
CONTENIDO

Prólogo	xxi
1. Conceptos de programación orientada a objetos	1
La evolución de la programación	2
Programación lineal	2
Programación modular	3
Programación estructurada	4
Abstracción de datos	5
¿Qué es la POO?	5
Trabajando con objetos	6
Definición de objetos	9
Clases	10
Mensajes: activación de objetos	12
Programa orientado a objetos	14
Herencia	14
Polimorfismo	16
Reutilización	18
Los lenguajes de POO	18
Smalltalk	19
Eiffel	20
Pascal orientado a objetos	20
Objective-C	21
C++	22
Los traductores de lenguajes	22
Compiladores C++	23
Compiladores C++ de código C	24
Compiladores C++ de código nativo	24
Creación de programas C++	25
Edición de programas fuentes	26
Compilación y enlace	29

Unix	29
DOS	29
Turbo C++ y Borland C++	29
Microsoft C/C++ 7.0	30
Symantec C++/Zortech C++	30
Enlazador	31
Utilización de bibliotecas	31
Ejecución del programa	32
Compilación separada	33
Make: una utilidad para compilación separada	33
Utilización de la herramienta <i>make</i>	34
El proceso de puesta a punto de programas	35
Resumen	37

2. C++: un C «mejor»	39
Orígenes de C++	40
Nuevas palabras reservadas	40
Comentarios	41
Identificadores	41
El primer programa C++	42
Constantes	44
Tipos de datos	46
Tipo carácter	46
Tipo vacío (<i>void</i>)	47
Tipos enumerados	47
Tipos referencia	50
Operadores	52
Operadores derivados	53
Operadores especiales de C++	53
Orden de prioridad y asociatividad de operadores	54
Declaraciones	55
Declaraciones y definiciones	55
Inicialización de variables	58
Moldes (<i>cast</i>)	59
El especificador constante (<i>const</i>)	60
Constantes C++ <i>versus</i> constantes C	61
Punteros y direcciones de constantes simbólicas	64
Punteros a un tipo de dato constante	64
Punteros constantes	65
Punteros constantes a variables constantes	65
Diferencias entre <i>const</i> de C++ y <i>const</i> de C	66
El especificador de tipo <i>volatile</i>	68
Conversiones de tipos	69
Conversiones no seguras	71
<i>sizeof</i> (<i>char</i>)	71

El tipo de dato <code>void</code>	72
Punteros a <code>void</code> (genéricos)	72
Salidas y entradas	74
Salida	74
Salida con formato	76
Entrada	76
Estructuras de control	78
for	78
while	79
do	79
if, if-else	79
switch	80
Salidas de bucles	82
goto	82
Sentencia nula	83
Sentencia return y función exit	83
El operador de resolución de ámbito <code>::</code>	84
Estructuras y uniones	85
Estructuras	86
Uniones	87
Uniones anónimas	88
Tipos definidos: typedef	89
Asignación dinámica de memoria: new y delete	90
new	90
delete	93
Ventajas de new y delete	95
Resumen	96
Ejercicios	96
3. Entradas y salidas básicas	99
La biblioteca <i>iostream</i>	99
Dispositivos estándar	101
Visualizar la salida	104
Ejemplos de E/S en C y C++	105
Manipuladores de salida	107
Control de caracteres	109
Conversión de bases numéricas	110
Control de formato	111
Entrada de datos	114
Lectura de líneas de datos	116
La jerarquía de clases <i>iostream</i>	117
Los manipuladores de formatos de bits	117
Formatos de coma flotante	120
Alineación de números de coma flotante en tablas	120
Entrada y salida estilo C	121

x Contenido

Comparación de programas C y C++ 122
Resumen 123
Ejercicios 123

4. Funciones 125

La función `main()` 126
Escritura de funciones en C++ 128
 Definición de funciones 128
 Declaración de una función (prototipos) 129
 Funciones que devuelven in tipo `void` 131
 Funciones que toman `void` como argumento 132
 Punteros a `void` en funciones 132
 Un ejercicio práctico de construcción de funciones 133
 Declaración *versus* definición: prototipos 134
 Verificación de tipos 137
Compilación separada 138
 Archivos de cabecera 138
 Prototipos de funciones y archivos de cabecera 141
Clases de almacenamiento en C++ 143
 Variables `register` 145
 Resumen de clases de almacenamiento 146
 Paso de argumentos 148
 Variable referencia 149
 Parámetros por valor 151
 Parámetros por referencia 152
 Comparación de paso por valor, paso por puntero y paso por referencia 156
 Argumentos *array* 158
Lista de argumentos 159
 El modificador `const` 159
 El modificador `volatile` 160
 Funciones con número de argumentos no especificado 161
 Las funciones del archivo *Stdarg.h* 162
 Argumentos por defecto (omisión) 164
Funciones que devuelven valores 170
 Funciones que devuelven un tipo `void` 170
 Funciones que devuelven referencias 171
 Comparación de retornos por valor, por puntero y por referencia. 172
✓ Funciones en línea (`inline`) 172
 Funciones en línea *versus* macros 174
 Funciones en línea y archivos de cabecera 175
 Ventajas y desventajas de las funciones en línea 176
 Funciones recursivas 177
 Enlace de tipos seguro 177
 Deformar nombres 178

