Contenido

Prefacio xviii

Antes de empezar xxxix

1 Introducción a las computadoras, Internet y World Wide Web 1

1.1 Introducción 2
1.2 ¿Qué es una computadora? 3
1.3 Organización de una computadora 4
1.4 Los primeros sistemas operativos 4
1.5 Computación personal, distribuida y cliente/servidor 5
1.6 Internet y World Wide Web 5
1.7 Web 2.0 6
1.8 Lenguajes máquina, lenguajes ensambladores y lenguajes de alto nivel 6
1.9 Historia de C y C++ 7
1.10 Biblioteca estándar de C++ 8
1.11 Historia de Java 9
1.12 FORTRAN, COBOL, Pascal y Ada 9
1.13 BASIC, Visual Basic, Visual C++, C# y .NET 10
1.14 Tendencia clave de software: la tecnología de los objetos 10
1.15 Entorno de desarrollo típico en C++ 11
1.16 Generalidades acerca de C++ y este libro 13
1.17 Prueba de una aplicación en C++ 14
1.18 Tecnologías de software 19
1.19 Programación de juegos con las bibliotecas Ogre 20
1.20 Futuro de C++: Bibliotecas Boost de código fuente abierto, TR1 y C++0x 20
1.21 Ejemplo práctico de Ingeniería de Software: introducción a la tecnología de objetos y el UML 21
1.22 Repaso 25
1.23 Recursos Web 25

2 Introducción a la programación en C++ 35

2.1 Introducción 36
2.2 Su primer programa en C++: imprimir una línea de texto 36
2.3 Modificación de nuestro primer programa en C++ 39
2.4 Otro programa en C++: suma de enteros 40
2.5 Conceptos acerca de la memoria 44
2.6 Aritmética 45
2.7 Toma de decisiones: operadores de igualdad y relacionales 47
2.8 (Opcional) Ejemplo práctico de Ingeniería de Software: cómo examinar la especificación de requerimientos del ATM 52
2.9 Repaso 59

3 Introducción a las clases y los objetos 67

3.1 Introducción 68
3.2 Clases, objetos, funciones miembro y miembros de datos 68
3.3 Generalidades acerca de los ejemplos del capítulo 69
3.4 Definición de una clase con una función miembro 70
3.5 Definición de una función miembro con un parámetro 72
3.6 Miembros de datos, funciones *establecer* y funciones *obtener* 75
3.7 Inicialización de objetos mediante constructores 81
3.8 Colocar una clase en un archivo separado para fines de reutilización 84
3.9 Separar la interfaz de la implementación 87
3.10 Validación de datos mediante funciones *establecer* 93
3.11 (Opcional) Ejemplo práctico de Ingeniería de Software: identificación de las clases en la especificación de requerimientos del ATM 97
3.12 Repaso 102

4 Instrucciones de control: parte 1 109

4.1 Introducción 110
4.2 Algoritmos 110
4.3 Seudocódigo 111
4.4 Estructuras de control 112
4.5 Instrucción de selección `if` 115
4.6 Instrucción de selección doble `if...else` 116
4.7 Instrucción de repetición `while` 120
4.8 Cómo formular algoritmos: repetición controlada por un contador 121
4.9 Cómo formular algoritmos: repetición controlada por un centinela 126
4.10 Cómo formular algoritmos: instrucciones de control anidadas 135
4.11 Operadores de asignación 139
4.12 Operadores de incremento y decremento 139
4.13 (Opcional) Ejemplo práctico de Ingeniería de Software: identificación de los atributos de las clases en el sistema ATM 142
4.14 Repaso 146

5 Instrucciones de control: parte 2 159

5.1 Introducción 160
5.2 Fundamentos de la repetición controlada por contador 160
5.3 Instrucción de repetición `for` 162
5.4 Ejemplos acerca del uso de la instrucción `for` 165
5.5 Instrucción de repetición `do...while` 169
5.6 Instrucción de selección múltiple `switch` 171
5.7 Instrucciones `break` y `continue` 179
5.8 Operadores lógicos 180
5.9 Confusión entre los operadores de igualdad (==) y de asignación (=) 184
5.10 Resumen sobre programación estructurada 185
5.11 (Opcional) Ejemplo práctico de Ingeniería de Software: cómo identificar los estados y actividades de los objetos en el sistema ATM 189
5.12 Repaso 193

