Chapters 24–26 and Appendices F–K are PDF documents posted online at the book’s Companion Website, which is accessible from www.pearsonhighered.com/deitel.

## Preface

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction to Computers and C++</td>
<td>1</td>
</tr>
<tr>
<td>1.1</td>
<td>Introduction</td>
<td>2</td>
</tr>
<tr>
<td>1.2</td>
<td>Computers and the Internet in Industry and Research</td>
<td>2</td>
</tr>
<tr>
<td>1.3</td>
<td>Hardware and Software</td>
<td>5</td>
</tr>
<tr>
<td>1.3.1</td>
<td>Moore’s Law</td>
<td>6</td>
</tr>
<tr>
<td>1.3.2</td>
<td>Computer Organization</td>
<td>6</td>
</tr>
<tr>
<td>1.4</td>
<td>Data Hierarchy</td>
<td>7</td>
</tr>
<tr>
<td>1.5</td>
<td>Machine Languages, Assembly Languages and High-Level Languages</td>
<td>9</td>
</tr>
<tr>
<td>1.6</td>
<td>C++</td>
<td>10</td>
</tr>
<tr>
<td>1.7</td>
<td>Programming Languages</td>
<td>11</td>
</tr>
<tr>
<td>1.8</td>
<td>Introduction to Object Technology</td>
<td>14</td>
</tr>
<tr>
<td>1.9</td>
<td>Typical C++ Development Environment</td>
<td>17</td>
</tr>
<tr>
<td>1.10</td>
<td>Test-Driving a C++ Application</td>
<td>19</td>
</tr>
<tr>
<td>1.11</td>
<td>Operating Systems</td>
<td>25</td>
</tr>
<tr>
<td>1.11.1</td>
<td>Windows—A Proprietary Operating System</td>
<td>25</td>
</tr>
<tr>
<td>1.11.2</td>
<td>Linux—An Open-Source Operating System</td>
<td>26</td>
</tr>
<tr>
<td>1.11.3</td>
<td>Apple’s OS X; Apple’s iOS for iPhone®, iPad® and iPod Touch® Devices</td>
<td>26</td>
</tr>
<tr>
<td>1.11.4</td>
<td>Google’s Android</td>
<td>27</td>
</tr>
<tr>
<td>1.12</td>
<td>The Internet and World Wide Web</td>
<td>27</td>
</tr>
<tr>
<td>1.13</td>
<td>Some Key Software Development Terminology</td>
<td>29</td>
</tr>
<tr>
<td>1.14</td>
<td>C++11 and the Open Source Boost Libraries</td>
<td>31</td>
</tr>
<tr>
<td>1.15</td>
<td>Keeping Up to Date with Information Technologies</td>
<td>32</td>
</tr>
<tr>
<td>1.16</td>
<td>Web Resources</td>
<td>33</td>
</tr>
</tbody>
</table>

## 2 Introduction to C++ Programming; Input/Output and Operators

### 2.1 Introduction

38
## Contents

2.2 First Program in C++: Printing a Line of Text 39  
2.3 Modifying Our First C++ Program 43  
2.4 Another C++ Program: Adding Integers 44  
2.5 Memory Concepts 48  
2.6 Arithmetic 49  
2.7 Decision Making: Equality and Relational Operators 53  
2.8 Wrap-Up 57  

3 Introduction to Classes, Objects and Strings 66  
3.1 Introduction 67  
3.2 Defining a Class with a Member Function 67  
3.3 Defining a Member Function with a Parameter 70  
3.4 Data Members, set Member Functions and get Member Functions 74  
3.5 Initializing Objects with Constructors 79  
3.6 Placing a Class in a Separate File for Reusability 83  
3.7 Separating Interface from Implementation 87  
3.8 Validating Data with set Functions 92  
3.9 Wrap-Up 97  

4 Control Statements: Part 1; Assignment, ++ and -- Operators 104  
4.1 Introduction 105  
4.2 Algorithms 105  
4.3 Pseudocode 106  
4.4 Control Structures 107  
4.5 if Selection Statement 110  
4.6 if...else Double-Selection Statement 112  
4.7 while Repetition Statement 116  
4.8 Formulating Algorithms: Counter-Controlled Repetition 118  
4.9 Formulating Algorithms: Sentinel-Controlled Repetition 124  
4.10 Formulating Algorithms: Nested Control Statements 134  
4.11 Assignment Operators 139  
4.12 Increment and Decrement Operators 140  
4.13 Wrap-Up 143  

