

Contents

Preface xxix

Part I

The Foundation of C++: The C Subset

	1	An Overview of C 3
		The Origins of C 4
		C Is a Middle-Level Language 4
		C Is a Structured Language 6
		C Is a Programmer's Language 8
		The Form of a C Program 9
		The Library and Linking 10
		Separate Compilation 12
		Understanding the .C and .CPP File Extensions 12
	2	Expressions 13
		The Five Basic Data Types 14



Modifying the Basic Types	15
Identifier Names	16
Variables	17
Where Variables Are Declared	17
Local Variables	17
Formal Parameters	21
Global Variables	21
Access Modifiers	23
const	23
volatile	24
Storage Class Specifiers	25
extern	25
static Variables	27
register Variables	29
Variable Initializations	31
Constants	31
Hexadecimal and Octal Constants	32
String Constants	33
Backslash Character Constants	33
Operators	34
The Assignment Operator	34
Type Conversion in Assignments	35
Multiple Assignments	36
Arithmetic Operators	37
Increment and Decrement	37
Relational and Logical Operators	39
Bitwise Operators	42
The ? Operator	47
The & and * Pointer Operators	47
The Compile-Time Operator sizeof	49
The Comma Operator	50
The Dot (.) and Arrow (->) Operators	51
The [] and () Operators	51
Precedence Summary	52
Expressions	53
Order of Evaluation	53
Type Conversion in Expressions	53
Casts	54
Spacing and Parentheses	55
Shorthand Assignments	56
  3 Statements	 57
True and False in C and C++	58

Selection Statements	59
if	59
Nested ifs	60
The if-else-if Ladder	62
The ? Alternative	63
The Conditional Expression	66
switch	67
Nested switch Statements	70
Iteration Statements	70
The for Loop	70
for Loop Variations	72
The Infinite Loop	76
for Loops with No Bodies	77
The while Loop	77
The do-while Loop	79
Declaring Variables within Selection and Iteration Statements ...	81
Jump Statements	82
The return Statement	82
The goto Statement	83
The break Statement	83
The exit() Function	85
The continue Statement	86
Expression Statements	88
Block Statements	88
 4 Arrays and Null-Terminated Strings	89
Single-Dimension Arrays	90
Generating a Pointer to an Array	92
Passing Single-Dimension Arrays to Functions	92
Null-Terminated Strings	94
Two-Dimensional Arrays	96
Arrays of Strings	100
Multidimensional Arrays	101
Indexing Pointers	102
Array Initialization	105
Unsize Array Initializations	106
A Tic-Tac-Toe Example	108
 5 Pointers	113
What Are Pointers?	114
Pointer Variables	115
The Pointer Operators	115
Pointer Expressions	116

Pointer Assignments	117
Pointer Arithmetic	117
Pointer Comparisons	119
Pointers and Arrays	120
Arrays of Pointers	122
Multiple Indirection	123
Initializing Pointers	124
Pointers to Functions	126
C's Dynamic Allocation Functions	129
Problems with Pointers	131



6 Functions	137
The General Form of a Function	138
Scope Rules of Functions	138
Function Arguments	139
Call by Value, Call by Reference	139
Creating a Call by Reference	140
Calling Functions with Arrays	142
argc and argv—Arguments to main()	144
The return Statement	147
Returning from a Function	147
Returning Values	149
Returning Pointers	151
Functions of Type void	152
What Does main() Return?	153
Recursion	153
Function Prototypes	155
Standard Library Function Prototypes	157
Declaring Variable-Length Parameter Lists	158
Old-Style Versus Modern Function Parameter Declarations	158
Implementation Issues	159
Parameters and General-Purpose Functions	159
Efficiency	159



7 Structures, Unions, Enumerations, and User-Defined Types	161
Structures	162
Accessing Structure Members	165
Structure Assignments	165
Arrays of Structures	166
Passing Structures to Functions	166
Passing Structure Members to Functions	167
Passing Entire Structures to Functions	167

Structure Pointers	169
Declaring a Structure Pointer	170
Using Structure Pointers	170
Arrays and Structures Within Structures	173
Bit-Fields	174
Unions	176
Enumerations	180
Using sizeof to Ensure Portability	183
typedef	184

