases iostream	
14.2.1. La clase streambuf	296
14.2.2. Jerarquía de clases ios	
14.2.3. Flujos estándar	297

14.2.4. Entradas/salidas en archivos	
14.2.5. Entradas/salidas en un buffer de memoria	298
14.2.6. Archivos de cabecera	
14.2.7. Entrada/salida de caracteres y flujos	299
14.3. Salida a la pantalla y a al impresora	300
14.3.1. operadores de inserción en casacad	301
14.3.2 Las funciones miembro put() y write()	302
14.3.3. Impresión de la salida en una impresora	303
14.4. Lectura del teclado	304
14.4.1. Lectura de datos carácter	
14.4.2. Lectura de datos cadena	306
14.4.3. Funciones miembro get() y getline()	308
14.4.4. Problemas en la utilización de getline()	310
14.5. Formateando de la salida	312
14.6. Manipuladores	313
14.6.1. Bases de numeración	314
14.6.2. Anchura de los campos	316
14.6.3. Rellenado de caracteres	317
14.6.4. Precisión de números reales	318
14.7. Indicadores de formato	319
14.7.1. Uso de setiosflags() y resetiosflags()	319
14.7.2. Las funciones miembro setf() y unsetf()	
14.8. Resumen	322
<b>15. Archivos</b>	
15.1. Concepto de archivo	325
15.1.1. Tipos de acceso a los archivos de disco	326
15.2. Archivos secuenciales	
15.2.1. Apertura y cierre de archivos	327
15.2.2 Escritura de un archivo	329
15.2.3. escritura en una impresora	330
15.2.4. añadir datos a un archivo	331
15.2.5. Lectura de un archivo	332
15.3. Archivo binarios	334
15.3.1. Lectura y escritura de archivos de datos binarios	335
15.4. Archivo de datos de acceso aleatorio	336
15.4.1. Apertura de archivos de acceso de aleatorio	337
15.4.2. La función seekg()	338
15.4.3. La función tellg()	340
15.4.4. La función seekp()	
15.4.5. La función tellp()	
15.4.6. Programa ejemplo aleaoto.cpp	341
15.4.7. Un archivo binario con acceso aleatorio	
15.5. Resumen	344
<b>Parte III. Programación orientada a objetos</b>	
<b>16. Conceptos de programación orientada a objetos</b>	347
16.1. La evolución de la programación	
16.1.1. Programación lineal	
16.1.2. Programación modular	348
16.1.3. Programación estructurada	349

16.1.4. Abstracción de datos	
16.2. ¿Que es la programación orientada a objetos?	350
16.2.1. Trabajar con objetos	
16.2.2 Definición de objetos	353
16.2.3. Clases	354
16.2.4. Mensajes: Activación de objetos	356
16.2.5. Programación orientada objetos	357
16.2.6. Herencia	358
16.2.7. Polimorfismo	
16.2.8. Reutilización	361
16.3. Los lenguajes de programación orientados a objetos	
16.3.1. Smalltalk	362
16.3.2. Pascal orientado objetos	363
16.4. Resumen	364
<b>17. Clases</b>	
17.1. Clases y objetos	365
17.1.1. Declaración de una clase	366
17.1.2. Visibilidad de una clase	367
17.1.3. Reglas de construcción de clases	369
17.1.4. Funciones en línea	371
17.2. Estructuras y uniones frente a clases	
17.3. Constructores	374
17.3.1. Reglas de los constructores	
17.3.2. Clases con múltiples constructores	375
17.3.3 Constructor de copia	377
17.3.4. Constructor por defecto	378
17.3.5. Uso del constructor	
17.4. Destruidores	379
17.4.1. Reglas del destructor	
17.4.2. Funcionamiento del destructor	380
17.5. Funciones amigas	381
17.6. Miembros estáticos	384
17.7.1. Miembros dato	
17.8. Resumen	385
<b>18. Herencia y polimorfismo</b>	387
18.1. Herencia	388
18.1.1. Anulación de funciones (overriding)	391
18.2. Herencia múltiple	392
18.3. Funciones virtuales	393
18.4. Clases abstractas	395
18.5. Clases base virtuales	397
18.5.1. Reglas de llamada a constructores/destructores	
18.6. Sobrecarga de operadores	399
18.6.1. Sintaxis de la sobrecarga de operadores	400
18.6.2. Operadores que se pueden sobrecargar	401
18.6.3. Operadores que no se pueden sobrecargar	
18.6.4. Reglas de sobrecarga de operaciones	402
18.6.5. ¿Cómo elegir una transfusión miembro y una función amiga?	
18.7. Polimorfismo	405