5 Control Statements: Part 2; Logical Operators 157  
5.1 Introduction 158  
5.2 Essentials of Counter-Controlled Repetition 158  
5.3 for Repetition Statement 159
### Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.4 Examples Using the for Statement</td>
<td>163</td>
</tr>
<tr>
<td>5.5 do...while Repetition Statement</td>
<td>168</td>
</tr>
<tr>
<td>5.6 switch Multiple-Selection Statement</td>
<td>169</td>
</tr>
<tr>
<td>5.7 break and continue Statements</td>
<td>178</td>
</tr>
<tr>
<td>5.8 Logical Operators</td>
<td>180</td>
</tr>
<tr>
<td>5.9 Confusing the Equality (==) and Assignment (≡) Operators</td>
<td>185</td>
</tr>
<tr>
<td>5.10 Structured Programming Summary</td>
<td>186</td>
</tr>
<tr>
<td>5.11 Wrap-Up</td>
<td>191</td>
</tr>
<tr>
<td>6 Functions and an Introduction to Recursion</td>
<td>201</td>
</tr>
<tr>
<td>6.1 Introduction</td>
<td>202</td>
</tr>
<tr>
<td>6.2 Program Components in C++</td>
<td>203</td>
</tr>
<tr>
<td>6.3 Math Library Functions</td>
<td>204</td>
</tr>
<tr>
<td>6.4 Function Definitions with Multiple Parameters</td>
<td>205</td>
</tr>
<tr>
<td>6.5 Function Prototypes and Argument Coercion</td>
<td>210</td>
</tr>
<tr>
<td>6.6 C++ Standard Library Headers</td>
<td>212</td>
</tr>
<tr>
<td>6.7 Case Study: Random Number Generation</td>
<td>214</td>
</tr>
<tr>
<td>6.8 Case Study: Game of Chance; Introducing enum</td>
<td>219</td>
</tr>
<tr>
<td>6.9 C++11 Random Numbers</td>
<td>224</td>
</tr>
<tr>
<td>6.10 Storage Classes and Storage Duration</td>
<td>225</td>
</tr>
<tr>
<td>6.11 Scope Rules</td>
<td>228</td>
</tr>
<tr>
<td>6.12 Function Call Stack and Activation Records</td>
<td>231</td>
</tr>
<tr>
<td>6.13 Functions with Empty Parameter Lists</td>
<td>235</td>
</tr>
<tr>
<td>6.14 Inline Functions</td>
<td>236</td>
</tr>
<tr>
<td>6.15 References and Reference Parameters</td>
<td>237</td>
</tr>
<tr>
<td>6.16 Default Arguments</td>
<td>240</td>
</tr>
<tr>
<td>6.17 Unary Scope Resolution Operator</td>
<td>242</td>
</tr>
<tr>
<td>6.18 Function Overloading</td>
<td>243</td>
</tr>
<tr>
<td>6.19 Function Templates</td>
<td>246</td>
</tr>
<tr>
<td>6.20 Recursion</td>
<td>248</td>
</tr>
<tr>
<td>6.21 Example Using Recursion: Fibonacci Series</td>
<td>252</td>
</tr>
<tr>
<td>6.22 Recursion vs. Iteration</td>
<td>255</td>
</tr>
<tr>
<td>6.23 Wrap-Up</td>
<td>258</td>
</tr>
<tr>
<td>7 Class Templates array and vector; Catching Exceptions</td>
<td>278</td>
</tr>
<tr>
<td>7.1 Introduction</td>
<td>279</td>
</tr>
<tr>
<td>7.2 arrays</td>
<td>279</td>
</tr>
<tr>
<td>7.3 Declaring arrays</td>
<td>281</td>
</tr>
</tbody>
</table>
Contents

