Contents

Preface xxv
Acknowledgments xxix

Part I

The Foundation of C++: The C Subset

1 An Overview of C 3
The Origins of the C Language 4
A Middle-Level Language 4
A Structured Language 6
A Programmer's Language 7
Compilers Versus Interpreters 9
The Form of a C Program 9
The Library and Linking 10
Separate Compilation 11
A C Program's Memory Map 12
A Review of Terms 13

2 Variables, Constants, Operators, and Expressions 15
Identifier Names 16
Data Types 16
Type Modifiers 17
Access Modifiers 19

Copyright 2001 The McGraw-Hill Companies, Inc. Click Here for Terms of Use.

Declaration of Variables 19
Local Variables 20
Formal Parameters 22
Global Variables 22
Storage Class Specifiers 24
extern 25
static Variables 26
static Local Variables 27
static Global Variables 28
register Variables 30
Assignment Statements 31
Multiple Assignments 31
Type Conversion in Assignments 31
Variable Initializations 33
Constants 33
Backslash Character Constants 34
Operators 35
Arithmetic Operators 35
Increment and Decrement 37
Relational and Logical Operators 38
Bitwise Operators 40
The ? Operator 44
The & and * Pointer Operators 45
The sizeof Compile-Time Operator 47
The Comma Operator 48
The Dot (.) and Arrow (–>) Operators 48
The [ ] and ( ) Operators 49
Precedence Summary 49
Expressions 50
Type Conversion in Expressions 50
Casts 51
Spacing and Parentheses 53
C Shorthand 53

3 Program Control Statements 55
True and False 56
Selection Statements 56
if 57
Nested ifs 58
The if-else-if Ladder 59
The ? Alternative 60
switch 63
Nested switch Statements 65
Iteration Statements (Loops) 66
The for Loop 66
for Loop Variations 67
The Infinite Loop 70
for Loops with No Bodies 71

The while Loop ... 72
do-while ... 74
Jump Statements ... 75
break ... 75
exit( ) ... 77
continue ... 78
Labels and goto ... 80
Expression Statements ... 81
Block Statements ... 81

4 Functions ... 83
The General Form of a Function ... 84
The return Statement ... 84
Returning from a Function ... 84
Returning Values ... 85
What Does main() Return? ... 87
Understanding the Scope of a Function ... 87
Function Arguments ... 88
Call by Value, Call by Reference ... 88
Creating a Call by Reference ... 89
Calling Functions with Arrays ... 91
argc and argv—Arguments to main( ) ... 95
Function Prototypes ... 101
Standard Library Function Prototypes ... 103
Old-Style Versus Modern Parameter Declarations ... 104
The "Implicit int" Rule ... 105
Declaring Variable Length Parameter Lists ... 106
Returning Pointers ... 106
Recursion ... 108
Pointers to Functions ... 109
Implementation Issues ... 112
Parameters and General-Purpose Functions ... 112
Efficiency ... 113

5 Arrays ... 115
Single-Dimension Arrays ... 116
Generating a Pointer to an Array ... 117
Passing Single-Dimension Arrays to Functions ... 118
Null-Terminated Strings ... 119
Two-Dimensional Arrays ... 121
Arrays of Strings ... 125
Multidimensional Arrays ... 127
Indexing Pointers ... 127
Allocated Arrays ... 129
Array Initialization ... 131
Unsized-Array Initializations ... 133
A Tic-Tac-Toe Example ... 134

6 Pointers 139
Pointers Are Addresses 140
Pointer Variables 141
The Pointer Operators 141
Pointer Expressions 143
Pointer Assignments 143
Pointer Arithmetic 144
Pointer Comparisons 145
Dynamic Allocation and Pointers 147
Understanding const Pointers 149
Pointers and Arrays 150
Pointers to Character Arrays 151
Arrays of Pointers 153
Pointers to Pointers: Multiple Indirection 154
Initializing Pointers 156
Pointers to Functions 157
Problems with Pointers 160