8 C-Style Console I/O 187

An Important Application Note	188
Reading and Writing Characters	189
A Problem with getchar()	190
Alternatives to getchar()	190
Reading and Writing Strings	192
Formatted Console I/O	195
printf()	195
Printing Characters	196
Printing Numbers	196
Displaying an Address	198
The %n Specifier	198
Format Modifiers	199
The Minimum Field Width Specifier	199
The Precision Specifier	200
Justifying Output	201
Handling Other Data Types	202
The * and # Modifiers	202
scanf()	203
Format Specifiers	203
Inputting Numbers	203
Inputting Unsigned Integers	205
Reading Individual Characters Using scanf()	205
Reading Strings	205
Inputting an Address	206
The %n Specifier	206
Using a Scanset	206
Discarding Unwanted White Space	207
Non-White-Space Characters in the Control String	208
You Must Pass scanf() Addresses	208
Format Modifiers	208
Suppressing Input	209

9	File I/O	211
	C Versus C++ File I/O	212
	Streams and Files	212
	Streams	212
	Text Streams	213
	Binary Streams	213
	Files	213
	File System Basics	214
	The File Pointer	215
	Opening a File	215
	Closing a File	217
	Writing a Character	218
	Reading a Character	218
	Using <code>fopen()</code> , <code>getc()</code> , <code>putc()</code> , and <code>fclose()</code>	218
	Using <code>feof()</code>	220
	Working with Strings: <code>fputs()</code> and <code>fgets()</code>	222
	<code>rewind()</code>	223
	<code>ferror()</code>	224
	Erasing Files	226
	Flushing a Stream	227
	<code>fread()</code> and <code>fwrite()</code>	227
	Using <code>fread()</code> and <code>fwrite()</code>	228
	<code>fseek()</code> and Random-Access I/O	229
	<code>fprintf()</code> and <code>fscanf()</code>	231
	The Standard Streams	232
	The Console I/O Connection	234
	Using <code>freopen()</code> to Redirect the Standard Streams	235
10	The Preprocessor and Comments	237
	The Preprocessor	238
	<code>#define</code>	238
	Defining Function-like Macros	240
	<code>#error</code>	241
	<code>#include</code>	242
	Conditional Compilation Directives	242
	<code>#if</code> , <code>#else</code> , <code>#elif</code> , and <code>#endif</code>	243
	<code>#ifdef</code> and <code>#ifndef</code>	245
	<code>#undef</code>	246
	Using <code>defined</code>	247
	<code>#line</code>	248
	<code>#pragma</code>	248
	The <code>#</code> and <code>##</code> Preprocessor Operators	248
	Predefined Macro Names	250

C-Style Comments	250
------------------------	-----

Part II

C++

11	An Overview of C++	255
	The Origins of C++	256
	What Is Object-Oriented Programming?	257
	Encapsulation	258
	Polymorphism	258
	Inheritance	259
	Some C++ Fundamentals	259
	A Sample C++ Program	260
	A Closer Look at the I/O Operators	263
	Declaring Local Variables	264
	No Default to int	265
	The bool Data Type	266
	Old-Style vs. Modern C++	266
	The New C++ Headers	268
	Namespaces	269
	Working with an Old Compiler	270
	Introducing C++ Classes	270
	Function Overloading	274
	Operator Overloading	278
	Inheritance	278
	Constructors and Destructors	283
	The C++ Keywords	287
	The General Form of a C++ Program	288
12	Classes and Objects	289
	Classes	290
	Structures and Classes Are Related	293
	Unions and Classes Are Related	295
	Anonymous Unions	297
	Friend Functions	298
	Friend Classes	302
	Inline Functions	303
	Defining Inline Functions Within a Class	306
	Parameterized Constructors	307
	Constructors with One Parameter: A Special Case	309
	Static Class Members	310
	Static Data Members	310

Static Member Functions	315
When Constructors and Destructors Are Executed	317
The Scope Resolution Operator	319
Nested Classes	320
Local Classes	320
Passing Objects to Functions	321
Returning Objects	323
Object Assignment	324

13 Arrays, Pointers, References, and the Dynamic Allocation

Operators	327
Arrays of Objects	328
Creating Initialized vs. Uninitialized Arrays	330
Pointers to Objects	331
Type Checking C++ Pointers	333
The this Pointer	334
Pointers to Derived Types	336
Pointers to Class Members	339
References	341
Reference Parameters	341
Passing References to Objects	345
Returning References	346
Independent References	347
References to Derived Types	348
Restrictions to References	349
A Matter of Style	349
C++'s Dynamic Allocation Operators	349
Initializing Allocated Memory	351
Allocating Arrays	352
Allocating Objects	353
The nothrow Alternative	358
The Placement Forms of new and delete	359