7.4 Examples Using arrays 281
7.4.1 Declaring an array and Using a Loop to Initialize the array’s Elements 281
7.4.2 Initializing an array in a Declaration with an Initializer List 282
7.4.3 Specifying an array’s Size with a Constant Variable and Setting array Elements with Calculations 283
7.4.4 Summing the Elements of an array 286
7.4.5 Using Bar Charts to Display array Data Graphically 286
7.4.6 Using the Elements of an array as Counters 288
7.4.7 Using arrays to Summarize Survey Results 289
7.4.8 Static Local arrays and Automatic Local arrays 291
7.5 Range-Based for Statement 293
7.6 Case Study: Class GradeBook Using an array to Store Grades 295
7.7 Sorting and Searching arrays 302
7.8 Multidimensional arrays 304
7.9 Case Study: Class GradeBook Using a Two-Dimensional array 307
7.10 Introduction to C++ Standard Library Class Template vector 314
7.11 Wrap-Up 320

8 Pointers 334
8.1 Introduction 335
8.2 Pointer Variable Declarations and Initialization 335
8.3 Pointer Operators 337
8.4 Pass-by-Reference with Pointers 339
8.5 Built-In Arrays 344
8.6 Using const with Pointers 346
8.6.1 Nonconstant Pointer to Nonconstant Data 347
8.6.2 Nonconstant Pointer to Constant Data 347
8.6.3 Constant Pointer to Nonconstant Data 348
8.6.4 Constant Pointer to Constant Data 349
8.7 sizeof Operator 350
8.8 Pointer Expressions and Pointer Arithmetic 353
8.9 Relationship Between Pointers and Built-In Arrays 355
8.10 Pointer-Based Strings 358
8.11 Wrap-Up 361

9 Classes: A Deeper Look; Throwing Exceptions 377
9.1 Introduction 378
11.3 Relationship between Base and Derived Classes
11.3.1 Creating and Using a CommissionEmployee Class 486
11.3.2 Creating a BasePlusCommissionEmployee Class Without Using Inheritance 491
11.3.3 Creating a CommissionEmployee–BasePlusCommissionEmployee Inheritance Hierarchy 497
11.3.4 CommissionEmployee–BasePlusCommissionEmployee Inheritance Hierarchy Using protected Data 501
11.3.5 CommissionEmployee–BasePlusCommissionEmployee Inheritance Hierarchy Using private Data 504
11.4 Constructors and Destructors in Derived Classes 509

12 Object-Oriented Programming: Polymorphism 517
12.1 Introduction 518
12.2 Introduction to Polymorphism: Polymorphic Video Game 519
12.3 Relationships Among Objects in an Inheritance Hierarchy 519
12.3.1 Invoking Base-Class Functions from Derived-Class Objects 520
12.3.2 Aiming Derived-Class Pointers at Base-Class Objects 523
12.3.3 Derived-Class Member-Function Calls via Base-Class Pointers 524
12.3.4 Virtual Functions and Virtual Destructors 526
12.4 Type Fields and switch Statements 533
12.5 Abstract Classes and Pure virtual Functions 533
12.6 Case Study: Payroll System Using Polymorphism 535
12.6.1 Creating Abstract Base Class Employee 536
12.6.2 Creating Concrete Derived Class SalariedEmployee 540
12.6.3 Creating Concrete Derived Class CommissionEmployee 542
12.6.4 Creating Indirect Concrete Derived Class BasePlusCommissionEmployee 544
12.6.5 Demonstrating Polymorphic Processing 546
12.7 (Optional) Polymorphism, Virtual Functions and Dynamic Binding “Under the Hood” 550
12.8 Case Study: Payroll System Using Polymorphism and Runtime Type Information with Downcasting, dynamic_cast, typeid and type_info 553
12.9 Wrap-Up 557
13 Stream Input/Output: A Deeper Look  
13.1 Introduction  
13.2 Streams  
13.2.1 Classic Streams vs. Standard Streams  
13.2.2 istream Library Headers  
13.2.3 Stream Input/Output Classes and Objects  
13.3 Stream Output  
13.3.1 Output of char * Variables  
13.3.2 Character Output Using Member Function put  
13.4 Stream Input  
13.4.1 get and getline Member Functions  
13.4.2 istream Member Functions peek, putback and ignore  
13.4.3 Type-Safe I/O  
13.5 Unformatted I/O Using read, write and gcount  
13.6 Introduction to Stream Manipulators  
13.6.1 Integral Stream Base: dec, oct, hex and setbase  
13.6.2 Floating-Point Precision (precision, setprecision)  
13.6.3 Field Width (width, setw)  
13.6.4 User-Defined Output Stream Manipulators  
13.7 Stream Format States and Stream Manipulators  
13.7.1 Trailing Zeros and Decimal Points (showpoint)  
13.7.2 Justification (left, right and internal)  
13.7.3 Padding (fill, setfill)  
13.7.4 Integral Stream Base (dec, oct, hex, showbase)  
13.7.5 Floating-Point Numbers; Scientific and Fixed Notation (scientific, fixed)  
13.7.6 Uppercase/Lowercase Control (uppercase)  
13.7.7 Specifying Boolean Format (boolalpha)  
13.7.8 Setting and Resetting the Format State via Member Function flags  
13.8 Stream Error States  
13.9 Tying an Output Stream to an Input Stream  
13.10 Wrap-Up  