6 Funciones y una introducción a la recursividad 203

6.1 Introducción 204
6.2 Componentes de los programas en C++ 205
6.3 Funciones matemáticas de la biblioteca 206

6.4 Definiciones de funciones con varios parámetros 207
6.5 Prototipos de funciones y coerción de argumentos 211
6.6 Archivos de encabezado de la Biblioteca estándar de C++ 213
6.7 Ejemplo práctico: generación de números aleatorios 215
6.8 Ejemplo práctico: juego de probabilidad, introducción a las enumeraciones 220
6.9 Clases de almacenamiento 223
6.10 Reglas de alcance 225
6.11 La pila de llamadas a funciones y los registros de activación 228
6.12 Funciones con listas de parámetros vacías 231
6.13 Funciones en línea 232
6.14 Referencias y parámetros de referencias 233
6.15 Argumentos predeterminados 237
6.16 Operador de resolución de ámbito unario 239
6.17 Sobrecarga de funciones 240
6.18 Plantillas de funciones 243
6.19 Recursividad 245
6.20 Ejemplo sobre el uso de la recursividad: serie de Fibonacci 247
6.21 Comparación entre recursividad e iteración 250
6.22 (Opcional) Ejemplo práctico de Ingeniería de Software: identificación de las operaciones de las clases en el sistema ATM 253
6.23 Repaso 258

7 Arreglos y vectores 277

7.1 Introducción 278
7.2 Arreglos 279
7.3 Declaración y creación de arreglos 280
7.4 Ejemplos acerca del uso de los arreglos 280
7.4.1 Declaración de un arreglo y uso de un ciclo para inicializar los elementos del arreglo 281
7.4.2 Inicialización de un arreglo en una declaración mediante una lista inicializadora 281
7.4.3 Especificación del tamaño de un arreglo con una variable constante y establecimiento de los elementos de un arreglo con cálculos 283
7.4.4 Suma de los elementos de un arreglo 285
7.4.5 Uso de gráficos de barra para mostrar los datos de un arreglo en forma gráfica 286
7.4.6 Uso de los elementos de un arreglo como contadores 287
7.4.7 Uso de arreglos para sintetizar los resultados de una encuesta 288
7.4.8 Uso de arreglos tipo carácter para almacenar y manipular cadenas 291
7.4.9 Arreglos locales estáticos y arreglos locales automáticos 292
7.5 Paso de arreglos a funciones 294
7.6 Ejemplo práctico: la clase `LibroCalificaciones` que usa un arreglo para almacenar las calificaciones 298
7.7 Búsqueda de datos en arreglos mediante la búsqueda lineal 304
7.8 Ordenamiento de arreglos mediante el ordenamiento por inserción 305
7.9 Arreglos multidimensionales 307
7.10 Ejemplo práctico: la clase `LibroCalificaciones` que usa un arreglo bidimensional 310
7.11 Introducción a la plantilla de clase `vector` de la Biblioteca estándar de C++ 316
7.12 (Opcional) Ejemplo práctico de Ingeniería de Software: colaboración entre los objetos en el sistema ATM 320
7.13 Repaso 326

8 Apuntadores y cadenas basadas en apuntadores 341

8.1 Introducción 342
8.2 Declaraciones e inicialización de variables apuntadores 342
8.3 Operadores de apuntadores 343
8.4 Paso de argumentos a funciones por referencia mediante apuntadores 346

8.5 Uso de const con apuntadores 349
8.6 Ordenamiento por selección mediante el uso del paso por referencia 355
8.7 Operador sizeof 358
8.8 Expresiones y aritmética de apuntadores 360
8.9 Relación entre apuntadores y arreglos 362
8.10 Arreglos de apuntadores 366
8.11 Ejemplo práctico: simulación para barajar y repartir cartas 367
8.12 Apuntadores a funciones 372
8.13 Introducción al procesamiento de cadenas basadas en apuntador 376
8.14 Repaso 384