14 Function Overloading, Copy Constructors, and Default

Arguments	361
Function Overloading	362
Overloading Constructor Functions	364
Overloading a Constructor to Gain Flexibility	364
Allowing Both Initialized and Uninitialized Objects	366
Copy Constructors	368
Finding the Address of an Overloaded Function	372
The overload Anachronism	373

	Default Function Arguments	374
	Default Arguments vs. Overloading	378
	Using Default Arguments Correctly	380
	Function Overloading and Ambiguity	380
15	Operator Overloading	385
	Creating a Member Operator Function	386
	Creating Prefix and Postfix Forms of the	
	Increment and Decrement Operators	391
	Overloading the Shorthand Operators	392
	Operator Overloading Restrictions	392
	Operator Overloading Using a Friend Function	393
	Using a Friend to Overload ++ or --	395
	Friend Operator Functions Add Flexibility	398
	Overloading new and delete	400
	Overloading new and delete for Arrays	405
	Overloading the nothrow Version of new and delete	408
	Overloading Some Special Operators	409
	Overloading []	409
	Overloading ()	413
	Overloading ->	415
	Overloading the Comma Operator	416
16	Inheritance	419
	Base-Class Access Control	420
	Inheritance and protected Members	422
	Protected Base-Class Inheritance	426
	Inheriting Multiple Base Classes	427
	Constructors, Destructors, and Inheritance	428
	When Constructor and Destructor	
	Functions Are Executed	428
	Passing Parameters to Base-Class Constructors	432
	Granting Access	436
	Virtual Base Classes	439
17	Virtual Functions and Polymorphism	445
	Virtual Functions	446
	Calling a Virtual Function Through a Base Class	
	Reference	449
	The Virtual Attribute Is Inherited	450
	Virtual Functions Are Hierarchical	452
	Pure Virtual Functions	455
	Abstract Classes	457

Using Virtual Functions	457
Early vs. Late Binding	460
18 Templates	461
Generic Functions	462
A Function with Two Generic Types	465
Explicitly Overloading a Generic Function	465
Overloading a Function Template	468
Using Standard Parameters with Template Functions ...	468
Generic Function Restrictions	469
Applying Generic Functions	470
A Generic Sort	471
Compacting an Array	472
Generic Classes	474
An Example with Two Generic Data Types	478
Applying Template Classes: A Generic Array Class	479
Using Non-Type Arguments with Generic Classes	481
Using Default Arguments with Template Classes	483
Explicit Class Specializations	485
The typename and export Keywords	486
The Power of Templates	487
19 Exception Handling	489
Exception Handling Fundamentals	490
Catching Class Types	496
Using Multiple catch Statements	497
Handling Derived-Class Exceptions	499
Exception Handling Options	500
Catching All Exceptions	500
Restricting Exceptions	502
Rethrowing an Exception	504
Understanding terminate() and unexpected()	505
Setting the Terminate and Unexpected Handlers	506
The uncaught_exception() Function	507
The exception and bad_exception Classes	508
Applying Exception Handling	508
20 The C++ I/O System Basics	511
Old vs. Modern C++ I/O	512
C++ Streams	513
The C++ Stream Classes	513
C++'s Predefined Streams	514
Formatted I/O	515

	Formatting Using the ios Members	515
	Setting the Format Flags	516
	Clearing Format Flags	517
	An Overloaded Form of setf()	518
	Examining the Formatting Flags	520
	Setting All Flags	521
	Using width(), precision(), and fill()	522
	Using Manipulators to Format I/O	524
	Overloading << and >>	528
	Creating Your Own Inserters	528
	Creating Your Own Extractors	534
	Creating Your Own Manipulator Functions	537
	21 C++ File I/O	541
	<fstream> and the File Classes	542
	Opening and Closing a File	542
	Reading and Writing Text Files	545
	Unformatted and Binary I/O	547
	Characters vs. Bytes	547
	put() and get()	548
	read() and write()	550
	More get() Functions	553
	getline()	553
	Detecting EOF	555
	The ignore() Function	557
	peek() and putback()	558
	flush()	558
	Random Access	559
	Obtaining the Current File Position	563
	I/O Status	563
	Customized I/O and Files	565
	22 Run-Time Type ID and the Casting Operators	569
	Run-Time Type Identification (RTTI)	570
	A Simple Application of Run-Time Type ID	576
	typeid Can Be Applied to Template Classes	578
	The Casting Operators	580
	dynamic_cast	580
	Replacing typeid with dynamic_cast	584
	Using dynamic_cast with Template Classes	586
	const_cast	588
	static_cast	590
	reinterpret_cast	590