14 File Processing  
14.1 Introduction  
14.2 Files and Streams  
14.3 Creating a Sequential File  
14.4 Reading Data from a Sequential File  
14.5 Updating Sequential Files
14.6 Random-Access Files  611
14.7 Creating a Random-Access File  612
14.8 Writing Data Randomly to a Random-Access File  617
14.9 Reading from a Random-Access File Sequentially  619
14.10 Case Study: A Transaction-Processing Program  621
14.11 Object Serialization  628
14.12 Wrap-Up  628

15  Standard Library Containers and Iterators  638
15.1 Introduction  639
15.2 Introduction to Containers  640
15.3 Introduction to Iterators  644
15.4 Introduction to Algorithms  649
15.5 Sequence Containers  649
   15.5.1 vector Sequence Container  650
   15.5.2 list Sequence Container  658
   15.5.3 deque Sequence Container  662
15.6 Associative Containers  664
   15.6.1 multiset Associative Container  665
   15.6.2 set Associative Container  668
   15.6.3 multimap Associative Container  669
   15.6.4 map Associative Container  671
15.7 Container Adapters  673
   15.7.1 stack Adapter  673
   15.7.2 queue Adapter  675
   15.7.3 priority_queue Adapter  676
15.8 Class bitset  677
15.9 Wrap-Up  679

16  Standard Library Algorithms  690
16.1 Introduction  691
16.2 Minimum Iterator Requirements  691
16.3 Algorithms  693
   16.3.1 fill, fill_n, generate and generate_n  693
   16.3.2 equal, mismatch and lexicographical_compare  695
   16.3.3 remove, remove_if, remove_copy and remove_copy_if  697
   16.3.4 replace, replace_if, replace_copy and replace_copy_if  700
16.3.5 Mathematical Algorithms  702
16.3.6 Basic Searching and Sorting Algorithms  706
16.3.7 swap, iter_swap and swap_ranges 710
16.3.8 copy_backward, merge, unique and reverse 711
16.3.9 inplace_merge, unique_copy and reverse_copy 714
16.3.10 Set Operations 716
16.3.11 lower_bound, upper_bound and equal_range 719
16.3.12 Heapsort 721
16.3.13 min, max, minmax and minmax_element 724
16.4 Function Objects 726
16.5 Lambda Expressions 729
16.6 Standard Library Algorithm Summary 730
16.7 Wrap-Up 732

17 Exception Handling: A Deeper Look 740
17.1 Introduction 741
17.2 Example: Handling an Attempt to Divide by Zero 741
17.3 Rethrowing an Exception 747
17.4 Stack Unwinding 748
17.5 When to Use Exception Handling 750
17.6 Constructors, Destorctors and Exception Handling 751
17.7 Exceptions and Inheritance 752
17.8 Processing new Failures 752
17.9 Class unique_ptr and Dynamic Memory Allocation 755
17.10 Standard Library Exception Hierarchy 758
17.11 Wrap-Up 759

18 Introduction to Custom Templates 765
18.1 Introduction 766
18.2 Class Templates 766
18.3 Function Template to Manipulate a Class-Template Specialization Object 771
18.4 Notype Parameters 773
18.5 Default Arguments for Template Type Parameters 773
18.6 Overloading Function Templates 774
18.7 Wrap-Up 774

19 Custom Templatized Data Structures 777
19.1 Introduction 778
19.2 Self-Referential Classes 779
19.3 Linked Lists 780
## Contents