7 Structures, Unions, and User-Defined Types 163
Structures 164
Accessing Structure Members 166
Structure Assignments 167
Arrays of Structures 168
An Inventory Example 168
Passing Structures to Functions 175
Passing Structure Members to Functions 175
Passing Entire Structures to Functions 176
Structure Pointers 177
Declaring a Structure Pointer 177
Using Structure Pointers 177
Arrays and Structures Within Structures 181
Bit-Fields 182
Unions 184
Enumerations 186
An Important Difference Between C and C++ 189
Using sizeof to Ensure Portability 189
typedef 191

8 Input, Output, Streams, and Files 193
C Versus C++ I/O 194
Streams and Files 195
Streams 195
Files 196
Console I/O 197
Reading and Writing Characters 197
Reading and Writing Strings: gets( ) and puts( ) 200
Formatted Console I/O 201
printf( ) 201
scanf( ) 209

The C File System 216
The File Pointer 217
Opening a File 218
Writing a Character 220
Reading a Character 220
Closing a File 221
Using fopen( ), getc( ), putc( ), and fclose( ) 221
Using feof( ) 223
Working with Strings: fgets( ) and fputs( ) 224
fread( ) and fwrite( ) 225
fseek( ) and Random Access I/O 227
fprintf( ) and fscanf( ) 230
Erasing Files 231
ferror( ) and rewind( ) 231
The Console Connection 232

9 The Preprocessor and Comments 235
#define 236
#error 240
#include 240
Conditional Compilation Directives 241
#if, #else, #elif, and #endif 242
#ifdef and #ifndef 244
#undef 245
Using defined 246
#line 247
#pragma 247
..... 251
#import 251
The # and ## Preprocessor Operators 252
Predefined Macro Names 253
Comments 255

Part II

The C++ Builder Function Library

10 Linking, Libraries, and Headers 259
The Linker 260
Library Files Versus Object Files 261
The Standard Library Versus C++ Builder Extensions 262
Headers 262
Macros in Headers 264

11 I/O Functions 265
int access(const char *filename, int mode) 266
int chmod(const char *filename, int mode) 267
int chsize(int handle, long size) 268

void clearerr(FILE *stream) 269
int close(int fd)
int _rtl_close(int fd) 270
int _creat(const char *filename, int pmode)
int _rtl_creat(const char *filename, int attrib)
int creatnew(const char *filename, int attrib)
int creattemp(char *filename, int attrib) 271
int dup(int handle)
int dup2(int old_handle, int new_handle) 273
int eof(int fd) 274
int fclose(FILE *stream)
int _fcloseall(void) 275
FILE *fdopen(int handle, char *mode) 276
int feof(FILE *stream) 276
int ferror(FILE *stream) 277
int fflush(FILE *stream) 278
int fgetc(FILE *stream) 278
int fgetchar(void) 279
int *fgetpos(FILE *stream, fpos_t *pos) 279
char *fgets(char *str, int num, FILE *stream) 281
long filelength(int handle) 282
int fileno(FILE *stream) 282
int _flushall(void) 283
FILE *fopen(const char *fname, const char *mode) 283
int fprintf(FILE *stream, const char *format, arg-list) 285
int fputc(int ch, FILE *stream) 286
int fputchar(int ch) 287
int fputs(const char *str, FILE *stream) 288
size_t fread(void *buf, size_t size, size_t count,
FILE *stream) 288
FILE *freopen(const char *fname, const char *mode,
FILE *stream) 289
int fscanf(FILE *stream, const char *format, arg-list) 290
int fseek(FILE *stream, long offset, int origin) 291
int fsetpos(FILE *stream, const fpos_t *pos) 292
FILE *_fsopen(const char *fname, const char *mode,
int shflg) 294
int fstat(int handle, struct stat *statbuf) 295
long ftell(FILE *stream) 296
size_t fwrite(const void *buf, size_t size, size_t count,
FILE *stream) 296
int getc(FILE *stream) 297
int getch(void)
int getche(void) 298
int getchar(void) 299
char *gets(char *str) 300
int getw(FILE *stream) 301
int isatty(int handle) 302
int lock(int handle, long offset, long length) 302