9 Clases: un análisis más detallado, parte I 407

9.1 Introducción 408
9.2 Ejemplo práctico con la clase Tiempo 409
9.3 Alcance de las clases y acceso a los miembros de una clase 414
9.4 Separar la interfaz de la implementación 416
9.5 Funciones de acceso y funciones utilitarias 416
9.6 Ejemplo práctico de la clase Tiempo: constructores con argumentos predeterminados 419
9.7 Destructores 423
9.8 Cuándo se hacen llamadas a los constructores y destructores 424
9.9 Ejemplo práctico con la clase Tiempo: una trampa sutil (devolver una referencia a un miembro de datos private) 427
9.10 Asignación predeterminada a nivel de miembros 429
9.11 (Opcional) Ejemplo práctico de Ingeniería de Software: inicio de la programación de las clases del sistema ATM 431
9.12 Repaso 437

10 Clases: un análisis más detallado, parte 2 443

10.1 Introducción 444
10.2 Objetos const (constantes) y funciones miembro const 444
10.3 Composición: objetos como miembros de clases 452
10.4 Funciones friend y clases friend 458
10.5 Uso del apuntador this 461
10.6 Administración dinámica de memoria con los operadores new y delete 466
10.7 Miembros de clase static 467
10.8 Abstracción de datos y ocultamiento de información 472
10.8.1 Ejemplo: tipo de datos abstracto arreglo 473
10.8.2 Ejemplo: tipo de datos abstracto cadena 474
10.8.3 Ejemplo: tipo de datos abstracto cola 474
10.9 Clases contenedoras e iteradores 474
10.10 Clases proxy 475
10.11 Repaso 477

11 Sobrecarga de operadores: objetos String y Array 483

11.1 Introducción 484
11.2 Fundamentos de la sobrecarga de operadores 485
11.3 Restricciones acerca de la sobrecarga de operadores 485
11.4 Las funciones de operadores como clase miembro vs. funciones globales 487
11.5 Sobrecarga de los operadores de inserción de flujo y extracción de flujo 488
11.6 Sobrecarga de operadores unarios 491

11.7 Sobrecarga de operadores binarios 491
11.8 Ejemplo práctico: la clase Array 492
11.9 Conversión entre tipos 502
11.10 Ejemplo práctico: la clase String 502
11.11 Sobrecarga de ++ y -- 513
11.12 Ejemplo práctico: una clase Fecha 514
11.13 La clase string de la Biblioteca estándar 518
11.14 Constructores explicit 521
11.15 Repaso 524

12 Programación orientada a objetos: herencia 535

12.1 Introducción 536
12.2 Clases base y clases derivadas 537
12.3 Miembros protected 539
12.4 Relación entre las clases base y las clases derivadas 539
12.4.1 Creación y uso de una clase EmpleadoPorComision 540
12.4.2 Creación de una clase EmpleadoBaseMasComision sin usar la herencia 544
12.4.3 Creación de una jerarquía de herencia EmpleadoPorComision-Empleado-BaseMasComision 549
12.4.4 La jerarquía de herencia EmpleadoPorComision-EmpleadoBaseMasComision mediante el uso de datos protected 553
12.4.5 La jerarquía de herencia EmpleadoPorComision-EmpleadoBaseMasComision mediante el uso de datos private 559
12.5 Los constructores y destructores en las clases derivadas 566
12.6 Herencia public, protected y private 573
12.7 Ingeniería de software mediante la herencia 573
12.8 Repaso 574