23	Namespaces, Conversion Functions, and Other Advanced Topics	593
	Namespaces	594
	Namespace Fundamentals	594
	using	598
	Unnamed Namespaces	600
	Some Namespace Options	601
	The std Namespace	603
	Creating Conversion Functions	605
	const Member Functions and mutable	609
	Volatile Member Functions	611
	Explicit Constructors	612
	Using the asm Keyword	613
	Linkage Specification	614
	Array-Based I/O	615
	The Array-Based Classes	616
	Creating an Array-Based Output Stream	616
	Using an Array as Input	618
	Input/Output Array-Based Streams	620
	Using Dynamic Arrays	621
	Using Binary I/O with Array-Based Streams	622
	Summarizing the Differences	
	Between C and C++	623
24	Introducing the Standard Template Library	625
	An Overview of the STL	626
	Containers	626
	Algorithms	627
	Iterators	627
	Other STL Elements	628
	The Container Classes	629
	General Theory of Operation	630
	Vectors	631
	Accessing a Vector Through an Iterator	635
	Inserting and Deleting Elements in a Vector	637
	Storing Class Objects in a Vector	639
	Lists	641
	Understanding end()	645
	push_front() vs push_back()	647
	Sort a List	648
	Merging One List with Another	649
	Storing Class Objects in a List	651
	Maps	654

Storing Class Objects In a Map	658
Algorithms	660
Counting	660
Removing and Replacing Elements	666
Reversing a Sequence	668
Transforming a Sequence	669
Using Function Objects	671
Unary and Binary Function Objects	671
Using the Built-in Function Objects	671
Creating a Function Object	674
Using Binders	676
The string Class	679
Some string Member Functions	683
Strings Are Containers	689
Putting Strings into Other Containers	690
Final Thoughts on the STL	691

Part III

The Standard Function Library

25 The C-Based I/O Functions	695
clearerr	696
fclose	697
feof	697
ferror	697
fflush	698
fgetc	698
fgetpos	698
fgets	699
fopen	699
fprintf	701
fputc	701
fputs	702
fread	702
freopen	702
fscanf	703
fseek	703
fsetpos	704
ftell	704
fwrite	705
getc	705
getchar	706
gets	706

perror	706
printf	707
putc	710
putchar	710
puts	710
remove	711
rename	711
rewind	711
scanf	711
setbuf	715
setvbuf	715
sprintf	716
sscanf	716
tmpfile	716
tmpnam	717
ungetc	717
vprintf, vfprintf, and vsprintf	718

26 The String and Character Functions 719

isalnum	720
isalpha	720
iscntrl	721
isdigit	721
isgraph	721
islower	721
isprint	722
ispunct	722
isspace	722
isupper	723
isxdigit	723
memchr	723
memcmp	723
memcpy	724
memmove	724
memset	725
strcat	725
strchr	725
strcmp	726
strcoll	726
strcpy	727
strcspn	727
strerror	727
strlen	727

strncat	728
strncmp	728
strncpy	729
strpbrk	729
strchr	729
strspn	730
strstr	730
strtok	730
strxfrm	731
tolower	731
toupper	731
27 The Mathematical Functions	733
acos	734
asin	734
atan	735
atan2	735
ceil	735
cos	736
cosh	736
exp	736
fabs	737
floor	737
fmod	737
frexp	737
ldexp	738
log	738
log10	738
modf	739
pow	739
sin	739
sinh	740
sqrt	740
tan	740
tanh	741
28 Time, Date, and Localization Functions	743
asctime	744
clock	745
ctime	745
difftime	746
gmtime	746
localeconv	746

	localtime	748
	mktime	748
	setlocale	748
	strftime	749
	time	750
	29 The Dynamic Allocation Functions	753
	calloc	754
	free	754
	malloc	755
	realloc	755
	30 Utility Functions	757
	abort	758
	abs	758
	assert	759
	atexit	759
	atof	759
	atoi	760
	atol	760
	bsearch	760
	div	761
	exit	762
	getenv	762
	labs	762
	ldiv	763
	longjmp	763
	mblen	763
	mbstowcs	764
	mbtowc	764
	qsort	764
	raise	765
	rand	766
	setjmp	766
	signal	766
	srand	767
	strtod	767
	strtol	768
	strtoul	768
	system	769
	va_arg, va_start, and va_end	769
	wcstombs	770
	wctomb	770

31	The Wide-Character Functions	771
	The Wide-Character Classification Functions	772
	The Wide-Character I/O Functions	775
	The Wide-Character String Functions	775
	Wide-Character String Conversion Functions	775
	Wide-Character Array Functions	778
	Multibyte/Wide-Character Conversion Functions	779