int locking(int handle, int mode, long length) 303
long lseek(int handle, long offset, int origin) 304
int open(const char *filename,
int access, unsigned mode)
int _rtl_open(const char *filename, int access) 306
void perror(const char *str) 308
int printf (const char *format, arg-list) 309
int putc(int ch, FILE *stream) 312
int putch(int ch) 313
int putchar(int ch) 313
int puts(const char *str) 314
int putw(int i, FILE *stream) 314
int read(int fd, void *buf, unsigned count)
int _rtl_read(int fd, void *buf, unsigned count) 315
int remove(const char *fname) 316
int rename(const char *oldfname,
const char *newfname) 317
void rewind(FILE *stream) 318
int _rtl_chmod (const char *filename,
int get_set, int attrib) 319
int scanf(const char *format, arg-list) 319
void setbuf(FILE *stream, char *buf) 324
int setmode(int handle, int mode) 324
int setvbuf(FILE *stream, char *buf,
int mode, size_t size) 325
int sopen(const char *filename, int access,
int shflag, int mode) 325
int sprintf(char *buf, const char *format, arg-list) 328
int sscanf(char *buf, const char *format, arg-list) 328
int stat(char *filename, struct stat *statbuf) 329
long tell(int fd) 330
FILE *tmpfile(void) 330
char *tmpnam(char *name) 331
int ungetc(int ch, FILE *stream) 332
int ungetch(int ch) 333
int unlink(const char *fname) 334
int unlock(int handle, long offset, long length) 334
int vprintf(const char *format, va_list arg_ptr)
int vfprintf(FILE *stream, const char *format,
va_list arg_ptr)
int vsprintf(char *buf, const char *format, va_list arg_ptr) 335
int vscanf(const char *format, va_list arg_ptr)
int vfscanf(FILE *stream, const char *format,
va_list arg_ptr)
int vsscanf(const char *buf,
const char *format, va_list arg_ptr) 336
int write(int handle, void *buf, int count)
int _rtl_write(int handle, void *buf, int count) 338

12 String, Memory, and Character Functions 341
int isalnum(int ch) 342
int isalpha(int ch) 343
int isascii(int ch) 344
int iscntrl(int ch) 344
int isdigit(int ch) 345
int isgraph(int ch) 346
int islower(int ch) 347
int isprint(int ch) 348
int ispunct(int ch) 348
int isspace(int ch) 349
int isupper(ch) 350
int isxdigit(int ch) 351
void *memccpy(void *dest, const void *source,
int ch, size_t count) 351
void *memchr(const void *buffer, int ch, size_t count) 352
int memcmp(const void *buf1,
const void *buf2, size_t count) 353
int memicmp(const void *buf1,
const void *buf2, size_t count) 353
void *memcpy(void *dest, const void *source,
size_t count) 354
void *memmove(void *dest, const void *source,
size_t count) 355
void *memset(void *buf, int ch, size_t count) 356
void movmem(const void *source, void *dest,
unsigned count) 356
void setmem(void *buf, unsigned count, char ch) 357
char *stpcpy(char *str1, const char *str2) 357
char *strcat(char *str1, const char *str2) 358
char *strchr(const char *str, int ch) 359
int strcmp(const char *str1, const char *str2) 359
int strcoll(const char *str1, const char *str2) 360
char *strcpy(char *str1, const char *str2) 360
size_t strcspn(const char *str1, const char *str2) 361
char *strdup(const char *str) 362
char *_strerror(const char *str) 362
char *strerror(int num) 363
int stricmp(const char *str1, const char *str2)
int strcmpi(const char *str1, const char *str2) 363
size_t strlen(const char *str) 364
char *strlwr(char *str) 365
char *strncat(char *str1, const char *str2, size_t count) 365
int strncmp(const char *str1, const char *str2, size_t count)
int strnicmp(const char *str1, const char *str2, size_t count) 367
int strncmpi(const char *str1, const char *str2, size_t count) 367
char *strncpy(char *dest, const char *source,
size_t count) 368
char *strnset(char *str, int ch, size_t count) 369

char *strpbrk(const char *str1, const char *str2) 369
char *strrchr(const char *str, int ch) 370
char *strrev(char *str) 371
char *strset(char *str, int ch) 371
size_t strspn(const char *str1, const char *str2) 372
char *strstr(const char *str1, const char *str2) 373
char *strtok(char *str1, const char *str2) 373
char *strupr(char *str) 375
size_t strxfrm(char *dest, const char *source, size_t count) 375
int tolower(int ch)
int _tolower(int ch) 376
int toupper(int ch)
int _toupper(int ch) 376