13 Programación orientada a objetos: polimorfismo 579

13.1 Introducción 580
13.2 Ejemplos de polimorfismo 581
13.3 Relaciones entre los objetos en una jerarquía de herencia 582
13.3.1 Invocación de funciones de la clase base desde objetos de una clase derivada 583
13.3.2 Cómo orientar los apuntadores de una clase derivada a objetos de la clase base 589
13.3.3 Llamadas a funciones miembro de una clase derivada a través de apuntadores de la clase base 590
13.3.4 Funciones virtuales 591
13.3.5 Resumen de las asignaciones permitidas entre objetos y apuntadores de la clase base y de la clase derivada 596
13.4 Tipos de campos e instrucciones switch 597
13.5 Clases abstractas y funciones virtual puras 597
13.6 Ejemplo práctico: sistema de nómina mediante el uso de polimorfismo 599
13.6.1 Creación de la clase base abstracta Empleado 600
13.6.2 Creación de la clase derivada concreta EmpleadoAsalariado 603
13.6.3 Creación de la clase derivada concreta EmpleadoPorHoras 605
13.6.4 Creación de la clase derivada concreta EmpleadoPorComision 607
13.6.5 Creación de la clase derivada concreta indirecta EmpleadoBaseMasComision 608
13.6.6 Demostración del procesamiento polimórfico 610
13.7 (Opcional) Polimorfismo, funciones virtuales y vinculación dinámica "detrás de las cámaras" 614
13.8 Ejemplo práctico: sistema de nómina mediante el uso de polimorfismo e información de tipos en tiempo de ejecución con conversión descendente, dynamic_cast, typeid y type_info 617
13.9 Destructores virtuales 620
13.10 (Opcional) Ejemplo práctico de Ingeniería de Software: incorporación de la herencia en el sistema ATM 620
13.11 Repaso 627

14 Plantillas 631

14.1 Introducción 632
14.2 Plantillas de funciones 632
14.3 Sobrecarga de plantillas de funciones 635
14.4 Plantillas de clases 636
14.5 Parámetros sin tipo y tipos predeterminados para las plantillas de clases 641
14.6 Notas acerca de las plantillas y la herencia 642
14.7 Notas acerca de las plantillas y funciones friend 642
14.8 Notas acerca de las plantillas y miembros `static` 643
14.9 Repaso 643

15 Entrada y salida de flujos 648

15.1 Introducción 649
15.2 Flujos 650
15.2.1 Comparación entre flujos clásicos y flujos estándar 650
15.2.2 Archivos de encabezado de la biblioteca `iostream` 651
15.2.3 Clases y objetos de entrada/salida de flujos 651
15.3 Salida de flujos 653
15.3.1 Salida de variables `char *` 653
15.3.2 Salida de caracteres mediante la función miembro `put` 653
15.4 Entrada de flujos 654
15.4.1 Funciones miembro `get` y `getline` 654
15.4.2 Funciones miembro `peek`, `putback` e `ignore` de `istream` 657
15.4.3 E/S con seguridad de tipos 657
15.5 E/S sin formato mediante el uso de `read`, `write` y `gcount` 657
15.6 Introducción a los manipuladores de flujos 658
15.6.1 Base de flujos integrales: `dec`, `oct`, `hex` y `setbase` 658
15.6.2 Precisión de punto flotante (`precision`, `setprecision`) 659
15.6.3 Anchura de campos (`width`, `setw`) 660
15.6.4 Manipuladores de flujos de salida definidos por el usuario 662
15.7 Estados de formato de flujos y manipuladores de flujos 663
15.7.1 Ceros a la derecha y puntos decimales (`showpoint`) 663
15.7.2 Justificación (`left`, `right` e `internal`) 664
15.7.3 Relleno de caracteres (`fill`, `setfill`) 665
15.7.4 Base de flujos integrales (`dec`, `oct`, `hex`, `showbase`) 667
15.7.5 Números de punto flotante: notación científica y fija (`scientific`, `fixed`) 667
15.7.6 Control de mayúsculas/minúsculas (`uppercase`) 668
15.7.7 Especificación de formato booleano (`boolalpha`) 669
15.7.8 Establecer y restablecer el estado de formato mediante la función miembro `flags` 670
15.8 Estados de error de los flujos 671
15.9 Enlazar un flujo de salida a un flujo de entrada 673
15.10 Repaso 673