Verificación estricta de tipos en C++	180
Resumen	180
Ejercicios	181
5. Objetos y clases	185
Abstracción de datos	185
Tipos de datos definidos por el usuario en C	186
Tipos abstractos de datos en C++	187
✓ Concepto de clase	188
Componentes de una clase	190
Control de acceso a una clase: visibilidad	191
Declaración de variables de una clase determinada	193
Declarar funciones miembro	194
Declaración de variable y ejecución de una función miembro ..	196
Representación gráfica de una clase	197
Las clases con estructuras	198
Las etiquetas de struct y union	199
Objetos	200
Acceso a los miembros de una clase	202
Ambito de una clase	203
Clases vacías	203
Clases anidadas	204
Especificadores de almacenamiento de los miembros de una clase.	207
Los miembros dato	207
Miembros estáticos de una clase	208
Miembros dato estáticos privados	209
Ambito de una clase	211
Especificadores de acceso a los miembros de una clase	213
Acceso protegido	213
Acceso público	214
Acceso privado	215
Funciones miembro	216
El puntero (this)	218
Funciones miembro estáticas (static)	221
Funciones miembro const	222
Funciones miembro en línea (inline)	223
Funciones miembro volatile	224
Funciones miembro especiales	225
✓ Constructores	225
Llamando a constructores	228
Constructores por defecto	229
Constructores con argumentos	229
Constructores con argumento por defecto	231
Mecanismo alternativo de paso de argumentos	233
Constructores sobrecargados	234

Constructores copiadores	235
El constructor de copia por defecto	237
Paso de objetos por valor	237
Retorno de objetos por valor	238
Creación de objetos	238
Destructores	239
Escritura de los destructores	241
Destructores públicos	242
Destructores privados	242
Creación y supresión dinámica de objetos	242
Amigas (friend)	244
Funciones amigas	244
Clases amigas	245
Una aplicación de clases: la clase Pila	246
Resumen	249
Ejercicios	249
6. Sobrecarga de funciones y operadores	255
Sobrecarga de funciones	255
Declaración de funciones sobrecargadas	258
Funciones no miembros sobrecargadas	259
Funciones miembro sobrecargadas	259
Sobrecarga de funciones amigas	260
La palabra reservada overload 	261
Sobrecarga de operadores	265
¿Cómo conoce el compilador qué función operador se utiliza?	268
¿Cuál es la prioridad de las funciones operador?	268
Restricciones en los operadores sobrecargados	268
El mecanismo de sobrecarga de operadores	269
Declaración de funciones operador	271
Funciones operador amigas	272
Consejos para sobrecargar con éxito operadores	272
Sobrecarga de operadores unitarios	274
Versiones prefija y postfija de los operadores ++ y -- 	277
Sobrecargar un operador unitario como función miembro	278
Sobrecarga de un operador unitario como una función amiga	279
Sobrecarga de operadores binarios	280
Sobrecarga de un operador binario como función miembro	280
Ejemplo de sobrecarga de un operador binario	281
Otro ejemplo de sobrecarga de un operador binario	282
Sobrecarga de un operador binario como una función amiga	284
Sobrecargando el operador de asignación	284
Sobrecargando el operador de llamada a funciones () 	286
Sobrecargando el operador subíndice [] 	287
Sobrecarga de operadores de flujo	288
Sobrecarga de flujo de salida	288

Sobrecarga de flujo de entrada	290
Conversión de datos y operadores de conversión forzada de tipos	291
Conversión entre tipos básicos	292
Conversión entre objetos y tipos básicos	293
Funciones de conversión	293
Sobrecarga de new y delete : asignación dinámica	295
Sobrecarga de new	296
Sobrecarga del operador delete	297
Manipulación de sobrecarga de operadores	298
Una aplicación de sobrecarga de operadores	300
Resumen	302
Ejercicios	303
7. Herencia y jerarquía de clases	307
Herencia: una visión general	308
Clases derivadas	309
Conceptos fundamentales de derivación	311
La herencia en C++	313
Creación de una clase derivada	314
Acceso a la clase base	316
¿Qué cosas se heredan y qué cosas no se heredan?	322
Clases de derivación	323
Derivación pública (public)	323
Derivación privada (private)	324
Derivación protegida (protected)	325
Ventajas e inconvenientes de la derivación <i>privada</i> y <i>protegida</i>	326
Conversiones	329
Conversiones a void*	331
Conversiones de tipos predefinidos a tipos definidos por el usuario	331
Conversiones de tipos definidos por el usuario a tipos predefinidos	332
Conversiones con herencia	333
Conversiones utilizadas cuando se pasan argumentos a funciones	334
Una aplicación elemental de herencia	334
Constructores y destructores en herencia	336
Llamada de constructores de la clase base	337
Orden de inicialización	341
Llamada al destructor de una clase base	343
Manipulación de clases base	344
Clases base públicas y privadas	344
La creación de una clase base con una sección protegida	346
Anulación de funciones miembro	348
Una aplicación de anulación de funciones miembro	349
Redefinición de funciones miembro heredadas	351