13 Mathematical Functions 379
double acos(double arg)
long double acosl(long double arg) 380
double asin(double arg)
long double asinl(long double arg) 381
double atan(double arg)
long double atanl(long double arg) 382
double atan2(double y, double x)
long double atan2l(long double y, long double x) 383
double cabs(struct complex znum)
long double cabsl(struct _complexl znum) 383
double ceil(double num) long double ceill
(long double num) 384
double cos(double arg)
long double cosl(long double arg) 385
double cosh(double arg)
long double coshl(long double arg) 386
double exp(double arg)
long double expl(long double arg) 387
double fabs(double num)
long double fabsl(long double num) 387
double floor(double num) long double floorl
(long double num) 388
double fmod(double x, double y) long double fmodl
(long double x, long double y) 388
double frexp(double num, int *exp)
long double frexpl(long double num, int *exp) 389
double hypot(double x, double y)
long double hypotl(long double x, long double y) 390
double ldexp(double num, int exp)
long double ldexpl(long double num, int exp) 390
double log(double num)
long double logl(long double num) 391
double log10(double num) long double log10l
(long double num) 392

int _matherr(struct exception *err) int _matherrl
(struct _exceptionl *err) 392
double modf(double num, double *i)
long double modfl(long double num, long double *i) 394
double poly(double x, int n, double c[])
long double polyl(long double x, int n, long double c[]) 394
double pow(double base, double exp) long double powl
(long double base, long double exp) 395
double pow10(int n)
long double pow10l(int n) 396
double sin(double arg)
long double sinl(long double arg) 397
double sinh(double arg)
long double sinhl(long double arg) 397
double sqrt(double num)
long double sqrtl(long double num) 398
double tan(double arg)
long double tanl(long double arg) 399
double tanh(double arg)
long double tanhl(long double arg) 399

14 Time, Date, and System-Related Functions 401
char *asctime(const struct tm *ptr) 403
clock_t clock(void) 404
char *ctime(const time_t *time) 405
double difftime(time_t time2, time_t time1) 406
void disable(void)
void _disable(void) 407
unsigned _dos_close(int fd) 407
unsigned _dos_creat(const char *fname,
unsigned attr, int *fd) 408
unsigned _dos_creatnew(const char *fname,
unsigned attr, int *fd) 408
void _dos_getdate(struct dosdate_t *d)
void _dos_gettime(struct dostime_t *t) 409
unsigned _dos_getdiskfree(unsigned char drive,
struct diskfree_t *dfptr) 410
void _dos_getdrive(unsigned *drive) 411
unsigned _dos_getfileattr(const char *fname,
unsigned *attrib) 412
unsigned _dos_getftime(int fd, unsigned *fdate,
unsigned *ftime) 413
unsigned _dos_open(const char *fname,
unsigned mode, int *fd) 414
unsigned _dos_read(int fd, void *buf, unsigned count,
unsigned *numread) 416
unsigned _dos_setdate(struct dosdate_t *d)
unsigned _dos_settime(struct dostime_t *t) 417