16 Manejo de excepciones 682

16.1 Introducción 683
16.2 Generalidades acerca del manejo de excepciones 684
16.3 Ejemplo: manejo de un intento de dividir entre cero 684
16.4 Cuándo utilizar el manejo de excepciones 689
16.5 Volver a lanzar una excepción 690
16.6 Especificaciones de excepciones 691
16.7 Procesamiento de excepciones inesperadas 692
16.8 Limpieza de la pila 692
16.9 Constructores, destructores y manejo de excepciones 694

16.10 Excepciones y herencia 694
16.11 Procesamiento de las fallas de new 694
16.12 La clase auto_ptr y la asignación dinámica de memoria 698
16.13 Jerarquía de excepciones de la Biblioteca estándar 700
16.14 Otras técnicas para manejar errores 701
16.15 Repaso 702

17 Procesamiento de archivos 708

17.1 Introducción 709
17.2 Jerarquía de datos 709
17.3 Archivos y flujos 711
17.4 Creación de un archivo secuencial 712
17.5 Cómo leer datos de un archivo secuencial 715
17.6 Actualización de archivos secuenciales 720
17.7 Archivos de acceso aleatorio 721
17.8 Creación de un archivo de acceso aleatorio 721
17.9 Cómo escribir datos al azar a un archivo de acceso aleatorio 726
17.10 Cómo leer de un archivo de acceso aleatorio en forma secuencial 728
17.11 Ejemplo práctico: un programa para procesar transacciones 730
17.12 Generalidades acerca de la serialización de objetos 735
17.13 Repaso 736

18 La clase string y el procesamiento de flujos de cadena 745

18.1 Introducción 746
18.2 Asignación y concatenación de objetos string 747
18.3 Comparación de objetos string 749
18.4 Subcadenas 751
18.5 Intercambio de objetos string 752
18.6 Características de los objetos string 752
18.7 Búsqueda de subcadenas y caracteres en un objeto string 754
18.8 Reemplazo de caracteres en un objeto string 756
18.9 Inserción de caracteres en un objeto string 758
18.10 Conversión a cadenas estilo C 758
18.11 Iteradores 760
18.12 Procesamiento de flujos de cadena 761
18.13 Repaso 764

19 Búsqueda y ordenamiento 769

19.1 Introducción 770
19.2 Algoritmos de búsqueda 770
19.2.1 Eficiencia de la búsqueda lineal 770
19.2.2 Búsqueda binaria 772
19.3 Algoritmos de ordenamiento 776
19.3.1 Eficiencia del ordenamiento por selección 776
19.3.2 Eficiencia del ordenamiento por inserción 777
19.3.3 Ordenamiento por combinación (una implementación recursiva) 777
19.4 Repaso 783

20 Estructuras de datos 788

20.1 Introducción 789
20.2 Clases autorreferenciadas 790

20.3 Asignación dinámica de memoria y estructuras de datos 790
20.4 Listas enlazadas 791
20.5 Pilas 804
20.6 Colas 807
20.7 Árboles 810
20.8 Repaso 818
Sección especial: construya su propio compilador 826

21 Bits, caracteres, cadenas estilo C y estructuras 837

21.1 Introducción 838
21.2 Definiciones de estructuras 838
21.3 Inicialización de estructuras 840
21.4 Uso de estructuras con funciones 840
21.5 `typedef` 840
21.6 Ejemplo: simulación para barajar y repartir cartas de alto rendimiento 841
21.7 Operadores a nivel de bits 843
21.8 Campos de bits 851
21.9 Biblioteca de manejo de caracteres 854
21.10 Funciones de conversión de cadenas basadas en apuntador 859
21.11 Funciones de búsqueda de la biblioteca de manejo de cadenas basadas en apuntador 863
21.12 Funciones de memoria de la biblioteca de manejo de cadenas basadas en apuntador 867
21.13 Repaso 871