¿Qué función se utiliza?	352
Resolución de ámbito en redefinición de funciones	352
La jerarquía de clases	352
Una aplicación de jerarquía de clases	353
Herencia múltiple	356
Diagramas de herencia de clases	358
Sintaxis de la herencia múltiple	359
Ambigüedades en herencia múltiple	360
Uso del operador de resolución de ámbito	363
Constructores y destructores en herencia múltiple	364
Problemas que resuelve la herencia múltiple	366
Un ejemplo de utilización de constructores y destructores	367
Problemas de la herencia múltiple	369
Herencia repetida	372
Clases base virtuales y no virtuales	374
Constructores y destructores en clases derivadas	375
Invocación a los destructores	377
Uso de conversiones de tipo	378
Reglas de dominio	379
Clases base virtuales	382
Otro ejemplo de clases base virtuales	383
Inicialización de clases base virtuales	384
Resumen	386
Ejercicios	386

8. Funciones virtuales y polimorfismo 391

Ligadura dinámica	392
Ligadura estática	392
Ligadura dinámica	393
Un ejemplo de ligadura estática <i>versus</i> ligadura dinámica	393
Polimorfismo	395
✓ Funciones virtuales	397
Redefinición de funciones miembro de la base	402
Acceso a funciones virtuales a través de punteros	410
Acceso a funciones miembro con puntero	410
Acceso a funciones miembro virtuales con punteros	412
Limitaciones de funciones virtuales	412
Constructores y destructores virtuales	413
Funciones virtuales puras	413
Funciones virtuales puras y clases base abstractas	414
Destructores virtuales	415
✓ Clases abstractas	417
Generalización de clases	421
Utilización de clases abstractas	422
Polimorfismo	425

Una aplicación de polimorfismo: clase figura	425
Añadir una nueva clase a la clase figura	429
Resumen	430
Ejercicios	431
9. Entrada/salida avanzada (<i>Flujos:Streams</i>)	433
Flujos	433
La biblioteca <i>iostream</i>	437
Flujos predefinidos	438
Estructura de la biblioteca <i>iostream</i>	438
La jerarquía de clases <i>iostream</i>	440
Jerarquía de clases	443
Los operadores de inserción (<<) y de extracción (>>)	446
El operador de inserción <<	446
Funciones de salida de <i>ostream</i> : <code>put()</code> y <code>flush()</code>	451
El operador de extracción >>	453
Flujos de entrada	456
Función <code>get()</code>	459
Función <code>getline()</code>	460
Función <code>ignore()</code>	461
Función <code>read()</code>	462
Función <code>putback()</code>	462
Función <code>peek()</code>	463
E/S con formatos	464
Ajustar la anchura de los campos: <code>width()</code>	464
Control de precisión de formato: <code>precision()</code>	467
Control de formato con relleno de caracteres: <code>fill()</code>	468
Indicadores de formatos	470
Funciones miembro de control de indicadores de formato	471
Impresión de ceros a la derecha	471
Conceptos avanzados de la función: <code>setf()</code>	473
Repaso de manipuladores	478
Sobrecarga de los operadores <<, >>	482
Una aplicación del operador << sobrecargado	483
Una aplicación del operador >> sobrecargado	484
Una aplicación de sobrecarga de ambos operadores	485
Resumen	486
Ejercicios	487
10. Archivos	491
La biblioteca de flujos	491
Entrada y salida en archivos	492
Apertura de un archivo	493
Crear un archivo de texto	496
Leer un archivo de texto	497