void _dos_setdrive(unsigned drive, unsigned *num) 418
unsigned _dos_setfileattr(const char *fname,
unsigned attrib) 418
unsigned _dos_setftime(int fd, unsigned fdate,
unsigned ftime) 419
long dostounix(struct date *d, struct time *t) 421
unsigned _dos_write(int fd, void *buf, unsigned count,
unsigned *numwritten) 422
void enable(void)
void _enable(void) 422
void ftime(struct timeb *time) 423
void geninterrupt(int intr) 424
void getdate(struct date *d)
void gettime(struct time *t) 424
void getdfree(unsigned char drive, struct dfree *dfptr) 425
int getftime(int handle, struct ftime *ftptr) 426
struct tm *gmtime(const time_t *time) 427
int kbhit(void) 428
struct tm *localtime(const time_t *time) 428
time_t mktime(struct tm *p) 429
void setdate(struct date *d)
void settime(struct time *t) 430
int setftime(int handle, struct ftime *t) 431
void sleep(unsigned time) 432
int stime(time_t *t) 432
char *_strdate(char *buf)
char *_strtime(char *buf) 433
size_t strftime(char *str, size_t maxsize,
char const *fmt, const struct tm *time) 434
time_t time(time_t *time) 434
void tzset(void) 436
void unixtodos(long utime, struct date *d,
struct time *t) 436

15 Dynamic Allocation 439
void *alloca(size_t size) 440
void *calloc(size_t num, size_t size) 441
void free(void *ptr) 442
int heapcheck(void) 443
int heapcheckfree(unsigned fill) 444
int heapchecknode(void *ptr) 445
int _heapchk(void) 446
int heapfillfree(unsigned fill) 446
int _heapmin(void) 447
int _heapset(unsigned fill) 448
int heapwalk(struct heapinfo *hinfo)
int _rtl_heapwalk(_HEAPINFO *hinfo) 448
void *malloc(size_t size) 450
void *realloc(void *ptr, size_t newsize) 451

16 Directory Functions 453
int chdir(const char *path) 454
int _chdrive(int drivenum) 454
void closedir(DIR *ptr)
DIR *opendir(char *dirname)
struct dirent *readdir(DIR *ptr)
void rewinddir(DIR *ptr) 455
unsigned _dos_findfirst(const char *fname, int attr,
struct find_t *ptr)
unsigned _dos_findnext(struct find_t *ptr) 456
int findfirst(const char *fname, struct ffblk *ptr, int attrib)
int findnext(struct ffblk *ptr) 457
void fnmerge(char *path, const char *drive, const char
*dir, const char *fname, const char *ext)
int fnsplit(const char *path, char *drive, char *dir,
char *fname, char *ext) 459
char *_fullpath(char *fpath, const char *rpath, int len) 461
int getcurdir(int drive, char *dir) 461
char *getcwd(char *dir, int len) 462
char *_getdcwd(int drive, char *path, int len) 463
int getdisk(void) 464
int _getdrive(void) 464
void _makepath(char *pname, const char *drive,
const char *directory, const char *fname,
cont char *extension) 465
int mkdir(const char *path) 466
char *mktemp(char *fname) 467
int rmdir(const char *path) 467
char *searchpath(const char *fname) 468
int setdisk(int drive) 469
void _splitpath(const char *fpath, char *drive, char
*directory char *fname, char *extension) 469

17 Process Control Functions 471
void abort(void) 472
int atexit(void (*func)(void)) 473
unsigned long _beginthread(void (*func)(void *),
unsigned stksize, void *arglist)
unsigned long _beginthreadex(void *secattr,
unsigned stksize, unsigned (*start)(void *),
void *arglist, unsigned createflags,
unsigned *threadID)
unsigned long _beginthreadNT(void (*func)(void *),
unsigned stksize, void *arglist,
void *secattr, unsigned createflags,
unsigned *threadID); 474
void _c_exit(void)
void _cexit(void) 476