19.4 Stacks 794  
19.5 Queues 799  
19.6 Trees 803  
19.7 Wrap-Up 811  

### 20 Searching and Sorting 822

20.1 Introduction 823  
20.2 Searching Algorithms  
20.2.1 Linear Search 824  
20.2.2 Binary Search 827  
20.3 Sorting Algorithms  
20.3.1 Insertion Sort 832  
20.3.2 Selection Sort 834  
20.3.3 Merge Sort (A Recursive Implementation) 837  
20.4 Wrap-Up 843  

### 21 Class string and String Stream Processing: A Deeper Look 849

21.1 Introduction 850  
21.2 string Assignment and Concatenation 851  
21.3 Comparing strings 853  
21.4 Substrings 856  
21.5 Swapping strings 856  
21.6 string Characteristics 857  
21.7 Finding Substrings and Characters in a string 859  
21.8 Replacing Characters in a string 861  
21.9 Inserting Characters into a string 863  
21.10 Conversion to Pointer-Based char * Strings 864  
21.11 Iterators 865  
21.12 String Stream Processing 867  
21.13 C++11 Numeric Conversion Functions 870  
21.14 Wrap-Up 871  

### 22 Bits, Characters, C Strings and structs 879

22.1 Introduction 880  
22.2 Structure Definitions 880  
22.3 typedef 882  
22.4 Example: Card Shuffling and Dealing Simulation 882
22.5 Bitwise Operators  885
22.6 Bit Fields  894
22.7 Character-Handling Library  897
22.8 C String-Manipulation Functions  903
22.9 C String-Conversion Functions  910
22.10 Search Functions of the C String-Handling Library  915
22.11 Memory Functions of the C String-Handling Library  919
22.12 Wrap-Up  923

23 Other Topics  938
23.1 Introduction  939
23.2 \texttt{const}\_\texttt{cast} Operator  939
23.3 \texttt{mutable} Class Members  941
23.4 \texttt{namespace}  943
23.5 Operator Keywords  946
23.6 Pointers to Class Members (\texttt{.} and \texttt{-}\texttt{*})  948
23.7 Multiple Inheritance  950
23.8 Multiple Inheritance and \texttt{virtual} Base Classes  955
23.9 Wrap-Up  959

List of Chapters on the Web  965

A Operator Precedence and Associativity  967

B ASCII Character Set  969

C Fundamental Types  970

D Number Systems  972
D.1 Introduction  973
D.2 Abbreviating Binary Numbers as Octal and Hexadecimal Numbers  976
D.3 Converting Octal and Hexadecimal Numbers to Binary Numbers  977
D.4 Converting from Binary, Octal or Hexadecimal to Decimal  977
D.5 Converting from Decimal to Binary, Octal or Hexadecimal  978
D.6 Negative Binary Numbers: Two’s Complement Notation  980
Contents

E Preprocessor  985
E.1 Introduction  986
E.2 \#include Preprocessing Directive  986
E.3 \#define Preprocessing Directive: Symbolic Constants  987
E.4 \#define Preprocessing Directive: Macros  987
E.5 Conditional Compilation  989
E.6 \#error and \#pragma Preprocessing Directives  990
E.7 Operators # and ##  991
E.8 Predefined Symbolic Constants  991
E.9 Assertions  992
E.10 Wrap-Up  992

List of Appendices on the Web  997

Index  999

Online Chapters and Appendices
Chapters 24–26 and Appendices F–K are PDF documents posted online at the book’s Companion Website, which is accessible from www.pearsonhighered.com/deitel.

24 C++11 Additional Features  24-1

25 ATM Case Study, Part 1: Object-Oriented Design with the UML  25-1
25.1 Introduction  25-2
25.2 Introduction to Object-Oriented Analysis and Design  25-2
25.3 Examining the ATM Requirements Document  25-3
25.4 Identifying the Classes in the ATM Requirements Document  25-10
25.5 Identifying Class Attributes  25-17
25.6 Identifying Objects’ States and Activities  25-21
25.7 Identifying Class Operations  25-25
25.8 Indicating Collaboration Among Objects  25-32
25.9 Wrap-Up  25-39