18.8. Ligadura dinámica y estática	407
18.8.1. Ligadura dinámica frente a ligadura estática	408
18.9. Anulación, restricciones y revisión de funciones virtuales	409
18.9.1. Anulación de funciones	410
18.9.2. Restricciones de las funciones virtuales	
18.10. Criterios de diseño de clases abstractas	412
18.10.1. Implementación de clases abstractas	413
18.11. Resumen	414
<b>Parte IV. Construcción de programas en C y C++</b>	
<b>19. Diseño práctico de programas en C++/C</b>	419
19.1. Compilación separada de programas	
19.1.1. Programas multiarchivo	
19.1.2. Bibliotecas de clases	421
19.2. Almacenamiento extern y static	
Extern	
Static	423
19.3. Estructuras de un programa C	425
19.4. Compilación separada de clases	426
19.5. Estructura de un programa C++	432
19.5.1. Recordatorio de archivos de cabecera	436
19.6. programas multiarchivo	437
19.6.1. ¿Que se debe poner en un archivo fuente?	
19.6.2. Referencias externas	438
19.7. Construcción de archivos proyecto	440
19.7.1. El gestor de proyectos	
19.7.2. Abrir un proyecto	441
19.7.3. Añadir archivos fuente	443
19.7.4. Construcción del proyecto	444
19.7.5. Modificar la lista de un proyecto	
19.7.6. Trabajar con archivos proyecto	445
19.8. Transporte de aplicaciones desde C a C++	
19.8.1. Enlace entre programas C y C++	446
19.9. Resumen	448
<b>20. Programación eficiente. (consejos prácticos y errores típicos)</b>	
20.1. Depuración de programas	449
20.1.1. Errores durante la depuración	450
20.2. Errores en arrays	
20.2.1. Mal uso de los índices	451
20.2.2. Índices fuera de rango	
20.3. Errores en cadenas	452
20.3.1. Confundir comillas simples y dobles	
20.3.2. Cálculo de la longitud de la cadena	453
20.4. Errores en comentarios	454
20.5. Errores en corchetes y llaves	
20.6. Errores en funciones	455
20.6.1. Pasar un argumento por valor en lugar de por referencia	
20.6.2. Fallos en el valor de retorno de a función	456
20.6.3. No incluir el archivo de cabecera de una función en tiempo de ejecución	457

20.7. Errores en marcos	
20.7.1. Omisión de paréntesis en los argumentos de marcos	
20.7.2. Especificación no valida de macros tipo función	458
20.8. errores con operadores	
20.8.1. Mal uso de operadores de incremento (++) y decremento(--)	459
20.8.2. confusión de operadores de asignación	
20.8.3. Fallos en la precedencia de operadores	
20.9. Errores en punteros	460
20.9.1. Olvido del operador en dirección (&)	
20.9.2. Fallos al inicializar en puntero	461
20.9.3. Declaración de u puntero con el tipo incorrecto	
20.10. Errores en sentencias de selección (switch,if-else)	462
20.11. Errores en separadores	463
20.12. Errores básicos frecuentes	464
20.13. errores en clases	469
20.14. Resumen	475
<b>21. Turbo C++ 4.5. y 3.2. para Windows</b>	
21.1. Características diferenciales de turbo C++4.5	477
21.1.1 Nuevas características	
21.1.2. Características modificadas	
21.2. Instalación de Turbo C++ 4.5. para Windows	478
21.2.1. El entorno integrado de desarrollo (EID)	
21.3. El sistema de menús	480
21.4. El sistema de ventanas	482
21.4.1. Cuadros de dialogo	483
21.5. Puesta a punto de un programa	484
21.5.1. Compilación y ejecución del programa	
21.6. El entorno integrado de desarrollo (EID)	485
21.6.1. El menú File	
21.6.2., El menú Edit	490
21.6.3. El menú Search	491
21.6.4. El menú View	494
21.6.5. El menú Project	497
21.6.6. El menú Debug	499
21.6.7. El menú Tool	502
21.6.8. El menú Options	503
21.6.9. El menú Windows	507
21.6.10. El menú Help	509
21.7. Turbo C++3.1.	510
21.7.1. Puesta a punto de un programa	514
<b>Apéndice A. Los menús de turbo C ++ 3.0/1</b>	517
A.1. Menú Sytem	518
A.2. Menú File	519
A.3. Menu Edith	520
A.4. Menu Search (alt-s)	521
A.5. Menu Search (alt-r)	522
A.6. Menu Run (alt-c)	523
A.7. Menu Debug (alt-d)	524
A.8. Menu Project (alt-p)	525

A.9. menu Options (alt-o)	526
A.10. Menu Windows (alt-w)	527
A.11. Menu Help F1)	529
A.12. Resumen	529
<b>Apéndice B. Prioridad y asociativa de operadores</b>	<b>531</b>
<b>Bibliografía</b>	<b>535</b>
<b>Índice analítico</b>	<b>537</b>