void _endthread(void)
void _endthreadex(unsigned threadvalue) 477
int execl(char *fname, char *arg0, . . ., char *argN, NULL)
int execle(char *fname, char *arg0, . . ., char *argN,
NULL, char *envp[ ]) 477
int execlp(char *fname, char *arg0, . . ., char *argN, NULL)
int execlpe(char *fname, char *arg0, . . ., char *argN,
NULL, char *envp[ ]) 477
int execv(char *fname, char *arg[ ])
int execve(char *fname, char *arg[ ], char *envp[ ]) int execvp
(char *fname, char *arg[ ])
int execvpe(char *fname, char *arg[ ], char *envp[ ]) 477
void exit(int status)
void _exit(int status) 479
int getpid(void) 480
int spawnl(int mode, char *fname, char *arg0, . . .,
char *argN, NULL)
int spawnle(int mode, char *fname, char *arg0, . . .,
char *argN, NULL, char *envp[ ])
int spawnlp(int mode, char *fname, char *arg0, . . .,
char *argN, NULL)
int spawnlpe(int mode, char *fname, char *arg0, . . .,
char *argN, NULL, char *envp[ ])
int spawnv(int mode, char *fname, char *arg[ ])
int spawnve(int mode, char *fname, char *arg[ ], char *envp[ ])
int spawnvp(int mode, char *fname, char *arg[ ])
int spawnvpe(int mode, char *fname, char *arg[ ],
char *envp[ ]) 481
int wait(int *status) 484

18 Screen-Based Text Functions 487
char *cgets(char *inpstr) 488
void clreol(void)
void clrscr(void) 489
int cprintf(const char *fmt, . . .) 490
int cputs(const char *str) 491
int cscanf(char *fmt, . . .) 492
void delline(void) 493
int gettext(int left, int top, int right, int bottom, void *buf) 494
void gettextinfo(struct text_info *info) 494
void gotoxy(int x, int y) 495
void highvideo(void) 496
void insline(void) 496
void lowvideo(void) 497
int movetext(int left, int top, int right, int bottom,
int newleft, int newtop) 498
void normvideo(void) 498
int puttext(int left, int top, int right, int bottom, void *buf) 499

void textattr(int attr) 499
void textbackground(int color) 500
void textcolor(int color) 501
void textmode(int mode) 502
int wherex(void)
int wherey(void) 503
void window(int left, int top, int right, int bottom) 504

19 Miscellaneous Functions 505
int abs(int num) 506
void assert(int exp) 507
double atof(const char *str)
long double _atold(const char *str) 508
int atoi(const char *str) 509
long atol(const char *str) 509
void *bsearch(const void *key, const void *base, size_t num,
size_t size, int (*compare)(const void *, const void *)) 510
unsigned int _clear87(void) 512
unsigned int _control87(unsigned fpword,
unsigned fpmask) 512
div_t div(int numerator, int denominator) 513
char *ecvt(double value, int ndigit, int *dec, int *sign) 514
void _ _emit_ _(unsigned char arg, . . .) 514
char *fcvt(double value, int ndigit, int *dec, int *sign) 515
void _fpreset(void) 515
char *gcvt(double value, int ndigit, char *buf) 516
char *getenv(const char *name) 516
char *getpass(const char *str) 517
unsigned getpid(void) 517
char *itoa(int num, char *str, int radix) 518
long labs(long num) 519
ldiv_t ldiv(long numerator, long denominator) 519
void *lfind(const void *key, const void *base, size_t *num,
size_t size, int (*compare)(const void *, const void *)
void *lsearch(const void *key, void *base, size_t *num, size_t size,
int (*compare)(const void *, const void *)) 520
struct lconv *localeconv(void) 522
void longjmp(jmp_buf envbuf, int val) 523
char *ltoa(long num, char *str, int radix)
char *ultoa(unsigned long num, char *str, int radix) 525
unsigned long _lrotl(unsigned long l, int i)
unsigned long _lrotr(unsigned long l, int i) 526
max(x,y)
min(x,y) 526
int mblen(const char *str, size_t size) 527
size_t mbstowcs(wchar_t *out, const char *in, size_t size) 527
int mbtowc(wchar_t *out, const char *in, size_t size) 528
int putenv(const char *evar) 529