22 Biblioteca de plantillas estándar (STL) 881

22.1 Introducción a la Biblioteca de plantillas estándar (STL) 882
22.1.1 Introducción a los contenedores 884
22.1.2 Introducción a los iteradores 887
22.1.3 Introducción a los algoritmos 892
22.2 Contenedores de secuencia 893
22.2.1 Contenedor de secuencia `vector` 894
22.2.2 Contenedor de secuencia `list` 900
22.2.3 Contenedor de secuencia `deque` 903
22.3 Contenedores asociativos 904
22.3.1 Contenedor asociativo `multiset` 904
22.3.2 Contenedor asociativo `set` 907
22.3.3 Contenedor asociativo `multimap` 908
22.3.4 Contenedor asociativo `map` 910
22.4 Adaptadores de contenedores 911
22.4.1 Adaptador `stack` 911
22.4.2 Adaptador `queue` 913
22.4.3 Adaptador `priority_queue` 914
22.5 Algoritmos 915
22.5.1 `fill`, `fill_n`, `generate` y `generate_n` 916
22.5.2 `equal`, `mismatch` y `lexicographical_compare` 917
22.5.3 `remove`, `remove_if`, `remove_copy` y `remove_copy_if` 919
22.5.4 `replace`, `replace_if`, `replace_copy` y `replace_copy_if` 921
22.5.5 Algoritmos matemáticos 923
22.5.6 Algoritmos básicos de búsqueda y ordenamiento 926
22.5.7 `swap`, `iter_swap` y `swap_ranges` 928
22.5.8 `copy_backward`, `merge`, `unique` y `reverse` 929
22.5.9 `inplace_merge`, `unique_copy` y `reverse_copy` 931
22.5.10 Operaciones `set` 933
22.5.11 `lower_bound`, `upper_bound` y `equal_range` 935
22.5.12 Ordenamiento de montón (heapsort) 937

22.5.13 `min` y `max` 939
22.5.14 Algoritmos de la STL que no se cubren en este capítulo 940
22.6 La clase `bitset` 941
22.7 Objetos de funciones 944
22.8 Conclusión 947
22.9 Recursos Web de la STL 947

23 Programación de juegos con Ogre 955

23.1 Introducción 956
23.2 Instalación de Ogre, OgreAL y OpenAL 956
23.3 Fundamentos de la programación de juegos 956
23.4 El juego de Pong: recorrido a través del código 959
23.4.1 Inicialización de Ogre 959
23.4.2 Creación de una escena 967
23.4.3 Agregar elementos a la escena 968
23.4.4 Animación y temporizadores 978
23.4.5 Entrada del usuario 979
23.4.6 Detección de colisiones 980
23.4.7 Sonido 984
23.4.8 Recursos 985
23.4.9 Controlador de Pong 985
23.5 Repaso 986
23.6 Recursos Web de Ogre 987

24 Bibliotecas Boost, Reporte técnico 1 y C++0x 995

24.1 Introducción 996
24.2 Centros de recursos de C++ (y relacionados) en línea de Deitel 996
24.3 Bibliotecas Boost 996
24.4 Cómo agregar una nueva biblioteca a Boost 997
24.5 Instalación de las Bibliotecas Boost 997
24.6 Las Bibliotecas Boost en el Reporte técnico 1 (TR1) 997
24.7 Uso de expresiones regulares con la biblioteca `Boost.Regex` 1000
24.7.1 Ejemplo de una expresión regular 1000
24.7.2 Cómo validar la entrada del usuario mediante expresiones regulares 1002
24.7.3 Cómo reemplazar y dividir cadenas 1005
24.8 Apuntadores inteligentes con `Boost.Smart_ptr` 1007
24.8.1 Uso de `shared_ptr` y conteo de referencias 1007
24.8.2 `weak_ptr`: observador de `shared_ptr` 1011
24.9 Reporte técnico 1 1016
24.10 C++0x 1017
24.11 Cambios en el lenguaje básico 1017
24.12 Repaso 1021