Part IV

The Standard C++ Class Library

32	The Standard C++ I/O Classes	783
	The I/O Classes	784
	The I/O Headers	786
	The Format Flags and I/O Manipulators	787
	Several Data Types	789
	The streamsize and streamoff Types	789
	The streampos and wstreampos Types	789
	The pos_type and off_type Types	789
	The openmode Type	789
	The iostate Type	790
	The seekdir type	790
	The failure Class	790
	Overload << and >> Operators	790
	The General-Purpose I/O Functions	791
	bad	791
	clear	791
	eof	791
	exceptions	792
	fail	792
	fill	792
	flags	793
	flush	793
	fstream, ifstream, and ofstream	793
	gcount	794
	get	794
	getline	795
	good	796
	ignore	796
	open	796
	peek	797
	precision	798

	put	798
	putback	798
	rdstate	798
	read	799
	readsome	799
	seekg and seekp	800
	setf	801
	setstate	801
	str	802
	stringstream, istream, ostream	802
	sync_with_stdio	803
	tellg and tellp	804
	unsetf	804
	width	804
	write	805
	33 The STL Container Classes	807
	The Container Classes	808
	bitset	810
	deque	812
	list	815
	map	818
	multimap	820
	multiset	823
	queue	825
	priority_queue	826
	set	827
	stack	829
	vector	830
	34 The STL Algorithms	835
	adjacent_find	836
	binary_search	836
	copy	837
	copy_backward	837
	count	837
	count_if	838
	equal	838
	equal_range	838
	fill and fill_n	839
	find	839
	find_end	839
	find_first_of	839

find_if	840
for_each	840
generate and generate_n	840
includes	841
inplace_merge	841
iter_swap	841
lexicographical_compare	842
lower_bound	842
make_heap	842
max	843
max_element	843
merge	843
min	844
min_element	844
mismatch	844
next_permutation	845
nth_element	845
partial_sort	845
partial_sort_copy	846
partition	846
pop_heap	846
prev_permutation	847
push_heap	847
random_shuffle	847
remove, remove_if, remove_copy, and remove_copy_if ..	848
replace, replace_copy, replace_if, and replace_copy_if ...	848
reverse and reverse_copy	849
rotate and rotate_copy	849
search	850
search_n	850
set_difference	850
set_intersection	851
set_symmetric_difference	851
set_union	852
sort	852
sort_heap	852
stable_partition	853
stable_sort	853
swap	853
swap_ranges	854
transform	854
unique and unique_copy	854
upper_bound	855

35	STL Iterators, Allocators, and Function Objects	857
	Iterators	858
	The Basic Iterator Types	858
	The Low-Level Iterator Classes	859
	The Predefined Iterators	860
	Two Iterator Functions	868
	Function Objects	868
	Function Objects	869
	Binders	870
	Negators	871
	Adaptors	872
	Allocators	875
36	The String Class	877
	The <code>basic_string</code> Class	878
	The <code>char_traits</code> Class	890
37	The Numeric Classes	893
	The <code>complex</code> Class	894
	The <code>valarray</code> Class	898
	The <code>slice</code> and <code>gslice</code> Classes	913
	The Helper Classes	916
	The Numeric Algorithms	916
	<code>accumulate</code>	916
	<code>adjacent_difference</code>	917
	<code>inner_product</code>	918
	<code>partial_sum</code>	919
38	Exception Handling and Miscellaneous Classes	921
	Exceptions	922
	<code><exception></code>	922
	<code><stdexcept></code>	923
	<code>auto_ptr</code>	924
	The <code>pair</code> Class	926
	Localization	927
	Other Classes of Interest	927

Part V

Applying C++

39	Integrating New Classes: A Custom String Class	931
	The <code>StrType</code> Class	932

	The Constructor and Destructor Functions	934
	I/O on Strings	935
	The Assignment Functions	937
	Concatenation	938
	Substring Subtraction	941
	The Relational Operators	943
	Miscellaneous String Functions	944
	The Entire StrType Class	945
	Using the StrType Class	954
	Creating and Integrating New Types in General	957
	A Challenge	957
	40 An Object-Oriented Expression Parser	959
	Expressions	960
	Parsing Expressions: The Problem	961
	Parsing an Expression	962
	The Parser Class	964
	Dissecting an Expression	965
	A Simple Expression Parser	967
	Understanding the Parser	973
	Adding Variables to the Parser	974
	Syntax Checking in a Recursive-Descent Parser	984
	Building a Generic Parser	985
	Some Things to Try	993
	Index	995