void qsort(void *base, size_t num, size_t size,
int (*compare) (const void *, const void *)) 529
int raise(int signal) 531
int rand(void) 532
int random(int num)
void randomize(void) 533
unsigned short _rotl(unsigned short val, int num)
unsigned short _rotr(unsigned short val, int num) 533
void _setcursortype(int type) 534
int setjmp(jmp_buf envbuf) 535
void _searchenv(const char *fname, const char *ename,
char *fpath) 536
char *setlocale(int type, const char *locale) 537
void (*set_new_handler(void (* newhand)( )))( ) 538
void (*signal (int signal, void (*sigfunc) (int func)))(int) 538
void srand(unsigned seed) 539
unsigned int _status87(void) 540
double strtod(const char *start, char **end)
long double _strtold(const char *start, char **end) 540
long strtol(const char *start, char **end, int radix)
unsigned long strtoul(const char *start, char **end,
int radix) 542
void swab(char *source, char *dest, int num) 543
int system(const char *str) 543
int toascii(int ch) 544
unsigned umask(unsigned access) 544
int utime(char *fname, struct utimbuf *t) 545
void va_start(va_list argptr, last_parm)
void va_end(va_list argptr)
type va_arg(va_list argptr, type) 546
size_t wcstombs(char *out, const wchar_t *in, size_t size) 548
int wctomb(char *out, wchar_t in) 548

Part III

C++

20 An Overview of C++ 551
The Origins of C++ 552
What Is Object-Oriented Programming? 553
Encapsulation 554
Polymorphism 554
Inheritance 555
Some C++ Fundamentals 555
C++ Programs Use the .CPP Extension 558
A Closer Look at Headers and Namespaces 559
Modern-Style Headers 559
Namespaces 560

Introducing C++ Clases 560
Function Overloading 565
Operator Overloading 568
Inheritance 568
Constructors and Destructors 572
The C++ Keywords 576
Two New Data Types 577

21 A Closer Look at Classes and Objects 579
Parameterized Constructors 580
Constructors with One Parameter: A Special Case 584
Friend Functions 585
Default Function Arguments 590
Using Default Arguments Correctly 594
Classes and Structures Are Related 594
Unions and Classes Are Related 596
Anonymous Unions 597
Inline Functions 598
Creating Inline Functions Inside a Class 600
Passing Objects to Functions 601
Returning Objects 604
Object Assignment 605
Arrays of Objects 605
Initializing Arrays of Objects 607
Creating Initialized Versus Uninitialized Arrays 609
Pointers to Objects 610

22 Function and Operator Overloading 613
Overloading Constructor Functions 614
Localizing Variables 616
Localizing the Creation of Objects 617
Function Overloading and Ambiguity 619
Finding the Address of an Overloaded Function 622
The this Pointer 623
Operator Overloading 624
Friend Operator Functions 631
References 636
Reference Parameters 636
Passing References to Objects 639
Returning References 640
Independent References 641
Using a Reference to Overload a Unary Operator 643
Overloading [ ] 646
Applying Operator Overloading 650

23 Inheritance, Virtual Functions, and Polymorphism 657
Inheritance and the Access Specifiers 658
Understanding the Access Specifiers 658
Base Class Access Control 660

FREAKFLY

Constructors and Destructors in Derived Classes 663
Multiple Inheritance 667
Passing Parameters to a Base Class 669
Pointers and References to Derived Types 671
References to Derived Types 673
Virtual Functions 674
Why Virtual Functions? 679
Pure Virtual Functions and Abstract Types 684
Early Versus Late Binding 686

24 The C++ I/O Class Library 689
Why C++ Has Its Own I/O System 690
Old Versus Modern C++ I/O 690
C++ Streams 691
The C++ Predefined Streams 691
The C++ Stream Classes 691
Creating Your Own Inserters and Extractors 692
Creating Inserters 693
Overloading Extractors 696
Formatting I/O 698
Formatting Using the ios Member Functions 698
Using Manipulators 702
Creating Your Own Manipulator Functions 705
File I/O 708
Opening and Closing a File 708
Reading and Writing Text Files 710
Unformatted and Binary I/O 712
Using get( ) and put( ) 712
Using read( ) and write( ) 714
Detecting EOF 715
Random Access 716

25 Templates, Exceptions, and RTTI 719
Generic Functions 720
A Function with Two Generic Types 722
Explicitly Overloading a Generic Function 723
Overloading a Function Template 725
Generic Function Restrictions 725
Generic Classes 726
An Example with Two Generic Data Types 730
Exception Handling 731
Exception Handling Fundamentals 731
Catching Class Types 737
Using Multiple catch Statements 738
Handling Derived-Class Exceptions 739
Exception Handling Options 740
Catching All Exceptions 741
Restricting Exceptions 742
Rethrowing an Exception 744