Abrir múltiples archivos	499
Proceso desde la línea de órdenes	502
Verificación del estado de un flujo	504
Indicadores de error en un flujo	507
Un programa práctico de archivos	507
Añadir datos a un archivo	509
Archivos binarios	511
Utilizando archivos binarios para entrada y salida	512
Lectura/escritura de archivos binarios	514
Archivos binarios <i>versus</i> archivos de texto	515
La función <code>gcount</code>	516
Lectura/escritura de valores binarios de una clase	517
Archivos aleatorios (acceso directo)	518
Funciones de búsqueda: <code>seekp</code> y <code>tellp</code>	519
Una aplicación de lectura/escritura aleatoria	522
Lectura y escritura de objetos	524
Utilización de la impresora como un flujo	528
Resumen	529
Ejercicios	530
11. Plantillas (<i>templates</i>)	533
Clases genéricas	534
La simulación de funciones genéricas	535
Clases genéricas con macros	537
Clases genéricas realizadas con Herencia y Polimorfismo	539
Plantillas (<i>templates</i>)	541
Plantillas de funciones	541
Emulación de plantillas de funciones en <code>CC++</code> (sin <i>template</i>)	541
Definición de una plantilla de funciones	542
Una función genérica de ordenación	545
Utilización de plantillas de funciones con clases	547
Correspondencia de argumentos con funciones plantilla	547
Plantillas de clases	548
Declaración de una plantilla de clase	551
Una clase genérica NuevaPila	553
Definición de funciones miembro de una clase plantilla	553
Utilización de una clase plantilla	556
Argumentos de una plantilla	558
Más aplicaciones de plantillas	559
Resumen	562
Ejercicios	563
12. Tratamiento de excepciones	565
Concepto de excepciones	565
Lanzamiento (disparo) de excepciones	567

El bloque de prueba <code>try</code> y la cláusula <code>catch</code>	568
El manejador de excepciones y la cláusula <code>throw</code>	569
La lista <code>throw</code> de una función.....	570
Relanzamiento de excepciones.....	571
Un ejemplo.....	571
Aplicaciones prácticas de excepciones.....	573
Primera aplicación.....	573
Otra aplicación de excepciones.....	575
Una última aplicación de manejo de excepciones.....	576
Alternativas al tratamiento de excepciones.....	577
Resumen.....	578
Ejercicios.....	578
13. Construcción de programas en C++ (Compilación separada)....	579
Compilación separada de programas.....	579
Programas multiarchivo.....	581
Bibliotecas de clases.....	581
Almacenamiento <code>extern</code> y <code>static</code>	582
<code>extern</code>	582
<code>static</code>	583
Compilación condicional.....	585
Evitar definiciones múltiples.....	586
Visualizar mensajes de error.....	587
Reglas.....	588
Estructura de un programa C.....	589
Compilación separada de clases.....	590
Estructura de un programa C++.....	592
¿Qué son archivos de cabecera?.....	594
Inclusión de archivos.....	595
Programas multiarchivo.....	596
¿Qué se debe poner en un archivo fuente?.....	597
Referencias externas.....	597
Construcción de archivos proyecto.....	599
Abrir un proyecto.....	600
Añadir archivos fuente.....	601
Programas mixtos C/C++.....	602
Resumen.....	604
Ejercicios.....	605
14. Reutilización de software.....	607
La reutilización tradicional.....	607
Reutilización por herencia.....	609
Ventajas de la reutilización por herencia.....	609
Recompilaciones en C++.....	610
La reutilización como herramienta de diseño.....	611

Creación de programas orientados a objetos	612
Identificar las clases	612
Determinar las relaciones entre clase: jerarquías	614
Diseño de una biblioteca de clases	618
Crear una ventana	618
Aplicación lista enlazada	620
Aplicación pila	623
Aplicación cola	625
Otra aplicación pila	626
Resumen	629
Ejercicios	629
15. Desde C a C++. Consejos prácticos de programación	631
Consideraciones para transportar aplicaciones desde C	631
Diferencias de sintaxis C y C++	632
Palabras reservadas	632
Declaraciones y definiciones	633
Comentarios	633
Moldes	634
Enumeraciones	634
Constantes	635
Estructuras, uniones y clases	635
Constantes <code>char</code>	636
Uniones anónimas	636
El operador de resolución de ámbito	636
<code>asm</code>	637
<code>new</code> y <code>delete</code>	638
Punteros <code>void</code>	639
Funciones en C++	639
<code>main()</code>	639
Prototipos de funciones	640
Funciones en línea (<code>inline</code>)	641
Valores de argumentos por defecto	641
Verificación de tipos	643
Consejos de programación para programadores C	644
Programación con C y C++	652
¿Cómo llamar a funciones C desde un programa C++?	652
¿Cómo hacer los archivos de cabecera transportables entre C y C++?	654
Transferencia de C a C++	655
Resumen	655
Ejercicios	656
APENDICES	659
A. Sintaxis de C++	661

B. Palabras reservadas C/C++	667
C. Diccionario de palabras reservadas C++	673
D. Prioridad y asociatividad de operadores	695
E. Archivos de cabecera y macros ANSI C	699
F. Biblioteca de funciones ANSI C	713
G. La biblioteca de funciones <i>iostream</i>	735
H. Códigos fuente de programas seleccionados del libro	755
GLOSARIO	831
BIBLIOGRAFIA	840
INDICE	842