25 Otros temas 1028

25.1 Introducción 1029
25.2 Operador `const_cast` 1029
25.3 `Espacios de nombres` 1030
25.4 Palabras clave de operadores 1034
25.5 Miembros de clases `mutable` 1036
25.6 Apuntadores a miembros de clases (`.*` y `->*`) 1037
25.7 Herencia múltiple 1039
25.8 Herencia múltiple y clases base `virtual` 1043
25.9 Repaso 1047

Las páginas 1051 a 1156, que corresponden a los apéndices, se encuentran en formato PDF en el CD-ROM que acompaña a este libro

A Tabla de precedencia de operadores y asociatividad 1051

A.1 Precedencia de operadores 1051

B Conjunto de caracteres ASCII 1053

C Tipos fundamentales 1054

D Sistemas numéricos 1056

D.1 Introducción 1057
D.2 Abreviatura de los números binarios como números octales y hexadecimales 1059
D.3 Conversión de números octales y hexadecimales a binarios 1060
D.4 Conversión de un número binario, octal o hexadecimal a decimal 1061
D.5 Conversión de un número decimal a binario, octal o hexadecimal 1061
D.6 Números binarios negativos: notación de complemento a dos 1062

E Temas sobre código heredado de C 1067

E.1 Introducción 1068
E.2 Redirección de la entrada/salida en sistemas UNIX/LINUX/Mac OS X y Windows 1068
E.3 Listas de argumentos de longitud variable 1069
E.4 Uso de argumentos de línea de comandos 1071
E.5 Observaciones acerca de la compilación de programas con varios archivos de código fuente 1072
E.6 Terminación de los programas con `exit` y `atexit` 1074
E.7 Calificador de tipo `volatile` 1075
E.8 Sufijos para constantes enteras y de punto flotante 1075
E.9 Manejo de señales 1076
E.10 Asignación dinámica de memoria con `calloc` y `realloc` 1078
E.11 Bifurcación incondicional: `goto` 1078
E.12 Uniones 1080
E.13 Especificaciones de vinculación 1082
E.14 Repaso 1083

F Preprocesador 1088

F.1 Introducción 1089
F.2 La directiva del procesador `#include` 1089
F.3 La directiva del preprocesador `#define`: constantes simbólicas 1090
F.4 La directiva del preprocesador `#define`: macros 1090
F.5 Compilación condicional 1092
F.6 Las directivas del preprocesador `#error` y `#pragma` 1092
F.7 Los operadores # y ## 1093
F.8 Constantes simbólicas predefinidas 1093
F.9 Aserciones 1093
F.10 Repaso 1094

G Código del caso de estudio del ATM 1098

G.1 Implementación del caso de estudio del ATM 1098
G.2 La clase ATM 1099
G.3 La clase Pantalla 1104
G.4 La clase Teclado 1105
G.5 La clase DispensadorEfectivo 1106
G.6 La clase RanuraDeposito 1108
G.7 La clase Cuenta 1109
G.8 La clase BaseDatosBanco 1111
G.9 La clase Transaccion 1114
G.10 La clase SolicitudSaldo 1115
G.11 La clase Retiro 1117
G.12 La clase Deposito 1121
G.13 El programa de prueba EjemploPracticoATM.cpp 1124
G.14 Repaso 1124

H UML 2: tipos de diagramas adicionales 1125

H.1 Introducción 1125
H.2 Tipos de diagramas adicionales 1125

I Uso del depurador de Visual Studio 1127

I.1 Introducción 1128
I.2 Los puntos de interrupción y el comando **Continuar** 1128
I.3 Las ventanas **Variables locales** e **Inspección** 1132
I.4 Control de la ejecución mediante los comandos **Paso a paso por instrucciones**, **Paso a paso por procedimientos**, **Paso a paso para salir** y **Continuar** 1135
I.5 La ventana **Automático** 1137
I.6 Repaso 1138

J Uso del depurador de GNU C++ 1141

J.1 Introducción 1142
J.2 Los puntos de interrupción y los comandos run, stop, continue y print 1142
J.3 Los comandos print y set 1147
J.4 Control de la ejecución mediante los comandos step, finish y next 1149
J.5 El comando watch 1151
J.6 Repaso 1153

Bibliografía 1159

Índice 1165