Understanding terminate( ) and unexpected( ) 745
Setting the Terminate and Unexpected Handlers 745
The uncaught_exception( ) Function 747
Applying Exception Handling 747
Run-Time Type Identification (RTTI) 748
Casting Operators 750

26 Miscellaneous C++ Topics 755
Dynamic Allocation Using new and delete 756
Allocating Objects 759
Another Way to Watch for Allocation Failure 762
Overloading new and delete 763
Overloading new and delete for Arrays 768
static Class Members 771
static Data Members 771
static Member Functions 772
Virtual Base Classes 775
const Member Functions and mutable 780
Volatile Member Functions 782
Using the asm Keyword 782
Linkage Specification 783
The .* and ->* Operators 784
Creating Conversion Functions 786
Copy Constructors 788
Granting Access 791
Namespaces 794
Namespace Fundamentals 795
using 798
Unnamed Namespaces 800
Some Namespace Options 801
The std Namespace 803
Explicit Constructors 805
typename and export 806
Differences Between C and C++ 807

27 The Standard Template Library and the string Class 809
An Overview of the STL 810
Containers 810
Algorithms 811
Iterators 811
Other STL Elements 812
The Container Classes 813
General Theory of Operation 814
Vectors 815
Accessing a Vector Through an Iterator 819
Inserting and Deleting Elements in a Vector 821
Storing Class Objects in a Vector 823
Lists 825
Understanding end( ) 829
push_front( ) Versus push_back( ) 831

Sort a List 832
Merging One List with Another 833
Storing Class Objects in a List 835
Maps 837
Storing Class Objects in a Map 841
Algorithms 843
Counting 846
Removing and Replacing Elements 849
Reversing a Sequence 851
Transforming a Sequence 852
Using Function Objects 853
Unary and Binary Function Objects 854
Using the Built-in Function Objects 854
Creating a Function Object 857
Using Binders 859
The string Class 861
Some string Member Functions 866
Strings Are Containers 871
Putting Strings into Other Containers 872
Final Thoughts on the STL 873

Part IV

The C++ Builder Integrated Development Environment

28 The Integrated Development Environment 877
The Four IDE Windows 878
The Menu Window 878
File 880
Edit 882
Search 883
View 885
Project 886
Run 888
Component 889
Tools 890
Help 894
Toolbars 894
Object Inspector Window 895
Form Window 895
Code (Unit) Window 895
Using Speed Menus 896
Using Context-Sensitive Help 896

29 Developing Applications Using the IDE 897
Types of Applications 898
New 898
Project1 900
Forms 902

Dialogs 902
Projects 902
The Component Palette 903
Standard Components 904
Additional Components 904
Win32 Components 905
System Components 907
Dialogs Components 908
Win 3.1 Components 909
Samples Components 909
ActiveX Components 910
Internet Components 911
Servers Components 911
Creating a Console Application 911
Using the IDE to Create a Console Application 912
Compiling the Sample Programs in This Book 916
Using the Command Line Compiler 916
Creating a Simple Windows Application 918
Preliminary Steps 918
Create the Application 920
Building the GUI Form 920
Adding Label and Edit Components 922
Using the ActionList and ImageList Components 924
Building a Basic Menu 931
Creating a Toolbar 933
Building Command Buttons 935
Adding a Help | About Dialog Box 936
Adding Code and Finishing the Application 937

30 Using C++ Builder's Integrated Debugging Environment 941
Preparing Your Programs for Debugging 942
What Is a Source-Level Debugger? 942
Debugger Basics 942
Single-Stepping 943
Breakpoints 945
Setting Unconditional Source Breakpoints 946
Setting Conditional Source Breakpoints 947
Watching Variables 949
Watched-Expression Formats 950
Qualifying a Variable's Name 953
Watching the Stack 954
Evaluating an Expression 955
Pausing a Program 956
Using the CPU Window 956
A Debugging Tip 957

Index 959