26 ATM Case Study, Part 2: Implementing an Object-Oriented Design  26-1
26.1 Introduction  26-2
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>26.2 Starting to Program the Classes of the ATM System</td>
<td>26-2</td>
</tr>
<tr>
<td>26.3 Incorporating Inheritance into the ATM System</td>
<td>26-8</td>
</tr>
<tr>
<td>26.4 ATM Case Study Implementation</td>
<td>26-15</td>
</tr>
<tr>
<td>26.4.1 Class ATM</td>
<td>26-16</td>
</tr>
<tr>
<td>26.4.2 Class Screen</td>
<td>26-23</td>
</tr>
<tr>
<td>26.4.3 Class Keypad</td>
<td>26-25</td>
</tr>
<tr>
<td>26.4.4 Class CashDispenser</td>
<td>26-26</td>
</tr>
<tr>
<td>26.4.5 Class DepositSlot</td>
<td>26-28</td>
</tr>
<tr>
<td>26.4.6 Class Account</td>
<td>26-29</td>
</tr>
<tr>
<td>26.4.7 Class BankDatabase</td>
<td>26-31</td>
</tr>
<tr>
<td>26.4.8 Class Transaction</td>
<td>26-35</td>
</tr>
<tr>
<td>26.4.9 Class BalanceInquiry</td>
<td>26-37</td>
</tr>
<tr>
<td>26.4.10 Class Withdrawal</td>
<td>26-39</td>
</tr>
<tr>
<td>26.4.11 Class Deposit</td>
<td>26-44</td>
</tr>
<tr>
<td>26.4.12 Test Program ATMCaseStudy.cpp</td>
<td>26-47</td>
</tr>
<tr>
<td>26.5 Wrap-Up</td>
<td>26-47</td>
</tr>
</tbody>
</table>

### C Legacy Code Topics

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>F.1 Introduction</td>
<td>F-2</td>
</tr>
<tr>
<td>F.2 Redirecting Input/Output on UNIX/Linux/Mac OS X and Windows Systems</td>
<td>F-2</td>
</tr>
<tr>
<td>F.3 Variable-Length Argument Lists</td>
<td>F-3</td>
</tr>
<tr>
<td>F.4 Using Command-Line Arguments</td>
<td>F-5</td>
</tr>
<tr>
<td>F.5 Notes on Compiling Multiple-Source-File Programs</td>
<td>F-7</td>
</tr>
<tr>
<td>F.6 Program Termination with exit and atexit</td>
<td>F-9</td>
</tr>
<tr>
<td>F.7 Type Qualifier volatile</td>
<td>F-10</td>
</tr>
<tr>
<td>F.8 Suffixes for Integer and Floating-Point Constants</td>
<td>F-10</td>
</tr>
<tr>
<td>F.9 Signal Handling</td>
<td>F-11</td>
</tr>
<tr>
<td>F.10 Dynamic Memory Allocation with calloc and realloc</td>
<td>F-13</td>
</tr>
<tr>
<td>F.11 Unconditional Branch: goto</td>
<td>F-14</td>
</tr>
<tr>
<td>F.12 Unions</td>
<td>F-15</td>
</tr>
<tr>
<td>F.13 Linkage Specifications</td>
<td>F-18</td>
</tr>
<tr>
<td>F.14 Wrap-Up</td>
<td>F-19</td>
</tr>
</tbody>
</table>

### UML 2: Additional Diagram Types

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>G.1 Introduction</td>
<td>G-1</td>
</tr>
<tr>
<td>G.2 Additional Diagram Types</td>
<td>G-2</td>
</tr>
</tbody>
</table>

### Using the Visual Studio Debugger

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>H. Introduction</td>
<td>H-1</td>
</tr>
</tbody>
</table>
Contents

H.1 Introduction H-2
H.2 Breakpoints and the Continue Command H-2
H.3 Locals and Watch Windows H-8
H.4 Controlling Execution Using the Step Into, Step Over, Step Out and Continue Commands H-11
H.5 Autos Window H-13
H.6 Wrap-Up H-14

I Using the GNU C++ Debugger I-1
I.1 Introduction I-2
I.2 Breakpoints and the run, stop, continue and print Commands I-2
I.3 print and set Commands I-8
I.4 Controlling Execution Using the step, finish and next Commands I-10
I.5 watch Command I-13
I.6 Wrap-Up I-15

J Using the Xcode Debugger J-1

K Test Driving a C++ Program on Mac OS X K-1

[Note: The test drives for Windows and Linux are in Chapter 1.]